ADVANCED MACHINE LEARNING - PROJECT 2

Klaudia Gruszkowska, Zofia Łągiewka, Jacek Zalewski
Faculty of Mathematics and Information Science
Warsaw University of Technology

June 3, 2024

Contents

1 Introduction

The goal of the project is to build a parsimonious model to maximize *profit* function:

$$profit = €1000 \times accuracy - €200 \times number\ of\ features \tag{1}$$

In each experiment, we perform feature selection, train a model on the selected features, and evaluate the model's accuracy. Our evaluation metrics include overall accuracy, the accuracy of the top 20% of predictions for the samples with the highest class 1 probability, and income calculated according to equation 1. The $accuracy$ in the formula 1 is estimated on top 20% most probable predictions on test split. In every experiment, the data is divided into training and testing split (ratio: $80 : 20$). All experiments are repeated multiple times for different random splits.

2 Feature selection

In this section, we will present results obtained using different strategies - combinations of various feature selectors and models. Different thresholds of feature importance were tested. Below we present the results of the models trained on feature importance above $threshold = 0.008$. The following strategies were considered:

- `selector = SelectFromModel(RandomForestClassifier(), threshold=0.008)`
  `model = GaussianNB()`
- `selector = SelectFromModel(RandomForestClassifier(), threshold=0.008)`
  `model = RandomForestClassifier()`
- `selector = SelectFromModel(RandomForestClassifier(), threshold=0.008)`
  `model = ExtraTreesClassifier(bootstrap=False, criterion="gini", max_features=0.95, min_samples_leaf=7, min_samples_split=2, n_estimators=100)`
- `selector = SelectFromModel(ExtraTreesClassifier(bootstrap=False, criterion="gini", max_features=0.95, min_samples_leaf=7, min_samples_split=2, n_estimators=100), threshold=0.008)`
  `model = ExtraTreesClassifier(bootstrap=False, criterion="gini", max_features=0.95, min_samples_leaf=7, min_samples_split=2, n_estimators=100)`
- `selector = SelectFromModel(ExtraTreesClassifier(bootstrap=False, criterion="gini", max_features=0.95, min_samples_leaf=7, min_samples_split=2, n_estimators=100), threshold=0.008)`
  `model = SVC(probability=True)`

Each of the strategies was evaluated 100 times. The results, in the form of the boxplots, are visible in Figure 1. As we can see the best average income was obtained using the combination of `SelectFromModel()` with `RandomForestClassifier()` feature extractor and Gaussian Naive Bayes model. The average *income* value for that strategy was €6863, whereas for the worst performing strategy it was €6162.

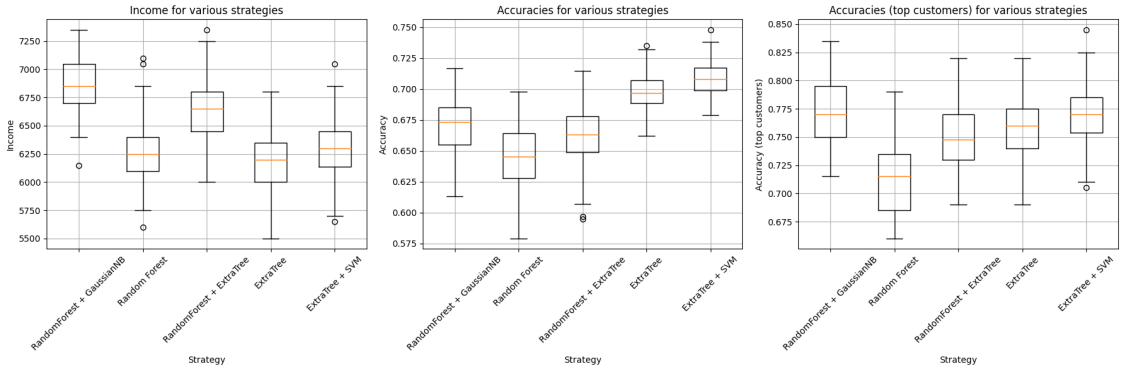

Figure 1: Results of the described strategies

3 Hyperparameter tuning and different feature combinations

We noticed that the results obtained from feature selectors described in the previous section often include features from range 101-105. The choice of those variables was also confirmed by *Boruta* algorithm. Below in Figure 2 we present the frequency of each feature set chosen by `SelectFromModel() with RandomForestClassifier()` feature selector.

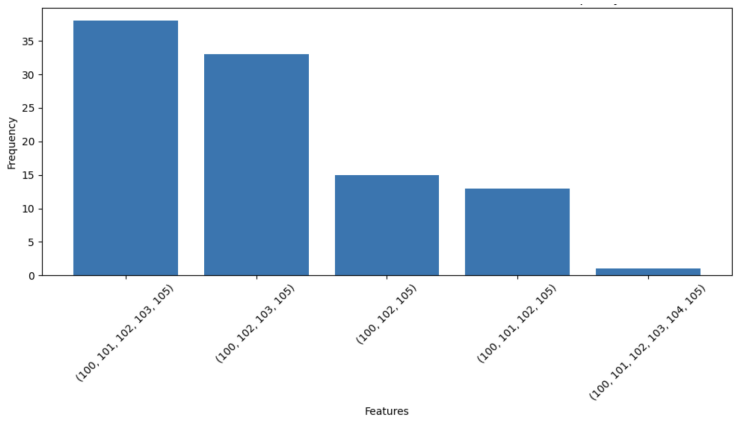

Figure 2: SelectFromModel with Random Forest - feature choices and their frequency

Moreover, we investigated the distribution of all the features. In the aforementioned feature range, we noticed the discrepancy between the distributions of class 0 and 1, which could potentially make the classification easier.

Since we expected that the use of some features in the range 101-105, could increase the results, we checked several combinations of length 2, 3, and 4. The best results were obtained using sets [102, 103, 105] and [101, 102, 103, 105]. The numerical results will be presented in the following section.

Once we found those two well-performing feature sets, we decided to try to increase the income by increasing the accuracy of the model. To do so, we used *sklearn.model_selection.GridSearchCV* to tune the hyperparameters of models. As different models have different parameters we defined dictionaries of a set of all possible values of the model.

3.1 Feature set [102, 103, 105]

In this section, we will describe the experiments using different models and model parameters trained and evaluated on the feature set [102, 103, 105].

3.1.1 ExtraTreeClasifier

We performed GridSearch for the ExtraTreeClassifier, testing 5 parameters, with a total of 30 values.
The returned values were: `'bootstrap': False, 'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 30`.
The model was then trained using 1000 different train/test splits and the average income was approximately €6839.15.

3.1.2 KNeighborsClassifier

Next, we used GridSearch for the KNeighborsClassifier. We tested approximately 25 different values of the parameters.
The best performing values were as follows: `'algorithm': 'auto', 'leaf_size': 1, 'metric': 'chebyshev', 'n_neighbors': 150, 'p': 1, 'weights': 'distance'`.
The average income calculated over a 1000 iteration was €6732.5, which is slightly worse than the result from the previous section.

3.1.3 RandomForestClassifier

For RandomForest total of 10 parameter values were tested. The best-performing ones were: `'criterion': 'gini', 'max_depth': 7, 'n_estimators': 200`.
The average income reached €6716, which is also slightly worse than the ExtraTreeClasifier described earlier.

3.1.4 GaussianNB

In the case of Gaussian Naive Bayes, there weren't many parameters to be tested, as the only one that could significantly influence the results was `'var_smoothing'`. For said parameters we tested 10000 different values, evenly distributed between 0 and 1. The GridSearch returned 1 as the best-performing value.
The average income increased significantly compared to previous experiments as it reached approximately € 7350, however we decided that using such an extreme `'var_smoothing'` value ultimately is not beneficial. When we tried to predict the 1000 customers on the test set given, the number of clients with predicted value 1 was lower than 1000.

3.2 Feature set [101, 102, 103, 105]

3.2.1 MLP

A total of 16 parameter values was tested out of which the best-performing ones were: `'activation': 'relu', 'alpha': 0.01, 'hidden_layer_sizes': (100,), 'learning_rate': 'invscaling', 'learning_rate_init': 0.001, 'max_iter': 400, 'solver': 'adam'`
The result obtained was satisfactory as we can see an increase of income, with average income being approximately €6900.

3.2.2 RandomForestClassifier

Similarly as in the three feature set case, here, 10 parameter values were tested. The best-performing ones were:`'criterion': 'entropy', 'max_depth': 8, 'n_estimators': 200`, and the average income value obtained was approximately €6638.

3.2.3 GaussianNB

Once again, analogously as for the three feature set, for GaussianNB we tested only one value with the same parameters as described before. The GridSearch once again returned 1. The average income obtained was equal to approximately € 7031. Similarly as before we decided not to use it, due to the extreme `'var_smoothing'` value.
However, when running the model with default parameters the results obtained were still quite satisfactory. The average income reached approximately € 7014. The results are visible in Table 1, and the income distribution for this model is shown in Figure 3.

Table 1: Final results

accuracy	accuracy on top 20%	income	income std
0.66	0.78	€7014	€265

3.3 Different feature sets

Furthermore, we have tested Gaussian Naive Bayes with default parameters on all 3 and 4-element combinations of features from 101 to 105. The model trained on features 101, 102, 103, and 105 is giving the highest income. Figure 4 presents the income for different sets of features.

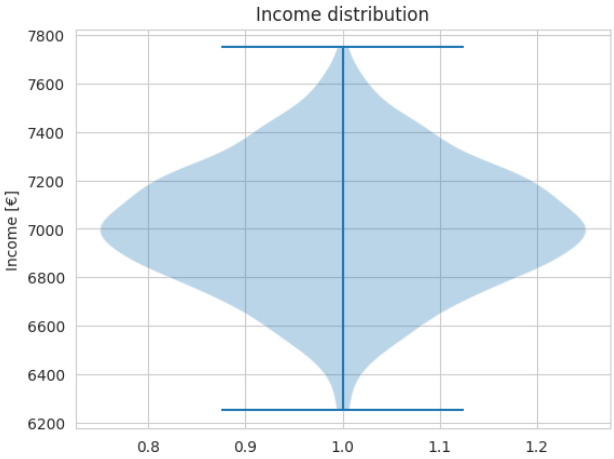

Figure 3: Income distribution of final model

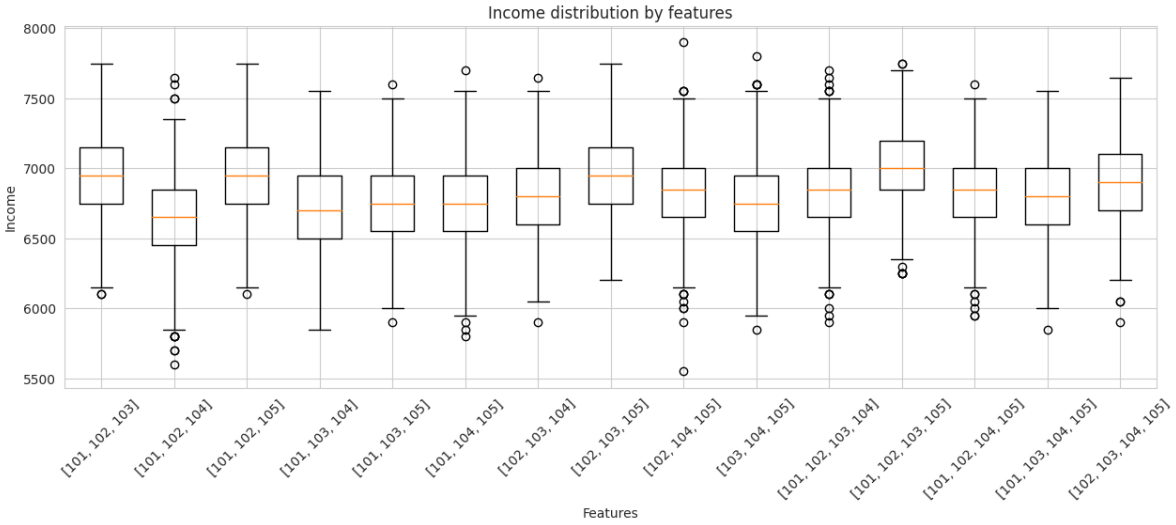

Figure 4: Results of the Gaussian Naive Bayes for different feature sets

3.4 Conclusions

In summary, we managed to reach approximately 78% accuracy, when considering the customers who are most likely to benefit from the offer. The final model was trained on 3 features (101, 102, 103, 105), which overall yielded the income of more than €7000. The best results were obtained by Gaussian Naive Bayes, specifically the model described in Section 3.2.3.