
Warsaw University of Technology

# Feauture selection and predictive models for bank's marketing offer

*Advance Machine Learning*

authored by

Filip Kucia (335724)

Szymon Trochimiak (307459)

Michał Taczała (303775)

supervisor

Katarzyna Woźnica

Warsaw 2024

# 1 Methodology

Remark: most of the calculations and plots were created in Weights and Biases (Wandb). All indexes in the report are 1-based, however, all indexes in the code are 0-based.

## 1.1 Custom Evaluation Function

The function `calculate_gain`, defined below, serves as a custom evaluation function to assess the performance of a predictive model by calculating a metric known as "gain" expressed as expected profit in Euro €.

1. **Select Top Predictions:** Calculate the number of top predictions to consider ($k$), set to 20% of the total number of predictions $n$, thus $k = \lfloor 0.2n \rfloor$.

2. **Sort Predictions by Probability:** Let $A$ be a list of all predictions. Order these predictions by the predicted probabilities in descending order and save indices of top-$k$ predictions from $A$ to a list $B$, so that $\forall \left(\mathbb{Z}^+ \ni i < j \leqslant k\right) : a_{b_i} \geqslant a_{b_j}$.

3. **Calculate Gain:** Let $C$ be a list of ground truth values: 1 or 0. $C$ has the same size and order as $A$. Let $p$ be the number of features used for predictions. The gain is computed using the following formula:

$$\text{gain} = \frac{\sum_{i=1}^{k} c_{b_i}}{k} \times 10000 - p \times 200$$

# 2 Data Exploration

Based on the correlation matrix calculated for all features, we removed features 2 to 10 as they are all highly correlated with feature 1 (correlation $> 0.75$).

Here is a link to an interactive report with plots examining the density distribution of different features and comparison of these distributions between test and train datasets. Based on that we concluded that distributions from both datasets match.

Remark: We hypothesize that first 200 features are derived from a normal distribution based on visual analysis. Moreover, Jarque-Berra and Shapiro-Wilk Tests say, that there is

Table 1: Ranking of features by importance from different methods

| **Method/Feature** | **101** | **103** | **106** | **104** | **102** | **105** | **1** |
|---|---|---|---|---|---|---|---|
| **Random Forest Importance** | 0.01093 | 0.01019 | 0.00900 | 0.00808 | 0.00747 | 0.00651 | 0.00336 |
| **Optuna Importance Rank** | 1 | 2 | 3 | 4 | 5 | 6 | 12 |

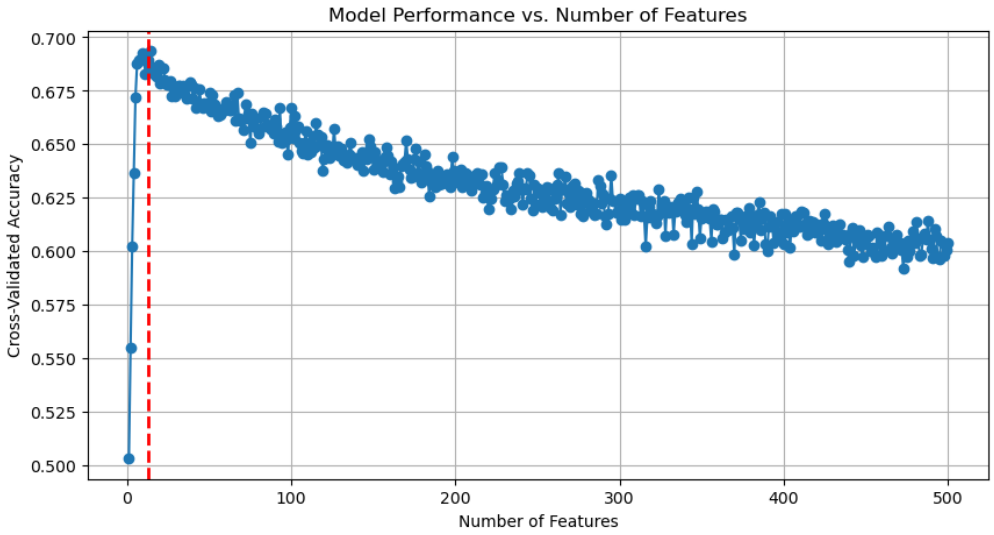
a high probability that the first 190 features are derived from a normal distribution because, for 190 of them, the p-values obtained are above the 0.05 confidence level.

# 3 Feauture Selection

We utilized several methods to optimize feature selection for a Random Forest classifier:

- **Optuna and Boruta-Enhanced Algorithm:** Optuna is an automatic hyperparameter optimization framework which searches through a high-dimensional space of possible hyperparameters to find the best configuration for a model in our case the Random Forest classifier. Optuna adjusts parameters, like the number of trees, depth of trees, and splitting criteria. It utilizes a Random Forest classifier to iteratively assess and eliminate less significant features based on permutation importance.
- **Random Forest Importance:** Features are evaluated for their ability to decrease impurity within the tree nodes, critical for effective data classification.
- **Iterative/Cross-Validation:** Employs cross-validation to measure model performance with different feature subsets, aiding in understanding model complexity versus performance.

The result of iterative and cross-validation approach is a plot that shows the relationship between the number of features used and the cross-validated accuracy of a RandomForest model. In the plot, a red dashed line marks 13 features, which is the point where the cross-validated accuracy is highest. Adding more features beyond this point worsens the accuracy.




### Feature Subset Selection

The final selection of features was based on a combined assessment of feature importance and model performance. Starting with an initial set of 500 features, we reduced our selection to a subset of features, specifically $\{1, 101, 102, 103, 104, 105, 106\}$. These features were identified as yielding the most favorable results and were therefore selected for further modeling.

## 4 Machine Learning Models used

| **Model** | **Parameters** |
|---|---|
| SVM | kernel: poly, degree: 2 |
| Random Forest Classifier | n_estimators: 100, max_depth: 5 |
| LDA - Standard | - |
| **LDA - Polynomial Features** | degree: 4, interaction_only: false |
| LDA - Spline Transform | degree: 4, knots: 5 |
| QDA | - |
| Naive Bayes | smoothing: 1e-9 |

Table 2: Summary of Machine Learning Models and Their Parameters

Table 3: Summary of model gains averaged for 100 runs for 0.2 and 0.5 train-test split.

| **Model** | **Avg 0.2** | **Std 0.2** | **Avg 0.5** | **Std 0.5** | **Features used** |
|---|---|---|---|---|---|
| **LDA Polynomial** | 7064 | 242.7 | 7011 | 119.7 | 102,103,104,106 |
| SVM | 7029 | 278.8 | 7047 | 122.1 | 102,103,104,106 |
| NB | 7021 | 264.5 | 7013 | 137 | 102,103,104,106 |
| QDA | 6914 | 272.7 | 6903 | 168.2 | 102,103,104 |
| LDA Spline | 6920 | 268.8 | 6835 | 142.1 | 101,102,103,106 |
| Random Forest | 6439 | 317.8 | 6366 | 200.5 | 1,103,104,106 |
| LDA | 5251 | 314.4 | 5205 | 231.4 | 1,103,106 |

### 4.1 Neural Networks

We experimented with simple fully connected neural networks of various sizes (1-5 hidden layers and 50-20000 parameters). Smaller networks displayed issues with learning stagnation and noisy updates, potentially due to insufficient model complexity or inappropriate hyper-parameters. Larger networks, on the other hand, tended to overfit the data.

### 4.2 KAN - Kolmogorov-Arnold Networks

KANs use a unique architecture that includes a layer of univariate functions followed by a linear combination of their outputs, theoretically capable of representing any multivariate continuous function with a single hidden layer.

## 5 Best model - submitted one

Our best-performed model is LDA Polynomial, which achieves an average gain from 100 runs of €7064. Although it uses 4 features which caused it to have €800 penalty, it outperforms other models. Moreover, this model is the most consistent one, as it achieves the smallest standard deviation from all tested models. For a single run, the maximum obtained gain is €8200.