# Optimal personalized treatment learning models with insurance applications

Leo Guelman

de Barcelona

PhD in Economics | Leo Guelman

Universitat de Barcelona

PhD in Economics

# Optimal personalized treatment learning models with insurance applications

Leo Guelman

Universitat

PhD in Economics

**Thesis title:**

# Optimal personalized treatment learning models with insurance applications

**PhD student:**
Leo Guelman

**Advisor:**
Montserrat Guillén

**Date:**
October 2014

Universitat de Barcelona

# Contents

**Acknowledgements** **xv**

**Abstract** **xvii**

**1 Introduction** **1**

**2 The personalized treatment learning problem** **5**
2.1 Problem formulation . . . . . 5
2.2 Examples of the personalized treatment learning problem 7
2.2.1 *Example 1*: The red/blue envelope problem . . . . 7
2.2.2 *Example 2*: Evaluating personalized treatment effects from a job training program . . . . . 8
2.2.3 *Example 3*: Identifying breast cancer patients who may benefit from a new treatment . . . . . 10

**3 Existing personalized treatment learning models** **13**
3.1 Introduction . . . . . 13
3.2 Indirect estimation methods . . . . . 13
3.2.1 Difference score method . . . . . 13
3.2.2 Interaction method . . . . . 14
3.2.3 L2-SVM method . . . . . 14
3.3 Direct estimation methods . . . . . 15
3.3.1 Modified covariate method . . . . . 15
3.3.2 Modified outcome method . . . . . 17
3.3.3 Causal $K$-nearest-neighbor (CKNN) . . . . . 19
3.3.4 Matching before randomization . . . . . 20

**4 Uplift random forests** **23**
4.1 Introduction . . . . . 23
4.2 Definition of uplift random forests . . . . . 23
4.3 Input variable importance . . . . . 28
4.4 Performance of uplift random forests . . . . . 30
4.4.1 Number of covariates . . . . . 30
4.4.2 Split criteria comparison . . . . . 32
4.4.3 Uplift random forests and overfitting . . . . . 33

**5 Causal conditional inference forests** **35**
5.1 Introduction . . . . . 35
5.2 Definition of causal conditional inference forests . . . . . 35
5.3 Additional considerations . . . . . 39
5.3.1 Categorical covariates . . . . . 39
5.3.2 Stopping criteria . . . . . 41
5.3.3 Observational studies . . . . . 41
5.3.4 Multiple treatments . . . . . 41

**6 Simulations** **43**

**7 Model assessment and selection for PTL models** **49**
7.1 Introduction . . . . . 49
7.2 The Qini curve and the Qini coefficient . . . . . 49
7.3 Optimal Qini curves . . . . . 52
7.4 Top uplift . . . . . 54
7.5 Resampling methods . . . . . 54
7.5.1 $K$-fold cross-validation . . . . . 55
7.5.2 Monte Carlo cross-validation . . . . . 56
7.5.3 Bootstrap methods . . . . . 56
7.5.4 Selecting final tuning parameters . . . . . 57
7.6 Model calibration . . . . . 60

**8 Empirical applications in insurance marketing** **63**
8.1 Introduction . . . . . 63
8.2 An insurance cross-sell application . . . . . 64
8.2.1 The data . . . . . 64
8.2.2 Building the model . . . . . 65

8.2.3 Results . . . . . 65
8.3 An insurance customer retention case . . . . . 67
8.3.1 The data . . . . . 67
8.3.2 Building the models . . . . . 69
8.3.3 Results . . . . . 70

**9 Personalized treatment learning in observational studies 73**
9.1 Introduction . . . . . 73
9.2 Price elasticity in insurance . . . . . 74
9.3 Price elasticity as a personalized treatment learning problem 77
9.4 The method . . . . . 78
9.4.1 Unconfoundedness and common support . . . . . 78
9.4.2 Propensity score . . . . . 80
9.4.3 Matching: A short review . . . . . 82
9.5 An application to auto insurance price elasticity estimation 85
9.5.1 The data . . . . . 85
9.5.2 Building the model . . . . . 86
9.5.3 Propensity score estimates . . . . . 88
9.5.4 Matching and covariate balance . . . . . 90
9.5.5 Price-elasticity functions . . . . . 94
9.6 Managerial implications: Price optimization . . . . . 97
9.7 Discussion . . . . . 100

**10 Loss cost considerations 103**
10.1 Introduction . . . . . 103
10.2 Pricing in non-life insurance . . . . . 104
10.3 Predictive learning and boosting . . . . . 105
10.4 AdaBoost . . . . . 106
10.5 Additive models and boosting . . . . . 107
10.6 Gradient boosting trees . . . . . 109
10.7 Injecting randomness and regularization . . . . . 111
10.8 Interpretation . . . . . 112
10.9 Application to auto insurance loss cost modeling . . . . . 113
10.9.1 The data . . . . . 113
10.9.2 Building the model . . . . . 116
10.9.3 Results . . . . . 118

10.10 Discussion . . . . . 125

**11 Software: The uplift** R package **127**
11.1 Introduction . . . . . 127
11.2 Creating simulated datasets . . . . . 127
11.3 Exploratory data analysis . . . . . 129
11.4 Fitting personalized treatment learning models . . . . . 135
11.5 Model assessment and profiling . . . . . 140
11.6 Case study: A bank's direct mail campaign . . . . . 143

**12 Conclusions and future challenges** **155**

**Bibliography** **157**

**Appendix A: uplift package manual** **175**

**Appendix B: Manuscripts linked to chapters** **203**

# List of Figures

2.1 Optimal decision boundaries for the red/blue envelope example in two simulated scenarios. The covariates $X_1$ and $X_2$ are independently generated from a uniform distribution $U(-1, 1)$. In the left figure, the optimal personalized treatment rule $\mathcal{H}^* =$ {blue envelope} if $1 - X_1 - X_2 > 0$, and $\mathcal{H}^* =$ {red envelope} otherwise. In the right figure, $\mathcal{H}^* =$ {blue envelope} if $(0.5 - X_1^2 - X_2^2)(X_1^2 + X_2^2 - 0.3) > 0$, and $\mathcal{H}^* =$ {red envelope} otherwise. . . . . 8

2.2 Causal conditional inference tree on the NSW data. The tree is designed to partition the universe of subjects into subgroups with heterogeneous treatment effects. The tree identifies the characteristics of workers for whom the program was most beneficial. . . . . 9

2.3 Boxplots of the actual average treatment effect for the high/low-scored patients based on 100 random training/test data partitions. The overall average treatment effect is shown by the red horizontal dotted line. Tamoxifen is more effective than the alternative treatment on high-scored patients. The opposite holds for low-scored patients. . . . . 11

4.1 Alternative split criteria used by uplift random forests as a function of $P\{Y(A) = 1\}$, $A \in \{0, 1\}$. . . . . 29

4.2 Model performance from fitting uplift random forests with $n = \sqrt{p}$ versus $n = p/3$ for an increasing ratio of total variables to relevant variables. A relevant variable is one that interacts with the treatment. Performance is measured by the Spearman's rank correlation coefficient between the estimated personalized treatment effect (PTE) and the "true" PTE. The dots outside the boxplots represent outliers. We used the "1.5 rule" for determining if a data point is an outlier: less than $Q1 - 1.5 \times (Q3 - Q1)$ or greater than $Q3 + 1.5 \times (Q3 - Q1)$, where $Q1$ and $Q3$ represent the first and third quartiles, respectively. . 31

4.3 Model performance comparison among uplift split criteria for increasing values of $\sigma_0$. . . . . 32

4.4 Impact of controlling the depth level of the individual uplift trees on model performance under two levels of noise: $\sigma_0 = \sqrt{2}$ (Low) and $\sigma_0 = 4\sqrt{2}$ (High). . . . . 34

6.1 Boxplots of the Spearman's rank correlation coefficient between the estimated treatment effect $\hat{\tau}(X)$ and the "true" treatment effect $\tau(X)$ for all methods. The plots illustrate the results for simulation scenarios 1–4, which model a situation with "stronger" treatment heterogeneity effects, under zero and moderate correlation among the covariates ($\rho = 0$ and $\rho = 0.5$) and two levels of noise ($\sigma_0 = \sqrt{2}$ and $\sigma_0 = 2\sqrt{2}$). The boxplots within each simulation scenario are shown in decreasing order of performance based on the average correlation. The dots outside the boxplots represent outliers, determined using the "1.5 rule". . . . . 46

6.2 Boxplots of the Spearman's rank correlation coefficient between the estimated treatment effect $\hat{\tau}(X)$ and the "true" treatment effect $\tau(X)$ for all methods. The plots illustrate the results for simulation scenarios 5–8, which model a situation with "weaker" treatment heterogeneity effects, under zero and moderate correlation among the covariates ($\rho = 0$ and $\rho = 0.5$) and two levels of noise ($\sigma_0 = \sqrt{2}$ and $\sigma_0 = 2\sqrt{2}$). The boxplots within each simulation scenario are shown in decreasing order of performance based on the average correlation. . . . . 47

7.1 Sample Qini curves corresponding to model $M$ and random targeting strategies. . . . . 51

7.2 Qini curves corresponding to the optimal model $M^{\text{opt}}$ and random model in the absence (*left*) and presence (*right*) of negative effects. . . . . 53

7.3 Qini coefficient performance profile of causal conditional inference forests under increasing $P$ values based on different resampling methods. For each method, we show the mean and ± one standard error performance estimates. The results are based on the bank's direct mail campaign dataset. . . . . 59

7.4 Calibration plot for a causal conditional inference forest model fitted to the *bank* dataset. . . . . 61

8.1 Boxplots of the actual average treatment effect (ATE) for each decile based on 100 random training/validation data splits. The first (tenth) decile represents the 10% of clients with highest (lowest) predicted PTE. Clients with higher estimated PTE were, on average, positively influenced to buy *as a result* of the marketing intervention activity. . . 67

8.2 This figure illustrates the Qini curve for each model based on the client retention dataset. This curve shows the cumulative number of incremental retained customers relative to the cumulative number of targets (both expressed as a percentage of the total targets). The diagonal line depicts the theoretical incremental retained customers from random targeting. The Qini coefficient ($q$) is obtained by subtracting the area under the random curve from the area under the Qini curve and represents a single estimate of model performance. The approach based on uplift random forest (red) performs best in this application. . . . . 71

9.1 Estimated propensity scores for all treatment dichotomies. Given a treatment dichotomy $(j, k)$, each plot illustrates the distribution of the probability of assignment to rate change level $j$ relative to level $k$, conditional on the background covariates. Within each dichotomy, the rate change level with fewer units is represented by $j$, and the other level with $k$. The propensity scores are labeled "Linear Propensity Scores" to reflect the fact that they are in the log-odds scale. . . . . 91

9.2 Empirical quantile-quantile plot of `premium` before and after matching in the (5,1) treatment dichotomy. This figure displays the quantiles of `premium` for treatment 5 vs. treatment 1 before (left) and after (right) matching. A red 45° reference line is also plotted (indicating perfect balance). Balance for this variable was clearly improved after matching. . . . . 94

9.3 Price-elasticity functions. The plots illustrate the average estimated lapse rate measured at each rate change level for selected subpopulations within the insurance portfolio. Continuous covariates have been categorized at the quartiles of their distributions (labeled with the numbers 1 to 3 in the plots). . . . . 96

9.4 Expected profit efficient frontier. The efficient frontier represents the maximum expected profit that the company can obtain at a given desired retention rate. The expected profit is expressed in terms of change, measured in percentage points, relative to the profit at the current state. The current situation for this company is not optimal, in that it is possible to obtain an increase in profit at the current retention level (A), a higher retention at the current profit (B), or even a reduction in client base while increasing profit (C). . . . . 99

10.1 The relationship between training and cross-validation error and the optimal number of boosting iterations (shown by the vertical green line). . . . . 117
10.2 Relative importance of the predictors for the frequency (*left*) and severity (*right*) models. See Table 10.1 for predictor variable codes. . . . . 119
10.3 Partial dependence plots (frequency model). . . . . 121
10.4 Partial dependence plots (severity model). . . . . 122
10.5 Partial dependence of claim severity on *years licensed* (DC2) and *horsepower to weight ratio* (VC4). . . . . 123
10.6 Prediction accuracy of GB relative to GLM (based on test sample). . . . . 124

11.1 Relative importance of PTE predictors based on `ccif`. . 149

# List of Tables


6.1 Simulation scenarios . . . . . 44

8.1 Cross-sell rates by group . . . . . 65
8.2 Attrition rates by group . . . . . 68
8.3 Top decile uplift . . . . . 71

9.1 Lapse rates by rate change level . . . . . 86
9.2 Balance results of the covariates before and after matching for the first four treatment dichotomies . . . . . 93

10.1 Overview of loss cost predictors . . . . . 114

11.1 Distribution of predictors using `modelProfile` . . . . . 143
11.2 Distribution of specific predictors using `modelProfile` . . 153

# Acknowledgements

First and foremost, I would like to express to my PhD advisor, Dr. Montserrat Guillén, my great appreciation for her support and brilliant guidance during my dissertation work. Her mentorship drew me along a path of growth, learning, and discovery, which seminally contributed to my research and professional skills. Her continuous encouragement for my ideas and trust in my work made writing this thesis a genuinely enjoyable experience. I have been very privileged to have had as an advisor someone who is not only enthusiastic and caring, but also very supportive at the personal level. I cannot thank Montserrat enough for making me feel so comfortable during my stays in Barcelona. I shall always appreciate her guidance in introducing me to the extraordinary world of rigorous scientific research and publishing.

Also at the University of Barcelona, I owe my gratitude to Dr. Ana Maria Pérez-Marín, for our productive collaboration in various research projects, always leading to a positive outcome.

I would also like to thank the Royal Bank of Canada and RBC Insurance for awarding me their Major Educational Sponsorship, which entirely funded my PhD studies. I would especially like to thank Christopher Cooney and Don De la Paz for their unstinting support. Without them, this work would not have been possible.

A very special thanks to my lovely wife, Evelyn, and my beautiful son, Ulises, for inspiring me to bring out the best in myself every day. Their support and understanding of all the effort and commitment my PhD work entailed have been essential during my dissertation process. They are the most important part of everything I accomplish.

# Abstract

In many important settings, subjects can show significant heterogeneity in response to a stimulus or "treatment". For instance, a treatment that works for the overall population might be highly ineffective, or even harmful, for a subgroup of subjects with specific characteristics. Similarly, a new treatment may not be better than an existing treatment in the overall population, but there is likely a subgroup of subjects who would benefit from it. The notion that "one size may not fit all" is becoming increasingly recognized in a wide variety of fields, ranging from economics to medicine. This has drawn significant attention to personalize the choice of treatment, so it is optimal for each individual. An optimal personalized treatment is the one that maximizes the probability of a desirable outcome. We call the task of learning the optimal personalized treatment *personalized treatment learning (PTL)*.

From the statistical learning perspective, building PTL models imposes important challenges, primarily because the optimal treatment is unknown on a given training data set. In this thesis, we formalize the PTL problem from a *causal inference* perspective and provide a comprehensive description of the existing methods to solve this problem. We contribute to the PTL literature by proposing two novel methods, namely *uplift random forests* and *causal conditional inference forests*. Our proposal outperforms the existing methods based on an extensive numerical simulation and real-world data.

Next, we introduce the concept of PTL models to *insurance marketing* and *pricing* applications. In particular, we contribute to the insurance literature in these areas by proposing PTL methods to optimize *client retention* and *cross-selling* in insurance from experimental data. We also illustrate an application of these methods to *price-elasticity estimation*

and *insurance economic price optimization* in the context of observational data. In the insurance field, the selection of the optimal personalized treatment also requires consideration of the expected insurance losses of each individual policyholder within the portfolio. We contribute to the non-life insurance ratemaking literature by proposing a novel application of *gradient boosting models* to estimate *insurance loss cost*, with key important advantages over the conventional *generalized linear model* approach.

A key problem facing research in this field, has been the lack of publicly available statistical software to estimate PTL models. We implement most of the existing methods for fitting these models, as well as our proposed ones, in a package named **uplift**, which is now released and freely available from the CRAN (Comprehensive R Archive Network) repository under the R statistical computing environment.

# 1 Introduction

In the past two decades, rapid advances in data collection and storage technology have created vast quantities of data. The field of statistics has been revolutionized by the development of algorithmic and data models (Breiman, 2001b) in response to challenging new problems coming from science and industry, mostly resulting from an increasing size and complexity in the data structures. In this context, the concept of *learning from data* (Abu-Mostafa et al., 2012) has emerged as the task of extracting "implicit, previously unknown, and potentially useful information from data" (Frawley et al., 1992). A distinction is usually made between *supervised* and *unsupervised* learning. In the former, the objective is to predict the value of a response variable based on a collection of *observable* covariates. In the latter, there is no response variable to "supervise" the learning process, and the objective is to find structures and patterns among the covariates.

In many important settings, the values of some covariates are not only observable, but they can be chosen at the discretion of a decision maker (Zliobaitė and Pechenizkiy, 2010). For instance, a doctor can choose the medical treatment for a patient among a set of alternatives, a company can decide the type of marketing intervention activity (direct mail, phone call, email, etc.) to make an offer to a client, a bank can decide the credit limit to offer a client on a credit card. In all these examples, the objective is not necessarily to predict a response variable with high accuracy, but to select the optimal action or "treatment" for each subject based on his or her individual characteristics. Optimal is understood here as the treatment that maximizes the probability of a desirable outcome. We call the task of learning the optimal personalized treatment *personalized treatment learning (PTL)*.

A key challenge in building models for PTL is that the quantity we are trying to predict (i.e., the optimal personalized treatment) is unknown on a given training data set. As each subject can only be exposed to a single treatment, the value of the subject's response under alternative treatments is unobserved, a problem also known as *the fundamental problem of causal inference* (Holland, 1986). This aspect makes this problem unique within the discipline of learning from data.

The underlying motivation for PTL is that subjects can show significant heterogeneity in response to treatments, so making an accurate treatment choice for each subject becomes essential. For instance, a new treatment may not be better than an existing treatment in the overall population, but it might be beneficial/harmful for a subgroup of subjects. The idea that "one size may not fit all" has been increasingly recognized in a variety of disciplines, ranging from economics to medicine. Alemi et al. (2009) argue that improved statistical methods are needed for personalized treatments and propose an adapted version of the *K-nearest-neighbor (KNN)* classifier (Cover and Hart, 1967). Imai and Ratkovic (2013) propose a method that adapts the *support vector machine* classifier (Vapnik, 1995) and then apply it to a widely known dataset pertaining to the National Supported Work program (LaLonde, 1986; Dehejia and Wahba, 1999) to identify the characteristics of workers who greatly benefit from (or are negatively affected by) a job training program. Tian et al. (2014) propose a method designed to deal with high-dimensional covariates and use it to identify breast cancer patients who may or may not benefit from a specific treatment based on the individual patient's gene expression profile. In the context of insurance, Guelman et al. (2012, 2014c) propose a method based on an adapted version of *random forests* to identify policyholders who are positively/negatively impacted by a client retention program. The same authors subsequently propose an algorithm called *causal conditional inference forests* to optimize insurance cross-sell strategies (Guelman et al., 2014a,b). Also, Guelman and Guillén (2014) describe a framework to determine the optimal rate change (playing the role of the treatment) for each individual policyholder for the purpose of maximizing the overall expected profitability of an insurance portfolio. In addition to the methods discussed above, other methods have been proposed in the literature, mostly in the context of clinical trials and direct

marketing (Su et al., 2009; Qian and Murphy, 2011; Zhao et al., 2012; Jaśkowski and Jaroszewicz, 2012; Larsen, 2009; Radcliffe and Surry, 2011; Rubin and Waterman, 2006; Tang et al., 2013). However, considering the critical importance of these methods to many scientific disciplines and policy making, PTL models have received relatively little attention in the literature.

In this thesis, we provide a comprehensive description of the existing PTL methods and propose two novel methods. Our proposal outperforms the existing methods in an extensive numerical study. We illustrate several novel applications of the proposed methods to insurance marketing in the context of experimental data. We also present an application of PTL models to economic price optimization in the context of observational data within the field of insurance. We implement most of the statistical methods and algorithms described in this thesis in a package named **uplift** (Guelman, 2014), which is now freely available from the CRAN (Comprehensive R Archive Network) repository under the R statistical computing environment. Although PTL models have applications to a wide variety of fields, in this thesis we focus mostly on insurance-related applications. In this context, the selection of the optimal personalized treatment also requires consideration of the expected insurance losses of each individual policyholder within the portfolio. We describe an unprecedented application of *gradient boosting models* to estimate loss cost in non-life insurance, with key advantages over the conventional *generalized linear model* approach.

This thesis is organized as follows. Chapter 2 defines the scope of the PTL problem and gives some examples. Chapter 3 follows with a detailed description of seven existing methods for tackling this problem. In Chapters 4 and 5 we introduce two new proposed methods. In Chapter 6 we report the finite sample performance of all methods under an extensive numerical simulation. Chapter 7 describes model assessment and selection for PTL models. Chapter 8 describes two empirical applications in insurance marketing in the context of client retention and cross-selling. Chapter 9 illustrates an application to price elasticity modeling and economic price optimization in auto insurance. Insurance loss cost considerations are discussed in Chapter 10. Lastly, in Chapter 11 we provide a practical guide for using the **uplift** package. In Appendix A,

we include the package manual with further details about the package functionality. Our published and submitted articles linked to this thesis are listed in Appendix B.

# 2 The personalized treatment learning problem

## 2.1 Problem formulation

We frame the *personalized treatment learning (PTL)* problem in the context of Rubin's model of causality (Rubin, 1974, 1977, 1978, 2005). Under this model, we conceptualize the learning problem in terms of the potential outcomes under treatment alternatives, only one of which is observed for each subject. The causal effect of a treatment on a subject is defined in terms of the difference between an observed outcome and its counterfactual. The notation introduced below will be used throughout the thesis, except where indicated otherwise.

In the following, we use upper-case letters to denote random variables and lower-case letters to denote values of the random variables. Assume that a sample of subjects is randomly assigned to two treatment arms, denoted by $A$, $A \in \{0, 1\}$, also referred as control and treatment states, respectively. Let $Y(a) \in \{0, 1\}$ denote a binary potential outcome of a subject if assigned to treatment $A = a$, $a = \{0, 1\}$. The observed outcome is $Y = AY(1) + (1 - A)Y(0)$. Each subject is characterized by a $p$-dimensional vector of baseline covariates $\mathbf{X} = (X_1, \ldots, X_p)^\top$. We assume the data consists of $L$ independent and identically distributed realizations of $(Y, A, \mathbf{X})$, $\{(Y_\ell, A_\ell, \mathbf{X}_\ell),\ \ell = 1, \ldots, L\}$.

Under the assumption of randomization, treatment assignment $A$ ignores its possible impact on the outcomes $Y(0)$ and $Y(1)$, and hence they are independent – using the notation of Dawid (1979), $\{Y_\ell(0), Y_\ell(1) \perp A_\ell\}$. In this context, the *average treatment effect* (ATE) can be estimated by

$$\begin{aligned}\tau &= E[Y_\ell(1) - Y_\ell(0)] \\ &= E[Y_\ell|A_\ell = 1] - E[Y_\ell|A_\ell = 0].\end{aligned} \tag{2.1}$$

In observational studies, subjects assigned to different treatment conditions are not exchangeable and thus direct comparisons can be misleading (Rosenbaum and Rubin, 1983).

In many circumstances, subjects can show significant heterogeneity in response to treatments, in which case the ATE is of limited value. The problem addressed in this thesis is the identification of subgroups of subjects for which the treatment is most beneficial (or most harmful). As discussed by Holland and Rubin (1989), the most granular level of causal inference is the *individual treatment effect* (ITE), defined by $Y_\ell(1) - Y_\ell(0)$ for each subject $\ell = \{1, \ldots, L\}$. However, this is an unobserved quantity, as a subject is never observed simultaneously in both treatment states. The best approximation to the ITE that is possible to obtain in practice is the *subpopulation treatment effect* (STE), which is defined for a subject with individual covariate profile $\mathbf{X}_\ell = \mathbf{x}$ by

$$\begin{aligned}\tau(\mathbf{x}) &= E[Y_\ell(1) - Y_\ell(0)|\mathbf{X}_\ell = \mathbf{x}] \\ &= E[Y_\ell|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 1] - E[Y_\ell|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 0].\end{aligned} \tag{2.2}$$

Understanding the precise nature of the STE variability can be extremely valuable in personalizing the choice of treatment, so that it is most appropriate for each individual. Henceforward in this thesis, we use the term *personalized treatment effect* (PTE) to refer to the subpopulation treatment effect (2.2).

A *personalized treatment rule* $\mathcal{H}$ is a map from the space of baseline covariates $\mathbf{X}$ to the space of treatments $A$, $\mathcal{H}(\mathbf{X}) : \mathbb{R}^p \to \{0, 1\}$. An *optimal treatment rule* is one that maximizes the expected outcome, $E[Y(\mathcal{H}(\mathbf{X}))]$, if the personalized treatment rule is implemented for the whole population. Notice that since $Y$ is binary, this expectation has a probabilistic interpretation. That is, $E[Y(\mathcal{H}(\mathbf{X}))] = P\Big(Y(\mathcal{H}(\mathbf{X})) = 1\Big)$ and thus $\tau(\mathbf{x}) \in [-1, 1]$.

A straightforward calculation gives the optimal personalized treatment rule $\mathcal{H}^* = \text{argmax}_{\mathcal{H}} E[Y(\mathcal{H}(\mathbf{X}))]$ for a subject with covariates $\mathbf{X}_\ell = \mathbf{x}$ as $\mathcal{H}^* = 1$ if $\tau(\mathbf{x}) > 0$, and $\mathcal{H}^* = 0$ otherwise. In many situations, the alternative treatments have unequal costs, in which case the decision rule can simply be replaced by $\mathcal{H}^* = 1$ if $\tau(\mathbf{x}) > c$, and $\mathcal{H}^* = 0$ otherwise, for some constant threshold $c \in [-1, 1]$.

## 2.2 Examples of the personalized treatment learning problem

The PTL problem is encountered in many scientific disciplines and policy making. We describe some examples below.

### 2.2.1 Example 1: The red/blue envelope problem

A common business problem in *direct marketing* is to decide which existing or potential customers a company should contact to promote a product or service. For simplicity, suppose the company sends a product offer by direct mail to its existing customers in either a red or blue envelope. Further, assume the cost of the offer is the same with both colors. Some customers may be more likely to buy the product with the red envelope and others with the blue one. The PTL problem is to select the optimal treatment for each customer – namely, the envelope color that maximizes the probability of purchase. Figure 2.1 illustrates two artificial scenarios for the optimal envelope color for each customer based on a simplified case with two baseline covariates. The company should send the blue (red) envelope to customers represented by the blue (red) dots with corresponding covariates $X_1$ and $X_2$. The optimal decision boundaries are shown in green. The problem is non-trivial as the optimal envelope color for each customer is unknown on a given training dataset.

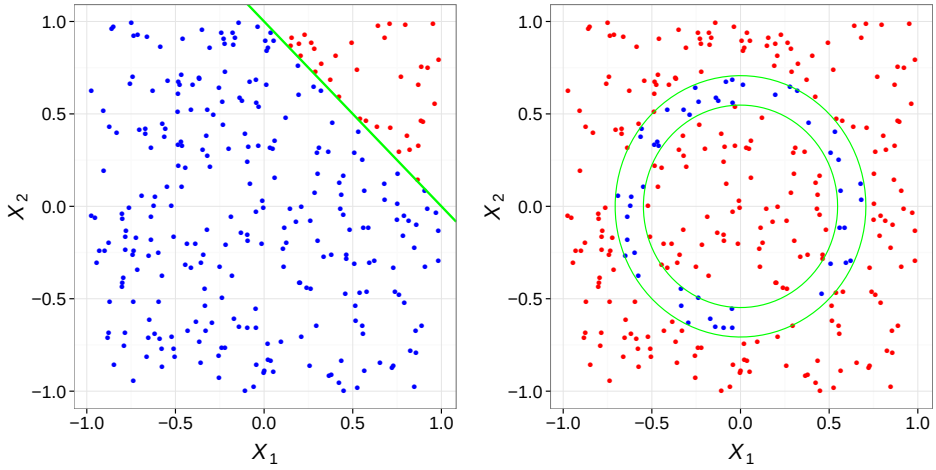


Figure 2.1: Optimal decision boundaries for the red/blue envelope example in two simulated scenarios. The covariates $X_1$ and $X_2$ are independently generated from a uniform distribution $U(-1, 1)$. In the left figure, the optimal personalized treatment rule $\mathcal{H}^* = \{\text{blue envelope}\}$ if $1 - X_1 - X_2 > 0$, and $\mathcal{H}^* = \{\text{red envelope}\}$ otherwise. In the right figure, $\mathcal{H}^* = \{\text{blue envelope}\}$ if $(0.5 - X_1^2 - X_2^2)(X_1^2 + X_2^2 - 0.3) > 0$, and $\mathcal{H}^* = \{\text{red envelope}\}$ otherwise.

### 2.2.2 Example 2: Evaluating personalized treatment effects from a job training program

The National Supported Work Demonstration (NSW) program was a short-term employment program implemented in the mid-1970s in the United States and designed to help individuals facing economic and social difficulties to move into the labor market. Unlike other employment programs, the NSW program was a field experiment, in which a heterogenous group of individuals were randomly assigned to join the program. Those assigned to the program (i.e., the treatment group) participated in various types of work and received counselling in a sheltered environment. Those assigned to the control group did not receive any type of support. Information on the pre-treatment variables was obtained from initial surveys and Social Security Administration records, and includes earnings at the start of the program, age, education, ethnicity, and marital status.

The NSW dataset has been extensively analyzed by LaLonde (1986) and Dehejia and Wahba (1999). One of the main points of focus in those studies was in estimating the ATE, as defined in (2.1). Our interest here is in identifying the characteristics of workers for whom the program was

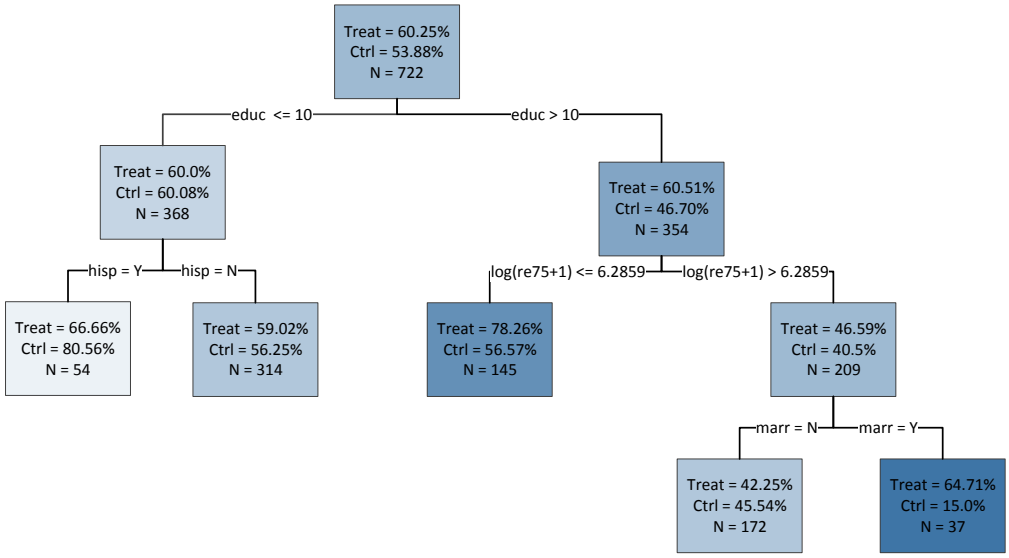


Figure 2.2: Causal conditional inference tree on the NSW data. The tree is designed to partition the universe of subjects into subgroups with heterogeneous treatment effects. The tree identifies the characteristics of workers for whom the program was most beneficial.

most beneficial. For illustration purposes, we focused on the same set of individuals described in Imai and Ratkovic (2013), consisting of 297 treated subjects and 425 controls. The outcome of interest is a binary indicator which denotes whether the earnings increased after the job training program (measured in 1978) relative to the earnings at the start of the program. We built a *causal conditional inference tree*, which we discuss further in Chapter 5. Figure 2.2 shows the results. Each node in the tree shows the proportion of treated and control subjects with increased earnings after the job training program. It appears that the program was most effective for individuals with higher number of years of education and relatively lower earnings at the start of the program, as well as those married with higher earnings. The program was least effective for Hispanics with low education.

### 2.2.3 Example 3: Identifying breast cancer patients who may benefit from a new treatment

A number of studies have reported that breast cancer can be classified into molecular subtypes on the basis of distinct gene expression profiles (Loi et al., 2007). These subtypes have been associated with different clinical outcomes with respect to the effectiveness of specific treatments (Tian et al., 2014). A publicly available dataset[1], collected by Loi et al. (2007), consists of 277 patients treated with tamoxifen and 137 patients treated with an alternative method. Each patient is characterized by 44,928 gene expression measurements and by demographic information. In this example, the goal is to identify which patients are more likely to benefit from (or be harmed by) tamoxifen.

For illustrative purposes, we define the outcome of interest as a binary indicator of distant metastasis–free five-year survival. This outcome was transformed using the *modified outcome method*, discussed in Chapter 3, for the purpose of estimating the personalized treatment effect of tamoxifen. We restricted the analysis to patients with complete information. After exclusions, there were 221 and 116 patients treated with tamoxifen and the alternative treatment, respectively. We selected 75 patients from each treatment arm as the training set and left the remaining number of patients as a test sample.

We first used principal component analysis to preliminarily reduce the dimensionality of the covariate space. From this analysis, we decided to keep the first 20 principal components[2]. We subsequently fitted a LASSO (least absolute shrinkage and selection operator) logistic regression (Tibshirani, 1996) on the modified outcome, using the principal components as predictors. The LASSO penalty was selected via a 10-fold cross-validation procedure. Patients were then classified into high/low treatment score groups, depending on whether the predicted tamoxifen-treatment effect was greater/smaller than the median level.

[1] The data can be downloaded from: http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE6532.

[2] We used a *scree plot* to guide the selection of an appropriate number of principal components. A scree plot is a plot of the proportion of the total variance explained by each component (in decreasing order) versus its number (Johnson and Wichern, 2001, p. 441).

Figure 2.3 shows the boxplots of the actual ATE for the high/low-scored patients based on 100 random training/test data partitions. The actual treatment effect is measured as the difference between the fraction of patients with distant metastasis–free five-year survival in the tamoxifen group and the corresponding fraction in the alternative treatment group. Notice that tamoxifen is more effective than the alternative treatment on high-scored patients (i.e., the fraction of patients with distant metastasis–free five-year survival in the tamoxifen group is higher than that in the alternative treatment group). On low-scored patients, the alternative treatment is more effective than tamoxifen.

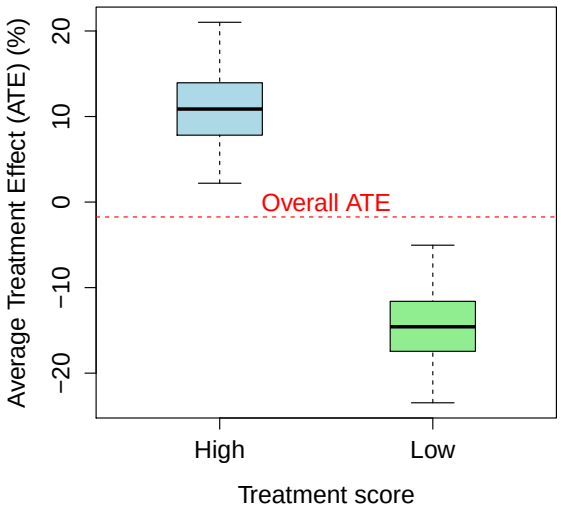


Figure 2.3: Boxplots of the actual average treatment effect for the high/low-scored patients based on 100 random training/test data partitions. The overall average treatment effect is shown by the red horizontal dotted line. Tamoxifen is more effective than the alternative treatment on high-scored patients. The opposite holds for low-scored patients.

# 3 Existing personalized treatment learning models

## 3.1 Introduction

In this chapter, we describe seven methods discussed in the literature for estimating the personalized treatment effect (PTE). We can generally classify the methods under *indirect estimation methods* and *direct estimation methods*. The former methods propose a systematic two-stage procedure to estimate the PTE. In the first stage, they attempt to achieve high accuracy in predicting the outcome $Y$ conditional on the covariates $\mathbf{X}$ and treatment $A$. In the second stage, they subtract the predicted value of $Y$ under each treatment to obtain a PTE estimate. The latter methods attempt to directly estimate the difference $E[Y(1) - Y(0)|\mathbf{X} = \mathbf{x}]$ in the potential responses between the two treatments conditional on the covariates $\mathbf{X}$. The task is non-trivial as the "true" treatment effect is not observed at the individual subject level on a given training data set.

## 3.2 Indirect estimation methods

### 3.2.1 Difference score method

The most intuitive approach to estimating the PTE, known as the *difference score* method, is to fit two independent models for the response $Y$, one based on the treated subjects, $E[Y|\mathbf{X}, A = 1]$, and one based on the control subjects, $E[Y|\mathbf{X}, A = 0]$ (Larsen, 2009). An estimate of the PTE for a subject with covariate $\mathbf{X}_\ell = \mathbf{x}$ is then obtained by subtracting the estimated values of the response from the two models. That is,

$$\hat{\tau}(\mathbf{x}) = (\hat{Y}_\ell | \mathbf{X} = \mathbf{x}_\ell, A_\ell = 1) - (\hat{Y}_\ell | \mathbf{X} = \mathbf{x}_\ell, A_\ell = 0). \tag{3.1}$$

Any conventional statistical or algorithmic binary classification method may serve to fit the models.

### 3.2.2 Interaction method

A second method in the same spirit as the difference score method is the *interaction* approach proposed by Lo (2002). This method consists in fitting a single model to the response on the main effects and adding interaction terms between each covariate $\mathbf{X} = (X_1, \ldots, X_p)^\top$ and the treatment indicator $A$. If the model is fitted using standard logistic regression, the estimated parameters of the interaction terms measure the additional effect of each covariate due to treatment. An estimate of the PTE for a subject with covariates $\mathbf{X}_\ell = \mathbf{x}$ is obtained by subtracting the predicted probabilities by setting, in turn, $A_\ell = 1$ and $A_\ell = 0$ in the fitted model.

### 3.2.3 L2-SVM method

The interaction method represents an improvement over the difference score method in that it provides a formal means of performing significance tests of the interaction parameters between the treatment and the covariates. However, it suffers from *overfitting* problems – which occur when a model describes the random error or noise in the data instead of the true underlying relationship – when including all interaction effects with a high-dimensional covariate space (Zhao and Zeng, 2013). Although overfitting may be prevented by using LASSO logistic regression (Tibshirani, 1996) for variable selection and shrinkage, this method places the same LASSO constraints over main and treatment heterogeneity parameters. This may be problematic as the variability in the response attributable to the interaction effects is usually a small fraction of the variability attributable to the main effects. To address this problem, a third method proposed by Imai and Ratkovic (2013), called *L2-SVM*, is an adapted version of the support vector machine (SVM) classifier (Vapnik, 1995). The SVM can be expressed as a penalization method (Hastie et al., 2009,

p. 426) and this can be adapted to include separate LASSO constraints over the main and treatment heterogeneity parameters. Specifically, let $Y_\ell^* = 2Y_\ell - 1 \in \{-1, 1\}$ and consider the optimization problem

$$\min_{(\alpha,\theta)} \sum_{\ell=1}^{L} \left|1 - Y_\ell^*(\mu + \alpha^\top \mathbf{X}_\ell + \theta^\top \mathbf{X}_\ell A_\ell)\right|_+^2 + \lambda_{\mathbf{X}} \sum_{j=1}^{p} |\alpha_j| + \lambda_{\mathbf{XA}} \sum_{j=1}^{p} |\theta_j|, \quad (3.2)$$

where $\lambda_{\mathbf{X}}$ and $\lambda_{\mathbf{XA}}$ are pre-specified separate LASSO penalties for the main effect parameters $\alpha$ and treatment heterogeneity parameters $\theta$, respectively, $|t|_+ \equiv \max(t, 0)$ is the hinge-loss (Wahba, 2002), and $\mu$ is a constant term.

After model (3.2) is estimated, a PTE estimate can be obtained as follows. Let $\hat{R}_\ell = \hat{\mu} + \hat{\alpha}^\top \mathbf{X}_\ell + \hat{\theta}^\top \mathbf{X}_\ell A_\ell$ and $\hat{R}_\ell^*$ denote the predicted value $\hat{R}_\ell$ truncated at positive and negative one. The PTE is estimated as the difference in the truncated values of the predicted response under each treatment condition. That is,

$$\hat{\tau}(\mathbf{x}) = \frac{1}{2}\Big[(\hat{R}_\ell^* | \mathbf{X} = \mathbf{x}_\ell, A_\ell = 1) - (\hat{R}_\ell^* | \mathbf{X} = \mathbf{x}_\ell, A_\ell = 0)\Big], \quad (3.3)$$

where $\hat{\tau}(\mathbf{x}) \in [-1, 1]$.

A key problem with the indirect estimation methods is the mismatch between the target variable they attempt to estimate and the target variable defined in (2.2). For instance, even when any of the indirect estimation methods are correctly specified to predict $Y_\ell$ conditional on covariates $\mathbf{X}_\ell = \mathbf{x}$ and treatment $A$, it is not guaranteed that these models can accurately predict $Y_\ell(1) - Y_\ell(0)$ conditional on the same covariates. This is because these methods emphasize the prediction accuracy on the response, not the accuracy in estimating the change in the response *caused* by the treatment at the subject level.

## 3.3 Direct estimation methods

### 3.3.1 Modified covariate method

This method, proposed by Tian et al. (2014), consists in modifying the covariates in a simple way, and then fitting an appropriate regression model using the modified covariates. A key advantage of this approach is

that it avoids having to model the main effects directly.

Specifically, the *modified covariate* method involves performing the following steps: i) transform the treatment indicator as $A_\ell^* = 2A_\ell - 1 \in \{-1, 1\}$, ii) transform each covariate in $\mathbf{X}_\ell$ as $\mathbf{Z}_\ell = \mathbf{X}_\ell^* A_\ell^*/2$, where $\mathbf{X}^*$ is the centered version of $\mathbf{X}$, and iii) fit a regression model to predict the outcome variable $Y$ on the modified covariates $\mathbf{Z}$. For instance, using a logistic regression model, estimate

$$P(Y = 1|\mathbf{X}, A) = \frac{\exp(\gamma^\top \mathbf{Z})}{1 + \exp(\gamma^\top \mathbf{Z})}. \tag{3.4}$$

Under the very general assumption that $P(A^* = 1) = P(A^* = -1) = 1/2$, a surrogate to the PTE for a subject with covariates $\mathbf{X}_\ell = \mathbf{x}$ is given by

$$\hat{\tau}(\mathbf{x}) = \frac{\exp(\hat{\gamma}^\top \mathbf{x}/2) - 1}{\exp(\hat{\gamma}^\top \mathbf{x}/2) + 1}. \tag{3.5}$$

To see that (3.5) is an appropriate estimate, we must consider the maximum likelihood estimator of model (3.4). It is easy to see (Tian et al., 2014) that the optimizer of $E\{l(Y, f(\mathbf{X})A^*)\}$ is given by

$$f^*(\mathbf{x}) = \log\Big\{\frac{1 + \tau(\mathbf{x})}{1 - \tau(\mathbf{x})}\Big\}, \tag{3.6}$$

where $l(.)$ is the Bernoulli log-likelihood, $f(\mathbf{X}) = \gamma^\top \mathbf{X}^*/2$, and $\tau(\mathbf{x})$ is the PTE defined in (2.2). Therefore, (3.5) may serve as an estimate of the PTE.

In case the dimension of $\mathbf{X}$, $p$, is high, appropriate variable selection procedures can be applied to the modified data directly. For instance, an $L1$-*regularized* logistic regression (Hastie et al., 2009, p. 125) can be estimated by minimizing $\frac{1}{L}\sum_{\ell=1}^{L} -\{Y_\ell \gamma^\top \mathbf{Z}_\ell - \log(1+\exp(\gamma^\top \mathbf{Z}_\ell)) + \lambda_0 \sum_{j=1}^{p} |\gamma_j|\}$, where $\lambda_0$ is a pre-specified LASSO penalty.

In the derivation above, the assumption of equal probability of treatments is used. This may be perceived as very restrictive since this assumption is unlikely to hold in practice. However, various resampling methods from the machine learning literature (Weiss and Provost, 2003; Estabrooks et al., 2004; Chawla, 2005) could be used for the purpose of balancing treatments.

### 3.3.2 Modified outcome method

The *modified outcome method*, proposed by Jaśkowski and Jaroszewicz (2012), consists in first defining a new outcome variable $W$ such that

$$W_\ell = \begin{cases} 1 & \text{if } A_\ell = 1 \text{ and } Y_\ell = 1 \\ 1 & \text{if } A_\ell = 0 \text{ and } Y_\ell = 0 \\ 0 & \text{otherwise,} \end{cases}$$

and then fitting a binary regression model to $W$ on the baseline covariates $\mathbf{X}$. If we assume that a value of $Y = 1$ is more desirable that $Y = 0$, we can intuitively think of $W = 1$ as the event of obtaining a potential outcome under treatment which is at least as good as the observed outcome. The probability of this event is given by

$$\begin{aligned} P(W_\ell = 1|\mathbf{X}_\ell = \mathbf{x}) &= P(W_\ell = 1|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 1)P(A_\ell = 1|\mathbf{X}_\ell = \mathbf{x}) + \\ &\quad P(W_\ell = 1|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 0)P(A_\ell = 0|\mathbf{X}_\ell = \mathbf{x}) \\ &= P(Y_\ell = 1|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 1)P(A_\ell = 1|\mathbf{X}_\ell = \mathbf{x}) + \\ &\quad P(Y_\ell = 0|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 0)P(A_\ell = 0|\mathbf{X}_\ell = \mathbf{x}) \\ &= P(Y_\ell = 1|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 1)P(A_\ell = 1) + \\ &\quad P(Y_\ell = 0|\mathbf{X}_\ell = \mathbf{x}, A_\ell = 0)P(A_\ell = 0), \end{aligned}$$

where the last equality follows from the randomization assumption. Now, making the same assumption as in the modified covariate method that $P(A = 1) = P(A = 0) = 1/2$, we obtain

$$\begin{aligned} \tau(\mathbf{x}) &= P(Y_\ell = 1|A_\ell = 1, \mathbf{X}_\ell = \mathbf{x}) - P(Y_\ell = 1|A_\ell = 0, \mathbf{X}_\ell = \mathbf{x}) \\ &= 2P(W_\ell = 1|\mathbf{X}_\ell = \mathbf{x}) - 1. \end{aligned}$$

Hence, if for instance a logistic regression model is used to estimate

$$P(W = 1|\mathbf{X}, A) = \frac{\exp(\beta^\top \mathbf{X})}{1 + \exp(\beta^\top \mathbf{X})}, \tag{3.7}$$

then

$$\hat{\tau}(\mathbf{x}) = 2\frac{\exp(\hat{\beta}^\top \mathbf{X})}{1+\exp(\hat{\beta}^\top \mathbf{X})} - 1 \tag{3.8}$$

can be used as a surrogate to the PTE.

We next show that the maximum likelihood estimator (MLE) of the working models (3.7) and (3.4) are equivalent and so they produce similar PTE estimates.

**Proposition 3.3.1.** *Maximum likelihood estimates of personalized treatment effects from the modified covariate and modified outcome methods are equivalent.*

*Proof.* From the modified outcome method, we have under the logistic model for binary response

$$E\{l(W, g(\mathbf{X}))|\mathbf{X}, A=1\} = E(W|\mathbf{X}=\mathbf{x}, A=1)g(\mathbf{X}) - \log(1+e^{g(\mathbf{X})}),$$

and

$$E\{l(W, g(\mathbf{X}))|\mathbf{X}, A=0\} = E(W|\mathbf{X}=\mathbf{x}, A=0)g(\mathbf{X}) - \log(1+e^{g(\mathbf{X})}),$$

where $g(\mathbf{X}) = \beta^\top \mathbf{X}$ and $l(.)$ is the log-likelihood function. Thus,

$$\begin{aligned}
\mathcal{L}(g) &= E\{l(W, g(\mathbf{X}))\} \\
&= E_{\mathbf{X}}\Big[\frac{1}{2}E_W\{l(W, g(\mathbf{X}))|\mathbf{X}, A=1\} + \frac{1}{2}E_W\{l(W, g(\mathbf{X}))|\mathbf{X}, A=0\}\Big] \\
&= E_{\mathbf{X}}\Big[\frac{1}{2}\{E(Y|\mathbf{X}, A=1)g(\mathbf{X}) - \log(1+e^{g(\mathbf{X})})\} + \\
&\quad \frac{1}{2}\{(1-E(Y|\mathbf{X}, A=0))g(\mathbf{X}) - \log(1+e^{g(\mathbf{X})})\}\Big] \\
&= \frac{1}{2}E_{\mathbf{X}}\Big[\tau(\mathbf{X})g(\mathbf{X}) + g(\mathbf{X}) - 2\log(1+e^{g(\mathbf{X})})\Big],
\end{aligned}$$

where $\tau(\mathbf{X}) = E[Y|\mathbf{X}=\mathbf{x}, A=1] - E[Y|\mathbf{X}=\mathbf{x}, A=0]$. Therefore,

$$\frac{\partial \mathcal{L}}{\partial g} = \frac{1}{2} E_{\mathbf{X}} \Big[ \tau(\mathbf{X}) + 1 - 2 \frac{e^{g(\mathbf{X})}}{(1 + e^{g(\mathbf{X})})} \Big].$$

Thus,

$$g^*(\mathbf{x}) = \log \Big\{ \frac{1 + \tau(\mathbf{x})}{1 - \tau(\mathbf{x})} \Big\},$$

or equivalently,

$$\tau(\mathbf{x}) = \frac{e^{g^*(\mathbf{x})} - 1}{e^{g^*(\mathbf{x})} + 1}.$$

That is, the loss minimizer of $\mathcal{L}(g)$, $g^*(\mathbf{x})$, is equal to $f^*(\mathbf{x})$ in (3.6), which is the loss minimizer of $E\{Y f(\mathbf{X}) A - \log(1 + \mathbf{exp}(f(\mathbf{X})A))\}$ from the modified covariate method. □

### 3.3.3 Causal $K$-nearest-neighbor (CKNN)

A simple nonparametric method briefly discussed by Alemi et al. (2009) and also by Su et al. (2012) for estimating the PTE is to use a modified version of the $K$-nearest-neighbor (KNN) classifier (Cover and Hart, 1967).

The basic idea of the *CKNN* algorithm is that to estimate the PTE for a target subject, we may wish to weight the evidence of subjects similar to the target more heavily. Consider a subject with covariates $\mathbf{X}_\ell = \mathbf{x}$ and a neighborhood of $\mathbf{x}$, $\mathcal{S}_k(\mathbf{x})$, represented by a sphere centered at $\mathbf{x}$ containing precisely $K$ subjects, regardless of their outcome $Y$ and treatment type $A$. An estimate of the PTE is given by

$$\hat{\tau}(\mathbf{x}) = \frac{\sum_{\ell:\, \mathbf{x}_\ell \in \mathcal{S}_k(\mathbf{x})} Y_\ell A_\ell}{\sum_{\ell:\, \mathbf{x}_\ell \in \mathcal{S}_k(\mathbf{x})} A_\ell} - \frac{\sum_{\ell:\, \mathbf{x}_\ell \in \mathcal{S}_k(\mathbf{x})} Y_\ell (1 - A_\ell)}{\sum_{\ell:\, \mathbf{x}_\ell \in \mathcal{S}_k(\mathbf{x})} (1 - A_\ell)}. \tag{3.9}$$

The CKNN approach proposed in (3.9) assigns an equal weight of 1 to each of the $K$ subjects within the neighborhood $\mathcal{S}_k(\mathbf{x})$ and 0 weight to all other subjects. Alternatively, it is common to use kernel smoothing methods to assign weights that die off smoothly with the distance $||\mathbf{x}_\ell - \mathbf{x}||$ for all subjects $\ell = \{1, \ldots, L\}$. Also, notice that (3.9) is defined if at least one control and one treated subject are in the neighborhood of $\mathbf{x}$ ($K \geq 2$).

A severe limitation of this method is that the entire training data have to be stored to score new subjects, leading to expensive computations for large data sets.

### 3.3.4 Matching before randomization

As preliminarily discussed in Chapter 1, one of the key challenges in building personalized treatment learning (PTL) models is that the quantity we are trying to predict (i.e., the PTE) is unknown on a given training data set. This problem is known as *the fundamental problem of causal inference* (Holland, 1986), and results from the fact that each subject can only be exposed to one treatment condition; thus the value of the response under the alternative treatment, also called the *counterfactual response*, is not observed.

One way to think about the counterfactual responses is that they are missing values, and therefore they could be imputed to represent their uncertainty. A special type of randomized design can help in this direction. Until now we have assumed that subjects in the study were randomly allocated to two treatment arms, denoted by $A$, $A \in \{0, 1\}$, also referred as control and treatment states, respectively. An alternative method often used in medical research is to perform *matching before randomization*. Matching divides a group of $L$ subjects into $L/2$ pairs to minimize covariate differences within pairs. Then one subject in each pair is picked at random for treatment, and the other is assigned to control (Greevy et al., 2004). The essential idea is to form matched pairs of subjects who are as similar as possible in terms of their covariates $\mathbf{X}$ before the time of exposure to the intervention, but who differ in the type of treatment they receive.

Matching before randomization involves solving a so-called nonbipartite matching problem. In graphic-theoretic terminology, a graph $G = (V, E)$, consists of a set of nodes or vertices $V = \{v_i, i = 1, \ldots, L\}$ and a set of edges $E = \{[v_i, v_j], v_i \neq v_j \in V\}$. A matching $M \subset E$ in $G$ is a set of pairs of nodes $\{[v_i, v_j]\}$ between which an edge exists, such that no two edges share a common node. If the cardinality of $M$ is $|V|/2$, the matching is called complete. If we represent each subject in the study by a node, we can evaluate the "closeness" between any pair of matched subjects by

the distance between their covariates. A common measure of distance is the Mahalanobis distance, discussed in more detail in Chapter 9 (see Section 9.4.3). A traditional algorithm to create the matches is to first pair the two subjects with smallest distance, set them aside, pair the next two subjects with smallest distance, and so on. This algorithm is called *greedy* and it does not produce optimal nonbipartite matchings. An *optimal* matching algorithm minimizes the total distance summed over all the pairs. Optimal matching can be formulated as a constrained optimization problem. Let $d_{ij}$ be the distance associated with edge $[v_i, v_j] \in E$. The optimal matching problem is to find the set of

$$a_{ij} = \begin{cases} 1 & \text{if } [v_i, v_j] \in M \\ 0 & \text{if } [v_i, v_j] \notin M, \end{cases}$$

which solves the minimization problem

$$\min \sum_{[v_i, v_j] \in E} d_{ij} a_{ij} \tag{3.10}$$

subject to

$$\sum_{j:[v_i, v_j] \in E} a_{ij} = 1 \text{ for } v_i \in V. \tag{3.11}$$

The optimal nonbipartite matching problem can be solved using the *shortest augmentation path* algorithm (Derigs, 1988).

The process for estimating the individual treatment effect from a matched randomized design can be formulated as follows. Suppose a matched pair is composed of subjects $\ell_1$ and $\ell_2$ who have been assigned to treatment ($A = 1$) and control ($A = 0$), respectively. For each of these subjects we know the value of the response under the assigned treatment, but not the counterfactual response. However, as these subjects were matched, they must be similar in terms of their covariates[1]. Intuitively, we could use the observed response on one subject of each pair to fill in the "missing" counterfactual response for the other subject of the pair. That is, we can use the observed response of subject $\ell_1$ under $A = 1$ to fill

[1] Ideally, the covariates used in matching should be those that are relevant for the response variable (see Greevy et al., 2004 for details).

in the unobserved response of subject $\ell_2$ under that treatment. Similarly, we can use the observed response of subject $\ell_2$ under $A = 0$ to fill in the unobserved response of subject $\ell_1$ under control. An estimate of the subject-level treatment effect is then obtained by simply differencing the observed and (imputed) counterfactual responses between subjects of a matched pair. Subsequently, we can fit a PTL model by using these estimates as the response variable and the background covariates as predictors.

In differencing the observed and imputed values for the counterfactual response, we can distinguish between four possible outcomes: A) subjects who would respond *positively* regardless of which treatment they are assigned – that is, $Y(a) = 1$ for $a = \{0, 1\}$, B) subjects who would respond *negatively* regardless of which treatment they are assigned – that is, $Y(a) = 0$ for $a = \{0, 1\}$, C) subjects who would respond positively to treatment but negatively to the control condition – that is, $Y(1) = 1$ and $Y(0) = 0$, and D) subjects who would respond negatively to treatment but positively to the control condition – that is, $Y(1) = 0$ and $Y(0) = 1$.

A multinomial classification model could be built to predict the probabilities associated with the four possible outcomes[2].

Matching before randomization is not only useful for building PTL models. We can also obtain more efficient estimates of the average treatment effect (ATE – Equation 2.1) – in the sense of smaller variance of an unbiased estimate – using a matched design. Matched randomization significantly improves covariate balance relative to unmatched randomization (Greevy et al., 2004). Covariate balance refers to the extent to which the distributions of the covariates differ between treatment and control groups. Although the common suggestion is to control chance imbalance in the covariates using a model that allows for covariate adjustment, the adjusted estimate of the ATE is more precise when covariates are more nearly balanced (Snedecor and Cochran, 1980, p. 368).

[2] It is commonly of interest to predict the subjects for whom the treatment is effective. In this case, we could build a binary classification model for the probability of being in class C.

# 4 Uplift random forests

## 4.1 Introduction

*Uplift random forests* are a tree-based method proposed by Guelman et al. (2014c) to estimate personalized treatment effects (personalized treatment effect or PTE estimation is known as uplift modeling in the marketing literature). Tree-based models represent an intuitive approach to estimating the PTE defined in Equation (2.2), as appropriate split criteria can be designed to partition the covariate space into meaningful subgroups with heterogeneous treatment effects. The standard *random forest* (Breiman, 2001a) methodology is inherited, but the individual trees are grown using split criteria more appropriate to the problem at hand. We follow the split criteria proposed by Rzepakowski and Jaroszewicz (2012). In this chapter, we discuss the uplift random forest algorithm in detail and provide performance benchmarks for its key tuning parameters.

## 4.2 Definition of uplift random forests

Rzepakowski and Jaroszewicz (2012) propose a decision tree model to estimate PTEs. However, a major concern with their method is that it is based on a single tree. A key problem with trees is their high variance as a result of the hierarchical nature of the splitting process: the effect of an error in the top split is propagated down to all of the splits below. The instability of trees is even higher in the personalized treatment learning (PTL) case, as the treatment heterogeneity effects are usually dominated by the main effects. In this setting, perturbing the learning set can cause significant changes in the tree constructed.

We could smooth out the variance of single trees using *bagging* (short for *bootstrap aggregation*) methods (Breiman, 1996) to achieve higher stability. The idea behind bagging is to fit a sequence of noisy models (such as trees), each built on bootstrap[1] replicates of the training set, and then average the result. However, the benefit of averaging in bagging is limited by the fact that each tree is essentially built using the same candidate predictors, leading to high correlation between pair of bagged trees in the sequence (see Hastie et al., 2009). The random forest algorithm (Breiman, 2001a) further improves the variance reduction of bagging by building a sequence of de-correlated trees. This is achieved by choosing the best split at each node among a subset of predictors randomly selected at that node.

The pseudocode for the proposed uplift random forest algorithm is shown in Algorithm 1. Briefly, an ensemble of $B$ trees are grown, each built on a fraction $\nu$ of the training data[2] (where the fraction includes both treatment and control subjects). The sampling, motivated by Friedman (2002), incorporates randomness as an integral part of the fitting procedure. This not only reduces the correlation between the trees in the sequence, but also reduces the computing time by the fraction $\nu$. A typical value for $\nu$ can be $1/2$, although for large data sets, it can be substantially smaller. The tree-growing process involves selecting $n \leq p$ covariates at random as candidates for splitting. This adds an additional layer of randomness, which further reduces the correlation between trees, and hence reduces the variance of the ensemble. The split rule is based on a measure of distributional divergence, as defined in Rzepakowski and Jaroszewicz (2012), discussed below. The individual trees are grown to maximal depth (i.e., no pruning is done). The prediction from each individual tree is induced by its terminal nodes. The estimated PTE for a data point $\mathbf{X} = \mathbf{x}$ is obtained by averaging the predictions of the individual trees in the ensemble.

The split criteria are based on the objective of maximizing the distance in the class distributions of the response $Y$ between the treatment and

[1] Given a dataset with $L$ instances, a bootstrap sample consists in drawing a sample with replacement of size $L$ from the data (Efron and Tibshirani, 1986).

[2] In the standard random forest algorithm, bootstrap samples of the training data are drawn before fitting each tree. Our motivation for sampling a fraction of the data instead, was to reduce computational time on large data sets.

control groups. To that end, it is sensible to borrow the concept of distributional divergence from information theory. In particular, if we let $P\{Y(1)\}$ and $P\{Y(0)\}$ be the class probability distributions over the response variable $Y$ for the treatment and control, respectively, then the *Kullback-Leibler distance* (*KL*) or *relative entropy* (Cover and Thomas, 1991, p. 19) between the two distributions is given by

$$KL\Big(P\{Y(1)\}||P\{Y(0)\}\Big) = \sum_{i\in\{0,1\}} P\{Y(1)=i\}\,\log\frac{P\{Y(1)=i\}}{P\{Y(0)=i\}}, \quad (4.1)$$

where the logarithm is to base[3] $e$. The Kullback-Leibler distance is always nonnegative and it is zero if and only if $P\{Y(1)\} = P\{Y(0)\}$. Since the *KL* distance is nonsymmetric, it is not a true distance measure. However, it is frequently useful to think of *KL* as a measure of divergence between distributions.

For any node, suppose there is a candidate split $\Omega$ which divides it into two child nodes, $v_{\mathrm{L}}$ and $v_{\mathrm{R}}$, denoting the left and right node respectively. Further, let $L$ be the total number of subjects in the parent node and suppose $L_{v_{\mathrm{L}}}$ and $L_{v_{\mathrm{R}}}$ represent the number of subjects that go into $v_{\mathrm{L}}$ and $v_{\mathrm{R}}$, respectively. Conditional on a split $\Omega$, distributional divergence can be expressed as the *KL* distance within each child node, weighted by the proportion of subjects in each node

$$KL\Big(P\{Y(1)\}||P\{Y(0)\}\Big|\Omega\Big) = \frac{1}{L}\sum_{j\in\{v_{\mathrm{L}},v_{\mathrm{R}}\}} L_j KL\Big(P\{Y(1)\}||P\{Y(0)\}\Big|j\Big). \quad (4.2)$$

Now, define $KL_{\mathrm{gain}}$ as the increase in the *KL* distance from a split $\Omega$, relative to the *KL* distance in the parent node:

$$KL_{\mathrm{gain}}(\Omega) = KL\Big(P\{Y(1)\}||P\{Y(0)\}\Big|\Omega\Big) - KL\Big(P\{Y(1)\}||P\{Y(0)\}\Big). \quad (4.3)$$

[3] In the information theory literature, the logarithm to base two is used in (4.1) if information is measured in units of *bits*, or to base $e$ if information is measured in *nats*.

The final splitting rule adds a normalization factor to (4.3). This factor attempts to penalize splits with unbalanced proportions of subjects associated with each child node, as well as splits that result in unbalanced treatment/control proportion in each child node (since such splits are not independent of the group assignment). The final split criterion is then given by

$$KL_{\text{ratio}}(\Omega) = \frac{KL_{\text{gain}}(\Omega)}{KL_{\text{norm}}(\Omega)}, \tag{4.4}$$

where

$$KL_{\text{norm}}(\Omega) = H\Big(\frac{L(1)}{L}, \frac{L(0)}{L}\Big) KL\Big(P\{\Omega(1)\}||P\{\Omega(0)\}\Big) + \\ \frac{L(1)}{L} H\Big(P\{\Omega(1)\}\Big) + \frac{L(0)}{L} H\Big(P\{\Omega(0)\}\Big). \tag{4.5}$$

$L(A)$ in (4.5) denotes the number of subjects in treatment $A \in \{0, 1\}$, $P\{\Omega(A)\}$ represents the probability distribution over the split outcomes $\{v_{\text{L}}, v_{\text{R}}\}$ for subjects with treatment $A$, and $H(.)$ is the *entropy* function, defined by $H(P\{\Omega(A)\}) = -P\{\Omega(A) = v_{\text{L}}\}\log(P\{\Omega(A) = v_{\text{L}}\}) - P\{\Omega(A) = v_{\text{R}}\}\log(P\{\Omega(A) = v_{\text{R}}\})$ and $H(\frac{L(1)}{L}, \frac{L(0)}{L}) = -\frac{L(1)}{L}\log(\frac{L(1)}{L}) - \frac{L(0)}{L}\log(\frac{L(0)}{L})$.

The last two terms in (4.5) penalize splits with a large number of outcomes, by means of the sum of the entropies of the split outcomes in treatment and control groups weighted by the proportion of training cases in each group. This may be seen as unnecessary, given that our uplift tree is constructed on binary splits (in contrast to Rzepakowski and Jaroszewicz, 2012), which have important advantages over multiway splits. However, as discussed in Quinlan (1993) and Mingers (1989), this normalization factor appears to be advantageous even when all splits are binary but differ in the proportion of training cases associated with the two outcomes. The first term penalizes uneven splits, as measured by the divergence in the distribution of the split outcomes between the groups. This term is multiplied by the entropy of the proportion of subjects in treatment and control groups. This explicitly imposes a smaller penalty when there are not enough data in one of these groups.

A problem with the $KL_{\text{ratio}}$ is that extremely low values of the $KL_{\text{norm}}$ may favor splits despite their low $KL_{\text{gain}}$ values. To avoid this, the $KL_{\text{ratio}}$

criterion selects splits that maximize $KL_{\text{ratio}}$, subject to the constraint that $KL_{\text{gain}}$ must be at least as great as the average $KL_{\text{gain}}$ value over all splits considered.

**Algorithm 1** Uplift random forest

1: **for** $b = 1$ to $B$ **do**
2: Sample a fraction $\nu$ of the training observations $L$ without replacement
3: Grow an uplift decision tree $UT_b$ on the sampled data:
4: **for** each terminal node **do**
5: **repeat**
6: Select $n$ covariates at random from the $p$ covariates
7: Select the best variable/split-point among the $n$ covariates based on $KL_{\text{ratio}}$
8: Split the node into two branches
9: **until** a minimum node size $l_{\text{min}}$ is reached
10: **end for**
11: **end for**
12: Output the ensemble of uplift trees $UT_b;\ b = \{1, \ldots, B\}$
13: The predicted PTE for a new data point $\mathbf{x}$ is obtained by averaging the predictions of the individual trees in the ensemble: $\hat{\tau}(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} UT_b(\mathbf{x})$

Although our exposition is based on the Kullback-Leibler distance as a measure of distributional divergence, other split criteria can be used for selecting the best split at each node. Alternative measures of distributional divergence may include squared Euclidean distance (4.6), Chi-squared divergence (4.7), and $L1$-norm divergence (4.8):

$$E\Big(P\{Y(1)\}||P\{Y(0)\}\Big) = \sum_{i\in\{0,1\}} \Big(P\{Y(1)=i\} - P\{Y(0)=i\}\Big)^2, \quad (4.6)$$

$$\chi^2\Big(P\{Y(1)\}||P\{Y(0)\}\Big) = \sum_{i\in\{0,1\}} \frac{\Big(P\{Y(1)=i\} - P\{Y(0)=i\}\Big)^2}{P\{Y(1)=i\}}, \quad (4.7)$$

$$L1\Big(P\{Y(1)\}||P\{Y(0)\}\Big) = \sum_{i\in\{0,1\}} \Big|P\{Y(1)=i\} - P\{Y(0)=i\}\Big|. \qquad (4.8)$$

Substituting the $KL$ distance above by these functions, we obtain, respectively, the following split criteria: $E_{\text{ratio}}$, $\chi^2_{\text{ratio}}$, and $L1_{\text{ratio}}$. Figure 4.1 shows the functional form for the various split criteria. All look similar in shape, except for $L1$ which is not differentiable. $KL$ and the $\chi^2$ distance are notably more sensitive to changes in the node probabilities than $E$ and $L1$. In addition, only the last two are symmetric. A comprehensive discussion of these measures (among others) can be found in Lee (1999).

## 4.3 Input variable importance

An additional piece of information, generally desirable in a PTL model, is a measure of the relative importance of the input variables in predicting the PTE. In conventional decision trees, Breiman et al. (1984) proposed the following measure as an approximation of the relative influence of a predictor $X_j$, $j = \{1, \ldots, p\}$:

$$\hat{I}_j = \sum_{\substack{\text{all splits}\\ \text{on } X_j}} \hat{V}_s, \qquad (4.9)$$

where $\hat{V}_s$ is the empirical improvement in the split-criterion as a result of using $X_j$ as a splitting variable at the non-terminal node $s$. That is, the measure of importance given to $X_j$ is the sum of the values given by the split-criterion produced over all internal nodes for which it was chosen as the splitting variable. For the standard random forest, this *relative influence* measure is naturally extended by averaging (4.9) over the collection of trees[4]. We have implemented variable importance for the uplift random forest algorithm in exactly the same way.

[4] Random forests also use an alternative variable influence measure based on *out-of-bag* (OOB) samples (see Breiman, 2001a).

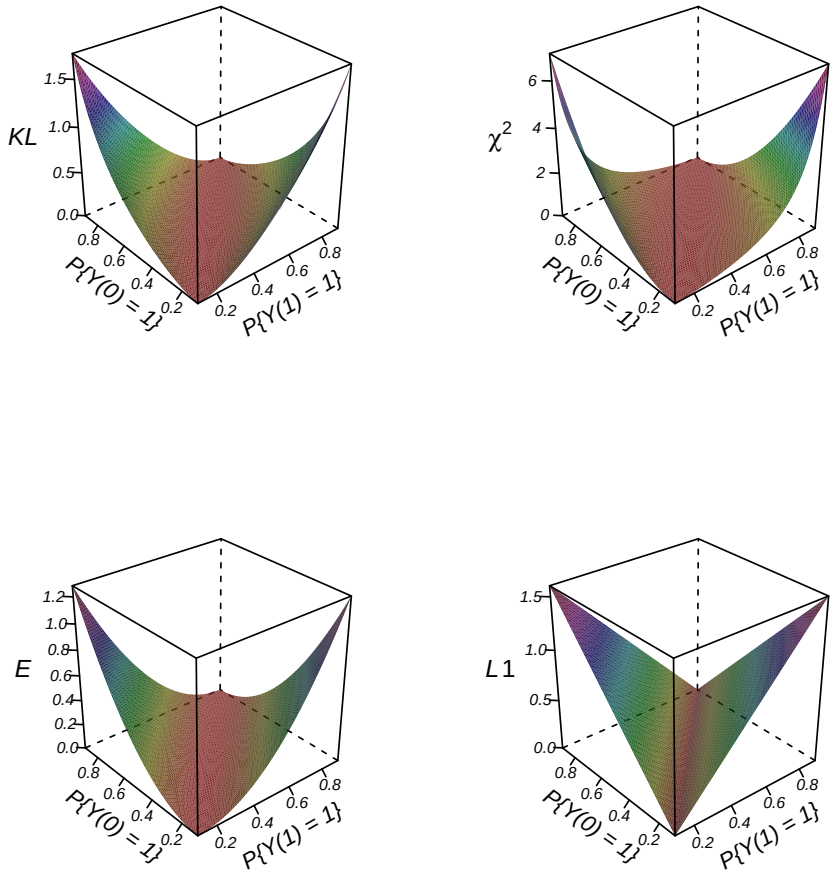


Figure 4.1: Alternative split criteria used by uplift random forests as a function of $P\{Y(A) = 1\}$, $A \in \{0, 1\}$.

## 4.4 Performance of uplift random forests

In this section we examine the performance of uplift random forests under different values of its key tuning parameters. For this discussion, we focus on the simulation setting to be described in Chapter 6. Performance is measured by the Spearman's rank correlation coefficient between the estimated PTE $\hat{\tau}(\mathbf{x})$ derived from each uplift random forest fit and the "true" simulated treatment effect in an independent test set. In all cases, training and test sets are composed of $L = 200$ and $L = 10,000$, respectively. The results are based on 100 repetitions of the simulation.

### 4.4.1 Number of covariates

One of the parameters[5] in Algorithm 1 is the number $n$ of covariates selected at random as candidates for splitting. For the standard random forest, the inventors recommend using[6] $n = \sqrt{p}$. However, as discussed by Hastie et al. (2009), when the number of variables is large, but the fraction of relevant variables is small, random forests are likely to perform poorly with small $n$. This is because at each split, the chance of selecting the relevant variables decreases as the number of noise variables increases. This result can also be confirmed for the uplift random forests case in the context of PTL.

To support this claim, we have simulated data according to model (6.1), with increasing values of $p$ (while the other parameters were fixed at $\eta_j = (-1)^{(j+1)} I(3 \leq j \leq 10)/2$, $\rho = 0$, and $\sigma_0 = \sqrt{2}$). This models a situation with a fixed number of four covariates that interact with the treatment relative to an increasing number of irrelevant covariates. In each case, we tested using $n = \sqrt{p}$ versus $n = p/3$. Figure 4.2 shows the performance results. Notice that with a small number of irrelevant variables, performance is very similar with both methods for choosing $n$. However, as the number of total variables increases relative to the number of treatment heterogeneity variables, $n = p/3$ outperforms the alternative.

[5] Technically called a hyperparameter, as it is not estimated as part of the fitting procedure, but fixed in advance by the user.

[6] More specifically, $n = \text{floor}(\sqrt{p})$ is commonly used in software implementations of random forests, where floor is a function that maps $\sqrt{p}$ to the largest integer not greater that $\sqrt{p}$.

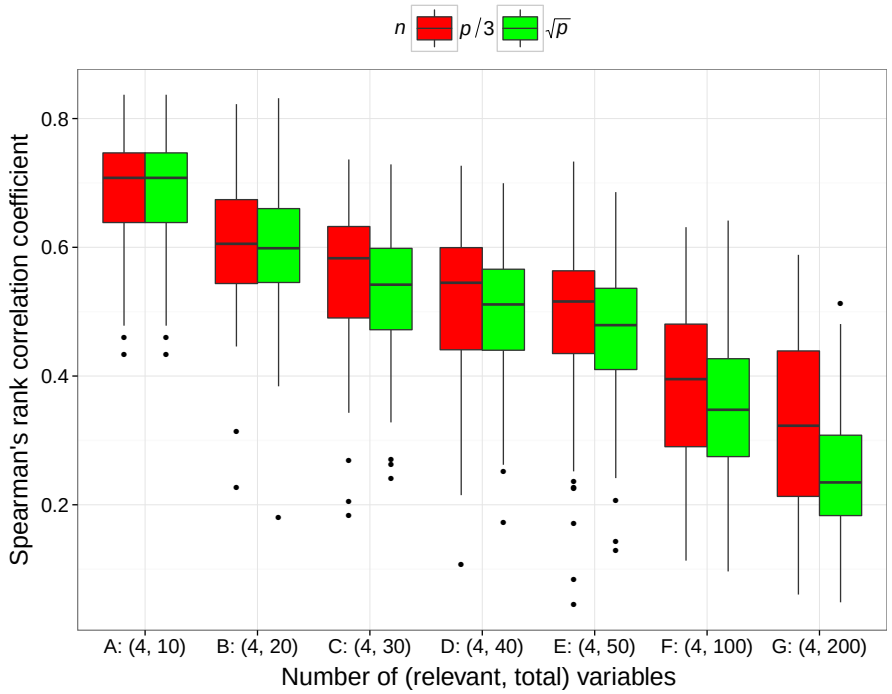


Figure 4.2: Model performance from fitting uplift random forests with $n = \sqrt{p}$ versus $n = p/3$ for an increasing ratio of total variables to relevant variables. A relevant variable is one that interacts with the treatment. Performance is measured by the Spearman's rank correlation coefficient between the estimated personalized treatment effect (PTE) and the "true" PTE. The dots outside the boxplots represent outliers. We used the "1.5 rule" for determining if a data point is an outlier: less than $Q1 - 1.5 \times (Q3 - Q1)$ or greater than $Q3 + 1.5 \times (Q3 - Q1)$, where $Q1$ and $Q3$ represent the first and third quartiles, respectively.

### 4.4.2 Split criteria comparison

We provide here a performance comparison among the split criteria discussed in Section 4.2. For that purpose, we fitted uplift random forests based on each of the four split criteria under increasing levels of noise in the data. Specifically, we tested $\sigma_0 = \sqrt{2},\ 2\sqrt{2},\ 3\sqrt{2}, \text{and } 4\sqrt{2}$ (while the other simulation parameters were fixed at $\eta_j = (-1)^{(j+1)} I(3 \leq j \leq 10)/2$, $\rho = 0$, and $p = 20$). The performance results are shown in Figure 4.3. Chi-squared divergence performs best or next to the best in all settings, closely followed by Kullback-Leibler distance, Euclidean distance, and $L1$-norm. However, the results illustrated here are limited to a particular simulation setting. We encourage testing different split criteria in empirical applications.

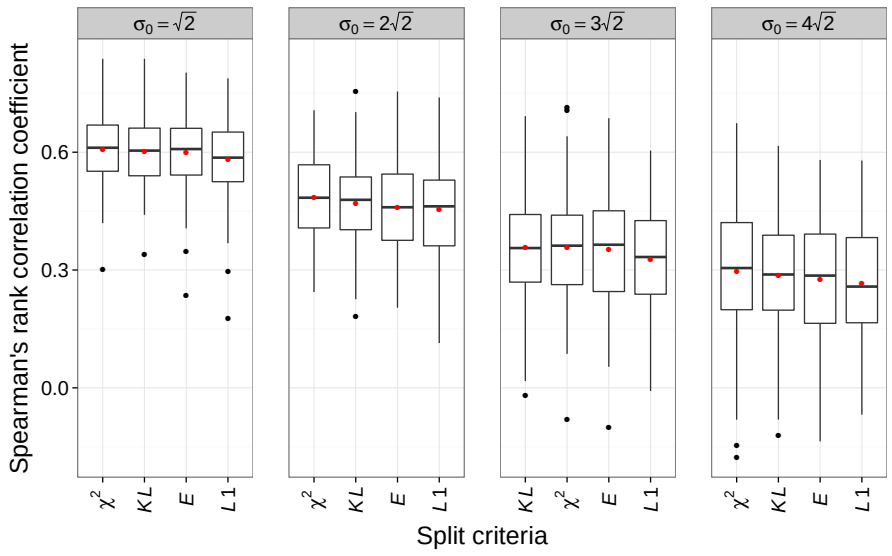


Figure 4.3: Model performance comparison among uplift split criteria for increasing values of $\sigma_0$.

### 4.4.3 Uplift random forests and overfitting

A potential concern with uplift random forests is that, similarly to the standard random forest algorithm, trees are grown to maximal depth (i.e., no pruning is done). While this helps with regard to bias, there is the usual tradeoff with variance. For the standard algorithm, Segal (2004) demonstrates small gains in performance by controlling the depths of the individual trees grown in the forest. Also, Hastie et al. (2009) conclude that using full-grown trees seldom costs much, and results in one less tuning parameter.

We fitted uplift random forests using trees grown to maximal depth (`md`) against controlling the depth level of the individual trees up to 2 levels (`id2`) and 3 levels (`id3`). In each case, we tested performance under increasing values of the number of uplift trees $B$ and two levels of noise, $\sigma_0 = \sqrt{2}$ (Low), and $\sigma_0 = 4\sqrt{2}$ (High) (and fixed $\eta_j = (-1)^{(j+1)} I(3 \leq j \leq 10)/2$, $\rho = 0$, and $p = 20$). Figure 4.4 shows the model performance values averaged over the 100 repetitions of the simulation.

Under the low noise scenario, the results show that uplift trees grown to maximal depth do not overfit. This may not be entirely surprising, as it is hard to overfit these data when the levels of noise are low. However, in noisy settings, gains can be obtained by controlling the depth of the individual uplift trees. Notice that in this case, trees grown to maximal depth perform worse than constraining the depth level. Lastly, note that performance tends to increase with the number of uplift trees in the forest, as more trees add stability to the procedure. However, at some point, additional trees start to become "redundant" and performance ceases to improve as a result.

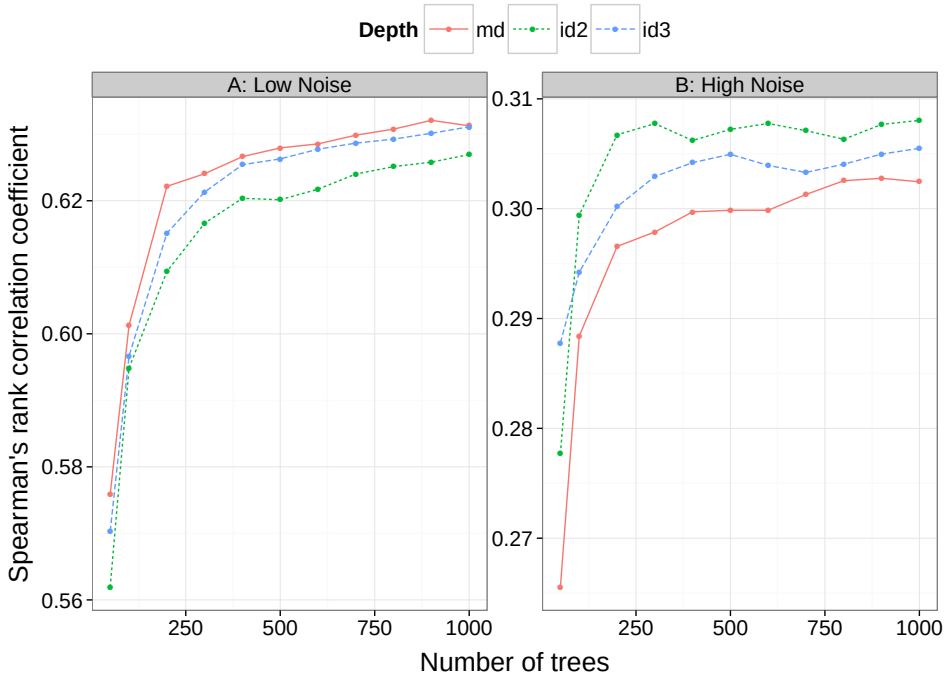


Figure 4.4: Impact of controlling the depth level of the individual uplift trees on model performance under two levels of noise: $\sigma_0 = \sqrt{2}$ (Low) and $\sigma_0 = 4\sqrt{2}$ (High).

# 5 Causal conditional inference forests

## 5.1 Introduction

In this chapter, we introduce a tree-based method to estimate PTEs with important enhancements over the *uplift random forest* algorithm described in Chapter 4. There are two fundamental aspects in which uplift random forests could be improved: overfitting and the selection bias towards covariates with many possible splits. The development of the framework introduced here to tackle these issues was motivated by the *unbiased recursive partitioning* method proposed by Hothorn et al. (2006). Following this framework, we have considerably improved the generalization performance of uplift random forests by solving both the overfitting and biased variable selection problems. The key to the solution is separating the variable selection and the splitting procedure, coupled with a statistically motivated and computationally efficient stopping criterion based on the theory of permutation tests developed by Strasser and Weber (1999).

## 5.2 Definition of causal conditional inference forests

In the uplift random forest algorithm, the individual trees are grown to maximal depth. While this helps to reduce bias, there is the usual tradeoff with variance. Maximal-depth trees could be highly unstable and this may overemphasize learning patterns and noise in the data which

may not recur in future samples. This problem, known as overfitting, can be exacerbated in the context of personalized treatment learning (PTL) models. In these models, the variability in the response from the treatment heterogeneity effects tends to be small relative to the variability in the response from the main effects. If the fitted model is not able to distinguish well between the relative strength of these two effects and the levels of noise in the data are relatively high, this may easily translate into overfitting problems (see Section 4.4.3). In conventional decision trees (Breiman et al., 1984; Quinlan, 1993), overfitting is solved by a pruning procedure. This consists in traversing the tree bottom up and testing for each (non-terminal) node, whether collapsing the subtree rooted at that node into a single leaf would improve the model's generalization performance. Tree-based methods proposed in the literature to estimate PTEs (Rzepakowski and Jaroszewicz, 2012; Su et al., 2012; Radcliffe and Surry, 2011) use some sort of pruning. However, the pruning procedures used by these methods are all *ad hoc* and lack a theoretical foundation.

Besides the overfitting problem, the second concern is the bias of variable selection towards covariates with many possible splits or missing values. This problem is also present in conventional decision trees, such as CART (Breiman et al., 1984) and C4.5 (Quinlan, 1993), and results from the maximization of the split criterion over all possible splits simultaneously (Kass, 1980; Breiman et al., 1984, p. 42).

A solution to both the overfitting and biased variable selection problems can be obtained by making the selection of the variable to split a node independent of the split criterion (Hothorn et al., 2006). Additionally, a statistically motivated and computationally efficient stopping criterion based on the theory of permutation tests developed by Strasser and Weber (1999) is incorporated. The result is an improvement in predictive performance relative to the uplift random forest method. In Chapter 6, we conduct an extensive numerical study to support this claim.

The pseudocode of the proposed *causal conditional inference forest* algorithm is shown in Algorithm 2. The most relevant aspects to discuss are steps 7-12. Specifically, for each terminal node in the tree we test the global null hypothesis of no interaction effect between the treatment $A$ and *any* of the $n$ covariates selected at random from the set of $p$ covariates. The global hypothesis of no interaction is formulated in terms of $n$ partial

hypotheses $H_0^j : E[W|X_j] = E[W],\ j = \{1, \ldots, n\}$, with the global null hypothesis $H_0 = \cap_{j=1}^{n} H_0^j$, where $W$ is defined as in the modified outcome method discussed in Section 3.3.2. Thus, a conditional independence test of $W$ and $X_j$ has a causal interpretation for the treatment effect[1] for subjects with baseline covariate $X_j$. Multiplicity in testing can be handled via Bonferroni-adjusted $P$ values or alternative adjustment procedures (Wright, 1992; Shaffer, 1995; Benjamini and Hochberg, 1995). When we are not able to reject $H_0$ at a prespecified significance level $\alpha$, we stop the splitting process at that node. Otherwise, we select the $j^*$th covariate $X_{j*}$ with the smallest adjusted $P$ value. The algorithm then induces a partition $\Omega^*$ of the covariate $X_{j*}$ into two disjoint sets $\mathcal{M} \subset X_{j*}$ and $X_{j*} \setminus \mathcal{M}$ based on the split criterion discussed below. This statistical approach prevents overfitting, without requiring any form of pruning or cross-validation.

One approach to measuring the independence between $W$ and $X_j$ would be to use a classical statistical test, such as a Pearson's chi-squared. However, the assumed distribution in these tests is only a valid approximation to the actual distribution in the large-sample case, and this does not likely hold near the leaves of the decision tree. Instead, we measure independence based on the theoretical framework of permutation tests (Hothorn et al., 2006), which is admissible for arbitrary sample sizes. Strasser and Weber (1999) developed a comprehensive theory based on a general functional form of multivariate linear statistics appropriate for arbitrary independence problems. Specifically, to test the null hypothesis of independence between $W$ and $X_j,\ j = \{1, \ldots, n\}$, we use linear statistics of the form

$$\mathcal{T}_j = \text{vec}\left( \sum_{\ell=1}^{L} g(X_{j\ell}) h(W_\ell, (W_1, \ldots, W_L))^\top \right) \in \mathbb{R}^{u_j v \times 1} \tag{5.1}$$

where $g : X_j \to \mathbb{R}^{u_j \times 1}$ is a transformation of the covariate $X_j$ and $h : W \to \mathbb{R}^{v \times 1}$ is called the *influence function*. The "vec" operator transforms the $u_j \times v$ matrix into a $u_j v \times 1$ column vector. The distribution of $\mathcal{T}_j$ under the null hypothesis can be obtained by fixing $X_{j1}, \ldots, X_{jL}$ and conditioning on all possible permutations $S$ of the responses $W_1, \ldots, W_L$. A univariate

[1] As discussed in Section 3.3.2, the modified covariate method requires $P(A = 1) = P(A = 0) = 1/2$. This is incorporated in line 2 of Algorithm 2.

test statistic $c$ is then obtained by standardizing $\mathcal{T}_j \in \mathbb{R}^{u_j v \times 1}$ based on its conditional expectations $\mu_j \in \mathbb{R}^{u_j v \times 1}$ and covariance $\Sigma_j \in \mathbb{R}^{u_j v \times u_j v}$, as derived by Strasser and Weber (1999). A common choice is the maximum of the absolute values of the standardized linear statistic

$$c_{\max}(\mathcal{T}, \mu, \Sigma) = \max \left| \frac{\mathcal{T} - \mu}{\operatorname{diag}(\Sigma)^{1/2}} \right|, \tag{5.2}$$

or a quadratic form

$$c_{\text{quad}}(\mathcal{T}, \mu, \Sigma) = (\mathcal{T} - \mu)\Sigma^{+}(\mathcal{T} - \mu)^{\top}, \tag{5.3}$$

where $\Sigma^{+}$ is the Moore-Penrose inverse of $\Sigma$.

In step 11 of Algorithm 2, we select the covariate $X_{j*}$ with smallest adjusted $P$ value. The $P$ value $P_j$ is given by the number of permutations $s \in S$ of the data with corresponding test statistic exceeding the observed test statistic $t \in \mathbb{R}^{u_j v \times 1}$. That is,

$$P_j = \mathbb{P}(c(\mathcal{T}_j, \mu_j, \Sigma_j) \geq c(t_j, \mu_j, \Sigma_j) | S).$$

For moderate to large samples sizes, it might not be possible to obtain the exact distribution (calculated exhaustively) of the test statistic. However, we can approximate the exact distribution by computing the test statistic from a random sample of the set of all permutations $S$. In addition, Strasser and Weber (1999) showed that the asymptotic distribution of the test statistic given by (5.2) tends to multivariate normal with parameters $\mu$ and $\Sigma$ as $L \to \infty$. The test statistic (5.3) follows an asymptotic chi-squared distribution with degrees of freedom given by the rank of $\Sigma$. Therefore, asymptotic $P$ values can be computed for these test statistics.

Once we select the covariate $X_{j*}$ to split, we next use a split criterion which explicitly attempts to find subgroups with heterogeneous treatment effects. Specifically, we use the following measure proposed by Su et al. (2009), also implemented later by Radcliffe and Surry (2011) for assessing the PTE from a split $\Omega$:

$$G^2(\Omega) = \frac{(L-4)\Big\{\Big(\bar{Y}_{v_\mathrm{L}}(1) - \bar{Y}_{v_\mathrm{L}}(0)\Big) - \Big(\bar{Y}_{v_\mathrm{R}}(1) - \bar{Y}_{v_\mathrm{R}}(0)\Big)\Big\}^2}{\hat{\sigma}^2\Big\{1/L_{v_\mathrm{L}}(1) + 1/L_{v_\mathrm{L}}(0) + 1/L_{v_\mathrm{R}}(1) + 1/L_{v_\mathrm{R}}(0)\Big\}} \tag{5.4}$$

where $v_\mathrm{L}$ and $v_\mathrm{R}$ denote the left and right child nodes from a candidate split $\Omega$, respectively, $L_{i\in\{v_\mathrm{L},v_\mathrm{R}\}}(A)$ denotes the number of observations in child node $i$ exposed to treatment $A \in \{0,1\}$, and

$$\bar{Y}_{i\in\{v_\mathrm{L},v_\mathrm{R}\}}(1) = \frac{\sum_{\forall \ell \in i} Y_\ell A_\ell}{\sum_{\forall \ell \in i} A_\ell}, \tag{5.5}$$

$$\bar{Y}_{i\in\{v_\mathrm{L},v_\mathrm{R}\}}(0) = \frac{\sum_{\forall \ell \in i} Y_\ell (1-A_\ell)}{\sum_{\forall \ell \in i} (1-A_\ell)}, \tag{5.6}$$

$$\hat{\sigma}^2 = \sum_{A\in\{0,1\}} \sum_{i\in\{v_\mathrm{L},v_\mathrm{R}\}} L_i(A)\bar{Y}_i(A)(1-\bar{Y}_i(A)). \tag{5.7}$$

The best split is given by $G^2(\Omega^*) = \max_\Omega G^2(\Omega)$ – that is, the split that maximizes the criterion $G^2(\Omega)$ among all permissible splits. It can easily be seen (Su et al., 2009) that the split criterion given in (5.4) is equivalent to a chi-squared test of the interaction effect between the treatment and the covariate $X_{j*}$ dichotomized at the value given by the split $\Omega$.

## 5.3 Additional considerations

### 5.3.1 Categorical covariates

A binary split on a categorical covariate $X$ with $q$ distinct unordered categories $C = \{c_1, \ldots, c_q\}$ has $2^{q-1} - 1$ possible partitions. This would result in a computational burden for large $q$. For standard decision trees Breiman et al. (1984) and Ripley (1996) showed that for strictly concave split criteria, it is possible to obtain a considerable shortcut in the computations. Simply order the covariate according to the proportion falling in outcome class 1 and then treat this predictor as ordinal. In building a causal conditional inference tree, a similar shortcut can be obtained by first ordering the categorical covariate according to the

estimated treatment effect within each category and then treating the covariate as ordinal. Su et al. (2009) showed that this gives an optimal split in terms of the criterion defined in (5.4) among all $2^{q-1} - 1$ possible splits.

**Algorithm 2** Causal conditional inference forest

**for** $b = 1$ to $B$ **do**
  Draw a sample with replacement from the training observations $L$ such that $P(A = 1) = P(A = 0) = 1/2$
  Grow a conditional causal inference tree $CCIT_b$ to the sampled data:
  **for** each terminal node **do**
    **repeat**
      Select $n$ covariates at random from the $p$ covariates
      Test the global null hypothesis of no interaction effect between the treatment $A$ and any of the $n$ covariates (i.e., $H_0 = \cap_{j=1}^{n} H_0^j$, where $H_0^j : E[W|X_j] = E[W]$) at a level of significance $\alpha$ based on a permutation test
      **if** the null hypothesis $H_0$ cannot be rejected **then**
        **Stop**
      **else**
        Select the $j^*$th covariate $X_{j*}$ with the strongest interaction effect (i.e., the one with the smallest adjusted $P$ value)
        Choose a partition $\Omega^*$ of the covariate $X_{j*}$ into two disjoint sets $\mathcal{M} \subset X_{j*}$ and $X_{j*} \setminus \mathcal{M}$ based on the $G^2(\Omega)$ split criterion
      **end if**
    **until** a minimum node size $l_{\min}$ is reached
  **end for**
**end for**
Output the ensemble of causal conditional inference trees $CCIT_b$; $b = \{1, \ldots, B\}$
The predicted PTE for a new data point $\mathbf{x}$, is obtained by averaging the predictions of the individual trees in the ensemble: $\hat{\tau}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} CCIT_b(\mathbf{x})$

### 5.3.2 Stopping criteria

In Section 5.2 we discussed a statistically motivated stopping criterion for causal conditional inference trees based on the theory of permutation tests developed by Strasser and Weber (1999). Additionally, in our **uplift** R package (Guelman, 2014) implementation, a split at any given node in the tree is attempted if any of the following conditions are met: (i) the total number of observations in the node is higher than a minimum preset size, (ii) the number of control and treated observations in the node is higher than a minimum preset size, (iii) the depth level of the tree is less than a maximum threshold.

### 5.3.3 Observational studies

Both uplift random forests and causal conditional inference forests are designed to estimate PTEs in the context of randomized designs. Under randomization, the assignment of subjects to treatment and control groups is independent of their baseline covariates. As the sample size grows, random assignment tends to balance covariates, in the sense that both groups have similar distributions of covariates. Observational studies differ from experiments in that randomization is not used to assign treatments. In this setting, subjects exposed to different treatment conditions are not directly comparable. In the absence of randomization, the propensity score (Rosenbaum and Rubin, 1983) is a device for constructing a matched sample of control and treated subjects that balance many observed covariates. Once a matched sample is constructed, the proposed tree-based methods to estimate PTEs can be implemented on the matched subjects. However, conclusions about the individual effect of treatments will only be valid in the region of the covariate space in which control and treated subjects were matched. The propensity score and matching algorithms in the context of observational data are discussed in detail in Chapter 9.

### 5.3.4 Multiple treatments

We have only considered the case of binary treatments. It would be worthwhile to examine the extent to which the causal conditional inference

forests method can be extended to multi-category or continuous treatment settings. For the former, one option would be to do a series of pairwise comparisons, but this may not be optimal in the sense of identifying the best treatment. It is important to note that many well-known classical tests (e.g., Pearson's chi-squared, Cochran-Mantel-Haenszel, Wilcoxon-Mann-Whitney) can be formulated from (5.1) by choosing the appropriate transformation $g$, influence function $h$ and test statistic $c$ to map the linear statistic $\mathcal{T}$ into the real line. This sheds light on the possible extensions of the proposed method to response variables measured in arbitrary scales and multi-category or continuous treatment settings.

# 6 Simulations

In this chapter, we conduct a numerical study for the purpose of assessing the finite sample performance of the analytical methods introduced in Chapters 3, 4, and 5. Most of these methods require specialized software for implementation. We have developed a software package in R named **uplift** (Guelman, 2014) that implements a variety of algorithms for building and testing personalized treatment learning (PTL) models. Currently, the following methods are implemented: uplift random forests (`upliftRF`), causal conditional inference forests (`ccif`), causal $K$-nearest-neighbor (`cknn`), modified covariate method (`mcm`), and modified outcome method (`mom`). The **uplift** package functionality is discussed in detail in Chapter 11. We also used the package **FindIt**, which implements the L2-SVM method (`l2svm`) and was developed by the authors of the method (Imai and Ratkovic, 2013). Finally, the difference score (`dsm`) and interaction (`int`) methods can be implemented straightforwardly using readily available software.

Our simulation framework is based on the one described in Tian et al. (2014), but with a few modifications. We evaluate the performance of the aforementioned methods in eight simulation settings, by varying (i) the relative strength of the main effects relative to the treatment heterogeneity effects, (ii) the degree of correlation among the covariates, and (iii) the noise levels in the response.

We generated $L$ independent binary samples from the regression model

$$Y = I\left(\Big[\sum_{j=1}^{p}\eta_j X_j + \sum_{j=1}^{p}\delta_j X_j A_j^* + \epsilon\Big] \geq 0\right), \tag{6.1}$$

where the covariates $(X_1, \ldots, X_p)$ follow a mean-zero multivariate normal distribution with covariance matrix $(1-\rho)\mathbf{I}_p + \rho\mathbf{1}^\top\mathbf{1}$, $A_\ell^* = 2A_\ell - 1 \in$

$\{-1, 1\}$ was generated with equal probability at random, $A_\ell \in \{0, 1\}$, and $\epsilon \sim N(0, \sigma_0^2)$. We let $L = 200$ (size of the training set), $p = 20$, and $(\delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \ldots, \delta_p) = (1/2, -1/2, 1/2, -1/2, 0, \ldots, 0)$.

Table 6.1 shows the simulation scenarios. The first four scenarios model a situation in which the variability in the response from the main effects is twice as big as that from the treatment heterogeneity effects, whereas in the last four scenarios the variability in the response from the main effects is four times as big as that from the treatment heterogeneity effects. Each of these scenarios were tested under zero and moderate correlation among the covariates ($\rho = 0$ and $\rho = 0.5$), and two levels of noise ($\sigma_0 = \sqrt{2}$ and $\sigma_0 = 2\sqrt{2}$).

Table 6.1: Simulation scenarios

| Scenario | $\eta_j$ | $\rho$ | $\sigma_0$ |
|---|---|---|---|
| 1 | $(-1)^{(j+1)} I(3 \leq j \leq 10)/2$ | 0 | $\sqrt{2}$ |
| 2 | $(-1)^{(j+1)} I(3 \leq j \leq 10)/2$ | 0 | $2\sqrt{2}$ |
| 3 | $(-1)^{(j+1)} I(3 \leq j \leq 10)/2$ | 0.5 | $\sqrt{2}$ |
| 4 | $(-1)^{(j+1)} I(3 \leq j \leq 10)/2$ | 0.5 | $2\sqrt{2}$ |
| 5 | $(-1)^{(j+1)} I(3 \leq j \leq 10)$ | 0 | $\sqrt{2}$ |
| 6 | $(-1)^{(j+1)} I(3 \leq j \leq 10)$ | 0 | $2\sqrt{2}$ |
| 7 | $(-1)^{(j+1)} I(3 \leq j \leq 10)$ | 0.5 | $\sqrt{2}$ |
| 8 | $(-1)^{(j+1)} I(3 \leq j \leq 10)$ | 0.5 | $2\sqrt{2}$ |

*Note.* This table displays the numerical settings considered in the simulations. Each scenario is parametrized by the strength of the main effects, $\eta_j$, $j = \{1, \ldots, p\}$, the correlation among the covariates, $\rho$, and the magnitude of the noise, $\sigma_0$.

The key benefit of simulations in the context of PTEs is that the "true" treatment effect is known for each subject, a value which is not observed in empirical data. The performance of the analytical methods was measured using the *Spearman's rank correlation* coefficient between the estimated treatment effect $\hat{\tau}(X)$ derived from each model, and the "true" treatment effect

$$\begin{aligned}\tau(\mathbf{X}) &= E[Y(1) - Y(0)|\mathbf{X}] \\ &= P\left(\sum_{j=1}^{p}(\eta_j + \delta_j)X_j \leq \epsilon\right) - P\left(\sum_{j=1}^{p}(\eta_j - \delta_j)X_j \leq \epsilon\right) \\ &= F\left(\sum_{j=1}^{p}(\eta_j + \delta_j)X_j\right) - F\left(\sum_{j=1}^{p}(\eta_j - \delta_j)X_j\right), \qquad (6.2)\end{aligned}$$

in an independently generated test set with a sample size of 10,000. In (6.2), $F$ denotes the cumulative distribution function of a normal random variable with mean zero and variance $\sigma_0^2$.

Variable selection for the `mcm`, `mom`, `dsm`, and `int` methods was performed using the LASSO logistic regression via a 10-fold cross-validation procedure. Based on this selection method, we found cases where the LASSO procedure could not select any non-zero covariate based on cross-validation. In these cases, we simply forced the correlation coefficient to be zero in the test set since the method did not find anything informative. For this reason, we alternatively fitted these methods based on random forests (Breiman, 2001a) using their default settings[1]. We refer to these methods based on random forest fits as `mcm-RF`, `mom-RF`, `dsm-RF` and `int-RF`. The optimal values for the LASSO penalties in (3.2) for the `l2svm` method, and the value of $K$ in (3.9) for the `cknn` method, were also selected via 10-fold cross-validation. Lastly, the methods `upliftRF` and `ccif` were fitted using their default settings[2].

The results over 100 repetitions of the simulation for the first and last four simulation scenarios are shown in Figures 6.1 and 6.2, respectively. These figures illustrate the boxplots of the Spearman's rank correlation coefficient between $\hat{\tau}(X)$ and $\tau(X)$. The boxplots within each simulation scenario are shown in decreasing order of performance based on the average correlation. The `ccif` method performed either the best or next to the best in all eight scenarios.

[1] Specifically, we fitted the models using $B = 500$ trees and $n = \sqrt{p}$ as the number of variables randomly sampled as candidates at each split.

[2] In both cases, we used $B = 500$ trees and $n = p/3$ as the number of variables randomly sampled as candidates at each split. For `ccif` we set the $P$ value $= 0.05$.

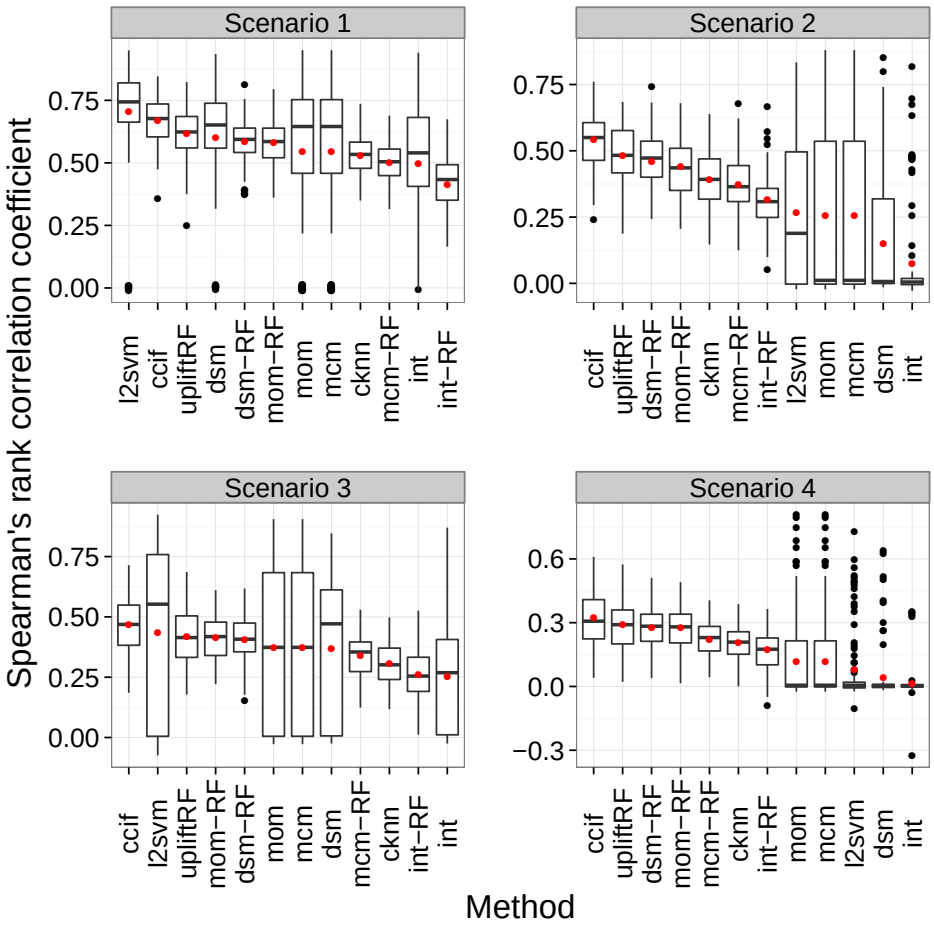


Figure 6.1: Boxplots of the Spearman's rank correlation coefficient between the estimated treatment effect $\hat{\tau}(X)$ and the "true" treatment effect $\tau(X)$ for all methods. The plots illustrate the results for simulation scenarios 1–4, which model a situation with "stronger" treatment heterogeneity effects, under zero and moderate correlation among the covariates ($\rho = 0$ and $\rho = 0.5$) and two levels of noise ($\sigma_0 = \sqrt{2}$ and $\sigma_0 = 2\sqrt{2}$). The boxplots within each simulation scenario are shown in decreasing order of performance based on the average correlation. The dots outside the boxplots represent outliers, determined using the "1.5 rule".

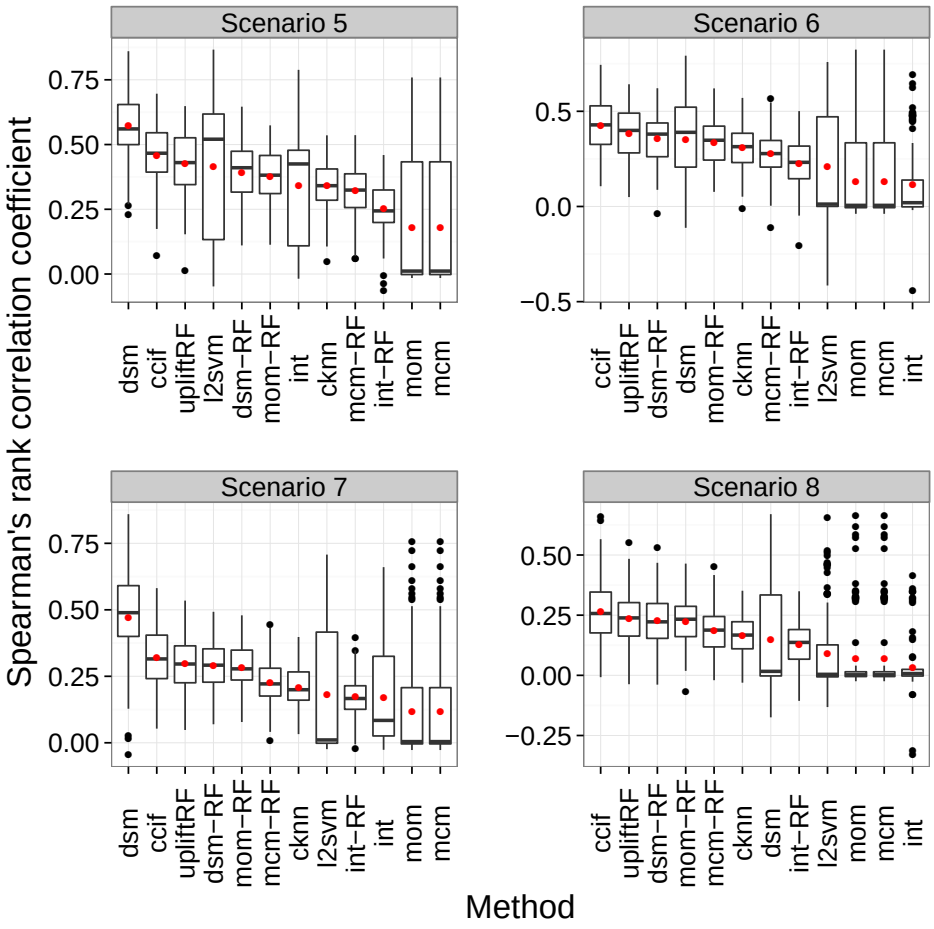


Figure 6.2: Boxplots of the Spearman's rank correlation coefficient between the estimated treatment effect $\hat{\tau}(X)$ and the "true" treatment effect $\tau(X)$ for all methods. The plots illustrate the results for simulation scenarios 5–8, which model a situation with "weaker" treatment heterogeneity effects, under zero and moderate correlation among the covariates ($\rho = 0$ and $\rho = 0.5$) and two levels of noise ($\sigma_0 = \sqrt{2}$ and $\sigma_0 = 2\sqrt{2}$). The boxplots within each simulation scenario are shown in decreasing order of performance based on the average correlation.

# 7 Model assessment and selection for personalized treatment learning models

## 7.1 Introduction

In conventional supervised learning models, *model assessment* refers to estimating the *prediction error* (sometimes called the *generalization error*) of the model. This is generally accomplished by choosing an appropriate loss function to define the lack of fit between the predicted and the actual values of the response variable at the individual observational unit. An independent test sample is commonly used for this purpose. Assessing model performance is more complex for personalized treatment learning (PTL) models as the actual value of the response – the "true" treatment effect – is unknown at the individual subject level. However, we can still assess model performance by comparing *groups* of observations exposed to different treatments. In this chapter we describe methods for model assessment and selection in the context of PTL models.

## 7.2 The Qini curve and the Qini coefficient

The *Qini curve* (Radcliffe, 2007; Radcliffe and Surry, 2011) is a two-dimensional depiction of model performance for PTL models. It represents a natural extension of the *Gains curve*[1] (Blattberg et al., 2008, p. 319)

[1] The Gains curve is also associated with the Receiver Operating Characteristic (ROC) curve. In fact, Hand and Till (2001) point out that the more familiar Gini coefficient is related to the Area under the ROC curve (AUROC) by: $\text{Gini} + 1 = 2 \times \text{AUROC}$.

commonly used by conventional supervised learning models. We formalize the relevant concepts below.

Let $\ell = \{1, \ldots, L\}$ be the set of instances on a given test sample randomized in control and treatment arms, denoted by $A \in \{0, 1\}$, respectively, $Y \in \{0, 1\}$ be a binary response variable and $\hat{\tau}_\ell^M$, $\ell = \{1, \ldots, L\}$ be the personalized treatment effect (PTE) (Equation 2.2) predictions induced by a model $M$. Also, let $\phi \in [0, 1]$ be a given fraction of the $L$ instances with *highest* predicted PTE, and $\mathcal{S}_\phi \subset L$ the subset of instances in this fraction.

We define $R^{A=1}(\phi)$ as the number of positive responses in the treatment group within the fraction $\phi$, expressed as a percentage of the total number of instances in the treatment group. That is,

$$R^{A=1}(\phi) = \frac{\sum_{\forall \ell \in \mathcal{S}_\phi} Y_\ell A_\ell}{\sum_{\ell=1}^{L} A_\ell}. \tag{7.1}$$

Similarly, we define $R^{A=0}(\phi)$ as the number of positive responses in the control group within the fraction $\phi$, expressed as a percentage of the total number of instances in the control group. That is,

$$R^{A=0}(\phi) = \frac{\sum_{\forall \ell \in \mathcal{S}_\phi} Y_\ell (1 - A_\ell)}{\sum_{\ell=1}^{L} (1 - A_\ell)}. \tag{7.2}$$

For a given $\phi$, we define the *net lift* as

$$\textit{net lift}(\phi) = R^{A=1}(\phi) - R^{A=0}(\phi). \tag{7.3}$$

A Qini curve is constructed by plotting $\textit{net lift}(\phi)$ at increasing values of $\phi \in [0, 1]$. This is demonstrated in Figure 7.1. This figure can be interpreted as follows: on the $x$-axis we show the fraction of individuals in the population in which the action is performed, and on the $y$-axis we show the incremental number of positive responses between treatment and control groups[2], expressed as a percentage of the size of the target population. Clearly, if we treat all individuals in the population, the net lift will be equivalent to the average treatment effect (ATE; see Equation 2.1). A benchmark for model $M$ can be represented by the

[2] If the number of individuals in the treatment and control groups differ, the incremental number of responses should be scaled based on the relative sizes of these groups.

strategy of randomly selecting subjects to perform the action. This is represented in the figure by the diagonal line. For example, if we perform the action on 30% of the population, we expect to obtain 30% of the net lift relative to performing the action on the entire population.

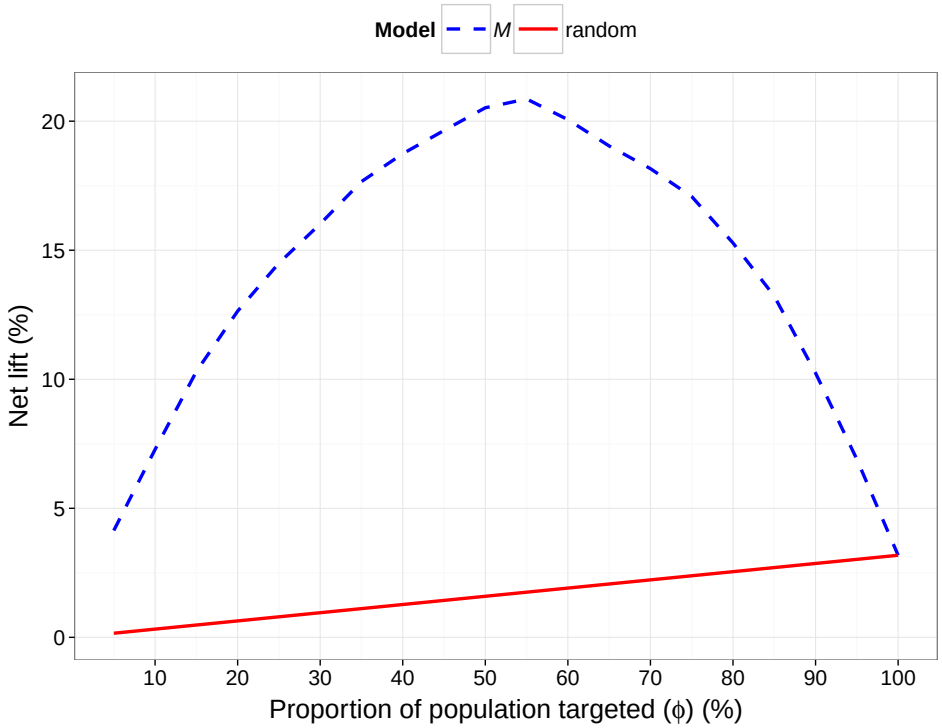


Figure 7.1: Sample Qini curves corresponding to model $M$ and random targeting strategies.

The Qini coefficient $q$ is a single estimate of model performance obtained by subtracting the area under the random curve from the area under the Qini curve. The area under each curve can be approximated by partitioning the domain of $\phi \in [0, 1]$ into $J$ panels, or $J + 1$ grid points $0 = \phi_1 < \phi_2 < \ldots < \phi_{J+1} = 1$, and computing[3]

[3] The approximation uses the formula for the area of a trapezoid, given by the average height times the width of the base.

$$q_m \approx \frac{1}{2}\sum_{j=1}^{J}\Big(\phi_{j+1}-\phi_j\Big)\Big(\textit{net lift}(\phi_{j+1})+\textit{net lift}(\phi_j)\Big), \qquad (7.4)$$

where $m \in \{M, \text{random}\}$ and the Qini coefficient $q = q_M - q_{\text{random}}$. In general, given a set of $n$ PTL models $\{M_1, M_2, \ldots, M_n\}$ and their associated Qini coefficients $\{q_{M_1}, q_{M_2}, \ldots, q_{M_n}\}$ measured on the same test data, the preferred model $M^*$ is the one with maximum Qini coefficient.

Notice that the Qini coefficient may take any value along a continuous line. A negative value implies that the model performs worse than randomly selecting subjects to perform the action. The absolute value of the Qini coefficient is less meaningful than its relative value among different models. It is only advisable to compare Qini values on the same test data.

## 7.3 Optimal Qini curves

The concept of optimal Qini curves can be more readily explained using an example. For simplicity, suppose the sample is composed of $L = 200$ subjects and that $\sum_{\ell=1}^{L} A_\ell = \sum_{\ell=1}^{L}(1 - A_\ell) = 100$ – so that an equal number of 100 subjects is randomly allocated to treatment and control groups. Assume the response rates in the treatment and control groups are 20% and 10%, respectively. That is, $R^{A=1}(\phi = 1) = 20\%$ and $R^{A=0}(\phi = 1) = 10\%$. Clearly, the ATE (or equivalently, the *net lift* at $\phi = 1$) is 10%. What is the optimal Qini curve representing the best possible model we could build in this case? The answer to this question depends on a key feature of PTL models and relates to the presence of *negative effects*. A subject is negatively impacted by treatment if the response under treatment is worse than it would be under control. Thus, if we assume that a value of $Y = 1$ is more desirable than a value of $Y = 0$, a negative effect happens when the subject's response is $Y = 0$ under $A = 1$, but would have been $Y = 1$ under $A = 0$.

Let us now consider two scenarios. In the first scenario, assume there are no negative effects and that the treatment positively impacted 10 subjects (their value of $Y = 1$ under $A = 1$, but would have been $Y = 0$ under $A = 0$). Figure 7.2 (left) shows the optimal Qini curve, $M^{\text{opt}}$, under

this scenario. In this case, an optimal model is able to identify the 10 subjects positively impacted by the treatment within the first decile of the target population. In the second scenario, assume the treatment positively impacted 20 subjects. As the response rate in the control group is 10%, that can only happen if, in addition, the treatment negatively impacted 10 subjects. The optimal Qini curve under this scenario is shown in Figure 7.2 (right). In this case, an optimal model would allocate the 20 subjects with positive treatment effect within the first two deciles of the target population, and the 10 subjects with negative impact in the last decile.

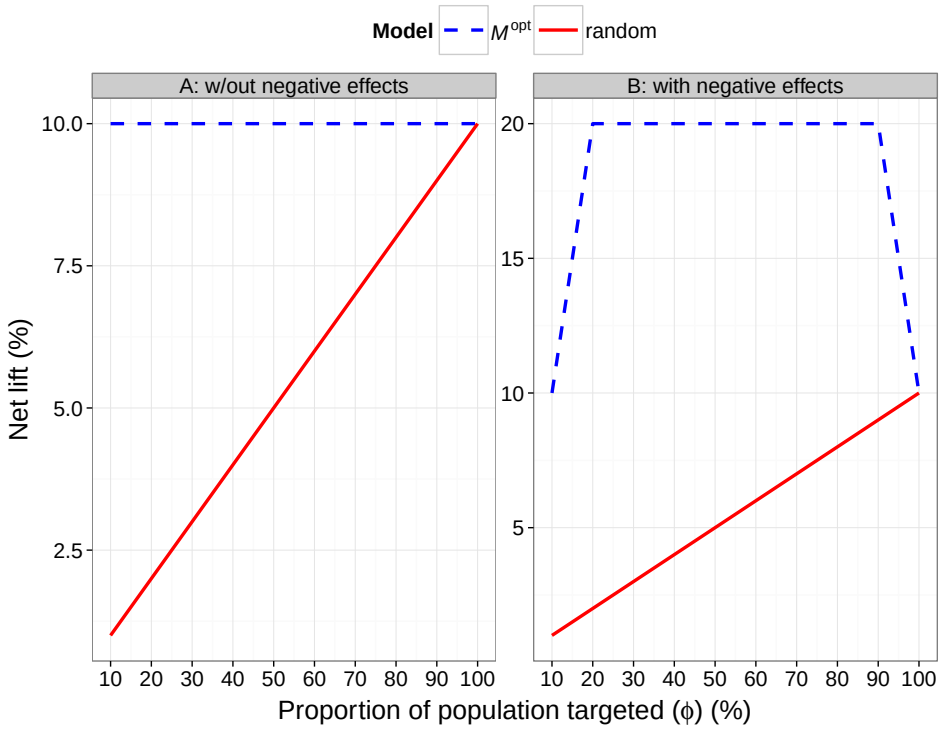


Figure 7.2: Qini curves corresponding to the optimal model $M^{\text{opt}}$ and random model in the absence (*left*) and presence (*right*) of negative effects.

## 7.4 Top uplift

In certain applications, we may not be interested in obtaining an estimate of model performance on the entire target population, but only on a predetermined fraction $\phi \in [0, 1]$ of this population. This is generally the case, for example, in direct marketing applications, where a company may target at most 10% to 30% of a given population. The *top uplift* at $\phi \in [0, 1]$ is obtained by computing the actual ATE on the fraction $\phi$ of the total $L$ subjects with *highest* predicted PTE. That is,

$$top\ uplift(\phi) = \frac{\sum_{\forall \ell \in \mathcal{S}_\phi} Y_\ell A_\ell}{\sum_{\forall \ell \in \mathcal{S}_\phi} A_\ell} - \frac{\sum_{\forall \ell \in \mathcal{S}_\phi} Y_\ell (1 - A_\ell)}{\sum_{\forall \ell \in \mathcal{S}_\phi} (1 - A_\ell)}. \tag{7.5}$$

## 7.5 Resampling methods

Conventional statistical learning methods can be highly adaptable, capable of accurately learning patterns in the data on which the model was built. Unfortunately, these methods may easily overemphasize learning patterns and noise in the data which are unlikely to hold in new samples – in other words, they tend to overfit the data. Overfitting should be avoided as it degrades a model's predictive performance.

As discussed in Chapters 4 and 5, the problem of overfitting is generally worse in the context of PTL models compared to conventional statistical learning models. Treatment heterogeneity effects tend to be significantly weaker compared to main effects, and if the levels of noise in the data are relatively high, a model may have difficulty in accurately learning the underlying treatment heterogeneity relationships from the data (see Section 4.4.3).

All models discussed in this thesis have tuning parameters that allow us to control the degree of flexibility with which they can learn the structure in the data. A tuning parameter is one that is fixed in advance by the user, as opposed to being determined as part of the fitting procedure. For instance, in the context of uplift random forests, we mainly discussed two tuning parameters: the number $n$ of covariates selected at random as candidates for splitting and the number $B$ of uplift trees in the forest. Additionally, in the context of causal conditional inference forests, the $P$ value in Algorithm 2 can also be considered as a tuning parameter, as it

controls the maximum acceptable $P$ value required to split a node in the tree.

Performance measures such as the Qini coefficient or top uplift computed on training data usually overestimate the expected model's performance on new samples. In the presence of a very large data set, a designated validation set can be used for model selection by directly estimating these measures under varying values of the tuning parameters. In addition, a test set can be used for a final performance evaluation of the chosen model. The test set contains observations which were not used for model training or selection. However, we may often have small samples at our disposal and this is where *resampling methods* come into play. These methods consist in drawing a subset of samples to fit a model and leaving the remaining samples to estimate the model's performance. This process is applied repeatedly and the performance results from the individual fits are aggregated. This section discusses the most relevant resampling techniques.

### 7.5.1 $K$-fold cross-validation

This method involves splitting the data into $K$ mutually exclusive subsets or "folds" of equal size. We subsequently set aside the first fold and fit a model to the remaining $K-1$ folds. This model is then used to obtain a performance estimate on the first fold. This process is repeated $K$ times, each time setting aside a different fold. The performance estimates over the $K$ folds are then averaged. For instance, if the Qini coefficient $q$ is used as a performance estimate, the cross-validated $q$ over the $K$ folds is computed as

$$CV_{(K)} = \frac{1}{K}\sum_{k=1}^{K} q_k. \tag{7.6}$$

The most common choice for $K$ is 10, but there is no specific rule. There is a fundamental trade-off between *bias* and *variance* in choosing $K$. As $K$ gets larger, the difference in size between the training set (composed of $K-1$ folds) and the entire data set gets smaller. As a result, the bias of the cross-validation estimator also gets smaller. In this context, the bias refers to the difference between the estimated and true (expected)

performance values. For instance, with $K = L$ (called *Leave-One-Out* cross-validation), the cross-validation estimator is approximately unbiased for the true performance value. However, a lower bias usually comes at a trade-off with a higher variance. As $K$ gets larger, the variance of the cross-validation estimator also gets larger. The reason is that with a large $K$, we are effectively averaging quantities that are highly positively correlated, since the training sets are very similar to each other. The variance of the mean of highly correlated identically distributed quantities is higher than with quantities that are not as highly correlated. Overall, $K = 10$ is usually chosen as a good compromise (see Kohavi, 1995).

### 7.5.2 Monte Carlo cross-validation

This method involves randomly selecting (without replacement) some fraction $\alpha$ of the entire samples to form a training set, and assigning the remaining samples to the validation set. A model is fitted to the training set and the performance measure estimated based on the validation set. This process is then repeated $K$ times, generating new training and validation partitions each time. The final performance estimate is derived by averaging the individual performance estimates over all runs.

Compared to $K$-fold cross-validation, Monte Carlo cross-validation allows us to explore more possible partitions, although it is unlikely to exhaust all $\binom{L}{\alpha L}$ possibilities. As discussed previously, the choice of the proportion of observations used for training and validation involve a bias/variance trade-off. The higher the proportion of observations used in training, the lower the bias but the greater the variance. If the interest is in model selection (i.e., determining the best tuning parameters for a given method) as opposed to model assessment (e.g., predicting the future (expected) top decile uplift with high accuracy), the absolute accuracy of the cross-validation estimator is less relevant than the relative accuracy and we might be willing to trade off bias for low variance, assuming the bias affects all values of the tuning parameters similarly.

### 7.5.3 Bootstrap methods

Given a dataset with $L$ instances, a bootstrap sample consists in drawing a sample (with replacement) of size $L$ from the data (Efron and Tibshirani,

1986). This is done $B$ times, resulting in $B$ bootstrap datasets. The probability of an instance being chosen in a given bootstrap sample is $1-(1-1/L)^L \approx 0.632$. The instances not selected in the bootstrap sample are referred to as *out-of-bag* (OOB) samples. For a given iteration of the bootstrap, a model is fitted to the bootstrap sample and assessed on the OOB samples using a performance measure. From this procedure, we can estimate any aspect of the distribution of the performance measure obtained from the $B$ replications, such as its mean and variance.

The bootstrap method will tend to produce performance estimates with low variance but high bias, as the training sample size is just slightly higher than in 2-fold cross-validation. To account for this bias, the ".632 bootstrap" estimator (Efron, 1983) of performance is defined as

$$\text{perf}_{\text{boot.632}} = \frac{1}{B}\sum_{b=1}^{B}\Big(0.632 \times \text{simple bootstrap performance estimate} + 0.368 \times \text{apparent performance}\Big). \quad (7.7)$$

The *apparent performance* represents the estimated performance measure evaluated on the training set. In estimating performance based on the Qini coefficient or top uplift, the idea of (7.7) is to correct the downward bias in the simple bootstrap performance estimate by averaging it with the upward biased apparent performance estimate. This method will produce over-optimistic performance estimates if the model severely overfits the data, since the apparent performance will dominate the overall performance in this case. A subsequent bootstrap method proposed by Efron and Tibshirani (1997), called ".632+ bootstrap", attempts to put greater weight on the bootstrap performance estimate in situations where the amount of overfitting is large.

### 7.5.4 Selecting final tuning parameters

We discuss here an example for choosing the best tuning parameters in a model based on the estimated Qini coefficient. We will focus on selecting the $P$ value within the causal conditional inference forests method. This is the $P$ value required to split a node in a tree (see Algorithm 2). A

higher $P$ value creates trees with higher interaction depth, which are therefore more complex. As previously discussed, if the model is overly complex, it will easily emphasize learning patterns and noise in the data which may not be reproduced in new samples. Therefore, the selection of the final $P$ value aims to avoid overfitting the model.

For the discussion in this section, we use data from a real direct mail campaign implemented by an international bank. In this campaign, an experiment was carried out by which 6,256 clients were randomly assigned in equal proportions to a treatment and a control group. Clients in the treatment group received a promotion to buy a certain financial product. Clients in the control group did not receive the promotion. We use causal conditional inference forests to identify which clients are more likely to buy this product as a result of the promotion. The performance of the fitted models is assessed based on the Qini coefficient estimated from different resampling methods. Further details about this dataset and a more in-depth analysis are provided in Chapter 11.

Figure 7.3 shows the estimated Qini coefficient as a function of the $P$ value. The *apparent performance* (upper left panel) shows the estimated performance when using all samples to both fit and assess the model. Notice that performance improves as the $P$ value is increased, although more complex models overfit the data. The remaining panels show the estimated performance from three resampling methods: 10-fold cross-validation, Monte Carlo cross-validation (MCCV), and bootstrap. The MCCV method was performed with a 75/25 training/validation split, and the process was repeated 10 times for each $P$ value. For the bootstrap method, recall that in Algorithm 2 we draw a sample with replacement from the training observations before fitting each tree. Therefore, the out-of-bag samples were used as the validation set. We also performed 10 bootstrap replications for each $P$ value. In the figure, we show the mean and $\pm$ one standard error performance estimates. The more instances we leave for the validation set, the higher the bias of our estimate; however, fewer validation set instances produce a wider confidence interval for the estimated Qini. For instance, 10-fold CV only leaves 10% of the data at each iteration for validation, and so the confidence intervals are significantly larger than with the other resampling methods. The MCCV and bootstrap methods indicate a $P$ value of 0.05 as the one that

maximizes the estimated Qini coefficient. This is consistent with the default value used for this tuning parameter. For the 10-fold CV method, any $P$ value lower than 0.15 would produce approximately the same performance. However, as mentioned, our confidence in these estimates is significantly lower.

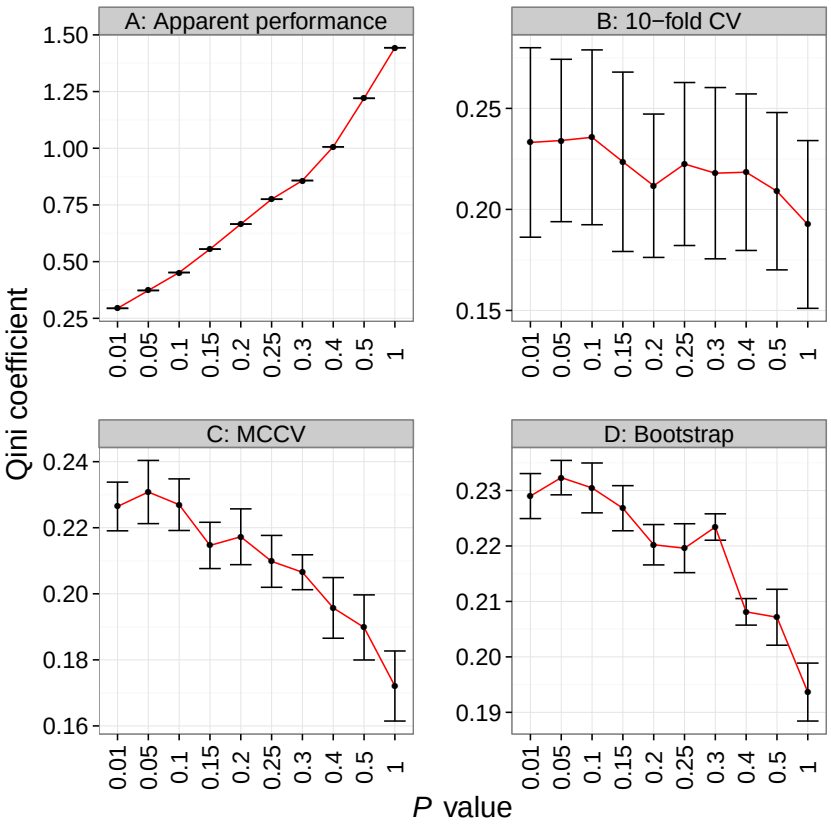


Figure 7.3: Qini coefficient performance profile of causal conditional inference forests under increasing $P$ values based on different resampling methods. For each method, we show the mean and ± one standard error performance estimates. The results are based on the bank's direct mail campaign dataset.

## 7.6 Model calibration

The Qini coefficient and top uplift methods can assess the ability of a model to rank-order subjects in terms of their expected personalized treatment impact. In some applications, in addition to the ranking of subjects, we are also interested in having well-calibrated PTE probabilities. That is, the predicted PTE should be equal to the observed PTE. Model calibration can be assessed using a *calibration plot*, constructed as follows. We first use a fitted model to obtain the PTE predictions on each subject, preferably on a test set. Next, we rank-order the personalized treatment predictions and group them into bins with approximately equal numbers of observations in each. Then we plot the average predicted versus the average actual treatment effect for each bin.

As an illustration, Figure 7.4 shows the calibration plot for a causal conditional inference forest model fitted to the *bank* data introduced in Section 7.5.4. The blue dots show the average observed PTE for each bin versus the average predicted PTE. Notice that the points fall along a 45° line (red), implying that the model has produced well-calibrated probabilities.

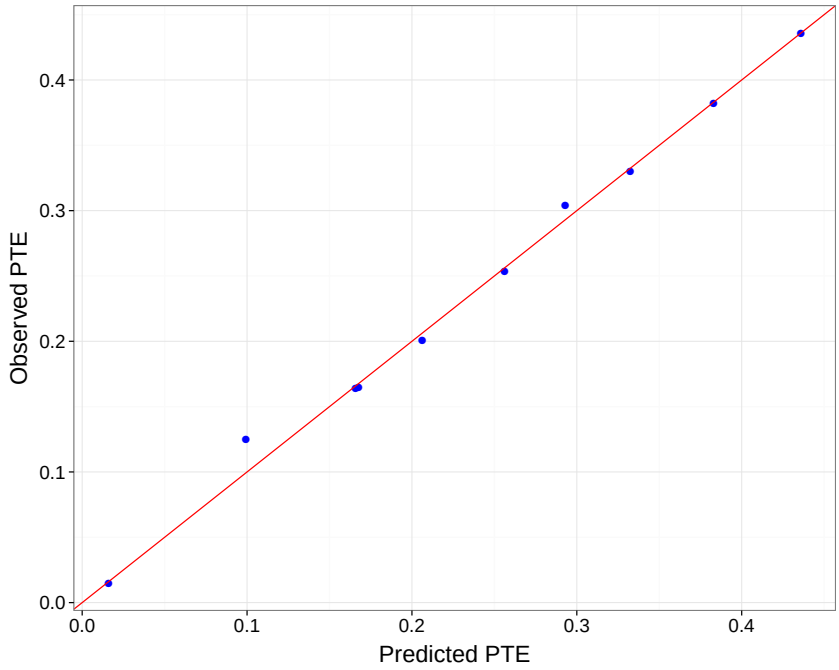


Figure 7.4: Calibration plot for a causal conditional inference forest model fitted to the *bank* dataset.

# 8 Empirical applications in insurance marketing

## 8.1 Introduction

In this chapter, we illustrate two empirical applications of the proposed methods to estimate personalized treatment effects (PTEs; see Chapters 4 and 5) to insurance marketing data. The first is an insurance cross-sell application. Recent relevant contributions to insurance cross-sell consider the potential heterogeneity in the profitability of the cross-sell attempt from an actuarial perspective. In Thuring et al. (2012), a method for selecting policyholders for cross-selling additional insurance contracts using multivariate credibility is implemented. In Kaishev et al. (2013), a new method is proposed for optimal cross-sell selection based on expected profit maximization and mean-variance optimization. Nevertheless, in this context the heterogeneity in response to the marketing intervention activity should also be considered. Personalized treatment learning (PTL) models provide the specific response of each individual to a particular treatment or intervention.

The second application is related to insurance client retention. Client retention is a concept that encompasses all efforts made by a selling company to retain its customers. It has obvious links with marketing strategies, quality, customer service and profitability. When looking at client retention in the context of insurance products, the number of existing contributions is still scarce and mainly focuses on predictive models for the probability of a customer switching to another company (Donkers et al., 2007; Morik and Köpcke, 2004; Smith et al., 2000). Rather than focusing on clients with a high probability of leaving, PTL models

can be used to identify the target clients who are more likely to respond positively to a retention activity.

## 8.2 An insurance cross-sell application

### 8.2.1 The data

The data used for this analysis are based on a direct mail campaign implemented by a large Canadian insurer between June 2012 and May 2013. The objective of the campaign was to drive more business from the existing portfolio of auto insurance clients by cross-selling them a home insurance policy with the company. The standard savings via multiproduct discount was prominently featured and positioned as the key element in the offer to the clients. In addition to the direct mail, the same clients were also contacted over the phone to further motivate them to initiate a home policy quote. A randomly selected control group was included as part of the campaign design, consisting of clients who were not mailed or called. The response variable is determined by whether the client purchased the home policy between the mail date and 3 months thereafter. In addition to the response, the dataset contains approximately 50 covariates related to the auto policy, including driver and vehicle characteristics and general policy information.

Table 8.1 shows the cross-sell rates by group. The average treatment effect (ATE) of **0.34%** (2.55% – 2.21%) is not statistically significant, with a $P$ value of 0.23 based on a chi-squared test. However, the average treatment effect would be of limited value if policyholders show significant heterogeneity in response to the marketing intervention activity. Our objective is to estimate the PTE and use it to construct an optimal treatment rule for the auto insurance portfolio – namely, the policyholder-treatment assignment that maximizes the expected profits from the campaign.

Table 8.1: Cross-sell rates by group

| | Treatment | Control |
|---|---|---|
| Purchased home policy = N | 30,184 | 3,322 |
| Purchased home policy = Y | 789 | 75 |
| Cross-sell rate | 2.55% | 2.21% |

*Note.* This table displays the cross-sell rate for the treatment and control groups. The average treatment effect (ATE) is **0.34%** (2.55%−2.21%), which is not statistically significant ($P$ value = 0.23).

### 8.2.2 Building the model

We used causal conditional inference forests (`ccif`) to estimate the PTE from the marketing intervention activity. To objectively examine the performance of the proposed method, we randomly split the data into training and validation sets in a 70/30 ratio. A preliminary analysis showed that model performance is not highly sensitive to the values of its tuning parameters (i.e., number of trees $B$ and number of variables $n$ randomly sampled as candidates at each split), as long as they are specified within a reasonable range. Thus, we fitted a `ccif` to the training data using its default parameter values. Specifically, in Algorithm 2, we used $B = 500$, $n$ $(= p/3) = 16$, and 0.05 as the level of significance $\alpha$ for the $P$ values. We next ranked policyholders in the validation data set based on their estimated PTE (from high to low), and grouped them into deciles. We then computed the actual ATE within each decile (defined as the difference in cross-sell rates between the treatment and control groups).

### 8.2.3 Results

Figure 8.1 shows the boxplots of the actual ATE for each decile based on 100 random training/validation data partitions. The results show that clients with higher estimated PTE were, on average, positively influenced to buy as a result of the marketing intervention activity, with ATEs above 1% for the first three deciles as compared with the ATE

of 0.34% over all deciles. Also, notice there is a subgroup of clients (deciles 8-10) whose purchase behavior was negatively impacted by the campaign. Negative reactions to sale attempts have been recognized in the literature (Günes et al., 2010; Kamakura, 2008; Byers and So, 2007) and may happen for a variety of reasons. For instance, the marketing activity may trigger a decision to shop for better multiproduct rates among other insurers. Moreover, if the client currently owns a home policy with another insurer, she may decide to switch her auto policy to that insurer instead. We found evidence of higher auto policy cancellation rates in the higher deciles. In addition, some clients may perceive the call as intrusive and likely be annoyed by it, generating a negative reaction.

In the context of insurance, it is important to consider not only the PTE from the cross-sell activity, but also the expected insurance losses from the targeted clients (Thuring et al., 2012; Kaishev et al., 2013; Englund et al., 2009). We determined the expected profitability from targeting each decile by subtracting the fixed and variable campaign expenses from the product between the ATE and the expected lifetime-value of a home policy[1]. Based on these considerations, Figure 8.1 shows that only clients in deciles 1-3 have positive expected profits from the marketing activity and should be targeted. The incremental profits from clients in deciles 4-7 is outweighed by the incremental costs, and so the company should avoid targeting these clients. Clients in deciles 8-10 have negative reactions to the campaign and clearly should not be targeted either.

[1] The expected lifetime-value (LTV) of a home policy in decile $i = \{1, \ldots, 10\}$ is given by $LTV_i = [\text{Prem}_i - \hat{LC}_i - \text{Exp}_i] \sum_{t=1}^{5} P(S_{it}) r^t$, where Prem is the average policy premium, $\hat{LC}$ is the predicted insurance loss per policy-year, Exp captures the fixed and variable expenses for servicing the policy (excluding campaign expenses), $P(S_{it})$ is the probability that a policyholder in decile $i = \{1, \ldots, 10\}$ will continue with the home product beyond year $t = \{1, \ldots, 5\}$, and $r^t$ is the interest discount factor.

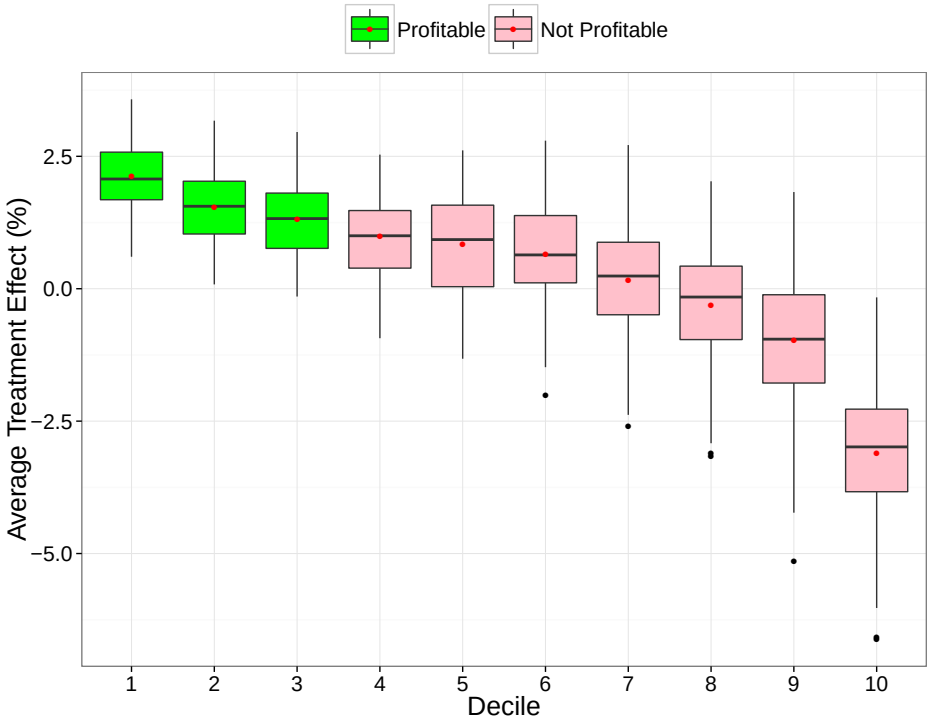


Figure 8.1: Boxplots of the actual average treatment effect (ATE) for each decile based on 100 random training/validation data splits. The first (tenth) decile represents the 10% of clients with highest (lowest) predicted PTE. Clients with higher estimated PTE were, on average, positively influenced to buy *as a result* of the marketing intervention activity.

## 8.3 An insurance customer retention case

### 8.3.1 The data

As a result of an upcoming increase in auto insurance rates, a large Canadian insurer was interested in designing retention strategies to minimize the attrition rate of its existing book of clients. For that purpose, an experimental retention program was implemented in which policyholders coming up for renewal were randomly allocated to either a treatment or to a control group. Policyholders under the treatment group received a letter in the mail, which notified them about the rate

increase and clearly explained the reasons for it. This group also received a courtesy call made by one of the company's licensed insurance advisors. The purpose of the call was to verbally reinforce the items described in the letter, and to ensure that all applicable discounts were in place for the policy (such as those resulting from also insuring the property with the company and more than one vehicle). In addition, the advisors were trained to deal with situations of customer dissatisfaction with the company.

No retention efforts were applied to the control group. As the campaign was designed under randomized assignment, the results observed in this group represent a natural benchmark for comparison. Table 8.2 shows the attrition results. The observed difference in attrition rates between the treated group and the control group is very small (0.3%). Certainly, this result is not enough to cover the cost of the campaign. However, our aim here is to determine whether the campaign had positive retention effects on some subgroup of clients, which were however offset by negative effects on other subgroups. If so, the company would ideally only target clients with positive impact in the future. Adverse effects can happen in retention programs (Stauss et al., 2005; Guillén et al., 2011), for example, if the customer is already dissatisfied and perceives the call as intrusive, or if the treatment triggers a behavior to shop for a better price among other insurers.

Table 8.2: Attrition rates by group

| | Overall | Treatment | Control |
|---|---|---|---|
| Retained policies | 10,857 | 7,492 | 3,365 |
| Canceled policies | 1,111 | 757 | 354 |
| Attrition rate | 9.3% | 9.2% | 9.5% |

### 8.3.2 Building the models

For this experimental evaluation, we compare four PTL models: the uplift random forest method (`upliftRF`), a single uplift tree with pruning (`single-tree`), the difference score method (`dsm`), and the interaction method (`int`). All these models are described in Chapters 3 and 4, except the single uplift tree, described in Rzepakowski and Jaroszewicz (2012).

Model selection in the case of `upliftRF` involves determining the optimal value of its two parameters: the number of covariates $n$ randomly sampled as candidates at each split from the set of $p$ covariates, and the number of trees $B$. For that purpose, we first specified a grid of suitable choices for these parameters. Specifically, we used $n = 1, 2, 3, \ldots, 31$ $(= p)$ and $B = 100$ to 2000 (incremented by 100). Euclidean distance was used as the split criterion, and the sample fraction $\nu$ of training observations was fixed at 0.5. The parameters were selected to maximize the out-of-bag Qini coefficient. For `single-tree`, we also fitted the model to the same fraction, and pruned the tree on the remaining fraction. This was done by traversing the tree bottom up and testing, for each (non-terminal) node, whether collapsing the subtree rooted at that node with a single leaf would improve accuracy as measured by the Qini coefficient, in which case the subtree was replaced and the process repeated. For the `dsm` approach, we fitted two independent stepwise logistic regression models on the treatment and control observations, and then subtracted the class probabilities from the two models. We also used a logistic model for the `int` approach.

To maximize the usage of the training data, model assessment for all models was done using a $K$-fold cross-validation procedure with $K = 10$. This involves splitting the training data into $K$ equal parts, following the model selection procedure described above on $K - 1$ parts of the data, and then evaluating the value for the Qini coefficient on the $k$th part. This was done for $k = 1, 2, \ldots, K$ and then the $K$ estimated values for the Qini were averaged. Notice this mimics the application of the classifier to an independent test set, since model selection is not done using the left-out $K$ samples.

### 8.3.3 Results

Figure 8.2 shows the Qini curves (see Chapter 7) from all methods. The Qini curve shows the cumulative number of incremental retained clients relative to the cumulative number of targets (both expressed as a percentage of the total targets). That is, the curve shows the expected increase in the overall retention rate (or equivalently, the reduction in attrition rate) as a result of targeting a given proportion of the population. Also, if we randomly target $\gamma$ percent of the population, we expect to obtain $\gamma$ percent of incremental retained clients relative to targeting the entire population. This is depicted by the diagonal line.

Based on the Qini coefficient, the `upliftRF` performs best in this application, with the `int` approach being second. The `dsm` and the `single-tree` perform worst. However, none of these models dominate the others at all target volumes. The `upliftRF` model performs much better for low target volumes, which is desirable in this application. This model identified the 30% of clients for whom the retention program was highly effective. At this target volume, the overall attrition rate falls by 1.7%, from 9.5% to 7.8%. Any additional targeted client would result in a smaller reduction in attrition, as a result of null or negative effects of the campaign on the remaining clients.

In addition, Table 8.3 shows the difference in attrition rates between treatment and control groups for the top decile targets from each model. Notice that the `int` and `dsm` methods are able to identify the top 10% of clients with highest attrition rate, but not necessarily the clients most positively impacted by the retention activity. The results confirm that our approach is suited to selecting a group of clients for which the retention program is highly effective, even when the impact of the program on the overall population is negligible. Targeting the group with highest expected impact is essential for the profitability of the campaign.

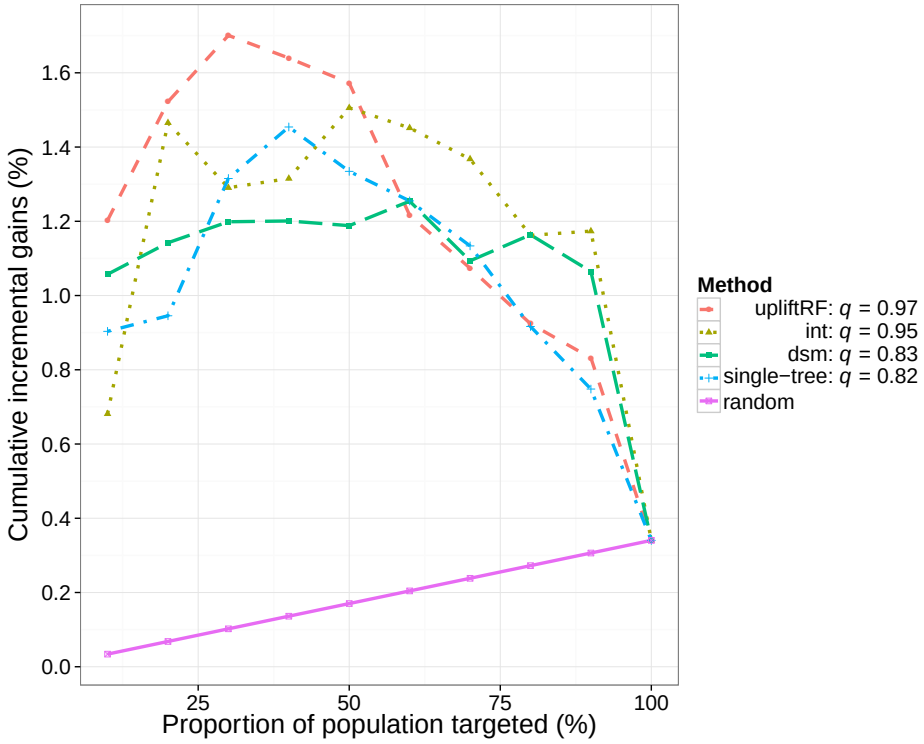


Figure 8.2: This figure illustrates the Qini curve for each model based on the client retention dataset. This curve shows the cumulative number of incremental retained customers relative to the cumulative number of targets (both expressed as a percentage of the total targets). The diagonal line depicts the theoretical incremental retained customers from random targeting. The Qini coefficient ($q$) is obtained by subtracting the area under the random curve from the area under the Qini curve and represents a single estimate of model performance. The approach based on uplift random forest (red) performs best in this application.

Table 8.3: Top decile uplift

| | Attrition rate (%) | | |
|---|---|---|---|
| | Control | Treatment | Uplift |
| `upliftRF` | 21.24 | 9.21 | 12.03 |
| `dsm` | 33.60 | 23.03 | 10.57 |
| `single-tree` | 13.98 | 5.21 | 8.77 |
| `int` | 27.41 | 20.60 | 6.81 |
| `random` | 9.50 | 9.20 | 0.30 |

# 9 Personalized treatment learning in observational studies: An empirical application to insurance price elasticity modeling

## 9.1 Introduction

In prior chapters, we have assumed that the treatment is assigned to the observational units using some sort of randomization procedure. In this chapter, we relax this assumption and present an empirical application of personalized treatment learning (PTL) to observational data in the context of insurance price elasticity modeling. We draw on the terminology and framework of *causal inference* (Holland, 1986) and use the term *causal effect* to refer to an effect that is attributable to a specific treatment. An effect is attributable to a specific treatment if it would not have been observed had the subject been exposed to an alternative treatment condition instead (Rosenbaum, 2002).

Understanding the precise nature of price sensitivities at the individual policyholder level is extremely valuable for insurers. A rate increase has a direct impact on the premium customers are paying, but there is also the indirect impact as a result of the "causal effect" of the rate change on the customer's decision to renew the policy term. A rate increase may impair its intended impact on the overall profitability of the portfolio if it causes a large number of policyholders to lapse their policy and switch

to an alternative insurer.

The difficulty in measuring price elasticity from most insurance databases is that historical rate changes are reflective of a risk-based pricing exercise. As a result, the specific rate change to which a customer is exposed is a deterministic function of her observed covariates. The nature of the data is thus observational, rather than experimental. In this context, measuring the causal effect of a rate change on the policyholder's lapse outcome requires special modeling considerations. Conventional modeling approaches aimed to directly fit the lapse outcome as a function of the rate change and background covariates are likely to be inappropriate for the problem at hand.

In this chapter, we propose a PTL framework to measure price elasticity in the context of automobile insurance. One of the strengths of our approach is transparency about the extent to which the database can support causal effects from rate changes. The model also allows us to more reliably estimate price-elasticity functions at the individual policyholder level. Since the causal effect of a rate change varies across individuals, making an accurate rate change choice at the individual subject level is essential. The rate to which each subject is exposed could be optimized on the basis of the individual's characteristics, in order to maximize the overall expected profitability of the portfolio.

This chapter is organized as follows. We first introduce the concept of price elasticity in insurance and formalize the price elasticity estimation problem from a PTL perspective. We follow this with an overview of the key assumptions required to derive unbiased estimates of the average causal effects caused by treatment interventions from observational data. Propensity scores and matching algorithms are discussed next. The second half of the chapter presents a detailed application of our approach to price elasticity estimation in the context of auto insurance. Finally, we outline managerial implications, and offer some reflections on shifting to a causal inference paradigm for estimating price elasticity.

## 9.2 Price elasticity in insurance

Cost-based pricing of individual risks is a fundamental concept in the actuarial ratemaking literature. The goal of ratemaking methodologies

is to estimate the future costs related to the insurance coverage. The loss cost approach defines the price of an insurance policy as the ratio of the *estimated costs* of all expected future claims against the coverage provided by the policy to the *risk exposure*, plus expenses (Denuit et al., 2007). There is a wealth of actuarial literature regarding appropriate methodologies for using exposure and claims data in order to calculate indicated rates (Finger, 2006; Brown and Gottlieb, 2007).

A revised set of rates will impact the profitability of an insurance portfolio due to its direct impact on the premiums that policyholders are paying. However, there is also an indirect impact resulting from the policyholders' reaction to the rate change. As basic auto insurance is mandatory in many countries, a rate change exceeding a certain threshold will make a policyholder more likely to shop for an alternative insurer and potentially switch to another company. If the rate change causes a large number of customers to lapse their policies, the revised rates could impair the intended impact on the profitability of the insurance portfolio.

In recent years, insurers have been switching from purely cost-based to partially demand-based pricing. Price optimization strategies (Santoni and Gómez Alvado, 2008) aim to integrate cost-based pricing and the customer's willingness to pay into an overall pricing framework. A key component of this framework involves predicting, to a high degree of accuracy, how customers will respond to alternative rate changes, conditional on each customer's characteristics being held fixed[1].

If we consider the rate change as a treatment with varying "dose" levels, the main problem involves the selection of optimal treatments for individuals on the basis of estimates of potential outcomes resulting from treatment alternatives. A similar kind of estimation problem is found in many disciplines, ranging from economics to medicine. In this sense, the price elasticity problem can be conceived under a causal inference framework, which is typically interested in questions of the form "what would happen to a subject had she been exposed to treatment $B$ instead of $A$?" The alternative choice $B$ is a counterfactual with an associated potential outcome. Thus, considerations about potential outcomes from alternative treatment choices seem an inescapable part of

[1] An additional issue is the reaction to new products or cross-selling (see, for instance Kaishev et al., 2013; Thuring et al., 2012).

the price elasticity estimation problem.

A randomized controlled experiment is generally the best approach for drawing statistical inferences about effects caused by treatments. The most effective way to measure price elasticity at the portfolio level would be to randomize the allocation of policyholders to various treatment levels and then measure the impact on retention. However, in the most common situation, insurance databases contain historical price changes that reflect a risk-based pricing approach. Under these conditions, treatment assignment is a deterministic function of the policyholder's observed risk characteristics. The nature of the data is thus observational rather than experimental, as randomization has not been used to assign treatments. In the absence of experimental design, causal inference is more difficult and requires appropriate modeling techniques.

The standard actuarial approach to measuring price elasticity in insurance is to model the policyholder's lapse behavior as a function of the rate change and the policyholder's covariates (Anderson et al., 2007; Yeo et al., 2001; Smith et al., 2000). The key assumption is that the inclusion of those covariates will adjust for the potential exposure correlations between price elasticity and other explanatory variables. This approach is unreliable for estimating causal effects from observational data due to masked extrapolation problems, and to the sensitivity of the results to unwarranted assumptions about the form of the extrapolation (Rubin, 1973; Rubin, 1979; Morgan and Winship, 2007, p. 129; Berk, 2004, p. 115; Guo and Fraser, 2010, p. 82). The problem is even worse when the number of explanatory variables is large, as groups may differ in a multivariate direction and so non-overlap problems are more difficult to detect (Rubin, 1997). Standard statistical software can be remarkably deceptive for this objective because regression diagnostics do not include a careful analysis of the distribution of the predictors across treatment groups. When the overlap (formally defined as *common support*; see Equation 9.2 in Section 9.4.1) is too limited, the data cannot support any causal conclusions about the differential effects of treatments (Englund et al., 2008; Guelman et al., 2012; Guillén et al., 2012).

In this chapter, we propose a method for estimating price elasticity with roots in Rubin's causal model (Rosenbaum and Rubin, 1983; Rosenbaum and Rubin, 1984; Rubin and Waterman, 2006). One of the strengths of

our approach is its transparency regarding data support for estimating the impact of rate changes on customer retention at the portfolio level. Our model also allows us to more reliably estimate individual price-elasticity functions. As the causal effect of a rate change varies across individuals, an accurate choice of the treatment at the individual subject level is essential. Each subject's treatment could be optimized on the basis of individual characteristics, and thus maximize the overall positive impact of the rate change intervention.

## 9.3 Price elasticity as a personalized treatment learning problem

We postulate the problem in the context of Rubin's model of causality introduced in Section 2.1. Recall that this model conceptualizes the PTL problem in terms of potential outcomes under each treatment, only one of which is observed for each subject. For the remainder of this chapter we use the words *treatment* and *rate change* interchangeably.

The insurance portfolio is composed of $L$ policyholders characterized by baseline covariates $\mathbf{X} = (X_1, \ldots, X_p)^\top$. For each $\ell \in \{1, 2, ..., L\}$, we postulate the existence of *potential responses* $R_\ell(a)$ to denote the renewal outcome[2] that would be observed from policyholder $\ell$ if assigned to treatment $a \in A$. In the binary treatment case, $A = \{0, 1\}$. Here we allow $A$ to take a discrete number of $T$ values, $A = \{1, 2, \ldots, T\}$. Further, we let $\{Z_{\ell a} \mid a \in A\}$ be a set of $T$ binary treatment indicators, such that $Z_{\ell a} = 1$ if subject $\ell$ received treatment $A = a$, and $Z_{\ell a} = 0$ otherwise. The *observed response* for subject $\ell$ is $r_\ell = \sum_{a=1}^{T} Z_{\ell a} R_\ell(a)$.

Our interest lies in estimating price elasticity, defined here as the expected renewal outcomes that causally result from the rate change interventions. Here causation is in the sense of *ceteris paribus*, meaning that we hold all policyholder's covariates constant. Our aim is to obtain an estimate of the price-elasticity functions at the policyholder level, $\hat{R}_\ell(a) \; \forall \; a = \{1, \ldots, T\}$, and in particular in differences of the form $\hat{R}_\ell(j) - \hat{R}_\ell(k)$, the causal effect of exposing subject $\ell$ to treatment $j$ rather than

[2] We denote the renewal outcome equal to 1 if the policyholder lapses (does not renew), and 0 otherwise.

to treatment $k$ (for any $j \neq k$). We then use these individual estimates to construct an aggregate price-elasticity function at the portfolio level, $\hat{\mu}(a) = (1/L)\sum_{\ell=1}^{L} \hat{R}_\ell(a)$. If the variability of the causal effect $\hat{R}_\ell(j) - \hat{R}_\ell(k)$ is large over $L$, then the average defined by $\hat{\mu}(a)$ may not represent the causal effect on a specific policyholder $\ell$. The assumption that the effect of $A$ is the same on every subject is known as the *constant treatment effect* assumption, and it is relaxed in this study.

In the context of observational data, policyholders exposed to different rate change levels are not directly comparable, so price-elasticity estimation requires adjustment for differences in the pre-treatment covariates. As discussed above, when the number of covariates is large and their distribution varies substantially among the different rate change levels, simple covariance adjustment methods are typically inadequate. We propose using propensity scores (Rosenbaum and Rubin, 1983) and matching algorithms (Gu and Rosenbaum, 1993) as a method for removing all biases associated with differences in the pre-treatment variables. Our methodology offers a rigorous analysis of price-elasticity in the context of auto insurance based on causal inference foundations. The next section discusses our method in detail.

## 9.4 The method

Without loss of generality, in this section we present our method in a simplified case. We focus on the binary treatment case, with $A = \{0, 1\}$, and let $Z_\ell = 1$ if subject $\ell$ received treatment $A = 1$ (the *treated* subjects), and $Z_\ell = 0$ if the subject received the alternative treatment $A = 0$ (the *control* subjects). In the context of this study, multi-valued treatments are handled by analyzing a set of binary treatment dichotomies. That is, given $T$ treatments, we analyze the $T(T-1)/2$ unordered dichotomies[3].

### 9.4.1 Unconfoundedness and common support

The fundamental problem of PTL is that each subject receives only a single treatment, and thus $R_\ell(a)$ is only observed for a single value of $A$.

[3] For example, with three treatments ($T = 3$), there are $3 = T(T-1)/2$ unordered treatment dichotomies: $\{(1, 2), (1, 3), (2, 3)\}$.

Hence, causal inference is, in a sense, a missing data problem because the counterfactual outcomes are never observed. In principle, if treatment assignment were randomized across the portfolio of policyholders, implying that the assignment ignored the possible impact of the treatment on the outcomes, then estimating the average causal effects of rate changes would be straightforward. Randomization tends to balance observed and unobserved covariates across the treatments, as subjects are drawn from the same population. In this context, the average treatment effect (ATE) of treatment 1 relative to treatment 0 can be estimated from (2.1).

As discussed above, such randomization is unlikely to happen in most insurance databases from which price elasticity is to be estimated, since treatment assignment is a deterministic function of the policyholder's observed risk characteristics. The nature of the data is thus observational, as randomization is not used to assign treatments. In this setting, covariates are not likely to be balanced across treatment groups. Estimating casual effects is more difficult, since now the groups are not directly comparable. However, much progress can be made under two assumptions. The first, the *unconfoundedness* assumption, states that conditional on $\mathbf{X}_\ell$, the outcomes $(R_\ell(1), R_\ell(0))$ are independent of treatment $Z_\ell$:

$$\Big(R_\ell(1), R_\ell(0)\Big) \perp Z_\ell | \mathbf{X}_\ell. \tag{9.1}$$

This condition implies that treatment assignment may depend upon the observed covariates $\mathbf{X}$, but not on unobserved covariates or potential responses after controlling for $\mathbf{X}$. This assumption is non-testable, but very likely to hold in our study, as all the historical variables used to assign policyholders to rate change levels are observable covariates, which have been stored and are accessible to us for the modeling exercise.

The second assumption is that of *common support* (also called overlap), which states that every unit in the population has a chance of receiving both treatments:

$$0 < \pi(\mathbf{X}_\ell) \equiv P(Z_\ell = 1 | \mathbf{X}_\ell) < 1, \tag{9.2}$$

where $\pi(\mathbf{X})$ is known as the *propensity score*, discussed in the next section. This assumption is at risk in situations where treatment assignments are based on 'hard rules' (e.g., every policyholder whose *age* > some constant

receives a rate change, and no rate change otherwise). However, in many situations, rate changes are implemented using much more convoluted frameworks, creating opportunities for finding common support situations. In Rosenbaum and Rubin (1983), *unconfoundedness* and *common support* together constitute a property known as *strong ignorability*, which is necessary for identifying average treatment effects.

Common support may not hold globally, but only for a subset of the covariate space. Causal effect estimation is still possible for the region of $\mathbf{X}$ in which the treatment and control observations overlap. In the specific case that the support of $\mathbf{X}$ for the treated is a subset of the support of $\mathbf{X}$ for the control, then a quantity of common interest is the *average treatment effect for the treated (ATT)*, which is identifiable under unconfoundedness, and it is estimated as

$$\begin{aligned} ATT(Z_\ell = 1) &= E[R_\ell(1)|Z_\ell = 1] - E[R_\ell(0)|Z_\ell = 1] \\ &= E_{\mathbf{X}_\ell|Z_\ell=1}\{E[R_\ell|\mathbf{X}_\ell, Z_\ell = 1] - E[R_\ell|\mathbf{X}_\ell, Z_\ell = 0]|Z_\ell = 1\}, \end{aligned} \tag{9.3}$$

where the subscripts $\mathbf{X}_\ell|Z_\ell = 1$ indicate that the outer expectation is taken over the distribution of $\mathbf{X}$ in the treated group. Finding treated and control observations with similar values of the covariates will be impractical, if not impossible, when there are many covariates. Alternative methods must be used, and we discuss these in the next two sections.

### 9.4.2 Propensity score

The propensity score is the conditional probability of assignment to the treatment condition given the pre-treatment covariates,

$$\pi(\mathbf{X}_\ell) = P(Z_\ell = 1|\mathbf{X}_\ell). \tag{9.4}$$

In a randomized experiment, treatment assignment is performed by a coin flip, and so the propensity score $\pi(\mathbf{X}) = 1/2$ for all subjects. In this case, the results observed in the treatment and control groups are directly comparable as subjects are likely to be similar. In contrast, in an observational study, the propensity score is typically unknown,

and must be estimated from observed quantities. Direct comparisons can be misleading as some individuals are more likely than others to receive one of the treatments, and so $\pi(\mathbf{X}) \neq 1/2$ for some individuals. However, suppose we pair subjects with different treatments, but the same propensity score. The individual pairs might have different covariate values, but their difference will be irrelevant for predicting treatment assignment. Intuitively, this also suggests that the distribution of the observed covariates will be similar for treated and control subjects with the same propensity score. This thought is formalized by the *balancing property* of the propensity score, which states that treatment $Z$ and the observed covariates $\mathbf{X}$ are conditionally independent given the propensity score $\pi(\mathbf{X})$,

$$Z_\ell \perp \mathbf{X}_\ell | \pi(\mathbf{X}_\ell). \tag{9.5}$$

Instead of having to match subjects exactly on their covariates $\mathbf{X}$, the balancing property allows us to match only on a single variable, namely the propensity score, and this will tend to balance all the observed covariates. Notice that this property only ensures balance on the observed covariates. In that sense, randomization is a much more effective tool to balance covariates, as it also provides a basis for expecting that unobserved covariates and potential responses are also balanced[4].

The balancing property holds independently of whether the treatment assignment is strongly ignorable or not. However, a second property of the propensity score is a key result which shows that if treatment assignment is strongly ignorable given $\mathbf{X}$, then it is also strongly ignorable given the propensity score $\pi(\mathbf{X})$. That is, if (9.1) and (9.2) hold, then the following also hold:

$$\Big(R_\ell(1), R_\ell(0)\Big) \perp Z_\ell | \pi(\mathbf{X}_\ell) \tag{9.6}$$

and

$$0 < P\Big(Z_\ell = 1 | \pi(\mathbf{X}_\ell)\Big) < 1. \tag{9.7}$$

This property implies that if treatment assignment is strongly ignorable,

[4] One should still expect imbalances on observed covariates to occur in a randomized setting – in fact, 1 out of 20 covariates should differ at 0.05 level by chance alone.

then pair matching based on the propensity score is sufficient to produce unbiased estimates of the average treatment effect.

### 9.4.3 Matching: A short review

The essential idea of matching algorithms is to find pairs of subjects, where one member of the pair has been treated and the other is a control subject, but they are otherwise identical in terms of their observed covariates prior to exposure. Finding such pairs for all subjects is a difficult or impossible task when $\mathbf{X}$ contains many covariates, and this is where the propensity score comes into play. Matching serves two purposes. First, if we can find among the $L$ subjects a total of $2J$ distinct subjects matched in $J$ pairs, we will have reconstructed a randomized experiment from observational data. Inference about average effects caused by treatments would then be straightforward. Second, having formed the $J$ closely matched pairs, we could use the observed response on one subject of the pair to fill in the "missing" counterfactual response for the other subject of the pair, thereby using the difference between the responses as an estimate of the subject-level causal effect (Rubin and Waterman, 2006).

Using matching to find pairs of subjects may be straightforward in concept, but there are many variants on how this can be achieved in practice. The selection of a matching method essentially involves three choices. First, there is the definition of *distance* between treated and control subjects in terms of their observed covariate vectors. Second, there is the choice of the *algorithm* used to form the matched pairs to make the distance small. Lastly, there is the choice of the *structure of the match*, which involves deciding the number of treated and control subjects that should be included in each matched set.

Let us first consider the definition of distance. A common method for multivariate matching is based on Mahalanobis distance (Cochran and Rubin, 1973; Rubin, 1979). The Mahalanobis distance between any two subjects, say $\ell_1$ and $\ell_2$, with covariate vectors $\mathbf{X}_{\ell_1}$ and $\mathbf{X}_{\ell_2}$ is given by

$$MD(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}) = \{(\mathbf{X}_{\ell_1} - \mathbf{X}_{\ell_2})^\top S^{-1}(\mathbf{X}_{\ell_1} - \mathbf{X}_{\ell_2})\}^{\frac{1}{2}}, \tag{9.8}$$

where $S$ is the sample covariance matrix of $\mathbf{X}$, which corresponds to the data matrix of observations containing the $\mathbf{X}_\ell^\top$ row vectors, $\ell = \{1, \ldots, L\}$.

The Mahalanobis distance is appropriate for multivariate normal data, but it can exhibit odd behavior in the presence of highly skewed distributions or heavily tied covariates. A more robust alternative it to use the rank-based Mahalanobis distance

$$RMD(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}) = \{(rk(\mathbf{X}_{\ell_1}) - rk(\mathbf{X}_{\ell_2}))^\top (UDU)^{-1}(rk(\mathbf{X}_{\ell_1}) - rk(\mathbf{X}_{\ell_2})\}^{\frac{1}{2}}, \tag{9.9}$$

where the covariates in $\mathbf{X}_\ell$ are replaced by their ranks $rk(\mathbf{X}_\ell)$, $D$ is the covariance matrix of the ranks, and $U$ is a diagonal matrix whose elements are the ratios of the standard deviation of the untied ranks to the standard deviation of the tied ranks of the covariates.

Propensity score matching involves matching a treated unit to the nearest control unit based on the distance along the propensity score

$$PS(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}) = |\hat{\pi}(\mathbf{X}_{\ell_1}) - \hat{\pi}(\mathbf{X}_{\ell_2})|. \tag{9.10}$$

In practice, the propensity score must be estimated by, for example, a logistic regression model. This being the case, distance is generally defined in terms of the estimated linear predictor, rather than on the estimated propensity score $\hat{\pi}(\mathbf{X})$, thus avoiding compression of probabilities near zero and one. Additionally, the linear predictor is often more nearly normally distributed, which offers a technically justified advantage under certain data conditions and matching methods (see Rosenbaum and Rubin 1985; Rubin 1976).

Matched samples may be evaluated based on two different, but desirable, features. One is based on the *balance* criterion, which refers to obtaining a similar distribution of the observed covariates $\mathbf{X}$ for treated and control units. The other is based on a stronger *distance* criterion, which is judged by the closeness of the individual matched pairs in terms of their covariate values. The main disadvantage of propensity score matching is that matched units with the same estimated propensity score may have different patterns of covariates $\mathbf{X}$, and this is ignored by (9.10). A hybrid alternative, the Mahalanobis distance with propensity score calipers $MP(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2})$, insists that subjects be close on the propensity score, but once this is achieved, the values of $\mathbf{X}$ matter. This distance is

set to infinity if $\pi(\mathbf{X}_{\ell_1})$ and $\pi(\mathbf{X}_{\ell_2})$ differ by more than a caliper of width $w$, and otherwise it is the Mahalanobis distance. That is,

$$MP(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}) = \begin{cases} MD(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}), & \text{if } PS(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}) \leq w \\ \infty, & \text{if } PS(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}) > w. \end{cases} \tag{9.11}$$

The width $w$ of the caliper is generally specified as a multiple of the standard deviation of the propensity score, with a value required to obtain balance on the propensity score. Instead of setting $MP(\mathbf{X}_{\ell_1}, \mathbf{X}_{\ell_2}) = \infty$ for violations of the propensity score constraint, it may be more appropriate to add a penalty function to the distance (Rosenbaum, 2002). The matching algorithm will attempt to respect the caliper, but will prefer to slightly violate it for a few matched pairs when the caliper cannot be satisfied for all the pairs.

A matching algorithm attempts to find pairs of subjects based on the defined distance. A common approach is a best-first or greedy algorithm. Let $L_1$ and $L_0$ be the number of treated and control subjects, respectively, and assume $L_1 \leq L_0$. Units under each treatment are first randomly ordered, and the first treated subject is paired with the nearest control subject, then the second treated subject is paired with the nearest of the remaining $L_0 - 1$ control subjects, and so on. A greedy algorithm will not generally find an optimal pair match in the sense of minimizing the total distance within pairs. The key difficulty is that two or more treated units may have the same control as their closest match, and greedy matching resolves this problem arbitrarily. The alternative to greedy matching is optimal pair matching, which can be reduced to finding a flow of minimum cost in a certain network (Rosenbaum, 1989). This is a standard combinatorial optimization problem for which readily available algorithms exist (Bertsekas, 1998).

Finally, there is the choice of the structure of the match. This may be performed using pair- or 1-to-1 matching, matching to a fixed number of $m \geq 2$ controls, or matching with a variable number of controls. The optimal structure for producing similarity within matched sets can be shown to be a *full match* (Rosenbaum, 1991) where some matched sets may contain one treated subject with one or more controls, while other

matched sets may contain multiple treated units with one control. This is intuitive because the flexible arrangement of a full match may group several controls with a single treated subject in regions of the covariate space where controls are vastly more numerous, and similarly, it may group several treated subjects with a single control in regions where treated subjects are relatively plentiful. Also, because a full match includes as special cases all of the other matching structures, it will produce matched sets that are at least as close as those produced by any of those structures.

## 9.5 An application to auto insurance price elasticity estimation

### 9.5.1 The data

The data used for this analysis were extracted from a large database owned by a major Canadian direct insurer. It consists of $L = 230,507$ auto insurance policies that were given a renewal offer between June 2010 and May 2012, and includes more than 60 pre-treatment covariates describing various characteristics of the policy, the vehicle and the driver.

The company sends a renewal package to its customers 45 days prior to the expiry date of the current policy term. This package clearly specifies what the new rate would be for the upcoming policy year, in the event that the customer decides to renew. The new rate could either be lower than, equal to, or higher than the current rate. The treatment is the rate change to which the customer was exposed, computed as the percentage change in premium from the current to the new rate. This is a continuous variable, but for the purpose of this study it was categorized into 5 ordered values $A = \{1, 2, \ldots, 5 = T\}$.

The response variable is the renewal outcome of the policy (renewed or lapsed), measured 30 days after the effective date of the new policy term. Up to that point, the customer is guaranteed her money back if she decides to terminate the policy. Table 9.1 shows that the lapse rate

increases with the rate change level[5], as one would expect. In addition, the price sensitivity appears to be higher for price increases than for price decreases. However, as discussed above, differences in lapse rates among groups are not directly comparable, as they might be driven by differences in the covariates.

Table 9.1: Lapse rates by rate change level

| rate change level | rate change (%) | n. obs. | lapse rate (%) |
|---|---|---|---|
| 1 | $[-20, 0)$ | 63,212 | 3.03 |
| 2 | [0, 5) | 44,609 | 3.45 |
| 3 | [5, 10) | 40,455 | 4.44 |
| 4 | [10, 20) | 51,283 | 7.77 |
| 5 | [20, 40] | 30,948 | 14.22 |
| **All** | | **230,507** | **5.92** |

*Note.* This table displays, for each of five rate change levels, the rate change interval, number of policyholders, and lapse rate.

### 9.5.2 Building the model

In this study, we followed the PTL framework described in Section 9.3. Our work has also been influenced by Fahner (2012), but ours is clearly different in the exposition, the specific application, and the details of the model building process.

Our ultimate goal is to obtain estimates of lapse probabilities for all policyholders under each treatment. As each policyholder is only exposed to a single treatment, the rest remain counterfactual. The ideal way to think about the unobserved counterfactual outcomes is that they are missing values, and therefore should be multiply imputed to represent their uncertainty.

[5] In this analysis, the rate change was combined from all the different coverages (third party liability, damage to the car, etc). Additionally, it would be relevant to investigate the potential heterogeneity in price-elasticity from the individual coverages.

We outline below the conceptual steps involved in the estimation process. In short, we first fit a series of lapse probability models, one for each rate change level. We subsequently use propensity scores and matching algorithms to find pairs of policyholders who were exposed to distinct rate change levels, but who are otherwise comparable in terms of their pre-treatment covariates. Having found those pairs, we then use the estimated lapse probability from each subject of the pair to fill in the "missing" counterfactual response for the other subject of the pair. Finally, we fit a "global" model, which allows us to predict price-elasticity under each rate change level and value of the covariates.

---

**Model estimation steps**:

1. *Estimate a lapse model for each individual treatment.* For each treatment $A = a;\ a = \{1, 2, \ldots, 5\}$, obtain an estimate of the lapse probability $\hat{R}_\ell(a)$ by regressing $R_\ell$ on $\mathbf{X}_\ell$ based only on the subjects that received treatment $A = a$. That is, estimate $E[R_\ell|\mathbf{X}_\ell, A]$.

2. *Propensity score analysis and matching.* This step involves:

   a) Given the five treatments ($T = 5$), estimate the propensity scores $\pi(\mathbf{X}_\ell)$ for all $10 = T(T-1)/2$ treatment dichotomies, and identify common support (i.e., overlap) regions. Specifically, given a treatment dichotomy $(j, k)$, estimate $E[Z_{\ell j}|\mathbf{X}_\ell, A \in (j, k)]$.

   b) For each treatment dichotomy $(j, k)$, form pairs of policyholders (one from each treatment) using one of the matching algorithms described in Section 9.4.3.

3. *Infer the counterfactual outcomes from the matched pairs.* Consider a matched pair including subjects $\ell_1$ and $\ell_2$, which have been exposed to treatments $j$ and $k$, respectively. We use the estimate $\hat{R}_{\ell_2}(k)$ to fill in for the counterfactual outcome of subject $\ell_1$ under treatment $k$. Similarly, we use the estimate $\hat{R}_{\ell_1}(j)$ to fill in for the counterfactual outcome of subject $\ell_2$ under treatment $j$. The causal effect of exposing subject $\ell_1$ to treatment $j$ rather than to treatment $k$ can then be obtained by differencing the observed and counterfactual

outcomes between the matched pairs, $\hat{R}_{\ell_1}(j) - \hat{R}_{\ell_2}(k)$. In the case subject $\ell_1$ cannot find a match among the subjects treated with $k$, then the data cannot support causal effect estimates for this subject and treatment dichotomy, at least not without making strong external assumptions involving model-based extrapolation.

4. *Develop a "global model" of the response.* Develop a global model $\hat{\hat{R}}_\ell(a)$, obtained by fitting the estimates $\hat{R}_\ell(a)$ of the observed responses, plus the estimates of a subset of the counterfactual responses (i.e., as far as the overlap situation permits) on the vector of observed characteristics $\mathbf{X}_\ell$ and treatment level $A$. This model allows us to predict the response for each treatment $A$ and value of $\mathbf{X}$. That is, estimate $E[\hat{R}_\ell(a)|\mathbf{X}_\ell, A]$.

---

The idea is that up to step 3, we try to avoid risky extrapolation by restricting inference to the overlap regions only. Only in step 4 may we choose to ignore the overlap structure by inferring the full combinatorial set (all covariates $\mathbf{X}$ and treatments $A$). The inclusion of the counterfactual responses in the estimation of the global model reduces exposure to extrapolation problems. Our experience concurs with others' (Fahner, 2012) that this approach gives more control and insight during the modeling process than trying to fit a global model directly to the observed data points.

### 9.5.3 Propensity score estimates

In practice, the propensity score (9.4) is generally estimated with a logistic regression model. Accurate estimates require inclusion of all variables that simultaneously influence the treatment status and the outcome, and have the correct functional form specification. The balancing property (9.5) of the propensity score is then examined to determine whether refinements to the model are required. This can be accomplished by stratifying the estimated propensity score at the quintiles of its distribution, and then testing whether balance has been achieved for each covariate within each stratum (Rosenbaum and Rubin, 1984). If there are significant differences in the distribution of a covariate between the treatment and comparison

groups within a stratum, then adding higher-order terms or interactions of the covariate may improve balance. Failure to satisfy this condition under all model specifications would allow us to conclude that the treatments do not overlap along all dimensions.

As can be anticipated, moving back and forth between balance statistics and changing the model specification is a monotonous process. In the context of this study, much better results, with considerable less model tuning required, were obtained by estimating the propensity scores using *gradient boosting models (GBMs)* (Friedman, 2002). GBMs estimate the log-odds of treatment assignment, $h(\mathbf{X}_\ell) = \log(\pi(\mathbf{X}_\ell)/(1-\pi(\mathbf{X}_\ell)))$, by iteratively fitting a collection of simple regression trees, and then combining them to produce a "strong" learning algorithm. Fit is measured by the Bernoulli deviance, $-\sum_{\ell=1}^{L}(Z_\ell h(\mathbf{X}_\ell) - \log(1+\exp(h(\mathbf{X}_\ell))))$, with smaller values indicating a better fit. These models have a number of appealing properties for propensity score estimation. First, GBMs offer a general data modeling algorithm that allows for a flexible nonlinear effect of the covariates on the propensity score. Results are invariant under order preserving transformations of covariates, so there is no need to consider functional form revision of the variables (e.g., log, power, squared-root, etc.). Second, since the propensity score is estimated from a sequence of tree-based models, complex interactions are identified within the fitting process. Finally, GBMs offer a built-in variable selection procedure, in the sense that the estimated model does not necessarily use all the covariates. We discuss GBMs and their application to auto insurance ratemaking in Chapter 10.

In fitting propensity score models, it is important to realize that the goal is to obtain estimates of the propensity score that statistically balance the covariates between each treatment dichotomy, rather than one that estimates the true propensity score as accurately as possible. Thus, the model parameters should not be chosen to minimize prediction error, but to maximize covariate balance. The estimated propensity scores may tend to overfit the data, in the sense of producing better covariate balance than would be expected under chance in the data set used to construct the score, but this is not a problem given the objective (Joffe and Rosenbaum, 1999).

In this study, similarly to McCaffrey et al. (2004), GBMs were selected[6] to minimize the *average standardized absolute mean* difference in the covariates *(ASAM)*. For each covariate, we calculated the absolute value of the difference between the mean for the treatment group and the weighted[7] mean for the comparison group, divided by the standard deviation for the treatment group. We subsequently averaged these values across all covariates to obtain the ASAM.

Figure 9.1 shows the distribution of the final fitted propensity scores for each treatment dichotomy. The propensity scores are labeled "Linear propensity scores" to reflect the fact that they are in the log-odds scale. These plots provide a simple, yet powerful diagnostic on the data examined. We note that the overlap between distributions tends to be much higher for rate changes that are closer, relative to those that are farther apart. A key strength of the propensity score method is that it dramatically alerts us to this fact. For example, it is clear that proportionally fewer of the subjects in the treatment dichotomy $(5, 1)$ are similar than those in the dichotomy $(2, 1)$. This suggests that finding appropriate matches will be more difficult in the former dichotomy. Also, in most treatment dichotomies, there are subjects exposed to one rate change level with higher estimated propensity scores relative to the other rate change level, indicating there is a combination of covariate values not appearing in both groups. The next section provides key insights for understanding the nature of the differences in the propensity score distributions.

### 9.5.4 Matching and covariate balance

In this study, we tested the matching algorithms, distance definitions and matching structures described in Section 9.4.3. The best results, in the sense of producing closely matched pairs and balanced matched samples, were obtained by optimal pair matching using the Mahalanobis distance,

[6] Model selection for GBM fundamentally involves selecting the values for two tuning parameters: the depth level of the individual trees fitted at each iteration and the number of fitted trees.

[7] The weights for subject $\ell$ in the comparison group are defined by $w_\ell = \hat{\pi}(\mathbf{X}_\ell)/(1-\hat{\pi}(\mathbf{X}_\ell))$, the odds that a subject with covariates $\mathbf{X}_\ell$ will be assigned to treatment.

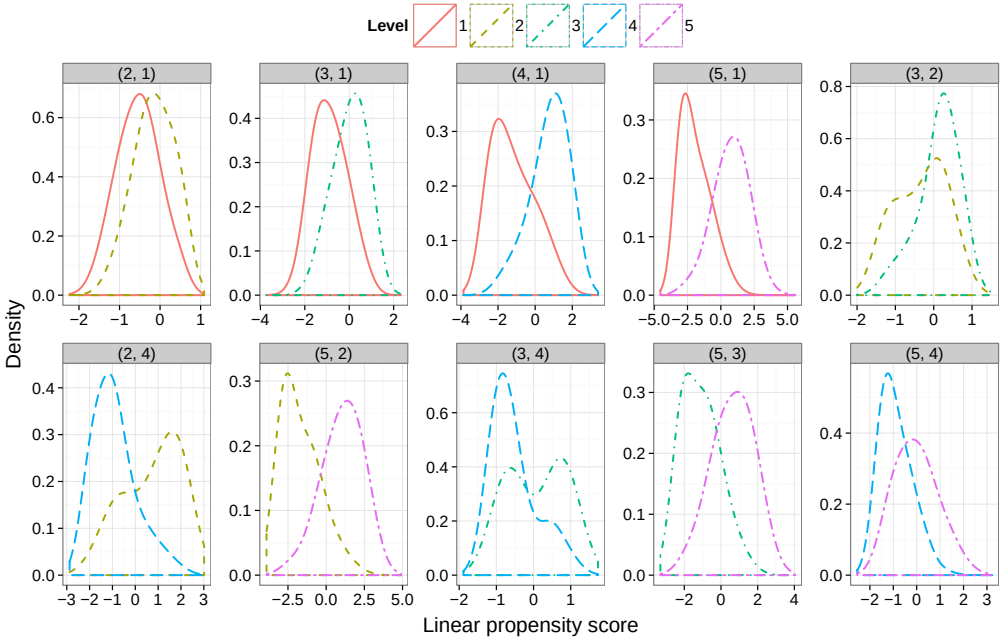


Figure 9.1: Estimated propensity scores for all treatment dichotomies. Given a treatment dichotomy $(j, k)$, each plot illustrates the distribution of the probability of assignment to rate change level $j$ relative to level $k$, conditional on the background covariates. Within each dichotomy, the rate change level with fewer units is represented by $j$, and the other level with $k$. The propensity scores are labeled "Linear Propensity Scores" to reflect the fact that they are in the log-odds scale.

including the propensity score as an additional covariate and propensity score calipers. Specifically, for each treatment dichotomy, we used a minimum-cost network solver, as described in Hansen and Klopfer (2006), to find optimal matched pairs of policyholders (one from each rate change level) such that the sum of the distances (9.11) between the matched pairs is minimized. There is a trade-off between producing closely matched pairs and maximizing the number of matches thus obtained. The width of the propensity score calipers in (9.11) were selected to obtain a good compromise between these two objectives.

Table 9.2 displays the balance results for some of the most important covariates used in estimating the propensity score. Balance is shown for the first four treatment dichotomies before and after matching. Notice that before matching, the means of the covariates differ considerably within each treatment dichotomy. This is more evident for dichotomies with larger rate change differences. This provides insights for understanding the nature of the distributional differences in the propensity scores (see Figure 9.1 in Section 9.5.3). After matching, the differences in the means of the covariates between groups diminished substantially. For instance, in the treatment dichotomy (5,1), we started with $30,948$ subjects treated with $A = 5$ and a comparison group of $63,212$ subjects treated with $A = 1$. Subjects treated with $A = 5$ have, on average, lower current premium (`premium`), are less likely to have a home insurance policy with the company (`home`), less likely to have more than one vehicle (`multi_vehicle`), less likely to have policy discounts through an employer benefit (`group`), less likely to be in the best driving record category (`drv_rec7`), and less likely to have full coverage option (`full_cov`). By design, the matching algorithm required exact matches on "at-fault" accidents during the prior year (`accident`). This is to ensure we are controlling for premium changes resulting from accidents caused by the driver, as opposed to rate changes strictly driven by the company. The matched sample is composed of $26,592$ subjects ($13,296$ from each treatment).

When checking covariate balance, it is important not only to examine differences in means, but to check more general summaries of the distribution. The quantile-quantile (QQ) plot in Figure 9.2 shows a clear improvement in balance for the variable `premium` in the (5,1) treatment pair after matching.

Table 9.2: Balance results of the covariates before and after matching for the first four treatment dichotomies

| | Pair (2,1) | | Pair (3,1) | | Pair (4,1) | | Pair (5,1) | |
|---|---|---|---|---|---|---|---|---|
| | $A = 2$ | $A = 1$ | $A = 3$ | $A = 1$ | $A = 4$ | $A = 1$ | $A = 5$ | $A = 1$ |
| `premium`(Avg.) | | | | | | | | |
| Before | 2,012 | 2,362 | 2,041 | 2,362 | 2,144 | 2,362 | 2,130 | 2,362 |
| After | 2,113 | 2,111 | 2,285 | 2,230 | 2,410 | 2,507 | 2,463 | 2,487 |
| `yrs_lic`(Avg.) | | | | | | | | |
| Before | 25.9 | 22.9 | 25.5 | 22.9 | 24.7 | 22.9 | 22.5 | 22.9 |
| After | 25.0 | 25.2 | 24.0 | 24.2 | 22.3 | 22.5 | 20.8 | 22.0 |
| `home`(%) | | | | | | | | |
| Before | 77.6 | 78.6 | 73.6 | 78.6 | 70.9 | 78.6 | 63.6 | 78.6 |
| After | 77.8 | 79.1 | 74.1 | 75.2 | 71.9 | 73.4 | 69.1 | 71.5 |
| `multi_vehicle`(%) | | | | | | | | |
| Before | 54.8 | 57.2 | 44.4 | 57.2 | 37.9 | 57.2 | 29.6 | 57.2 |
| After | 57.2 | 58.7 | 50.2 | 51.0 | 46.8 | 47.1 | 43.5 | 40.8 |
| `group`(%) | | | | | | | | |
| Before | 12.0 | 14.0 | 12.6 | 14.0 | 8.2 | 14.0 | 6.2 | 14.0 |
| After | 12.2 | 12.3 | 12.0 | 12.3 | 11.5 | 11.1 | 8.8 | 8.4 |
| `drv_rec7`(%) | | | | | | | | |
| Before | 65.8 | 55.6 | 64.1 | 55.6 | 62.1 | 55.6 | 40.8 | 55.6 |
| After | 60.8 | 61.9 | 55.5 | 57.3 | 48.5 | 51.4 | 38.2 | 43.2 |
| `full_cov`(%) | | | | | | | | |
| Before | 93.5 | 89.4 | 90.1 | 89.4 | 84.9 | 89.4 | 70.9 | 89.4 |
| After | 92.8 | 93.0 | 90.1 | 88.7 | 84.1 | 82.0 | 80.4 | 77.5 |
| `accident`(%) | | | | | | | | |
| Before | 1.82 | 1.82 | 2.16 | 1.82 | 2.75 | 1.82 | 10.80 | 1.82 |
| After | 1.90 | 1.90 | 2.20 | 2.20 | 3.22 | 3.22 | 5.17 | 5.17 |
| `lease_flag`(%) | | | | | | | | |
| Before | 13.4 | 13.1 | 11.7 | 13.1 | 10.5 | 13.1 | 8.42 | 13.1 |
| After | 13.4 | 13.6 | 12.2 | 12.6 | 11.5 | 12.4 | 10.8 | 9.9 |
| `veh_age`(Avg.) | | | | | | | | |
| Before | 5.87 | 6.14 | 6.67 | 6.14 | 7.19 | 6.14 | 8.48 | 6.14 |
| After | 5.96 | 5.94 | 6.59 | 6.72 | 7.08 | 7.11 | 7.42 | 7.84 |
| `prop_score`(Avg.) | | | | | | | | |
| Before | 0.465 | 0.377 | 0.507 | 0.315 | 0.655 | 0.280 | 0.642 | 0.175 |
| After | 0.437 | 0.436 | 0.438 | 0.431 | 0.499 | 0.490 | 0.429 | 0.419 |
| **n. obs.** | | | | | | | | |
| Before | 44,609 | 63,212 | 40,455 | 63,212 | 51,283 | 63,212 | 30,948 | 63,212 |
| After | 37,171 | 37,171 | 28,873 | 28,873 | 23,684 | 23,684 | 13,296 | 13,296 |

*Note.* This table displays the mean of the most relevant covariates before and after matching across the first four treatment dichotomies. The differences in the means of the covariates between groups diminished substantially on the matched subjects.

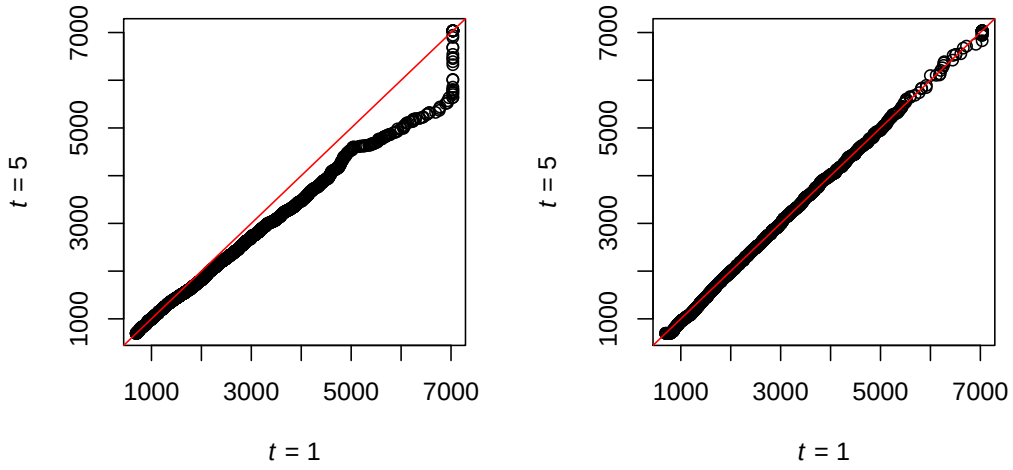


Figure 9.2: Empirical quantile-quantile plot of `premium` before and after matching in the (5,1) treatment dichotomy. This figure displays the quantiles of `premium` for treatment 5 vs. treatment 1 before (left) and after (right) matching. A red 45° reference line is also plotted (indicating perfect balance). Balance for this variable was clearly improved after matching.

### 9.5.5 Price-elasticity functions

Now that we have achieved good balance among all treatment dichotomies, we can proceed to estimate the global model as discussed in the last step of Section 9.5.2. This model allows us to obtain estimates of lapse probabilities under each rate change level $A$ and covariate values $\mathbf{X}$.

We fitted this model using a *generalized additive model* (Hastie and Tibshirani, 1990), with continuous covariates represented by penalized cubic regression splines. The degree of smoothness of model terms was estimated as part of fitting process and selected by *generalized cross-validation (GCV)*. Interaction terms between the rate chance and each covariate were tested and added into the model, guided by the GCV scores as well as domain expertise. This allows for heterogeneity in price elasticities to be estimated at the individual policyholder level. As

previously discussed, understanding the precise nature of individual price sensitivities can be extremely valuable. Each subject's treatment could be optimized on the basis of her individual characteristics.

Having estimated the global model, we then averaged the individual estimates to construct aggregate price-elasticity functions at the portfolio level. This was done for various subpopulations within the insurance portfolio. A subpopulation of the portfolio can be obtained by restricting the values of the covariates to a subset $\omega$. Suppose the portfolio has $L_\omega = |\{\ell : \mathbf{X}_\ell \in \omega\}|$ subjects with covariate values in this subset. The estimated price-elasticity for subpopulation $\omega$ and treatment $A = a$ is defined as $\hat{PE}(\omega, a) = (1/L_\omega) \sum_{\forall \ell : \mathbf{X}_\ell \in \omega} \hat{\hat{R}}_\ell(a)$.

The results are illustrated in Figure 9.3. The plots show the estimated lapse rate measured at each rate change level for the selected subpopulations. For ease of interpretation, continuous covariates were categorized at the quartiles of their distributions (labeled with the numbers 1 to 3 in the plots). There is a clear interaction effect between the rate change level and "at-fault" accidents during the prior year (`accident`). Insureds with recent accidents already expect a rate increase and thus have a lower price sensitivity. Also, as expected, the higher the current premium (`premium`), the higher the price elasticity for a given rate change, but this relation tends to be much stronger with the increase in the rate change level. Similarly, younger policyholders (`age`) and newer customers (`term`) tend to be more price-elastic. All the remaining variables have the expected effect on price elasticity[8]. Overall, price elasticity tends to be higher for rate increases than for rate decreases. A rate increase provides an incentive to shop for an alternative insurer, whereas a rate decrease does inhibit switching but to a lesser extent.

[8] An additional issue is the role of the product understanding in the renewal decision. In some cases, the level of insurance literacy can be limited, potentially influencing decisions.

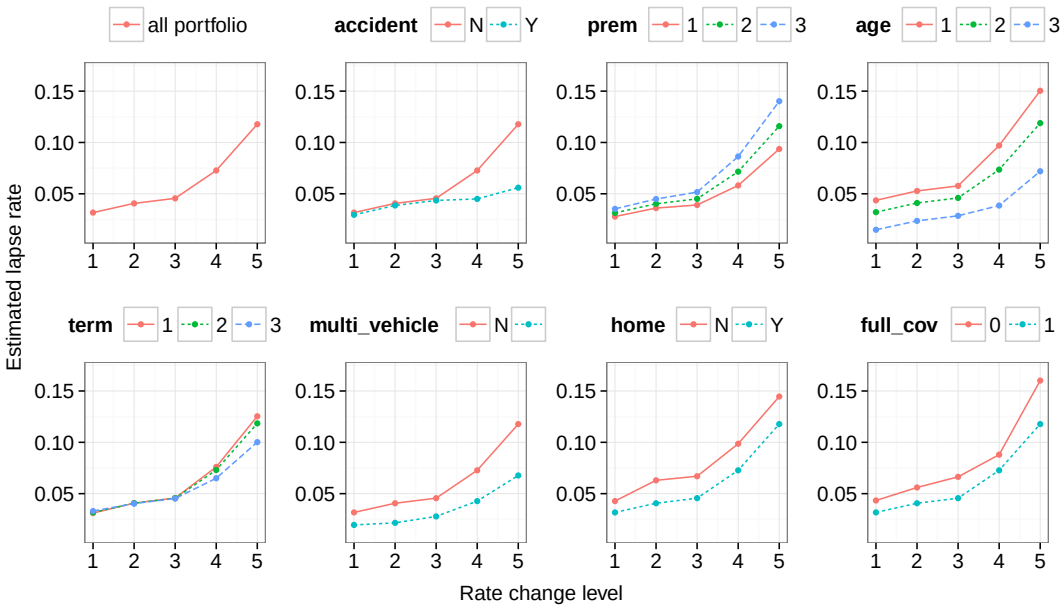


Figure 9.3: Price-elasticity functions. The plots illustrate the average estimated lapse rate measured at each rate change level for selected subpopulations within the insurance portfolio. Continuous covariates have been categorized at the quartiles of their distributions (labeled with the numbers 1 to 3 in the plots).

## 9.6 Managerial implications: Price optimization

In this section, we briefly illustrate an application of the derived estimates of price sensitivities to assist managers in optimizing the expected profit of an insurance portfolio. The question is: which rate change should be applied to each policyholder to maximize the overall expected profit for the company subject to a fixed overall retention rate? By understanding the precise nature of the price elasticities at the policyholder level, the individual rates can be optimized based on each customer's willingness to pay. The causal inference framework used to derive estimates of lapse probabilities at the individual subject level under each rate change scenario allows us to solve this problem effectively.

The problem can be expressed as an integer program. As before, the portfolio is composed of $L$ policyholders, $\ell = \{1, 2, \ldots, L\}$, characterized by a vector of pre-treatment covariates $\mathbf{X}_\ell$. Each subject can be exposed to a rate change level $A = \{1, 2, \ldots, 5 = T\}$, and we let $Z_{\ell a}$ be a binary indicator that takes a value of 1 if subject $\ell$ is exposed to rate change $A = a$ and 0 otherwise. $RC_a$ is the actual rate change associated with treatment $A = a$. The lapse estimates $\hat{\hat{R}}_\ell(a)$ represent the lapse probability of subject $\ell$ if exposed to rate change level $A = a$. In addition, $\text{Prem}_\ell$ is the current premium, $\hat{LR}_{\ell a}$ the predicted loss ratio (i.e., the ratio of the predicted insurance losses relative to premium)[9], and $\alpha$ the overall lapse rate of the portfolio.

The objective function is to maximize the expected profit of the portfolio

$$\underset{Z_{\ell a} \forall \ell \forall a}{\text{Max}} \sum_{\ell=1}^{L} \sum_{a=1}^{T} Z_{\ell a} \left[ \text{Prem}_\ell (1 + RC_a)(1 - \hat{LR}_{\ell a})(1 - \hat{\hat{R}}_\ell(a)) \right] \tag{9.12}$$

[9] Specifically, $\hat{LR}_{\ell a} = \hat{LC}_\ell / \text{Prem}_\ell (1 + RC_a)$, where $\hat{LC}_\ell$ represents the expected loss cost for policyholder $\ell$. The expected loss cost was derived using a variety of qualitative customer attributes with traditional quantitative rating risk factors to accurately predict the likelihood that each policyholder $\ell$ may experience a claim in the future and the expected claim cost. The data supporting this analysis are based on a 5-year exposures and claim experience from the same insurer, with losses including the most recent case reserve estimates. The final loss cost estimates also include an overall base level adjustment for pure IBNR (Incurred But Not Reported) and development of known claims.

subject to the constraints

$$\sum_{a=1}^{T} Z_{\ell a} = 1 \quad \forall \ell, \tag{9.13a}$$

$$Z_{\ell a} \in \{0, 1\}, \tag{9.13b}$$

$$\frac{1}{L} \sum_{\ell=1}^{L} \sum_{a=1}^{T} Z_{\ell a} \hat{\hat{R}}_{\ell}(a) \leq \alpha. \tag{9.13c}$$

Equations (9.13a) and (9.13b) ensure that each policyholder is assigned a rate change level, and (9.13c) ensures that the portfolio has a lapse rate which does not exceeds $\alpha$.

We have solved this optimization problem using the data discussed in Section 9.5.1 along with the estimated lapse probabilities from Section 9.5.5. The results for a sequence of $(1 - \alpha)$ values[10] are illustrated in Figure 9.4. The *efficient frontier* represents the maximum expected profit that this company can obtain at a given desired retention rate. The expected profit is expressed in terms of change, measured in percentage points, relative to the current profit level of the company. This insurer may choose to be at any given point in the efficient frontier depending on its strategic objectives of market share and profitability. However, any point below the efficient frontier is suboptimal in that it is possible to increase profits while maintaining the retention level, or alternatively, increase retention while maintaining profitability. For instance, at the current state, the company may choose to move in the "$A$" direction and increase profits by almost 18%, without sacrificing customer retention. Alternatively, the company may choose to shift in the "$B$" direction and increase retention with no loss in profits. This might be a good strategy if the company is aiming to gain market share. Finally, it may choose to move in the "$C$" direction if the objective is to retain only the most profitable customers. In this sense, the causal effects discussed in this chapter could be adopted as a leading factor in making commercial decisions that keep the portfolio in good shape.

Another consideration is in relation to situations where the insurer

[10] The optimization problem was solved for 8 equally spaced values of $\alpha$, and the results were then interpolated.

may want to limit a certain class of risks in the portfolio, or write more business in certain regions or distribution channels. These situations can be handled by imposing additional constraints in the optimization problem. At the optimum, the company maximizes expected profits subject to such additional constraints.

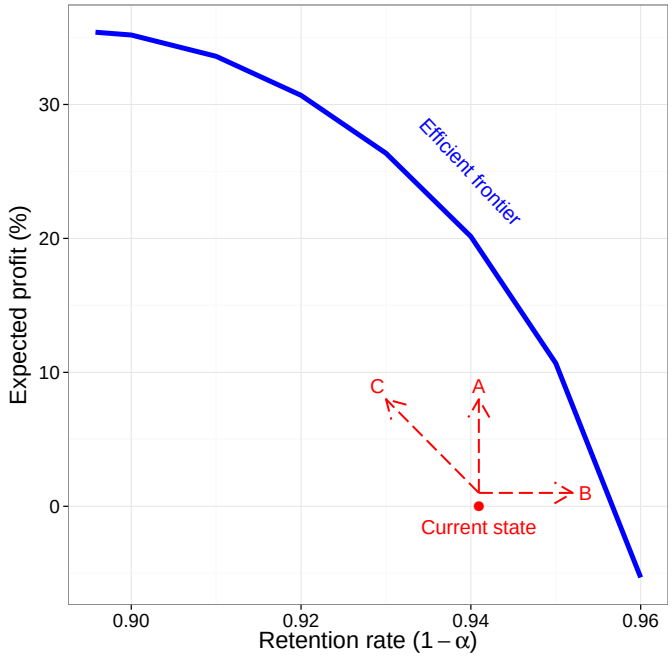


Figure 9.4: Expected profit efficient frontier. The efficient frontier represents the maximum expected profit that the company can obtain at a given desired retention rate. The expected profit is expressed in terms of change, measured in percentage points, relative to the profit at the current state. The current situation for this company is not optimal, in that it is possible to obtain an increase in profit at the current retention level (A), a higher retention at the current profit (B), or even a reduction in client base while increasing profit (C).

## 9.7 Discussion

In this chapter, we have considered a shift in the paradigm for measuring price elasticity in the context of auto insurance, from traditional statistical analysis to a causal inference approach. Price elasticity is ultimately concerned with the effect of a rate change on each policyholder's renewal outcome. The problem that motivates the study is therefore not associational but causal in nature. As each policyholder is exposed to a single rate change level, the rest remain counterfactual. The counterfactual model of causality developed by Rubin represents a useful framework to conceive the price elasticity estimation problem. Under this framework, counterfactuals are thought of as missing values, which are multiply imputed to represent their uncertainty.

Additionally, rate changes reflected in most insurance databases are not the result of a carefully designed experiment, but of a risk-based pricing model. Addressing causal inference questions in the presence of observational data requires appropriate data analysis methods. Conventional analysis based on statistical or algorithmic data models (regression, decision trees, neural nets, etc.), which attempt to fit the observed data points directly, is subject to hidden extrapolation problems that do not raise warning flags. We have shown that the propensity score is a straightforward method that alerts the analyst to inadequately overlapping covariate distributions, for which the data may not support causal conclusions without relying on untestable assumptions about the form of the extrapolation. Further, we have shown that optimal pair matching is a useful method for identifying common support regions from which estimates of the counterfactual renewal outcomes can be derived locally in those regions. These estimates were subsequently used jointly with the estimates of the observed renewal outcomes to obtain a global price-elasticity function. This function allowed us to predict the renewal outcome for the full combinatorial set of rate change levels and covariates.

Besides its role in understanding the precise nature of price elasticities at the individual subject level, our model may assist managers in selecting an optimal rate change level for each policyholder for the purpose of maximizing overall profits. Moreover, valuable insights can be gained by considering the company's current position of market share and prof-

itability relative to the optimal values along the efficient frontier provided by the model. The key managerial decision is then to determine the direction in which the company should move toward the frontier, given that each solution point places a different weight on profitability and market share as objectives.

Some decision model components can be sensitive to the type of insurer. In particular, this type of portfolio analysis is more relevant to a direct insurer than to a brokerage-based insurer. In the latter context, the ability to optimize the rates based on price elasticity considerations is reduced, as the renewal decision is not necessarily driven by the client alone, but is likely to be influenced by the broker, possibly due to the commission rates offered by various competitors. Another consideration is in relation to individuals with insurance plans from their employment agreements. The applicability of the proposed model in this case is highly dependent on the regulatory pricing environment. For instance, if instead of having a single employee discount, the regulation allows for different discount levels based on price elasticity considerations, then the model still applies.

Although the methodology may prove more involved compared to the conventional approach, it offers rigorous analysis of causal effects from non-experimental data. We hope this analysis will stimulate more appreciation for the importance of causal inference, and its relevance for price elasticity estimation in an insurance context.

# 10 Loss cost considerations

## 10.1 Introduction

Most of the personalized treatment learning (PTL) applications described in this thesis are related to insurance. In this setting, the selection of the optimal personalized treatment requires not only the estimation of personalized treatment effects (PTEs), but also the estimation of the expected insurance losses for each individual policyholder within the portfolio. For instance, insurance loss cost was considered when selecting the optimal treatment in the context of cross-selling in Chapter 8, and also in the context of price elasticity and optimization in Chapter 9. In this chapter, we build on the concept of loss cost estimation in non-life insurance and propose a novel application of gradient boosting trees for insurance loss cost modeling and prediction.

*Gradient boosting (GB)* (see Friedman et al., 2000; Friedman, 2001, 2002) is an iterative algorithm that combines simple parametrized functions with "poor" performance (high prediction error) to produce a highly accurate prediction rule. In contrast to other statistical learning methods that usually provide comparable accuracy (e.g., neural networks and support vector machines), GB gives interpretable results, while requiring little data preprocessing or tuning of the parameters. The method is highly robust to less than clean data and can be applied to classification or regression problems from a variety of response distributions (Gaussian, Bernoulli, Poisson, and Laplace). Complex interactions are modeled simply, missing values in the predictors are managed almost without loss of information, and feature selection is performed as an integral part of the procedure. These properties make GB a good candidate for insurance loss cost modeling. However, to the best of our knowledge, the

application of this method to insurance pricing has not been explored to date. This chapter presents the theory of GB and its application to the problem of predicting auto "at-fault" accident loss cost using data from a major Canadian insurer. The predictive accuracy of the model is compared against the conventional *generalized linear model (GLM)* approach.

We first introduce the concept of pricing in non-life insurance and formalize it as a predictive learning problem. The core of the chapter follows, comprising a detailed description of gradient boosting trees. We next describe an empirical application of this method to auto insurance loss cost modeling and prediction. We conclude with a review of GB as an effective alternative to GLM for this application.

## 10.2 Pricing in non-life insurance

GLMs (McCullagh and Nelder, 1989) are widely recognized as an accepted framework for building pricing models in non-life insurance. These models are based on a traditional approach to statistical modeling which starts by assuming that data are generated by a given stochastic data model (e.g., Gaussian, gamma, Poisson, etc.). There is vast insurance pricing literature on such models (Anderson et al., 2007; Antonio and Valdez, 2012; Brockman and Wright, 1992; Haberman and Renshaw, 1996). They are attractive in the sense of producing interpretable parameters which are combined in a multiplicative fashion to obtain an estimate of *loss cost*, defined here as the portion of the premium which covers losses and related expenses (not including loadings for the insurance company's expenses, premium taxes, contingencies, and profit margins). Model validation is usually done using goodness-of-fit tests and residual examination.

In the past two decades, the rapid development in computation and information technology has created an immense amount of data. The field of statistics has been revolutionized by the creation of new tools that have helped analyze the increasing size and complexity in data structures. Most of these tools originated from the *algorithmic modeling* culture as opposed to the *data modeling* culture (Breiman, 2001b). In contrast to data modeling, algorithmic modeling does not assume any specific model for the data, but treats the data mechanism as unknown. As a result,

algorithmic models significantly increase the class of functions that can be approximated relative to data models. They are more efficient in handling large and complex data sets and in fitting nonlinearities to the data. Model validation is measured by the degree of predictive accuracy and this objective is usually emphasized over producing interpretable models. It is probably this lack of interpretability in most algorithmic models that has kept their application to insurance pricing problems very limited so far. Chapados et al. (2001) used several data-mining methods to estimate auto insurance losses, while Francis (2001) illustrates the application of neural networks to insurance pricing problems such as the prediction of frequencies and severities. Kolyshkina et al. (2004) demonstrate the use of *multivariate adaptive regression splines (MARS)* to enhance GLM building.

Among algorithmic models, GB is unique in the sense of achieving both predictive accuracy and model interpretation goals. The latter objective is particularly important in business environments, where models must generally be approved by non-statistically trained decision-makers who need to understand how the output from the "black box" is being produced. Given its other advantageous features, discussed in the preceding section, GB appears highly suited to insurance loss cost modeling.

## 10.3 Predictive learning and boosting

As before, the predictive learning problem can be characterized by a vector of inputs or predictor variables $\mathbf{X} = \{X_1, \ldots, X_p\}$ and an output or target variable $Y$. In this application, the input variables are a collection of quantitative and qualitative attributes of the vehicle and the insured, and the output is the actual loss cost.

Given a collection of $L$ instances $\{(Y_\ell, \mathbf{X}_\ell), \ell = 1, \ldots, L\}$ of known $(Y, \mathbf{X})$ values, the goal is to use these data to obtain an estimate of the function that maps the input vector $\mathbf{X}$ into the values of the output $Y$. This function can then be used to make predictions on instances where only the $\mathbf{X}$ values are observed. Formally, we wish to learn a prediction function $\hat{f}(\mathbf{X}) : \mathbf{X} \to Y$ that minimizes the expectation of some loss function $\Lambda(Y, f)$ over the joint distribution of all $(Y, \mathbf{X})$-values:

$$\hat{f}(\mathbf{X}) = \underset{f(\mathbf{X})}{\operatorname{argmin}} \; E_{Y,\mathbf{X}} \Lambda\Big(Y, f(\mathbf{X})\Big). \tag{10.1}$$

Boosting methods are based on the intuitive idea that combining many "weak" rules to approximate (10.1) should result in classification and regression models with improved predictive performance compared to a single model. A weak rule is a learning algorithm which performs only a little bit better than a coinflip. The aim is to characterize "local rules" relating variables, such as "if an insured characteristic A is present while B is absent, then a claim has high probability of occurring." Although this rule alone would not be strong enough to make accurate predictions for all insureds, it is possible to combine many such rules to produce a highly accurate model. This idea, known as the "the strength of weak learnability" (Schapire, 1990), was originated in the machine learning community with the introduction of *AdaBoost*, which is described in the next section.

## 10.4 AdaBoost

AdaBoost (short for adaptive boosting) is a popular boosting algorithm due to Freund and Schapire (1996). Consider a classification problem with a binary response variable coded as $Y \in \{-1, 1\}$ and classifier $\hat{f}(\mathbf{X})$ taking one of those two values. The Adaboost algorithm is outlined as Algorithm 3. In short, the algorithm generates a sequence of weak classifiers induced on a distribution of weights over the training set. One such weak classifier often used in AdaBoost is a single-split classification tree with only two terminal nodes. Initially, all observation weights are set equally, but on each iteration, the training observations that were misclassified in the previous step receive more weight in the next iteration. Thus, the algorithm is forced in each successive iteration to focus on observations that are difficult to classify correctly. The final classifier is a weighted majority vote of the individual weak classifiers. The weight assigned to each weak classifier gets larger as its weighted error rate measured on the training set gets smaller.

The success of AdaBoost for classification problems was seen as a mysterious phenomenon by the statistics community until Friedman et al.

(2000) demonstrated the connection between boosting and statistical concepts such as additive modeling and maximum likelihood. Their main result is that it is possible to rederive AdaBoost as a method for fitting an additive model in a forward stagewise manner, yielding significant understanding as to why this algorithm tends to outperform a single base model: by fitting an additive model of different and potentially simple functions, it expands the class of functions that can be approximated.

**Algorithm 3** AdaBoost

1: Initialize observation weights $w_\ell = \frac{1}{L}$
2: **for** $b = 1$ to $B$ **do**
3: Fit $f_b(\mathbf{X})$ as the weak classifier on the training data using $w_\ell$
4: Compute the weighted error rate as $err_b = \frac{\sum_{\ell=1}^{L} w_\ell . I(Y_\ell \neq f_b(\mathbf{X}_\ell))}{\sum_{\ell=1}^{L} w_\ell}$
5: Let $\alpha_b = \log((1 - err_b)/err_b)$
6: Update $w_\ell \leftarrow w_\ell . \exp[\alpha_b . I(Y_\ell \neq f_b(\mathbf{X}_\ell))]$, scaled to sum to one $\forall \ell \in \{1, \ldots, L\}$
7: **end for**
8: Output $\hat{f}(\mathbf{X}) = \text{sign}[\sum_{b=1}^{B} \alpha_b . \hat{f}_b(\mathbf{X})]$

## 10.5 Additive models and boosting

Our discussion in this section will be focused on the regression problem, where the output $Y$ is quantitative and the objective is to estimate the mean $E(Y|\mathbf{X}) = f(\mathbf{X})$. The standard linear regression model assumes a linear form for this conditional expectation,

$$E(Y|\mathbf{X}) = f(\mathbf{X}) = \sum_{j=1}^{p} \theta_j X_j. \tag{10.2}$$

An additive model extends the linear model by replacing the linear component $\eta = \sum_{j=1}^{p} \theta_j X_j$ with an additive predictor of the form $\eta = \sum_{j=1}^{p} f_j(X_j)$. We assume

$$E(Y|\mathbf{X}) = f(\mathbf{X}) = \sum_{j=1}^{p} f_j(X_j), \tag{10.3}$$

where $f_1(.), \ldots, f_p(.)$ are smooth functions. There is a separate smooth function $f_j$ for each of the $p$ input variables $X_j$ or, more generally, each component $f_j$ is a function of a prespecified subset of the input variables. These functions are not assumed to have a parametric form, but instead they are estimated in a nonparametric fashion.

This model can be extended by considering additive models with functions $f_b(X)$, $b = \{1, \ldots, B\}$ of potentially all the input variables. In this context,

$$f(\mathbf{X}) = \sum_{b=1}^{B} f_b(\mathbf{X}) = \sum_{b=1}^{B} \theta_b h(\mathbf{X}; \gamma_b), \tag{10.4}$$

where the functions $h(\mathbf{X}; \gamma_b)$ are usually taken to be simple functions characterized by a set of parameters $\gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_B\}$ and a multiplier $\theta_b$. This form includes models such as neural networks, wavelets, multivariate adaptive regression splines and regression trees (Hastie et al., 2009). In a boosting context, $\theta_b h(\mathbf{X}; \gamma_b)$ represents the "weak learner" and $f(\mathbf{X})$ the weighted majority vote of the individual weak learners.

Estimation of the parameters in (10.4) amounts to solving

$$\min_{\{\theta_b, \gamma_b\}_1^B} \sum_{\ell=1}^{L} \Lambda \left( Y_\ell, \sum_{b=1}^{B} \theta_b h(\mathbf{X}_\ell; \gamma_b) \right), \tag{10.5}$$

where $\Lambda(Y, f(\mathbf{X}))$ is the chosen loss function (10.1) to define lack of fit. A "greedy" forward stepwise method solves (10.5) by sequentially fitting a single weak learner and adding it to the expansion of prior fitted terms. The corresponding solution values of each new fitted term are not readjusted as new terms are added into the model. This is outlined in Algorithm 4.

If squared error is used as the loss function, line 3 simplifies to

**Algorithm 4** Forward Stagewise Additive Modeling

1: Initialize $f_0(\mathbf{X}) = 0$
2: **for** $b = 1$ to $B$ **do**
3: Obtain estimates $\theta_b$ and $\gamma_b$ by minimizing $\sum_{\ell=1}^{L} \Lambda(Y_\ell, f_{b-1}(\mathbf{X}_\ell) + \theta h(\mathbf{X}_\ell; \gamma))$
4: Update $f_b(\mathbf{X}) = f_{b-1}(\mathbf{X}) + \theta_b h(\mathbf{X}; \gamma_b)$
5: **end for**
6: Output $\hat{f}(\mathbf{X}) = f_B(\mathbf{X})$

$$\begin{aligned}\Lambda\Big(Y_\ell, f_{b-1}(\mathbf{X}_\ell) + \theta h(\mathbf{X}_\ell; \gamma)\Big) &= \Big(Y_\ell - f_{b-1}(\mathbf{X}_\ell) - \theta h(\mathbf{X}_\ell; \gamma)\Big)^2 \\ &= \Big(r_{\ell b} - \theta h(\mathbf{X}_\ell; \gamma)\Big)^2, \end{aligned} \tag{10.6}$$

where $r_{\ell b}$ is the residual of the $\ell$th observation at the current iteration. Thus, for squared-error loss, the term $\theta_b h(\mathbf{X}; \gamma_b)$ fitted to the current residuals is added to the expansion in line 4. It is also fairly easy to show (Hastie et al., 2009) that the AdaBoost algorithm described in Section 10.4 is equivalent to forward stagewise modeling based on an exponential loss function of the form $\Lambda(Y, f(\mathbf{X})) = \exp(-Y f(\mathbf{X}))$.

## 10.6 Gradient boosting trees

Squared error and exponential error are plausible loss functions commonly used for regression and classification problems, respectively. However, there may be situations in which other loss functions are more appropriate. For instance, binomial deviance is far more robust than exponential loss in noisy settings where the Bayes error rate[1] is not close to zero, or in situations where the target classes are mislabeled. Similarly, the performance of squared error significantly degrades for long-tailed error distributions or the presence of "outliers" in the data. In such situations, other loss functions such as absolute error or Huber loss (Huber, 1964) are more appropriate.

Under these alternative specifications for the loss function and for a

[1] The Bayes error rate represents the lowest achievable error rate for a given classification problem (Duda et al., 2001, Section 2.3).

particular weak learner, the solution to line 3 in Algorithm 4 is difficult to obtain. The gradient boosting algorithm solves the problem using a two-step procedure which can be applied to any differentiable loss function. The first step estimates $\gamma_b$ by fitting a weak learner $h(\mathbf{X};\gamma)$ to the negative gradient of the loss function (i.e., the "pseudo-residuals") using least squares. In the second step, the optimal value of $\theta_b$ is determined given $h(\mathbf{X};\gamma_b)$. The procedure is shown in Algorithm 5.

**Algorithm 5** Gradient Boosting

1: Initialize $f_0(\mathbf{x})$ to be a constant, $f_0(\mathbf{X}) = \underset{\theta}{\operatorname{argmin}} \sum_{\ell=1}^{L} \Lambda(Y_\ell, \theta)$
2: **for** $b = 1$ to $B$ **do**
3: Compute the negative gradient as the working response

$$r_\ell = -\left[\frac{\partial \Lambda(Y_\ell, f(\mathbf{X}_\ell))}{\partial f(\mathbf{X}_\ell)}\right]_{f(\mathbf{X})=f_{b-1}(\mathbf{X})}, \ \ell = \{1, \ldots, L\}$$

4: Fit a regression model to $r_\ell$ by least squares using the input $\mathbf{X}_\ell$ and get the estimate $\gamma_b$ of $\theta h(\mathbf{X};\gamma)$
5: Estimate $\theta_b$ by minimizing $\Lambda(Y_\ell, f_{b-1}(\mathbf{X}_\ell) + \theta h(\mathbf{X}_\ell;\gamma_b))$
6: Update $f_b(\mathbf{X}) = f_{b-1}(\mathbf{X}) + \theta_b h(\mathbf{X};\gamma_b)$
7: **end for**
8: Output $\hat{f}(\mathbf{X}) = f_B(\mathbf{X})$

For squared-error loss, the negative gradient in line 3 is just the usual residuals, so in this case the algorithm is reduced to standard least squares boosting. With absolute error loss, the negative gradient is the sign of the residuals. Least squares is used in line 4 independently of the chosen loss function.

Although boosting is not restricted to trees, our work will focus on the case in which the weak learners represent a "small" regression tree, since they were proven to be a convenient representation for the weak learners $h(\mathbf{X};\gamma)$ in the context of boosting. In this specific case, the algorithm above is called *gradient boosting trees* and the parameters $\gamma_b$ represent the split variables, their split values and the fitted values at each terminal node of the tree. Henceforth in this paper, the term "gradient boosting"

will be used to denote gradient boosting trees.

## 10.7 Injecting randomness and regularization

Two additional ingredients for the gradient boosting algorithm were proposed by Friedman, namely regularization through shrinkage of the contributed weak learners (Friedman, 2001) and injecting randomness in the fitting process (Friedman, 2002).

The generalization performance of a statistical learning method is related to its prediction capabilities on independent test data. Fitting a model too closely to the training data can lead to poor generalization performance. Regularization methods are designed to prevent "overfitting" by placing restrictions on the parameters of the model. In the context of boosting, this translates into controlling the number of iterations $B$ (i.e., the number of trees) during the training process. An independent test sample or cross-validation can be used to select the optimal value of $B$. However, an alternative strategy, shown to provide better results, relates to scaling the contribution of each tree by a factor $\tau \in (0, 1]$. This implies changing line 6 in Algorithm 5 to

$$f_b(\mathbf{X}) = f_{b-1}(\mathbf{X}) + \tau.\theta_b h(\mathbf{X}; \gamma_b). \tag{10.7}$$

The parameter $\tau$ has the effect of retarding the learning rate of the series, so the series has to be longer to compensate for the shrinkage, but its accuracy is better. It has been shown empirically that small shrinkage factors ($\tau < 0.1$) yield dramatic improvements over boosting series built with no shrinkage ($\tau = 1$) (Friedman, 2001). The trade-off is that lower values of $\tau$ require a larger value of $B$ for the same test error and so computational time increases. A strategy for model selection often used is practice is to set the value of $\tau$ as small as possible (i.e. between 0.01 and 0.001) and then choose $B$ by early stopping.

The second modification introduced in the algorithm was to incorporate randomness as an integral part of the fitting procedure. This involves taking a simple random sample, without replacement, of usually approximately 1/2 the size of the full training data set at each iteration. This sample is then used to fit the weak learner (line 4 in Algorithm 5) and

compute the model update for the current iteration. As a result of this randomization procedure, the variance of the individual weak learner estimates at each iteration increases, but there is less correlation between these estimates at different iterations. The net effect is a reduction in the variance of the combined model. In addition, this randomization procedure has the benefit of reducing the computational demand. For instance, taking half-samples reduces computation by almost 50%.

## 10.8 Interpretation

Accuracy and interpretability are two fundamental objectives of predictive learning. However, these objectives do not always coincide. In contrast to other statistical learning methods providing comparable accuracy (e.g., neural networks and support vector machines), gradient boosting gives interpretable results. An important measure often useful for interpretation is the relative influence of the input variables on the output. As discussed in Chapter 4, for a single decision tree, Breiman et al. (1984) proposed the following measure as an approximation of the relative influence of a predictor $X_j$, $j = \{1, \ldots, p\}$:

$$\hat{I}_j = \sum_{\substack{\text{all splits} \\ \text{on } X_j}} \hat{V}_s, \tag{10.8}$$

where $\hat{V}_s$ is the empirical improvement in the split-criterion as a result of using $X_j$ as a splitting variable at the non-terminal node $s$. For gradient boosting, this relative influence measure is naturally extended by averaging $\hat{I}_j$ over the collection of trees.

Another important interpretation component is given by a visual representation of the partial dependence of the approximation $\hat{f}(\mathbf{X})$ on a subset $\mathbf{X}_c$ of size $c < p$ of the input vector $\mathbf{X}$. The dependency of $\hat{f}(\mathbf{X})$ on the remaining predictors $\mathbf{X}_{\bar{c}}$ (i.e. $\mathbf{X}_{\bar{c}} \cup \mathbf{X}_c = \mathbf{X}$) must be conditioned out. This can be estimated from the training data by

$$\hat{f}(\mathbf{X}_c) = \frac{1}{L} \sum_{\ell=1}^{L} \hat{f}(\mathbf{X}_c, \mathbf{X}_{\ell\bar{c}}). \tag{10.9}$$

Note that this method requires predicting the response over the training

sample for each set of the joint values of $\mathbf{X}_c$, which can be computationally very demanding. However, for regression trees, a weighted transversal method (Friedman, 2001) can be used, from which $\hat{f}(\mathbf{X}_c)$ is computed using only the tree, without reference to the data themselves.

## 10.9 Application to auto insurance loss cost modeling

### 10.9.1 The data

The data used for this analysis were extracted from a large database from a major Canadian insurer. It consists of policy and claim information at the individual vehicle level. There is one observation for each period of time during which the vehicle was exposed to the risk of having an at-fault collision accident. Mid-term changes and policy cancellations would result in a corresponding reduction in the exposure period.

The data set includes 426,838 vehicle-years of earned exposure from January 2006 to June 2009, and 14,984 claims incurred during the same period of time, with losses based on best reserve estimates as of December 2009. The input variables (for an overview, see Table 10.1) were measured at the start of the exposure period, and are represented by a collection of quantitative and qualitative attributes of the vehicle and the insured. The output is the actual *loss cost*, which is calculated as the ratio of the total amount of losses to the earned exposure. In practice, insurance legislation may restrict the usage of certain input variables to calculate insurance premiums. Although our analysis was developed assuming a free rating regulatory environment, the techniques described here can be applied regardless of the limitations imposed by any specific legislation.

For statistical modeling purposes, we first partitioned the data into training (70%) and test (30%) data sets. The training set was used for model training and selection, and the test set to assess the predictive accuracy of the selected gradient boosting model against the generalized linear model. To ensure that the estimated performance of the model, as measured on the test sample, is an accurate approximation of the expected performance on future "unseen" cases, the inception date of

Table 10.1: Overview of loss cost predictors

- *Driver characteristics*
  - DC1. Age of principal operator
  - DC2. Yrs licensed
  - DC3. Age licensed
  - DC4. License class
  - DC5. Gender
  - DC6. Marital status
  - DC7. Prior insurance
  - DC8. Postal code risk score
  - DC9. Insurance lapses
  - DC10. Insurance suspensions
- *Accident and conviction history*
  - AC1. Number of chargeable accidents (last 1-3 yrs)
  - AC2. Number of chargeable accidents (last 4-6 yrs)
  - AC3. Number of non-chargeable accidents (last 1-3 yrs)
  - AC4. Number of non-chargeable accidents (last 4-6 yrs)
  - AC5. Number of driving convictions (last 1-3 yrs)
  - AC6. Accident-benefit claims (last 1-6 yrs)
- *Policy characteristics*
  - PC1. Years since policy inception
  - PC2. Presence of multi-vehicle
  - PC3. Collision deductible
  - PC4. Billing type
  - PC5. Billing status
  - PC6. Rating territory
  - PC7. Presence of occasional driver under 25 yrs
  - PC8. Presence of occasional driver over 25 yrs
  - PC9. Group business
  - PC10. Business origin
  - PC11. Home policy
- *Vehicle characteristics*
  - VC1. Vehicle make
  - VC2. Vehicle purchased new or used
  - VC3. Vehicle leased
  - VC4. Horsepower to weight ratio
  - VC5. Vehicle age
  - VC6. Vehicle price

policies in the test set is posterior to the one of policies used to build and select the model.

Loss cost is usually broken down into two components: *claim frequency* (calculated as the ratio of the number of claims to the earned exposure) and *claim severity* (calculated as the ratio of the total amount of losses to the number of claims). Some factors affect claim frequency and claim severity differently, and therefore we considered them separately. For the claim frequency model, the target variable was coded as binary since only a few records had more than one claim during a given exposure period. The exposure period was treated as an *offset* variable in the model (i.e., a variable with a known parameter of 1).

The actual claim frequency measured on the entire sample is 3.51%. This represents an imbalanced or skewed class distribution for the target variable, with one class represented by a large sample (i.e., the non-claimants) and the other represented by only a few (i.e., the claimants). Classification of data with imbalanced class distribution has proved a significant drawback for the performance attainable by most standard classifier algorithms, which assume a relatively balanced class distribution (Jha et al., 2012; Sun et al., 2007). These classifiers tend to output the simplest hypothesis which best fits the data and, as a result, classification rules that predict the small class tend to be fewer and weaker compared to those that predict the majority class. This may hinder the detection of claim predictors and ultimately decrease the predictive accuracy of the model. To address this issue, we rebalanced the class distribution for the target in the frequency model by resampling the data space. Specifically, we under-sampled instances from the majority class to attain a 10% representation of claims in the training sample. The test sample was not modified and thus contains the original class distribution for the target. In econometrics, this sample scheme is known as *choice-based* or endogenous stratified sampling (Greene, 2003), and it is also popular in the computer science community (Chan and Stolfo, 1998; Estabrooks et al., 2004). The "optimal" class distribution for the target variable based on under-sampling is generally dependent on the specific data set (Weiss and Provost, 2003), and it is usually considered as an additional tuning parameter to optimize based on the performance measured on a validation sample.

The estimation of a classification model from a rebalanced sample can be efficient but will overestimate the actual claim frequency. An appropriate statistical method is required to correct this bias, and several alternatives exist for that purpose. In this application, we used the method of *prior correction*, which fundamentally involves adjusting the predicted values based on the actual claim frequency in the population. This correction is described for the logit model in King and Zeng (2001), and the same method has been successfully used in a boosting application to predict customer churn (Lemmens and Croux, 2006).

### 10.9.2 Building the model

The first choice in building the model involves selecting an appropriate loss function $\Lambda(Y, f(\mathbf{X}))$ as in (10.1). Squared-error loss, $\sum_{\ell=1}^{L} (Y_\ell - f(\mathbf{X}_\ell))^2$, and Bernoulli deviance, $-2\sum_{\ell=1}^{L} (Y_\ell f(\mathbf{X}_\ell) - \log(1 + \exp(f(\mathbf{X}_\ell)))$, were used to define prediction error for the severity[2] and frequency models, respectively. Second, it is necessary to select the shrinkage parameter $\tau$ applied to each tree and the sub-sampling rate, as defined in Section 10.7. The former was set at the fixed value of 0.001 and the latter at 50%. Third, the size of the individual trees $S$ and the number of boosting iterations $B$ (i.e., the number of trees) need to be selected. The size of the trees was selected by sequentially increasing the interaction depth of the individual trees, starting with an additive model (single-split regression trees), followed by two-way interactions, and up to six-way interactions. This was done in turn for the frequency and severity models. For each of these models, we ran 20,000 boosting iterations using the training data set.

A drawback of the under-sampling scheme described in Section 10.9.1 is that we may lose important information from the majority class when under-sampled. To maximize the usage of the information available in

[2] For the severity model, we also tested a squared error loss function on the log-transformed dependent variable, since claim severity has a positively skewed distribution. However, we found no gains in predictive accuracy from this alternative specification in this analysis. We also note that although it is possible to derive squared error loss from the principle of maximum likelihood on the assumption of Gaussian distributed target data, the use of squared error loss does not require the target data to have a Gaussian distribution (Bishop, 1995, p. 197).

the training data, the optimal values of $S$ and $B$ were chosen on the basis of the smallest estimated prediction error from a $K$-fold cross-validation (CV) procedure with $K = 10$, as described in Section 7.5.1. A three-way interaction gave best results in both frequency and severity models. Based on this level of interaction, Figure 10.1 shows the training and CV error as a function of the number of iterations for the severity model. The optimal value of $B$ was set at the level at which the CV error ceases to decrease.

The test data set was not used for model selection purposes, but to assess the generalization error of the final chosen model relative to the GLM approach. The latter model was estimated based on the same training data and using binomial and gamma distributions for the response variables in the frequency and severity models, respectively.

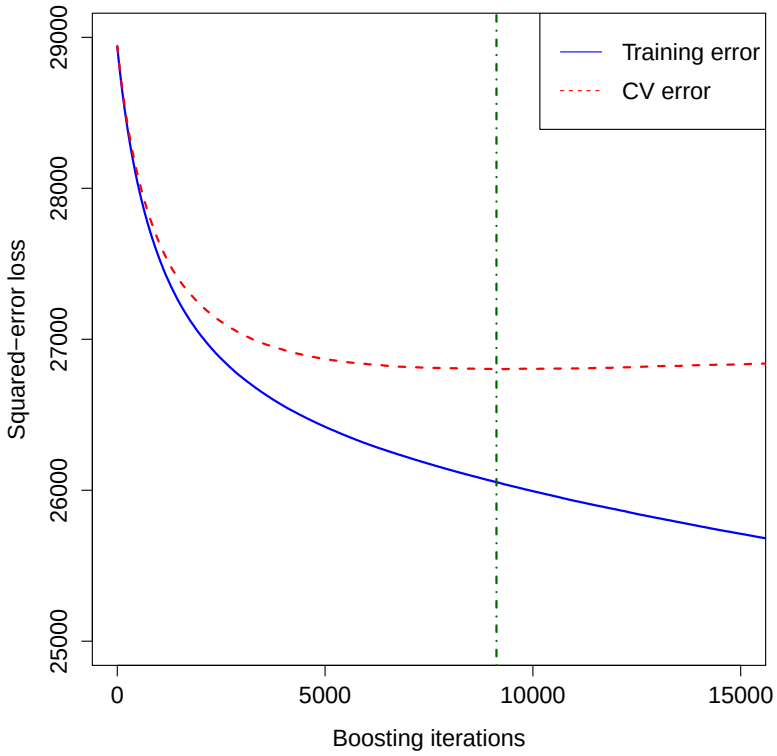


Figure 10.1: The relationship between training and cross-validation error and the optimal number of boosting iterations (shown by the vertical green line).

### 10.9.3 Results

Figure 10.2 displays the relative importance of the 10 most influential predictor variables for the frequency and severity models. The relative importance of each predictor is computed from Equation (10.8). Since these measures are relative, a value of 100 was assigned to the most important predictor and the others were scaled accordingly. There is a clear differential effect between the models. For instance, the number of *years licensed* of the principal operator of the vehicle is the most relevant predictor in the frequency model, while it is far less important in the severity model. Among the other influential predictors in the frequency model, we find the *presence of an occasional driver under 25 yrs*, the *number of driving convictions*, and the *age* of the principal operator. For the severity model, the *vehicle age* is the most influential predictor, followed by the *price of the vehicle* and the *horsepower to weight ratio*.

Partial dependence plots offer additional insights in the way these variables affect the dependent variable in each model. These plots depict the marginal effect of each predictor on claim frequency and severity by averaging out the effect of the other predictors in the model (see Equation 10.9). Figure 10.3 shows the partial dependence plots for the frequency model. The vertical scale is in the log odds and the red marks at the base of each plot show the deciles of the distribution of the corresponding variable. Claim frequency has a nonmonotonic partial dependence on *years licensed*. It decreases over the main body of the data and increases nearly at the end. The partial dependence on *age* initially decreases abruptly up to a value of approximately 30, followed by a long plateau up to 70, when it steeply increases. The variables *vehicle age* (widely recognized as an important predictor in automobile claim frequency models; see Brockman and Wright, 1992) and *postal code risk score* have a roughly monotonically decreasing partial dependence. Claim frequency is also estimated to increase with the number of driving convictions and it is higher for vehicles with an occasional driver under 25 years of age.

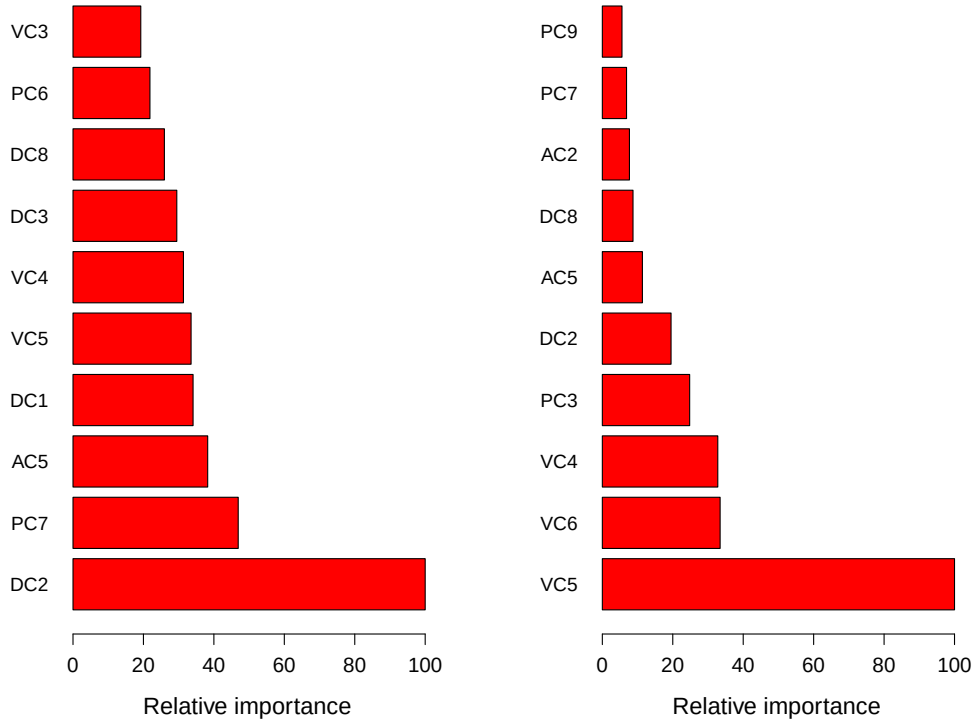


Figure 10.2: Relative importance of the predictors for the frequency (*left*) and severity (*right*) models. See Table 10.1 for predictor variable codes.

Note that these plots are not necessarily smooth, since there is no smoothness constraint imposed on the fitting procedure. This is the consequence of using a tree-based model. If a smooth trend is observed, this is the result of the estimated nature of the dependence of the predictors on the response and it is purely dictated by the data. Also note that no monotonic constraints were imposed on the predictors in this modeling exercise. In other practical situations, it is sometimes desirable to maintain a monotonic relationship between the response and some predictors. Such monotonic constraints can be incorporated within the context of boosting models (Hofner et al., 2011).

Figure 10.4 shows the partial dependence plots for the severity model. The nature of the dependence of *vehicle age* and *price of the vehicle* is naturally due to the fact that newer and more expensive cars would cost more to repair in the event of a collision. The shape of these curves is fairly linear over the vast majority of the data. The variable *horsepower to weight ratio* measures the actual performance of the vehicle's engine. The upward trend observed in the curve is anticipated, since drivers with high performance engines will generally drive at a higher speed. All the remaining variables have the expected partial dependence effect on claim severity.

An interesting relationship is illustrated in Figure 10.5, which shows the joint dependence between *years licensed* and *horsepower to weight ratio* on claim severity. There appears to be an interaction effect between these two variables. Claim severity tends to be higher for low values of *years licensed*, but this relation tends to be much stronger for high values of *horsepower to weight ratio*.

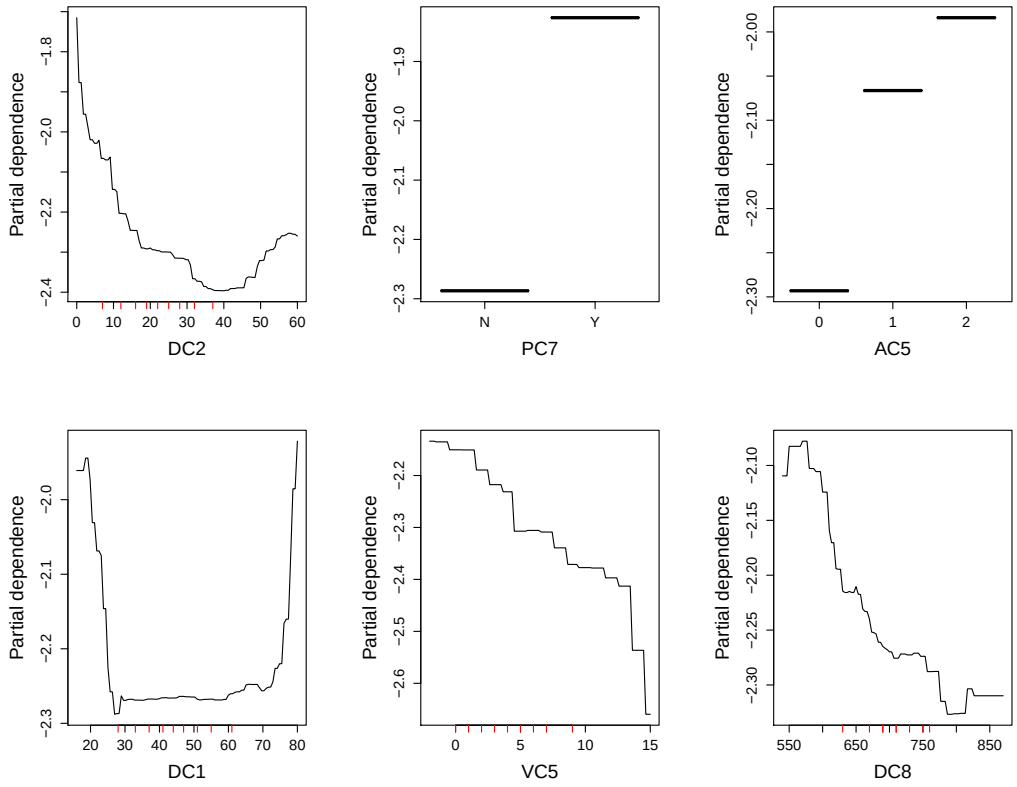


Figure 10.3: Partial dependence plots (frequency model).

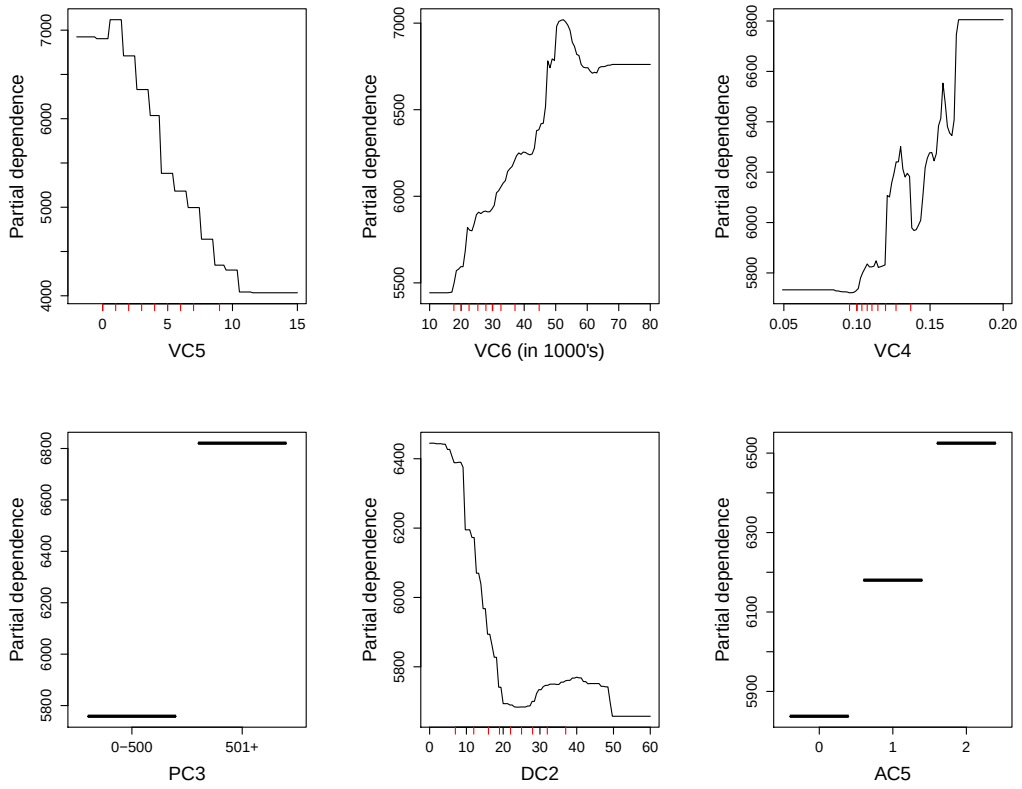


Figure 10.4: Partial dependence plots (severity model).

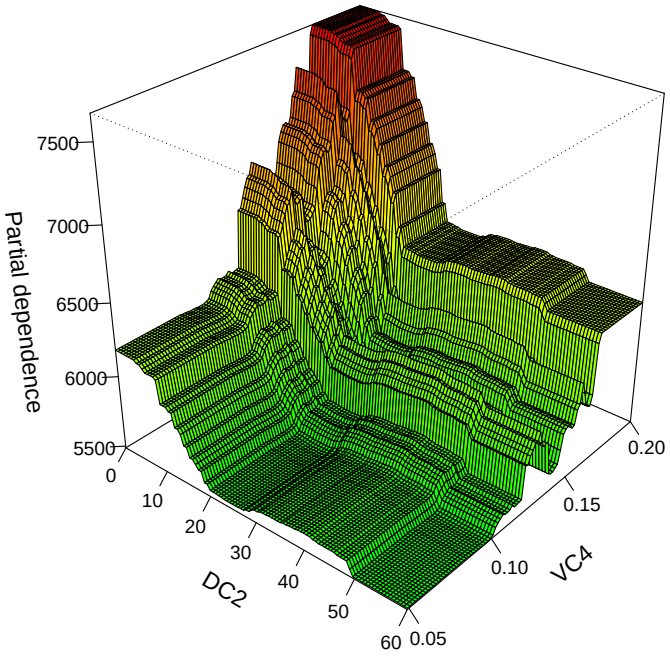


Figure 10.5: Partial dependence of claim severity on *years licensed* (DC2) and *horsepower to weight ratio* (VC4).

We next compare the predictive accuracy of GB against the conventional GLM approach based on the test sample. This was done by calculating the ratio of the rate we would charge based on the GB model to the rate we would charge based on the GLM. Then we grouped the observations into five fairly equally sized buckets ranked by the ratio. Finally, for each bucket we calculated the GLM-loss ratio, defined as the ratio of the actual losses to the GLM predicted loss cost. Figure 10.6 displays the results. Note that the GLM-loss ratio increases whenever the GB model would suggest charging a higher rate than the GLM. The steep upward trend in the GLM-loss ratio curve indicates the higher predictive performance of GB relative to GLM.

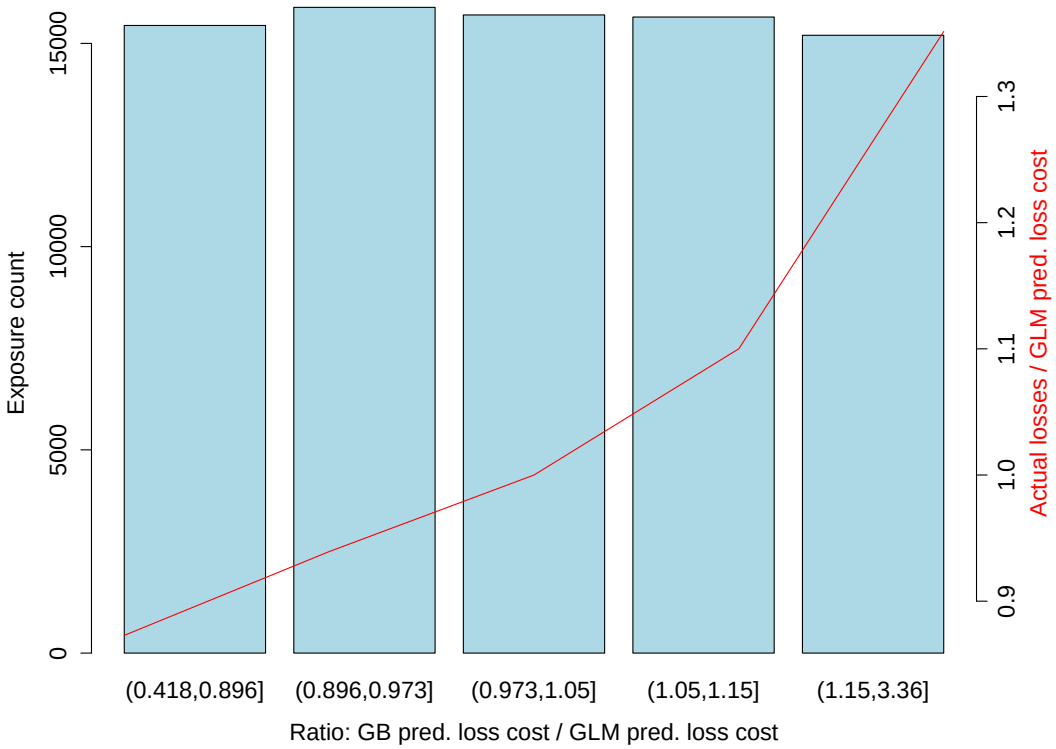


Figure 10.6: Prediction accuracy of GB relative to GLM (based on test sample).

## 10.10 Discussion

In this chapter, we have described the theory of gradient boosting (GB) and its application to the analysis of auto insurance loss cost modeling. GB was presented as an additive model that sequentially fits a relatively simple function (weak learner) to the current residuals by least squares. The most important practical steps in building a model using this methodology have been described.

Estimating loss cost involves solving regression and classification problems with several challenges. The large number of categorical and numerical predictors, the presence of nonlinearities in the data and the complex interactions among the inputs is often the norm. In addition, data might not be clean and/or may have missing values for some predictors. GB fits this data structure very well. First, based on the sample data used in this analysis, the level of accuracy in prediction was shown to be higher for GB relative to the conventional generalized linear model approach. This is not surprising since GLMs are, in essence, relatively simple linear models and are thus constrained by the class of functions they can approximate. Second, as opposed to other nonlinear statistical learning methods such as neural networks and support vector machines, GB provides interpretable results via the relative influence of the input variables and their partial dependence plots. This is a critical aspect to consider in a business environment, where models usually must be approved by non–statistically trained decision makers who need to understand how the output from the "black box" is being produced. Third, GB requires very little data preprocessing which is one of the most time consuming activities in a data mining project. Lastly, model selection is done as an integral part of the GB procedure, and so it requires little "detective" work on the part of the analyst.

In short, gradient boosting is a good alternative method to generalized linear models for building insurance loss cost models. The package **gbm** implements gradient boosting methods under the R statistical computing environment (Ridgeway, 2007).

# 11 Software: The uplift R package

## 11.1 Introduction

The software package **uplift** (Guelman, 2014) is the first integrated R package (R Core Team, 2013) that enables the user to build personalized treatment learning (PTL) models (also called uplift models). The package implements a variety of methods for fitting PTL models, including most of those described in Chapters 3, 4, and 5. In addition, it incorporates specialized functions for exploratory data analysis (EDA), assessing model performance, profiling fitted models, and simulating data. **uplift** is available from the CRAN (Comprehensive R Archive Network) repository at `http://www.cran.r-project.org/package=uplift`. In this chapter, we provide an overview of the package functionality through various examples with synthetic and real-world data.

The chapter is organized as follows. We start by creating a simulated dataset in Section 11.2, used in some of the examples. Section 11.3 describes various useful tools for exploring data. In Section 11.4, we demonstrate the usage of the package for fitting PTL models. Section 11.5 describes model assessment and profiling. We finalize the chapter with a real case study on a direct mail campaign from an international bank.

## 11.2 Creating simulated datasets

The function `sim_pte` allows the user to simulate data according to specification (6.1). The properties of the simulated dataset are customizable, but `sim_pte` uses the following default arguments in the function call:

```
> sim_pte(n = 1000, p = 20, rho = 0, sigma = sqrt(2),
+     beta.den = 4)
```

These input arguments control the following parameters:

- `n`: number of observations (parameter $L$ in (6.1)).
- `p`: number of predictors (parameter $p$ in (6.1)).
- `rho`: covariance between predictors (parameter $\rho$ in (6.1)).
- `sigma`: magnitude of noise (parameter $\sigma_0$ in (6.1)).
- `beta.den`: size of main effects relative to interaction effects ($\eta_j = (-1)^{(j+1)} I(3 \leq j \leq 10)$/`beta.den` in (6.1)).

Throughout this chapter, we will use the dataset `simdata` based on the following simulation settings:

```
> library(uplift)
> set.seed(1)
> simdata <- sim_pte(n = 1000, p = 20, rho = 0.2,
+     sigma = sqrt(2), beta.den = 1)
```

This models a situation with 20 covariates, but only the first 4 interact with the treatment. Also, the total variability in the response from the main effects is four times as big as that from the treatment heterogeneity effects. A quick inspection of `simdata` gives the following:

```
> head(simdata, 2)
  y treat      X1     X2      X3      X4      X5      X6
1 0     1 -0.3092 2.3359 -0.5295 0.62640 -0.4921  0.2818
2 0     1 -0.9688 0.4199 -0.3191 0.07015  1.0181 -1.1715
      X7      X8     X9      X10    X11    X12      X13
1 1.3147 -0.8925 0.8024  0.30315 1.7406 1.1887 -0.45917
2 0.1781 -0.4389 0.3376 -0.07075 0.0724 0.9971  0.08526
......
```

This is a data frame including the binary response variable `y` $\in \{0, 1\}$, the treatment indicator `treat` $\in \{-1, 1\}$, the predictor variables `X`, and the "true" treatment effect score `ts` defined in (6.2) (not shown). We show below the average response rate for `treat`$= -1$ and `treat`$= 1$ (control and treatment groups, henceforth), giving 53.0% and 53.4%, respectively. Thus, based on this simulation setting, the average treatment effect (ATE, defined in Equation 2.1) is negligible. Our goal is to estimate the personalized treatment effect (PTE) as this quantity varies across observations.

```
> prop.table(table(simdata$y, simdata$treat), 2)

      -1     1
  0 0.470 0.466
  1 0.530 0.534
```

## 11.3 Exploratory data analysis

An important aspect about the data, which we may want to check first, is whether the treatment has been randomly assigned to the observational units, or if there was some sort of underlying non-random mechanism by which subjects were exposed to treatment. The `checkBalance` function can be used for this purpose. This function is simply a wrapper for `xBalance` from the **RItools** package (Bowers et al., 2010). Given covariates, a treatment variable, and (optionally) a stratifying factor, it calculates standardized mean differences along each covariate, and tests for conditional independence of the treatment variable and the covariates.

Under randomization, treatment and control groups should be approximately similar (i.e., balanced) in their distributions of covariates. However, one should still expect imbalances on observed covariates to occur in a randomized setting – in fact, 1 out of 20 covariates should differ at the 0.05 level by chance alone. The PTL models implemented by **uplift** are designed for experimental data. In the presence of observational data, the analyst should use the appropriate methods discussed in Chapter 9 to first create a balanced sample.

We show below the balance properties for the first five covariates and a test for the null hypothesis of overall balance of covariates against the alternative of lack of balance. This test follows a chi-squared distribution and the reported $P$ value is 0.709, so there is little evidence against the null of balance (as it should be in this case, given the construction of the artificial data).

```
> simdata$treat <- ifelse(simdata$treat ==
+     1, 1, 0)
> balForm <- as.formula(paste("treat ~", paste("X",
+     1:20, sep = "", collapse = "+")))
> cb <- checkBalance(balForm, data = simdata)
> cb$results[1:5, , ]
    stat
vars   treat=0    treat=1 adj.diff adj.diff.null.sd
  X1 -0.019586  0.0169046  0.03649          0.06628
  X2  0.002043  0.0522843  0.05024          0.06350
  X3 -0.011385 -0.0426792 -0.03129          0.06083
  X4 -0.033873  0.0007947  0.03467          0.06265
......
> cb$overall
        chisquare df p.value
unstrat     16.11 20  0.7098
```

Next, we explore potentially useful PTE predictors. The function `explore` provides a basic exploratory tool by computing the average value of the response variable for each predictor and treatment type. We illustrate below the usage of `explore` on two of the covariates. This function accepts a formula expression of the form "response $\sim$ predictors". A special term of the form `trt()` must be used in the model equation to identify the treatment variable.

The result is a list of matrices, one for each variable. The first two columns display the number of responses in the control and treatment groups, the next two columns show the average response for the control and treatment groups, and the last column shows the uplift (difference

between treatment and control average response). By default, continuous predictors are binned into quartiles, but this can be easily changed using the argument `nbins` in the function call. As expected, since $X_1$ is a treatment heterogeneity effect, but $X_{10}$ is not, this is reflected in the magnitude of uplift values over the range of each of these predictors.

```
> explore(y ~ X1 + X10 + trt(treat), data = simdata)
$X1
                 N(Treat=0) N(Treat=1) Ybar(Treat=0)
[-3.18,-0.718]          127        123        0.5906
(-0.718,-0.0071]        120        130        0.5917
(-0.0071,0.715]         138        112        0.5145
(0.715,3.23]            115        135        0.4174
                 Ybar(Treat=1)  Uplift
[-3.18,-0.718]          0.4553 -0.1353
(-0.718,-0.0071]        0.5154 -0.0763
(-0.0071,0.715]         0.5536  0.0391
(0.715,3.23]            0.6074  0.1900

$X10
                  N(Treat=0) N(Treat=1) Ybar(Treat=0)
[-3.7,-0.659]            127        123        0.4094
(-0.659,-0.00775]        124        126        0.4435
(-0.00775,0.65]          121        129        0.6116
(0.65,3.46]              128        122        0.6562
                  Ybar(Treat=1)  Uplift
[-3.7,-0.659]            0.4228  0.0133
(-0.659,-0.00775]        0.4683  0.0247
(-0.00775,0.65]          0.5349 -0.0767
(0.65,3.46]              0.7131  0.0569
```

An additional useful function for exploratory data analysis in the context of PTL models is the *net information value* (Larsen, 2009). Variable selection is an important step in applications of statistical and machine learning methods. The predictive performance of these methods tends to degrade if the dimensionality of the input space is higher than optimal (Kohavi and John, 1997). This is due to the *curse of dimensionality* (Bellman,

1961), which refers to the extent to which increasing the dimensionality of the input space leads to a point where data is very sparse, providing a poor sample representation. The same problem is present in PTL models. In addition, these models have an additional complexity for variable selection due to the treatment heterogeneity effects usually being a small fraction of the main effects. Variable selection not only aids in improving the predictive performance of the model, but also provides a better understanding of the underlying process generating the data.

An approach to variable selection used by conventional statistical learning models is based on information-theoretic criteria, which rely on empirical estimates of the mutual information between each variable and the target. Let $Y \in \{0, 1\}$ be a binary response variable and $\mathbf{X} = (X_1, \ldots, X_p)^\top$ a vector of baseline predictors. The *weight of evidence* (Siddiqi, 2006) has its roots in the logit transform, or equivalently called the log-odds:

$$\text{logit}\Big(P(Y|\mathbf{X})\Big) = \log\left(\frac{P(Y=1|\mathbf{X})}{P(Y=0|\mathbf{X})}\right). \tag{11.1}$$

From Bayes' theorem, (11.1) can be rewritten as

$$\begin{aligned}\log\left(\frac{P(Y=1|\mathbf{X})}{P(Y=0|\mathbf{X})}\right) &= \log\left(\frac{P(Y=1)P(\mathbf{X}|Y=1)}{P(Y=0)P(\mathbf{X}|Y=0)}\right) \\ &= \log\left(\frac{P(Y=1)}{P(Y=0)}\right) + \log\left(\frac{P(\mathbf{X}|Y=1)}{P(\mathbf{X}|Y=0)}\right). \end{aligned} \tag{11.2}$$

The second term in (11.2) is defined as the *weight of evidence (WOE)*, which can be re-expressed as

$$WOE = \log\left(\frac{P(Y=1|\mathbf{X})}{P(Y=0|\mathbf{X})}\right) - \log\left(\frac{P(Y=1)}{P(Y=0)}\right). \tag{11.3}$$

The right hand side of (11.3) is a measure of the difference between the log odds after the observation of $\mathbf{X}$ and before that observation. A $WOE > 0$ ($WOE < 0$) means that the odds given $\mathbf{X}$ is higher (lower) than

the overall odds by a factor of $\exp(WOE)$. A $WOE$ value of 0 means average odds.

The *information value (IV)* is a measure of the predictive power of a predictor, commonly used in credit risk scorecard applications (Anderson, 2007). It is defined as

$$\begin{aligned} IV &= \sum_{\mathbf{X}} \Big( P(\mathbf{X}|Y=1) - P(\mathbf{X}|Y=0) \Big) \log \left( \frac{P(\mathbf{X}|Y=1)}{P(\mathbf{X}|Y=0)} \right) \\ &= \sum_{\mathbf{X}} \Big( P(\mathbf{X}|Y=1) - P(\mathbf{X}|Y=0) \Big) \, WOE. \end{aligned} \tag{11.4}$$

Equation (11.4) can be derived from the Kullback-Leibler (Cover and Thomas, 1991, p. 19) measure of divergence associated with two probability distributions $Q = P(X|Y=1)$ and $R = P(X|Y=0)$, defined in (4.1). It is given by

$$D(Q||R) = \sum_{\mathbf{X}} Q \, \log \left( \frac{Q}{R} \right). \tag{11.5}$$

We see that (11.5) is not symmetric (i.e., $D(Q||R) \neq D(R||Q)$). The $IV$ is its symmetric version, also known as the $J$-divergence (Jeffreys, 1946; Kullback and Leibler, 1951). It is given by

$$J(Q||R) = D(Q||R) + D(R||Q) = \sum_{\mathbf{X}} (Q - R) \, \log \left( \frac{Q}{R} \right). \tag{11.6}$$

A natural extension of the WOE for PTL is to consider the $J$-divergence between $Q$ and $R$ from both treatment and control groups. Specifically, let $A \in \{0, 1\}$ be a binary treatment and let $Q^T = P(\mathbf{X}|Y=1, A=1)$, $Q^C = P(\mathbf{X}|Y=1, A=0)$, $R^T = P(\mathbf{X}|Y=0, A=1)$, and $R^C = P(\mathbf{X}|Y=0, A=0)$. The *net information value (NIV)* is given by

$$
\begin{aligned}
NIV &= J(Q^T/R^T \;||\; Q^C/R^C) \\
&= D(Q^T/R^T \;||\; Q^C/R^C) + D(Q^C/R^C \;||\; Q^T/R^T) \\
&= \sum_{\mathbf{X}} Q^T/R^T \log\left(\frac{Q^T/R^T}{Q^C/R^C}\right) + \sum_{X} Q^C/R^C \log\left(\frac{Q^C/R^C}{Q^T/R^T}\right) \\
&= \sum_{\mathbf{X}} (Q^T/R^T - Q^C/R^C)\Big[\log(Q^T/R^T) - \log(Q^C/R^C)\Big] \\
&\propto \sum_{\mathbf{X}} (Q^T R^C - Q^C R^T) NWOE, \qquad (11.7)
\end{aligned}
$$

where $NWOE = \log(Q^T/R^T) - \log(Q^C/R^C)$ is the *net weight of evidence*, defined as the difference in the weight of evidence between the treatment and control groups.

Larsen (2009) also suggests assessing the stability of the NIV over the training data and computing an *adjusted net information value (ANIV)*. Our method for computing the ANIV is as follows:

1. Draw $B$ bootstrap samples from the training data and compute the NIV for each variable in each sample.

2. Compute the mean, $\overline{NIV}$, and standard deviation, sd($NIV$), of the NIV for each variable over the $B$ replications.

3. The ANIV for a given variable is computed by subtracting a penalty term from the mean NIV. Specifically, $ANIV = \overline{NIV} - \frac{\text{sd}(NIV)}{\sqrt{B}}$.

The function `niv` computes the net information value and the adjusted net information value. We illustrate below the usage of this function on our simulated dataset. The function produces a list with two components:

- `niv_val`: a matrix with the following columns: `niv` (the average net information value for each variable over all bootstrap samples), `penalty` (the penalty term), and `adj_niv` (the adjusted net information value).

- `nwoe`: a list of matrices with the net weight of evidence for each variable. As an example, this is shown below for the variable $X_3$ only.

```
> modelForm <- as.formula(paste("y ~", "trt(treat) +",
+     paste("X", 1:20, sep = "", collapse = "+")))
> niv_res <- niv(modelForm, B = 50, plotit = FALSE,
+     data = simdata)
> niv_res$niv_val[order(niv_res$niv_val[, 3],
+     decreasing = TRUE), ]
      niv penalty adj_niv
X3  5.303  0.2445   5.059
X1  5.297  0.2541   5.043
X4  4.917  0.2506   4.666
X2  4.609  0.2191   4.389
X7  4.190  0.1698   4.021
......
> niv_res$nwoe$X3[, 5:7]
                 ct1.woe ct0.woe    nwoe
[-2.81,-1.24]     0.4516  0.8607 -0.4091
(-1.24,-0.886]    0.0036  0.6530 -0.6495
(-0.886,-0.545]   0.2067  0.7373 -0.5306
(-0.545,-0.267]  -0.1740 -0.1627 -0.0112
(-0.267,-0.0521] -0.1726  0.4746 -0.6471
......
```

## 11.4 Fitting personalized treatment learning models

In this section, we demonstrate three key methods implemented by **uplift** for building PTL models. We fit these methods relying mostly on their default options. Model tuning is discussed in Section 11.6.

We first fit an *uplift random forest* to `simdata` using the function `upliftRF`. This is a tree-based model with split criteria specifically designed to estimate PTEs (see Chapter 4 for details).

```
> upliftRF_fit <- upliftRF(modelForm, data = simdata,
+     split_method = "ED")
```

This simple form of `upliftRF` hides a number of important options which have been set to their default values, including:

- `mtry`: the number of variables to be tested in each node; the default is `floor(sqrt(ncol(x)))`.
- `ntree`: the number of trees to generate in the forest; default is `ntree = 100`.
- `split_method`: the split criterion. Possible values are: `"ED"` (Euclidean distance), `"Chisq"` (Chi-squared divergence), `"KL"` (Kullback-Leibler divergence), `"L1"` ($L1$-norm divergence) and `"Int"` (interaction method).
- `interaction.depth`: The maximum depth of interactions among covariates. 1 implies an additive model, 2 implies a model with up to 2-way interactions, and so on. The default is to grow trees to maximal depth, constrained on the arguments specified by `minsplit` and `minbucket`.
- `bag.fraction`: the fraction of training observations randomly selected for the purpose of fitting each tree in the forest. The default is `bag.fraction = 0.5`.
- `minsplit`: the minimum number of observations that must exist in a node in order for a split to be attempted.
- `minbucket_ct0`: the minimum number of control observations in any terminal node.
- `minbucket_ct1`: the minimum number of treatment observations in any terminal node.
- `keep.inbag`: if set to `TRUE`, an `nrow(x)` by `ntree` matrix is returned, whose entries are the "in-bag" samples in each tree.

- `verbose`: print status messages?

The `summary` function when applied to an object of class `"upliftRF"` returns the function call, a measure of the relative importance of each predictor (shown below), and some additional parameters used in the fitted model.

```
> summary(upliftRF_fit, plotit = FALSE)$importance
   var rel.imp
1   X1   9.385
2   X3   7.933
3   X2   7.703
4   X4   6.987
5  X11   5.686
6  X14   5.536
7  X19   5.346
8   X7   5.236
9   X6   5.130
......
```

We next fit a *causal conditional inference forest* (see Chapter 5 for details), implemented by the function `ccif`. Causal conditional inference trees estimate PTEs by binary recursive partitioning in a causal conditional inference framework. Roughly, the algorithm works as follows:

1. For each terminal node in the tree we test the global null hypothesis of no interaction effect between the treatment and any of the `mtry` covariates selected at random from the set of $p$ covariates. Stop if this hypothesis cannot be rejected. Otherwise select the input variable with strongest interaction effect. The interaction effect is measured by a $P$ value corresponding to a permutation test (Strasser and Weber, 1999) for the partial null hypothesis of independence between each input variable and a transformed response. Specifically, the response is transformed so the impact of the input variable on the response has a causal interpretation for the treatment effect.

2. Implement a binary split in the selected input variable.

3. Recursively repeat steps 1 and 2.

In `ccif`, the independence test between each input and the transformed response is performed by calling `independence_test` from the **coin** package (Hothorn et al., 2008). Additional arguments from this function can be passed to `ccif` via {...}. This is the case below with the argument `distribution = approximate(B=999)`, which returns a Monte Carlo function that draws $B(= 999)$ random permutations of the transformed responses to derive the distribution of the test statistic under the null hypothesis of no interaction between the treatment and each covariate.

Most of the arguments described above for `upliftRF` are also available for `ccif`. The latter incorporates additional arguments which are specifically relevant for this method, including:

- `pvalue`: the maximum acceptable $P$ value required in order to make a split.
- `bonferroni`: apply a Bonferroni adjustment to $P$ value?
- {...}: additional arguments passed to `independence_test{coin}`.

```
> ccif_fit <- ccif(formula = modelForm,
+     data = simdata, split_method = "Int",
+     distribution = approximate(B = 999))
```

We next fit a *causal K-nearest-neighbor*. The essential idea of this method is that when estimating the PTE for a new target observation, we should weight the evidence of similar training observations to that target more heavily (see Section 3.3.3). Thus, we first generate a test sample composed of 10,000 observations based on the same simulation parameters used to create `simdata`.

```
> ### Simulate test data
> set.seed(12345)
> simdata_test <- sim_pte(n = 10000, p = 20,
+     rho = 0.2, sigma = sqrt(2), beta.den = 1)
> simdata_test$treat <- ifelse(simdata_test$treat ==
+     1, 1, 0)
```

We now score the test observations using the training data as follows:

```
> upliftKNN_fit <- upliftKNN(simdata[, 3:22],
+     simdata_test[, 3:22], simdata$y, simdata$treat,
+     k = 30, dist.method = "euclidean", p = 2,
+     ties.meth = "min", agg.method = "mean")
```

The object `upliftKNN_fit` now contains a matrix of predictions for each test case under control and treatment. This is shown below for the first few predictions.

```
> head(upliftKNN_fit)
          0      1
[1,] 0.5000 0.5667
[2,] 0.5667 0.7000
[3,] 0.4000 0.3000
[4,] 0.4000 0.5667
[5,] 0.5333 0.6000
......
```

The arguments used above for the function `upliftKNN` include:

- `k`: number of neighbors considered.
- `dist.method`: the distance used in calculating the neighbors. Any method supported by the function `dist` from the **stats** package is valid.
- `p`: the power of the Minkowski distance.
- `ties.meth`: method to handle ties for the $k$th neighbor. The default is `min` which uses all ties. Alternatives include `max` which uses none if there are ties for the $k$th nearest neighbor, `random` which selects among the ties randomly, and `first` which uses the ties in their order in the data.
- `agg.method`: method to combine responses of the nearest neighbors, defaults to `"mean"`. The alternative is `"majority"`.

In addition, currently **uplift** incorporates two additional functions to facilitate the data preprocessing steps required for fitting models based on the *modified covariate method* (Section 3.3.1) and the *modified outcome method* (Section 3.3.2). The corresponding functions are `tian_transf` and `rvtu`, respectively.

## 11.5 Model assessment and profiling

Model assessment for PTL models was discussed in Chapter 7. **uplift** incorporates two functions for assessing models, namely `performance` and `qini`. We start by using the `predict` methods on the `"upliftRF"` and `"ccif"` objects to predict the PTEs on our `simdata_test` dataset. Predictions from the `upliftKNN` were already obtained in the previous section.

```
> ### upliftRF predictions
> upliftRF_pred <- predict(upliftRF_fit, simdata_test)
> ### ccif predictions
> ccif_pred <- predict(ccif_fit, simdata_test)
```

The `predict.ccif` and `predict.upliftRF` methods produce a matrix of predictions containing the conditional class probabilities under each treatment. For illustration, we return the first few predictions from `ccif_pred` below. The first column (`pr.y1_ct1`) represents $P(Y = 1|\mathbf{X}, A = 1)$ and the second (`pr.y1_ct0`) represents $P(Y = 1|\mathbf{X}, A = 0)$.

```
> head(ccif_pred)
     pr.y1_ct1 pr.y1_ct0
[1,]    0.5319    0.5590
[2,]    0.5123    0.5518
[3,]    0.5550    0.5261
[4,]    0.5734    0.4429
[5,]    0.5612    0.5005
......
```

The function `performance` provides a method for assessing model performance. Essentially this function 1) computes the differences in

the predicted conditional class probabilities $P(Y = 1|\mathbf{X}, A = 1)$ and $P(Y = 1|\mathbf{X}, A = 0)$, 2) ranks these differences and groups them into bins with equal number of observations in each, and 3) computes the actual difference in the mean of the response variable between the treatment and control groups. This function returns a matrix with the following columns: the number of groups (`group`), the number of observations in the treated group (`n.ct1`), the number of observations in the control group (`n.ct0`), the number of observations in the treated group where $Y = 1$ (`n.y1_ct1`) (not shown), the number of observations in the control group where $Y = 1$ (`n.y1_ct0`) (not shown), the mean of $Y$ over the treated group (`r.y1_ct1`), the mean of $Y$ over the control group (`r.y1_ct0`), and the difference between `r.y1_ct1` and `r.y1_ct0` (`uplift`).

```
> ccif_perf <- performance(ccif_pred[, 1],
+     ccif_pred[, 2], simdata_test$y, simdata_test$treat)
> ccif_perf[, -c(4, 5)]
      group n.ct1 n.ct0 r.y1_ct1 r.y1_ct0    uplift
 [1,]     1   503   497   0.6441   0.2636  0.380554
 [2,]     2   505   495   0.5703   0.3939  0.176358
 [3,]     3   501   499   0.5449   0.4369  0.108036
 [4,]     4   468   532   0.5363   0.5000  0.036325
 [5,]     5   499   501   0.5251   0.5170  0.008084
......
```

We can compare the top decile uplift (Section 7.4) among the three models from the code below.

```
> upliftRF_perf <- performance(upliftRF_pred[,
+     1], upliftRF_pred[, 2], simdata_test$y,
+     simdata_test$treat)
> upliftKNN_perf <- performance(upliftKNN_fit[,
+     2], upliftKNN_fit[, 1], simdata_test$y,
+     simdata_test$treat)
> top10Decile_perf <- data.frame(ccif = ccif_perf[1,
+     8], upliftRF = upliftRF_perf[1,
+     8], upliftKNN = upliftKNN_perf[1,
```

```
+     8])
> top10Decile_perf
         ccif upliftRF upliftKNN
uplift 0.3806   0.3308    0.2375
```

We can also assess model performance using the *Qini* coefficient, which is a more general performance measure for PTL models (see Section 7.2). The function `qini` computes the Qini coefficient from a `"performance"` object. We can compare the performance of the fitted models using this function as follows:

```
> qini <- data.frame(ccif = qini(ccif_perf,
+     plotit = FALSE)$Qini, upliftRF = qini(upliftRF_perf,
+     plotit = FALSE)$Qini, upliftKNN = qini(upliftKNN_perf,
+     plotit = FALSE)$Qini)
```

```
> qini
    ccif upliftRF upliftKNN
1 0.4788   0.4345    0.4165
```

We see that based on the default arguments used for fitting these models, the causal conditional inference forests performed best, both on top decile uplift and Qini.

Next, we use the function `modelProfile` to profile subjects based on their estimated PTE. This can be useful to inspect the characteristics of subjects with high/low treatment impact. Essentially, this function ranks the PTE predictions supplied in the left hand side of the model formula and classifies them into groups with equal number of observations. It subsequently calls the function `tabular` from the **tables** package (Murdoch, 2013) to compute the average of each numeric predictor and the distribution of each factor within each group. We illustrate this below for a subset of predictors.

```
> pte_pred <- ccif_pred[, 1] - ccif_pred[, 2]
> modelProfile(pte_pred ~ X1 + X2 + X3 + X4 + X5 + X6,
```

```
+     data = simdata_test, groups = 10, group_label = "D",
+     digits_numeric = 1)
```

Table 11.1: Distribution of predictors using `modelProfile`

| | | Group | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | ... | 8 | 9 | 10 | All |
| | n | 1000 | 1000 | 1000 | ... | 1001 | 999 | 1000 | 10000 |
| pte_pred | Avg. | 0.157 | 0.094 | 0.062 | ... | −0.057 | −0.085 | −0.132 | 0.004 |
| X1 | Avg. | 0.649 | 0.500 | 0.326 | ... | −0.248 | −0.394 | −0.605 | 0.002 |
| X2 | Avg. | −0.809 | −0.370 | −0.266 | ... | 0.315 | 0.471 | 0.704 | −0.006 |
| X3 | Avg. | 0.620 | 0.413 | 0.326 | ... | −0.270 | −0.362 | −0.586 | 0.005 |
| X4 | Avg. | −0.601 | −0.243 | −0.129 | ... | 0.172 | 0.263 | 0.455 | −0.006 |
| X5 | Avg. | −0.171 | −0.017 | 0.008 | ... | 0.055 | 0.068 | 0.028 | −0.003 |
| X6 | Avg. | 0.041 | 0.174 | 0.174 | ... | −0.026 | −0.079 | −0.281 | 0.025 |

## 11.6 Case study: A bank's direct mail campaign

In this section, we provide a comprehensive view of all the steps discussed previously in this chapter to conduct an analysis on a real case study using **uplift**. The objective of this case study is to identify which clients from an international bank are more likely to buy one of its financial products as a result of a marketing intervention activity. The data are based on a pilot direct mail campaign implemented by this bank, in which 6,256 clients were randomly assigned in equal proportions to a treatment and a control group. Clients in the treatment group received a promotion to buy a certain product. Clients in the control group did not receive the promotion. In addition to the response variable and the treatment indicator, the dataset (labeled `bankDM2`) includes 13 predictors describing various demographic and behavioral client characteristics.

We first check the response rate for the control and treatment groups, giving 38.4% and 14.8%, respectively. The ATE of 23.6% is significantly higher than what is usually seen in marketing campaigns of this kind. Certainly, the marketing intervention activity generated significant positive effects to buy the product. However, the cost of the promotion was

very high as well, so the company was interested in identifying a subgroup of clients for which the intervention was more effective than the average and targeting those clients in the post-pilot campaign deployment. In addition, the company's budget for the post-pilot campaign limited the quantity of targets to at most 30% of their client base.

```
> 100 * round(prop.table(table(bankDM2$response,
+     bankDM2$treatment), 2), 4)

        0     1
  0 85.17 61.60
  1 14.83 38.40
```

We next randomly partition `bankDM2` into training and test datasets in a 70/30 proportion. The former dataset is used for model building and the latter for performance assessment. We check that the ATE is consistent in the resulting partition[1].

```
> set.seed(455)
> samp.ind <- sample(1:nrow(bankDM2), 0.7 *
+     nrow(bankDM2), replace = FALSE)
> bankDM.train <- bankDM2[samp.ind, ]
> bankDM.test <- bankDM2[-samp.ind, ]
> ### Check average treatment effect is
> ### consistent in the resulting partition
> 100 * round(prop.table(table(bankDM.train$response,
+     bankDM.train$treatment), 2), 4)

        0     1
  0 85.11 61.60
  1 14.89 38.40
> 100 * round(prop.table(table(bankDM.test$response,
+     bankDM.test$treatment), 2), 4)

        0     1
```

[1] Alternatively, stratified random sampling could have been used.

```
  0 85.30 61.62
  1 14.70 38.38
```

Although the promotion was randomized with respect to clients, it is always desirable to test any departures from randomization. The results of `checkBalance` below indicate that we cannot reject the null hypothesis of overall balance of covariates between treatment and control groups ($P$ value $= 0.961$). Also, by inspecting the mean of the covariates in each group, we see that all of them are fairly balanced. This is shown for the first few covariates.

```
> treat.form <- treatment ~ age + gender +
+     withdrawals + deposit + credit_value +
+     discounts + transactions + bank_logs +
+     accruals + charges + cash_total + loan_payment +
+     e_trans
> cb <- checkBalance(treat.form, data = bankDM.train)
> round(cb$results[, c(1:3, 6:7), ], 2)
              stat
vars           treatment=0 treatment=1 adj.diff     z    p
  age                35.39       35.38    -0.01 -0.02 0.98
  gender0             0.47        0.48     0.01  0.59 0.55
  gender1             0.53        0.52    -0.01 -0.59 0.55
  withdrawals       100.86      100.27    -0.60 -1.01 0.31
......
> cb$overall
        chisquare df p.value
unstrat      5.56 13  0.9607
```

We now use the function `niv` to identify potentially useful PTE predictors. We compute the *net information value* and the *adjusted net information value* over $B = 100$ bootstrap samples of the training data. The results indicate that the first five potentially most influential predictors are: `gender`, `age`, `transactions`, `e_trans`, and `charges`.

```
> set.seed(1)
> Model.form <- response ~ trt(treatment) +
+     age + gender + withdrawals + deposit +
+     credit_value + discounts + transactions +
+     bank_logs + accruals + charges + cash_total +
+     loan_payment + e_trans
> niv_res <- niv(Model.form, B = 100, nbins = 4,
+     plotit = FALSE, data = bankDM.train)
> niv_res$niv_val[order(niv_res$niv_val[, 3],
+     decreasing = TRUE), ]
                niv penalty adj_niv
gender       4.9120  0.2027  4.7093
age          1.7385  0.0824  1.6561
transactions 1.5862  0.0993  1.4869
e_trans      1.3714  0.0878  1.2836
charges      1.2391  0.0785  1.1606
......
```

We now use the function `explore` for a univariate analysis of uplift. For illustration purposes, we show univariate results for `gender` and `age`. We can clearly see that the promotion persuaded males (`gender` = 1) to buy the product to a much larger extent than to females. This is also the case for relatively older clients.

```
> eda <- explore(Model.form, data = bankDM.train)
> eda$gender
  N(Treat=0) N(Treat=1) Ybar(Treat=0) Ybar(Treat=1) Uplift
0       1032       1052        0.1492        0.3099 0.1607
1       1157       1138        0.1487        0.4525 0.3039
> eda$age
        N(Treat=0) N(Treat=1) Ybar(Treat=0) Ybar(Treat=1)
[20,27]        612        636        0.1438        0.3412
(27,34]        535        512        0.1533        0.3691
(34,43]        537        539        0.1583        0.3673
(43,61]        505        503        0.1406        0.4712
        Uplift
```

```
[20,27] 0.1974
(27,34] 0.2159
(34,43] 0.2091
(43,61] 0.3306
```

Next, we fit three alternative PTL models to the training data: *causal conditional inference forest* (`ccif`), *uplift random forest* (`upliftRF`), and the *modified outcome method* (`mom`). We initially fit the first two models based on preliminary values for their hyperparameters. For the `ccif`, we also compute the relative importance of the PTE predictors using the function `varImportance` (Figure 11.1). For the `mom`, we perform stepwise model selection guided by the *Akaike information criterion* (*AIC*: see Venables and Ripley, 2002).

```
> ### Causal conditional inference
> ### forests (ccif)
> set.seed(1)
> ccif_fit1 <- ccif(Model.form, data = bankDM.train,
+     ntree = 1000, split_method = "Int",
+     distribution = approximate(B = 999),
+     verbose = TRUE)
> op <- par(mar = c(5, 10, 4, 2) + 0.1)
> varImportance(ccif_fit1, plotit = TRUE)
```

```
> ### Uplift random forests (upliftRF)
> set.seed(1)
> upliftRF_fit1 <- upliftRF(Model.form, data = bankDM.train,
+     ntree = 1000, interaction.depth = 3,
+     split_method = "KL", minsplit = 50, verbose = TRUE)
```

```
> ### Modified outcome method (mom)
> set.seed(1)
> bankDM.train.mom <- rvtu(Model.form, data = bankDM.train,
+     method = "undersample")
```

```
> Model.form.mom <- z ~ age + gender + withdrawals +
+     deposit + credit_value + discounts +
+     transactions + bank_logs + accruals +
+     charges + cash_total + loan_payment +
+     e_trans
> glm.fit1 <- glm(Model.form.mom, data = bankDM.train.mom,
+     family = "binomial")
> ### Perform stepwise model selection by AIC
> glm.fit_step <- stepAIC(glm.fit1, direction = "backward",
+     trace = 0)
```

Since the company's budget limits the number of targets to no more than 30% of their client base, we assess the performance of all fitted models by the third decile uplift measured on the test data. We also compare models based on the Qini coefficient. The results of the code below show that the `ccif` performs best on the third decile uplift criterion and the `upliftRF` performs best on the Qini criterion. The `mom` performs worst on both criteria.

```
> ### Get predictions on test data
> pred_upliftRF <- predict(upliftRF_fit1,
+     bankDM.test)
> pred_ccif <- predict(ccif_fit1,
+     bankDM.test)
> pred_mom <- 2 * predict(glm.fit_step,
+     bankDM.test) - 1
> ### Get uplift by decile
> ccif_perf <- performance(pred_ccif[,
+     1], pred_ccif[, 2], bankDM.test$response,
+     bankDM.test$treatment)
> upliftRF_perf <- performance(pred_upliftRF[,
+     1], pred_upliftRF[, 2], bankDM.test$response,
+     bankDM.test$treatment)
> mom_perf <- performance(pred_mom,
+     rep(0, length(pred_mom)), bankDM.test$response,
+     bankDM.test$treatment)
```

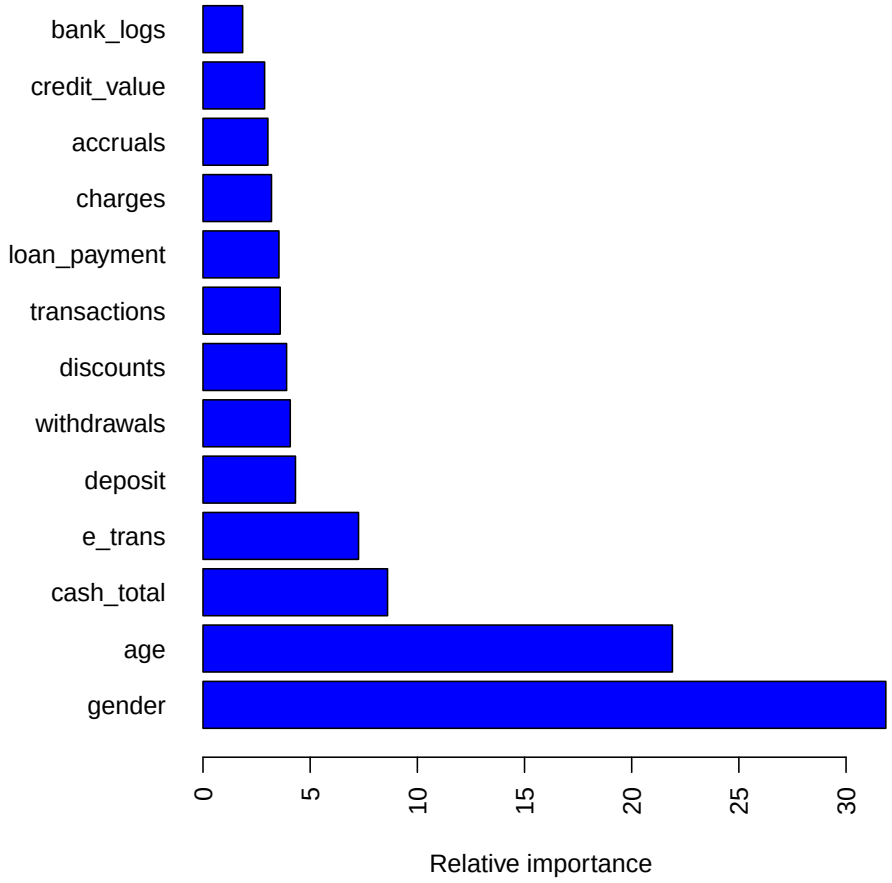


Figure 11.1: Relative importance of PTE predictors based on `ccif`.

```
> ### 3rd decile uplift
> Decile_3_perf <- data.frame(ccif = (sum(ccif_perf[1:3,
+     4])/sum(ccif_perf[1:3, 2])) -
+     (sum(ccif_perf[1:3, 5])/sum(ccif_perf[1:3,
+         3])), upliftRF = (sum(upliftRF_perf[1:3,
+     4])/sum(upliftRF_perf[1:3,
+     2])) - (sum(upliftRF_perf[1:3,
+     5])/sum(upliftRF_perf[1:3,
+     3])), mom = (sum(mom_perf[1:3,
+     4])/sum(mom_perf[1:3, 2])) -
+     (sum(mom_perf[1:3, 5])/sum(mom_perf[1:3,
+         3])))
> Decile_3_perf
    ccif upliftRF    mom
1 0.3674   0.3591 0.3571
```

```
> ### qini-coefficient
> qini <- data.frame(ccif = qini(ccif_perf,
+     plotit = FALSE)$Qini, upliftRF = qini(upliftRF_perf,
+     plotit = FALSE)$Qini, mom = qini(mom_perf,
+     plotit = FALSE)$Qini)
```

```
> qini
    ccif upliftRF    mom
1 0.2948    0.314 0.2639
```

The `ccif` method above was fitted based on preliminary values for its parameters: the number of trees `ntree` and the number of candidate predictors `mtry` to test at each node. We now attempt to further tune these parameters. The code below is used to obtain third decile out-of-bag uplift estimates (see Section 7.5.3) from a grid of suitable values of the tuning parameters.

```
> ### Criteria: out-of-bag 3rd decile
> ### uplift Tune parameters: mtry and
> ### ntree
> mtry.seq <- seq(3, 13, 2)
> ntree.seq <- seq(500, 1000, 100)
> ccif_fits <- vector("list", length(mtry.seq))
> oob_3d_pte <- matrix(nrow = length(ntree.seq),
+     ncol = length(mtry.seq))
> set.seed(1)
> for (i in mtry.seq) {
+     j <- which(i == mtry.seq)
+     ccif_fits[[j]] <- ccif(Model.form,
+         data = bankDM.train, mtry = i,
+         ntree = 1000, split_method = "Int",
+         distribution = approximate(B = 999),
+         keep.inbag = TRUE, verbose = TRUE)
+ }
> ### Obtain out-of-bag predictions
> for (i in ntree.seq) {
+     l <- which(i == ntree.seq)
+     for (j in 1:length(ccif_fits)) {
+         pred <- (predict(ccif_fits[[j]],
+             bankDM.train, n.trees = i,
+             predict.all = TRUE))$individual
+         ### Extract treatment/control
+         ### predictions
+         pred_ct1 <- sapply(1:length(pred),
+             function(k) pred[[k]][,
+                 1])
+         pred_ct0 <- sapply(1:length(pred),
+             function(k) pred[[k]][,
+                 2])
+         ### Insert NAs in observations used
+         ### for model fitting (the inbags)
+         for (h in 1:ncol(pred_ct1)) {
```

```
+             pred_ct1[unique(ccif_fits[[j]]$inbag[,
+                 h]), h] <- NA
+         }
+         for (h in 1:ncol(pred_ct0)) {
+             pred_ct0[unique(ccif_fits[[j]]$inbag[,
+                 h]), h] <- NA
+         }
+         ### Compute oob predictions by
+         ### removing 'inbag' samples
+         pred_ct1.oob <- apply(pred_ct1[,
+             1:i], 1, mean, na.rm = TRUE)
+         pred_ct0.oob <- apply(pred_ct0[,
+             1:i], 1, mean, na.rm = TRUE)
+         ### Get performance
+         perf <- performance(pred_ct1.oob,
+             pred_ct0.oob, bankDM.train$response,
+             bankDM.train$treatment)
+         ### Store 3rd decile oob pte
+         oob_3d_pte[l, j] <- (sum(perf[1:3,
+             4])/sum(perf[1:3, 2])) -
+             (sum(perf[1:3, 5])/sum(perf[1:3,
+                 3]))
+     }
+ }
```

```
> colnames(oob_3d_pte) <- mtry.seq
> rownames(oob_3d_pte) <- ntree.seq
> oob_3d_pte == max(oob_3d_pte)
        3     5     7     9    11    13
500  FALSE FALSE FALSE FALSE FALSE FALSE
600  FALSE FALSE FALSE FALSE  TRUE FALSE
700  FALSE FALSE FALSE FALSE FALSE FALSE
800  FALSE FALSE FALSE FALSE FALSE FALSE
900  FALSE FALSE FALSE FALSE FALSE FALSE
......
```

The values of `ntree` and `mtry` that maximize the out-of-bag third decile uplift estimates are 600 and 11, respectively. The results from the code below show that based on these tuned parameters, the third decile uplift has slightly improved on the test data. The uplift on this group is 37%, relative to 23% for all clients.

```
> ccif_tuned <- ccif_fits[[5]]  ### corresponds to mtry = 11
> pred_ccif_tuned <- predict(ccif_tuned,
+     bankDM.test, n.trees = 600)
> perf_ccif_tuned <- performance(pred_ccif_tuned[,
+     1], pred_ccif_tuned[, 2], bankDM.test$response,
+     bankDM.test$treatment)
> (sum(perf_ccif_tuned[1:3, 4])/sum(perf_ccif_tuned[1:3,
+     2])) - (sum(perf_ccif_tuned[1:3,
+     5])/sum(perf_ccif_tuned[1:3, 3]))
[1] 0.3706
```

Finally, we use `modelProfile` to profile the clients predicted to have highest/lowest PTEs from the marketing intervention activity.

```
> bankDM.test$pred <- pred_ccif_tuned[, 1] -
+     pred_ccif_tuned[, 2]
> modelProfile(pred ~ age + gender + cash_total +
+     e_trans, data = bankDM.test, groups = 10,
+     group_label = "D", digits_numeric = 1,
+     digits_factor = 2)
```

Table 11.2: Distribution of specific predictors using `modelProfile`

| | | Group | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | ... | 8 | 9 | 10 | All |
| | n | 188 | 188 | 187 | ... | 187 | 188 | 188 | 1877 |
| `pred` | Avg. | 0.4 | 0.3 | 0.3 | ... | 0.1 | 0.1 | 0.1 | 0.2 |
| age | Avg. | 46.7 | 39.5 | 35.7 | ... | 31.4 | 29.4 | 27.9 | 35.4 |
| `cash_total` | Avg. | 80.5 | 85.6 | 96.0 | ... | 94.7 | 103.2 | 102.2 | 100.0 |
| `e_trans` | Avg. | 180.5 | 196.6 | 198.3 | ... | 183.8 | 201.1 | 232.3 | 200.1 |
| `gender` = F | Pctn. | 0.00 | 1.06 | 10.16 | ... | 98.93 | 99.47 | 99.47 | 48.27 |
| `gender` = M | Pctn. | 100.00 | 98.94 | 89.84 | ... | 1.07 | 0.53 | 0.53 | 51.73 |

Table 11.2 illustrates the client profile based on the top 4 most influential predictors obtained above. We see that clients in the first 3 deciles are relatively older in `age`, have lower `cash_total` and lower `e_trans`, and are mostly males.

# 12 Conclusions and future challenges

In this thesis, we have introduced the concept of *personalized treatment learning* as the problem of learning the optimal "treatment" or action tailored to each individual for the purpose of maximizing the probability of a desirable outcome. We have formalized the personalized treatment learning problem from a *causal inference* perspective and provided a comprehensive description of the existing methods to solve this problem. We have contributed to the personalized treatment modeling literature by proposing two novel methods, namely *uplift random forests* (Guelman et al., 2012, 2014c) and *causal conditional inference forests* (Guelman et al., 2014b). Our proposal compares favorably with the existing methods based on an extensive numerical simulation and real-world data.

Personalized treatment learning models have applications to a wide variety of fields, ranging from economics to medicine, but their application to insurance has remained absent. We have contributed to the insurance literature by highlighting the value of personalized treatment learning models for insurance marketing and pricing applications. In particular, we have implemented the existing and our proposed methods to optimize client retention and cross-selling in insurance from experimental data. We have also illustrated an application of personalized treatment learning models to price-elasticity estimation and insurance economic price optimization in the context of observational data (Guelman and Guillén, 2014; Guelman et al., 2014a).

A key problem facing research in this field has been the lack of publicly

available statistical software to estimate personalized treatment learning models. We have implemented most of the existing statistical methods along with our proposed algorithms for fitting personalized treatment learning models in a package named **uplift** (Guelman, 2014), which is now released and freely available from the CRAN (Comprehensive R Archive Network) repository under the R statistical computing environment.

In the context of insurance, the selection of the optimal personalized treatment also requires consideration of the expected insurance losses of each individual policyholder within the portfolio. We have contributed to the insurance ratemaking literature by proposing a novel application of gradient boosting models to estimate insurance loss cost, with key important advantages over the conventional generalized linear model approach (Guelman, 2012).

We would also like to acknowledge the limitations of this thesis and highlight some future challenges in the area of personalized treatment learning models. First, we have only considered the case of binary treatments. It would be worthwhile to examine the extent to which the methods discussed in this thesis can be extended to multi-category or continuous treatment settings. Second, our work is limited to the case of a binary response variable. Clearly, extensions to continuous uncensored and survival responses would be useful in many applications. Third, we have only considered the case of personalized treatments in a single-decision setup. In dynamic treatment regimes, the treatment type is repeatedly adjusted according to an ongoing individual response (Murphy, 2005). In this context, the goal is to optimize a set of time-varying personalized treatments for the purpose of maximizing the probability of a long-term desirable outcome. Lastly, in this thesis, we have defined the personalized treatment effect in terms of the difference between the expected responses under alternative treatment conditions. However, in some settings, having a relative measure of the personalized treatment effect may be of further interest. In this case, the treatment effect is defined in terms of the ratio of the expected responses under the alternative treatments.

# Bibliography


Abu-Mostafa, Y. S., Magdon-Ismail, M., and Lin, H.-T. (2012). *Learning From Data*. AMLBook, Pasadena, CA.

Alemi, F., Erdman, H., Griva, I., and Evans, C. H. (2009). Improved statistical methods are needed to advance personalized medicine. *The Open Translational Medicine Journal*, 1:16–20.

Anderson, D., Feldblum, S., Modlin, C., Schirmacher, D., Schirmacher, E., and Thandi, N. (2007). A practitioner's guide to generalized linear models. Casualty Actuarial Society Study Note.

Anderson, R. (2007). *The Credit Scoring Toolkit: Theory and Practice for Retail Credit Risk Management and Decision Automation*. Oxford University Press, New York, NY.

Antonio, K. and Valdez, E. (2012). Statistical concepts of a priori and a posteriori risk classification in insurance. *AStA Advances in Statistical Analysis*, 96(2):187–224.

Bellman, R. E. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, 57(1):289–300.

Berk, R. (2004). *Regression Analysis: A Constructive Critique*. Sage, Thousand Oaks, CA.

Bertsekas, D. (1998). *Network Optimization: Continuous and Discrete Models*. Athena Scientific, Belmont, MA.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY.

Blattberg, R. C., Do, K. B., and Scott, N. A. (2008). *Database Marketing: Analyzing and Managing Customers*. Springer Science+Business Media, New York, NY.

Bowers, J., Fredrickson, M., and Hansen, B. (2010). *RItools: Randomization inference tools*. R package version 0.1-11.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Breiman, L. (2001a). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L. (2001b). Statistical modeling: The two cultures. *Statistical Science*, 16(3):199–215.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Chapman & Hall, New York, NY.

Brockman, M. J. and Wright, T. S. (1992). Statistical motor rating: Making effective use of your data. *Journal of the Institute of Actuaries*, 119:457–543.

Brown, R. and Gottlieb, L. (2007). *Introduction to Ratemaking and Loss Reserving for Property and Casualty Insurance*. Actex, Winsted, CT.

Byers, R. E. and So, K. C. (2007). Note—a mathematical model for evaluating cross-sales policies in telephone service centers. *Manufacturing & Service Operations Management*, 9(1):1–8.

Cai, T., Tian, L., Wong, P. H., and Wei, L. J. (2011). Analysis of randomized comparative clinical trial data for personalized treatment selections. *Biostatistics*, 12(2):270–282.

Chan, P. K. and Stolfo, S. J. (1998). Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In Agrawal, R., Stolorz, P. E., and Piatetsky-Shapiro, G., editors, *Proceedings of the Fourth International Conference on*

*Knowledge Discovery and Data Mining*, pages 164–168. AAAI Press, Palo Alto, CA.

Chapados, N., Bengio, Y., Vincent, P., Ghosn, J., Dugas, C., Takeuchi, I., and Meng, L. (2001). Estimating car insurance premia: a case study in high-dimensional data inference. In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems*, volume 14, pages 1369–1376. MIT Press, Cambridge, MA.

Chawla, N. V. (2005). Data mining for imbalanced datasets: An overview. In Maimon, O. and Rokach, L., editors, *The Data Mining and Knowledge Discovery Handbook*, pages 853–867. Springer Science+Business Media, Secaucus, NJ.

Cochran, W. G. and Rubin, D. B. (1973). Controlling bias in observational studies: A review. *Sankhyā, Series A, Indian Journal of Statistics*, 35:417–446.

Coussement, K. and den Poel, D. V. (2008). Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. *Expert Systems with Applications*, 34(1):313–327.

Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. Wiley, New York, NY.

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27.

Davis, J. C., Furstenthal, L., Desai, A. A., Norris, T., Sutaria, S., Fleming, E., and Ma, P. (2009). The microeconomics of personalized medicine: Today's challenge and tomorrow's promise. *Nature Reviews Drug Discovery*, 8(4):279–286.

Dawid, A. P. (1979). Conditional independence in statistical theory. *Journal of the Royal Statistical Society: Series B*, 41:1–31.

Dehejia, R. and Wahba, S. (1999). Causal effects in nonexperimental studies: Reevaluating the evaluation of training programs. *Journal of the American Statistical Association*, 94(443):1053–1062.

Denuit, M., Maréchal, X., Pitrebois, S., and Walhin, J. (2007). *Actuarial Modelling of Claim Counts*. Wiley, Hoboken, NJ.

Derigs, U. (1988). Solving non-bipartite matching problems via shortest path techniques. *Annals of Operations Research*, 13(1):225–261.

Donkers, B., Verhoef, P., and de Jong, M. (2007). Modeling CLV: A test of competing models in the insurance industry. *Quantitative Marketing and Economics*, 5(2):163–190.

Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification*. Wiley, New York, NY, 2nd edition.

Efron, B. (1983). Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78(382):316–331.

Efron, B. and Tibshirani, R. (1986). Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*, 1(1):54–75.

Efron, B. and Tibshirani, R. (1997). Improvements on cross-validation: The .632+ bootstrap method. *Journal of the American Statistical Association*, 92(438):548–560.

Englund, M., Guillén, M., Gustafsson, J., Nielsen, L., and Nielsen, J. (2008). Multivariate latent risk: A credibility approach. *Astin Bulletin*, 38(1):137–146.

Englund, M., Gustafsson, J., Nielsen, J. P., and Thuring, F. (2009). Multidimensional credibility with time effects: An application to commercial business lines. *Journal of Risk & Insurance*, 76(2):443–453.

Estabrooks, A., Jo, T., and Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20(1):18–36.

Fahner, G. (2012). Estimating causal effects of credit decisions. *International Journal of Forecasting*, 28:248–260.

Finger, R. J. (2006). Risk classification, practical aspects. In Teugels, J. and Sundt, B., editors, *Encyclopedia of Actuarial Science*. John Wiley & Sons, Ltd, Chichester, UK.

Francis, L. (2001). Neural networks demystified. Casualty Actuarial Society Forum.

Frawley, W. J., Piatetsky-Shapiro, G., and Matheus, C. J. (1992). Knowledge discovery in databases: An overview. *AI Magazine*, 13(3):57–70.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In Saitta, L., editor, *Proceedings of the Thirteenth International Conference on Machine Learning (ICML)*, pages 148–156. Morgan Kaufmann, San Francisco, CA.

Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–407.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.

Ginsburg, G. S. and McCarthy, J. J. (2001). Personalized medicine: revolutionizing drug discovery and patient care. *Trends in Biotechnology*, 19(12):491–496.

Greene, W. H. (2003). *Econometric Analysis*. Prentice Hall, Upper Saddle River, NJ, 5th edition.

Greevy, R., Lu, B., Silber, J. H., and Rosenbaum, P. (2004). Optimal multivariate matching before randomization. *Biostatistics*, 5(2):263–275.

Gu, X. and Rosenbaum, P. R. (1993). Comparison of multivariate matching methods: Structures, distances, and algorithms. *Journal of Computational and Graphical Statistics*, 2(4):405–420.

Guelman, L. (2012). Gradient boosting trees for auto insurance loss cost modeling and prediction. *Expert Systems with Applications*, 39(3):3659–3667.

Guelman, L. (2014). *uplift: Uplift modeling*. R package version 0.3.5.

Guelman, L. and Guillén, M. (2014). A causal inference approach to measure price elasticity in automobile insurance. *Expert Systems with Applications*, 41(2):387–396.

Guelman, L., Guillén, M., and Pérez-Marín, A. M. (2012). Random forests for uplift modeling: An insurance customer retention case. In Engemann, K. J., Lafuente, A. M. G., and Merigó, J. M., editors, *Modeling and Simulation in Engineering, Economics and Management*, pages 123–133. Springer Berlin Heidelberg, New York, NY.

Guelman, L., Guillén, M., and Pérez-Marín, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Guelman, L., Guillén, M., and Pérez-Marín, A. M. (2014b). A survey of personalized treatment models for pricing strategies in insurance. *Insurance: Mathematics and Economics*, 58:68–76.

Guelman, L., Guillén, M., and Pérez-Marín, A. M. (2014c). Uplift random forests. *Cybernetics & Systems*. Accepted.

Guillén, M., Nielsen, J. P., Scheike, T. H., and Pérez-Marín, A. M. (2012). Time-varying effects in the analysis of customer loyalty: A case study in insurance. *Expert Systems with Applications*, 39(3):3551–3558.

Guillén, M., Pérez-Marín, A. M., and Alcaniz, M. (2011). A logistic regression approach to estimating customer profit loss due to lapses in insurance. *Insurance Markets and Companies: Analyses and Actuarial Computations*, 2(2):43–55.

Günes, E. D., Aksin, O. Z., Örmeci, E. L., and Özden, S. H. (2010). Modeling customer reactions to sales attempts: If cross-selling backfires. *Journal of Service Research*, 13(2):168–183.

Guo, S. Y. and Fraser, M. W. (2010). *Propensity Score Analysis: Statistical Methods and Applications.* Sage, Thousand Oaks, CA.

Haberman, S. and Renshaw, A. E. (1996). Generalized linear models and actuarial science. *Journal of the Royal Statistical Society: Series D*, 45(4):407–436.

Hand, D. J. and Till, R. J. (2001). A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine Learning*, 45(2):171–186.

Hansen, B. B. and Klopfer, S. O. (2006). Optimal full matching and related designs via network flows. *Journal of Computational and Graphical Statistics*, 15(3):609–627.

Hansotia, B. and Rukstales, B. (2002). Incremental value modeling. *Journal of Interactive Marketing*, 16(3):35–46.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning.* Springer Science+Business Media, New York, NY.

Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized Additive Models.* Chapman & Hall, London, UK.

Hirano, K. and Imbens, G. W. (2005). The propensity score with continuous treatments. In Gelman, A. and Meng, X.-L., editors, *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives*, pages 73–84. John Wiley & Sons, Ltd, Chichester, UK.

Hofner, B., Müller, J., and Hothorn, T. (2011). Monotonicity-constrained species distribution models. *Ecology*, 92(10):1895–1901.

Holland, P. W. (1986). Statistics and causal inference. *Journal of the American Statistical Association*, 81(396):945–960.

Holland, P. W. and Rubin, D. B. (1989). Causal inference in retrospective studies. *Evaluation Review*, 12(3):203–231.

Hothorn, T., Hornik, K., de Wiel, M. A., and Zeileis, A. (2008). Implementing a class of permutation tests: The coin package. *Journal of Statistical Software*, 28(8):1–23.

Hothorn, T., Hornik, K., and Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3):651–674.

Huber, P. J. (1964). Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35(1):73–101.

Imai, K. and Ratkovic, M. (2013). Estimating treatment effect heterogeneity in randomized program evaluation. *Annals of Applied Statistics*, 7(1):443–470.

Imai, K. and van Dyk, D. A. (2004). Causal inference with general treatment regimes: Generalizing the propensity score. *Journal of the American Statistical Association*, 99(467):854–866.

Imbens, G. (2000). The role of the propensity score in estimating dose–response functions. *Biometrika*, 87:706–710.

Jaśkowski, M. and Jaroszewicz, S. (2012). Uplift modeling for clinical trial data. In *ICML 2012 Workshop on Clinical Data Analysis*, Edinburgh, Scotland.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society: Series A*, 186(1007):453–461.

Jha, S., Guillén, M., and Westland, J. C. (2012). Employing transaction aggregation strategy to detect credit card fraud. *Expert Systems with Applications*, 39(16):12650–12657.

Joffe, M. and Rosenbaum, P. R. (1999). Propensity scores. *American Journal of Epidemiology*, 150(4):327–333.

Johnson, R. A. and Wichern, D. W. (2001). *Applied Multivariate Statistical Analysis*. Prentice Hall, Upper Saddle River, NJ, 5th edition.

Kaishev, V. K., Nielsen, J. P., and Thuring, F. (2013). Optimal customer selection for cross-selling of financial services products. *Expert Systems with Applications*, 40(5):1748–1757.

Kamakura, W. A. (2008). Cross-selling. *Journal of Relationship Marketing*, 6(3-4):41–58.

Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Journal of the Royal Statistical Society: Series C*, 29(2):119–127.

King, G. and Zeng, L. (2001). Explaining rare events in international relations. *International Organization*, 55:693–715.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In Mellish, C. S., editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1137–1143. Morgan Kaufmann, San Francisco, CA.

Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2):273–324.

Kolyshkina, I., Wong, S., and Lim, S. (2004). Enhancing generalised linear models with data. Casualty Actuarial Society, Discussion Paper Program.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22:49–86.

LaLonde, R. J. (1986). Evaluating the econometric evaluations of training programs with experimental data. *American Economic Review*, 76:604–620.

Larsen, K. (2009). Net lift models. Slides of a talk given at the M2009 – 12$^{\text{th}}$ Annual SAS Data Mining Conference, October 26–27, Las Vegas, NV.

Lee, L. (1999). Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 25–32. Association for Computational Linguistics, Stroudsburg, PA.

Lemmens, A. and Croux, C. (2006). Bagging and boosting classification trees to predict churn. *Journal of Marketing Research*, 43(2):276–286.

Liang, H., Xue, Y., and Berger, B. A. (2006). Web-based intervention support system for health promotion. *Decision Support Systems*, 42(1):435–449.

Lo, V. S. Y. (2002). The true lift model – a novel data mining approach to response modeling in database marketing. *SIGKDD Explorations*, 4(2):78–86.

Loi, S., Haibe-Kains, B., Desmedt, C., Lallemand, F. A., Tutt, A. M., Gillet, C., Ellis, P., Harris, A., Bergh, J., Foekens, J. A., Klijn, J. G. M., Larsimont, D., Buyse, M., Bontempi, G., Delorenzi, M., Piccart, M. J., and Sotiriou, C. (2007). Definition of clinically distinct molecular subtypes in estrogen receptor-positive breast carcinomas through genomic grade. *Journal of Clinical Oncology*, 25(10):1239–1246.

Lu, B., Zanutto, E., Hornik, R., and Rosenbaum, P. R. (2001). Matching with doses in an observational study of a media campaign against drug abuse. *Journal of the American Statistical Association*, 96(456):1245–1253.

McCaffrey, D. F., Ridgeway, G., and Morral, A. R. (2004). Propensity score estimation with boosted regression for evaluating causal effects in observational studies. *Psychological Methods*, 9(4):403–425.

McCullagh, P. and Nelder, J. A. (1989). *Generalized Linear Models*. Chapman & Hall, London, UK, 2nd edition.

Mease, D. and Wyner, A. (2008). Evidence contrary to the statistical view of boosting. *Journal of Machine Learning Research*, 9:131–156.

Mingers, J. (1989). An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, 3(4):319–342.

Morgan, S. L. and Winship, C. (2007). *Counterfactuals and Causal Inference: Methods and Principles for Social Research*. Cambridge University Press, New York, NY.

Morik, K. and Köpcke, H. (2004). Analysing customer churn in insurance data – a case study. In Boulicaut, J.-F., Esposito, F., Giannotti, F.,

and Pedreschi, D., editors, *Knowledge Discovery in Databases: PKDD 2004*, pages 325–336. Springer-Verlag, Berlin, Germany.

Murdoch, D. (2013). *tables: Formula-driven table generation*. R package version 0.7.64.

Murphy, S. A. (2005). An experimental design for the development of adaptive treatment strategies. *Statistics in medicine*, 24(10):1455–1481.

Neslin, S. A., Gupta, S., Kamakura, W., Lu, J., and Mason, C. H. (2006). Defection detection: Measuring and understanding the predictive accuracy of customer churn models. *Journal of Marketing Research*, 43(2):204–211.

Ng, P. C., Murray, S. S., Levy, S., and Venter, J. C. (2009). An agenda for personalized medicine. *Nature*, 461(7265):724–726.

Pinquet, J., Guillén, M., and Ayuso, M. (2011). Commitment and lapse behavior in long-term insurance: A case study. *Journal of Risk and Insurance*, 78(4):983–1002.

Qian, M. and Murphy, S. A. (2011). Performance guarantees for individualized treatment rules. *Annals of Statistics*, 39(2):1180–1210.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Radcliffe, N. (2007). Using control groups to target on predicted lift: Building and assessing uplift model. *Direct Marketing Analytics Journal*, 2007:14–21.

Radcliffe, N. J. and Surry, P. D. (2011). Real-world uplift modelling with significance-based uplift trees. Technical Report TR-2011-1, Portrait.

Ridgeway, G. (2007). *Generalized boosted models: A guide to the gbm package*.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, New York, NY.

Rosenbaum, P. R. (1989). Optimal matching for observational studies. *Journal of the American Statistical Association*, 84(408):1024–1032.

Rosenbaum, P. R. (1991). A characterization of optimal designs for observational studies. *Journal of the Royal Statistical Society, Series B*, 53(3):597–610.

Rosenbaum, P. R. (2002). Attributing effects to treatment in matched observational studies. *Journal of the American Statistical Association*, 97(457):183–192.

Rosenbaum, P. R. (2010). *Design of Observational Studies*. Springer Science+Business Media, New York, NY.

Rosenbaum, P. R. and Rubin, D. B. (1983). The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55.

Rosenbaum, P. R. and Rubin, D. B. (1984). Reducing bias in observational studies using subclassification on the propensity score. *Journal of the American Statistical Association*, 79(387):516–524.

Rosenbaum, P. R. and Rubin, D. B. (1985). Constructing a control group using multivariate matched sampling methods that incorporate the propensity score. *The American Statistician*, 39(1):33–38.

Rubin, D. B. (1973). The use of matched sampling and regression adjustment to remove bias in observational studies. *Biometrics*, 29(1):185–203.

Rubin, D. B. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66:688–701.

Rubin, D. B. (1976). Multivariate matching methods that are equal percent bias reducing, I: Some examples. *Biometrics*, 32(1):109–120.

Rubin, D. B. (1977). Assignment to treatment group on the basis of a covariate. *Journal of Educational and Behavioral Statistics*, 2(1):1–26.

Rubin, D. B. (1978). Bayesian inference for causal effects: The role of randomization. *The Annals of Statistics*, 6(1):34–58.

Rubin, D. B. (1979). Using multivariate matched sampling and regression adjustment to control bias in observational studies. *Journal of the American Statistical Association*, 74:318–328.

Rubin, D. B. (1997). Estimating causal effects from large data sets using propensity scores. *Annals of Internal Medicine*, 127(8):757–763.

Rubin, D. B. (2005). Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100:322–331.

Rubin, D. B. and Waterman, R. P. (2006). Estimating the causal effects of marketing interventions using propensity score methodology. *Statistical Science*, 21(2):206–222.

Rzepakowski, P. and Jaroszewicz, S. (2012). Decision trees for uplift modeling with single and multiple treatments. *Knowledge and Information Systems*, 32(2):303–327.

Santoni, A. and Gómez Alvado, F. (2008). Sophisticated price optimization methods. Slides of a talk given at the Casualty Actuarial Society Ratemaking Seminar 2008, March 17–18, Cambridge, MA.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2):197–227.

Segal, M. R. (2004). Machine learning benchmarks and random forest regression. Technical report, University of California, San Francisco.

Shaffer, J. P. (1995). Multiple hypothesis testing. *Annual Review of Psychology*, 46(1):561–584.

Siddiqi, N. (2006). *Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring*. Wiley, Hoboken, NJ.

Sinha, A. P. and Zhao, H. (2008). Incorporating domain knowledge into data mining classifiers: An application in indirect lending. *Decision Support Systems*, 46(1):287–299.

Smith, K., Willis, R., and Brooks, M. (2000). An analysis of customer retention and insurance claim patterns using data mining: A case study. *The Journal of the Operational Research Society*, 51(5):532–541.

Snedecor, G. W. and Cochran, W. G. (1980). *Statistical Methods*. Iowa State University Press, Ames, IA, 7th edition.

Stauss, B., Schmidt, M., and Schoeler, A. (2005). Customer frustration in loyalty programs. *International journal of service industry management*, 16(3):229–252.

Strasser, H. and Weber, C. (1999). On the asymptotic theory of permutation statistics. *Mathematical Methods of Statistics*, 8:220–250.

Su, X., Kang, J., Fan, J., Levine, R. A., and Yan, X. (2012). Facilitating score and causal inference trees for large observational studies. *Journal of Machine Learning Research*, 13(10):2955–2994.

Su, X., Tsai, C.-L., Wang, H., Nickerson, D. M., and Li, B. (2009). Subgroup analysis via recursive partitioning. *Journal of Machine Learning Research*, 10:141–158.

Sun, Y., Kamel, M. S., Wong, A. K. C., and Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378.

Tang, H., Liao, S. S., and Sun, S. X. (2013). A prediction framework based on contextual data to support mobile personalized marketing. *Decision Support Systems*, 56(0):234–246.

Thuring, F., Nielsen, J. P., Guillén, M., and Bolancé, C. (2012). Selecting prospects for cross-selling financial products using multivariate credibility. *Expert Systems with Applications*, 39(10):8809–8816.

Tian, L., Alizadeh, A., Gentles, A., and Tibshirani, R. (2014). A simple method for detecting interactions between a treatment and a large number of covariates. Submitted.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58:267–288.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer, New York, NY.

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer Science+Business Media, New York, NY, 4th edition.

Wahba, G. (2002). Soft and hard classification by reproducing kernel hilbert space methods. *Proceedings of the National Academy of Sciences*, 99(26):16524–16530.

Weiss, G. M. and Provost, F. J. (2003). Learning when training data are costly: The effect of class distribution on tree induction. *J. Artif. Intell. Res. (JAIR)*, 19:315–354.

Wright, S. P. (1992). Adjusted p-values for simultaneous inference. *Biometrics*, 48(4):1005–1013.

Xu, D. J., Liao, S. S., and Li, Q. (2008). Combining empirical experimentation and modeling techniques: A design research approach for personalized mobile advertising applications. *Decision Support Systems*, 44(3):710–724.

Yeo, A., Smith, K., Willis, R., and Brooks, M. (2001). Modeling the effect of premium changes on motor insurance customer retention rates using neural networks. In Alexandrov, V., Dongarra, J., Juliano, B., Renner, R., and Tan, C., editors, *Computational Science – ICCS 2001*, pages 390–399. Springer-Verlag, Berlin, Germany.

Zhao, Y. and Zeng, D. (2013). Recent development on statistical methods for personalized medicine discovery. *Frontiers of Medicine*, 7(1):102–110.

Zhao, Y., Zeng, D., Rush, A. J., and Kosorok, M. R. (2012). Estimating individualized treatment rules using outcome weighted learning. *Journal of the American Statistical Association*, 107(499):1106–1118.

Zliobaitė, I. and Pechenizkiy, M. (2010). Learning with actionable attributes: Attention – boundary cases! In Fan, W., Hsu, W., Webb, G. I., 0001, B. L., Zhang, C., Gunopulos, D., and Wu, X., editors, *Proceedings of the 2010 IEEE International Conference on Data Mining Workshops*, pages 1021–1028. IEEE Computer Society, Washington, DC.

# Appendices

# Appendix A: uplift package manual

## Package 'uplift'

August 22, 2014

**Version** 0.3.5

**Date** 2014-01-03

**Title** Uplift Modeling

**Description** An integrated package for building and testing uplift models

**Author** Leo Guelman

**Maintainer** Leo Guelman <leo.guelman@gmail.com>

**License** GPL-2 | GPL-3

**Depends** R (>= 3.0.0), RItools, MASS, coin, tables, penalized

**Repository** CRAN

---

`uplift-package` *Uplift modeling*

---

**Description**

An integrated package for building and testing uplift models.

**Details**

| | |
|---|---|
| Package: | uplift |
| Type: | Package |
| Version: | 0.3.4 |
| Date: | 2014-01-03 |
| Depends: | R (>= 3.0.0), RItools, MASS, coin, tables, penalized |
| License: | GPL-2 \| GPL-3 |

**Author(s)**

Leo Guelman <leo.guelman@gmail.com>

---

`ccif` *Causal conditional inference forest*

---

**Description**

`ccif` implements recursive partitioning in a causal conditional inference framework.

**Usage**

```
## S3 method for class formula
ccif(formula, data, ...)

## Default S3 method:
ccif(
x,
y,
ct,
mtry = floor(sqrt(ncol(x))),
ntree = 100,
split_method = c("ED", "Chisq", "KL", "L1", "Int"),
interaction.depth = NULL,
pvalue = 0.05,
bonferroni = FALSE,
minsplit = 20,
minbucket_ct0 = round(minsplit/4),
minbucket_ct1 = round(minsplit/4),
keep.inbag = FALSE,
verbose = FALSE,
...)

## S3 method for class ccif
print(x, ...)
```

**Arguments**

| | |
|---|---|
| `data` | a data frame containing the variables in the model. It should include a variable reflecting the binary treatment assignment of each observation (coded as 0/1). |
| `x, formula` | a data frame of predictors or a formula describing the model to be fitted. A special term of the form `trt()` must be used in the model equation to identify the binary treatment variable. For example, if the treatment is represented by a variable named `treat`, then the right hand side of the formula must include the term `+trt(treat)`. |
| `y` | a binary response (numeric) vector. |

| | |
|---|---|
| ct | a binary (numeric) vector representing the treatment assignment (coded as 0/1). |
| mtry | the number of variables to be tested in each node; the default is floor(sqrt(ncol(x))). |
| ntree | the number of trees to generate in the forest; default is ntree = 100. |
| split_method | the split criteria used at each node of each tree; possible values are: "ED" (Euclidean distance), "Chisq" (Chi-squared divergence), "KL" (Kullback-Leibler divergence), "L1" ($L1$-norm divergence), and "Int" (Interaction method). |
| interaction.depth | the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. |
| pvalue | the maximum acceptable $P$ value required in order to make a split. |
| bonferroni | apply a Bonferroni adjustment to $P$ value. |
| minsplit | the minimum number of observations that must exist in a node in order for a split to be attempted. |
| minbucket_ct0 | the minimum number of control observations in any terminal <leaf> node. |
| minbucket_ct1 | the minimum number of treatment observations in any terminal <leaf> node. |
| keep.inbag | if set to TRUE, an nrow(x) by ntree matrix is returned, whose entries are the "in-bag" samples in each tree. |
| verbose | print status messages? |
| ... | additional arguments passed to independence_test{coin}. See details below. |

## Details

Causal conditional inference trees estimate *personalized treatment effects* (a.k.a. uplift) by binary recursive partitioning in a conditional inference framework. Roughly, the algorithm works as follows: 1) For each terminal node in the tree we test the global null hypothesis of no interaction effect between the treatment $T$ and any of the $n$ covariates selected at random from the set of $p$ covariates ($n \leq p$). Stop if this hypothesis cannot be rejected. Otherwise select the input variable with strongest interaction effect. The interaction effect is measured by a $P$ value corresponding to a permutation test (Strasser and Weber, 1999) for the partial null hypothesis of independence between each input variable and a transformed response. Specifically, the response is transformed so the impact of the input variable on the response has a causal interpretation for the treatment effect (see details in Guelman et al., 2014a). 2) Implement a binary split in the selected input variable. 3) Recursively repeat steps 1) and 2).

The independence test between each input and the transformed response is performed by calling independence_test{coin}. Additional arguments may be passed to this function via '...'.

All split methods are described in Guelman et al. (2014a, 2014c).

This function is very slow at the moment. It was built as a prototype in R. A future version of this package will provide an interface to C++ for this function, which is expected to improve speed significantly.

## Value

An object of class ccif, which is a list with the following components:

| | |
|---|---|
| call | the original call to ccif. |

| | |
|---|---|
| `trees` | the tree structure that was learned. |
| `split_method` | the split criteria used at each node of each tree. |
| `ntree` | the number of trees used. |
| `mtry` | the number of variables tested at each node. |
| `var.names` | a character vector with the name of the predictors. |
| `var.class` | a character vector containing the class of each predictor variable. |
| `inbag` | an `nrow(x)` by `ntree` matrix showing the in-bag samples used by each tree. |

**Author(s)**

Leo Guelman <leo.guelman@gmail.com>

**References**

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014c). Uplift random forests. *Cybernetics & Systems*. Accepted.

Hothorn, T., Hornik, K. and Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3): 651–674.

Strasser, H. and Weber, C. (1999). On the asymptotic theory of permutation statistics. *Mathematical Methods of Statistics*, 8: 220–250.

**Examples**

```
library(uplift)

### Simulate training data

set.seed(12345)
dd <- sim_pte(n = 100, p = 6, rho = 0, sigma =  sqrt(2), beta.den = 4)

dd$treat <- ifelse(dd$treat == 1, 1, 0)

### Fit model

form <- as.formula(paste(y ~, trt(treat) +, paste(X, 1:6, sep = , collapse = "+")))

fit1 <- ccif(formula = form,
             data = dd,
             ntree = 50,
             split_method = "Int",
             distribution = approximate (B=999),
             pvalue = 0.05,
             verbose = TRUE)
print(fit1)
summary(fit1)
```

checkBalance *Standardized differences for stratified comparisons*

## Description

This function is simply a wrapper for xBalance{RItools}. Given covariates, a treatment variable, and (optionally) a stratifying factor, it calculates standardized mean differences along each covariate, and tests for conditional independence of the treatment variable and the covariates.

## Usage

```
checkBalance(formula, data, report = "all", ...)
```

## Arguments

| | |
|---|---|
| formula | a formula containing an indicator of treatment assignment on the left hand side and covariates at right. |
| data | a data frame in which the formula and (optionally) strata are to be evaluated. |
| report | character vector listing measures to report for each stratification; a subset of c("adj.means","adj.mean.diffs","adj.mean.diffs.null.sd", "chisquare.test","std.diffs","z.scores","p.values","all"). *P* values reported are two-sided for the null hypothesis of no effect. The option "all" requests all measures. |
| ... | additional arguments passed to xBalance{RItools}. |

## Details

See help("xBalance") for details.

## Value

An object of class c("xbal", "list"). There are plot, print, and xtable methods for class xbal.

## Note

Evidence pertaining to the hypothesis that a treatment variable is not associated with differences in covariate values is assessed by comparing the differences of means (or regression coefficients), without standardization, to their distributions under hypothetical shuffles of the treatment variable, a permutation or randomization distribution. For the unstratified comparison, this reference distribution consists of differences (more generally, regression coefficients) when the treatment variable is permuted without regard to strata. For the stratified comparison, the reference distribution is determined by randomly permuting the treatment variable within strata, then recalculating the treatment-control differences (regressions of each covariate on the permuted treatment variable). Significance assessments are based on the large-sample normal approximation to these reference distributions.

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

### References

Hansen, B. B. and Bowers, J. (2008). Covariate balance in simple, stratified and clustered comparative studies. *Statistical Science*, 23(2):219–236.

Kalton, G. (1968). Standardization: A technique to control for extraneous variables. *Applied Statistics*, 17:118–136.

### Examples

```
library(uplift)

set.seed(12345)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

checkBalance(treat ~ X1 + X2 + X3 + X4 + X5 + X6 , data = dd)
```

---

`explore` *Explore data for uplift modeling*

---

### Description

This function provides a basic exploratory tool for uplift modeling, by computing the average value of the response variable for each predictor and treatment assignment.

### Usage

```
explore(formula,
        data,
        subset,
        na.action = na.pass,
        nbins = 4,
        continuous = 4,
        direction = 1)
```

### Arguments

| | |
|---|---|
| `formula` | a formula expression of the form response ~ predictors. A special term of the form `trt()` must be used in the model equation to identify the binary treatment variable. For example, if the treatment is represented by a variable named `treat`, then the right hand side of the formula must include the term `+trt(treat)`. |
| `data` | a data frame in which to interpret the variables named in the formula. |
| `subset` | an expression indicating which subset of the rows of data should be included. All observations are included by default. |
| `na.action` | a missing-data filter function. This is applied to the model frame after any subset argument has been used. Default is `na.action = na.pass`. |
| `nbins` | the number of bins created from numeric predictors. The bins are created based on quantiles, with a default value of 4 (quartiles). |
| `continuous` | specifies the threshold for when a variable is considered to be continuous (when there are at least `continuous` unique values). The default is 4. Factor variables are always considered to be categorical no matter how many levels they have. |

| | |
|---|---|
| direction | possible values are 1 (default) if uplift should be computed as the difference in the average response between treatment and control, or 2 between control and treatment. This only affects the uplift calculation as produced in the output. |

## Value

A list of matrices, one for each variable. The columns represent the number of responses over the control group, the number of the responses over the treated group, the average response for the control, the average response for the treatment, and the uplift (difference between treatment and control average response).

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

## Examples

```
library(uplift)

set.seed(12345)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

eda <- explore(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat), data = dd)
```

---

# modelProfile *Profile a fitted uplift model*

---

## Description

This function can be used to profile a fitted uplift model. Given a vector of scores (uplift predictions), it computes basic summary statistics for each predictor by score quantile.

## Usage

```
modelProfile(formula, data, groups = 10,
group_label = c("I", "D"), digits_numeric = 1, digits_factor = 4,
exclude_na = FALSE, LaTex = FALSE)
```

## Arguments

| | |
|---|---|
| formula | a formula expression of the form score ~ predictors, where the left hand side of the model formula should include the predictions from a fitted model. |
| data | a data frame in which to interpret the variables named in the formula. |
| groups | the number of groups of equal observations in which to partition the data set to show results. The default value is 10 (deciles). Other possible values are 5 and 20. |
| group_label | possible values are "I" or "D", for group number labels which are increasing or decreasing with the model score, respectively. |
| digits_numeric | the number of digits to show for numeric predictors. |
| digits_factor | the number of digits to show for factor predictors. |

| | |
|---|---|
| `exclude_na` | should the results exclude observations with missing values in any of the variables named in the formula? |
| `LaTex` | should the function output LaTex code? |

**Details**

This function ranks the variable supplied in the left hand side of the model formula and classifies it into groups of equal number of observations. It subsequently calls the function `tabular` from the `tables` package to compute the average of each numeric predictor and the distribution of each factor within each group.

**Value**

An object of S3 class `tabular`. See `help("tabular")` in the `tables` package for details.

**Author(s)**

Leo Guelman <leo.guelman@gmail.com>

**Examples**

```
library(uplift)

### Simulate data
set.seed(12345)
dd <- sim_pte(n = 1000, p = 5, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0) # required coding for upliftRF

### Fit upliftRF model
fit1 <- upliftRF(y ~ X1 + X2 + X3 + X4 + X5 + trt(treat),
                 data = dd,
                 mtry = 3,
                 ntree = 50,
                 split_method = "KL",
                 minsplit = 100,
                 verbose = TRUE)

### Fitted values on training data
pred <- predict(fit1, dd)

### Compute uplift predictions
uplift_pred <- pred[, 1] - pred[, 2]

### Put together data, predictions and add some dummy factors for illustration only
dd2 <- data.frame(dd, uplift_pred, F1 = gl(2, 50, labels = c("A", "B")),
                  F2 = gl(4, 25, labels = c("a", "b", "c", "d")))

### Profile data based on fitted model
modelProfile(uplift_pred ~ X1 + X2 + X3 + F1 + F2,
             data = dd2,
             groups = 10,
             group_label = "D",
             digits_numeric = 2,
             digits_factor = 4,
             exclude_na = FALSE,
             LaTex = FALSE)
```

niv *Adjusted net information value*

## Description

This function produces an adjusted net information value for each variable specified in the right hand side of the formula. This can be a helpful exploratory tool to (preliminarily) determine the predictive power of each variable for uplift.

## Usage

```
niv(formula, data, subset, na.action = na.pass, B = 10, direction = 1,
nbins = 10, continuous = 4, plotit = TRUE, ...)
```

## Arguments

| | |
|---|---|
| formula | a formula expression of the form response ~ predictors. A special term of the form `trt()` must be used in the model equation to identify the binary treatment variable. For example, if the treatment is represented by a variable named `treat`, then the right hand side of the formula must include the term `+trt(treat)`. |
| data | a data frame in which to interpret the variables named in the formula. |
| subset | an expression indicating which subset of the rows of data should be included. All observations are included by default. |
| na.action | a missing-data filter function. This is applied to the model frame after any subset argument has been used. Default is `na.action = na.pass`. |
| B | the number of bootstrap samples used to compute the adjusted net information value. |
| direction | if set to 1 (default), the net weight of evidence is computed as the difference between the weight of evidence of the treatment and control groups, or if 2, it is computed as the difference between the weight of evidence of the control and treatment groups. This will not change the adjusted net information value, but only the sign of the net weight of evidence values. |
| nbins | the number of bins created from numeric predictors. The bins are created based on quantiles, with a default value of 10 (deciles). |
| continuous | specifies the threshold for when a variable is considered to be continuous (when there are at least `continuous` unique values). The default is 4. Factor variables are always considered to be categorical no matter how many levels they have. |
| plotit | plot the adjusted net information value for each variable? |
| ... | additional arguments passed to `barplot`. |

## Details

The ordinary information value or IV (commonly used in credit scoring applications) is given by

$$IV = \sum_{i=1}^{G} (P(x = i|y = 1) - P(x = i|y = 0)) \times WOE_i$$

where $G$ is the number of groups created from a numeric predictor or categories from a categorical predictor, and the weight of evidence $WOE_i = ln(\frac{P(x=i|y=1)}{P(x=i|y=0)})$.

The net information value (NIV) is the natural extension of the IV for the case of uplift. It is computed as

$$NIV = 100 \times \sum_{i=1}^{G} (Q^T R^C - Q^C R^T) \times NWOE_i$$

where $Q^T = P(x = i|y = 1)^T$, $R^C = P(x = i|y = 0)^C$, $Q^C = P(x = i|y = 1)^C$, $R^T = P(x = i|y = 0)^T$, and $NWOE_i = WOE_i^T - WOE_i^C$ (the net weight of evidence).

The adjusted net information value is computed as follows:

1. Take $B$ bootstrap samples and compute the NIV for each variable on each sample.

2. Compute the mean and standard deviation of the NIV, $mean(NIV)$ and $sd(NIV)$, for each variable over all the $B$ bootstraps.

3. The adjusted NIV for a given variable is computed by subtracting a penalty term from the mean NIV: $mean(NIV) - \frac{sd(NIV)}{\sqrt{B}}$.

## Value

A list with two components:

`niv_val` — a matrix with the following columns: `niv` (the average net information value for each variable over all bootstrap samples), `penalty` (the penalty term calculated as described in the details above), and `adj_niv` (the difference between the prior two colums).

`nwoe` — a list of matrices, one for each variable. The columns represent the distribution of the responses $(y = 1)$ over the treated group (`ct1.y1`), the distribution of the non-responses $(y = 0)$ over the treated group (`ct1.y0`), the distribution of the responses $(y = 1)$ over the control group (`ct0.y1`), the distribution of the non-responses $(y = 0)$ over the control group (`ct0.y0`), the weight of evidence over the treated group (`ct1.woe`), the weight of evidence over the control group (`ct0.woe`), and the net weight of evidence (`nwoe`).

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

## References

Larsen, K. (2009). Net lift models. In *M2009 – 12th Annual SAS Data Mining Conference*.

## Examples

```
library(uplift)

set.seed(12345)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

niv.1 <- niv(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat), data = dd)
niv.1$niv
niv.1$nwoe
```

performance *Performance assessment for uplift models*

## Description

Provides a method for assessing performance for uplift models.

## Usage

```
performance(pr.y1_ct1, pr.y1_ct0, y, ct, direction = 1, groups = 10)
```

## Arguments

| | |
|---|---|
| pr.y1_ct1 | the predicted probability $P(y = 1|x)^T$. |
| pr.y1_ct0 | the predicted probability $P(y = 1|x)^C$. |
| y | the actual observed value of the response. |
| ct | a binary (numeric) vector representing the treatment assignment (coded as 0/1). |
| direction | possible values are 1 (default) if the objective is to maximize the difference in the response for Treatment minus Control, and 2 for Control minus Treatment. |
| groups | the number of groups of equal observations in which to partition the data set to show results. The default value is 10 (deciles). Other possible values are 5 and 20. |

## Details

Model performance is estimated by 1) computing the difference in the predicted conditional class probabilities $P(y = 1|x)^T$ and $P(y = 1|x)^C$, 2) ranking the difference and grouping it into 'buckets' with equal number of observations each, and 3) computing the actual difference in the mean of the response variable between the treatment and control groups for each bucket.

## Value

An object of class performance, which is a matrix with the following columns: (group) the number of groups, (n.ct1) the number of observations in the treated group, (n.ct0) the number of observations in the control group, (n.y1_ct1) the number of observations in the treated group with response = 1, (n.y1_ct0) the number of observations in the control group with response = 1, (r.y1_ct1) the mean of the response for the treated group, (r.y1_ct0) the mean of the response for the control group, and (uplift) the difference between r.y1_ct1 and r.y1_ct0 (if direction = 1).

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

## References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014c). Uplift random forests. *Cybernetics & Systems*. Accepted.

### Examples

```
library(uplift)

set.seed(123)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

### Fit uplift random forest

fit1 <- upliftRF(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat),
                 data = dd,
                 mtry = 3,
                 ntree = 100,
                 split_method = "KL",
                 minsplit = 200,
                 verbose = TRUE)
print(fit1)
summary(fit1)

### Get variable importance

varImportance(fit1, plotit = TRUE, normalize = TRUE)

### Predict on new data

dd_new <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd_new$treat <- ifelse(dd_new$treat == 1, 1, 0)

pred <- predict(fit1, dd_new)

### Evaluate model performance

perf <- performance(pred[, 1], pred[, 2], dd_new$y, dd_new$treat, direction = 1)
plot(perf[, 8] ~ perf[, 1], type ="l", xlab = "Decile", ylab = "uplift")
```

---

`predict.ccif` *Predictions from a fitted causal conditional inference forest model*

---

### Description

Prediction of new data using a causal conditional inference forest.

### Usage

```
## S3 method for class ccif
predict(object, newdata, n.trees = object$ntree, predict.all = FALSE, ...)
```

### Arguments

| | |
|---|---|
| `object` | an object of class `ccif`, as that created by the function `ccif`. |
| `newdata` | a data frame containing the values at which predictions are required. |
| `n.trees` | the number of trees used in the prediction; the default is `object$ntree`. |
| `predict.all` | should the predictions of all trees be kept? |
| `...` | not used. |

### Details

At the moment, all predictors passed for fitting the uplift model must also be present in `newdata`, even if they are not used as split variables by any of the trees in the forest.

### Value

If `predict.all = FALSE`, a matrix of predictions containing the conditional class probabilities: `pr.y1_ct1` represents $P(y = 1|x)^T$ and `pr.y1_ct0` represents $P(y = 1|x)^C$. This is computed as the average of the individual predictions over all trees.

If `predict.all = TRUE`, the returned object is a list with two components: `pred.avg` is the prediction (as described above) and `individual` is a list of matrices containing the individual predictions from each tree.

### Author(s)

Leo Guelman <leo.guelman@gmail.com>

### References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

### Examples

```
library(uplift)

### Simulate training data

set.seed(12345)
dd <- sim_pte(n = 100, p = 6, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

### Fit model

form <- as.formula(paste(y ~, trt(treat) +, paste(X, 1:6, sep = , collapse = "+")))

fit1 <- ccif(formula = form,
             data = dd,
             ntree = 50,
             split_method = "Int",
             pvalue = 0.05,
             verbose = TRUE)

### Predict on new data

dd_new <- sim_pte(n = 200, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)

pred <- predict(fit1, dd_new)
```

`predict.upliftRF` *Predictions from a fitted uplift random forest model*

### Description

Prediction of new data using an uplift random forest.

### Usage

```
## S3 method for class upliftRF
predict(object, newdata, n.trees = object$ntree, predict.all = FALSE, ...)
```

### Arguments

| | |
|---|---|
| `object` | an object of class `upliftRF`, as that created by the function `upliftRF`. |
| `newdata` | a data frame containing the values at which predictions are required. |
| `n.trees` | the number of trees used in the prediction; The default is `object$ntree`. |
| `predict.all` | should the predictions of all trees be kept? |
| `...` | not used. |

### Details

At the moment, all predictors passed for fitting the uplift model must also be present in `newdata`, even if they are not used as split variables by any of the trees in the forest.

### Value

If `predict.all = FALSE`, a matrix of predictions containing the conditional class probabilities: `pr.y1_ct1` represents $P(y = 1|x)^T$ and `pr.y1_ct0` represents $P(y = 1|x)^C$. This is computed as the average of the individual predictions over all trees.

If `predict.all = TRUE`, the returned object is a list with two components: `pred.avg` is the prediction (as described above) and `individual` is a list of matrices containing the individual predictions from each tree.

### Author(s)

Leo Guelman <leo.guelman@gmail.com>

### References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014c). Uplift random forests. *Cybernetics & Systems*. Accepted.

### Examples

```
library(uplift)

### Simulate data for uplift modeling

set.seed(123)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)
```

```
### Fit uplift random forest

fit1 <- upliftRF(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat),
                 data = dd,
                 mtry = 3,
                 ntree = 100,
                 split_method = "KL",
                 minsplit = 200,
                 verbose = TRUE)
summary(fit1)

### Predict on new data

dd_new <- sim_pte(n = 2000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd_new$treat <- ifelse(dd_new$treat == 1, 1, 0)

pred <- predict(fit1, dd_new)
head(pred)
```

---

qini *Computes the Qini coefficient q*

---

## Description

This function computes the Qini coefficient from a `performance` object (as created by the function `performance`).

## Usage

```
## S3 method for class performance
qini(x, direction = 1, plotit = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `x` | an object of class `performance`. |
| `direction` | possible values are `1` (default) if the objective is to maximize the difference in the response for Treatment minus Control, and `2` for Control minus Treatment. |
| `plotit` | plot the incremental gains from the fitted model? |
| `...` | additional arguments passed to `plot`. |

## Details

Qini coefficients represent a natural generalizations of the Gini coefficient to the case of uplift. Qini is defined as the area between the actual incremental gains curve from the fitted model and the area under the diagonal corresponding to random targeting. See the references below for details.

### Value

A list with the following components

| | |
|---|---|
| `Qini` | the Qini coefficient as defined above. |
| `inc.gains` | the incremental gain values from the fitted model. |
| `random.inc.gains` | the random incremental gains. |

### Author(s)

Leo Guelman <leo.guelman@gmail.com>

### References

Radcliffe, N. (2007). Using control groups to target on predicted lift: Building and assessing uplift models. *Direct Marketing Analytics Journal*, 2007:14–21.

Radcliffe, N. J. and Surry, P. D. (2011). Real-World Uplift Modelling with Significance-Based Uplift Trees. Technical Report, TR-2011-1, Portrait.

### Examples

```
library(uplift)

### Simulate data for uplift modeling

set.seed(123)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

### Fit uplift random forest

fit1 <- upliftRF(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat),
                 data = dd,
                 mtry = 3,
                 ntree = 100,
                 split_method = "KL",
                 minsplit = 200,
                 verbose = TRUE)
print(fit1)
summary(fit1)

### Predict on new data

dd_new <- sim_pte(n = 2000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd_new$treat <- ifelse(dd_new$treat == 1, 1, 0)

pred <- predict(fit1, dd_new)

### Evaluate model performance

perf <- performance(pred[, 1], pred[, 2], dd_new$y, dd_new$treat, direction = 1)

### Compute Qini coefficient

Q <- qini(perf, plotit = TRUE)
Q
```

rvtu *Response variable transform for uplift modeling*

## Description

This function transforms the data frame supplied in the function call by creating a new response variable and an equal number of control and treated observations. This transformed data set can be subsequently used with any conventional supervised learning algorithm to model uplift.

## Usage

```
rvtu(formula, data, subset, na.action = na.pass,
     method = c("undersample", "oversample", "weights", "none"))
```

## Arguments

| | |
|---|---|
| formula | a formula expression of the form response ~ predictors. A special term of the form `trt()` must be used in the model equation to identify the binary treatment variable. For example, if the treatment is represented by a variable named `treat`, then the right hand side of the formula must include the term `+trt(treat)`. |
| data | a data frame in which to interpret the variables named in the formula. |
| subset | an expression indicating which subset of the rows of data should be included. All observations are included by default. |
| na.action | a missing-data filter function. This is applied to the model frame after any subset argument has been used. Default is `na.action = na.pass`. |
| method | the method used to create the transformed data set. It must be one of `"undersample"`, `"oversample"`, `"weights"`, or `"none"`, with no default. See details below. |

## Details

The transformed response variable $z$ equals 1 if the observation has a response value of 1 and has been treated, or if it has a response value of 0 and has not been treated. Intuitively, $z$ equals 1 if we know that, for a given case, the outcome in the treatment group would have been at least as good as in the control group, had we known for this case the outcome in both groups. Under equal proportion of control and treated observations, it is easy to prove that $2P(z = 1|x) - 1 = P(y = 1|x)^T - P(y = 1|x)^C$ (Jaskowski and Jaroszewicz, 2012).

If the data has an equal number of control and treated observations, then `method = "none"` must be used. Otherwise, any of the other methods must be used.

If `method = "undersample"`, a random sample without replacement is drawn from the treated class (i.e., treated/control) with the majority of observations, such that the returned data frame will have balanced treated/control proportions.

If `method = "oversample"`, a random sample with replacement is drawn from the treated class with the minority of observations, such that the returned data frame will have balanced treated/control proportions.

If `method = "weights"`, the returned data frame will have a weight variable $w$ assigned to each observation. The weight assigned to the treated (control) observation equals $1-$ proportion of treated observations (proportion of treated observations).

### Value

A data frame including the predictor variables (RHS of the formula expression), the treatment ($ct = 1$) and control ($ct = 0$) assignment, the original response variable (LHS of the formula expression), and the transformed response variable for uplift modeling $z$. If `method = "weights"`, an additional weight variable $w$ is included.

### Author(s)

Leo Guelman <leo.guelman@gmail.com>

### References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Jaskowski, M. and Jaroszewicz, S. (2012) Uplift Modeling for Clinical Trial Data. In *ICML 2012 Workshop on Machine Learning for Clinical Data Analysis*, Edinburgh, Scotland.

### Examples

```
library(uplift)

### Simulate data

set.seed(1)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

### Transform response variable for uplift modeling
dd2 <- rvtu(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat), data = dd, method = "none")

### Fit a Logistic model to the transformed response
glm.uplift <- glm(z ~ X1 + X2 + X3 + X4 + X5 + X6, data = dd2, family = "binomial")

### Test fitted model on new data
dd_new <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd_new$treat <- ifelse(dd_new$treat == 1, 1, 0)
pred <- predict(glm.uplift, dd_new, type = "response")
perf <- performance(2 * pred - 1, rep(0, length(pred)), dd_new$y, dd_new$treat,
direction = 1)
perf
```

---

## `sim_pte` *Simulations for personalized treatment effects*

---

### Description

Numerical simulation for treatment effect heterogeneity estimation based on Tian et al. (2014).

### Usage

```
sim_pte(n = 1000, p = 20, rho = 0, sigma = sqrt(2), beta.den = 4)
```

## Arguments

| | |
|---|---|
| n | the number of observations. |
| p | the number of predictors. |
| rho | the covariance between predictors. |
| sigma | the multiplier of the error term. |
| beta.den | the size of the main effects relative to the interaction effects. See details below. |

## Details

sim_pte simulates data according to the following specification:

$$Y = I\Big(\sum_{j=1}^{p} \beta_j X_j + \sum_{j=1}^{p} \gamma_j X_j T + \sigma_0 \epsilon > 0\Big)$$

where $\gamma = (1/2, -1/2, 1/2, -1/2, 0, ..., 0)$, $\beta = (-1)^{j+1} I(3 \leq j \leq 10)/$beta.den, $(X_1, \ldots, X_p)$ follows a mean zero multivariate normal distribution with a compound symmetric variance-covariance matrix, $(1-\rho)\mathbf{I}_p + \rho \mathbf{1}^T \mathbf{1}$, $T = [-1, 1]$ is the treatment indicator and $\epsilon$ is $N(0, 1)$.

In this case, the "true" treatment effect score $(P(Y = 1|T = 1) - P(Y = 1|T = -1))$ is given by

$$\Phi\left(\frac{\sum_{j=1}^{p}(\beta_j + \gamma_j)X_j}{\sigma_0}\right) - \Phi\left(\frac{\sum_{j=1}^{p}(\beta_j - \gamma_j)X_j}{\sigma_0}\right).$$

## Value

A data frame including the response variable ($Y$), the treatment (treat=1) and control (treat=-1) assignment, the predictor variables ($X$) and the "true" treatment effect score (ts).

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

## References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Tian, L., Alizadeh, A., Gentles, A. and Tibshirani, R. (2014). A simple method for detecting interactions between a treatment and a large number of covariates. Submitted.

## Examples

```
library(uplift)
### Simulate training data

set.seed(12345)
dd <- sim_pte(n = 1000, p = 10, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0) # required coding for upliftRF

### Fit model
```

```
form <- as.formula(paste(y ~, trt(treat) +, paste(X, 1:10, sep = , collapse = "+")))

fit1 <- upliftRF(formula = form,
                 data = dd,
                 ntree = 100,
                 split_method = "Int",
                 interaction.depth = 3,
                 minsplit = 100,
                 minbucket_ct0 = 50,
                 minbucket_ct1 = 50,
                 verbose = TRUE)
summary(fit1)
```

---

`tian_transf` *Modify covariates for uplift modeling*

---

### Description

This function transforms the data frame supplied in the function call by creating a new set of modified covariates and an equal number of control and treated observations. This transformed data set can be subsequently used with any conventional supervised learning algorithm to model uplift.

### Usage

```
tian_transf(formula, data, subset, na.action = na.pass,
 method = c("undersample", "oversample", "none"),
 standardize = TRUE, cts = FALSE)
```

### Arguments

| | |
|---|---|
| `formula` | a formula expression of the form response ~ predictors. A special term of the form `trt()` must be used in the model equation to identify the binary treatment variable. For example, if the treatment is represented by a variable named `treat`, then the right hand side of the formula must include the term `+trt(treat)`. |
| `data` | a data frame in which to interpret the variables named in the formula. |
| `subset` | an expression indicating which subset of the rows of data should be included. All observations are included by default. |
| `na.action` | a missing-data filter function. This is applied to the model frame after any subset argument has been used. Default is `na.action = na.pass`. |
| `method` | the method used to create the transformed data set. It must be one of "undersample", "oversample" or "none", with no default. See details. |
| `standardize` | if `TRUE`, each variable is standardized to have unit $L2$ norm, otherwise it is left alone. Default is `TRUE`. |
| `cts` | if `TRUE`, contrasts for factors are created in a special way. See details. Default is `FALSE`. |

## Details

The covariates $x$ supplied in the right hand side of the model formula are transformed as $w = z \times T/2$, where $T = [-1, 1]$ is the treatment indicator and $z$ is the matrix of standardize $x$ variables.

If `cts = TRUE`, factors included in the formula are converted to dummy variables in a special way that is more appropriate when the returned model frame is used to fit a penalized regression. In this case, contrasts used for factors are given by penalized regression contrasts from the `penalized` package. Unordered factors are turned into as many dummy variables as the factor has levels, except when the number of levels is 2, in which case it returns a single contrast. This ensures a symmetric treatment of all levels and guarantees that the fit does not depend on the ordering of the levels. See `help(contr.none)` in `penalized` package. Ordered factors are turned into dummy variables that code for the difference between successive levels (one dummy less than the number of levels). See `help(contr.diff)` in `penalized package`.

If the data has an equal number of control and treated observations, then `method = "none"` should be used. Otherwise, any of the other methods should be used.

If `method = "undersample"`, a random sample without replacement is drawn from the treated class (i.e., treated/control) with the majority of observations, such that the returned data frame will have balanced treated/control proportions.

If `method = "oversample"`, a random sample with replacement is drawn from the treated class with the minority of observations, such that the returned data frame will have balanced treated/control proportions.

## Value

A model frame, including the modified covariates $w$ (the prefix "T_" is added to the name of each covariate to denote it has been modified), the treatment ($ct = 1$) and control ($ct = 0$) assignment and the response variable (LHS of model formula). The intercept is omitted from the model frame.

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

## References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Tian, L., Alizadeh, A., Gentles, A. and Tibshirani, R. (2014). A simple method for detecting interactions between a treatment and a large number of covariates. Submitted.

## Examples

```
library(uplift)

set.seed(1)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

dd2 <- tian_transf(y ~  X1 + X2 + X3 + trt(treat), data =dd, method = "none")
head(dd2)
```

`trt` *Mark treatment term*

### Description

This is a dummy function, used to mark the treatment term in various functions within the `uplift` package.

### Usage

```
trt(x)
```

### Arguments

| | |
|---|---|
| `x` | a numeric variable coded as 1 (treatment) and 0 (control). |

### Value

`x`, unchanged.

### Author(s)

Leo Guelman <leo.guelman@gmail.com>

`upliftKNN` *Uplift k-nearest-neighbor*

### Description

`upliftKNN` implements *k*-nearest-neighbor for uplift modeling.

### Usage

```
upliftKNN(train, test, y, ct, k = 1, dist.method = "euclidean",
          p = 2, ties.meth = "min", agg.method = "mean")
```

### Arguments

| | |
|---|---|
| `train` | a matrix or data frame of training set cases. |
| `test` | a matrix or data frame of test set cases. A vector will be interpreted as a row vector for a single case. |
| `y` | a numeric response variable (must be coded as 0/1 for binary response). |
| `ct` | a factor or numeric vector representing the treatment to which each training case is assigned. At least 2 groups are required (e.g. treatment and control). Multi-treatments are also supported. |
| `k` | the number of neighbors considered. |
| `dist.method` | the distance to be used in calculating the neighbors. Any method supported in function `dist` is valid. |
| `p` | the power of the Minkowski distance. |

| | |
|---|---|
| `ties.meth` | the method used to handle ties for the *k*th neighbor. The default is `"min"` which uses all ties. Alternatives include `"max"` which uses none if there are ties for the *k*th nearest neighbor, `"random"` which selects among the ties randomly and `"first"` which uses the ties in their order in the data. |
| `agg.method` | the method used to combine responses of the nearest neighbors. Defaults to `"mean"`; the alternative is `"majority"`. |

## Details

*k*-nearest-neighbor for uplift modeling for a test set from a training set. For each case in the test set, the *k* nearest training set vectors for each treatment type are found. The response value for the *k* nearest training vectors is aggregated based on the function specified in `agg.method`. For `"majority"`, classification is decided by majority vote (with ties broken at random).

## Value

A matrix of predictions for each test case and value of `ct`.

## Note

The code logic follows closely the `knn` and `knnflex` packages, the latter currently discontinued from CRAN.

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

## References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Su, X., Kang, J., Fan, J., Levine, R. A., and Yan, X. (2012). Facilitating score and causal inference trees for large observational studies. *Journal of Machine Learning Research*, 13(10): 2955-2994.

## Examples

```
library(uplift)

### Simulate data for uplift modeling

set.seed(1)

train <- sim_pte(n = 500, p = 10, rho = 0, sigma =  sqrt(2), beta.den = 4)
train$treat <- ifelse(train$treat == 1, 1, 0)

### Fit an uplift k-nearest-neighbor on test data

test <- sim_pte(n = 100, p = 10, rho = 0, sigma =  sqrt(2), beta.den = 4)
test$treat <- ifelse(test$treat == 1, 1, 0)

fit1 <- upliftKNN(train[, 3:8], test[, 3:8], train$y, train$treat, k = 1,
           dist.method = "euclidean", p = 2, ties.meth = "min",
           agg.method = "majority")
head(fit1)
```

`upliftRF` *Uplift random forests*

### Description

`upliftRF` implements random forests with split criteria designed for binary uplift modeling tasks.

### Usage

```
## S3 method for class formula
upliftRF(formula, data, ...)

## Default S3 method:
upliftRF(
x,
y,
ct,
mtry = floor(sqrt(ncol(x))),
ntree = 100,
split_method = c("ED", "Chisq", "KL", "L1", "Int"),
interaction.depth = NULL,
bag.fraction = 0.5,
minsplit = 20,
minbucket_ct0 = round(minsplit/4),
minbucket_ct1 = round(minsplit/4),
keep.inbag = FALSE,
verbose = FALSE,
...)

## S3 method for class upliftRF
print(x, ...)
```

### Arguments

| | |
|---|---|
| `data` | a data frame containing the variables in the model. It should include a variable reflecting the binary treatment assignment of each observation (coded as 0/1). |
| `x, formula` | a data frame of predictors or a formula describing the model to be fitted. A special term of the form `trt()` must be used in the model equation to identify the binary treatment variable. For example, if the treatment is represented by a variable named `treat`, then the right hand side of the formula must include the term `+trt(treat)`. |
| `y` | a binary response (numeric) vector. |
| `ct` | a binary (numeric) vector representing the treatment assignment (coded as 0/1). |
| `mtry` | the number of variables to be tested in each node; the default is `floor(sqrt(ncol(x)))`. |
| `ntree` | the number of trees to generate in the forest; default is `ntree = 100`. |

| | |
|---|---|
| split_method | the split criteria used at each node of each tree; Possible values are: "ED" (Euclidean distance), "Chisq" (Chi-squared divergence), "KL" (Kullback-Leibler divergence), "L1" ($L1$-norm divergence), and "Int" (Interaction method). |
| interaction.depth | the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. The default is to grow trees to maximal depth, constrained on the arguments specified in minsplit and minbucket. |
| bag.fraction | the fraction of the training set observations randomly selected for the purpose of fitting each tree in the forest. |
| minsplit | the minimum number of observations that must exist in a node in order for a split to be attempted. |
| minbucket_ct0 | the minimum number of control observations in any terminal <leaf> node. |
| minbucket_ct1 | the minimum number of treatment observations in any terminal <leaf> node. |
| keep.inbag | if set to TRUE, an nrow(x) by ntree matrix is returned, whose entries are the "in-bag" samples in each tree. |
| verbose | print status messages? |
| ... | optional parameters to be passed to the low level function upliftRF.default. |

## Details

Uplift random forests estimate *personalized treatment effects* (also called uplift) by binary recursive partitioning. The algorithm and split methods are described in Guelman et al. (2014a, 2014c).

## Value

An object of class upliftRF, which is a list with the following components:

| | |
|---|---|
| call | the original call to upliftRF. |
| trees | the tree structure that was learned. |
| split_method | the split criteria used at each node of each tree. |
| ntree | the number of trees used. |
| mtry | the number of variables tested at each node. |
| var.names | a character vector with the name of the predictors. |
| var.class | a character vector containing the class of each predictor variable. |
| inbag | an nrow(x) by ntree matrix showing the in-bag samples used by each tree. |

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

### References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014c). Uplift random forests. *Cybernetics & Systems*. Accepted.

Su, X., Tsai, C., Wang, H., Nickerson, D., and Li, B. (2009). Subgroup Analysis via Recursive Partitioning. *Journal of Machine Learning Research*, 10:141-158.

### Examples

```
library(uplift)

### Simulate data for uplift modeling

set.seed(123)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

### Fit uplift random forest

fit1 <- upliftRF(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat),
                 data = dd,
                 mtry = 3,
                 ntree = 100,
                 split_method = "KL",
                 minsplit = 200,
                 verbose = TRUE)
print(fit1)
summary(fit1)
```

---

## varImportance *Extract variable importance from* upliftRF- *or* ccif*-fitted objects*

---

### Description

This is the extractor function for variable importance of predictors.

### Usage

```
## S3 method for class upliftRF
varImportance(x, n.trees = x$ntree, plotit = TRUE, normalize = TRUE, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class upliftRF or ccif. |
| n.trees | the number of trees used in the prediction; The default is x$ntree. |
| plotit | plot variable importance? |
| normalize | if set to TRUE, the importance is scaled to add up to 100. |
| ... | additional arguments passed to barplot. |

## Details

At each split in each tree, the improvement in the split criterion is the importance measure attributed to the splitting variable; this is accumulated over all the trees in the forest separately for each variable.

## Value

A numeric vector with the variable importance.

## Author(s)

Leo Guelman <leo.guelman@gmail.com>

## References

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014c). Uplift random forests. *Cybernetics & Systems*. Accepted.

## Examples

```
library(uplift)

### Simulate data for uplift modeling

set.seed(123)
dd <- sim_pte(n = 1000, p = 20, rho = 0, sigma =  sqrt(2), beta.den = 4)
dd$treat <- ifelse(dd$treat == 1, 1, 0)

### Fit uplift random forest

fit1 <- upliftRF(y ~ X1 + X2 + X3 + X4 + X5 + X6 + trt(treat),
                 data = dd,
                 mtry = 3,
                 ntree = 100,
                 split_method = "KL",
                 minsplit = 200,
                 verbose = TRUE)
print(fit1)

### Get variable importance

varImportance(fit1, plotit = TRUE, normalize = TRUE)
```

# Appendix B: Manuscripts linked to chapters

**Chapters** 2, 3, 5, and 6:

Guelman, L., Guillén, M. and Pérez-Marín, A. M. (2014a). Optimal personalized treatment rules for marketing interventions: A review of methods, a new proposal, and an insurance case study. Submitted.

Guelman, L., Guillén, M. and Pérez-Marín, A. M. (2014b). A survey of personalized treatment models for pricing strategies in insurance. *Insurance: Mathematics and Economics*, 58:68–76.

**Chapters** 4, 7, and 8:

Guelman, L., Guillén, M. and Pérez-Marín, A. M. (2012). Random forests for uplift modeling: An insurance customer retention case. In Engemann, K. J., Lafuente, A. M. G., and Merigó, J. M., editors, *Modeling and Simulation in Engineering, Economics and Management*, pages 123–133. Springer Berlin Heidelberg, New York, NY.

Guelman, L., Guillén, M. and Pérez-Marín, A. M. (2014c). Uplift random forests. *Cybernetics & Systems*. Accepted.

**Chapter** 9:

Guelman, L. and Guillén, M. (2014). A causal inference approach to measure price elasticity in automobile insurance. *Expert Systems with*

*Applications*, 41(2):387–396.

**Chapter** 10:

Guelman, L. (2012). Gradient boosting trees for auto insurance loss cost modeling and prediction. *Expert Systems with Applications*, 39(3):3659–3667.

**Chapter** 11:

Guelman, L. (2014). *uplift: Uplift modeling*. R package version 0.3.5.