# AML Project 2 – Cost-Aware Customer Selection

01.06.2025

Bruno Tobiasz, Mikołaj Trębski

## 1. Problem overview

The goal of this project is to identify a subset of electricity customers who should receive an energy-saving offer, based on past consumption patterns. The dataset contains 5,000 samples with 500 features and binary target variable. While the reward is €10 per correctly identified customer (true positive), each feature used in the final model incurs a cost of €200. The goal is to maximize profit.

## 2. Evaluation Metric

The score function used to evaluate each model is defined as:

**Score** = 10 × TP − 200 × #features

Where TP refers to true positives on the test set, and the number of features is the count of input variables used by the model after feature selection. This metric balances model complexity (cost) against the value of correct predictions.

## 3. Feature selection

To determine the best subset of features to train the models on, two methods from Scikit-Learn package had been used:

1. SelectKBest() that chooses specified number ***k*** of features with highest scores,
2. RandomForestClassifier() that allows to determine subset of features based on calculated importance higher than ***threshold***.

For both methods, different values of ***k*** and ***threshold*** were verified in terms of performance. Starting from bigger subsets (from up to 500) to smaller (up to 2), easy conclusion could be made, that for certain threshold (more than ~25 features), the performance of all models was degrading or just constant on less satisfying level.

Apart from lower accuracies, the scores from score function were dramatically decreasing with higher number of features. That's why only smaller subsets were taken into account for finding the optimal solution.

The barplot for the feature importances is presented beneath:

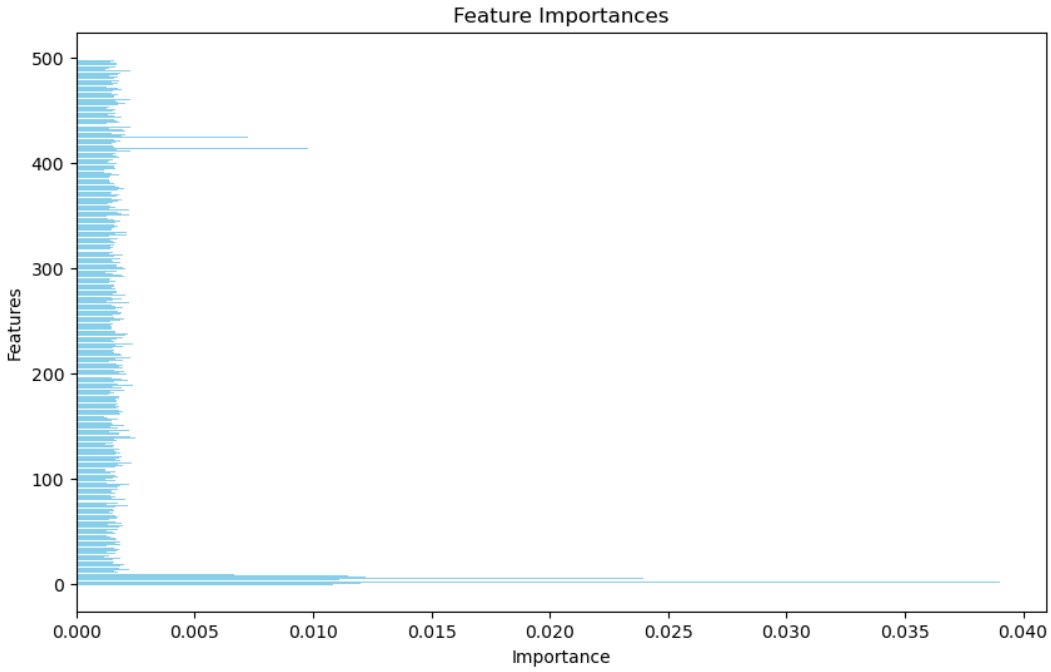


**Figure 1 Feature importances**

It is easily recognizable, that only a fraction (13 out of 500) of the features, had more significant importance e. j. higher than 0.003.

## 4. Models

To find the best possible model for the problem, 5 different algorithms had been tested with different combinations of parameters. All but AdaBoost were taken from Scikit-Learn library:

1. Logistic Regression
2. RandomForestClassifier,
3. AdaBoost,
4. GradientBoostingClassifier,
5. BaggingClassifier.

For LogisticRegression two hyperparameters were tested: C and penalty. For others: number of estimators, and maximum depth of tree.

## 5. Modeling Strategies

Five distinct strategies were evaluated, combining different classifiers with combinations of hyperparameters and feature selection techniques with various thresholds. To allow easier tests, each model had been checked individually.

The results, comprising of both scores and accuracies achieved by cross-validation on training set, allowed to find the model with best ratio between high score and high accuracy.

| Score | Accuracy | Model | Params | Feature_Selector | Num_Features |
|---|---|---|---|---|---|
| 4265.0 | 0.6918 | LogisticRegression | {'C': 0.01, 'penalty': 'l2'} | SelectKBest_k=2 | 2 |
| 4265.0 | 0.6918 | LogisticRegression | {'C': 0.01, 'penalty': 'l2'} | SelectFromModel_thresh=99.7th_percentile | 2 |
| 4195.0 | 0.6996 | LogisticRegression | {'C': 0.01, 'penalty': 'l1'} | SelectKBest_k=2 | 2 |
| 4195.0 | 0.6996 | LogisticRegression | {'C': 0.01, 'penalty': 'l1'} | SelectFromModel_thresh=99.7th_percentile | 2 |
| 4012.5 | 0.6892 | LogisticRegression | {'C': 0.01, 'penalty': 'l2'} | SelectKBest_k=3 | 3 |
| 4012.5 | 0.6892 | LogisticRegression | {'C': 0.01, 'penalty': 'l2'} | SelectFromModel_thresh=99.5th_percentile | 3 |
| 3995.0 | 0.6996 | LogisticRegression | {'C': 0.01, 'penalty': 'l1'} | SelectKBest_k=3 | 3 |
| 3995.0 | 0.6996 | LogisticRegression | {'C': 0.01, 'penalty': 'l1'} | SelectFromModel_thresh=99.5th_percentile | 3 |
| 3797.5 | 0.7014 | LogisticRegression | {'C': 0.01, 'penalty': 'l1'} | SelectKBest_k=4 | 4 |
| 3787.5 | 0.6986 | LogisticRegression | {'C': 0.01, 'penalty': 'l1'} | SelectFromModel_thresh=99.2th_percentile | 4 |
| 3722.5 | 0.6798 | LogisticRegression | {'C': 0.01, 'penalty': 'l2'} | SelectKBest_k=4 | 4 |
| 3717.5 | 0.6728 | LogisticRegression | {'C': 0.01, 'penalty': 'l2'} | SelectFromModel_thresh=99.2th_percentile | 4 |
| 3710.0 | 0.7076 | LogisticRegression | {'C': 0.1, 'penalty': 'l2'} | SelectKBest_k=2 | 2 |
| 3710.0 | 0.7076 | LogisticRegression | {'C': 0.1, 'penalty': 'l2'} | SelectFromModel_thresh=99.7th_percentile | 2 |
| 3652.5 | 0.7064 | LogisticRegression | {'C': 0.1, 'penalty': 'l1'} | SelectKBest_k=2 | 2 |
| 3652.5 | 0.7064 | LogisticRegression | {'C': 0.1, 'penalty': 'l1'} | SelectFromModel_thresh=99.7th_percentile | 2 |
| 3635.0 | 0.7062 | LogisticRegression | {'C': 1.0, 'penalty': 'l2'} | SelectFromModel_thresh=99.7th_percentile | 2 |
| 3635.0 | 0.7062 | LogisticRegression | {'C': 1.0, 'penalty': 'l2'} | SelectKBest_k=2 | 2 |
| 3627.5 | 0.7058 | LogisticRegression | {'C': 1.0, 'penalty': 'l1'} | SelectFromModel_thresh=99.7th_percentile | 2 |
| 3627.5 | 0.7058 | LogisticRegression | {'C': 1.0, 'penalty': 'l1'} | SelectKBest_k=2 | 2 |
| 3597.5 | 0.7014 | LogisticRegression | {'C': 0.01, 'penalty': 'l1'} | SelectKBest_k=5 | 5 |

Figure 2 Example results for LogisticRegression

For final step of determining best model out of the , best combination of parameters for each model was taken into account with lists of used hyperparameters:

| Score | Accuracy | Hyperparameters | Feature selection | Number of Features |
|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| **Logistic Regression:** | 3710 | 0.7076 | {'C': 0.1, 'penalty': 'l2'} | SelectFromModel_thresh=99.7th_percentile | 2 |
| | 3342,5 | 0.7172 | {'C': 0.1, 'penalty': 'l1'} | SelectKBest_k=4 | 4 |
| **Random Forest:** | 3500 | 0.7174 | {'n_estimators': 500, 'max_depth': 5} | SelectKBest_k=4 | 4 |
| | 3807,5 | 0.7048 | {'n_estimators': 500, 'max_depth': 3} | SelectFromModel_thresh=99.7th_percentile | 2 |
| **AdaBoost:** | 38600 | 0.7060 | {'n_estimators': 400, 'max_depth': None} | SelectKBest_k=2 | 2 |
| **GradientBoosting:** | 3342,5 | 0.7172 | {'C': 0.1, 'penalty': 'l1'} | SelectKBest_k=4 | 4 |
| **Bagging:** | 3055 | 0.7068 | {'n_estimators': 500} | SelectKBest_k=6 | 6 |

Figure 3 Subset of best combinations

With these models and used parameters, we evaluate the subset of variants on the same dataset split into training and testing sets.

## 5. Final Model

The final, most promising selected model was a **RandomForestClassifier** with **400** estimators and a maximum depth of **5**, using only **4 features** selected via SelectKBest method. This model produced the highest cost-adjusted score of **4380** and accuracy of **72.4%.**