Master Thesis

Profit maximization for direct marketing campaigns

An application of knowledge discovery for decision making

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Statistics

Florian Hochstrasser

14.06.2019

Supervisor: Jacques Zuber

Abstract

It is known that....

This work addressed the problem from the perspective of...

It could be shown that...

Henceforth, it should be considered that...

Contents

1 Introduction

Customer segmentation techniques are used in marketing to identify certain groups of customers in order to produce offers tailored to these groups. The ultimate goal is to maximize profit, which is achieved also through customer retention. In the case of direct marketing, especially when unit costs (the cost associated with addressing a customer) are significant, employing some customer segmentation technique is highly beneficial in terms of profit.

Historically, the RFM (Recency, Frequency, Monetization) model has been employed with success in designing direct marketing campaigns Kohavi and Parekh (2004). While definitions vary, generally recency refers to when the last purchase was made. Frequency denotes number of purchases in a certain time period. Monetization can represent the amount of the last purchase, cumulative spending or the average amount spent per purchase.

The RFM model proposed by Hughes (1996) is often used: Customers are binned into 5 segments for each of the RFM features individually and labeled ordinal, resulting in 125 cells that can then be used to identify customers most likely to respond. The best customers have a high score for each of the 3 features. The drawback of this approach is that generally, marketing efforts go towards the best customer segment.

Over time, different approaches, such as Chi-squared automatic interaction detection and logistic regression were proposed. While in some situations, these alternatives outperformed RFA (see McCarty and Hastak (2007)), RFA remained popular because of it's intuitive interpretation.

In this work, a radically data-driven approach was chosen. General machine learning algorithms were employed to predict potential donors and the net profit generated instead of building on previously developed, specialized models.

1.1 Task Background

A U.S. American veterans organization regularly conducts direct marketing campaigns, asking their members for donations (called gifts in the documentation) for the treatment of veterans with spinal injuries. The goal for the organization is to maximize net profit from their campaigns.

Only a small proportion of the members donate in reply to a campaign, while each letter sent out has a unit cost of 0.68 $US. In order to maximize profit, it is therefore desirable to only mail members who are likely to donate.

The members are grouped, among other criteria, by the recency of their last gift. Of these groups, the so-called *lapsed* donors are of particular interest. These are members who made their last gift to the organization 13 to 24 months prior to a given campaign. This group is important for two reasons: Firstly, the probability of a member donating decreases with the time the member has not donated.

Enticing these lapsed donors to give again therefore maintains an active member base. Secondly, the organization has found that there is a negative correlation between the dollar amount donated and the probability to respond to a campaign. This means it is important to include the most unlikely donors in future mailings because if they donate, the net revenue is particularly large. If these unlikely donors would be suppressed from future campaigns, the gains from additional *small dollar* lapsed donors would not offset the losses from the potential *high dollar* donors.

The data at hand was distributed for the purpose of the KDD-CUP of the year 1998[1]. The cup was until recently held yearly under the aegis of the special interest group on Knowledge Discovery and Data Mining (SIGKDD), which itself is part of the Association for Computing Machinery[2](ACM).

1.2 Goal

The ultimate goal is to beat the winner of the original cup in terms of the predicted net profit for the promotion. For this, a complete data analysis including data preprocessing, model evaluation and -selection and final prediction was to be performed. A requirement set by the supervisor of this thesis was that the solution be demonstrated using Python as a programming environment.

Furthermore, the thesis should support future work on the data set by providing a solid basis especially on the preprocessing of the data.

1.3 Conventions and Notes

A member of the organization will be referred to as an *example*. Each example is described by a set of *features* (explanatory variables) and has two *targets* (dependent variables) associated.

Software packages are denoted as `package` with specific modules contained in packages written as `package.module.Class`. Where available, software used is cited with the article in which it was published. Less established packages are cited by giving their public source code repositories.

All self-written code, including the reproducible analysis process, is published online at https://github.com/datarian/master-thesis-msc-statistics. These resources, especially the folder `notebooks`, form an integral part of this thesis.

[1] For an archive of past cups, see SIGKDD - KDD Cup
[2] https://acm.org

2 Data

The data set, which is freely available online[1], contains data on all members of the organization with a *lapsed* donation status (last donation 13 – 24 months ago) relative to the promotion sent out in June 1997.

The data is provided split in two sets, of which one is intended for learning, the other for validation. The features are identical between the two except for the target features that have been separated from the validation set.

In this section, the *learning* data set will be characterized.

2.1 General Structure

The input data with $n = 95412$ rows and $m = 479$ columns is structured as follows: $\mathbf{D} = \{\{\mathbf{x}_i, \mathbf{y}_i\}\}, i = 1...n, \mathbf{x} \in \mathbb{R}^{m-3}, \mathbf{y} \in \mathbb{R}^2$.

Each i represents one example. We have $m-3 = 476$ explanatory features, two targets and one unique identifier for each example.

The features are grouped into four blocks of information:

- Member database with personal particulars, interests and organization-internal information on examples: 79 features
- Characteristics of example's neighborhood from the US census 1990: 286 features
- Promotion history: History of past promotions sent to an example and the examples responses, per promotion and summary statistics: 54 features
- Giving history of an example's past donations, per promotion and summary statistics: 57 features

2.2 Exploratory Data Analysis

Below, the data is characterized with some key insights.

The detailed analysis can be studied online in the corresponding Jupyter notebook `3_EDA.ipynb`[2].

[1] See https://archive.ics.uci.edu/ml/datasets/KDD+Cup+1998+Data

[2] see github.com/datarian/master-thesis-code/notebooks/3_EDA.ipynb

2.2.1 Data Types

An analysis of the data set dictionary (Section .2) reveals the following data types.

- Index: CONTROLN, unique record identifier
- Dates: 48 features in yymm format.
- Binary: 30 features
- Categorical: 90 features
- Numeric: 286

Data was imported through `pandas.read_csv()`. The data types present after import are shown in Table 2.1. It is evident that several transformations are necessary to encode the features in their correct types.

Table 2.1: Data types after import of raw csv data

	Data content	Number of features
Integer	Discrete features, no missing values	297
Float	Continuous features and discrete features with missing values	48
Categorical	Nominal and ordinal features	24
Object	Features with alphanumeric values	109
Total		478

2.2.2 Targets

Of the two targets, one is binary (TARGET_B), the other continuous (TARGET_D). The former indicates whether an example has donated in response to the current promotion. The latter represents the dollar amount donated in response to the current promotion.

As can be seen in Figure 2.1, the binary target is imbalanced. Of all examples, only 5 % have donated. Extra care will have to be taken during model training to obtain a model with a low generalization error.

The distribution of the continuous target, including all examples with a donation amount > 0.0 $, is shown in Figure 2.2. Evidently, most donations are smaller than 25 $, the 50-percentile lying at 13 $ and the mean at 15.62 $. There are a few outliers for donations above 100 $, making the distribution right-skewed. Being monetary amounts, the observed values are discrete rather than continuous.

2.2.3 Skewness

Most of the numerical features are skewed. Due to the high dimensionality, individual assessment of the features through boxplots or histograms was not feasible. Instead, skewness was measured with `pandas.skew()`, which uses the Fisher-Pearson standardized moment coefficient $G_1 = \frac{\sqrt{n(n-1)}}{n-2}\frac{1}{n}\frac{\sum_{i=1}^{n}(x_i-\bar{x})^3}{s^3}$ and plotted together with the $\alpha = 5\%$ confidence bound for a normal distribution (see Figure 2.3). Evidently, no feature was found to be strictly normally distributed.

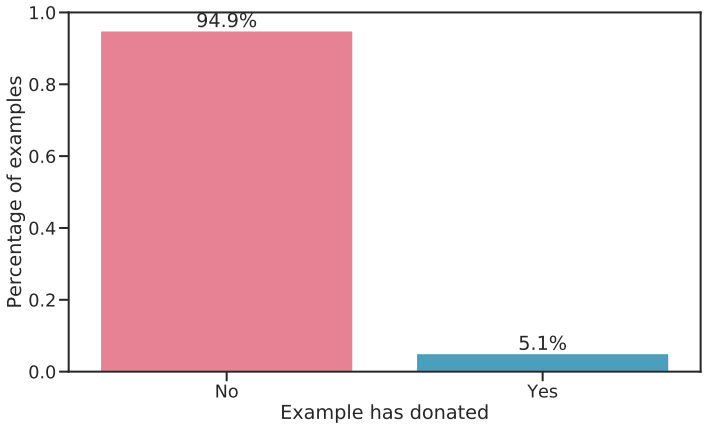

Figure 2.1: Ratio of examples who donated out of all examples sent a promotion. Only 5 % donated.

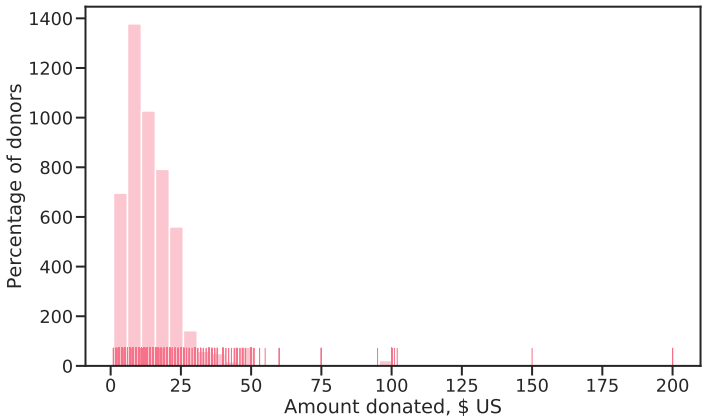

Figure 2.2: Distribution of TARGET_D, the donation amount in $ US (only amounts > 0.0 $ are shown). Most donations are below 25 $, peaks are visible at 50, 75 and 100 $, while the maximum donation amount is 200 $.

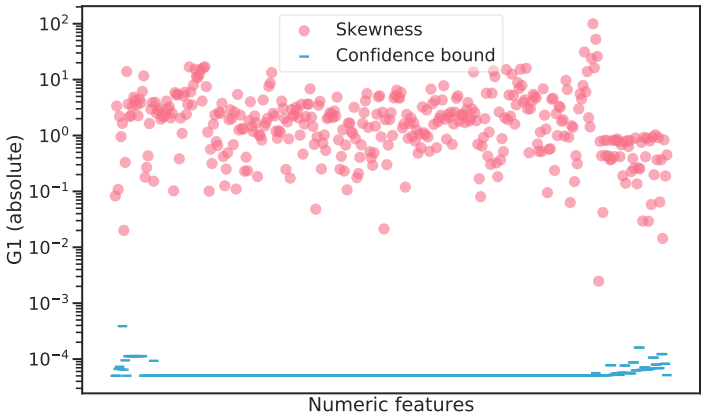

Figure 2.3: Fisher-Pearson standardized moment coefficient (G1) for all numeric features contained in the dataset. The confidence bound indicates the $\alpha = 5\%$ bound for the skewness of a normal distribution for any given feature. Absolute values were chosen to display the results on a log scale.

Looking at the 6 least skewed features (Figure 2.4), we find distributions that resemble normal or uniform, or binary features that are balanced.

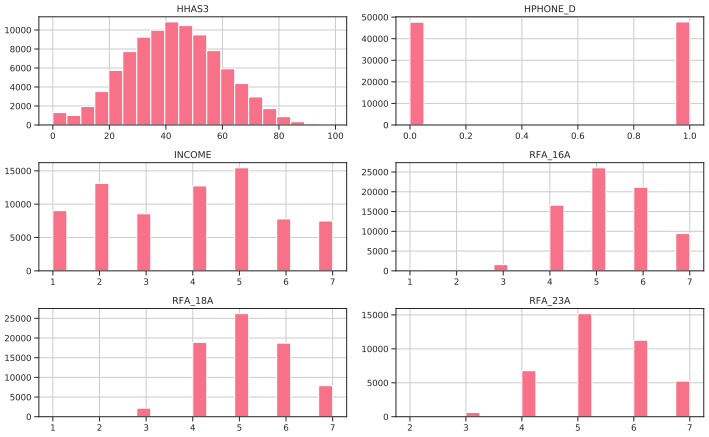

Figure 2.4: The 9 least skewed features. Skewness metric: adjusted Fisher-Pearson standardized moment coefficient.

The 6 most skewed features (Figure 2.5) show heavily right-skewed distributions which are the result of outliers.

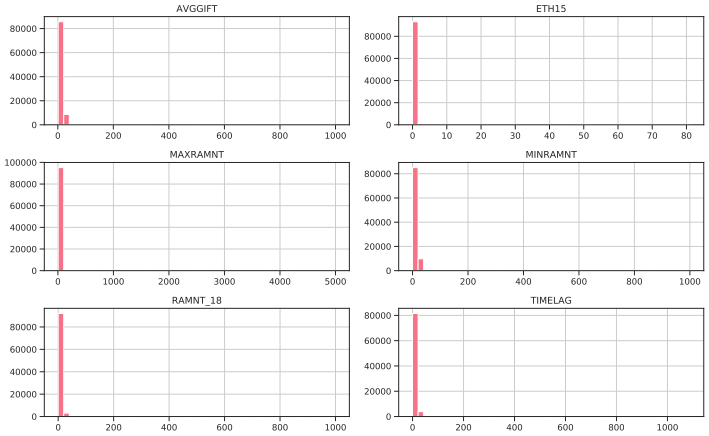

Figure 2.5: The 9 most skewed features. Skewness metric: adjusted Fisher-Pearson standardized moment coefficient.

2.2.4 Correlations

As for the assessment of skewness, the high dimensionality makes it hard to assess correlations in the data. A heatmap (see 2.6) was nevertheless produced to gain a high-level insight on correlations present in the data. From left to right, three regions can be distinguished: First, there are member database features, followed by a large center region comprised of the U.S. census features, and rightmost, there are promotion and giving history features. Between these blocks, only few features are correlated. Within each block however, we can see some quite strongly correlated data.

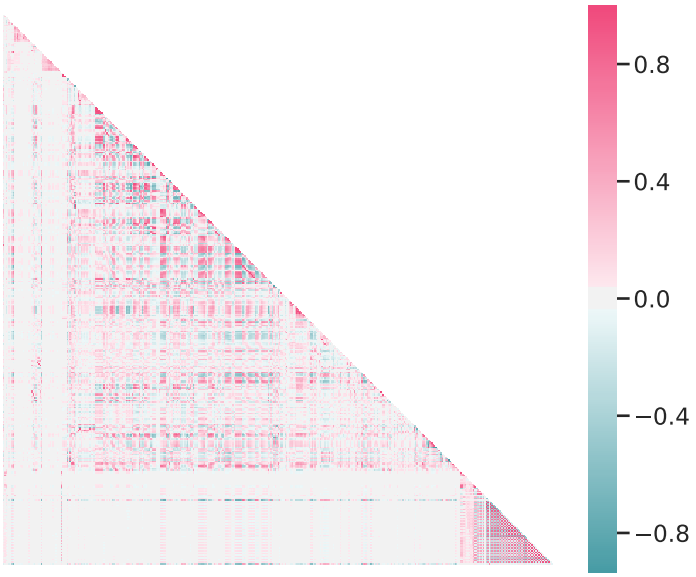

Figure 2.6: A heatmap showing correlations between all features in the data. Green means negative correlation, pink means positive correlation. Perfect correlation occurs at -1.0 and 1.0. Not all features are labelled due to display problems.

2.2.5 Donation Patterns

The data set documentation states that donation amounts are positively correlated with the time since the last donation. This means that the longer an example goes without donating, the higher the donation amount if it can be enticed into donating again. Figure (2.7) confirms this assumption. We see that starting from 15 months, the number of donations above 50 $ increases.

There is another insight gained when considering the number of donations an example has made, indicated by the point size, and donation amount: Frequent donors give relatively small sums, while the largest donations come from examples who rarely donate.

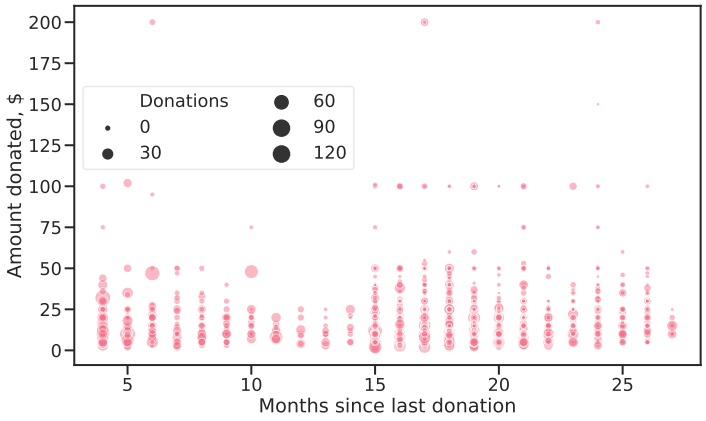

Figure 2.7: Donation amount for the current promotion against months since last donation. The dot size indicates the number of times an example has donated.

When looking at the recency-frequency-amount (RFA) features for the current promotion (which are typically used to model customer response in direct marketing), we can support the insights above. Since the data set contains only lapsed donors, the recency feature is constant and not of interest. Regarding the frequency of donations (number of donations 13 - 24 months prior to the promotion), shown in Figure 2.8, we see a clear trend. With increasing donation frequency, donation amounts decrease.

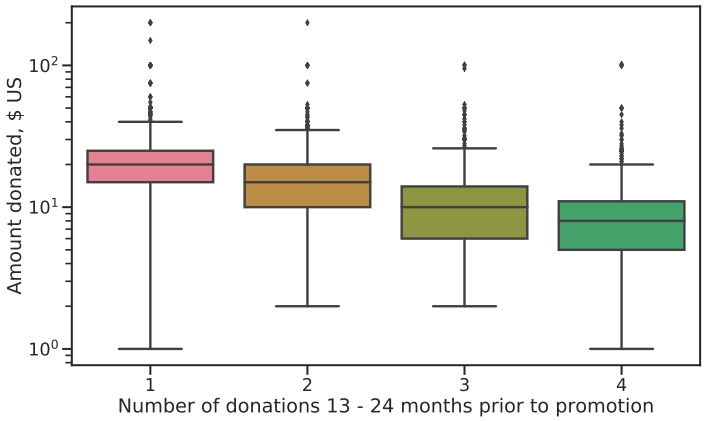

Figure 2.8: Frequency of donations in the 13-24 months prior to current promotion against amount donated. Frequent donors give smaller amounts.

From the amount classes of the last donation, we can see that most examples follow their habits and usually donate the same amounts across promotions (Figure 2.9).

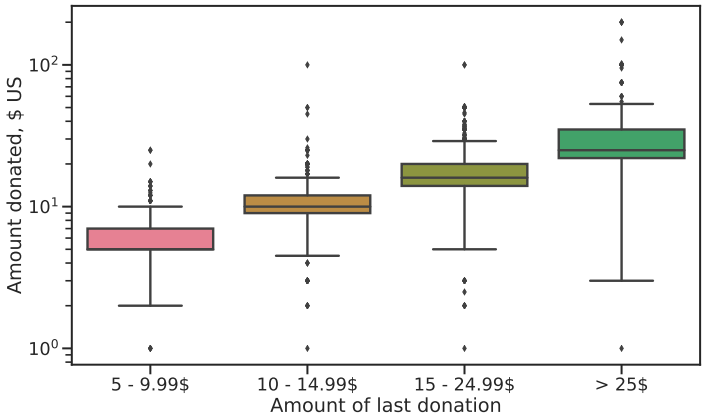

Figure 2.9: Amount of last donation prior to promotion against donation amount. Donors largely give equal donations across promotions.

Figure 2.10 shows the geographical distribution of donations. The large urban centers like San Francisco, Los Angeles, Miami, Chicago and Detroit are clearly visible. To a lesser extent, cities like Houston, Dallas, Minneapolis, Atlanta, Tampa, Seattle and Phoenix can be made out. Examples living there give small amounts. Big donors (large total donations with a high average) can be made out in rural areas in the Midwest and Texas. Interestingly, only very few donations come from the north-eastern states.

We can also see that examples living in rural areas tend to donate larger sums when looking at Figure 2.11.

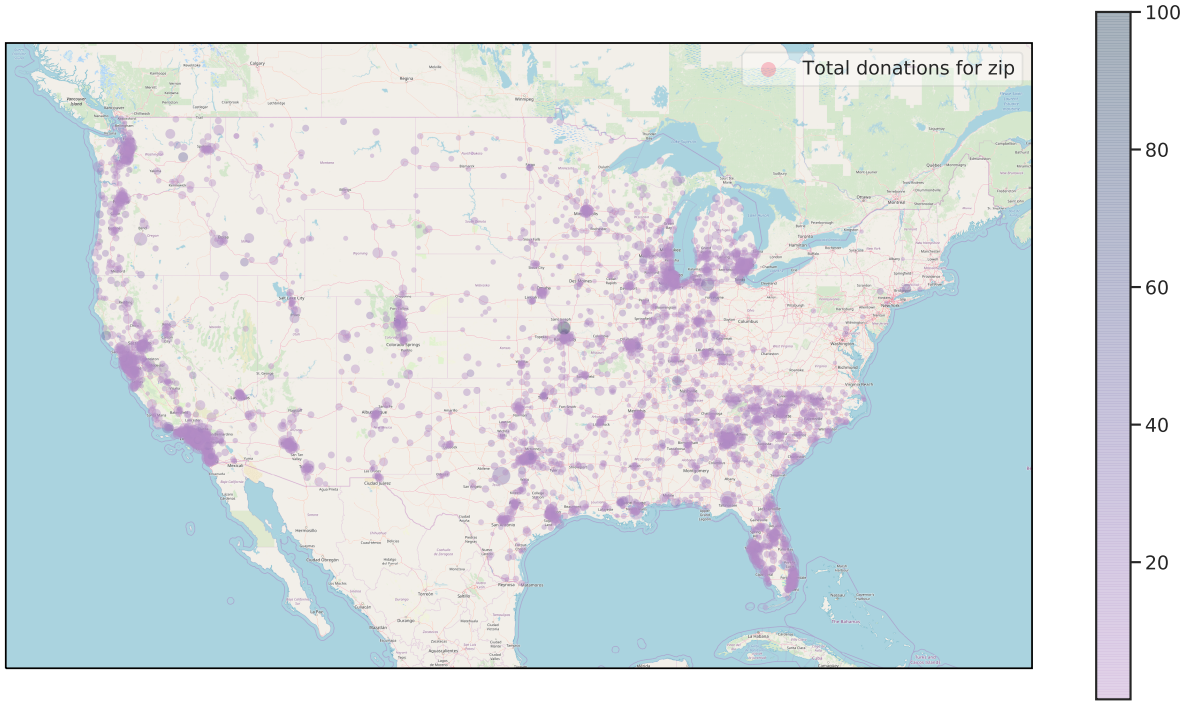

Figure 2.10: Geographical distribution of donations by zip code. Point size indicates total donations for a zip code while the hue shows average donation amount.

Shown are the living environments in progressively more rural settings against the donation amount in the current promotion.

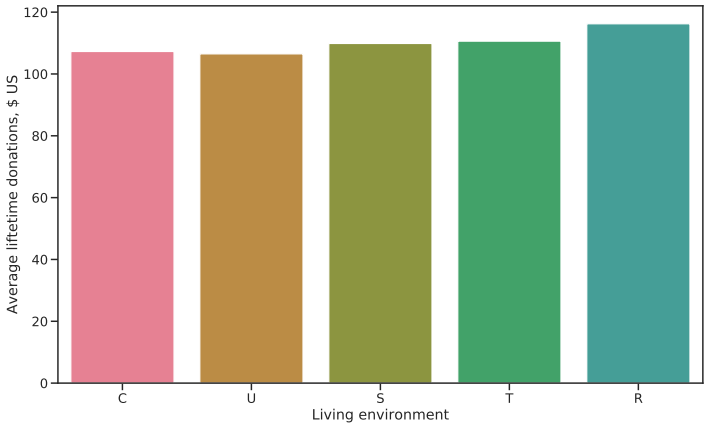

Figure 2.11: Average cumulative donation amount per capita by living environment (C = city, U = urban, S = suburban, T = town, R = rural). The more rural, the higher the average donations.

3 Experimental Setup and Methods

The general work flow with which the problem was solved is shown in Figure 3.1. Several iterations through the individual steps were necessary. The Jupyter notebooks[1] corresponding to each step in the process are meant to be run in the order indicated by their numbering. This ensures that persisted data and models are available when required.

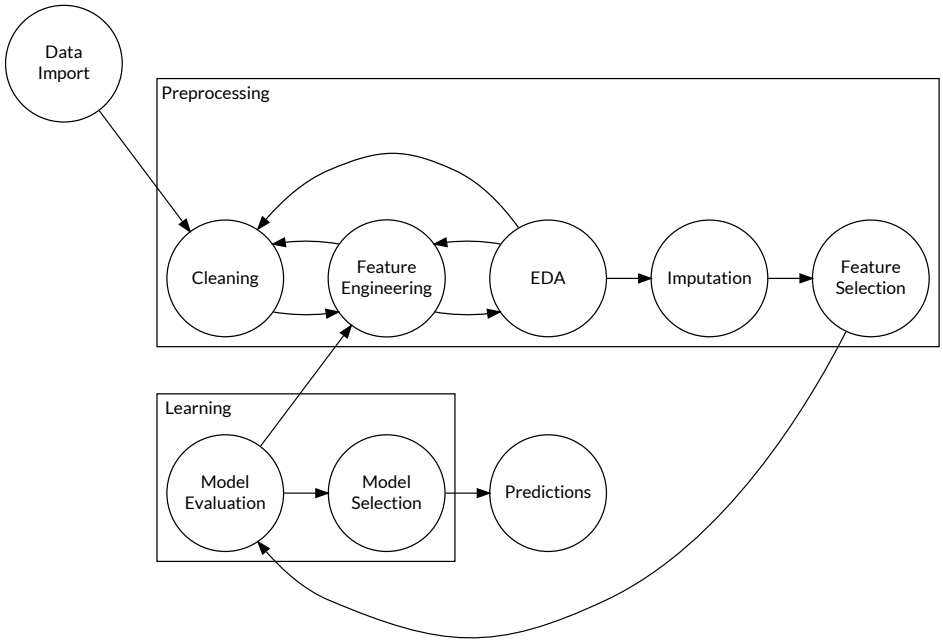

Figure 3.1: General work flow during preprocessing and model learning. Several iterations between atomic steps were necessary.

3.1 Tools Used

The problem itself was solved using the python language, making use of established packages: `numpy`: Oliphant (2006), `scipy`: McKinney (2010), `pandas`: McKinney (2011), `scikit-learn`: Pedregosa et al. (2011), `matplotlib`: Hunter (2007), `seaborn`: Waskom et al. (2014).

[1] see github.com/datarian/master-thesis-code/notebooks

Except for the development of a helper packge, most programming was performed in interactive Jupyter notebooks (see Kluyver et al. (2016)).

The report was written in `rmarkdown`: Allaire et al. (2016) using `knitr`: Xie (2015) and `bookdown`: Xie (2016) to render the document into several output formats.

All work was tracked in version control.

3.2 Data Handling

The complete data is distributed pre-split into a learning and validation data set.

The learning data set was used to establish the complete analysis pipeline, i.e. determining necessary preprocessing, exploratory data analysis, model evaluation and -selection and establishing the prediction method. Only then was the test data set used to make the final prediction.

In accordance with recommendations in Friedman, Hastie, and Tibshirani (2001), the learning data set was split 80/20 into training and validation sets (see Figure 3.2). The training set was used to train different models while the validation set served to tune hyper-parameters. The split was performed using a stratified sampling algorithm to preserve the target class frequencies.

The validation data set (named *test* data set hereafter) was treated as unseen data. Once the analysis pipeline was established, the validation data was used to make the final prediction, subjecting the data to the pipeline trained on the learning data set.

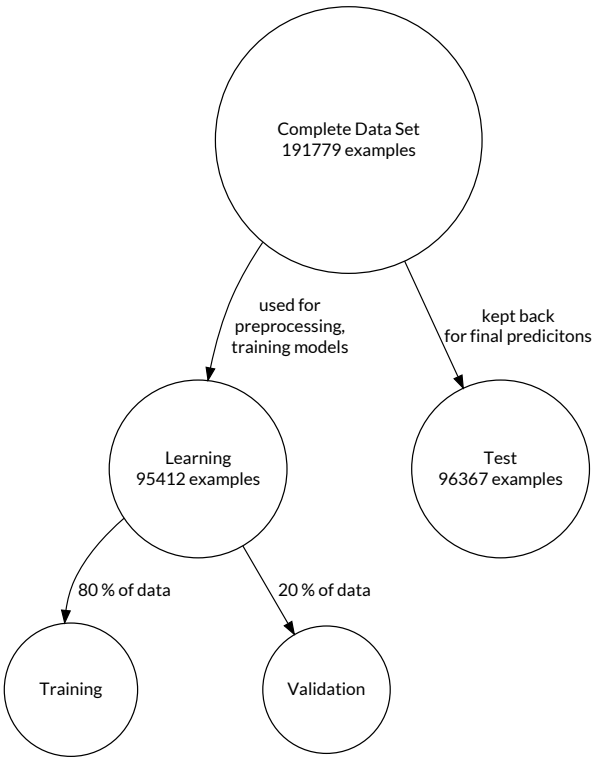

Figure 3.2: Data set use for training and predictions.

3.3 Data Preprocessing

The necessary preprocessing was guided by practical necessity (input errors, inconsistent categories), the requirements of the algorithms that were examined (Section @eval-and-select) and the requirements set out in the cup documentation:

- Only numeric features (required by some algorithms)
- Imputation of missing values (required by cup documentation)
- Removal of *sparse* features (required by cup documentation)

The transformations were established interactively in Jupyter notebooks. Once finalized, transformations were implemented in the python package `kdd98`[2].

3.3.1 Cleaning

The transformations applied can be studied in the Jupyter notebook *1_Preprocessing.ipynb*[3].

The cleaning stage of preprocessing encompassed the following transformations:

- Removing *noise*: Input errors, inconsistent encoding of binary / categorical features
- Dropping constant and sparse (i.e. those where only few examples have a value set) features
- Imputation of values missing at random (MAR)

MAR values in the sense of Rubin (1976) are missing conditionally on other features in the data. For example, there are three related features from the promotion and giving history: *ADATE*, the date of mailing a promotion, *RDATE*, the date of receiving a donation in response to the promotion and *RAMOUNT*, the amount received. For missing *RAMOUNT* values, we can check if *RDATE* is non-missing. If *RDATE* is missing, then the example most likely has not donated and we can set *RAMOUNT* to zero. If, on the other hand, both date features have a value, *RAMOUNT* is truly missing.

3.3.2 Feature Engineering

During feature engineering, all non-numeric (i.e. categorical) features were encoded into numeric values. Also, several features were transformed to better usable representations. Care was taken to keep the dimensionality of the data set as low as possible. The result of this transformation step was an all-numeric data set usable for downstream learning. The transformations applied in feature engineering are described in detail in the Jupyter notebook *2_Feature_Engineering.ipynb*[4].

For ordinal features, manual mappings from alphanumeric levels to integer numbers were specified.

For nominal features, two encoding techniques were employed, depending on the number of levels:

- One-hot encoding for ≤ 10 levels: For each level of a categorical feature, a new feature is created. An additional feature may be added to indicate missing values. Exactly one of these new features is set to 1, indicating the original level.

[2] Available from github.com/datarian/master-thesis-code/kdd98
[3] see github.com/datarian/master-thesis-code/notebooks/1_Preprocessing.ipynb
[4] see github.com/datarian/master-thesis-code/notebooks/2_Feature_Engineering.ipynb

- Binary encoding, > 10 levels: The levels of the categorical are first transformed ordinally (i.e. to a sequence of integer numbers). Then, these numbers are taken to the base of 2. (A 5, for example, becomes `101`). According to the number of levels, new features for the binary digits are created. As an example: To represent 60 levels, 6 features are required ($2^6 = 64$).

3.3.3 Imputation

Three different approaches were evaluated. The details are shown in Jupyter notebook *4_Imputation.ipynb*[5].

3.3.3.1 K-Nearest Neighbors

The kNN algorithm by Troyanskaya et al. (2001) works approximately as follows:

1. Construct the distance matrix D with distances between examples
2. Order all features with missing values descending by number of missing values
3. Starting with the feature with most missing, use the k nearest neighbors of each example with a missing value that have a value for the current feature to impute using either the mean or median.

The algorithm runs until all values are imputed.

While very attractive because of the intuitive approach and because it preserves data types in the features, the distance matrix is very memory-intensive for large data sets. Also, all features with more than 80 % missing values have to be removed from the data first.

3.3.3.2 Iterative imputation

Iterative imputation, implemented in package `fancyimpute`[6], works similar to the R-package `mice` (see van Buuren and Groothuis-Oudshoorn (2011)). Before imputation, all features have to be transformed to numerical data types. Categorical features were therefore encoded first, using a one-hot or binary encoding and preserving missing values.

1. Features are ordered by the fraction of missing values
2. Starting with the feature with most missing values, use the other features to build a model, using the current feature as the dependent variable and predict missing values.
3. Repeat step 2 until all features are complete
4. Repeat steps 2 – 3 n times, $n = 5$ was chosen

[5] see github.com/datarian/master-thesis-code/notebooks/4_Imputation.ipynb
[6] Available at: https://pypi.org/project/fancyimpute/, accessed on 30.06.2019

3.3.3.3 Median Imputation and Categorical Indicator

The 'sklearn.impute.SimpleImputer' was used on all numerical features. Since many features are skewed and have outliers, the median strategy was used. The missing values in each feature are imputed by the feature's median value.

As this implementation only supports numerical data types, categorical features were treated separately during feature engineering (Section 3.3.2): The one-hot or binary encoded categoricals had one more feature added, indicating missing values.

3.3.4 Feature Selection

One of the biggest caveats in machine learning is the infamous "Curse of Dimensionality" coined by Bellman (1966). The curse comes from the fact that with an increasing number of dimensions of the feature space, the number of possible combinations grows exponentially. In order to cover all possible combinations with several examples, a huge amount of examples would be required as a result. Hughes (1968) showed that for a fixed number of examples, model performance first increases with increasing number of dimensions but then decreases again. In the area of machine learning, high dimensionality frequently manifests in the form of overfitting, which leads to an unacceptably big generalization error Goodfellow, Bengio, and Courville (2016).

It is therefore beneficial to reduce the data set dimensionality while preserving as much relevant information as possible. A method to deal with the problem is called boruta, introduced by Kursa, Rudnicki, and others (2010). The algorithm was found to perform very well regarding selection of relevant features in Kursa and Rudnicki (2011). It works sequentially and removes features found to be less relevant at each iteration. By doing so, it solves the so-called all-relevant feature problem. The algorithm is actually a wrapper function around a random forest classifier. A random forest classifier is fast, can usually be run without parameters and returns an importance measure for each feature.

In short, the alogrithm works as follows:

1. The input matrix $\mathbf{X}$ of dimension $n \times p$ is extended with p so-called *shadow features*. The shadow features are permuted copies of the features in $\mathbf{X}$. They are therefore decorrelated with the target.
2. On the resulting matrix $\mathbf{X}^*$, a random forest classifier is trained and the Z-scores ($\frac{\bar{loss}}{sd}$) for each of the $2p$ features calculated.
3. The highest Z-score among the shadow features $MZSA$ is determined.
4. All original features are compared against $MZSA$ and those features with a higher score selected as important.
5. With the remaining features, a two-sided test for equality of the Z-scores with $MZSA$ is performed and all features with significantly lower score are deemed unimportant.
6. All shadow copies are removed, go to step 1.

The algorithm terminates when all attributes are marked as either important or not important or when the maximum number of iterations is reached.

For this thesis, a python implementation[7] was used. In effect, it is a port of the original R package by Kursa, Rudnicki, and others (2010) which conveniently implements `scikit-learn`'s API.

[7] see scikit-learn-contrib/boruta_py

3.4 Prediction

The desired quantity to predict is net profit. In order to predict this quantity, a two-step prediction procedure is applied, utilizing the binary target *TARGET_B* and the continuous target *TARGET_D*, respectively. For each step, a model is trained. One is a classifier, predicting $\hat{y}_b$, the probability of donating. The other is a regressor, predicting the donation amount $\hat{y}_d$. The classifier is trained on the complete learning data set, while the regressor is only trained on $\mathbf{X}_d = \{\mathbf{x_i} | y_{b,i} = 1, i = 1...n\}$. By using the non-random sample $\mathbf{X}_d$, bias is introduced, which has to be corrected. This approach resembles the two-stage Heckman procedure.

The quantity estimated is the *expected profit*. For unseen data (without information on the true response), it is calculated by:

$$E(Z) = \sum_{i=1}^{n} \hat{y}_{i,b} * \hat{y}_{i,d} * \alpha^* \tag{3.1}$$

where $\hat{\mathbf{y}}_b$ is the probability of donating, $\hat{\mathbf{y}}_d$ is the conditionally predicted donation amount of an estimator trained on the non-random sample $\mathbf{X}_d$ and $\alpha^* \in [0, 1]$ is a factor to correct for bias introduced due to the non-randomness of $\mathbf{X}_d$, learned beforehand.

The decision of whether to include example i in the promotion is governed by the following indicator function. Every example that has a predicted donation amount of more than the unit cost is included.

$$\mathbb{1}_{\hat{y}_{i,b} * \exp(\hat{y}_{i,dt}) * \alpha > \exp(u_t)}(\hat{y}_{i,dt}) \tag{3.2}$$

During model learning, the following profit optimization function was used to determine α^*, as well as making predictions for net profit with the learning and test data sets. The changes in the indicator function are due to technical reasons. Because linear models were used for predicting $\hat{y}_d$, the target was transformed to achieve better results. The exponential was introduced to deal with negative values resulting from the transformation.

$$\Pi_\alpha = \sum_{i=1}^{n} \mathbb{1}_{\hat{y}_{i,b} * \exp(\hat{y}_{i,dt}) * \alpha > \exp(u_t)}(\hat{y}_{i,dt}) * (y_{i,d} - u) \tag{3.3}$$

with $\mathbb{1}$ the indicator function, $\hat{y}_{i,dt}$ the predicted donation amount, Box-Cox transformed, $y_{i,d}$ the true donation amount for example i and u_t the unit cost in \$, transformed with the same parameter λ utilized to transform $\hat{\mathbf{y}}_d$.

3.4.1 Optimization of α^*

With equation (3.3), the estimated profit Π is calculated for a grid of α values, $\alpha \in [0, 1]$. The optimal value is then $\alpha^* = \underset{\alpha}{\operatorname{argmax}} f(\alpha)$ where f is a function that is fit to Π.

For f, a cubic spline s was used. α^* is then calculated as follows:

1. Fit $s(\Pi)$, the cubic spline on the estimated profits for the grid of α values

2. Derive $ds = \frac{\delta}{\delta\alpha}s$
3. Find the finite roots of ds, $\alpha_{\text{candidates}}$, representing candidates for α^*
4. Determine $\alpha^* = \underset{\alpha}{\text{argmax}} s(\alpha_{\text{candidates}})$

3.5 Model Evaluation and -Selection

During model evaluation, several algorithms were trained and their performance compared using a metric in order to select the best estimator. This was done independently for classifiers (predicting the binary target) and regressors (predicting the continuous target).

One of the powerful tools provided by `scikit-learn` are pipelines. They enable chaining transforming steps and estimators together. Pipelines were used to re-sample and scale data before model learning. Fitted pipelines can be persisted on disk and then be used later for predictions on other data.

3.5.1 Evaluation

Randomized grid search with 10-fold cross-validation (CV) was used for model evaluation. A python dictionary was used to store the best-performing pipeline per algorithm evaluated. The dictionary was persisted to disk and only updated when during a learning iteration, an algorithm produced a better score than the previous best score. This ensured that the best hyperparameter settings and sampling strategies were always retained for each of the algorithms evaluated.

3.5.1.1 Randomized Grid Search

In randomized grid search, distributions for hyperparameter values are specified instead of defining a fixed grid of values to search over (as in grid search cross-validation). The algorithm then runs a defined number of random combinations of the parameters (10 were used). Compared to the usual grid search, this can greatly speed up the learning process because good hyperparameter settings are identified with less iterations.

After one round of learning, the hyperparameter distributions are adjusted before the next iteration as follows: When the best value is found near the limits of the domain, the distribution is shifted in this direction. For values falling inside the domain of the distribution, the distribution is narrowed down towards the found value. This procedure is repeated until the hyperparameters converge.

3.5.1.2 Cross-Validation

CV splits the training data into several *folds* of equal size. The algorithm is trained as many times as there are folds, holding back one of the folds at each training step for validation using some specified performance metric and training with the rest of the data. This procedure enables quantification of the generalization error and the calculation of statistics that indicate the variance of the model.

3.5.1.3 Performance Metrics

For classification problems, the confusion matrix (see Figure 3.3) can be used to construct various performance metrics.

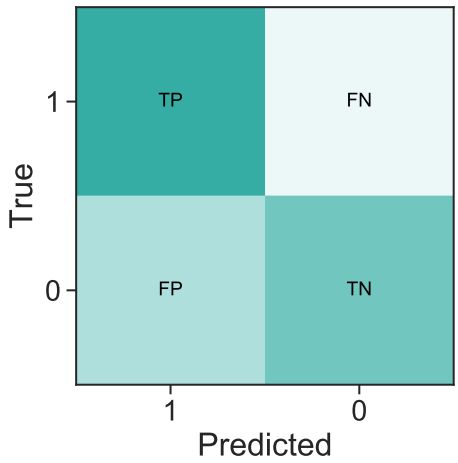

Figure 3.3: Definition of the confusion matrix for a two-class negative / positive (0 / 1) problem. If we predict "1" correctly, it is a *true positive* (TP), predicting "1" falsely is a *false positive* (FP). A *false negative* (FN) occurs when predicting "0" falsely and a *true negative* (TN) occurs when "0" was predicted correctly.

The definitions of some often-used metrics are given below[8]. The choice of metric depends on the goal of the prediction. Accuracy is often used, but in the case of imbalanced targets, as is the case here, it is not a desirable metric because the majority class dominates the metric.

$$\text{Recall / Sensitivity / True Positive Rate TPR} = \frac{TP}{TP+FN}$$
$$\text{Specificity / True Negative Rate TNR} = \frac{TN}{TN+FP}$$
$$\text{Precision / Positive Predictive Value PPV} = \frac{TP}{TP+FP}$$
$$\text{Negative Predictive Value NPV} = \frac{TN}{TN+FN}$$
$$\text{False Negative Rate FNR} = \frac{FN}{FN+TP}$$
$$\text{False Positive Rate FPR} = \frac{FP}{FP+TN}$$
$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$$
$$\text{F1 score} = \frac{2TP}{2TP+FP+FN}$$

[8] taken from https://en.wikipedia.org/wiki/Confusion_matrix, accessed on 28.05.2019

The goal for our model is to maximize net profit. To achieve this, we have to find a balance between predicting as many TP's as possible while keeping the number of FN's and FP's low. FN's are kept low by training a model that is good at predicting TP, at the cost of performing worse for TN. Likewise, a low FP means a high rate of TN at the cost of predicting TP.

The metrics that could be used beneficially are F1, recall and precision. One FP costs 0.68 $. Keeping in mind the distribution of profit (Section 2.2.2), one FN means loosing at least 0.32 $ of possible profit. The expected loss in profit for one FN is approximately 15 $, which means that with each TP, we can balance 22 FP's on average. Nevertheless, it is beneficial to keep FP as low as possible.

Therefore, *F1* was chosen as the performance metric for classification. It is the harmonic mean of recall and precision and presents a good compromise, especially for imbalanced data.

For regression, $R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$ was used. R^2 has the drawback of depending on the variance of the data used to fit the model and therefore is different for other data. It was however assumed that because learning and test data have the same generating function, R^2 can be used to select a regression model.

3.5.2 Dealing With Imbalanced Data

Several different approaches were explored:

- Random oversampling of the minority class
- Random undersampling of the majority class
- SMOTE (synthetic minority oversampling technique), variant borderline-1
- Explicitly specifying class weights
- Specifying sample weights using the true donation amount

The under-/oversampling algorithms in package `imblearn` by Lemaître, Nogueira, and Aridas (2017) were used.

3.5.3 Algorithms

A short introduction of each algorithm is given below. For each algorithm, the hyperparameters that were considered during learning are given. The choice of algorithms was made so as to cover a wide range of underlying concepts.

3.5.3.1 Random Forest

Random forest (RF), Breiman (2001), belongs to the family of so-called ensemble learners. RF can be used for classification and regression tasks. An ensemble of estimators cast their votes and the prediction is made by the majority vote. In the case of a random forest, the ensemble is made up of decision trees. RF's are insensitive towards scale differences in the individual features. The input data therefore does not have to be scaled before learning.

The algorithm learns classification and regression trees (CART, see Breiman et al. (1984)). For each tree in the forest, a random sample of the available features is drawn with replacement (bagging, or

bootstrap aggregating, see Breiman (1996)). By randomly selecting features for each tree, the variance of the ensemble estimator can be reduced. Furthermore, splits are made on a random subset of the features selected to grow the tree. These sources of randomness tend to increase the bias of the forest, yet the decrease in variance due to the averaging through majority vote outweighs the bias increase. Breiman (2001) shows that as the forest grows, the generalization error converges almost surely. This means that random forests are insensitive to overfitting.

An interesting aspect of RF is the out of bag (OOB) sample mechanism. It resembles CV by using all trees for prediction in which an example $z_i = (\mathbf{x_i}, y_i)$ did not appear during tree growth. It can be used for early stopping, terminating learning once OOB error stabilises.

Another important feature of RF is the assessment of *variable importance*. By summing the improvement for each split in every tree per feature, the importance for all features is calculated (see Friedman, Hastie, and Tibshirani (2001)).

The `RandomForestClassifier` and `RandomForestRegressor` included in `scikit-learn` were used for learning.

Hyperparameters

- `max_depth`, $\{1, 2, 3, ...\}$: depth of the trees, 2^n leafs maximum. Controls the interaction order of features.
- `min_samples_split`, $\{2, 3, 4, ...\}$: Minimum number of samples required for a split.
- `max_features`, $\{1, 2, ..., m\}$: Maximum number of the m features to consider when searching for a split. Friedman, Hastie, and Tibshirani (2001) recommend values in $m = \{1, 2, ...\sqrt{m}\}$, but for high dimensional data with few relevant features, larger m can lead to better results because the probability of including relevant features increases.
- `n_estimators`, $\{1, 2, ...\}$: Number of trees to grow. In combination with early stopping, this can be set to a high value since learning will stop when the loss converges.
- `class_weight` {balanced, $1, 2, ...\}$: Weights on target classes: "balanced" calculates weights according to class frequencies, integer values specify weight on majority class relative to minority

3.5.3.2 Gradient Boosting Machine

Boosting is a method that can be applied to any learning algorithm. The main idea behind boosting is to sequentially train an ensemble of weak learners which on their own are only slightly better than a random decision. The predictions of the individual weak learners are then combined into a majority vote. The idea was first mentioned by Kearns (1988). The first algorithm that gained widespread popularity was introduced by Freund and Schapire (1997) in the form of the algorithm AdaBoost.M1, intended for classification problems.

Gradient boosting machine (GBM) extends on this idea. Like a random forest, GBM learns many trees which form an ensemble. However, trees are learned in an additive manner. At each iteration, the tree that improves the model most (i.e. in the direction of the gradient of the loss function) is added. For this thesis, the package `XGBoost` by Chen and Guestrin (2016) was used.

Assume we have a data set with n examples and m features: $D = \{\{\mathbf{x_i}, y_i\}\}(|D| = n, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \mathbb{R})$. The implementation uses a tree ensemble using K regression trees to predict the outcome for an example in the data by summing up the weights predicted by each tree:

$$\hat{y}_i = \phi(\mathbf{x_i}) = \sum_{k=1}^{K} f_k(\mathbf{x_i}), f_k \in F \tag{3.4}$$

where $F = \{f(\mathbf{x}) = w_{q(x)}\}(q : \mathbb{R}^m \rightarrow T, w \in \mathbb{R}^T)$ is the space of regression trees. T is the number of leaves in a tree, q is the structure of each tree, mapping an example to the corresponding leaf index. Each tree f_k has an independent structure q and weights w at the terminal leafs. An example is classified on each tree in F and the weights of the corresponding leafs are summed up to calculate the final prediciton.

For learning the functions in F, the following loss function is minimized:

$$L(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k}^{K} \Omega(f_k) \tag{3.5}$$

Here, l is a differentiable, convex loss function that measures the difference between predictions and true values. Since l is convex, we are guaranteed to find a local minimum. $\Omega(f) = \gamma T + \frac{1}{2}\lambda||w||^2$ is a penalty on the complexity of the trees to counter over-fitting. The algorithm thus features integrated regularization.

Now, at each iteration t, the tree f_t that improves the model most is added. For this, we add $f_t(\mathbf{x_i})$ to the predictions at $t-1$.

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{t-1} + f_t(\mathbf{x_i})) + \Omega(f_t) \tag{3.6}$$

To find the best f_t to add, the gradients finally come into play. With g_i and h_i the first- and second-order gradient statistics of l, the loss function becomes:

$$\tilde{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{t-1} + g_i f_t(\mathbf{x_i}) + h_i f_t^2(\mathbf{x_i})) + \Omega(f_t) \tag{3.7}$$

A neat feature of XGBoost is it's ability to deal with missing values. This means that no imputation is necessary during preprocessing, reducing the risk of introducing additional noise through imputation and implicitly allowing to pick up patterns due to the presence of missing values.

Hyperparameters

- `learning_rate`, $[0, 1]$: *Shrinkage*, decreases step size for the gradient descent when $\eta < 1.0$, helping convergence. The number of estimators f_k has to be increased for small learning rates in order for the algorithm to converge.
- `n_iter_no_change`, $\{0, 1, 2, 3, ...\}$: Early stopping. Based on an evaluation set, learning stops when no improvement on the performance metric (misclassification error was chosen) is made for a fixed number of steps.
- `subsample`, $[0, 1]$: A random sample from the n examples of size $s, s < n$ is drawn for each iteration, countering overfitting and speeding up learning.
- `colsample_by_tree`, $[0, 1]$: A random sample of the m features is drawn for growing each tree.

3.5.3.3 GLMnet

The GLMnet is an implementation of a generalized linear model (GLM) with penalized maximum likelihood by Hastie and Qian (2014). Regularization is achieved through L^2 and L^1 penalties (i.e. ridge and the lasso or their combination known as elastic net Zou and Hastie (2005)).

For learning, glmnet tries many different λ values for a given α. Each λ is then evaluated through cross validation. The data (features and target) was transformed to mean zero and unit variance before learning.

For the binary classification task at hand, a logistic regression was performed. The logistic regression model for a two-class response $G = \{0, 1\}$ with target $y_i = I(g_i = 1)$:

$P(G = 1|X = x) = \frac{e^{\beta_0+\beta^T x}}{1+e^{\beta_0+\beta^T x}}$ or, in the log-odds transformation: $log\frac{P(G=1|X=x)}{P(G=0|X=x)} = \beta_0 + \beta^T x$.

The loss function is:

$$\min_{\beta_0,\beta} \frac{1}{N} \sum_{i=1}^{N} w_i l(y_i, \beta_0 + \beta^T x_i) + \lambda \left[(1-\alpha)||\beta||_2^2/2 + \alpha||\beta||_1\right] \tag{3.8}$$

where w_i are individual sample weights, $l()$ the (negative) log-likelihood of the parameters □ given the data, λ the amount of penalization and $\alpha \in [0, 1]$ the elastic net parameter, for $\alpha = 0$ pure ridge and for $\alpha = 1$ pure lasso.

For the regression task, a gaussian family model was used, having loss function:

$$\min_{(\beta_0,\beta)\in\mathbb{R}^{p+1}} \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - x_i^T\beta)^2 + \lambda \left[(1-\alpha)||\beta||_2^2/2 + \alpha||\beta||_1\right] \tag{3.9}$$

where $l()$ the (negative) log-likelihood of the parameters □ given the data, λ the amount of penalization and $\alpha \in [0, 1]$ the elastic net parameter, for $\alpha = 0$ pure ridge and for $\alpha = 1$ pure lasso.

Hyperparameters

- `n_splits`, $\{3, 4, 5, ...\}$: Number of CV-splits. Typical values are 3, 5 and 10.
- α, $[0, 1]$: parametrizes the elastic net. For $\alpha = 0$ pure ridge, for $\alpha = 1$ pure lasso.
- `scoring`: Scoring method for cross-validation (log-loss, classification error, accuracy, precision, recall, average precision, roc-auc)

3.5.3.4 Multilayer Perceptron

The multilayer perceptron (MLP) is a so-called feedforward neural network. The term feedforward means that information flows from the input layer through intermediary steps and then to the output.

The goal is to approximate the function f^*. For a classifier, $y = f^*(\mathbf{x})$ maps an example $\mathbf{x}$ to a category y. A feedforward network defines a mapping $\mathbf{y} = f(\mathbf{x}, \mathbf{w})$ and learns the weights w by approximating the function f.

The MLP consists of at least three layers: an input layer, an arbitrary number of hidden layers and an output layer. For a binary classification problem on a dataset with n examples and m features $D = \{\{\mathbf{x_i}, y_i\}\}, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \{0, 1\}, i = 1...n$, the input layer has m units, the output layer has 1 unit. The hidden layers each have an arbitrary number of units.

Each unit, except for the input layer, consists of a perceptron, which is in effect a linear model with some non-linear activation function applied:

$$y = \phi(\mathbf{w}^T\mathbf{x} + b) \tag{3.10}$$

where ϕ is a non-linear activation function, **w** is the vector of weights, **x** is the vector of inputs and b is the bias. For ϕ, typical functions are the hyperbolic tangens $tanh(\cdot)$, the logistic sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$, or, more recently, the rectified linear unit $relu(x) = max(0, x)$ proposed by Hahnloser et al. (2000).

For learning, the weights in each unit are initialized first (typically random, near-zero). Then, the training examples are fed to the network sequentially. For each example, the prediction error is calculated using a loss function, which is typically the negative log-likelihood.

Then, the partial derivatives of the loss function with respect to the weights are computed for each unit and the parameters updated using stochastic gradient descent through backpropagation.

For this thesis, the scikit-learn implementation `sklearn.neural_network.MLPClassifier` was used. The network topology was determined by trying several variants. Best results were achieved with a network featuring two hidden layers with 50 and 10 hidden units, respectively.

The complete network is a chain of functions, for the network trained here, it is:

$$\mathbf{y} = \phi^{(3)}(\mathbf{W}^{(3)T}\square^{(2)}(\mathbf{W}^{(2)T}\square^{(1)}(\mathbf{W}^{(1)T}\mathbf{x} + \mathbf{b}^{(1)}) + \mathbf{b}^{(2)}) + \mathbf{b}^{(3)}) \tag{3.11}$$

where **x** is the vector of input features, **y** is the vector of predicted outputs, $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{W}^{(3)}$ are the weight matrices for each layer and $\mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \mathbf{b}^{(3)}$ are the bias vectors for each layer and $\square^{(1)}$, $\square^{(2)}$, $\square^{(3)}$ are the sets of perceptrons in the corresponding layer.

3.5.3.5 Support Vector Machine

(Boser et al.,1992; Vapnik,1998; Schölkopf et al.,1999a)

$$y(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{n} w_i K(\mathbf{x}, \mathbf{x_i}) + w_0 \tag{3.12}$$

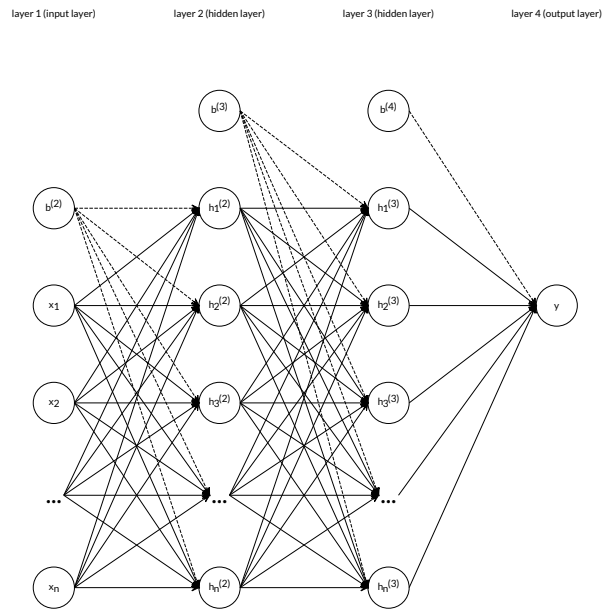

Figure 3.4: Neural network topology used. Two hidden layers $\mathbf{h^{(1)}}, \mathbf{h^{(2)}}$ are contained. $\mathbf{b^{(2)}}, \mathbf{b^{(3)}}$ and $\mathbf{b^{(4)}}$ are the bias vectors for the respective layers.

3.5.3.6 Bayesian Ridge Regression

Bayesian Ridge Regression (BR) can be seen as a Bayesian approach to l_2 regularization. The scikit-learn implementation was used. The algorithm implements Tipping (2001). We have the linear regression model for ordinary linear regression (see Albert (2009)), where we assume $\{y_i\}$ are conditionally independent, $var(y_i|\square, \mathbf{X}) = \sigma^2$ and the errors $\epsilon_i = y_i - E(y_i|\beta, X) \sim$ iid with mean 0 and variance σ^2:

$$p(\mathbf{y}|\square, \sigma^2, \mathbf{X}) \sim N_n(\mathbf{X}\square, \sigma^2\mathbf{I}) \tag{3.13}$$

where $\mathbf{y}$ is the vector of observed target values, X is the design matrix with $\mathbf{x}_1, ...\mathbf{x}_n$ examples and $N_k(\mu, A)$ is a multivariate normal distribution of dimension k with mean μ and variance-covariance matrix A.

We now put the following prior distribution on the coefficients β:

$$p(\beta|\lambda) = N(\beta|0, \lambda^{-1}(I)) \tag{3.14}$$

For α and λ, gamma distributions are chosen as prior distributions. The parameters of the gamma distributions $\alpha_1, \alpha_2, \lambda_1$ and λ_2 are chosen non-informative and are set to $\alpha_1 = \alpha_2 = \lambda_1 = \lambda_2 = 10^{-6}$.

4 Results and Discussion

Below, the results from data preprocessing, model evaluation, -selection and predictions are shown.

4.1 Preprocessing With Package kdd98

The self-written package `kdd98` ensures consistent data provisioning. It handles downloading and preprocessing of the data set for both the learning and validation set. All preprocessing steps are trained on the learning data set. The individual trained transformations are persisted on disk. After training, the transformations can be applied on the validation data. This process is transparent to the user. It is enough to instantiate the data provider for either learning or validation data set and request the data. Examples for usage can be found in the Jupyter notebooks.

The data sets can be obtained at the following intermediate steps from `kdd98.data_handler.KDD98DataProvider`:

- **raw**, as imported through `pandas.read_csv()`
- **preprocessed**, input errors removed, correct data types for all features, missing at random (MAR) imputations applied
- **numeric**, after feature engineering (encoded categories, date and zip code transformations)
- **imputed**, with missing values imputed
- **all-relevant**, filtered down to a set of relevant features

For some transformers, behavior can be controlled by specifying parameters. The package's architecture furthermore makes it easy to implement additional transformation steps.

The source code, along with a short introduction, is available online[1].

4.2 Imputation

The evaluation of several imputation strategies led to a straightforward approach: Categorical features had a *missing* level added during encoding (Section 3.3.2). All other features were imputed by their median value to account for the skewed distributions. The notebook `4_Imputation.ipynb`[2] contains details on the other approaches studied.

In concordance with the cup documentation's requirements, *sparse* features were dropped from the data set before imputation. A balance had to be found so as to not exclude too much information on the donation patterns. As a consequence, the threshold was set to ≥ 90 % missing values.

[1] see github.com/datarian/master-thesis-code/kdd98

[2] see github.com/datarian/master-thesis-code/notebooks/4_Imputation.ipynb

Missing data after removing sparse features is shown in Figure 4.1. The matrix displays the complete data set, with missing values indicated by white cells.

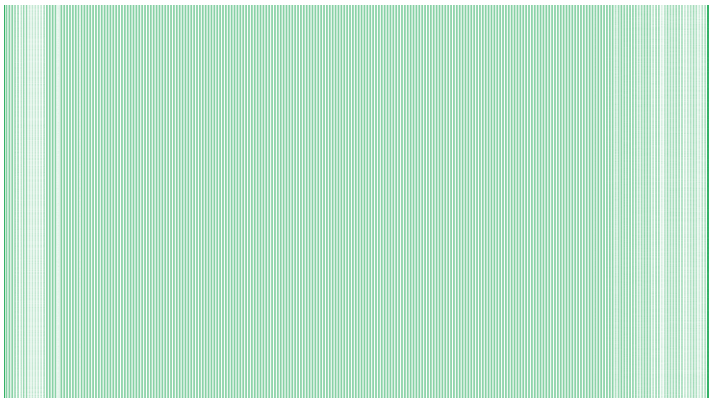

Figure 4.1: Data before imputation of numeric features. The big complete block at the center is the US census data.

The features with most and least missing values are shown in Figure 4.2. It is not surprising to find the *MONTHS_TO_DONATION_** features among those with most missing because few examples respond to the promotions with a donation.

Among the incomplete features with least missing values, we find several of the US census features. The *RFA_** features give the status in reference to promotion i. All examples who did donate at some point before have an RFA status. Thus, we can see when new members were added from the missing values in these features because newly added members do not have an RFA status yet.

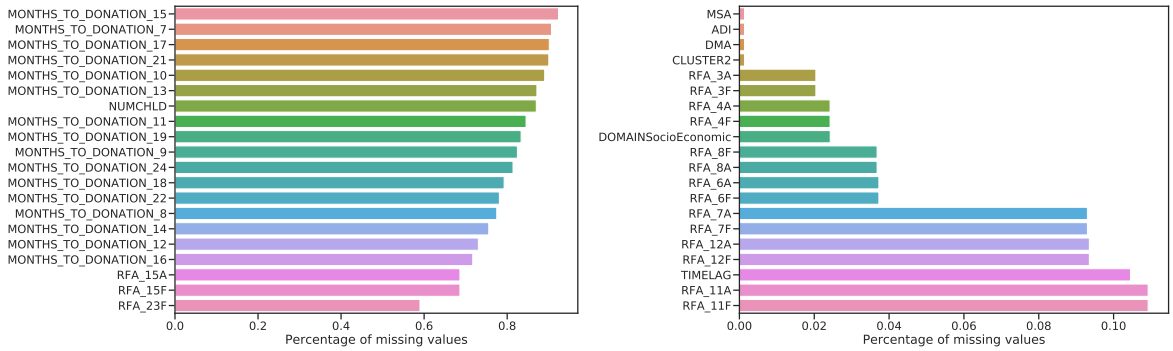

Figure 4.2: Features with most (left) and fewest (right) missing values.

4.3 Feature Selection

After preprocessing and feature engineering, 653 features were present in the data. Using boruta, 58 features were identified as important, resulting in a 91 % reduction of the number of features. For details, refer to[3].

[3] see make_github_link((“notebooks”, “5_Feature Extraction.ipynb”))

Three groups of features were selected.

Features from the **giving history** broadly correspond to those used in classical RFM models mentioned in literature (Section 1). It is reassuring to find them among the all-relevant features:

- Donation amount for promotion 14
- Summary features: All-time donation amount, all-time number of donations, smallest, average and largest donation, donation amount of most recent donation
- 24 Features on frequency and amount of donations as per the date of past donations
- Time since first donation, Time since largest donation, time since last donation
- Number of donations in response to card promotions
- Number of months between first and second donation
- An indicator for *star* donors

The **promotion history** features can be interpreted as a measure of the importance of the examples to the organization. Those who receive many promotions are deemed valuable:

- Number of promotions received
- Number promotions in last 12 months before the current promotion
- Number of card promotions received
- Number of card promotions in last 12 months before the current promotion

Features from the **US census** data seem to be concerned with the social status and wealth of the neighborhood of donors and by intuition make sense to be deemed relevant.

- Median and average home value
- Percentage of home values above some threshold (5 features)
- Percentage of renters paying more than 500 $
- DMA (designated market area, a geographical grouping)
- Median / average family / house income
- Per capita income
- Percentage of households with interest, rental or dividend income
- Percentage of adults with a Bachelor's degree
- Percentage of people born in state of residence

4.4 Classifiers

The four classifiers that were evaluated can be compared in terms of their receiver operating characteristic (ROC), which indicates overall model performance through an area under the curve value (ROC-AUC), precision-recall (PR) curves, or by their individual performance expressed through confusion matrices and the scores for several different metrics.

The ROC-AUC curve is constructed by evaluating the false positive rate (FPR) against the true positive rate (TPR) at various thresholds for the predicted class probabilities of examples in the training data. The closer the curve is to the top-left corner, the better a model performs (we have a large TPR and at the same time a low FPR). Looking at the ROC-AUC curves shown in Figure 4.3, we see that all classifiers performed rather weak. In the case of imbalanced data, the majority class dominates this metric. The

false positive rate is $FPR = \frac{FP}{TN}$. This means that as the false positives (FP) decrease, FPR does not change a lot.

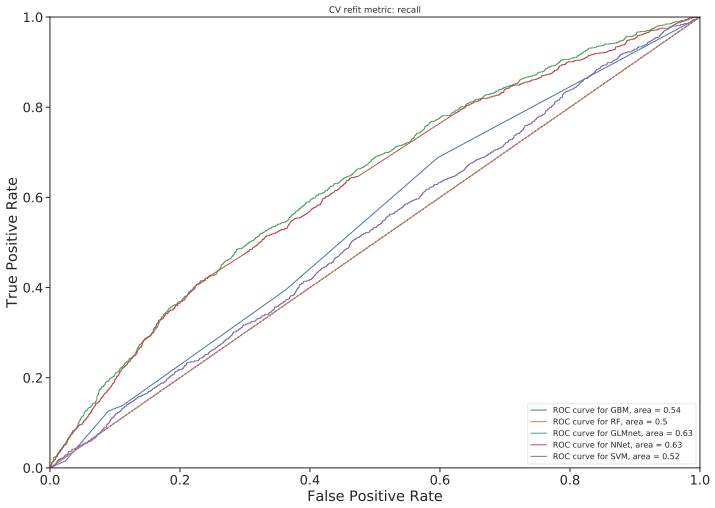

Figure 4.3: Comparison of ROC-AUC for the evaluated classifiers.

The PR curve with precision $P = \frac{TP}{TP+FP}$ "how many of the predicted positive cases are true positives?" plotted against recall $R = \frac{TP}{TP+FP}$ "how many of the true positive values can we predict correctly?" at different threshold values, is sensitive to false positives and, since TN is not involved, better suited for the imbalanced data at hand. Figure 4.4 shows the models in direct comparison. Here, the more the curve is towards the top right corner, the better. In that case, a high precision can be maintained at high recall levels.

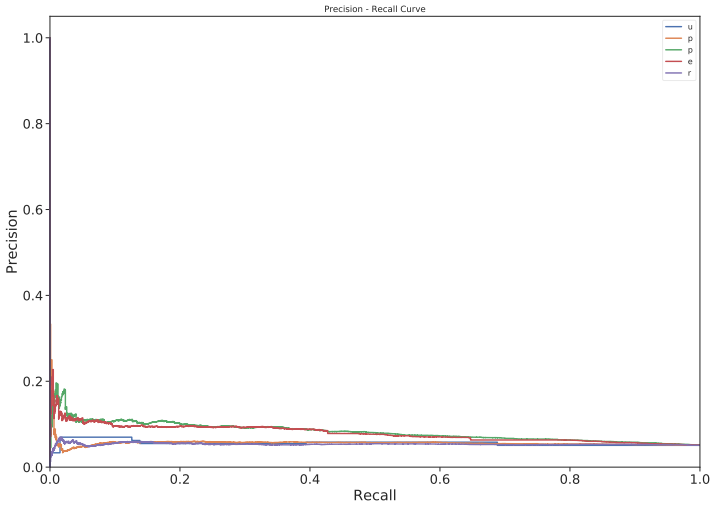

Figure 4.4: Comparison of ROC-AUC for the evaluated classifiers.

Evidently, all learned algorithms perform rather weak.

4.5 Regressors

Regressors were learned on the learning data set with *TARGET_D* as

For the regression models, the target was transformed using a Box-Cox transformation with parameter $\lambda = 0.024$. The goal was to linearize the target to improve regression model's performance. The transformed data somewhat resembles a normal distribution, although there are several modes to be made out.

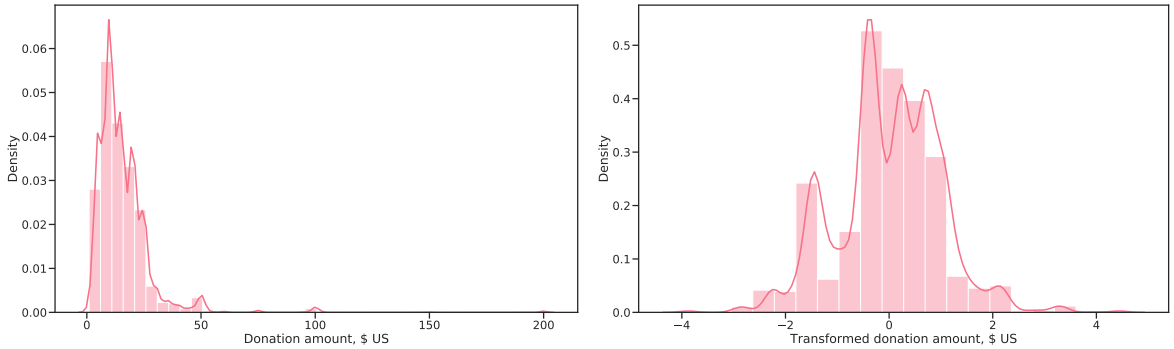

Figure 4.5: Target before transformation (left) and after a Box-Cox transformation (right).

The resulting distribution of predictions on the training data from the models evaluated is shown in Figure 4.6. Except for SVR, the models produce very similar results. We again find the multimodal distribution, with the SVR capturing the true frequencies of the donation amounts better than the other models.

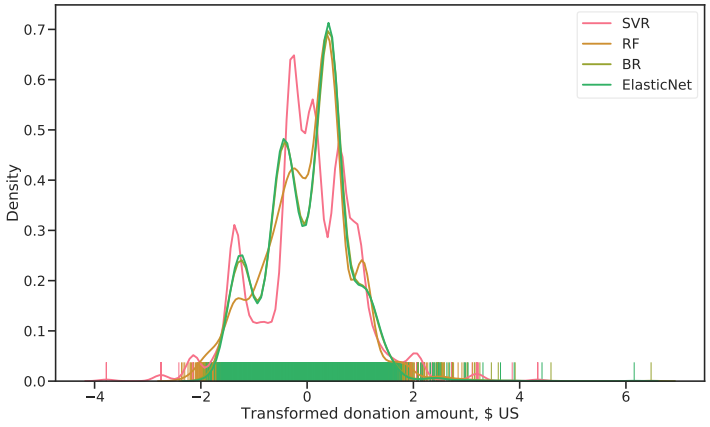

Figure 4.6: Distributions of the predicted donation amount for all regressors evaluated. Except for SVR, the models perform very similar.

The comparison of R^2 for the models evaluated shows SVR as the best performing model (see Figure 4.7). The SVR was therefore chosen for use in the final profit prediction.

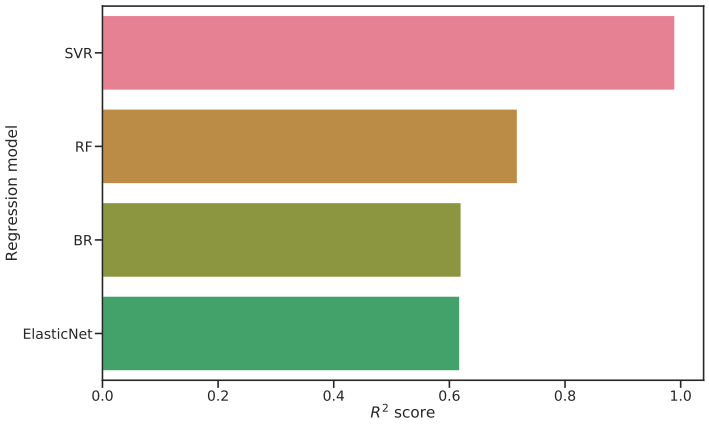

Figure 4.7: Evaluation metric R^2 for all regression models evaluated. SVR performs superior, with near-perfect R^2.

4.6 Prediction

The intermediate steps for arriving at the final prediction will be examined in this section.

4.6.1 Conditional Prediction of the Donation Amount

Using the regression model selected in section 4.5, the predicted donation amount for the learning data is strictly positive because the model was learned conditionally on the examples that made a donation (see Figure 4.8). Compared to the original distribution in Figure 4.5, we see a similar distribution, with the mode at approximately 13 $ and the largest donations in excess of 175 $.

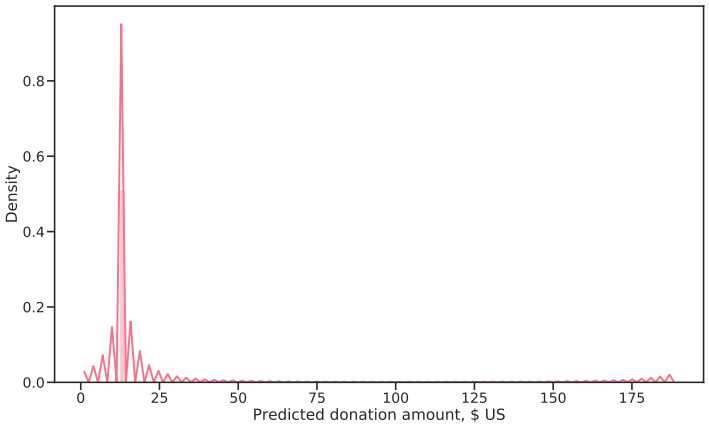

Figure 4.8: Conditionally predicted donation amounts, transformed back to the original scale (the model was trained on the Box-Cox transformed target).

4.6.2 Profit Optimization

The correction factor α, used to account for the bias introduced by learning the regression models on a non-random sample, was found as described in section 3.4. First, expected profit $E(Z)$ was calculated using equation (3.3) for a grid of α values. Then, a polynomial model and a cubic spline were fitted to the data and α optimized subsequently.

As can be seen in Figure 4.9, profit is very sensitive to α. This is not surprising given the distribution in Figure 4.8 which is narrowly concentrated around 13 \$. Furthermore, the curve is constant over much of the domain, meaning that all examples were selected from approximately $\alpha = 0.15$. The sharp spike combined with the large proportion of constant expected profit makes fitting a polynomial difficult. The cubic spline was therefore used to find the optimal value α^*.

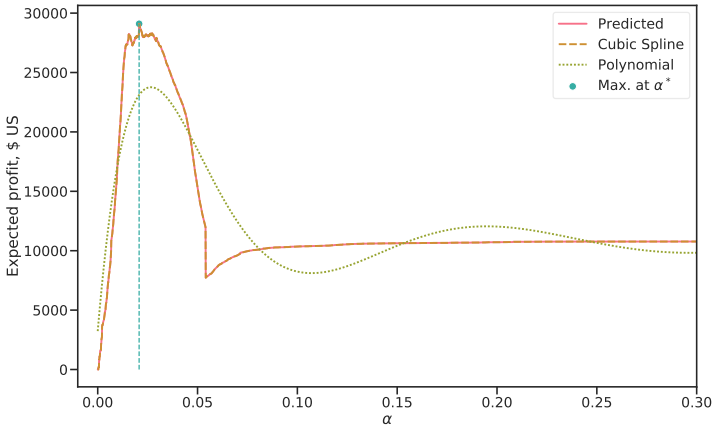

Figure 4.9: Expected profit for a range of α values in $[0, 1]$ with overlay-ed cubic spline and polynomial function of order 12. α^* was found by considering the roots of the derivate of the cubic spline, choosing the root where expected profit was highest.

4.6.3 Final Prediction

For final predictions, the complete learning data set was used to train the best performing classifier and regressor and find α^*.

The learned estimators were then applied on the test data set. The results are shown in 4.1. The theoretical maximum was calculated with: $\sum_{i=1}^{n} \mathbb{1}_{\{\text{TARGET}_{D,i}>0.0\}} * (\text{TARGET}_{D,i} - u)$ with u the unit cost of 0.68 \$ per mailing.

Table 4.1: Prediction results for the learning and test data set.

	Members mailed, #	Net Profit, \$ US	Theoretical maximum profit, \$ US
Learning	31530	28951	72375
Test	31880	3047	72775

5 Conclusions

5.1 Comparison With Cup Winners

The KDD-CUP committee evaluated the results based on the net revenue generated on the validation sample. The measure used was the sum (the actual donation amount - $0.68) over all records for which the expected revenue (or predicted value of the donation) is over $0.68. This measure is simple, objective and a direct measure of profit. Table 2 depicts the results. The participants are listed based on the last column.

Table 5.1: Top five of the KDD-CUP participants. N* denotes the number for which the predicted donation amount is > $0.68. Sum is the total profit, meaning the donation minus 0.68 for each example.

		Amount, $				
	N*	Min	Mean	Std	Max	Sum
GainSmarts	56330	-0.68	0.26	5.57	499.32	14712
SAS	55838	-0.68	0.26	5.64	499.32	14662
Quadstone	57836	-0.68	0.24	5.66	499.32	13954
CARRL	55650	-0.68	0.25	5.61	499.32	13825
Amdocs	51906	-0.68	0.27	5.69	499.32	13794

5.2 Biggest Problems Remaining

- Feature Extraction
- Feature Selection
- Choice of Models
- Poor prediction performance: Non-gaussian distribution of $\hat{y}_b$, which violates assumptions for Heckman-correction., bias-variance tradeoff

Prediction Error $PE(z) = \sigma_\epsilon^2 + \text{Bias}^2(\hat{f}(z)) + \text{Var}(\hat{f}(z))$

- Will be easy to work on these specific areas given the infrastructure created in this thesis

References

Albert, Jim. 2009. *Bayesian Computation with R*. Springer Science & Business Media.

Allaire, JJ, Joe Cheng, Yihui Xie, Jonathan McPherson, Winston Chang, Jeff Allen, Hadley Wickham, Aron Atkins, Rob Hyndman, and Ruben Arslan. 2016. *Rmarkdown: Dynamic Documents for R. R Package Version*. Vol. 1.

Bellman, Richard E. 1966. "Dynamic Programming." *Science* 153 (3731): 34–37.

Breiman, Leo. 1996. "Bagging Predictors." *Machine Learning* 24 (2): 123–40.

———. 2001. "Random Forests." *Machine Learning* 45 (January): 5–32.

Breiman, Leo, JH Friedman, R Olshen, and CJ Stone. 1984. "Classification and Regression Trees."

Chen, Tianqi, and Carlos Guestrin. 2016. "Xgboost: A Scalable Tree Boosting System." In *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 785–94. ACM.

Freund, Yoav, and Robert E Schapire. 1997. "A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting." *Journal of Computer and System Sciences* 55 (1): 119–39.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. 2001. *The Elements of Statistical Learning*. Vol. 1. 10. Springer series in statistics New York, NY, USA.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Hahnloser, Richard HR, Rahul Sarpeshkar, Misha A Mahowald, Rodney J Douglas, and H Sebastian Seung. 2000. "Digital Selection and Analogue Amplification Coexist in a Cortex-Inspired Silicon Circuit." *Nature* 405 (6789): 947.

Hastie, Trevor, and Junyang Qian. 2014. "Glmnet Vignette." *Retrieve from Http://Www. Web. Stanford. Edu/~ Hastie/Papers/Glmnet_Vignette.pdf. Accessed 30.05.2019* 20: 2016.

Hughes, AM. 1996. "Boosting Response with Rfm." *Marketing Tools* 5 (January): 4–7.

Hughes, Gordon. 1968. "On the Mean Accuracy of Statistical Pattern Recognizers." *IEEE Transactions on Information Theory* 14 (1): 55–63.

Hunter, JD. 2007. "Matplotlib: A 2D Graphics Environment." *Computing in Science Engineering* 9 (3): 90–95.

Kearns, Michael. 1988. "Thoughts on Hypothesis Boosting." *Unpublished Manuscript* 45: 105.

Kluyver, Thomas, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, et al. 2016. "Jupyter Notebooks – a Publishing Format for Reproducible Computational Workflows." Edited by F Loizides and B Schmidt. IOS Press.

Kohavi, Ron, and Rajesh Parekh. 2004. "Visualizing Rfm Segmentation." In *Proceedings of the 2004 Siam International Conference on Data Mining*, 391–99. SIAM.

Kursa, Miron B, and Witold Rudnicki. 2011. "The All Relevant Feature Selection Using Random Forest." *arXiv Preprint arXiv:1106.5112*, June.

Kursa, Miron B, Witold R Rudnicki, and others. 2010. "Feature Selection with the Boruta Package." *Journal of Statistical Software* 36 (11): 1–13.

Lemaître, Guillaume, Fernando Nogueira, and Christos K Aridas. 2017. "Imbalanced-Learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning." *Journal of Machine Learning Research* 18 (17): 1–5.

McCarty, John A, and Manoj Hastak. 2007. "Segmentation Approaches in Data-Mining: A Comparison of Rfm, Chaid, and Logistic Regression." *Journal of Business Research* 60 (6): 656–62.

McKinney, Wes. 2010. "Data Structures for Statistical Computing in Python." In *Proceedings of the 9th Python in Science Conference*, edited by Stéfan van der Walt and Jarrod Millman, 51–56.

———. 2011. "Pandas: A Foundational Python Library for Data Analysis and Statistics." *Python for High Performance and Scientific Computing* 14.

Oliphant, Travis E. 2006. *A Guide to Numpy*. Vol. 1. Trelgol Publishing USA.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12: 2825–30.

Rubin, Donald B. 1976. "Inference and Missing Data." *Biometrika* 63 (3): 581–92.

Tipping, Michael E. 2001. "Sparse Bayesian Learning and the Relevance Vector Machine." *Journal of Machine Learning Research* 1 (Jun): 211–44.

Troyanskaya, Olga, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. 2001. "Missing Value Estimation Methods for Dna Microarrays." *Bioinformatics* 17 (6): 520–25.

van Buuren, Stef, and Karin Groothuis-Oudshoorn. 2011. "Mice: Multivariate Imputation by Chained Equations in R." *Journal of Statistical Software, Articles* 45 (3): 1–67.

Waskom, Michael, O Botvinnik, P Hobson, J Warmenhoven, JB Cole, Y Halchenko, J Vanderplas, et al. 2014. "Seaborn: Statistical Data Visualization." *Seaborn: Statistical Data Visualization Seaborn 0* 5.

Xie, Yihui. 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Chapman and Hall/CRC.

———. 2016. *Bookdown: Authoring Books and Technical Documents with R Markdown*. Chapman and Hall/CRC.

Zou, Hui, and Trevor Hastie. 2005. "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67 (2): 301–20.

.1 Python Environment

Python was installed through the Anaconda distribution[1].

The analysis documented should be reproduced using the same environment, thereby ensuring that the packages produce the same results and expose the API's used in the code.

The environment can be created using the file `environment.yml` in the main repository directory at github.com/datarian/master-thesis-code/.

Once conda is installed the following two commands first set up the environment with all the analysis' required packages, then activates that environment. Only after running these commands should the package `kdd98` be installed and the notebooks evaluated.

```
> conda env create -f environment.yml
> source activate ma-thesis-fh
```

.2 Data Set Dictionary

Below, the content of the file *cup98DIC.txt are given verbatim.

```
======================================================================
EPSILON CONFIDENTIAL      EPSILON CONFIDENTIAL    EPSILON CONFIDENTIAL

    INFORMATION LISTED BELOW IS AVAILABLE UNDER THE TERMS OF THE
                      CONFIDENTIALITY AGREEMENT

EPSILON CONFIDENTIAL      EPSILON CONFIDENTIAL    EPSILON CONFIDENTIAL
======================================================================

+--------------------------------------------------------------------+
|                PARALYZED VETERANS OF AMERICA (PVA)                 |
|                  DATA DICTIONARY TO ACCOMPANY                      |
|                                                                    |
|                          KDD-CUP-98                                |
|                                                                    |
|          The Second International Knowledge Discovery and         |
|                 Data Mining Tools Competition                      |
|                                                                    |
|                Held in Conjunction with KDD-98                     |
|                                                                    |
|          The Fourth International Conference on Knowledge         |
```

[1] Available from https://www.anaconda.com

```
|                   Discovery and Data Mining                          |
|                    [www.kdnuggets.com] or                            |
|                 [www-aig.jpl.nasa.gov/kdd98] or                      |
|                [www.aaai.org/Conferences/KDD/1998]                   |
|                                                                      |
|                        Sponsored by the                              |
|                                                                      |
|      American Association for Artificial Intelligence (AAAI)        |
|                  Epsilon Data Mining Laboratory                      |
|                Paralyzed Veterans of America (PVA)                   |
+----------------------------------------------------------------------+
|                                                                      |
| Created:     7/20/98                                                 |
| Last update: 7/20/98                                                 |
| file name:   cup98DIC.txt                                            |
|                                                                      |
+----------------------------------------------------------------------+

Variable                    Description
--------------------------  ----------------------------------------
ODATEDW                     Origin Date. Date of donor's first gift
                            to PVA YYMM format (Year/Month).

OSOURCE                     Origin Source
                            - (Only 1rst 3 bytes are used)
                            - Defaulted to 00000 for conversion
                            - Code indicating which mailing list the
                              donor was originally acquired from
                            - A nominal or symbolic field.

TCODE                       Donor title code
                            000    = _
                            001    = MR.
                            001001 = MESSRS.
                            001002 = MR. & MRS.
                            002    = MRS.
                            002002 = MESDAMES
                            003    = MISS
                            003003 = MISSES
                            004    = DR.
                            004002 = DR. & MRS.
                            004004 = DOCTORS
                            005    = MADAME
                            006    = SERGEANT
                            009    = RABBI
                            010    = PROFESSOR
                            010002 = PROFESSOR & MRS.
```

```
010010 = PROFESSORS
011    = ADMIRAL
011002 = ADMIRAL & MRS.
012    = GENERAL
012002 = GENERAL & MRS.
013    = COLONEL
013002 = COLONEL & MRS.
014    = CAPTAIN
014002 = CAPTAIN & MRS.
015    = COMMANDER
015002 = COMMANDER & MRS.
016    = DEAN
017    = JUDGE
017002 = JUDGE & MRS.
018    = MAJOR
018002 = MAJOR & MRS.
019    = SENATOR
020    = GOVERNOR
021002 = SERGEANT & MRS.
022002 = COLNEL & MRS.
024    = LIEUTENANT
026    = MONSIGNOR
027    = REVEREND
028    = MS.
028028 = MSS.
029    = BISHOP
031    = AMBASSADOR
031002 = AMBASSADOR & MRS.
033    = CANTOR
036    = BROTHER
037    = SIR
038    = COMMODORE
040    = FATHER
042    = SISTER
043    = PRESIDENT
044    = MASTER
046    = MOTHER
047    = CHAPLAIN
048    = CORPORAL
050    = ELDER
056    = MAYOR
059002 = LIEUTENANT & MRS.
062    = LORD
063    = CARDINAL
064    = FRIEND
065    = FRIENDS
068    = ARCHDEACON
```

```
                    069    = CANON
                    070    = BISHOP
                    072002 = REVEREND & MRS.
                    073    = PASTOR
                    075    = ARCHBISHOP
                    085    = SPECIALIST
                    087    = PRIVATE
                    089    = SEAMAN
                    090    = AIRMAN
                    091    = JUSTICE
                    092    = MR. JUSTICE
                    100    = M.
                    103    = MLLE.
                    104    = CHANCELLOR
                    106    = REPRESENTATIVE
                    107    = SECRETARY
                    108    = LT. GOVERNOR
                    109    = LIC.
                    111    = SA.
                    114    = DA.
                    116    = SR.
                    117    = SRA.
                    118    = SRTA.
                    120    = YOUR MAJESTY
                    122    = HIS HIGHNESS
                    123    = HER HIGHNESS
                    124    = COUNT
                    125    = LADY
                    126    = PRINCE
                    127    = PRINCESS
                    128    = CHIEF
                    129    = BARON
                    130    = SHEIK
                    131    = PRINCE AND PRINCESS
                    132    = YOUR IMPERIAL MAJEST
                    135    = M. ET MME.
                    210    = PROF.

STATE               State abbreviation (a nominal/symbolic field)
ZIP                 Zipcode (a nominal/symbolic field)
MAILCODE            Mail Code
                    ” ”= Address is OK
                    B = Bad Address

PVASTATE            EPVA State or PVA State
                    Indicates whether the donor lives in a state
                    served by the organization's EPVA chapter
```

```
                        P = PVA State
                        E = EPVA State (Northeastern US)

DOB                     Date of birth (YYMM, Year/Month format.)
NOEXCH                  Do Not Exchange Flag (For list rental)
                        _ = can be exchanged
                        X = do not exchange

RECINHSE                In House File Flag
                        _ = Not an In House Record
                        X = Donor has given to PVA's In House program

RECP3                   P3 File Flag
                        _ = Not a P3 Record
                        X = Donor has given to PVA's P3 program

RECPGVG                 Planned Giving File Flag
                        _ = Not a Planned Giving Record
                        X = Planned Giving Record

RECSWEEP                Sweepstakes file flag
                        _ = Not a Sweepstakes Record
                        X = Sweepstakes Record

MDMAUD                  The Major Donor Matrix code
                        The codes describe frequency and amount of
                        giving for donors who have given a $100+
                        gift at any time in their giving history.
                        An RFA (recency/frequency/monetary) field.

                        The (current) concatenated version is a nominal
                      or symbolic field. The individual bytes could separately be
                        used as fields and refer to the following:

                        First byte: Recency of Giving
                          C=Current Donor
                          L=Lapsed Donor
                          I=Inactive Donor
                          D=Dormant Donor

                        2nd byte: Frequency of Giving
                          1=One gift in the period of recency
                          2=Two-Four gifts in the period of recency
                          5=Five+ gifts in the period of recency

                        3rd byte: Amount of Giving
                          L=Less than $100(Low Dollar)
```

	C=$100-499(Core) M=$500-999(Major) T=$1,000+(Top) 4th byte: Blank/meaningless/filler 'X' indicates that the donor is not a major donor. For more information regarding the RFA codes, see the promotion history field definitions.
DOMAIN	DOMAIN/Cluster code. A nominal or symbolic field. could be broken down by bytes as explained below. 1st byte = Urbanicity level of the donor's neighborhood U=Urban C=City S=Suburban T=Town R=Rural 2nd byte = Socio-Economic status of the neighborhood 1 = Highest SES 2 = Average SES 3 = Lowest SES (except for Urban communities, where 1 = Highest SES, 2 = Above average SES, 3 = Below average SES, 4 = Lowest SES.)
CLUSTER	CLUSTER Code indicating which cluster group the donor falls into. Each cluster is unique in terms of socio-economic status, urbanicty, ethnicity and a variety of other demographic characteristics. A nominal or symbolic field.
AGE	Overlay Age 0 = missing
AGEFLAG	Age Flag E = Exact I = Inferred from Date of Birth Field
HOMEOWNR	Home Owner Flag H = Home owner U = Unknown
CHILD03	Presence of Children age 0-3 B = Both, F = Female, M = Male

CHILD07	Presence of Childern age 4-7
CHILD12	Presence of Childern age 8-12
CHILD18	Presence of Childern age 13-18
NUMCHLD	NUMBER OF CHILDREN
INCOME	HOUSEHOLD INCOME
GENDER	Gender M = Male F = Female U = Unknown J = Joint Account, unknown gender
WEALTH1	Wealth Rating
HIT	MOR Flag # HIT (Mail Order Response) Indicates total number of known times the donor has responded to a mail order offer other than PVA's.
---	---
	The following variables indicate the number of known times the donor has responded to other types of mail order offers.
MBCRAFT	Buy Craft Hobby
MBGARDEN	Buy Gardening
MBBOOKS	Buy Books
MBCOLECT	Buy Collectables
MAGFAML	Buy General Family Mags
MAGFEM	Buy Female Mags
MAGMALE	Buy Sports Mags
PUBGARDN	Gardening Pubs
PUBCULIN	Culinary Pubs
PUBHLTH	Health Pubs
PUBDOITY	Do It Yourself Pubs
PUBNEWFN	News / Finance Pubs
PUBPHOTO	Photography Pubs
PUBOPP	Opportunity Seekers Pubs
---	---
DATASRCE	Source of Overlay Data Indicates which third-party data source the donor matched against 1 = MetroMail 2 = Polk 3 = Both

```
MALEMILI                % Males active in the Military
MALEVET                 % Males Veterans
VIETVETS                % Vietnam Vets
WWIIVETS                % WWII Vets
LOCALGOV                % Employed by Local Gov
STATEGOV                % Employed by State Gov
FEDGOV                  % Employed by Fed Gov

SOLP3                   SOLICIT LIMITATION CODE P3
                           = can be mailed (Default)
                        00 = Do Not Solicit or Mail
                        01 = one solicitation per year
                        02 = two solicitations per year
                        03 = three solicitations per year
                        04 = four solicitations per year
                        05 = five solicitations per year
                        06 = six solicitations per year
                        12 = twelve solicitations per year

SOLIH                   SOLICITATION LIMIT CODE IN HOUSE
                           = can be mailed (Default)
                        00 = Do Not Solicit
                        01 = one solicitation per year
                        02 = two solicitations per year
                        03 = three solicitations per year
                        04 = four solicitations per year
                        05 = five solicitations per year
                        06 = six solicitations per year
                        12 = twelve solicitations per year

MAJOR                   Major ($$) Donor Flag
                        _ = Not a Major Donor
                        X = Major Donor

WEALTH2                 Wealth Rating
                        Wealth rating uses median family income and
                        population statistics from each area to
                        index relative wealth within each state
                        The segments are denoted 0-9, with 9 being
                        the highest income group and zero being the
                        lowest. Each rating has a different meaning
                        within each state.

GEOCODE                 Geo Cluster Code indicating the level geography at which
                        a record matches the census data.
                        A nominal or symbolic field.
                        Blank=No code has been assigned or did not
```

```
                          match at any level.

-------------------------  ------------------------------------------------
                          The following variables reflect donor interests,
                          as collected from third-party data sources

COLLECT1                  COLLECTABLE (Y/N)
VETERANS                  VETERANS (Y/N)
BIBLE                     BIBLE READING (Y/N)
CATLG                     SHOP BY CATALOG (Y/N)
HOMEE                     WORK FROM HOME (Y/N)
PETS                      HOUSEHOLD PETS (Y/N)
CDPLAY                    CD PLAYER OWNERS (Y/N)
STEREO                    STEREO/RECORDS/TAPES/CD (Y/N)
PCOWNERS                  HOME PC OWNERS/USERS
PHOTO                     PHOTOGRAPHY (Y/N)
CRAFTS                    CRAFTS (Y/N)
FISHER                    FISHING (Y/N)
GARDENIN                  GARDENING (Y/N)
BOATS                     POWER BOATING (Y/N)
WALKER                    WALK FOR HEALTH (Y/N)
KIDSTUFF                  BUYS CHILDREN'S PRODUCTS (Y/N)
CARDS                     STATIONARY/CARDS BUYER (Y/N)
PLATES                    PLATE COLLECTOR (Y/N)

LIFESRC                   LIFE STYLE DATA SOURCE
                          Indicates source of the lifestyle variables listed
                          above
                          1 = MATCHED ON METRO MAIL ONLY
                          2 = MATCHED ON POLK ONLY
                          3 = MATCHED BOTH MM AND POLK

-------------------------  ------------------------------------------------

PEPSTRFL                  Indicates PEP Star RFA Status
                          blank = Not considered to be a PEP Star
                          'X'   = Has PEP Star RFA Status

-------------------------  ------------------------------------------------

                          The following variables reflect characteristics
                          of the donors neighborhood, as collected from the
                          1990 US Census.

POP901                    Number of Persons
POP902                    Number of Families
POP903                    Number of Households
```

POP90C1	Percent Population in Urbanized Area
POP90C2	Percent Population Outside Urbanized Area
POP90C3	Percent Population Inside Rural Area
POP90C4	Percent Male
POP90C5	Percent Female
ETH1	Percent White
ETH2	Percent Black
ETH3	Percent Native American
ETH4	Percent Pacific Islander/Asian
ETH5	Percent Hispanic
ETH6	Percent Asian Indian
ETH7	Percent Japanese
ETH8	Percent Chinese
ETH9	Percent Philipino
ETH10	Percent Korean
ETH11	Percent Vietnamese
ETH12	Percent Hawaiian
ETH13	Percent Mexican
ETH14	Percent Puerto Rican
ETH15	Percent Cuban
ETH16	Percent Other Hispanic
AGE901	Median Age of Population
AGE902	Median Age of Adults 18 or Older
AGE903	Median Age of Adults 25 or Older
AGE904	Average Age of Population
AGE905	Average Age of Adults >= 18
AGE906	Average Age of Adults >= 25
AGE907	Percent Population Under Age 18
CHIL1	Percent Children Under Age 7
CHIL2	Percent Children Age 7 - 13
CHIL3	Percent Children Age 14-17
AGEC1	Percent Adults Age18-24
AGEC2	Percent Adults Age 25-34
AGEC3	Percent Adults Age 35-44
AGEC4	Percent Adults Age 45-54
AGEC5	Percent Adults Age 55-64
AGEC6	Percent Adults Age 65-74
AGEC7	Percent Adults Age >= 75
CHILC1	Percent Children Age <=2
CHILC2	Percent Children Age 3-5
CHILC3	Percent Children Age 6-11
CHILC4	Percent Children Age 12-15
CHILC5	Percent Children Age 16-18
HHAGE1	Percent Households w/ Person 65+
HHAGE2	Percent Households w/ Person 65+ Living Alone
HHAGE3	Percent Households Headed by an Elderly Person Age 65+
HHN1	Percent 1 Person Households

HHN2	Percent 2 Person Households
HHN3	Percent 3 or More Person Households
HHN4	Percent 4 or More Person Households
HHN5	Percent 5 or More Person Households
HHN6	Percent 6 Person Households
MARR1	Percent Married
MARR2	Percent Separated or Divorced
MARR3	Percent Widowed
MARR4	Percent Never Married
HHP1	Median Person Per Household
HHP2	Average Person Per Household
DW1	Percent Single Unit Structure
DW2	Percent Detached Single Unit Structure
DW3	Percent Duplex Structure
DW4	Percent Multi (2+) Unit Structures
DW5	Percent 3+ Unit Structures
DW6	Percent Housing Units in 5+ Unit Structure
DW7	Percent Group Quarters
DW8	Percent Institutional Group Quarters
DW9	Non-Institutional Group Quarters
HV1	Median Home Value in hundreds
HV2	Average Home Value in hundreds
HV3	Median Contract Rent in hundreds
HV4	Average Contract Rent in hundreds
HU1	Percent Owner Occupied Housing Units
HU2	Percent Renter Occupied Housing Units
HU3	Percent Occupied Housing Units
HU4	Percent Vacant Housing Units
HU5	Percent Seasonal/Recreational Vacant Units
HHD1	Percent Households w/ Related Children
HHD2	Percent Households w/ Families
HHD3	Percent Married Couple Families
HHD4	Percent Married Couples w/ Related Children
HHD5	Percent Persons in Family Household
HHD6	Percent Persons in Non-Family Household
HHD7	Percent Single Parent Households
HHD8	Percent Male Householder w/ Child
HHD9	Percent Female Householder w/ Child
HHD10	Percent Single Male Householder
HHD11	Percent Single Female Householder
HHD12	Percent Households w/ Non-Family Living Arrangements
ETHC1	Percent White < Age 15
ETHC2	Percent White Age 15 - 59
ETHC3	Percent White Age 60+
ETHC4	Percent Black < Age 15
ETHC5	Percent Black Age 15 - 59
ETHC6	Percent Black Age 60+

HVP1	Percent Home Value >= $200,000
HVP2	Percent Home Value >= $150,000
HVP3	Percent Home Value >= $100,000
HVP4	Percent Home Value >= $75,000
HVP5	Percent Home Value >= $50,000
HVP6	Percent Home Value >= $300,000
HUR1	$ 1 or 2 Room Housing Units
HUR2	Percent >= 6 Room Housing Units
RHP1	Median Number of Rooms per Housing Unit
RHP2	Average Number of Rooms per Housing Unit
RHP3	Median Number of Persons per Housing Unit
RHP4	Average Number of Persons per Room
HUPA1	Percent Housing Units w/ 2 thru 9 Units at the Address
HUPA2	Percent Housing Units w/ >= 10 Units at the Address
HUPA3	Percent Mobile Homes or Trailers
HUPA4	Percent Renter Occupied Single Unit Structure
HUPA5	Percent Renter Occupied, 2 - 4 Units
HUPA6	Percent Renter Occupied, 5+ Units
HUPA7	Percent Renter Occupied Mobile Homes or Trailers
RP1	Percent Renters Paying >= $500 per Month
RP2	Percent Renters Paying >= $400 per Month
RP3	Percent Renters Paying >= $300 per Month
RP4	Percent Renters Paying >= $200 per Month
MSA	MSA Code
ADI	ADI Code
DMA	DMA Code
IC1	Median Household Income in hundreds
IC2	Median Family Income in hundreds
IC3	Average Household Income in hundreds
IC4	Average Family Income in hundreds
IC5	Per Capita Income
IC6	Percent Households w/ Income < $15,000
IC7	Percent Households w/ Income $15,000 - $24,999
IC8	Percent Households w/ Income $25,000 - $34,999
IC9	Percent Households w/ Income $35,000 - $49,999
IC10	Percent Households w/ Income $50,000 - $74,999
IC11	Percent Households w/ Income $75,000 - $99,999
IC12	Percent Households w/ Income $100,000 - $124,999
IC13	Percent Households w/ Income $125,000 - $149,999
IC14	Percent Households w/ Income >= $150,000
IC15	Percent Families w/ Income < $15,000
IC16	Percent Families w/ Income $15,000 - $24,999
IC17	Percent Families w/ Income $25,000 - 34,999
IC18	Percent Families w/ Income $35,000 - $49,999
IC19	Percent Families w/ Income $50,000 - $74,999
IC20	Percent Families w/ Income $75,000 - $99,999
IC21	Percent Families w/ Income $100,000 - $124,999

IC22	Percent Families w/ Income $125,000 - $149,999
IC23	Percent Families w/ Income >= $150,000
HHAS1	Percent Households on Social Security
HHAS2	Percent Households on Public Assistance
HHAS3	Percent Households w/ Interest, Rental or Dividend Income
HHAS4	Percent Persons Below Poverty Level
MC1	Percent Persons Move in Since 1985
MC2	Percent Persons in Same House in 1985
MC3	Percent Persons in Different State/Country in 1985
TPE1	Percent Driving to Work Alone Car/Truck/Van
TPE2	Percent Carpooling Car/Truck/Van)
TPE3	Percent Using Public Transportation
TPE4	Percent Using Bus/Trolley
TPE5	Percent Using Railways
TPE6	Percent Using Taxi/Ferry
TPE7	Percent Using Motorcycles
TPE8	Percent Using Other Transportation
TPE9	Percent Working at Home/No Transportation
PEC1	Percent Working Outside State of Residence
PEC2	Percent Working Outside County of Residence in State
TPE10	Median Travel Time to Work in minutes
TPE11	Mean Travel Time to Work in minutes
TPE12	Percent Traveling 60+ Minutes to Work
TPE13	Percent Traveling 15 - 59 Minutes to Work
LFC1	Percent Adults in Labor Force
LFC2	Percent Adult Males in Labor Force
LFC3	Percent Females in Labor Force
LFC4	Percent Adult Males Employed
LFC5	Percent Adult Females Employed
LFC6	Percent Mothers Employed Married and Single
LFC7	Percent 2 Parent Earner Families
LFC8	Percent Single Mother w/ Child in Labor Force
LFC9	Percent Single Father w/ Child in Labor Force
LFC10	Percent Families w/ Child w/ no Workers
OCC1	Percent Professional
OCC2	Percent Managerial
OCC3	Percent Technical
OCC4	Percent Sales
OCC5	Percent Clerical/Administrative Support
OCC6	Percent Private Household Service Occ.
OCC7	Percent Protective Service Occ.
OCC8	Percent Other Service Occ.
OCC9	Percent Farmers
OCC10	Percent Craftsmen, Precision, Repair
OCC11	Percent Operatives, Machine
OCC12	Percent Transportation
OCC13	Percent Laborers, Handlers, Helpers

EIC1	Percent Employed in Agriculture
EIC2	Percent Employed in Mining
EIC3	Percent Employed in Construction
EIC4	Percent Employed in Manufacturing
EIC5	Percent Employed in Transportation
EIC6	Percent Employed in Communications
EIC7	Percent Employed in Wholesale Trade
EIC8	Percent Employed in Retail Industry
EIC9	Percent Employed in Finance, Insurance, Real Estate
EIC10	Percent Employed in Business and Repair
EIC11	Percent Employed in Personnal Services
EIC12	Percent Employed in Entertainment and Recreation
EIC13	Percent Employed in Health Services
EIC14	Percent Employed in Educational Services
EIC15	Percent Employed in Other Professional Services
EIC16	Percent Employed in Public Administration
OEDC1	Percent Employed by Local Government
OEDC2	Percent Employed by State Government
OEDC3	Percent Employed by Federal Government
OEDC4	Percent Self Employed
OEDC5	Percent Private Profit Wage or Salaried Worker
OEDC6	Percent Private Non-Profit Wage or Salaried Worker
OEDC7	Percent Unpaid Family Workers
EC1	Median Years of School Completed by Adults 25+
EC2	Percent Adults 25+ Grades 0-8
EC3	Percent Adults 25+ w/ some High School
EC4	Percent Adults 25+ Completed High School or Equivalency
EC5	Percent Adults 25+ w/ some College
EC6	Percent Adults 25+ w/ Associates Degree
EC7	Percent Adults 25+ w/ Bachelors Degree
EC8	Percent Adults 25+ Graduate Degree
SEC1	Percent Persons Enrolled in Private Schools
SEC2	Percent Persons Enrolled in Public Schools
SEC3	Percent Persons Enrolled in Preschool
SEC4	Percent Persons Enrolled in Elementary or High School
SEC5	Percent Persons in College
AFC1	Percent Adults in Active Military Service
AFC2	Percent Males in Active Military Service
AFC3	Percent Females in Active Military Service
AFC4	Percent Adult Veterans Age 16+
AFC5	Percent Male Veterans Age 16+
AFC6	Percent Female Veterans Age 16+
VC1	Percent Vietnam Veterans Age 16+
VC2	Percent Korean Veterans Age 16+
VC3	Percent WW2 Veterans Age 16+
VC4	Percent Veterans Serving After May 1975 Only
ANC1	Percent Dutch Ancestry

ANC2	Percent English Ancestry
ANC3	Percent French Ancestry
ANC4	Percent German Ancestry
ANC5	Percent Greek Ancestry
ANC6	Percent Hungarian Ancestry
ANC7	Percent Irish Ancestry
ANC8	Percent Italian Ancestry
ANC9	Percent Norwegian Ancestry
ANC10	Percent Polish Ancestry
ANC11	Percent Portuguese Ancestry
ANC12	Percent Russian Ancestry
ANC13	Percent Scottish Ancestry
ANC14	Percent Swedish Ancestry
ANC15	Percent Ukranian Ancestry
POBC1	Percent Foreign Born
POBC2	Percent Born in State of Residence
LSC1	Percent English Only Speaking
LSC2	Percent Spanish Speaking
LSC3	Percent Asian Speaking
LSC4	Percent Other Language Speaking
VOC1	Percent Households w/ 1+ Vehicles
VOC2	Percent Households w/ 2+ Vehicles
VOC3	Percent Households w/ 3+ Vehicles
HC1	Percent Median Length of Residence
HC2	Percent Median Age of Occupied Dwellings in years
HC3	Percent Owner Occupied Structures Built Since 1989
HC4	Percent Owner Occupied Structures Built Since 1985
HC5	Percent Owner Occupied Structures Built Since 1980
HC6	Percent Owner Occupied Structures Built Since 1970
HC7	Percent Owner Occupied Structures Built Since 1960
HC8	Percent Owner Occupied Structures Built Prior to 1960
HC9	Percent Owner Occupied Condominiums
HC10	Percent Renter Occupied Condominiums
HC11	Percent Occupied Housing Units Heated by Utility Gas
HC12	Percent Occupied Housing Units Heated by Bottled, Tank or LP
HC13	Percent Occupied Housing Units Heated by Electricity
HC14	Percent Occupied Housing Units Heated by Fuel Oil
HC15	Percent Occupied Housing Units Heated by Solar Energy
HC16	Percent Occupied Housing Units Heated by Coal, Wood, Other
HC17	Percent Housing Units w/ Public Water Source
HC18	Percent Housing Units w/ Well Water Source
HC19	Percent Housing Units w/ Public Sewer Source
HC20	Percent Housing Units w/ Complete Plumbing Facilities
HC21	Percent Housing Units w/ Telephones
MHUC1	Median Homeowner Cost w/ Mortgage per Month dollars
MHUC2	Median Homeowner Cost w/out Mortgage per Month dollars
AC1	Percent Adults Age 55-59

AC2 Percent Adults Age 60-64

The fields listed below are from the promotion history file.

PROMOTION CODES:

The following lists the promotion codes and their respective field names (where XXXX refers to ADATE, RFA, RDATE and RAMNT.)

'97NK' ==> xxxx_2 (mailing was used to construct the target fields)
'96NK' ==> xxxx_3
'96TK' ==> xxxx_4
'96SK' ==> xxxx_5
'96LL' ==> xxxx_6
'96G1' ==> xxxx_7
'96GK' ==> xxxx_8
'96CC' ==> xxxx_9
'96WL' ==> xxxx_10
'96X1' ==> xxxx_11
'96XK' ==> xxxx_12
'95FS' ==> xxxx_13
'95NK' ==> xxxx_14
'95TK' ==> xxxx_15
'95LL' ==> xxxx_16
'95G1' ==> xxxx_17
'95GK' ==> xxxx_18
'95CC' ==> xxxx_19
'95WL' ==> xxxx_20
'95X1' ==> xxxx_21
'95XK' ==> xxxx_22
'94FS' ==> xxxx_23
'94NK' ==> xxxx_24

1st 2 bytes of the code refers to the year of the mailing while 3rd and 4th bytes refer to the following promotion codes/types:

LL mailings had labels only
WL mailings had labels only
CC mailings are calendars with stickers but do not have labels
FS mailings are blank cards that fold into

```
   thirds with labels
NK mailings are blank cards with labels
SK mailings are blank cards with labels
TK mailings have thank you printed on the
   outside with labels
GK mailings are general greeting cards (an
   assortment of birthday, sympathy, blank, & get
   well) with labels
XK mailings are Christmas cards with labels
X1 mailings have labels and a notepad
G1 mailings have labels and a notepad

This information could certainly be used to calculate
several summary variables that count the number of
occurrences of various types of promotions received
in the most recent 12-36 months, etc.

RFA (RECENCY/FREQUENCY/AMOUNT)
------------------------------

The RFA (recency/frequency/amount) status of the
donors (as of the promotion dates) is included in the
RFA fields.

The (current) concatenated version is a nominal
or symbolic field. The individual bytes could
separately be used as fields and refer to the
following:

First Byte of code is concerned with RECENCY
based on Date of the last Gift

F=FIRST TIME DONOR Anyone who has made their
  first donation in the last 6 months and has
  made just one donation.

N=NEW DONOR Anyone who has made their first
  donation in the last 12 months and is not a
  First time donor.  This is everyone who made
  their first donation 7-12 months ago, or
  people who made their first donation between
  0-6 months ago and have made 2 or more
  donations.

A=ACTIVE DONOR Anyone who made their first
  donation more than 12 months ago and has made
  a donation in the last 12 months.
```

L=LAPSING DONOR A previous donor who made their last donation between 13-24 months ago.

I=INACTIVE DONOR A previous donor who has not made a donation in the last 24 months. It is people who made a donation 25+ months ago.

S=STAR DONOR STAR Donors are individuals who have given to 3 consecutive card mailings.

Second Byte of code is concerned with FREQUENCY based on the period of recency. The period of recency for all groups except L and I is the last 12 months. For L it is 13-24 months ago, and for I it is 25-36 months ago. There are four valid frequency codes.

1=One gift in the period of recency
2=Two gift in the period of recency
3=Three gifts in the period of recency
4=Four or more gifts in the period of recency

Third byte of the code is the Amount of the last gift.

A=$0.01 - $1.99
B=$2.00 - $2.99
C=$3.00 - $4.99
D=$5.00 - $9.99
E=$10.00 - $14.99
F=$15.00 - $24.99
G=$25.00 and above

ADATE_2	Date the 97NK promotion was mailed
ADATE_3	Date the 96NK promotion was mailed
ADATE_4	Date the 96TK promotion was mailed
ADATE_5	Date the 96SK promotion was mailed
ADATE_6	Date the 96LL promotion was mailed
ADATE_7	Date the 96G1 promotion was mailed
ADATE_8	Date the 96GK promotion was mailed
ADATE_9	Date the 96CC promotion was mailed
ADATE_10	Date the 96WL promotion was mailed
ADATE_11	Date the 96X1 promotion was mailed
ADATE_12	Date the 96XK promotion was mailed

ADATE_13	Date the 95FS promotion was mailed
ADATE_14	Date the 95NK promotion was mailed
ADATE_15	Date the 95TK promotion was mailed
ADATE_16	Date the 95LL promotion was mailed
ADATE_17	Date the 95G1 promotion was mailed
ADATE_18	Date the 95GK promotion was mailed
ADATE_19	Date the 95CC promotion was mailed
ADATE_20	Date the 95WL promotion was mailed
ADATE_21	Date the 95X1 promotion was mailed
ADATE_22	Date the 95XK promotion was mailed
ADATE_23	Date the 94FS promotion was mailed
ADATE_24	Date the 94NK promotion was mailed
RFA_2	Donor's RFA status as of 97NK promotion date
RFA_3	Donor's RFA status as of 96NK promotion date
RFA_4	Donor's RFA status as of 96TK promotion date
RFA_5	Donor's RFA status as of 96SK promotion date
RFA_6	Donor's RFA status as of 96LL promotion date
RFA_7	Donor's RFA status as of 96G1 promotion date
RFA_8	Donor's RFA status as of 96GK promotion date
RFA_9	Donor's RFA status as of 96CC promotion date
RFA_10	Donor's RFA status as of 96WL promotion date
RFA_11	Donor's RFA status as of 96X1 promotion date
RFA_12	Donor's RFA status as of 96XK promotion date
RFA_13	Donor's RFA status as of 95FS promotion date
RFA_14	Donor's RFA status as of 95NK promotion date
RFA_15	Donor's RFA status as of 95TK promotion date
RFA_16	Donor's RFA status as of 95LL promotion date
RFA_17	Donor's RFA status as of 95G1 promotion date
RFA_18	Donor's RFA status as of 95GK promotion date
RFA_19	Donor's RFA status as of 95CC promotion date
RFA_20	Donor's RFA status as of 95WL promotion date
RFA_21	Donor's RFA status as of 95X1 promotion date
RFA_22	Donor's RFA status as of 95XK promotion date
RFA_23	Donor's RFA status as of 94FS promotion date
RFA_24	Donor's RFA status as of 94NK promotion date
---	---
	The following fields are summary variables from the promotion history file.
CARDPROM	Lifetime number of card promotions received to date. Card promotions are promotion type FS, GK, TK, SK, NK, XK, UF, UU.
MAXADATE	Date of the most recent promotion received (in YYMM, Year/Month format)
NUMPROM	Lifetime number of promotions received to date

CARDPM12	Number of card promotions received in the last 12 months (in terms of calendar months translates into 9603-9702)
NUMPRM12	Number of promotions received in the last 12 months (in terms of calendar months translates into 9603-9702)
-------------------------	--
	The following fields are from the giving history file.
RDATE_3	Date the gift was received for 96NK
RDATE_4	Date the gift was received for 96TK
RDATE_5	Date the gift was received for 96SK
RDATE_6	Date the gift was received for 96LL
RDATE_7	Date the gift was received for 96G1
RDATE_8	Date the gift was received for 96GK
RDATE_9	Date the gift was received for 96CC
RDATE_10	Date the gift was received for 96WL
RDATE_11	Date the gift was received for 96X1
RDATE_12	Date the gift was received for 96XK
RDATE_13	Date the gift was received for 95FS
RDATE_14	Date the gift was received for 95NK
RDATE_15	Date the gift was received for 95TK
RDATE_16	Date the gift was received for 95LL
RDATE_17	Date the gift was received for 95G1
RDATE_18	Date the gift was received for 95GK
RDATE_19	Date the gift was received for 95CC
RDATE_20	Date the gift was received for 95WL
RDATE_21	Date the gift was received for 95X1
RDATE_22	Date the gift was received for 95XK
RDATE_23	Date the gift was received for 94FS
RDATE_24	Date the gift was received for 94NK
RAMNT_3	Dollar amount of the gift for 96NK
RAMNT_4	Dollar amount of the gift for 96TK
RAMNT_5	Dollar amount of the gift for 96SK
RAMNT_6	Dollar amount of the gift for 96LL
RAMNT_7	Dollar amount of the gift for 96G1
RAMNT_8	Dollar amount of the gift for 96GK
RAMNT_9	Dollar amount of the gift for 96CC
RAMNT_10	Dollar amount of the gift for 96WL
RAMNT_11	Dollar amount of the gift for 96X1
RAMNT_12	Dollar amount of the gift for 96XK
RAMNT_13	Dollar amount of the gift for 95FS
RAMNT_14	Dollar amount of the gift for 95NK
RAMNT_15	Dollar amount of the gift for 95TK

RAMNT_16	Dollar amount of the gift for 95LL
RAMNT_17	Dollar amount of the gift for 95G1
RAMNT_18	Dollar amount of the gift for 95GK
RAMNT_19	Dollar amount of the gift for 95CC
RAMNT_20	Dollar amount of the gift for 95WL
RAMNT_21	Dollar amount of the gift for 95X1
RAMNT_22	Dollar amount of the gift for 95XK
RAMNT_23	Dollar amount of the gift for 94FS
RAMNT_24	Dollar amount of the gift for 94NK
	The following fields are summary variables from the giving history file.
RAMNTALL	Dollar amount of lifetime gifts to date
NGIFTALL	Number of lifetime gifts to date
CARDGIFT	Number of lifetime gifts to card promotions to date
MINRAMNT	Dollar amount of smallest gift to date
MINRDATE	Date associated with the smallest gift to date
MAXRAMNT	Dollar amount of largest gift to date
MAXRDATE	Date associated with the largest gift to date
LASTGIFT	Dollar amount of most recent gift
LASTDATE	Date associated with the most recent gift
FISTDATE	Date of first gift
NEXTDATE	Date of second gift
TIMELAG	Number of months between first and second gift
AVGGIFT	Average dollar amount of gifts to date
CONTROLN	Control number (unique record identifier)
TARGET_B	Target Variable: Binary Indicator for Response to 97NK Mailing
TARGET_D	Target Variable: Donation Amount (in $) associated with the Response to 97NK Mailing
HPHONE_D	Indicator for presence of a published home phone number
	(See the section on RFA for the meaning of the codes)
RFA_2R	Recency code for RFA_2
RFA_2F	Frequency code for RFA_2
RFA_2A	Donation Amount code for RFA_2
MDMAUD_R	Recency code for MDMAUD

```
MDMAUD_F                    Frequecy code for MDMAUD
MDMAUD_A                    Donation Amount code for MDMAUD

--------------------------  ------------------------------------------------

CLUSTER2                    Classic Cluster Code (a nominal symbolic field)
GEOCODE2                    County Size Code

=========================================================================
EPSILON CONFIDENTIAL      EPSILON CONFIDENTIAL    EPSILON CONFIDENTIAL

    INFORMATION LISTED BELOW IS AVAILABLE UNDER THE TERMS OF THE
                      CONFIDENTIALITY AGREEMENT

EPSILON CONFIDENTIAL      EPSILON CONFIDENTIAL    EPSILON CONFIDENTIAL
=========================================================================
```