Data Mining

Assignment 1: Classification

1 Ma INF 2022-2023

Name: Thomas Van Daele
Student ID: 20195723
e-mail: Thomas.VanDaele@student.uantwerpen.be
Github: https://github.com/thdaele/Data-Mining

April 16, 2023

1 Data visualization

First, I started with visualizing the data. I did this in a separate jupyter notebook to have the actual solution separated. In the visualization, I mainly looked for class imbalance in the categorical data. The most important findings here are that there is a lot of class imbalance inside the categorical data.

The data of the income class is imbalanced, with more data available about people with a lower income. The income class is the class that needs to be predicted so that we can calculate to who we need to send a promotion and what the estimated revenue is. (See figure 1)

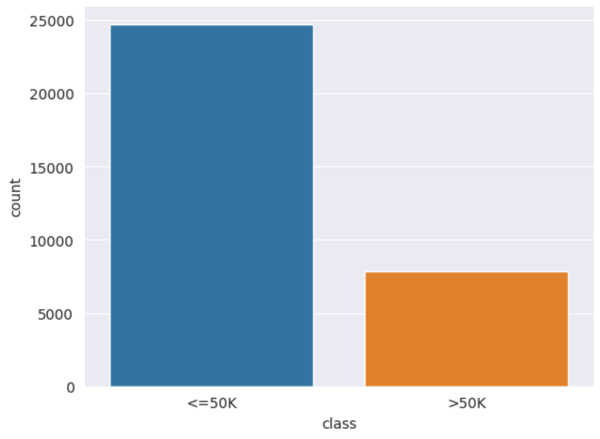

Figure 1: Imbalance in the income data

There is also imbalance in the data of the categorical classes race and sex, I wasn't planning on using these because it would be unethical to use these for a classification tasks like this. During my testing, it didn't matter if I included these in the training data or not. I got similar results with or without them, the models internally learned that the data about the race and sex was less important than for example the data about the education or the occupation. (See figure 2 and figure 3)

I ended up not using the data about the native country, since practically everyone in the training data was from the US. There is too little data about other countries, so the models won't be able to learn anything from it. (See figure 4)

2 Preprocessing

2.1 Dropping data

For the preprocessing, I started with dropping the columns in the data that I didn't need. For example RowID, since the index column on pandas dataframe

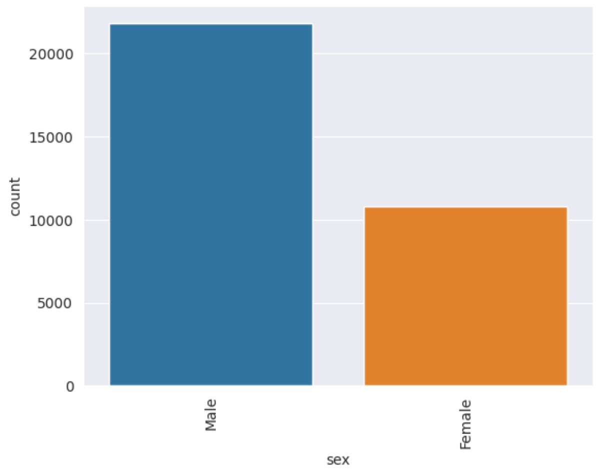

Figure 2: Imbalance in the sex data

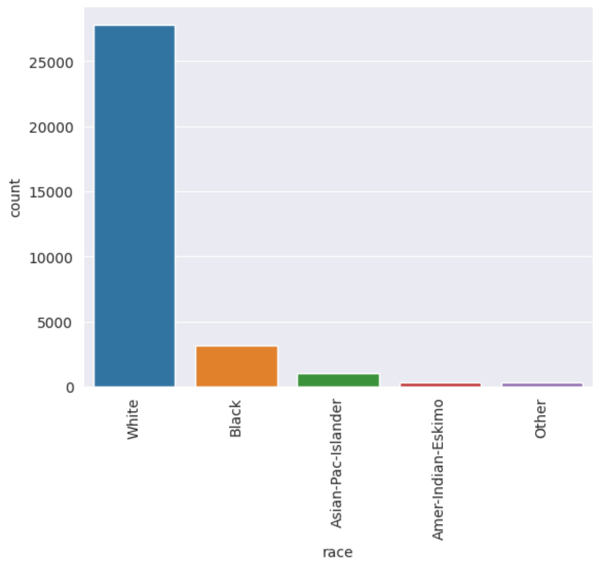

Figure 3: Imbalance in the race data

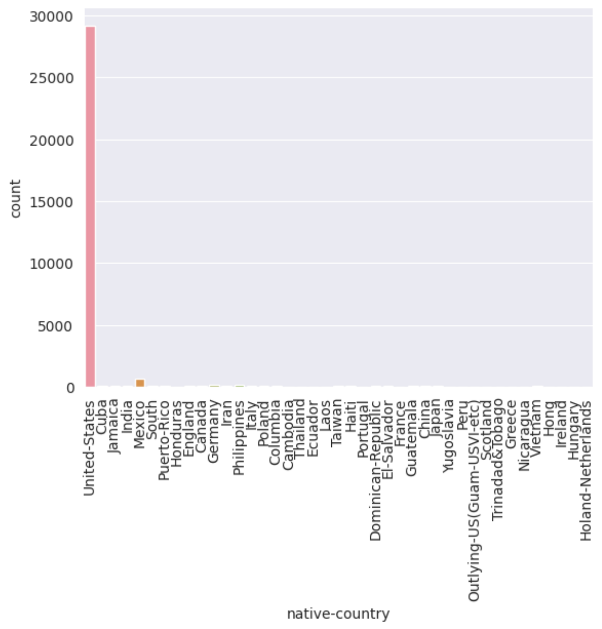

Figure 4: Imbalance in the native country data

can be used to get the customerID too and RowID made it more difficult to do the rest of the preprocessing since the rest of the preprocessing likes numbers and not strings. I also dropped the columns sex, race and native-country for the reasons mentioned in section 1.

2.2 Test data

The dataset is split in test and train data after I drop the columns that I won't use. I use a 70-30 train-test split.

2.3 Categorical data

To deal with categorical data, the columns have to be encoded. For the tree-based models I used one hot encoding and for the non-tree-based models I used ordinal encoding. For every category, I get all the unique possible values that are in the existing customer and the potential customer dataset. All these values are giving to the encoder to make sure it doesn't crash on categories that only exist in the potential customer dataset.

On the income class, I use LabelBinarizer to convert it to a numerical value.

2.4 Imputer

For the imputer I use KNNImputer, since KNNImputer is distance-based it is best to normalize the data first. This is done with MinMaxNormalizer.

2.5 Discretizer

The numerical columns are discretized for non-tree-based models, I didn't use it for tree-based models because those performed better without the discretizer. In this step, values for a variable are grouped together into discrete bins and each bin is assigned a unique integer such that the ordinal relationship between the bins are preserved.

2.6 Imbalanced learn

I used the SMOTE oversampling method from the package imbalanced-learn to deal with the imbalance in the income class. I used the RandomOverSampling from the same package first, but the SMOTE oversampling method does give better results. Imbalanced-learn is a package based on scikit-learn and integrates very well with it. I used pipelines on the entire preprocessing step to prevent data leakage.

3 Training

For the training, I applied a 10-fold cross validation on the training data. After the cross validation, I fitted the model on the training data and verified that it was working well on the test data using the classification report from scikit-learn. During the cross validation, we looked at the recall score. The recall score is an important score to verify how well the model is working in this case. We care less about false positive since they don't cost us too much. Having false negatives, on the other hand is much worse, since we just missed out on a lot of potential to make a profit.

4 Results

I applied a grid search on a model to improve it further and got the final results. We would send the promotion to 6914 people and expect to make a profit of 317984 euro.

The expected profit on each potential customer is calculated using the probability that the user belongs to a certain income class, this we get from the model we trained. Together with the average profit on that class and the chance they will accept the offer, we calculate the expected profit.