Uplift Regression: The R Package `tools4uplift`

Mouloud Belbahri *[1], Alejandro Murua[1], Olivier Gandouet[2], and Vahid Partovi Nia[3]

[1] *Department of Mathematics and Statistics, University of Montreal, Canada*
[2] *Department of Advanced Analytics, TD Insurance, Canada*
[3] *Montreal Research Center, Huawei Technologies Co., Ltd.*

Abstract

Uplift modeling aims at predicting the causal effect of an action such as a medical treatment or a marketing campaign on a particular individual, by taking into consideration the response to a treatment. The treatment group contains individuals who are subject to an action; a control group serves for comparison. Uplift modeling is used to order the individuals with respect to the value of a causal effect, e.g., positive, neutral, or negative. Though there are some computational methods available for uplift modeling, most of them exclude statistical regression models. The R Package **tools4uplift** intends to fill this gap. This package comprises tools for: i) quantization, ii) visualization, iii) feature selection, and iv) model validation.

1 Introduction

The term causal study refers to a study that tries to discover a cause-effect relationship. If there is a causal relationship between two events, the events are highly dependent. However, the converse might not be true, since association is not necessarily causation. If a clinical trial study is performed to isolate the causal effect, association and causation coincide. However, most frequently, the available data comes from observational studies, where causation and association\correlation differ. Most statistical models are concerned with correlation because they are used on observational data. In such cases, some adjustments are required to draw causal statements from the statistical models. It appears that this is not always obvious to some practitioners.

The statistical framework for causal inference was formally introduced by Rubin (1974). This framework is also associated with the potential outcome framework of Neyman (1923), also known as the Rubin causal model (Holland, 1986). A potential outcome is the theoretical response each unit would have manifested, had it been assigned to a particular treatment. Under randomization, these outcomes are independent of the assignment other patients receive. In practice, potential outcomes for an individual cannot be observed. A single unit is only assigned to either treatment or control, making direct observations in the other condition (called the *counterfactual* condition) and the observed individual causal effects, impossible. This is well-known as the fundamental problem of causal inference (Holland, 1986). Often, in a randomized experiment, researchers focus on the estimation of average treatment effects and the effect of the treatment is determined from this estimate. However, there might be a proportion of the population that may respond favorably to the treatment, and another proportion that may not, depending on whether or not individual treatment effects vary widely in the population. A decision based on an average treatment effect for a new arriving individual would require a baseline adjustment because of the heterogeneity in treatment response originated by many biologic, genetic and environmental characteristics.

In marketing, *response models* (Hanssens et al., 2003) of client behavior are based on historical data. They are used to predict the probability that a client responds to a marketing campaign, e.g., the client buys a product. Marketing campaigns using response models concentrate on clients associated with a high probability of a positive response. However, this strategy does not ensure a purchase. On the other hand,

[*] Corresponding author: mouloud.belbahri@gmail.com

customers may buy the product without any marketing effort. Therefore, it is important to extract the cause of the purchase and to isolate the effect of marketing. *Uplift models* (Radcliffe and Surry, 1999; Hansotia and Rukstales, 2001; Lo, 2002) provide a solution to the problem of isolating the marketing effect. Instead of modeling the different response or class probabilities, *uplift* attempts to model the difference between conditional response probabilities in the treatment and control groups. Uplift modeling aims at identifying groups of individuals on which a predetermined action will have the most positive effect.

In the R Package **tools4uplift** presented here, we make available to practitioners a combination of tools for uplift modeling, including some novel techniques introduced in this paper. Our package comprises tools for: i) quantization, ii) visualization, iii) feature selection, and iv) model validation, alongside their associated functions. We hope that the package will enable practitioners to save time and effort when analyzing their uplift data.

The methods implemented in the R Package **tools4uplift** are related to, but distinct from the ones implemented in the R Package **uplift** (Guelman, 2014). The functions included in **uplift** are designed for building and testing the uplift models proposed by Guelman et al. (2015). It focuses on the adaptation of non-parametric machine learning classifiers such as random forests and k-nearest neighbours. The R Package **tools4uplift** offers a complementary set of functions targeting regression models such as logistic regression. It focuses on building regression models adapted for uplift; it proposes two distinct methods for quantization and visualization of continuous variables; and it introduces a method to perform automatic variable selection in uplift regression models. Finally, the R Package **tools4uplift** also includes model validation functions.

The remaining of the paper is organized as follows. Section 2 introduces the notation, and discusses the general uplift modeling methodology, alongside its statistical background and its implementation in R. Section 3 shows an application of the proposed methodology to real data using **tools4uplift**. Some final remarks and conclusions are given in Section 4.

2 Uplift models

In marketing, we are interested in the conditional probability that a client buys a product given that he was targeted by a marketing campaign (the treatment group). We also want to measure the conditional probability that a client buys the product given that he was not targeted (the control group). Uplift attempts to model the difference between conditional class probabilities in the treatment and control groups. The variable of interest has two possible outcomes: whether or not the purchase is made.

The logistic regression model is a widely used statistical model that uses a logistic function to model a binary dependent variable. It is easy to implement and has an elegant interpretation, thanks, in particular, to the odds ratio. The odds ratio is the ratio that compares the change in odds of buying a product for two different sets of values of the factors in the model, e.g., change in age, gender, etc. The logistic regression model is in part more popular than other binary-outcome models because odds ratios are readily available.

A customer base is a historical list of clients to whom a business sold products and services. This list can be segmented along two dimensions in function of the response value (yes or no), and the associated treatment (yes or no), given rise to the following groups (Kane et al., 2014):

1. the “persuadables” who respond to the marketing action because they are targeted,
2. the “sure” individuals who respond whether or not they are targeted,
3. the “lost” individuals who do not respond, regardless of whether or not they are targeted, and
4. the “do not disturb” individuals who are less likely to respond, just because they are targeted.

In general, the interesting customers from a marketing point of view are the “persuadables” and the “do not disturb”. The persuadables provide incremental responses whereas the “do not disturb” individuals should not be disturbed because the marketing campaign has a negative effect on them. Uplift modeling attempts to separate customers into the four groups described above. The intuitive approach is to build two classification models. Recall that the uplift is the difference between two conditional probabilities. Hansotia and Rukstales (2001) proposed an indirect method to estimate the uplift based on a two-model approach. This consists of fitting two separated conditional probability models: one for the treated individuals, and another for the untreated individuals. The uplift is estimated as the difference between these two conditional probability models. The asset of this technique is its simplicity. However, both models focus on predicting only a one-class probability instead of making an effort to predict the uplift. Any conventional statistical or algorithmic binary-outcome classification method may serve to fit these models. In order to improve the accuracy of the two-model approach, Lo (2002) proposed an interaction model. Interactions may arise when considering the relationship among three or more variables, and describes a situation in which the simultaneous influence of two variables on a third is not additive. The methodology is based on adding explicit interaction terms between each covariate and the treatment indicator using a standard logistic regression. The parameters of the interaction terms measure the additional effect of each covariate because of the treatment. As in the two-model approach, an indirect estimation of the uplift is achieved by subtracting the predicted probabilities associated with the control group from the probabilities associated with the treatment group.

Other approaches to uplift modeling try to directly model the difference in conditional success probabilities between the treatment and control groups. Most current active research is in this direction. Such methods are mainly adaptation of three types of machine learning algorithms: a) decision tree learners (Rzepakowski and Jaroszewicz (2010), Radcliffe and Surry (2011), Guelman et al. (2015), Sołtys et al. (2015) or Zhao et al. (2017)), b) regression models adapted to the uplift (Radcliffe (2007) or Jaskowski and Jaroszewicz (2012)) and c) support vector machines for uplift (Zaniewicz and Jaroszewicz (2013), Kuusisto et al. (2014) or Zaniewicz and Jaroszewicz (2017)).

To formalize the problem, let $y \in \{0,1\}$ be a binary response variable, $\mathbf{x} = (x_1, \ldots, x_p)$ a vector of explanatory variables (predictors), and $\tau \in \{0,1\}$ the treatment indicator variable. The binary variable τ indicates if unit i is exposed to treatment ($\tau = 1$) or control ($\tau = 0$). Suppose that n independent units are observed

$$(y_i, \mathbf{x}_i, \tau_i),\ i = 1, \ldots, n.$$

Denote the potential outcomes under control and treatment by $\{y_i \mid \tau_i = 0\}$ and $\{y_i \mid \tau_i = 1\}$ respectively. The uplift model estimates

$$u(\mathbf{x}_i) = \Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i = 1) - \Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i = 0),\ i = 1, \ldots, n. \tag{1}$$

2.1 The two-model estimator

The *two-model* estimator (Hansotia and Rukstales, 2001) consists in the subtraction of logistic regression models for the treated and untreated populations.

Definition 1. Let

$$\Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i = 1, \beta_0^{(1)}, \boldsymbol{\beta}^{(1)}) = \frac{1}{1 + \exp\{-(\beta_0^{(1)} + \mathbf{x}_i^\top \boldsymbol{\beta}^{(1)})\}}$$

and

$$\Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i = 0, \beta_0^{(0)}, \boldsymbol{\beta}^{(0)}) = \frac{1}{1 + \exp\{-(\beta_0^{(0)} + \mathbf{x}_i^\top \boldsymbol{\beta}^{(0)})\}},$$

where $(\beta_0^{(k)}, \boldsymbol{\beta}^{(k)})$ for $k = \{0,1\}$ are the logistic regression parameters for control ($k = 0$) and treatment ($k = 1$) groups, and the superscript $^\top$ denote transposition. The two-model estimator predicts the uplift associated with a covariate vector $\mathbf{x}_{n+1}$ for a future individual as

$$\hat{u}(\mathbf{x}_{n+1}) = \frac{1}{1 + \exp\{-(\hat{\beta}_0^{(1)} + \mathbf{x}_{n+1}^\top \hat{\boldsymbol{\beta}}^{(1)})\}} - \frac{1}{1 + \exp\{-(\hat{\beta}_0^{(0)} + \mathbf{x}_{n+1}^\top \hat{\boldsymbol{\beta}}^{(0)})\}},$$

where $(\hat{\beta}_0^{(k)}, \hat{\boldsymbol{\beta}}^{(k)})$ for $k = \{0,1\}$ are the maximum likelihood estimates for each group. The R Package **tools4uplift** provides a straightforward implementation of this model with the functions `DualUplift()` and `DualPredict()`. The arguments are

```
DualUplift(data, treat, outcome, predictors)
DualPredict(data, treat, outcome, model, nb.group = 10, plotit = FALSE)
```

where `data, treat` and `outcome` are necessary arguments in order to fit and predict the two-model estimator with respect to `predictors`. The data frame `data` must contain the treatment, outcome and predictors variables. The names of these variables are used as the arguments of the `DualUplift()` function. Then, in order to predict the uplift for a new observation, the output of the `DualUplift()` function needs to be passed as the `model` argument of the `DualPredict()` function. Its remaining arguments `(nb.group, plotit)` are used for validation purposes and are explained in Section 2.3.

2.2 The interaction model estimator

The *interaction model* (Lo, 2002) uses a standard logistic regression with first order interactions terms as follows:

Definition 2. Let

$$\log\left(\frac{\Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i, \beta_0, \boldsymbol{\beta}, \gamma, \boldsymbol{\delta})}{1 - \Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i, \beta_0, \boldsymbol{\beta}, \gamma, \boldsymbol{\delta})}\right) = \beta_0 + \mathbf{x}_i^\top \boldsymbol{\beta} + \gamma\tau_i + \tau_i \mathbf{x}_i^\top \boldsymbol{\delta}$$

or equivalently

$$\Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i, \beta_0, \boldsymbol{\beta}, \gamma, \boldsymbol{\delta}) = \frac{1}{1 + \exp\{-(\beta_0 + \mathbf{x}_i^\top \boldsymbol{\beta} + \gamma\tau_i + \tau_i \mathbf{x}_i^\top \boldsymbol{\delta})\}},$$

where $(\beta_0, \boldsymbol{\beta}, \gamma, \boldsymbol{\delta})$ are the logistic regression parameters. The predicted uplift associated with the covariate vector $\mathbf{x}_{n+1}$ of a future individual is estimated by

$$\hat{u}(\mathbf{x}_{n+1}) = \frac{1}{1 + \exp\{-(\hat{\beta}_0 + \mathbf{x}_{n+1}^\top \hat{\boldsymbol{\beta}} + \hat{\gamma} + \mathbf{x}_{n+1}^\top \hat{\boldsymbol{\delta}})\}} - \frac{1}{1 + \exp\{-(\hat{\beta}_0 + \mathbf{x}_{n+1}^\top \hat{\boldsymbol{\beta}})\}},$$

where $(\hat{\beta}_0, \hat{\boldsymbol{\beta}}, \hat{\gamma}, \hat{\boldsymbol{\delta}})$ are the maximum likelihood estimates. The implementation of the interaction model estimator in R follows the same logic as the one of the two-model in Section 2.1. The functions `InterUplift()` and `InterPredict()` have the following arguments

```
InterUplift(data, treat, outcome, predictors, input = c("all", "best"))
InterPredict(data, treat, outcome, model, nb.group = 10, plotit = FALSE)
```

where the arguments `(data, treat, outcome, predictors, model, nb.group, plotit)` have the same role as in the `DualUplift()` and `DualPredict()` functions. The argument `input = c("all", "best")` is important because it specifies which model to use. If this argument is set to `"all"`, the function `InterUplift()` uses the list of predictors given in the argument `predictors` to create the interaction terms between the `treat` variable and the `predictors`, so as to fit the interaction model. The option `input = "best"` stands for "best features". In this case, `InterUplift()` uses the list of the selected main variables and interaction terms provided by the method `BestFeatures()` described later in Section 2.3 which performs variable selection. The output of `BestFeatures()` is exactly the list of the selected main variables and interaction terms for the interaction model.

2.3 Model validation and selection

Model validation is 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 units. Assessing model performance is more complex for uplift modeling, as the actual value of the response, that is, the *true* treatment effect, is unknown at the individual subject level. However, one can assess model performance by comparing groups of observations exposed to different treatments.

Most often used in economics, the Gini coefficient (Gini, 1997) aims at measuring the model's goodness-of-fit and is one of the measures used in direct marketing for traditional response models. One way of computing the Gini coefficient is to first draw a Lorenz curve (Lorenz, 1905). The plot depicting the Lorenz

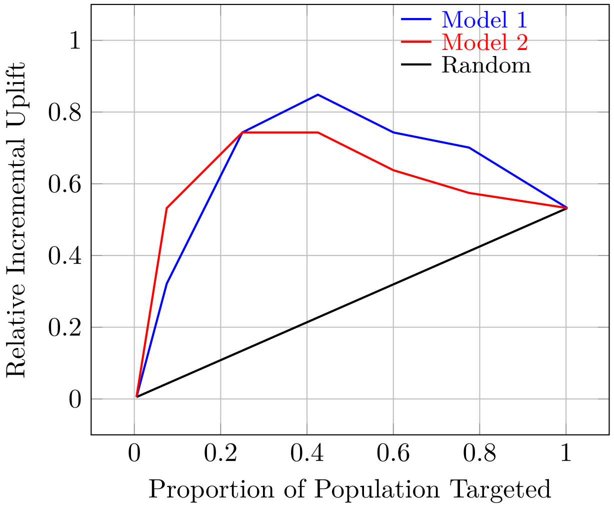

Figure 1: Example of Qini curves corresponding to two different uplift models compared to a random targeting strategy.

curve illustrates the goodness-of-fit of a response model. The predicted scores of the targeted observations are sorted in decreasing order. The horizontal axis represents the observed cumulative percentages associated to the sorted predicted scores with respect to the whole targeted sample. The vertical axis, the Lorenz curve, depicts the ratio of the cumulative response lift associated with each cumulative percentage to the total number of responses. The Gini coefficient is a single index of model performance based on the Lorenz curve. Radcliffe (2007) proposes a straightforward extension of the Lorenz curve and the Gini coefficient for uplift modeling: the *Qini curve* and the *Qini coefficient*. Basically, the Qini curve is a Lorenz curve where the predictive scores are replaced by the predicted uplifts. The intuition is that a good model should be able to select individuals with positive uplift first. More explicitly, for a given model, let $\hat{u}_{(1)} \geq \hat{u}_{(2)} \geq ... \geq \hat{u}_{(n)}$ be the sorted predicted uplifts. Let $\phi \in [0,1]$ be a given proportion and let $N_\phi = \{i : \hat{u}_i \geq \hat{u}_{(\phi n)}\} \subset \{1,\ldots,n\}$ be the subset of individuals with the $\phi n \times 100\%$ highest predicted uplifts $\hat{u}_i$. As a function of the fraction of population targeted ϕ, the incremental uplift or Qini curve is defined as

$$h(\phi) = \sum_{i \in N_\phi} y_i \tau_i - \sum_{i \in N_\phi} y_i (1-\tau_i) \frac{\sum\limits_{i \in N_\phi} \tau_i}{\sum\limits_{i \in N_\phi} (1-\tau_i)},$$

with $h(0) = 0$, by definition. For any $\phi \in [0,1]$, the relative incremental uplift $g(\phi)$ is given by $g(\phi) = h(\phi) / \sum\limits_{i=1}^{n} \tau_i$. Note that $g(1) = u_{\text{overall}}$ where u_{overall} is the overall observed uplift

$$u_{\text{overall}} = \frac{\sum\limits_{i=1}^{n} y_i \tau_i}{\sum\limits_{i=1}^{n} \tau_i} - \frac{\sum\limits_{i=1}^{n} y_i (1-\tau_i)}{\sum\limits_{i=1}^{n} (1-\tau_i)}.$$

The Qini curve is constructed by plotting $g(\phi)$ as a function of $\phi \in [0,1]$. This is illustrated in Figure 1. The figure can be interpreted as follows: the x-axis represents the fraction of targeted individuals and the y-axis shows the incremental number of positive responses relative to the total number of targeted individuals. The straight line between the points $(0,0)$ and $(1, u_{\text{overall}})$ in Figure 1 represents a benchmark to compare the performance of the model to a strategy that would randomly target subjects. The Qini coefficient q is a single index of model performance. It is defined as the area under the Qini curve. This area can be approximated using a Riemann sum such as the trapezoid formula: the domain of $\phi \in [0,1]$ is partitioned into J panels, or $J+1$ grid points $0 = \phi_1 < \phi_2 < ... < \phi_{J+1} = 1$, to define the Qini coefficient q as

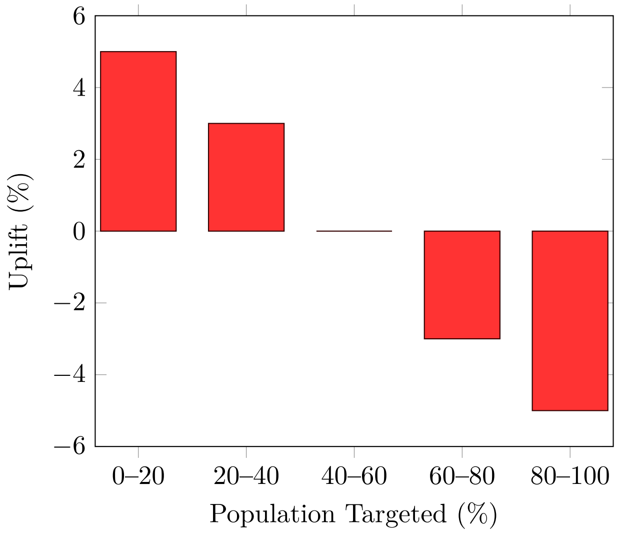

Figure 2: Example of a Qini barplot with 5 panels corresponding to an uplift model. A good model should order the observed uplift from highest to lowest.

$$q = \int_0^1 h(\phi)\mathrm{d}\phi \approx \frac{1}{2}\sum_{j=1}^{J}(\phi_{j+1} - \phi_j)\{h(\phi_{j+1}) + h(\phi_j)\}.$$

In general, when comparing several models, the preferred model is the one with maximum Qini coefficient. A combination of four functions (`QiniTable()`, `QiniArea()`, `QiniCurve()` and `QiniBarPlot()`) is available in the R Package **tools4uplift** for model evaluation based on the Qini. The first function is `QiniTable()`

```
QiniTable(data, treat, outcome, prediction, nb.group = 10)
```

where `data, treat, outcome` are the necessary arguments in order to fit an uplift model and `prediction` is the predicted uplift value for the `data`. The uplift values could be the output of the `DualPredict()`, `InterPredict()`, or any other statistical method that gives an uplift prediction. The `nb.group` argument represents the J panels used in order to construct the Qini curve and compute the Qini coefficient. The number of panels is usually $J \geq 2$ and, depending on the available data points, could be as large as the user would like. In practice, practitioners present the results with 10 groups (deciles). The following functions use `x`, the output of the `QiniTable()` as an input in order to compute the Qini coefficient and to plot the Qini curve.

```
QiniArea(x)
QiniCurve(x, title = "Model Performance: Qini Curve", color = NULL)
```

Using the results from the `QiniTable()` function, one can also draw a barplot representing the observed uplift between two grid points j and $j+1$ for $j = 0, ..., J$, as a function of the predicted uplift by the model, as shown in Figure 2. This is done with the following function.

```
QiniBarPlot(x, title = "Model Performance: Uplift by Group", color = NULL)
```

Model selection refers to selecting the right (or best) model according to a certain criteria. It is usually accomplished by selecting a subset of the variables available in a given dataset. Model selection is useful because it reduces the dimension of the model, avoids over-fitting, and improves model stability and accuracy. When the input space dimension is small, knowledge-based approaches to identify a good set of variables can easily be performed and is sometimes preferable. In other situations, we may have a large number of potentially important variables and it soon becomes a time consuming effort to follow a manual variable selection process. In this case, we may consider using automatic subset selection tools. Popular linear variable selection techniques are forward, backward, stepwise (Montgomery et al., 2012), and stage-wise selection (Hastie et al., 2007), as well as more recent techniques such as lasso (Tibshirani, 1996), and lar (Efron et al., 2004), among others. However, these techniques have not been designed for uplift models, so they need to be adapted. In this work, we have chosen to adapt lasso because of its popularity and success in selecting variables when dealing with complex and high-dimensional models. We suggest a two-stage approach. Our adapted lasso algorithm chooses the regularization hyper-parameter, that is, the penalty parameter, in adequacy with uplift models performance measures, i.e., by maximizing the Qini coefficient q.

Consider the interaction model of Section 2.2. Let $\lambda > 0$ be the penalty constant. For any given λ, let $(\hat{\beta}_0(\lambda), \hat{\boldsymbol{\beta}}(\lambda), \hat{\gamma}(\lambda), \hat{\boldsymbol{\delta}}(\lambda))$ be the value of the parameters that maximizes the penalized likelihood

$$\sum_{i=1}^{n} \left\{ y_i \log\left(\frac{p_i}{1-p_i}\right) + \log(1-p_i) \right\} + \lambda \|(\boldsymbol{\beta}, \gamma, \boldsymbol{\delta})\|_1,$$

where

$$p_i = \Pr(y_i = 1 \mid \mathbf{x}_i, \tau_i, \beta_0, \boldsymbol{\beta}, \gamma, \boldsymbol{\delta}) = \frac{1}{1 + \exp\{-(\beta_0 + \mathbf{x}_i^\top \boldsymbol{\beta} + \gamma\tau_i + \tau_i \mathbf{x}_i^\top \boldsymbol{\delta})\}}.$$

Let $q(\lambda)$ be the Qini coefficient associated with the model with parameters $(\hat{\beta}_0(\lambda), \hat{\boldsymbol{\beta}}(\lambda), \hat{\gamma}(\lambda), \hat{\boldsymbol{\delta}}(\lambda))$. Our lasso procedure solves

$$(\hat{\beta}_0(\hat{\lambda}), \hat{\boldsymbol{\beta}}(\hat{\lambda}), \hat{\gamma}(\hat{\lambda}), \hat{\boldsymbol{\delta}}(\hat{\lambda})) = \underset{\lambda}{\operatorname{argmax}}\ q(\lambda). \tag{2}$$

Using the `glmnet()` function from the **glmnet** R Package in order to generate the regularization path, we defined a new function `LassoPath()` that is callable directly from the R Package **tools4uplift**. This function is used inside the function `BestFeatures()` which returns the variables and interaction terms that maximize the Qini coefficient. The arguments of the function are

```
BestFeatures(data, treat, outcome, predictors, nb.lambda = 100,
             nb.group = 10, validation = FALSE, p = 0.3, value = FALSE)
```

where `data, treat, outcome` and `predictors` are defined as above. The argument `nb.lambda` is the number of different λ values that are used for model fitting.

If `validation` is set to `TRUE`, the function performs a two-fold cross-validation. By default, the validation set is set to a randomly chosen 30% of the data, `p = 0.3`. If `value` is set to `TRUE`, the function prints the λ that maximizes the Qini coefficient q as well as its value. Finally, the function returns a vector of names of the selected features. The output of the function can be used directly in the `InterUplift()` function in order to fit the second stage of the modeling process. The second stage of the modeling process estimates the coefficients of the selected variables by maximizing the non penalized likelihood.

2.4 Data manipulation

Data manipulation is an important aspect of statistical analysis. Feature engineering, exploration of missing values patterns, outliers detection and descriptive statistics are useful to get insight about the collected data to formalize the research question and must be performed before fitting any model. Quantization transforms a continuous variable into a categorical variable. Quantization of continuous variables into bins is extremely useful when trying to model non-linearity in the data. Alternatives consist of finding a good transformation such as splines. Quantization is also useful for storing data instances in fewer bits. A variable with 2^k categories can be embedded in only k bits. Existing algorithms for optimal partitioning of a continuous variable are suitable to response modeling but not to uplift modeling (Garcia et al., 2013). In practice, when exploring uplift data, the partition is performed with two options: equal length intervals and equal frequency intervals. For example, the bins are based on the deciles of the variable in the `niv()` function from **uplift** R Package. Here, we suggest a univariate supervised quantization tree-based algorithm for optimal partitioning similar to CART (Breiman et al., 1984) with a modified splitting criterion based on hypothesis testing for uplift. The same idea is extended to bivariate quantization in order to look for for potential interactions. Interactions may arise when considering the relationship among three or more variables, and describes a situation in which the simultaneous influence of two variables on a third is not additive. We build a nonparametric supervised quantization algorithm guided by the observed uplift, where the two-dimensional feature space is divided in rectangles. In addition, the R Package **tools4uplift** provides visualization tools for both quantization methods: uplift barplots for the univariate case, and heatmaps for the bivariate case.

Univariate supervised quantization. Suppose that we have n observations, and that we want to quantize a given continuous explanatory variable X. The objective is to partition the sample Ω (or root node) into two child nodes Ω_{left} and Ω_{right} based on X so that

$$u(X \mid \Omega_{\text{left}}) \neq u(X \mid \Omega_{\text{right}}), \tag{3}$$

at a pre-specified statistically significant level α. Therefore, we need to find the splitting point $X = x$ associated with the minimum p value of the uplift test satisfying p value $\leq \alpha$. The procedure is then repeated recursively into each child node until the stopping rule is satisfied.

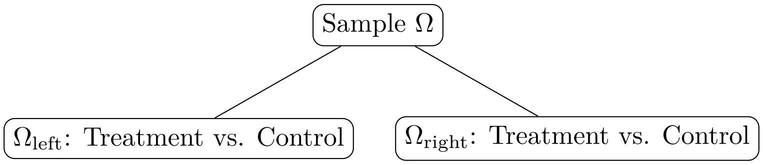

Figure 3: For one explanatory variable, the sample Ω is divided into four groups: left child node treatment group, left child node control group, right child node treatment group, and right child node control group. The goal is to find the most statistically significant split at a given significance level α (that is, a split satisfying p value $\leq \alpha$) along a given variable X.

To simplify the notation, let $g = 1, 2, 3, 4$ denote one of the following four groups: 1 = left node treatment group, 2 = left node control group, 3 = right node treatment group, and 4 = right node control group. Let p_g be the proportion of responses in group g, n_g be the number of observations in group g, and $n = \sum_{g=1}^{4} n_g$. The uplift statistical test in (3) can be written in terms of these proportions as follows:

$$\begin{cases} \mathrm{H}_0 : \ p_1 - p_2 = p_3 - p_4 \\ \mathrm{H}_1 : \ p_1 - p_2 \neq p_3 - p_4 \end{cases},$$

or equivalently

$$\begin{cases} \mathrm{H}_0 : \ p_1 - p_3 = p_2 - p_4 \\ \mathrm{H}_1 : \ p_1 - p_3 \neq p_2 - p_4 \end{cases}.$$

Let $\hat{p}_g$ be an estimator of p_g, $g = 1, 2, 3, 4$, that only depends on the data in the corresponding control or treatment group, that is, that does not use the samples from the other group. Under the assumption of randomization between treatment and control groups, $\hat{p}_1 - \hat{p}_3$ (treatment only) and $\hat{p}_2 - \hat{p}_4$ (control only) are independent. However, within each group, for instance, the treatment group, $\hat{p}_1$ and $\hat{p}_3$ are not independent. In order to build a statistical test, we need the expectation and variance of these estimators. Let us first represent the treatment group split in Table 1. Denote

$p_T = \Pr(y = 1 \mid \tau = 1)$: probability of success in the treatment group,
$n_T = \text{card}\{i : X_i < x\}$: number of observations in the left node treatment group,
N_T : number of observations in the treatment group,

and let Z be the random variable that counts the number of successes in the left node treatment group. It is easily seen that $Z \sim \text{Hypergeometric}(n_T, p_T N_T, N_T)$ where n_T is the number of draws, $p_T N_T$ is the number of successes in the population, and N_T is the population size. The random variable Z has the following properties

$$\mathbb{E}[Z] = n_T p_T, \tag{4}$$

$$\mathbb{V}[Z] = n_T\{p_T(1-p_T)\}\left(\frac{N_T - n_T}{N_T - 1}\right), \tag{5}$$

where $\mathbb{E}[.]$ stands for the mathematical expectation and $\mathbb{V}[.]$ stands for variance.

	Left Node	Right Node	Total
Success	z	$p_T N_T - z$	$p_T N_T$
Fail	$n_T - z$	$(1-p_T)N_T - n_T + z$	$(1-p_T)N_T$
Total	n_T	$N_T - n_T$	N_T

Table 1: Contingency table for treatment group observations split into left and right nodes.

We consider the following unbiased estimators based on the above table:

$$\hat{p}_1 = \frac{z}{n_T},$$
$$\hat{p}_3 = \frac{p_T N_T - z}{N_T - n_T}.$$

Using the Hypergeometric distribution properties (4) and (5), it is easy to show that

$$\mathbb{E}[\hat{p}_1 - \hat{p}_3] = \frac{N_T \mathbb{E}[Z]}{n_T(N_T - n_T)} - \frac{n_T p_T N_T}{n_T(N_T - n_T)} = 0,$$
$$\mathbb{V}[\hat{p}_1 - \hat{p}_3] = \frac{N_T^2 \mathbb{V}[Z]}{\{n_T(N_T - n_T)\}^2} = \frac{N_T^2 p_T(1-p_T)}{n_T(N_T - n_T)(N_T - 1)}.$$

The same development applies to the control group. The statistics associated with the uplift test

$$\begin{cases} \mathrm{H}_0: \ (p_1 - p_3) - (p_2 - p_4) = 0 \\ \mathrm{H}_1: \ (p_1 - p_3) - (p_2 - p_4) \neq 0 \end{cases}$$

is based on the asymptotic pivotal quantity

$$z_{\text{obs}} = \frac{(\hat{p}_1 - \hat{p}_2) - (\hat{p}_3 - \hat{p}_4)}{\sqrt{\mathbb{V}\{(\hat{p}_1 - \hat{p}_2) - (\hat{p}_3 - \hat{p}_4)\}}} \tag{6}$$

where, because of the assumption of independence between treatment and control groups samples,

$$\mathbb{V}\{(\hat{p}_1 - \hat{p}_2) - (\hat{p}_3 - \hat{p}_4)\} = \frac{N_T^2 p_T(1-p_T)}{n_T(N_T - n_T)(N_T - 1)} + \frac{N_C^2 p_C(1-p_C)}{n_C(N_C - n_C)(N_C - 1)}.$$

By the Central Limit Theorem, the statistics given by the right-hand-side of equation (6) is asymptotically normally distributed under the null hypothesis; therefore the test rejects H_0 at a level α when

$$\mid z_{\text{obs}} \mid > z_{\frac{\alpha}{2}}$$

where z_α denotes the upper-tail α-percentile of the standard normal distribution.

Algorithm 1. The algorithm we propose works as described in the following pseudo-code. Let X be a continuous explanatory variable and $m > 1$ be the number of possible split points:

1. Compute the range of the variable: $r = X_{(\max)} - X_{(\min)}$
2. Compute the distance between the split points $\frac{r}{m}$.
3. Generate a vector of all splits points $x_j = X_{(\min)} + j\frac{r}{m}$, for $j = 1, ..., m$.
4. Go through each value X_i and examine each candidate split point x_j:
    (a) If $X_i < x_j$, the case goes to the left child node;
    (b) Otherwise, the case goes to the right child node.
    (c) The best split point is the one that minimizes the p value of the uplift test.
5. Repeat step 4 for each child node until the stopping rule is satisfied.
6. Output the terminal nodes or leaves and final split points.

Remark 1. If X is a categorical (ordinal or nominal) explanatory variable with a large number K of different categories, we set the number of split points to test $m = K - 1$. We transform the categorical variable into an ordinal variable by sorting in an increasing manner its categories according to their corresponding uplift values. Using the ranking of these categories, we create a continuous variable which we feed into the above quantization algorithm. This idea is useful in practice as a complexity reduction mechanism when K is very large.

The function that performs the optimal partitioning is called `BinUplift()`. Its arguments are

```
BinUplift(data, treat, outcome, x, n.split = 10,
          alpha = 0.05, n.min = 30,
          ylim = NULL, ylab = "Uplift",
          title = "Binning Results", color = NULL)
```

where `data, treat, outcome` are the arguments for the data, treatment indicator and outcome variable of interest. The `x` argument is the name of the explanatory variable to quantize by trying `n.split` equidistant values in the range of the variable. The arguments `alpha` and `n.min` control the performance of the statistical test. `alpha` is the significance level of the test; `n.min` is the minimum number of observations in each group (treatment or control) required to consider a split. The remaining arguments specify plotting options. The function returns a barplot for variables that are successfully quantized. If it is not possible to quantize the variable at a level `alpha`, the function returns a message indicating that no split was possible at the given significance level.

Several variables can be quantized separately with one call to the function `BinUpliftEnhanced()`, which takes the same arguments as `BinUplift()`, except for `var.list`, which is a list containing the names of the variables to quantize. The function tries to quantize all these variables at a specified significance level `alpha`, and, if successful, it returns an augmented dataset with the quantized variables and a trace specifying which were quantized, and which were not.

Bivariate supervised quantization. Next, suppose that we want to quantize simultaneously two continuous explanatory variables X_1 and X_2 so as to construct a single categorical interaction variable $X_{1,2}$. The idea is to partition the plane into disjoint rectangles S based on their associated observed uplifts

$$u_S = \frac{\sum\limits_{i \in S} y_i \tau_i}{\sum\limits_{i \in S} \tau_i} - \frac{\sum\limits_{i \in S} y_i (1 - \tau_i)}{\sum\limits_{i \in S} (1 - \tau_i)}.$$

Algorithm 2. The algorithm we propose works as described in the following pseudo-code. Let X_1 and X_2 be two continuous explanatory variables, $b > 1$ be the number of intervals each variable will be cut into and $c \geq 2$ be the number of categories of the categorical variable $X_{1,2}$.

1. Find the minimum and the maximum values of X_1 and X_2.
2. Divide the feature space $\{X_{1,\min}, X_{1,\max}\} \times \{X_{2,\min}, X_{2,\max}\}$ into b^2 rectangles.
3. Compute the observed uplift in each rectangle.
4. Predict the individual uplift of each observation by the observed uplift of its rectangle u_S.
5. Create a new categorical variable $X_{1,2}$ with c categories sorted from the highest to the lowest predicted uplift.

Remark 2. The parameters b and c can be set to the optimizers of a cross-validation criterion based on the Qini coefficient.

The function that creates the heatmap and the associated bivariate qunatization is called `SquareUplift()`. Its arguments are

```
SquareUplift(data, var1, var2, treat, outcome, n.split = 10,
             n.min = 1, categorize = TRUE, nb.group = 3,
             plotit = TRUE, nb.col = 20)
```

where `data` is a data frame containing the variables of interest `var1, var2`. The arguments `n.split` and `nb.group` correspond to the parameters b and c of Algorithm 2. For visualization purposes, the argument `plotit` is set by default to `TRUE`. The function returns a heatmap of observed uplifts per rectangle containing a minimum of `n.min` observations per treatment and control groups. `SquareUplift()` also returns an augmented dataset with two new variables: a continuous variable `Uplift_var1_var2`, representing the observed uplift within each of the `n.split` × `n.split` rectangles, and a categorical variable `Cat_var1_var2` with `nb.group` categories.

3 Application

In this section, we analyze a publicly available dataset from a marketing campaign (Hillstrom, 2008) using the R Package **tools4uplift**. The data contain records of 64,000 customers who last purchased a product within twelve months. The individuals were randomly assigned to three groups; two groups were targeted by two different e-mail campaigns and one group served as control. The treatment assignment was performed in a randomized experiment fashion: a third of the individuals were randomly chosen to receive an e-mail campaign featuring men merchandise, another third were randomly chosen to receive an e-mail campaign featuring women merchandise, and the last third, the control group, did not receive any form of initiative. The results were tracked during a period of two weeks following the e-mail campaign. Some questions can be answered with an uplift model: What is the incremental response of customers targeted by any of two campaigns? Is there a way to optimally select the subset of customers that should be targeted? Conversely, Is there a subset of customers that should be removed from future campaigns? The historical customer attributes available include `recency` which indicates the number of months since the last purchase; `history` which is the amount in dollars spent in the past year; two binary variables indicating if the customer purchased `men` merchandise or `women` merchandise in the past year; the `zip_code` of the customer categorized as urban, suburban or rural; an indicator variable `newbie` indicating if the customer is a new customer in the past twelve months; and the `channel` from which the customer purchased in the past year, i.e., by phone, web or both. The treatment allocation variable included in the dataset is `segment`. In this application, we only focus on the target variable `visit` which is a binary variable indicating whether or not the customer visited the website. Moreover, to simplify the analysis, we restrict the treatment data to the treatment group `treat = 1` that received e-mail on women merchandise, and to the control group `treat = 0` that received no e-mail. The overall observed uplift for this marketing campaign is $u_{\text{overall}} = 4.5\%$.

Baseline models

First, we use the function `SplitUplift()` in order to split the dataset into training and test datasets with respect to the overall uplift. It is important to partition the data into subsets that keep the same distribution of treated versus nontreated and responders versus nonresponders. This is achieved by specifying the stratification variables in the argument `group = c("treat", "visit")`.

```
R>set.seed(1988)
R>split.data1 <- SplitUplift(data = data1, p = 0.7, group = c("treat", "visit"))
R>train <- split.data1[[1]]
R>valid <- split.data1[[2]]
```

Using the two-model estimator of Section 2.1, and the interaction model estimator of Section 2.2, we fit two baseline models for comparison purposes. First, we fit the two-model estimator using the following code

```
R># baseline model on train set: fitting the two-model estimator
R>base.tm <- DualUplift(train, "treat", "visit",
+                          predictors = colnames(train[, 1:9]))
```

The function returns a list of two elements. The first element is the baseline model fitted for nontreated individuals and the second is the baseline model fitted for treated individuals. Each model is a `glm` object which fits a logistic regression to each group.

```
R># baseline model for control group
R>base.tm[[1]]

Call:  glm(formula = model_formula, family = binomial(link = "logit"),
    data = mydata0)

Coefficients:
         (Intercept)               recency
          -2.1557961            -0.0675804
             history                  mens
           0.0007079             0.5428172
              womens        zip_code_Rural
           0.4789285             0.4931095
  zip_code_Surburban                newbie
           0.0631602            -0.6875501
channel_Multichannel         channel_Phone
          -0.2495831            -0.3802874

Degrees of Freedom: 14963 Total (i.e. Null);  14954 Residual
Null Deviance:     10100
Residual Deviance: 9696  AIC: 9716

R># baseline model for treatment group
R>base.tm[[2]]

Call:  glm(formula = model_formula, family = binomial(link = "logit"),
    data = mydata1)

Coefficients:
         (Intercept)               recency
          -2.0427178            -0.0425278
             history                  mens
           0.0004829             0.3444088
              womens        zip_code_Rural
           0.8500028             0.2901685
  zip_code_Surburban                newbie
```

```
          0.0425791            -0.4949860
channel_Multichannel         channel_Phone
         -0.0977911            -0.2400491

Degrees of Freedom: 14920 Total (i.e. Null);  14911 Residual
Null Deviance:     12540
Residual Deviance: 12140  AIC: 12160
```

Using the validation set, the function `DualPredict()` predicts the uplift.

```
R># predict the uplift on the validation set
R>base.tm.valid <- DualPredict(valid, "treat", "visit", model = base.tm,
+                                 nb.group = 5)[[1]]
```

Finally, to evaluate the quality of the baseline model, we compute the Qini coefficient with `QiniArea()`. Here, we use `nb.group = 5` to evaluate all the models.

```
R># evaluate the model performance
R>base.tm.perf <- QiniTable(base.tm.valid, "treat", "visit", "uplift_prediction",
+                            nb.group = 5)
R>QiniCurve(base.tm.perf, title = "")
R>QiniBarPlot(base.tm.perf, title = "")
R>QiniArea(base.tm.perf)
[1] 0.7236409
```

Next, we fit the interaction model estimator, and compare it to the two-model estimator. Both models perform similarly on the validation set. Figure 4 shows the performance of the interaction model using the functions `QiniCurve()` and `QiniBarPlot()`.

```
R># baseline model on train set: fitting the interaction estimator
R>base.inter <- InterUplift(train, "treat", "visit",
+                            predictors = colnames(train[, 1:9]), input = "all")
R>base.inter
Call:  glm(formula = model_formula, family = binomial(link = "logit"),
    data = data)

Coefficients:
              (Intercept)                      treat
               -2.1557961                  0.1130783
                  recency                    history
               -0.0675804                  0.0007079
                     mens                     womens
                0.5428172                  0.4789285
           zip_code_Rural         zip_code_Surburban
                0.4931095                  0.0631602
                   newbie       channel_Multichannel
               -0.6875501                 -0.2495831
            channel_Phone              treat:recency
               -0.3802874                  0.0250526
            treat:history                 treat:mens
               -0.0002250                 -0.1984084
             treat:womens       treat:zip_code_Rural
                0.3710743                 -0.2029410
  treat:zip_code_Surburban               treat:newbie
               -0.0205812                  0.1925641
```

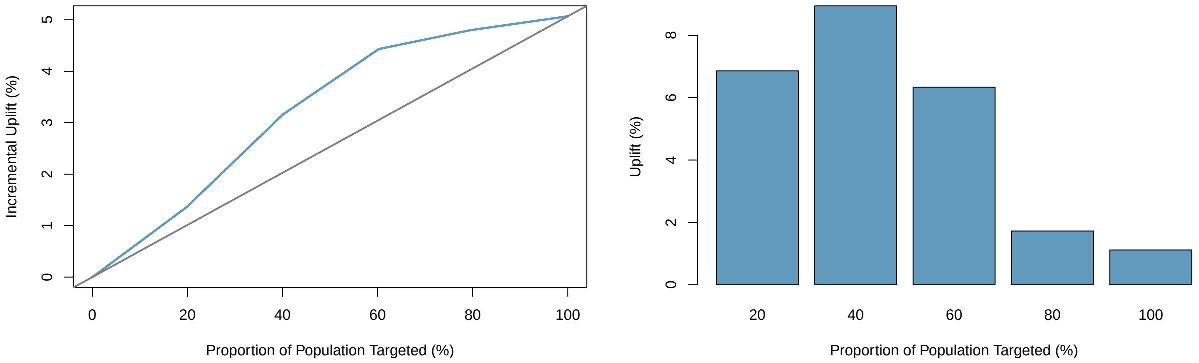

Figure 4: Performance of the interaction baseline model of Section 2.2 on a validation set. On the left panel, we see that the Qini coefficient is positive and outperforms random targeting ($q = 0.72$). On the right panel, we observe that the model does not sort ideally the individuals to target. A good model should order the observed uplift from highest to lowest (see Figure 2). The resulting object `QiniTable` is visualized using the `QiniCurve()` command (left panel) and the `QiniBarPlot()` command (right panel).

```
treat:channel_Multichannel         treat:channel_Phone
                 0.1517920                   0.1402384

Degrees of Freedom: 29884 Total (i.e. Null);  29865 Residual
Null Deviance:      22760
Residual Deviance: 21840  AIC: 21880
```

Once the coefficients of the logistic regression are estimated, we predict the uplift for the individuals in the validation set, and evaluate the quality of the model with the Qini functions.

```
R># predict the uplift on the validation set
R>base.inter.valid <- InterPredict(valid, "treat", "visit", model = base.inter,
+                                  nb.group = 5)[[1]]
R>
R># evaluate the model performance
R>base.inter.perf <- QiniTable(base.inter.valid, "treat", "visit",
+                              "uplift_prediction", nb.group = 5)
R>QiniCurve(base.inter.perf, title = "")
R>QiniBarPlot(base.inter.perf, title = "")
R>QiniArea(base.inter.perf)
[1] 0.7236409
```

Univariate quantization

The dataset contains two continuous variables, `recency` and `history`. We quantize both variables using the function `BinUplift()`. Figure 5 displays the barplots associated with the two quantizations.

```
R>bin.recency <- BinUplift(data = data1, treat = "treat", outcome = "visit",
+                          x = "recency", n.split = 12, alpha = 0.05,
+                          n.min = 30, ylim = c(0, 0.1), title="")

R>bin.recency
$‘out.tree‘
[1] "oups..no significant split"
```

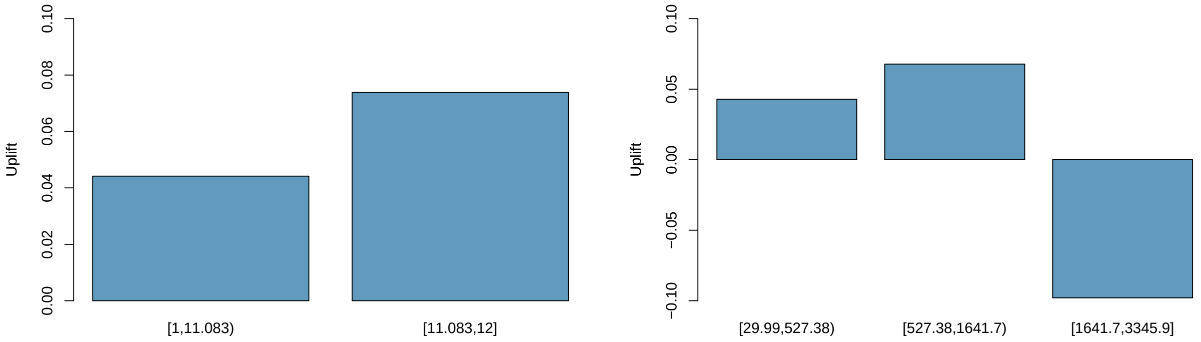

Figure 5: Univariate quantization with respect to the observed uplift. We can see that for `recency` (left panel), the optimal solution gives two groups with significantly ($\alpha = 0.10$) different positive uplift values. For `history` (right panel), the optimal solution ($\alpha = 0.05$) gives three groups. Note that the uplift distribution is non linear with respect to the historical expenses. The barplots were output by the function `BinUplift()`.

For a significance level of $\alpha = 0.05$, the decision tree does not find any significant partition of the data with respect to the `recency` variable. Hence, one can either keep the variable as continuous in the models or increase the level of significance α. For $\alpha = 0.10$, there is indeed a significant split. The following code implements the quantization of `recency` and `history`.

```
R># change the level of signification from 5% to 10%
R>bin.recency <- BinUplift(data = data1, treat = "treat", outcome = "visit",
+                           x = "recency", n.split = 12, alpha = 0.10,
+                           n.min = 30, ylim = c(0, 0.1), title="")
[1] "The variable recency has been cut at:"
[1] 11.08333
```

```
R>bin.history <- BinUplift(data = data1, treat = "treat", outcome = "visit",
+                           x = "history", n.split = 100, alpha = 0.05,
+                           n.min = 30, ylim = c(-0.1, 0.1), title="")
[1] "The variable history has been cut at:"
[1] 527.381
[1] "The variable history has been cut at:"
[1] 1641.715
```

Bivariate quantization

Searching for a possible interaction between `recency` and `history` with respect to the uplift, we use the function `SquareUplift()` in order to visualize the interaction in a heatmap and create a new categorical variable based on Algorithm 2 of Section 2.4.

The following code returns an augmented dataset with two new variables: `Uplift_recency_history`, a continuous variable representing the observed uplift within each of the `n.split` × `n.split` rectangles, and a categorical variable `Cat_recency_history` with `nb.group` categories.

```
R>data1 <- SquareUplift(data1, "recency", "history", "treat", "visit",
+                        n.split = 3, nb.group = 2)
```

The function also returns the associated heatmap displayed in Figure 6.

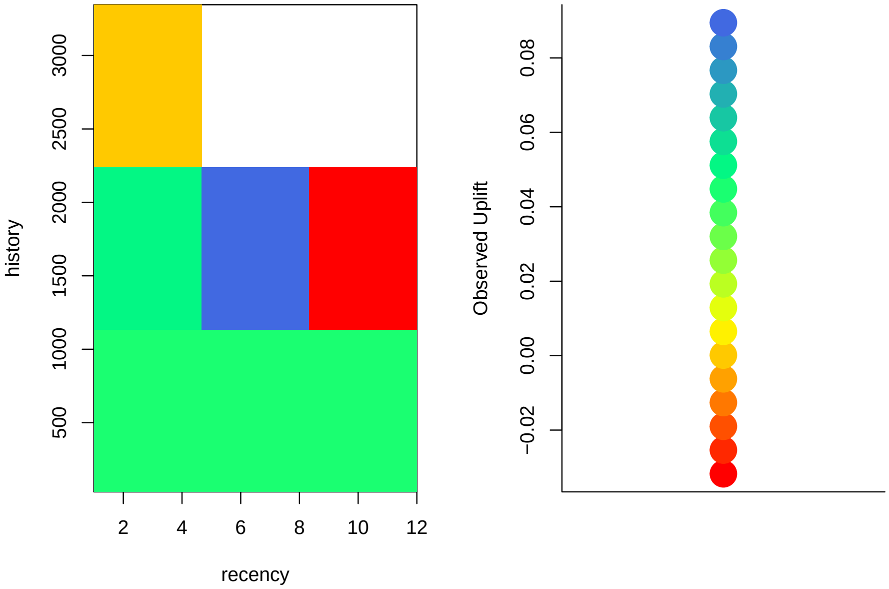

Figure 6: Bivariate quantization with respect to the observed uplift. By default, the `SquareUplift()` command returns the associated heatmap. The left panel shows the heatmap based on $b^2 = 9$ rectangles. Note that for customers that spent less than \$ 1,000 in the past year, the number of months since last purchase does not affect the observed uplift. On the other hand, the observed uplift is dependent on the recency of the last purchase for customers that spent more than \$ 1,000. The heatmap colors are based on the rainbow palette with the red color representing the lowest uplift and the blue color representing the highest uplift.

Model selection and comparison

The objective of this section is to improve the fitting of the interaction baseline model by performing variable selection. This is done using the `BestFeatures()` method. We compare several models that differ in the number and type of explanatory variables. For example, we compare the fittings with the quantized version of continuous variables against models fitted with the original variables. More specifically, we fit the interaction model estimator of Section 2.3 with different versions of the variables `recency` and `history`. We let the function `BestFeatures()` select the best set of predictors. The baseline model uses the original variables.

Other models are fitted using either the quantized `recency` and the original `history`, or the original `recency` and the quantized `history`, or both quantized variables. Another model is fitted using the `Cat_recency_history` categorical variable from the bivariate quantization algorithm. The following code implements the interaction model based on the best selected features (that is, those giving highest Qini coefficient). In this model both `recency` and `history` are the original continuous variables. The idea is to try to improve the baseline model with feature selection based on the Qini coefficient.

```
R># baseline with feature selection
R># feature selection using the lasso path
R>features <- BestFeatures(data = train, treat = "treat", outcome = "visit",
+                          predictors = colnames(train[, 1:9]), nb.group = 5)
R>features
 [1] "treat"                        "recency"
```

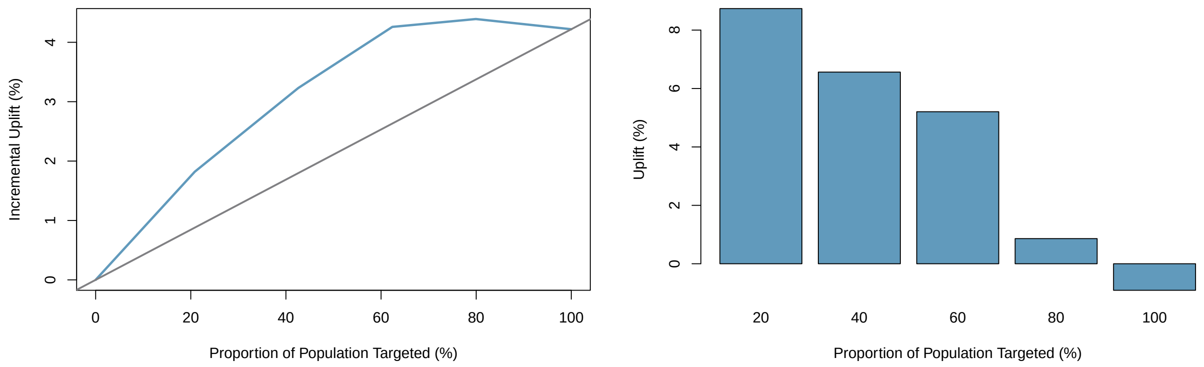

Figure 7: Performance of the best interaction model with automatic feature selection on a validation set. The Qini coefficient is $q = 0.99$.

```
 [3] "history"                      "mens"
 [5] "womens"                       "zip_code_Rural"
 [7] "zip_code_Surburban"           "newbie"
 [9] "channel_Multichannel"         "channel_Phone"
[11] "treat:recency"                "treat:mens"
[13] "treat:womens"                 "treat:zip_code_Rural"
[15] "treat:zip_code_Surburban"     "treat:newbie"
[17] "treat:channel_Multichannel" "treat:channel_Phone"
```

The function drops the interaction between `history` and `treat`. Next, we use the `features` vector in the `InterUplift()` function in order to fit an interaction model.

```
R># fitting the interaction estimator with selected features only
R>baseline.lasso <- InterUplift(train, "treat", "visit",
+                               predictors = features, input = "best")
R># predict the uplift on the validation set
R>baseline.lasso.valid <- InterPredict(valid, "treat", "visit",
+                               model = baseline.lasso, nb.group = 5)[[1]]

R># evaluate the model performance
R>baseline.lasso.perf <- QiniTable(baseline.lasso.valid, "treat", "visit",
+                               "uplift_prediction", nb.group = 5)
R>QiniCurve(baseline.lasso.perf, title = "")
R>QiniBarPlot(baseline.lasso.perf, title = "")
R>QiniArea(baseline.lasso.perf)
[1] 0.9532956
```

We can see that the baseline model shows an increase in the Qini coefficient from 0.72 to 0.95 when using the automatic feature selection method.

Next, we repeat this procedure by changing the versions (continuous or categorical) of the `recency` and `history` variables. The R Package **tools4uplift** makes it easy and fast to implement different models with feature selection, with both continuous and categorical variables. Table 2 displays the Qini coefficients associated with each of these models. The interaction model that works with the univariate quantizations of the variables `recency` and `history` yields the highest Qini coefficient, as well as a good ordering in terms of uplift of the population to target, i.e., from the highest to the lowest observed uplift. This is seen in Figure 7.

Function	Description
`BestFeatures()`	Feature selection for the interaction estimator
`BinUplift()`	Univariate quantization
`BinUpliftEnhanced()`	Univariate quantization - augmented data
`DualPredict()`	Predictions from a two-model estimator
`DualUplift()`	Two-model estimator
`InterPredict()`	Predictions from an interaction estimator
`InterUplift()`	Interaction estimator
`LassoPath()`	LASSO path for penalized logistic regression
`QiniArea()`	Qini coefficient
`QiniBarPlot()`	Uplift barplot
`QiniCurve()`	Qini curve
`QiniTable()`	Performance of an uplift estimator
`SplitUplift()`	Split data with respect to uplift distribution
`SquareUplift()`	Bivariate quantization

Table 3: Summary of the functions available in the R Package **tools4uplift**

Model	Feature Selection	Qini coefficient
Baseline	No	0.72
No quantization	Yes	0.95
Categorical `history` only	Yes	0.93
Categorical `recency` only	Yes	0.94
Univariate categorical `recency` and `history`	Yes	**0.99**
Bivariate categorical `recency` and `history`	Yes	0.97

Table 2: Comparison of model performances on a validation set. The non linearity introduced by the quantization of variables `recency` and `history` helps the uplift model to better segment the customers of the marketing campaign.

4 Summary

We present the methodology associated with the new R Package **tools4uplift** together with an application to a real world marketing campaign dataset, as an illustration of how the package could be used to analyse uplift data. The functions presented in this work are summarized in Table 3; their dependencies are shown in Figure 8. The purpose of **tools4uplift** is to give practitioners the necessary tools to get some insight about the uplift signal in the context of a randomized experiment. This work deals with four crucial steps in statistical modeling: i) quantization, ii) visualization, iii) feature selection, and iv) model validation. All the available functions in the package are thoroughly described and accompanied by a motivating example. The use of **tools4uplift** will enable practitioners to save time and effort when analyzing their uplift data.

Computational details

The results in this paper were obtained using R 3.2.3 with the Packages **tools4uplift** and **dummies**. R itself and all packages used are available from CRAN at `http://CRAN.R-project.org/`.

In order to analyze our algorithms in terms of runtime, two simple experiments were performed. The experiments were run on a desktop PC with Intel Core i7-7700 CPU @ 3.60GHz with 16 GB of RAM and Windows 10 operating system. For the univariate quantization algorithm of Section 2.4, Figure 9 shows the system runtime as a function of the number of split points to test at each node. For the bivariate quantization algorithm, the system runtime as a function of the number of rectangles is shown in Figure 10. In both cases, we see that the algorithms are computationally efficient.

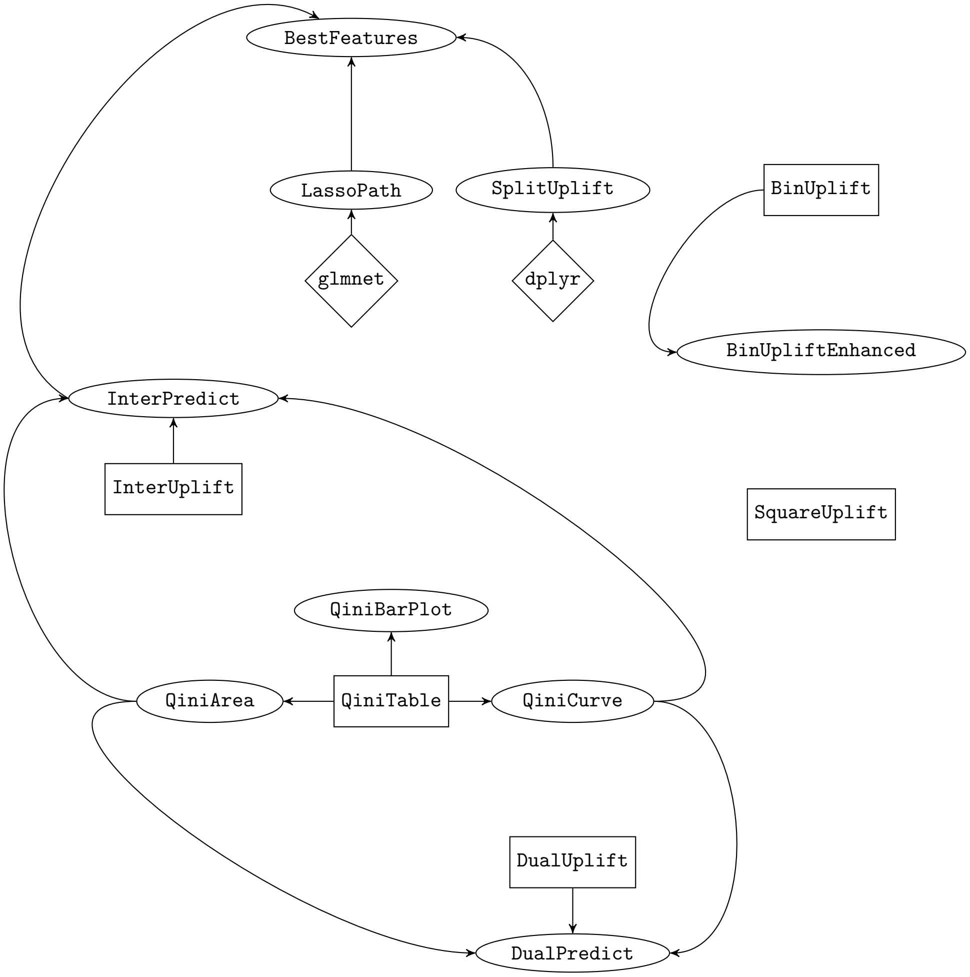

Figure 8: Diagram of function dependencies. Ellipses denote functions developed in the **tools4uplift** that are dependent on other functions. $A \longrightarrow B$ means that B depends on A. Rectangles denote independent functions and diamonds denote pre-existing R libraries.

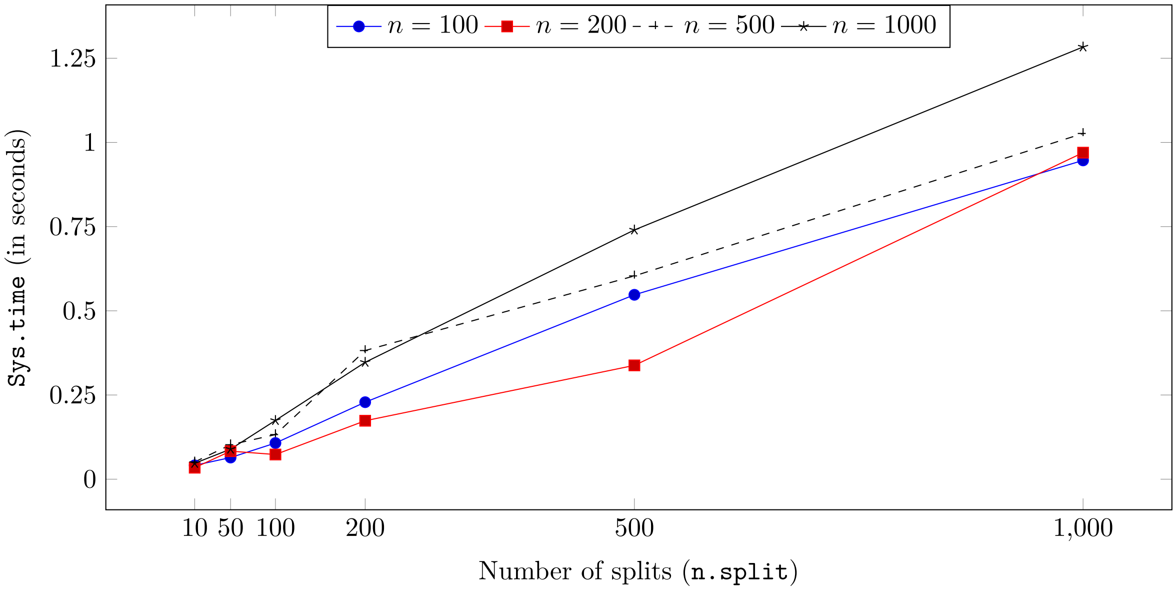

Figure 9: Runtimes (in seconds) of Algorithm 1 of the `BinUplift()` function (see Section 2.4) for univariate quantization of a continuous variable, as a function of the number of splits to test at each node of the decision tree. The different colored curves represent different sample sizes.

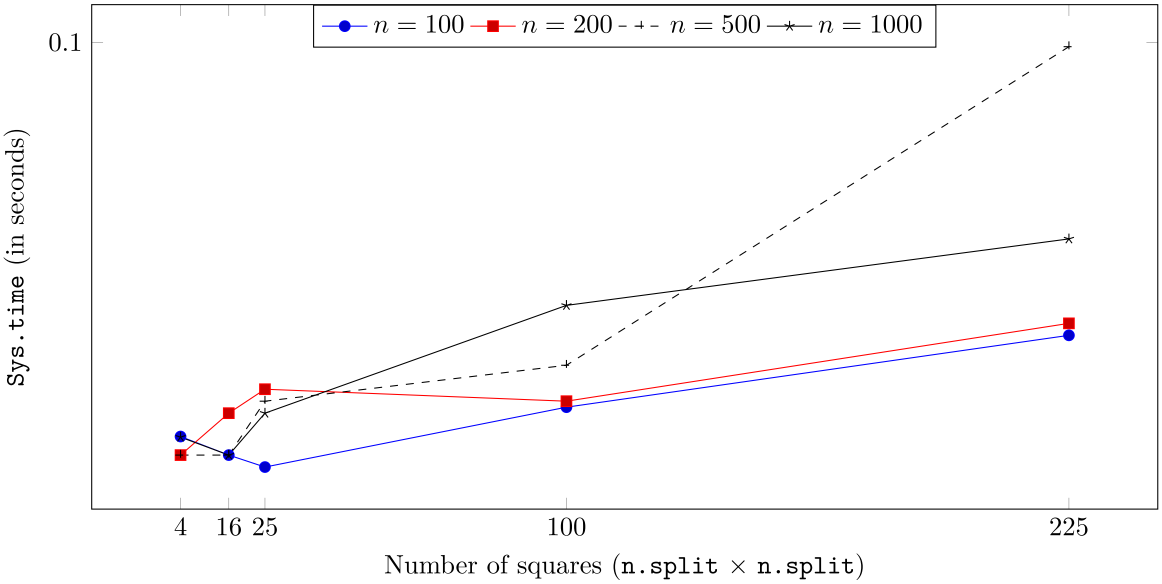

Figure 10: Runtimes (in seconds) of Algorithm 2 of the `SquareUplift()` function (see Section 2.4) for bivariate quantization of two continuous variables, as a function of the number of rectangles to use in order to estimate the uplift. The different colored curves represent different sample sizes. The maximum observed runtime in our experiment was 0.0993 seconds.

Acknowledgments

The authors would like to thank Marc-Olivier Billette and Marie-Christine Doucas. Mouloud Belbahri was supported by the MITACS acceleration program in the context of a research internship (http://www.mitacs.ca/en/programs/accelerate). Alejandro Murua was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) through grant number 327689-06. Vahid Partovi Nia was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) discovery grant 418034-2012.

References

Breiman, L., J. Friedman, C. J. Stone, and R. A. Olshen
1984. *Classification and regression trees.* CRC press.

Efron, B., T. Hastie, I. Johnstone, R. Tibshirani, et al.
2004. Least angle regression. *The Annals of Statistics*, 32(2):407–499.

Garcia, S., J. Luengo, J. A. Sáez, V. Lopez, and F. Herrera
2013. A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):734–750.

Gini, C.
1997. Concentration and dependency ratios. *Rivista di politica economica*, 87:769–792.

Guelman, L.
2014. uplift: Uplift modeling. *R package version 0.3*, 5.

Guelman, L. et al.
2015. *Optimal personalized treatment learning models with insurance applications.* PhD thesis, Universitat de Barcelona.

Hansotia, B. J. and B. Rukstales
2001. Direct marketing for multichannel retailers: Issues, challenges and solutions. *Journal of Database Marketing and Customer Strategy Management*, 9(3):259–266.

Hanssens, D. M., L. J. Parsons, and R. L. Schultz
2003. *Market response models: Econometric and time series analysis*, volume 12. Springer Science & Business Media.

Hastie, T., J. Taylor, R. Tibshirani, G. Walther, et al.
2007. Forward stagewise regression and the monotone lasso. *Electronic Journal of Statistics*, 1:1–29.

Hillstrom, K.
2008. The minethatdata e-mail analytics and data mining challenge. Data retrieved from https://blog.minethatdata.com/2008/03/minethatdata-e-mail-analytics-and-data.html.

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

Jaskowski, M. and S. Jaroszewicz
2012. Uplift modeling for clinical trial data. In *ICML Workshop on Clinical Data Analysis.*

Kane, K., V. S. Lo, and J. Zheng
2014. Mining for the truly responsive customers and prospects using true-lift modeling: Comparison of new and existing methods. *Journal of Marketing Analytics*, 2(4):218–238.

Kuusisto, F., V. S. Costa, H. Nassif, E. Burnside, D. Page, and J. Shavlik
2014. Support vector machines for differential prediction. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Pp. 50–65. Springer.

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

Lorenz, M. O.
1905. Methods of measuring the concentration of wealth. *Publications of the American Statistical Association*, 9(70):209–219.

Montgomery, D. C., E. A. Peck, and G. G. Vining
2012. *Introduction to linear regression analysis*, volume 821. John Wiley & Sons.

Neyman, J.
1923. On the application of probability theory to agricultural experiments. *Annals of Agricultural Sciences.*

Radcliffe, N.
2007. Using control groups to target on predicted lift: Building and assessing uplift models. *Direct Market J Direct Market Assoc Anal Council*, 1:14–21.

Radcliffe, N. J. and P. D. Surry
1999. Differential response analysis: Modeling true response by isolating the effect of a single action. *Credit Scoring and Credit Control VI. Edinburgh, Scotland.*

Radcliffe, N. J. and P. D. Surry
2011. Real-world uplift modelling with significance-based uplift trees. *White Paper TR-2011-1, Stochastic Solutions.*

Rubin, D. B.
1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688.

Rzepakowski, P. and S. Jaroszewicz
2010. Decision trees for uplift modeling. In *2010 IEEE International Conference on Data Mining*, Pp. 441–450. IEEE.

Sołtys, M., S. Jaroszewicz, and P. Rzepakowski
2015. Ensemble methods for uplift modeling. *Data Mining and Knowledge Discovery*, 29(6):1531–1559.

Tibshirani, R.
1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, Pp. 267–288.

Zaniewicz, Ł. and S. Jaroszewicz
2013. Support vector machines for uplift modeling. In *2013 IEEE 13th International Conference on Data Mining Workshops*, Pp. 131–138. IEEE.

Zaniewicz, Ł. and S. Jaroszewicz
2017. l_p-support vector machines for uplift modeling. *Knowledge and Information Systems*, 53(1):269–296.

Zhao, Y., X. Fang, and D. Simchi-Levi
2017. Uplift modeling with multiple treatments and general response types. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, Pp. 588–596. SIAM.