Advanced Machine Learning, Project 2 - Report

Julia Kaznowska, Jan Smoleń, Filip Szympliński

June 2024

1 Introduction

In project 2 we were tasked with creating high-accuracy, parsimonious model. We decided to try three feature selection techniques: Recursive Feature Elimination (RFE), SelectKBest and Boruta. Then, as those techniques are choosing all relevant features, we checked whether deleting correlated variables before selecting features yielded better results. We also paired Boruta with Sequential Feature Selection in order to try to achieve minimal optimal subset of features.

2 Approaches

For the purpose of evaluating chosen methods in context of the given task, we have created a function that aims to estimate the reward as defined in the project description. Logic behind it is as follows: use a model that can predict probability on evaluation set. Then, mark $\frac{1}{5}$ of the clients as potential customers, using clients with the highest returned probability. Multiply the number of correctly chosen customers by a coefficient equal to ratio of final test set size (5000) and evaluation set size. Finally, calculate the reward based on the formula from the task. This scaling is crucial to mimic trade-off presented in the final reward function.

Decorellation

One of the potential strategies to be used was to test whether deleting correlated features changed the results of feature selection. After examining Pearson correlation coefficient between variables, we have identified two groups of correlated variables. They are shown on Figure 1. Any pair of variables for which Pearson correlation coefficient was higher or equal to 0.8 was considered as highly correlated. During decorellation, features with indexes $1, 3, 4, 5, 6, 7, 8, 9$ (counting from 0) were removed.

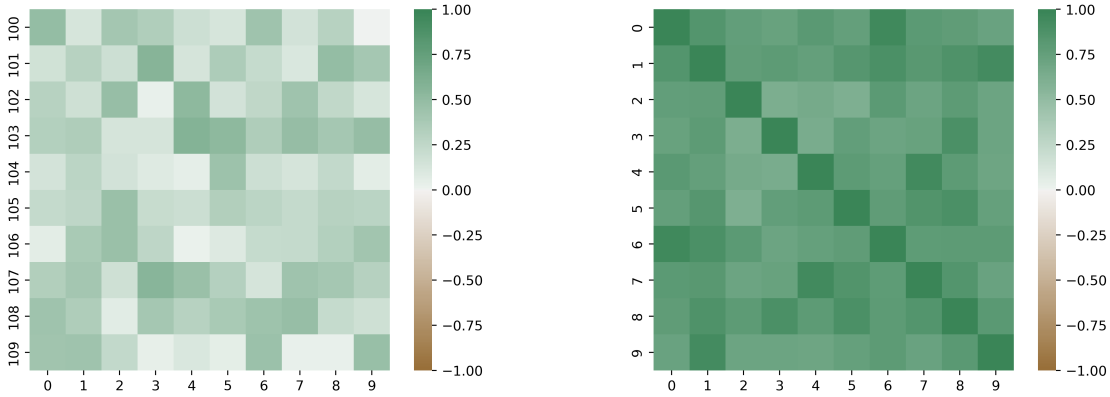

Figure 1: Two groups of features with significant correlation

Boruta + SFS

Another strategy was to divide feature selection process into two parts: first, we determine all relevant features by using Boruta algorithm. Then, we run Sequential Features Selection (SFS) using only obtained relevant variables. After each iteration, we use our function to estimate the reward on the final dataset, storing subset chosen at each step. For the feature selection process, we used Random Forest Classifier. We set the maximum depth to 2, following our

intuition that shallow trees may need smaller number of features to perform well. As for a model used for evaluating performance at each SFS step, any model that can predict probability can be used.

3 Experiments

Every experiment was repeated 5 times on different train-test splits, as to check the stability of the methods. The size of the split was 80:20. We tested two types of models: Random Forest and XGBoost.

Recursive Feature Elimination

As shown in the Figure 2, deleting correlated features on average brought better results. XGBoost reached higher score than Random Forest.

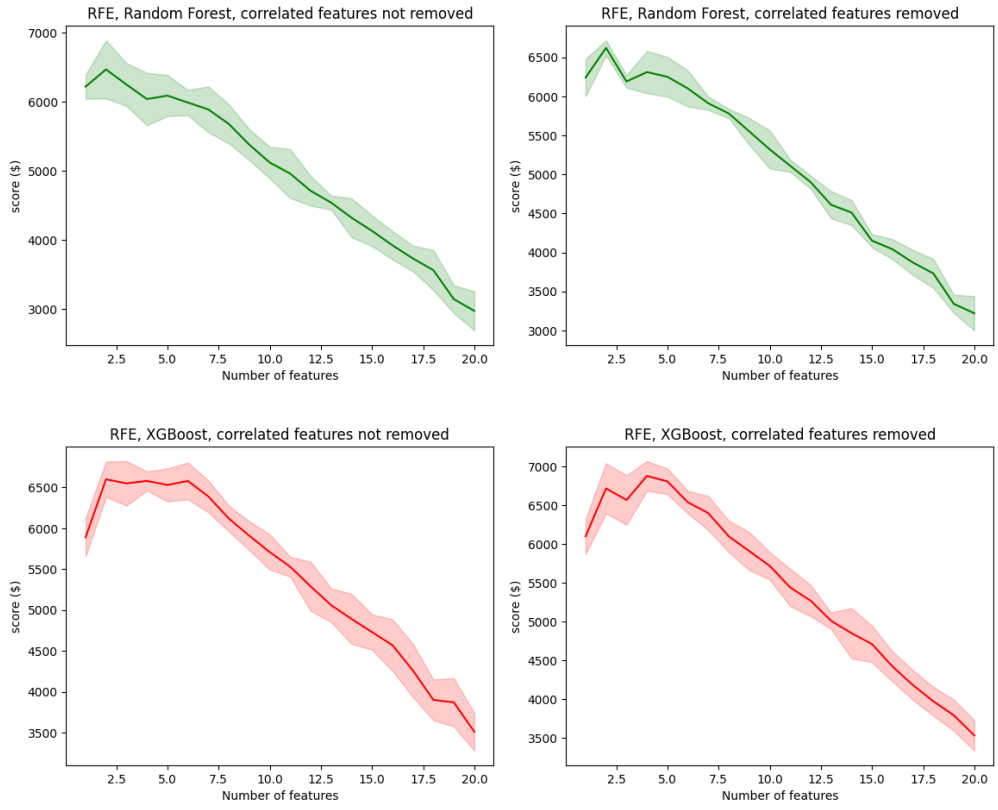

Figure 2: The score change for RFE depending on the number of features

SelectKBest

From Figure 3 we can draw a conclusion that removing correlated features for SelectKBest method causes the standard deviation to become smaller. Interestingly, it didn't change mean scores too much.

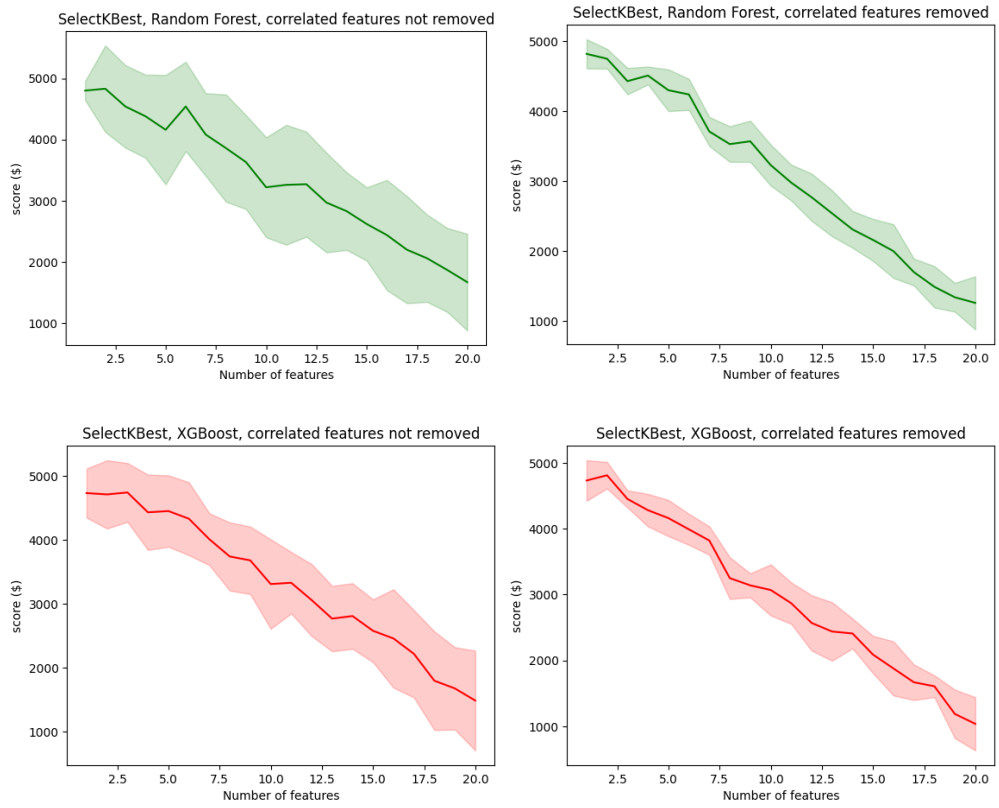

Figure 3: The score change for SelectKBest depending on the number of features

Boruta + SFS

Our first step was to evaluate only Boruta algorithm. It turned out that it did not bring the best results - the estimated score for the full dataset was 3780 ± 200, whereas after decorrelation: 5500 ± 370. That's why we decided on adding SFS in order to delete more features and obtain a better score. Although this strategy gave comparable results to RFE, they were characterized by a high standard deviation (see Figure 4), especially for lower number of features.

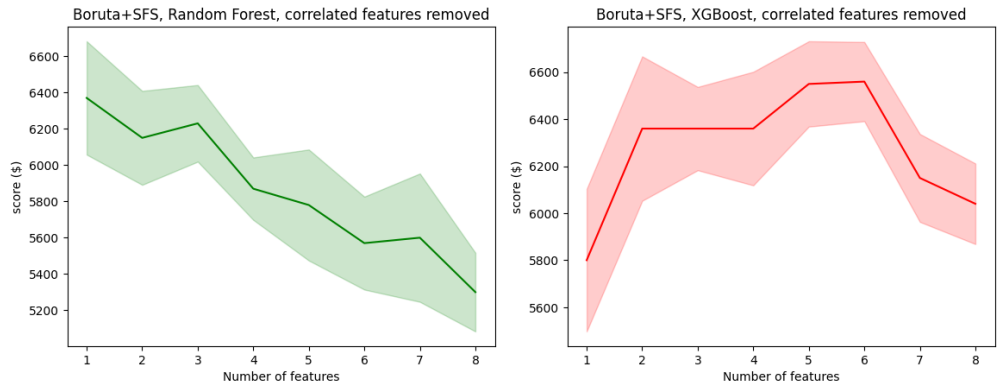

Figure 4: The score change for Boruta + SFS depending on the number of features

Summary

All the results are gathered in the tables below. Table 1 presents scores for Random Forest whereas Table 2 - for XGBoost. The best results for each method are in bold.

Table 1: Mean score [$] for Random Forest

Features no.	Decorrelation + RFE	RFE	Decorrelation + SelectKBest	SelectKBest	Decorrelation + Boruta + SFS
1	6240 ± 240	6220 ± 170	**4820** ± 210	4800 ± 160	**6370** ± 310
2	**6620** ± 100	**6470** ± 420	4750 ± 140	**4830** ± 710	6150 ± 260
3	6190 ± 80	6250 ± 310	4430 ± 190	4540 ± 670	6230 ± 210
4	6310 ± 270	6040 ± 380	4510 ± 130	4380 ± 680	5870 ± 170
5	6250 ± 250	6090 ± 300	4300 ± 300	4160 ± 890	5780 ± 300
6	6100 ± 230	5990 ± 180	4240 ± 220	4540 ± 730	5570 ± 260

Table 1 shows that SelectKBest had the worst scores of all the methods - with of without decorrelation. RFE with correlated features deleted performed the best. It turns out that for every method the best score is achieved by choosing only 1 or 2 features.

Table 2: Mean score [$] for XGBoost

Features no.	Decorrelation + RFE	RFE	Decorrelation + SelectKBest	SelectKBest	Decorrelation + Boruta + SFS
1	6100 ± 230	5890 ± 240	4730 ± 310	4730 ± 380	5800 ± 300
2	6720 ± 320	**6600** ± 220	**4810** ± 200	4710 ± 530	6360 ± 310
3	6570 ± 320	6550 ± 270	4450 ± 130	**4740** ± 460	6360 ± 180
4	**6880** ± 190	6580 ± 120	4280 ± 250	4430 ± 590	6360 ± 240
5	6810 ± 170	6530 ± 200	4160 ± 270	4450 ± 560	6550 ± 180
6	6540 ± 150	6580 ± 220	3990 ± 240	4330 ± 570	**6560** ± 170

From the Table 2 it can be seen that XGBoost yielded better results for high-scoring methods. The number of features bringing the best results was also higher for nearly all the methods. Similarly to Random Forest, SelectKBest performed the worst and RFE with no correlated features performed the best.

4 Final model

For all tested methods and models the best result was achieved by XGBoost with 4 features chosen by RFE after deleting correlated variables (Table 2). During experiments features with indices $101, 103, 106$ (indexing from 1) were always present as the top 3 features. In most cases variable 102 was chosen on the 4th place. This led to choosing columns with indices $101, 102, 103, 106$ for prediction. XGBoost with 100 estimators and max depth of 2 was the final classifier.

A potential flaw in our approach may be based in the fact that when creating final predictions, we use whole train set, but when estimating reward we have to limit ourselves to model based on subset (80%) of the training data. Therefore, our final model can perform better based on larger ammount of training data, which in turn will affect the trade-off between number of features and the scoring of the model. However, such problem would be very difficult to avoid.