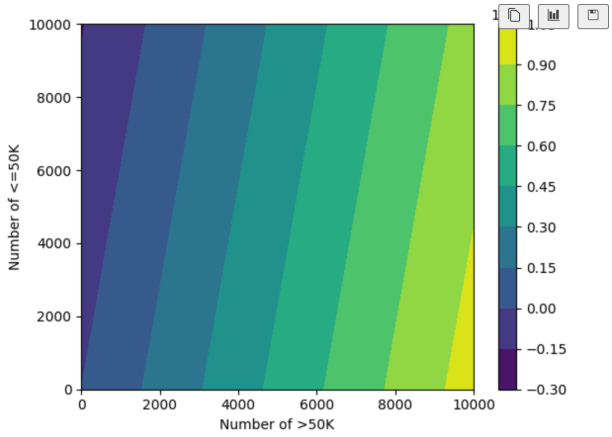
Jakub Niedziela s0224740

Data Mining Assignment 1: Classification – Report

1. Introduction

The main issue of this assignment of expected revenue. Available data contains information about existing customers statistics and the bracket of their income. From notes I know that there are 2 groups of customers – with higher income (above 50K) and lower income (lower than 50K). I need to predict their income class and based on these predictions calculate the expected revenue. For that it is known that on average higher income groups yields 980 euro of income, while lower income customers yield 310 euro loss. Also there is a 10% chance of positive reaction to the promotion for higher income group and 5% chance of positive reacting for lower income group. Based on this information I plotted possible combinations of expected revenue which are presented on fig.1 below. We can see that in order to maximize the revenue we need to choose correctly customers from >50K income group and if possible send promotional mail only to this group.

Fig. 1 – Revenue maximization.



2. Dataset description.

For obtaining statistics of data, discovering the variables and possible issues I have conducted analysis using python package ydata-profiling (newer version of pandas-profiling). It automatically generates different statistics for provided dataset. The report can be seen in file ***your_report.html***. From the report I was able to see some issues and intuition about dataset. The notes are as follows:

- high correlation between education and education-num
- high correlation between sex and relationship
- high imbalance in workclass -> 70% in private sector

- high imbalance in race -> 85% white ppl
- high imbalance in native country -> 90% from US
- 5.6% missing values in workclass
- 5.7% missing values in occupation
- 1.8% missing values in native country
- capital gain / loss cols have large amounts of 0's (both above 90%).

First idea was to drop one of highly corelated columns and as well as drop all the rows with missing values, although in the end I decided to choose different approach for each model I made, which will be described in next sections. I need to note that imputation of missing values is critical for good model performance, but it possibly can infect data. In general missing rows constituted about 7% of total training data. While this number is not big any data loss might lead to less model training and less accurate results. Furthermore there was one more issue with data – class imbalance. There were 75.9% of records in <=50K class and 24.1% in >50K. I deal with this issue later during modelling.

3. Modelling

After conducting analysis I proceeded to do the data modelling using binary classification algorithms. Firstly I used Logistic Regression to get the intuition which features might improve performance. In the end I decided not to use it as the performance gains were not significant, even while providing models with new feature interactions. Therefore in the end I used two models – Catboost Classifier and TabNet classifier.

3.1. CatBoost Classifier

CatBoost python package provides me with great tools, and little need of data preprocessing. The models from this library have little need of data preprocessing. For example we do not need to encode categorical features as well as impute missing values – the algorithms takes care of this issues automatically. Therefore I decided to try few approaches. In each I used cross validation with training data while fitting the model, and then test the model on test data which contains 20% of rows. In first try I used all features expect education column, as it had correlation equal to 1 with variable education_num. Basically the second column had the encoded values of the first column. My approach in this case was to deal with feature provided by catboost to deal with class imbalance – class weights based on the total number of objects in each class. There are two possibilities for this in catboost classifiers – one is Balanced and one is Square Root Balance which is square root of the first statistics. In second try I also dropped the education column and used Balanced weights. In third try I tried to group features

in 4 columns – worcklass (reduce to 3 groups – not employed, Government and private), education (reduce to 4 groups – Higher, Associate, Post-grad and School), marital-status (reduce to 3 groups – Married, No-spouse and Never-Married), race (reduce Asican-Pac-Islander and Amer-Indian-Eskimo to other). In this case I also dropped education-num column as here I chose the categorical version of the variable. I also used the Balanced class weights here because this choice yielded better results in previous models. In each of these 3 tries I used 5 fold cross validation and conducted randomized search for hyperparameters with 100 iterations. The results can be seen on 3 plots below.

Fig.2.1. First try.

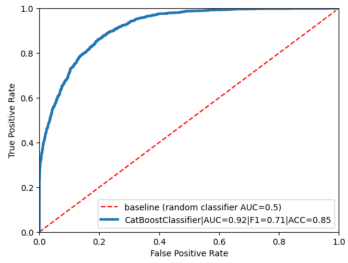


Fig.2.2. Second try.

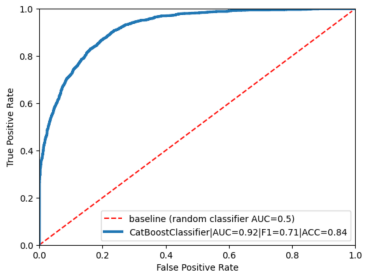
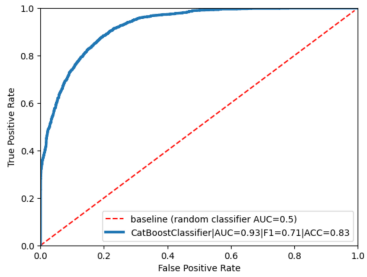


Fig. 2.3. Third try.



Based on results obtained from analysed classifiers I decided to go with the second one, as the method to deal with imbalance was simpler, and also there was not need to process data as in third try.

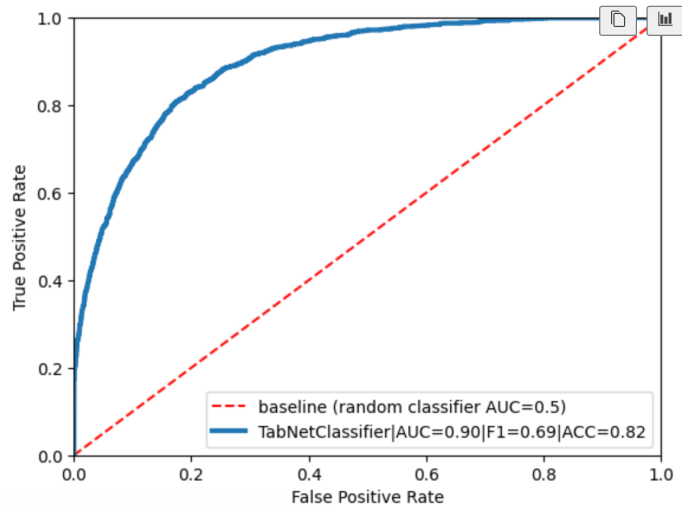
3.2. TabNet Classifier

TabNet is an interesting approach to modelling tabular data using Neural Networks. It is based on deep learning architecture using attention mechanism. "*TabNet uses sequential attention to choose which features to reason from at each decision step, enabling interpretability and more efficient learning as the learning capacity is used for the most salient features."*[1] Thanks to its characteristic it yields better results than simpler classifier, and has similar or better performance to catboost. To preprocess the data as input to Neural Network I had to encode the categorical variables as numerical ones using scikit-learn LabelEncoder. Then to deal with class imbalance I have also decided to upsample the training data. Upsampling data is a process of choosing random rows of minority class and adding them again to data until the class imbalance is gone. This also allows model to give greater attention to minority class, and predict it more accurately which is actually good in our case. After that I have trained the model for up to 200 epochs, while checking for the improvement in last 50. The training ended after

[1] Sercan O. Arık, Tomas Pfister. TabNet: Attentive Interpretable Tabular Learning. Google Cloud AI, 2019.

97 epochs with best epoch 47. The results can be seen on fig 4 below. The scores are slightly worse than catboost model, nevertheless according to AUC the model score can be described as fairly good.

Fig.4 TabNet model results.



4. Calibration

Model calibration is a technique which allows us to calibrate the results of a model as true probabilities. I have only applied a calibration to CatBoost model, as Artificial Neural Networks are trained on probabilistic grounds, therefore they do not need to be calibrated. The calibration process can be seen in **calibration.ipynb** notebook.

5. Predictions

For final predictions I decided to make an ensemble of both trained models as it makes results more robust, especially that CatBoost is a boosting model and TabNet is an Neural Network. To do that I simply predicted class probabilities using both methods and then combined them by adding and dividing them by 2 – average of predictions. Finally having obtained ensemble probabilities I choose a prediction threshold of **0.22**, which maximized the expected return of model which amounted to **279 984.63** euro. For given threshold it is calculated only for customers whose probabilities of being in higher bracket is larger than 0.22, then for each such record I multiply the P(>50K) by 980 and 0.1 (10% of customers react positively) and then I multiply P(<=50K) by -310 and 0.05 (5%). The final amount is summation of probabilities for all the customers predicted to be in higher income group. The indices of customers predicted to be in higher income class are presented in **results/potential_customers.csv.** All the code for this section can be found in **predictions.ipynb** notebook.

Appendix

1. All code used to obtain these results has been uploaded to github repository: https://github.com/dukekush/customers_classification
2. All analysis was done in python code, written using many python packages.
3. Most important files are:
    a. general.ipynb → problem description, data anaylsis
    b. your_report.html → detailed pandas profiling report
    c. catboost.ipynb → catboost models
    d. tabnet_final.ipynb → tabnet model
    e. calibration.ipynb → catboost calibration
    f. predictions.ipynb → ensemble, chosing threshold, final predictions and calculating expected revenue
    g. results/potential_customers.csv → file contatining ID's of customers predicted to be in higher income bracker (the main goal of the project)
    h. helpers.py → function used to preprocess data, cross validate models and plot results
    i. catboost_models/ → folder with saved 3 catboost models
    j. tabnet/ → folder with saved tabnet model and encoder dictionary (for reproducibility during preprocessing).