Advanced Machine Learning Project

Feature Selection

Natalia Safiejko, Wojciech Grabias, Zuzanna Piróg

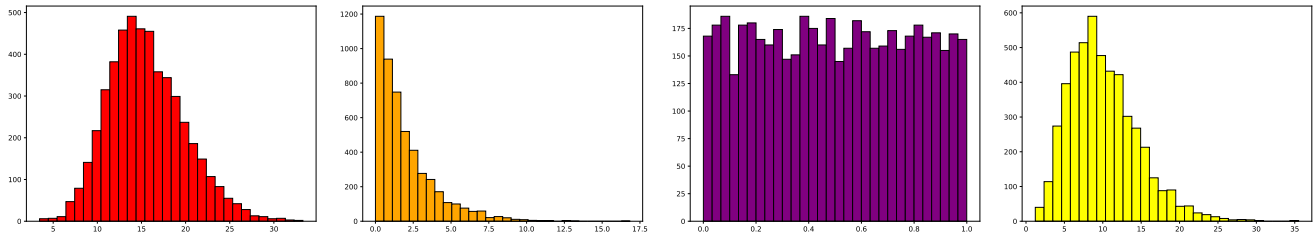
31.05.2025

Abstract

This project addresses the problem of building accurate and parsimonious binary classification models for high-dimensional customer data in the context of an energy-saving initiative. The objective is to identify households that are likely to exceed a specified electricity usage threshold while minimizing the number of costly input variables used. We explore five modeling strategies, each combining a machine learning algorithm (e.g., logistic regression, gradient boosting) with a feature selection method (e.g., mutual information, SHAP, VIF). Models are evaluated using a utility-based scoring framework that rewards correct predictions and penalizes the number of features used. The dataset consists of 5,000 training samples and 500 anonymized variables, with performance measured on a 5,000-sample test set. Evaluation includes both predictive accuracy (top-1,000 classification) and cost-effectiveness, with final selection based on net reward. The implementation includes modular code for preprocessing, model training, variable selection, and result visualization, ensuring reproducibility and interpretability of the final model.

1 Features distributions and scaling

An important observation of the dataset is that the feature distributions among the datasets are consistent within specific index ranges, as shown in Figure 1. The preprocessing pipeline applies tailored transformations based



(a) Indices 0–199 (b) Indices 200–299 (c) Indices 300–399 (d) Indices 400–499

Figure 1: Distributions examples of features from given index range

on the underlying data distributions. Normally distributed features are standardized using StandardScaler, while QuantileTransformer (with a normal output) transforms exponential features to approximate a Gaussian distribution. MinMaxScaler scales uniformly distributed features to a fixed range, and PowerTransformer (Yeo-Johnson) is used to normalize skewed normal features for better symmetry and variance stability. An alternative scaler was used interchangeably that only clips outliers and applies logarithmic transformation to exponentially-distributed data.

2 Methodology

Because of the nature of the project, where 1000 out of 5000 households need to be identified as class 1 representative, the criterion to be optimized is precision of a classifier at top-20% of highest-rank (probability wise) predicted observations. This criterion (further in the report referred as *score*), should be optimized with an additional constraints of variables considered by the classifier, which - due to the cost tradeoff - can be translated into that each variable addition needs to contribute to at least 2% gain in score in order to make it worth to take it into account.
Each of the feature selection methods either first filtered out variables with filter methods (mentioned in Section 3) due to the computational complexity of the method itself. Afterwards, the core feature selection mechanism selects most-important features. In the final stage, cost-optimization considering variable-score tradeoff was made. In order to verify that both visually and numerically, a k-fold validation (10 for EA and PAFS, 5 for SHAP) was introduced for each model and variable set configuration, as shown in Figure 2.

3 Filter methods

Filter methods were used as an initial step in the algorithms before (to a higher or lower extent). Their main goal was to exclude most-likely irrelevant features in order to trim the feature space, rather than choosing the best features for the final classifier.

3.1 VIF

In the methodology for this approach, an iterative approach based on the Variance Inflation Factor (VIF) was used. At each iteration, all variables with a VIF greater than 5 were identified, indicating potentially problematic multicollinearity. Among these, the variable that had the lowest mutual information with the target variable y was

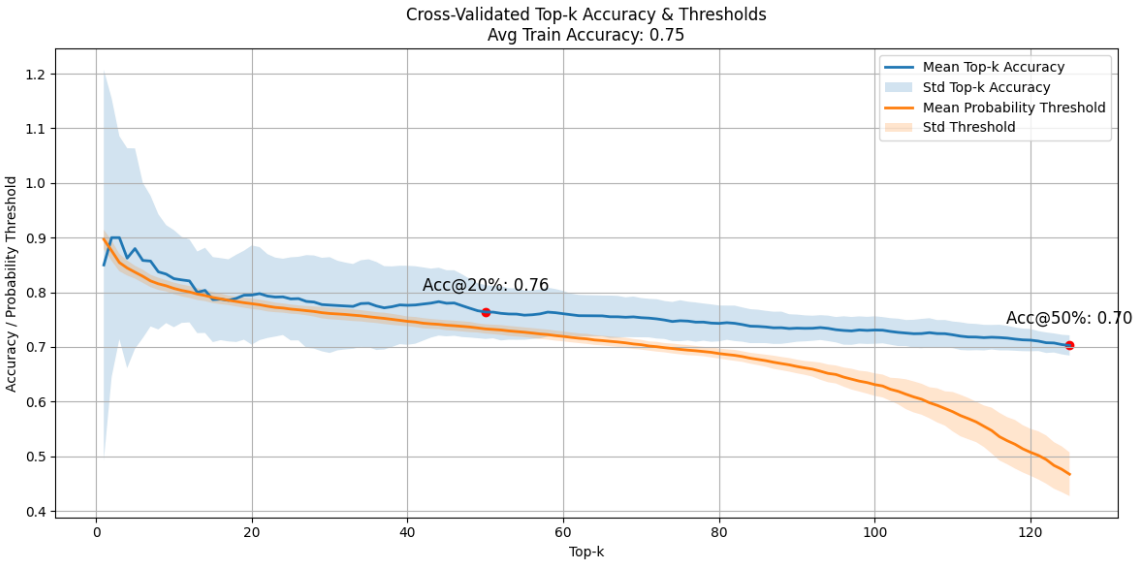


Figure 2: Visual representation of the score evaluation (accuracy is equivalent to precision at top-20% threshold)

selected, as it contributed the least useful information for prediction. This variable was then removed from the feature set. The process was repeated until all remaining variables had VIF values below the threshold, ensuring a more stable and interpretable model. This approach led to potential removal of 19 variables.

3.2 ANOVA F-test

The ANOVA F-test assesses the dependency between each feature and the target by comparing the variances across different groups. It computes an F-statistic for each feature, which reflects the ratio of between-group to within-group variance. Inverse of the p-values was used as the feature importance.

3.3 ReliefF

ReliefF assigns scores to features based on how well they differentiate between neighboring instances of different classes. It updates feature weights by comparing each instance to its nearest neighbors from the same and different classes. Direct feature importances of the method were used.

3.4 Mutual Information

Mutual Information quantifies the amount of shared information between a feature and the target variable. It evaluates how much knowing the value of a feature reduces uncertainty about the target. The exact MI value was used as the feature importance.

4 Basic feature selection methods

Basic feature selection methods verification was included in the `basic_feature_selction.ipynb` notebook. The notebook itself is split into sections, for which each is dedicated for one method mentioned below. It is important to denote that selection methods below mainly

4.1 Logistic regression with L1 penalty

With penalty factor of $C = \lambda^{-1} = 0.01$, L1 logistic regression provided 4 most important features (in comparison, 2 variables were chosen for $C = 0.005$ and more than 20 for $C = 0.05$), among which all of them were evaluated using non-penalized Logistic Regression algorithm and proven to be ultimately irrelevant, **worsening** the score in comparison to singleton feature space – $\{2\}$ with a score of 0.725.

4.2 XGBoost feature importance

After fitting the algorithm to the whole data, inherently-constructed feature importances indicated variable 2 as the most important, significantly more than the others (over $4\times$ higher feature importance value between the first and second variable). Greedy feature addition was then implemented with XGBoost itself as the classifier. Score-profit tradeoff again proven singleton $\{2\}$ to be optimal from this method point of view with a score of 0.718.

4.3 MARS

MARS was evaluated in three variants: with entire feature space, excluding VIF-filtered variables and excluding VIF-selected variables and variable 2. For each configuration, greedy addition of features importance-wise was performed with a MLP with two layers as the classifier. Best performance (tradeoff-wise) was again achieved for a singleton of variable $\{2\}$ with the score of 0.722 for VIF-filtered variables, score 0.738 for variables $\{2, 8\}$ in the all-features section and score 0.508 for single variable $\{209\}$ in the third variant. Further increase in analyzed feature space either decreased the model performance or did not provide sufficient score increase.

5 PAFS

`PrecisionAwareFeatureSelector` is a two-stage feature seector. It first fits three diverse base models (Random Forest, Gradient Boosting, L1-penalized Logistic Regression), normalizes each model's importances, and retains the top-k union of their signals. On this reduced dataset (feature-wise), it performs greedy forward selection: at every step it adds the feature that gives the best cross-validated score (using whichever of the three models performs best on that fold), and stops when either the pre-defined maximum number of features is reached, the score is no longer improved for a configurable patience window, or the Jaccard similarity of recent selections indicates convergence.
PAFS was tested in four different setups on default class parameters:

1. Entire feature space (Max score: 0.782)
2. Feature space excluding VIF-filtered variables (Max score: **0.783**)
3. Feature space excluding first 10 features (Max score: 0.766)
4. Feature space excluding VIF-filtered variables and variable 2 (Max score: 0.571)

Among those, VIF-excluded feature space provided (slightly) highest results. Because the Gradient Boosting model was used many times during feature selection with proven highest contribution, it was chosen to be the final model evaluating the final precision at top 20% of the out of fold data. The selected features were chosen as follows: first, all features from two best forward searches were combined, reaching the result of 0.765. Afterwards, highly-correlated features (3 and 4) were eliminated, reducing the performance only to 0.758. Then, variable 323 was deleted because it diminished its forward selection performance. The final step involved performing feature importance verification

using GradientBoosting algorithm, leading to the best performance reached with variables $\{2, 374\}$ with a score of 0.731.

6 Evolutionary Algorithm

The evolutionary feature selection pipeline employed a multi-stage process combining classical feature ranking methods with a genetic algorithm to optimize feature subsets for a Random Forest classifier. The key stages of the process were as follows:

1. **Pre-filtering**: Three feature scoring methods were applied to the dataset: ANOVA F-test, ReliefF (with 10 neighbors), and Mutual Information. For each method, the features were ranked by importance. The bottom 400 features common across all three methods were identified as likely irrelevant and removed, significantly reducing the feature space while retaining variables with stronger signals.

2. **Evolutionary Search**: A custom genetic algorithm was designed to search the reduced feature space for optimal subsets of features. Each individual (feature subset) in the population was represented as a binary vector indicating which features were included. The algorithm maintained a strict constraint: individuals were dynamically repaired to contain exactly 5 features per the `MAX_FEATURES` limit. This repair was done by training a small Random Forest on the current feature set and selecting the top features by importance. Fitness was evaluated as the cross-validated AUC score (5-fold CV) of a Random Forest classifier (`n_estimators=100`). The population evolved over 30 generations with tournament selection, two-point crossover, bit-flip mutation (with adaptive probability), and immigration (replacing the lowest-performing individuals each generation). The Hall of Fame retained the top 5 individuals.

3. **Final Model Selection and Subset Optimization**: The best individual from the evolutionary search (a feature subset) was used to train a final Random Forest classifier (`n_estimators=200`). To further refine the selected variables, all possible subsets of the selected features from a few iterations (2, 215, 238, 351, 424, 7, and 298) were exhaustively tested to reveal the best solution.

The evolutionary search achieved a best score of **0.77** (on the top 20% of the dataset) using 4 variables and **0.76** using 3 variables. The final selected variables were indices 2, 7, and 215, as in the problem it is more optimal to choose fewer variables when the difference in scores is smaller than 2%.

7 SHAP

The SHAP approach focused on interpretable feature selection using SHapley Additive exPlanations. The methodology began by preprocessing the data using custom tree-based feature scalers, followed by training an initial Gradient Boosting classifier with 150 estimators, a learning rate of 0.05, max depth of 4, and subsampling of 80% of the data. SHAP values were then computed for this model using TreeExplainer on a representative sample, with feature importance determined by the mean absolute SHAP values across all samples.

The top 5 most important features were selected based on their SHAP importance scores, revealing features `2, 462, 414, 6, 8` as most impactful, with feature `2` showing particularly dominant importance (0.710). A reduced model trained solely on these top 5 features achieved an accuracy of 0.7065 on the top 20% of predictions. It was further tested wherther all the detected features were essential for the best model performance. As it proved, using

only the two most important features (2 and 462) actually improved performance to 0.7154, suggesting these features contained the most predictive signal while reducing overfitting.

To address the high correlation among the first 10 variables while incorporating SHAP insights, we developed a custom feature engineering pipeline. The approach created three feature groups: (1) PCA components from the first 10 correlated variables, (2) SHAP-selected features excluding those in the first 10, and 1 other relevant feature being 462. The implementation used a `PCAShapFeatureEngineer` class that applied PCA (n_components=3) to the first 10 features while preserving the important SHAP-identified feature. This transformed feature set was then used in a pipeline with standard scaling and L1-penalized logistic regression (C=0.1).

The final model in this approach used 4 engineered features: three PCA components and the SHAP-selected feature 462 (feature 2 was excluded as it belonged to the first 10 correlated variables handled by PCA). This combined approach achieved a test accuracy of 0.70 $\pm$ 0.01. The PCA approach, while theoretically sound for handling correlated features, ultimately proved ineffective in practice. By converting the first 10 features into 3 principal components while retaining other important features, the model ended up using 11 total features - a number too high to provide meaningful improvement compared to the needed cost.

8 Final model and feature selection

Throughout the experiments, a score no higher than 0.79 was denoted by any variable subset or model considered. Furthermore, a score of no higher than 0.77 was denoted by any model with three variables or more. Because of that happening with a variety of models and a variety of different variable subsets, **a consensus was reached that the variable {2} would be the one considered in the final solution**, as it appeared in all of the methods' final results. On top of that, reaching a performance of score equal 0.74 would already make the variant the best, considering the variable-precision tradeoff. The remaining question was as to which model should be used. While it may still be possible that for a better-suited configuration of models and already-discovered variables, the end result would be better, this possibility did not seem feasible, due to the variety of different machine learning algorithms used.

In order to find the most suitable model, 10 different algorithms and/or ensembles using them were tested. **Among those, Support Vector Machine classifier was found to be most efficient with a score of 0.748** and stable results considering probability thresholds. Detailed results of all of the algorithms are presented in the `final_model_selection.ipynb` notebook. The results in a form of indices chosen to be of class 1 are located in `320628_vars.txt` file.