# Task 2: Brand Preference Prediction

This informal report contains a brief overview of the procedures and the results obtained in this second task. In this task I began by pre-processing the data and developing an exploratory data analysis to comprehend the data.

The objective of this task is to train a model on a train dataset and apply the said model on a different dataset to generate predictions. Briefly, we are intended to create a model capable of forecasting the computer brand preference out of Acer and Sony (I converted the original integer column into categorical to force caret to forecast a nominal target). As can be seen below, this nominal variable is slightly unbalanced:

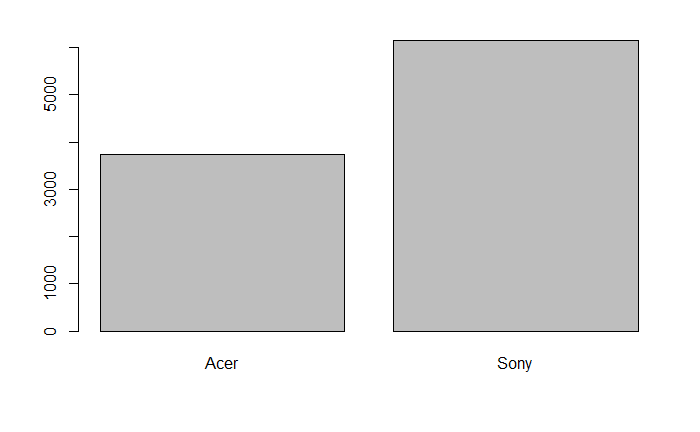


Figure 1 - Target count distribution

Both datasets contain the same information, and the behavior of each column is similar in both the datasets.

In order to forecast the target, a dataset with the following features has been provided:

1. **Salary:** This numerical variable seemed to be the most informative feature according to the boxplots created in the R notebook;
2. **Age:** This demographical numerical feature does not seem to be so relevant as the salary, but according to the boxplots, it could be important;
3. **Education:** This categorical feature does not seem to be relevant towards predicting this target;
4. **Car Brand:** This categorical feature does not seem to be relevant towards predicting this target;
5. **Zip Code:** This categorical feature does not seem to be relevant towards predicting this target;
6. **Credit:** This numerical feature does not seem to be relevant towards predicting this target;

In this exercise I was asked to develop two models, one based on Stochastic Gradient Boosting and another one based on Random Forest. Before training each model I simply divided the pre-processed dataset in train and test sets (the latter contained 25% of the data). Now, I will discuss each model individually below.

## Model 1: Stochastic Gradient Boosting Classifier:

In this first model, I utilized 10 fold cross-validation with an automatic tuning grid (automatic selection of hyper parameters) with length size 10, thus I compared the behaviour of 100 models with different hyper parameters.

It’s worth adding that I chose the Kappa as the optimization metric instead of the accuracy because this metric is more indicated for unbalanced datasets. After training I ran the model on the test dataset and obtained the following metric results:

|  |  |
| --- | --- |
| **Accuracy** | 0.9325 |
| **Kappa** | 0.8563 |

This model provides an accuracy of approximately 94% which is pretty high. The Kappa or Cohen's Kappa is at around 86% which is quite impressive since this metric is normalized at the baseline of random chance on our dataset. This dataset has a slightly unbalanced target, therefore the Kappa value is quite different from the accuracy value. It's worth adding that if the model was trained to optimize the accuracy, the metrics accuracy and Kappa obtained would be slightly lower.

Note that many other metrics have been calculated and commented in the R notebook. Anyway, I have omitted the many other metrics from this brief summary to avoid elongating this report.

For this model the overall relative importance of the top 3 most relevant features were:

|  |  |
| --- | --- |
| **Salary** | 100.0% |
| **Age** | 84.3% |
| **Credit** | 0.56% |

It seems that this model would perform fairly well if it only had access to the Salary and Age of the customer, the relative importance of the remaining features is much lower when forecasting the target.

## Model 2: Random Forest Classifier:

In this second model, I utilized 10 fold cross-validation with a manually tuned grid of length size 5 on the *mtry* (number of variables randomly sampled as candidates at each split) parameter.

Once again I chose the Kappa as the optimization metric instead of the accuracy because this metric is more indicated for unbalanced datasets. After training I ran the model on the test dataset and obtained the following metric results for the model:

|  |  |
| --- | --- |
| **Accuracy** | 0.9329 |
| **Kappa** | 0.8568 |

This model provides an accuracy of approximately 94% which is pretty high. The Kappa or Cohen's Kappa is at around 86% which is quite impressive since this metric is normalized at the baseline of random chance on our dataset. This dataset has a slightly unbalanced target, therefore the Kappa value is quite different from the accuracy value. It's worth adding that if the model was trained to optimize the accuracy, the metrics accuracy and Kappa obtained would be slightly lower.

Note that many other metrics have been calculated and commented in the R notebook. Anyway, I have omitted the many other metrics from this brief summary to avoid elongating this report.

For this model the overall relative importance of the top 3 most relevant features were:

|  |  |
| --- | --- |
| **Salary** | 100.0% |
| **Age** | 55.2% |
| **Credit** | 12.5% |

Contrarily to the previous model, this one relies more on the credit features. Anyhow, it seems that this model would perform fairly well if it only had access to the Salary and Age of the customer, the importance of the credit variable is still clearly lower than the top features according to importance.

## Model Comparison

Having created two models, it is now time to pick the one with better performance in order to create the predictions in the previously untouched dataset.

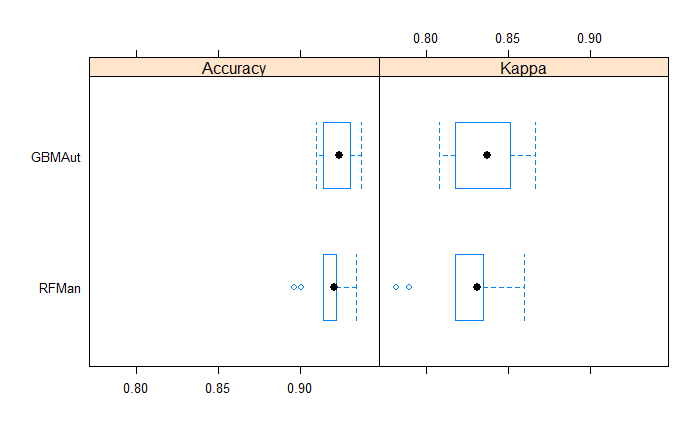


Figure 2 - Boxplot of Accuracy and Kappa for GBM and RF models

Looking at the table and boxplot of the chunk above it seems that both algorithms have a good performance in forecasting the target of this problem. However, the Stochastic Gradient Boost seems to provide a slightly better performing model as the interquartile range for both the accuracy and kappa falls is located at a higher value.

Below I have represented the predicted distribution a percent stacked barchart of the computer brand preference in the predictions and in the train dataset:

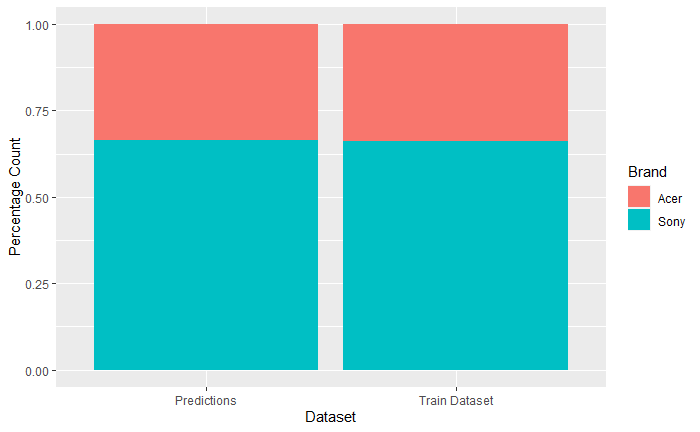


Figure 3 - Percent stacked barchart of Computer Brand for train and predictions

In the figure above I have created a chart that shows the preferences for Sony and Acer for the entire 15,000 existing customer survey. The *predictions* dataset contain the proportion of predicted computer brands on the incomplete survey and the *train dataset* corresponds to the actual proportion on the complete responses. It can be seen that this proportion is pretty much the same for both datasets, meaning that the test and train datasets contain clients with similar behaviors.