UDACITY CAPSTONE PROJECT
Kaggle: Santander Customer Satisfaction

Juho Salminen

Problem

The problem for this capstone project comes from Kaggle machine learning competition site, https://www.kaggle.com/c/santander-customer-satisfaction. Consumer bank Santander wants to predict which customers are satisfied and which are not, based on a information they have about the customers. The problem is important, because unsatisfied customers will not use the services as much as satisfied customers, will not recommend the company to their friends, and may change service provider, causing losses to the company. Furthermore, they usually do not voice their concerns, making it difficult for the companies to respond. Solving this problem would also benefit the customers. When Santander is able to identify customers that are not satisfied with its services, it can intervene and try to fix the issues from which the unsatisfied customers are suffering. The task is to predict if a customer is satisfied or dissatisfied with their banking experience, based on several hundred anonymized features.

Solution

As a solution to the problem described above, a machine learning algorithm needs to be implemented that can can predict customer satisfaction based on other information Santander has about them. In other words, the task is to classify customers to two groups. Most customers (roughly 96 %) are satisfied with their experience, which complicates the evaluation slightly, and makes the choice of metric even more important. If for instance correct predictions is used as a measurement, even a naive classifier that classifies all cases in one class will achieve impressive-looking 96 % accuracy. However, the selection of measurement is simplified, as Kaggle provides a measurement they use to evaluate competition submissions. The measurement is Area Under Receiver Operating Charasteristic Curve (AUROC)

Receiver operating characteristic (ROC) curve is a plot that illustrates the performance of a binary classifier as its discrimination threshold is varied. The curve is created by plotting the true positive rate against the false positive rate (FPR) at different threshold values.

The true positive rate measures how many correct positive results occur among all positive samples available during the test, while the false positive rate defines how many incorrect positive results occur among all negative samples available during the test.

$$TPR = \frac{\sum true\ positives}{\sum condition\ positive}$$

$$FPR = \frac{\sum true\ negative}{\sum condition\ negative}$$

A ROC space is defined by FPR and TPR as coordinate axes, and it shows the relative trade-off between true and false positives. The best ideal prediction would yield a point in the upper left corner of the ROC space, representing 100% sensitivity (no false negatives) and 100% specificity (no false positives). A random guess gives a point along a diagonal line from the left bottom to the top right corners. Fittingly to current case, this result holds regardless of the base rates of positive and negative cases. Area under curve, or AUROC), is then the probability that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative instance. A random guess or all zeros could be used as a baseline prediction. Kaggle uses all zeroes in this case, giving AUROC score of 0.50. The machine learning algorithm should do at least this well to be of any use for Santader in predicting customer satisfaction.

Analyzing the problem

The dataset provided by Kaggle consists of training and test datasets. Training set is used for training the algorithm, and test set is reserved for creating a submission to the competition. The training set consists of anonymized data on 76020 Santander customers. In addition to ID and TARGET variables there are 369 variables, with no missing values. Information on what the variables represent is not available. 127 variables are continuous and 242 are integer variables, possibly representing categorical data. Such a large number of variables makes visual exploration challenging; R was used for initial exploration of all variables, but here only the main findings are reported.

The measurement variable is provided by Kaggle, and it is called TARGET. This variable is either 0 or 1 depending on if the customer is satisfied or unsatisfied. A quick visual exploration of distributions shows that there are not major differences between the distributions of variables whether the TARGET is one or zero. Therefore, the difference between satisfied and unsatisfied customers is likely an interaction between two or more variables. No single feature sticks out as particularly promising for predicting customer satisfaction. 244 variables are integers and 127 are continuous. The integer variables might represent categorical data, which make warrant transformation to dummy variables.

Suitable machine learning algorithms should predict customer satisfaction classification (binary variable) given other information on the customer. Many of the variables are categorical. Therefore, any algorithm that can classify observations in two groups and can deal with categorical variables could potentially be used. The range of applicable algorithms is thus huge, from variations of linear regression to support vector machines, random forests, Bayesian approaches, and even neural networks or “deep learning”. As it seems there are no simple relationships between single variables and target classification, the chosen algorithm should take into account the interactions and possible non-linearities. Possible algorithms fitting these criteria include random forest, support vector machine and gradient boosting. Random forest is known as a well-performing “black box” algorithm. It is not easy to understand what is going on between the variables, but prediction performance is often good. Random forest can also deal with non-linearities and interactions between the variables. Support vector machine are also suitable for non-linearities, and it was considered to be one of the state-of-the art algorithms before the emergence of deep learning. Running time with the amount of data at hand might be an issue, though. Gradient boosting uses somewhat similar approach to random forest, and has performed well on Kaggle competitions in the past.

Data provided by Kaggle has already been transformed to machine-learning friendly format. There are no missing values, but some of the variables contain apparent errors or have always the same value. In one case there were only a few suspicious values, and they were imputed with median value. Other problematic variables were removed. Possible categorical variables are stored as integers and these might need converting. There is likely a lot of irrelevant information in the dataset. A subset of data may need to be selected for machine learning purposes. It is not clear what variables are the most suitable, but based on the visual exploration there is likely lots of correlation between several variables. Feature reduction is likely necessary, for instance by principal component analysis.

Implementing a solution

Data preprocessing included imputations on one variable and removal of several other, as described in table 1. A decision was made to initially use random forest algorithm, which makes it unnecessary to scale or normalize the features. Random forest can also be used for feature selection by plotting feature importances, which could be useful for improving performance. These considerations make random forest a good first-pass solution. Data was split to training and test sets, with 30 % of data retained for testing.

Variable	**Preprocessing**	**Justification**
var3	Values less than zero imputed with median	Only few values are less than zero, and they all are -999999. Likely an error in data, and median is a good guess.
Variables with zero variance	Removed	Do not contain information that helps discriminate between the cases.
(Duplicate columns. None left after previous steps.)	Removed	Contain only redundant information.

Table 1. Data preprocessing.

As a first-pass solution almost-default random forest from scikit-learn Python package was thus implemented. Default parameters were used, apart from setting number of estimators to 100 (more typical than default 10), and pre-specifying the initial random state for repeatability. AUROC score on the test set of the first-pass solution was not impressive at 0.520. Plotting feature importances of random forest showed that only a few features had large contributions to prediction performance.

As a second pass solution only the top 20 most important features were selected. Additionally, the random forest algorithm was set to use balanced class weights. The balanced mode adjust class weights automatically inversely proportional to class frequencies, which is helpful when the class frequencies are very different, as is the case here. Other parameters were kept as before. As a result, performance on both training and test sets improved. Now the AUROC score on test set was 0.542.

Next dimension reduction with principal component analysis (PCA) was implemented. Plotting the cumulative rate of unexplained variance for the most useful components showed that already the first 15 components contain in practice all variance in the dataset. The data was thus transformed

using PCA to have only 15 dimension, and a new random forest was trained. Performance on test set improved again. AUROC score on the test set was now 0.550. Also the feature importance plot looks now better: importance is more equally distributed, and two least important features have practically zero importance. It appears that PCA does a better job at extracting information from the dataset than merely selecting the most useful raw features.

I tried to improve the algorithm's performance by parameter optimization. Grid search was used to find best values for number of features to consider at each split of trees, maximum depth of trees, and number of trees to train. To shorten the training time, first maximum number of features and maximum depth of trees is optimized, and then a suitable number of trees was searched for using the optimized parameters from the previous step. These optimization steps did not yield significant improvements. Slight variation in performance is likely due to random effects.

Finally, a random forest with optimal parameters (maximum depth of trees: unlimited, number of features to consider: 8, number of estimators: 100) was trained on full training data set provided by Kaggle. This model was then used to make predictions on Kaggle's test set, and the resulting predictions were then submitted to competition on Kaggle's website. The submission scored 0.827418 (ranking 3158/4609 as of 22 April 2016) on the public leaderboard for Kaggle Santander competition. The score is well over all zeros benchmark at 0.50, and only slightly behind the top score 0.843972.

Reflection

The biggest challenge for me in this project was the large number of features. It made it difficult to explore or select features manually. Random Forest turned out to be a good algorithm also for exploration, as it can produce feature importances and thus identify promising variables. Another confusing thing was the AUROC score. I have read somewhere that AUROC has a reputation fo being sometimes unstable, and it seemed to be the case here. For some reason I got constantly low scores barely above the 0.50 benchmark, but on Kaggle's public leaderboard the performance was much better at 0.827. I considered trying out other algorithms, but based on the public leaderboard at Kaggle it seemed there was not much space for improvement. In the end, the achieved AUROC score is a significant improvement on random guessing or setting all values to zeros. The implemented random forest algorithm could thus help Sandander to find unsatisfied customers, which was the goal of this project.