# Advanced Machine Learning
# Project 2

Zuzanna Glinka, Nikola Miszalska, Malwina Wojewoda

June, 2024

## Contents

# 1 Exploratory data analysis

At the beginning of the task, we decided to familiarize ourselves with the data and try to draw from it some conclusions about the data that could influence later selection. To do this, we checked correlations and pairwise relationships between variables. We noticed that features 0-9 exhibit strong correlations among themselves, with coefficients around 0.8. What is more, features 100-109 display correlations with features 0-9, but weaker, approximately 0.3. In a later analysis, we found that in fact variables 100-105 are those with high predictive power. Nevertheless, the analysis provided limited insights into a solution, with further details available in the Appendix A.

# 2 Feature selection

We have tested several variable selection methods, including RFE, RFECV, Boruta, feature importance from random forest with different criterion functions, permutation importance for few models e.g. AdaBoost, XGBoost, Ridge among others.

To aggregate the results, we first extracted the importance coefficients of the selected variables from each method. For methods that only provided coefficients, we assigned them the same values of weights. Then we scaled these coefficients, so that the maximum value for each method was 1. After scaling, we averaged the importance values across all methods. If a variable was not selected by a particular method, its importance was considered as 0 for that method. These averaged scores for 20 best features are displayed on the bar chart shown on the Figure 1. In this report, we assume that the features are numbered starting from 1.

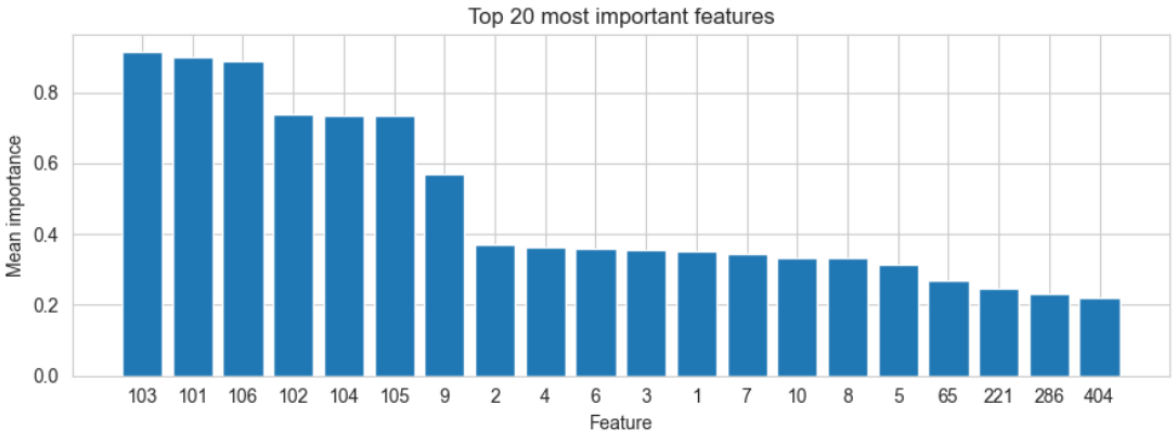


Figure 1: Weighted importance of features - top 20 most frequently selected variables.

With this approach, we were able to select a set of features that have high predictive potential, namely $(9, 101, 102, 103, 104, 105, 106)$. We have decided not to limit ourselves to a specific set, but in further steps test different combinations of these hyperparameters and choose different numbers of them, since we are aware that the actual predictive power may depend on the chosen model.

# 3 Altering dataset

A particularly important task in the context of maximizing the model's performance on the data was reducing the number of false positives. To achieve this, we implemented and tested the method discussed in the article focusing on this problem [1]. To minimize false positives, we retrained the model by adjusting the training data based on previous results. We took a partition of the false negatives and reclassified them as 0. This approach allowed the model to learn to identify these features with a bias towards class 0, thereby placing greater emphasis on minimizing false positives.

After a series of tests, it turned out that unfortunately, this method did not improve our results. Therefore, in subsequent experiments, we abandoned this approach.

# 4 Evaluation metric

Considering the specific method of prediction evaluation outlined in the task, we decided that this should be our primary metric for optimization. To achieve this, we defined a function as described in the Algorithm 1. It considers that company will pay €10 for each offer actually taken, and we must pay €200 for each used variable. The extra multiplier for taken offers compensates for the difference in size between the current smaller dataset and the future dataset used for testing, adjusting it accordingly. We label situations in which someone takes advantage of the offer as `1`, and those in which they do not as `0`.

This metric can range from -100 000 to 9 800. The lowest value occurs when the model includes all variables but has zero true positives. The highest value occurs when the model, using just one feature, achieves a precision of 1.

**Algorithm 1** Prediction Evaluation Function

**function** CUSTOM_EVALUATION_METRIC($probabilities, labels, numFeatures$)
    Sort the instances based on their probabilities in descending order.
    Select top 20% instances as predicted positives.
    Count how many of them have a label of 1, and call this *offerTakers*.
    Calculate the price:

$$price = 10 \times offerTakers \times \frac{5000}{\text{length}(labels)} - 200 \times numFeatures.$$

    **return** *price*
**end function**

# 5 Model selection

## 5.1 Initial model exploration

To begin with, we evaluated different classification models using their default settings to identify the most promising ones. This involved testing the models with different combinations of selected features. The results obtained from each model are presented in Figure 2 using a boxplot.

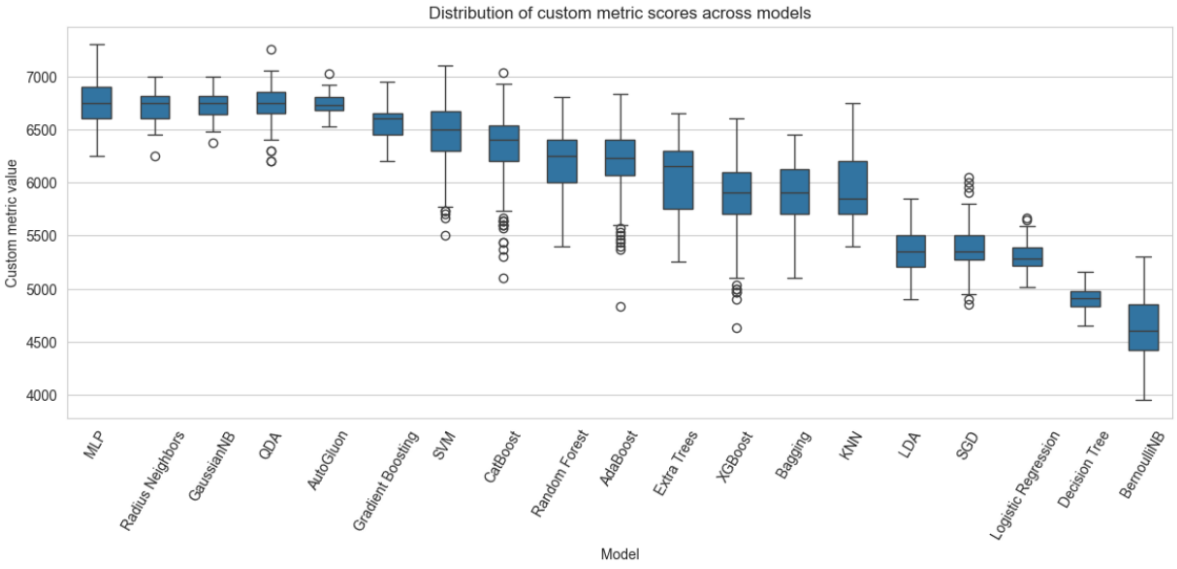


Figure 2: Distribution comparison of customized metric for base models.

Consequently, we narrowed down the number of models to 6, which we chose to concentrate on further. After selecting these models we tested each of them on every combination of 2 to 6-element subsets of features chosen during the preliminary feature selection stage described in Section 2. The aim of this search was to find, for each model, the subset of features that yields the best performance.

## 5.2 Tuning hyperparameters

As a next step, we performed hyperparameter search using both grid search and Optuna on the models chosen in the previous step, namely Multi-layer perceptron (MLP), Gradient Boosting, Radius Neighbors, Gaussian Naive Bayes, Quadratic Discriminant Analysis (QDA) and Support Vector Classifier (SVC).

We also decided to try AutoGluon package to see which approach gives better results and what models are choosing by this package. To evaluate models trained with AutoGluon we used our custom metric tailored to work with this tool.

We evaluated the results using 5-fold cross-validation. We opted for this approach because we were seeking a specific optimal solution, and the metric values varied significantly across different data splits. Choosing 5 folds was a compromise between obtaining an averaged solution and managing computational resources efficiently.

Table 1: Grid search hyperparameters.

| Classifier | Hyperparameter | Grid Search Space |
|---|---|---|
| **MLP** | hidden_layer_sizes | 1 to 4 layers, with 20 to 1000 neurons |
| | activation | relu, identity, tanh, logistic |
| | solver | sgd, adam, lbfgs |
| | alpha | 0.0001, 0.001, 0.01, 0.1 |
| | learning_rate | constant, adaptive, invscaling |
| | learning_rate_init | 0.001, 0.01, 0.1 |
| **Gradient Boosting** | n estimators | 100, 300 |
| | learning_rate | 0.01, 0.001 |
| | max_depth | 3,4,5,6,7 |
| | min_samples_split | 2,5,10 |
| | min_samples_leaf | 1,2,5 |
| | subsample | 0.5, 0.8 |
| **SVC** | C | from 0.001 to 150 |
| | gamma | from 0.001 to 1, scale, auto |
| | kernel | rbf, linear, sigmoid, poly |
| | degree | 1, 2, 3, 4, 5 |
| | coef0 | 0.05, 0.1, 0.3, 0.5 |
| **Radius Neighbors** | radius | 1.8, 2, 3, 5, 10 |
| | weights | uniform, distance |
| | algorithm | auto, ball_tree, kd_tree, brute |
| | leaf_size | 10, 20, 30, 40, 50 |
| | p | 1, 2 |
| **QDA** | reg_param | from 0.0 to 0.9, step: 0.1 |

## 5.3 Evaluation of the best models

Ultimately, we identified the models that yielded the best outcomes during 5-fold cross-validation. Additionally, we combined the models into ensemble using soft voting method. Table 2 presents used features, algorithm name, its hyperparameters and score on validation set for these selected solutions. To determine the most stable and high-performing model, we performed 5 rounds of 5-fold cross-validation on each split. The outcomes of this experiment are depicted in the Figure 3. The models

that participated in the soft voting, whose results are presented in this figure, are SVC, QDA, and Naive Bayes. These are the three models that yielded the highest averaged results after conducting 5-fold cross-validation five times.

Table 2: Optimal configurations for each model.

| Features | Model | Hyperparameters | Score |
|---|---|---|---|
| 102, 103, 104, 106 | SVC | `kernel`: poly, `degree`: 2, | 7330 |
| 102, 103, 104, 106 | QDA | `reg_param`: 0.5 | 7300 |
| 9, 101, 103 | MLP | `hidden_layer_sizes`: (50, 50, 50), `alpha`: 1e-05, `early_stopping`: True, `learning_rate`: adaptive | 6980 |
| 102, 103, 104, 106 | Gaussian NB | - | 6970 |
| 103, 104, 106 | AutoGluon (WeightedEnsemble_L2) | `presets`: good_quality, optimize_for_deployment | 6920 |
| 101, 102, 103 | Radius Neighbors | `radius`: 1.9, `weights`: distance, `p`: 1 | 6880 |

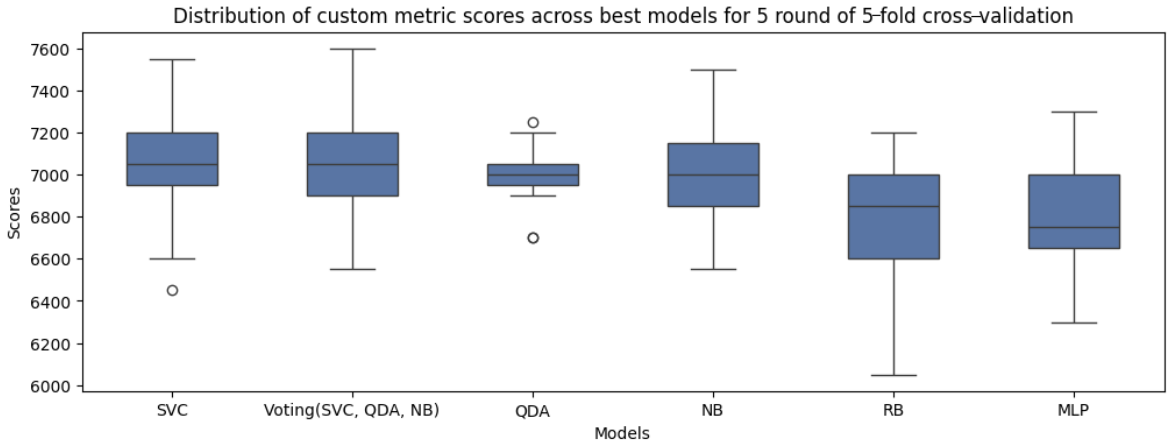


Figure 3: Distribution of results of the best configurations.

# 6 Final solution

Finally we chose single model SVC with hyperparameters: `kernel`: poly and `degree`: 2, trained on a data with selected features: 102, 103, 104, 106 (counting from 1). We decided to choose such configuration as it returned the best mean score on cross-validation with acceptable variation. To ensure the reproducibility of the result we set the `random_state` to 133.

At last we trained selected model on the entire x_train dataset and y_train labels, achieving a score of 7100 on the training data.

# 7 Conclusions

Arriving at the final solution required us to go through many steps and test various ideas. Through initial model analysis, testing different feature selection methods, and using our custom scoring metric, we identified an area for further, precise exploration to find the best solution. Tuning the hyperparameters of the top models and subsets of features highlighted several leading candidates for the final evaluation. While most models exhibited significant variance in their results, using 5-fold cross-validation allowed us to identify the final model, which provided the highest average score across all trials.

# References

[1] An emphasis on the minimization of false negatives/false positives in binary classification. *https://medium.com/an-emphasis-on-the-minimization-of-false-negatives-false-positives-in-binary-classification.*

# Appendix

## A Exploratory data analysis

Performed exploratory data analysis is quite limited, by the fact that the data was generated artificially and has no interpretation and any issues such as missing data.

### A.1 Correlations

To examine the relationships within the data, we employed both Pearson and Spearman correlation coefficients. Figure 4 illustrates a part of the correlation matrix generated from our analysis. Notably, features 0-9 exhibit strong correlations among themselves, with coefficients around 0.8. What is more, features 100-109 display correlations with features 0-9, but weaker, approximately 0.3. The figure displays only a subset of variables for better visibility. Other variables were not included here because their correlations were low, without any significant patterns.

We also examined the correlation with the target variable, which did not exceed 0.05 for any of the variables. Therefore, this correlation will not serve as an indicator for variable selection.

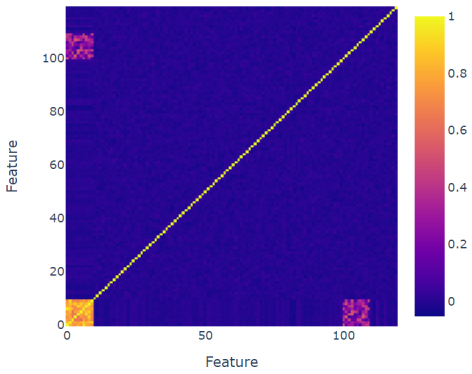


Figure 4: Correlation matrix of a subset of variables from 0 to 120.

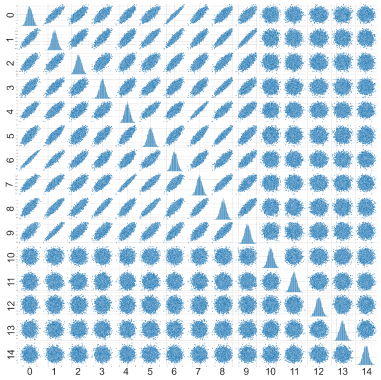


Figure 5: Pair plot of a subset of variables from 0 to 15.

### A.2 Feature distributions and their pairwise interactions

After initially examining the distributions, we analysed whether highly correlated variables exhibited any specific relationships with each other, and we also observed the overall distributions of the variables. We used a pair plot, depicted in the Figure 5, which included both highly correlated variables and those with minimal correlation. Our analysis of the histograms revealed that the distribution of each variable appears to be normal. Additionally, we checked the mean and variance of the variables and observed that although they varied, when rounded, they could be categorized into distinct groups, as detailed in the Table 3. Notably, variables 1-9 stood out from the rest due to their variance that cannot be easily characterized.

Table 3: Summary of rounded variable statistics

| Variable index | Mean | Standard deviation |
|---|---|---|
| 0-9 | 0 | from 1.6 to 2.2 |
| 10-199 | 0 | 1 |
| 200-399 | 0.5 | 0.3 |
| 400-499 | 10 | 4.5 |