Data Mining Case Study

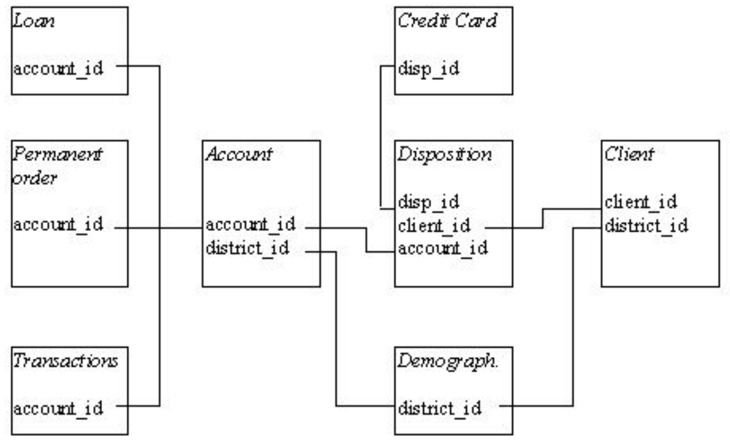
Group 67:
André Mori - up201700493@edu.fe.up.pt
Gonçalo Oliveira - up201705494@edu.fe.up.pt

Domain Description

The focal point of the case study is the analysis of an excerpt of a database from a bank operating in Czechia with records dated from 1993 to 1998. These records concern its accounts, credit cards, customers, demographic information, transfers and loans given by the bank.

The dataset contains a total of:

- 4500 accounts
- 5369 client
- 77 districts
- 426885 transactions
- 682 loans



Problem Definition

“The goal of this project is to develop a Data Mining model that can help the bank’s managers achieve their goal of predicting who is a good and profitable client and who is not, which will aid them in the process of choosing whose client to build a close relationship with, that is, the client to whom they should offer additional services. This problem will be divided in two tasks, a descriptive and a predictive one. The former consists of the identification of consumer segments while the latter consists in the prediction of whether a loan will end successfully or not.

In Data Mining terms, the problem at hand can be seen as a Binary Classification problem given that the purpose of the model is to predict whether a set of features falls in the class "loan ends successfully" or it's opposite. Like all Data Mining problems, the dataset must first be analyzed and interpreted (and eventually modified) in order to build the predictive model.”

Tools Used

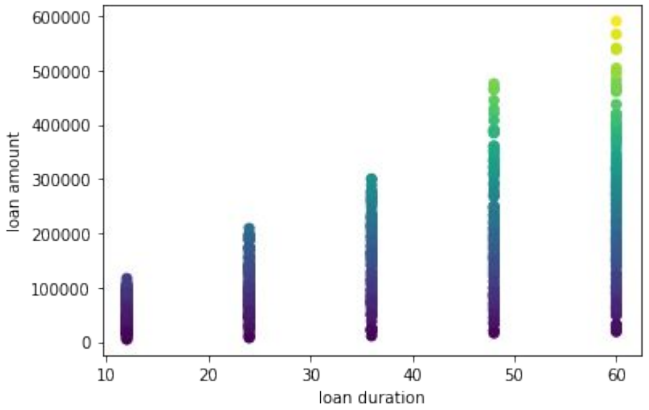
- Python
- Jupyter Notebook
- Pandas
- Numpy
- Scikit Learn
- Matplotlib
- VSCode
- Git

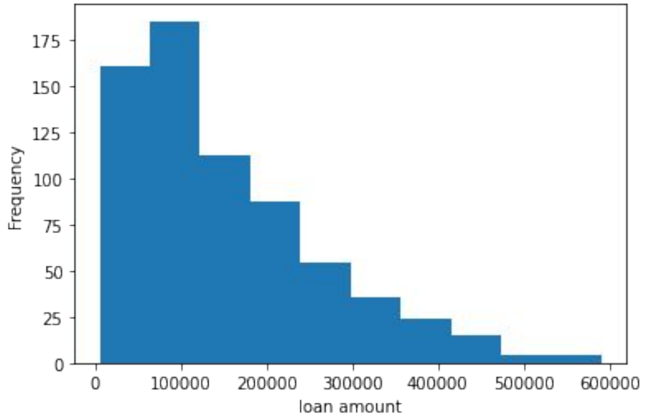
Exploratory Data Analysis

Main Findings

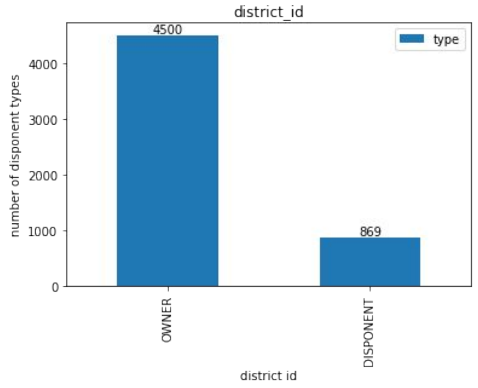
- The number of loans that default (46) is much smaller than the number of loans that end successfully (281);
- No correlation between loan amount and status of payment;
- Positive correlation between loan duration and loan amount, and loan duration and number of payments;
- The most commonly used type of credit card is ‘Classic’, however there are very few accounts that own a card;
- The distribution of accounts in each district is heavily skewed with the largest district having almost 3.6 times the number of accounts (554) of the second district (152);
- Only 6 of the 73 districts have more than 100 clients, with one having 663 clients.

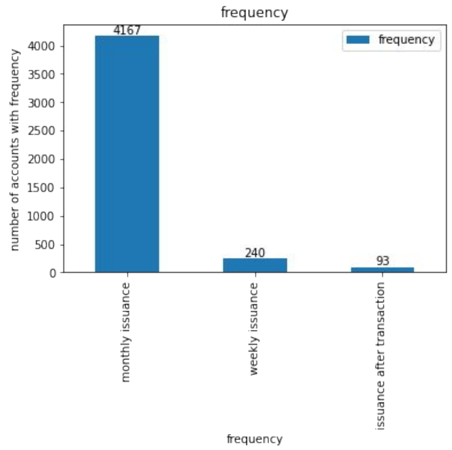
Exploratory Data Analysis

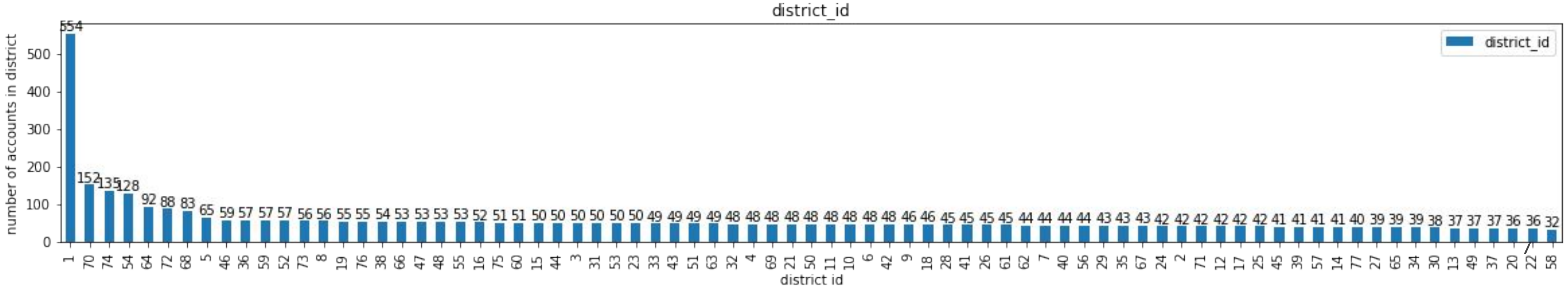




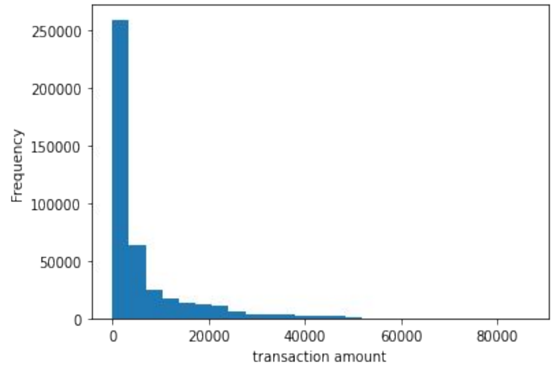
Exploratory Data Analysis

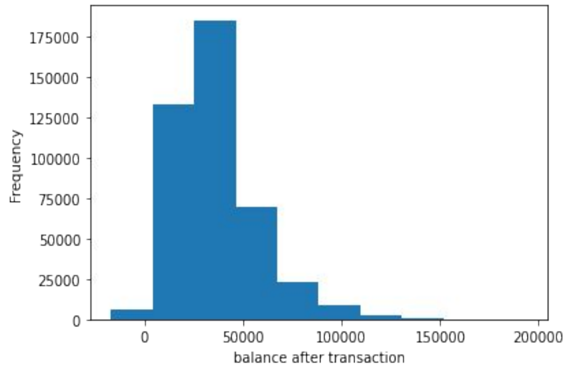


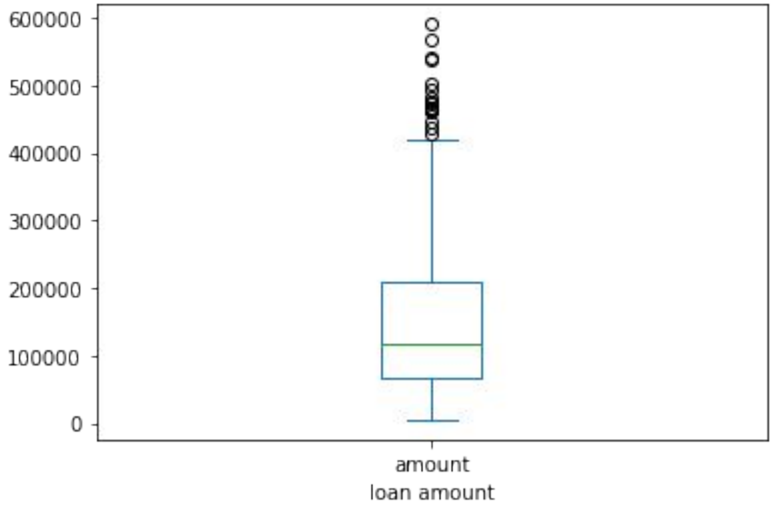




Exploratory Data Analysis







Descriptive Data Mining Task

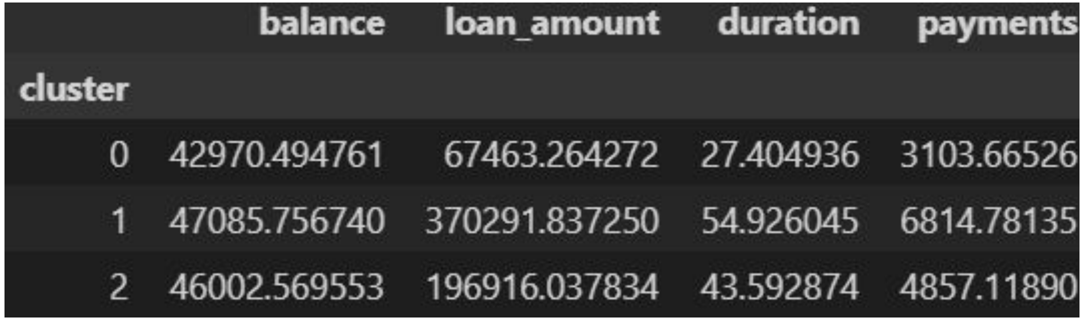
Experimental Setup

- Load dataset with all features generated;
- Used **KMeans**, **DBSCAN**, **KMedoids** algorithms;
- Find correlation between client and loans, client and balances.

Results (1/3)

The purpose of this clustering was to segment the groups according to characteristics of the loans that were taken and balance in their accounts.

The result shows a relation between the client's balance, loan amount and its duration and number of payments, meaning that a bigger balance is linked to a bigger amount and duration. Clients with more money also borrow more.

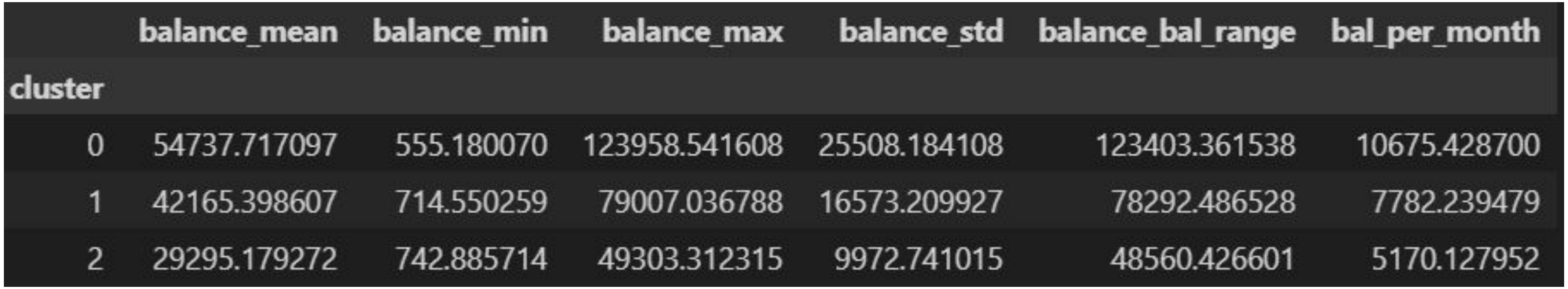


	balance	loan_amount	duration	payments
cluster				
0	42970.494761	67463.264272	27.404936	3103.66526
1	47085.756740	370291.837250	54.926045	6814.78135
2	46002.569553	196916.037834	43.592874	4857.11890

Results (2/3)

The purpose of this clustering was to segment the groups according to their economic power.

We can conclude that the group of people with a bigger balance spend more than those who have less, having a lower balance minimum and wider balance range.

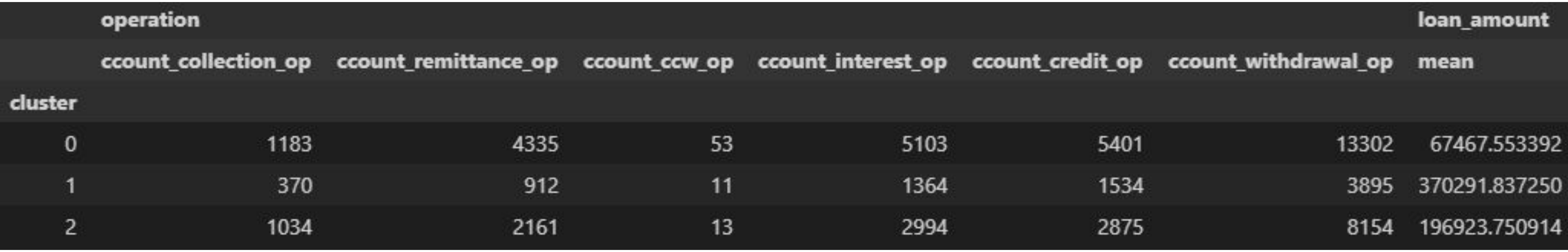


	balance_mean	balance_min	balance_max	balance_std	balance_bal_range	bal_per_month
cluster						
0	54737.717097	555.180070	123958.541608	25508.184108	123403.361538	10675.428700
1	42165.398607	714.550259	79007.036788	16573.209927	78292.486528	7782.239479
2	29295.179272	742.885714	49303.312315	9972.741015	48560.426601	5170.127952

Results (3/3)

The purpose of this clustering was to segment the groups according to their most frequent operation types.

We can conclude that the higher the number of credit card withdrawal operations, the higher the amount of the loan. The same can be said for the number of collection operations. Meanwhile, the high number of interest credits and credit in cash operations seems to correlate to a lower amount borrowed.



	operation						loan_amount
	ccount_collection_op	ccount_remittance_op	ccount_ccw_op	ccount_interest_op	ccount_credit_op	ccount_withdrawal_op	mean
cluster							
0	1183	4335	53	5103	5401	13302	67467.553392
1	370	912	11	1364	1534	3895	370291.837250
2	1034	2161	13	2994	2875	8154	196923.750914

Predictive Data Mining Task

Problem Definition

“A bank wants to improve its services. However, its managers only have a vague idea of who is or isn't a good client, which makes it hard to know who should be offered additional services that can bring profit to the bank, while improving the client's relationship with it.

Fortunately, the bank gathers information about their clients and their accounts, loans already granted and credit cards issued but how can the bank know which clients should be offered additional services from this data?”

To loan or not to loan?

Data Preparation

Data Cleaning

- Removed ***disp_id*** from final dataframe
- Only consider clients of ***type*** “OWNER” from *dispositions*
- Removed ***name*** and ***region*** from *districts*
- Use ***k_symbol*** when ***operation*** is null on *transaction*
- Extract birthdate and gender from ***birth_number*** on *clients* dataframe

Data Preparation

Data Transformation

- Transformed nominal to numeric attributes using **LabelEncoder**
- Data normalization using **QuantileTransformer** (normal distribution)

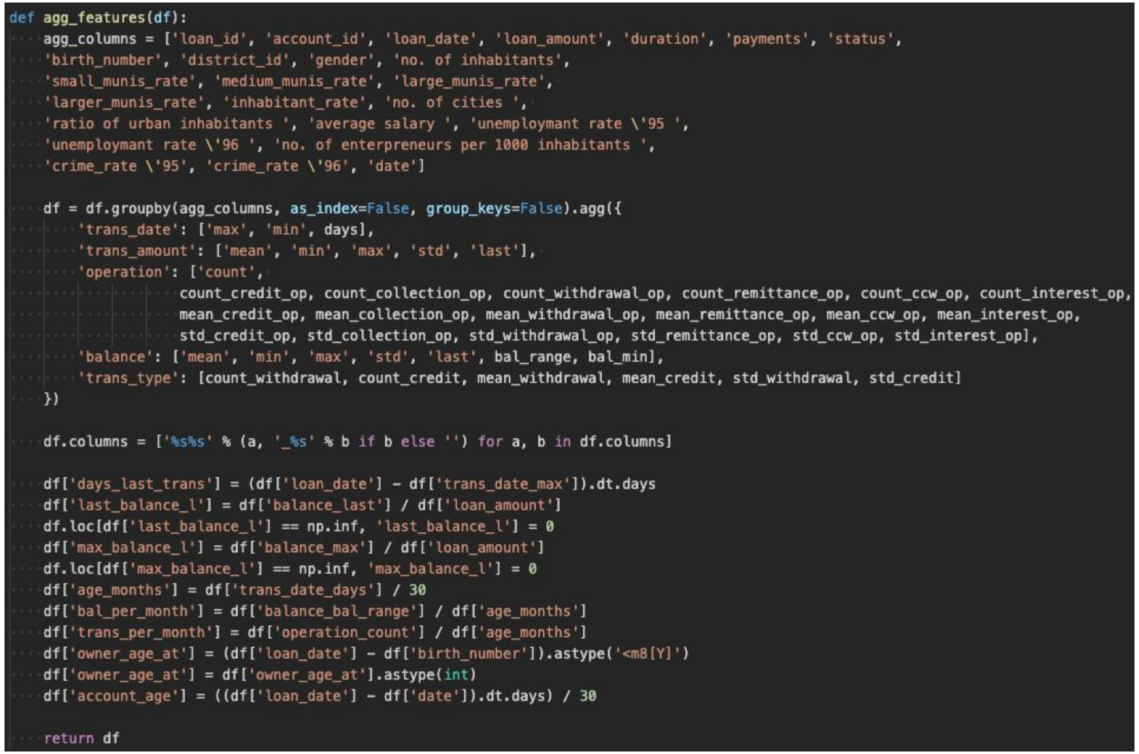
Missing Data

- Missing data on ***unemployment rate ‘95*** and ***no. of committed crimes ‘95*** from the *districts* table using **IterativeImputer** (Bayesian Ridge regression)

Feature Engineering

Generated over 40 new features, having an aggregate data frame of about 80 columns.

Most of them are regarding the client’s past transactions, since it has the most data available.



```python
def agg_features(df):
    agg_columns = ['loan_id', 'account_id', 'loan_date', 'loan_amount', 'duration', 'payments', 'status',
    'birth_number', 'district_id', 'gender', 'no. of inhabitants',
    'small_munis_rate', 'medium_munis_rate', 'large_munis_rate',
    'larger_munis_rate', 'inhabitant_rate', 'no. of cities ',
    'ratio of urban inhabitants ', 'average salary ', 'unemploymant rate \'95 ',
    'unemploymant rate \'96 ', 'no. of enterpreneurs per 1000 inhabitants ',
    'crime_rate \'95', 'crime_rate \'96', 'date']

    df = df.groupby(agg_columns, as_index=False, group_keys=False).agg({
        'trans_date': ['max', 'min', days],
        'trans_amount': ['mean', 'min', 'max', 'std', 'last'],
        'operation': ['count',
                    count_credit_op, count_collection_op, count_withdrawal_op, count_remittance_op, count_ccw_op, count_interest_op,
                    mean_credit_op, mean_collection_op, mean_withdrawal_op, mean_remittance_op, mean_ccw_op, mean_interest_op,
                    std_credit_op, std_collection_op, std_withdrawal_op, std_remittance_op, std_ccw_op, std_interest_op],
        'balance': ['mean', 'min', 'max', 'std', 'last', bal_range, bal_min],
        'trans_type': [count_withdrawal, count_credit, mean_withdrawal, mean_credit, std_withdrawal, std_credit]
    })

    df.columns = ['%s%s' % (a, '_%s' % b if b else '') for a, b in df.columns]

    df['days_last_trans'] = (df['loan_date'] - df['trans_date_max']).dt.days
    df['last_balance_l'] = df['balance_last'] / df['loan_amount']
    df.loc[df['last_balance_l'] == np.inf, 'last_balance_l'] = 0
    df['max_balance_l'] = df['balance_max'] / df['loan_amount']
    df.loc[df['max_balance_l'] == np.inf, 'max_balance_l'] = 0
    df['age_months'] = df['trans_date_days'] / 30
    df['bal_per_month'] = df['balance_bal_range'] / df['age_months']
    df['trans_per_month'] = df['operation_count'] / df['age_months']
    df['owner_age_at'] = (df['loan_date'] - df['birth_number']).astype('<m8[Y]')
    df['owner_age_at'] = df['owner_age_at'].astype(int)
    df['account_age'] = ((df['loan_date'] - df['date']).dt.days) / 30

    return df
```

Experimental Setup

After the data preparation process:

- Upsample classes with SMOTE to balance classes;
- Hyperparameter Tuning and Cross Validation using GridSearchCV and StratifiedKFold;
- Performance analysis (AUC, precision, recall, F1);
- Write predictions to a file.

Algorithms tested:

- Logistic Regression Classifier (Maximum Entropy)
- Support Vector Classification
- K-nearest Neighbors Classifier
- Multi-layer Perceptron Classifier
- Gaussian Naive Bayes Classifier

Results (1/2)

- Feature selection seems to lower the AUC score of the model, while increasing accuracy;
- Most times, Kaggle results follow the same trend as the predicted AUC score. When this doesn’t happen, it’s an indicator of overfitting;
- The best algorithm was LogisticRegression with a AUC score of 0.97 and a precision recall value of 0.8 and 0.9 on average;
- Other AUC scores:
    - MLPClassifier: 0.84
    - KNeighbors: 1.0 (but it is extremely overfitted, having a low score on Kaggle)
    - SVC: 0.5

Results (2/2)

	Logistic Regression	GaussianNaiveBayes	MLPClassifier	SVC
AUC	0.968	0.815	0.842	0.5
Precision	0.925	0.787	0.774	0.5
Recall	1.0	0.865	0.964	1.0
Accuracy	0.968	0.815	0.842	0.5
F1	0.969	0.824	0.859	0.66666

Conclusions, Limitations and Future Work

- Data normalization and feature engineering are crucial for obtaining better results;
- Many algorithms perform poorly due to low amount of data;
- Logistic regression had the best results;
- Study the effect of feature selection on the predictive power of the algorithm.

Annexes

Data Extraction

We modify the tables before passing them to the data preparation phase by:

- Transforming strings into datetime objects;
- Extracting the gender and birth date of a person from the birth_number column and calculating the client's age;
- Removing non-owner dispositions from the disposition table;
- Imputing missing values on the district table using a regressor;
- Creating ratios such as crime rate and ratio of small, medium and large cities per district;
- Dealing with null values by copying values from other columns;
- Multiplying withdrawal values by -1.

Data Aggregation

The aggregation operation creates new features representing statistics about:
* Transaction date: minimum (oldest transaction), maximum (latest transaction) and interval between transactions;
* Transaction type: counting of the various transaction types in the dataset, mean and standard deviation of transactions of a given type;
* Transaction amount: mean, minimum, maximum and standard deviation;
* Operation type: counting of the various operation types in the dataset, mean and standard deviation of operations of a given type.

The aggregation also creates features representing:
* The number of days since the last transaction;
* The ratio between the last balance and the amount borrowed;
* The ratio between the maximum balance and the amount borrowed;
* The number of days between the oldest and latest transaction in months;
* The average balance per month;
* The average amount in transactions in each month;
* The age of the borrower at the start of the loan;
* The age of the account in months.

After aggregating, the train dataset has 384 rows and 73 columns (a total of 36 new columns was created).

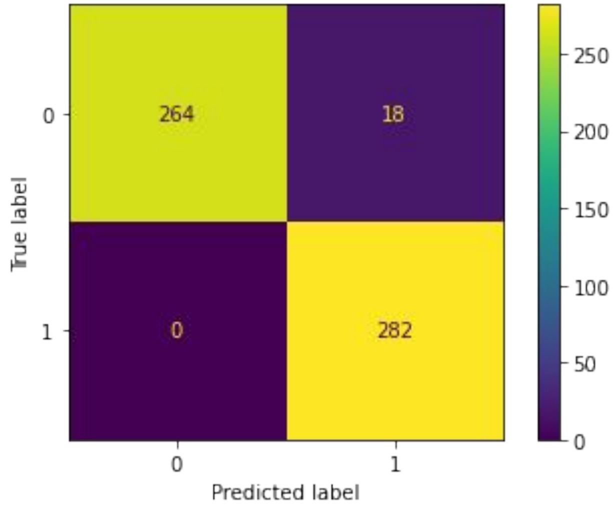
Model Selection Results

The best model in the training phase was Logistic Regression with a AUC score of 0.968. The same model, in the testing phase, scored 0.943 in the public leaderboard and 0.851 in the private leaderboard. The Multi-layer Perceptron Classifier was the best classifier for the private leaderboard with a score of 0.858 and a public leaderboard score of 0.918. Despite being slightly worse than the Logistic Regression model in the public leaderboard, it is also slightly better in the private leaderboard.

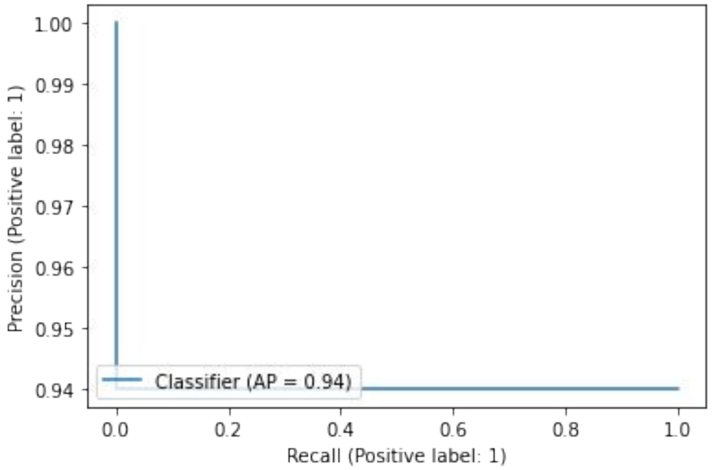
The Gaussian Naive Bayes Classifier came in third place at 0.771 in the private leaderboard. The other two algorithms, C-Support Vector Classification and K-Nearest Neighbors performed very poorly having an AUC score of 0.451 and 0.522 respectively in the private leaderboard.

Note: Other values and more information in the notebooks.

Model Selection - Logistic Regression (MaxEnt)

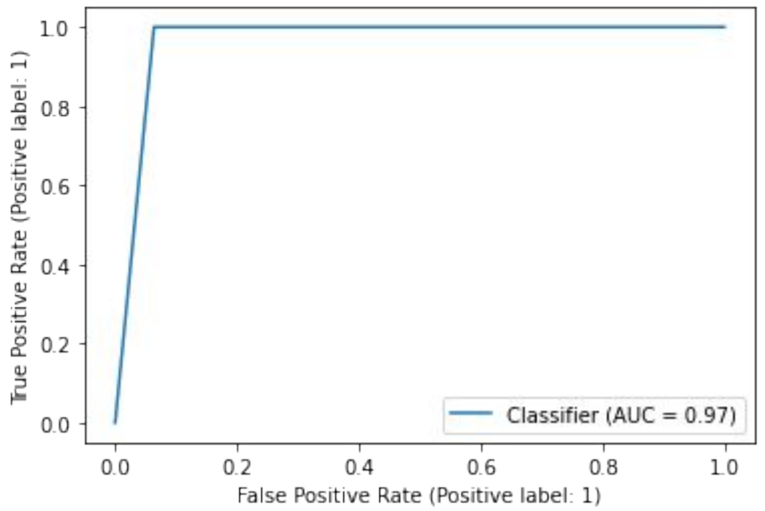


Confusion Matrix for the Logistic Regression algorithm



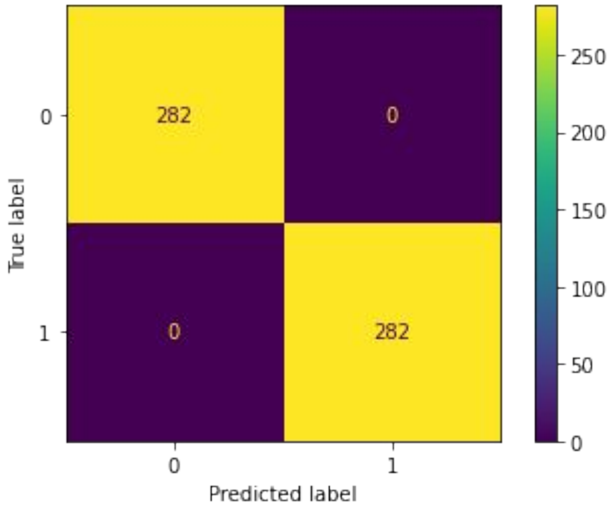
Precision-Recall curve for the Logistic Regression algorithm

Model Selection - Logistic Regression (MaxEnt)

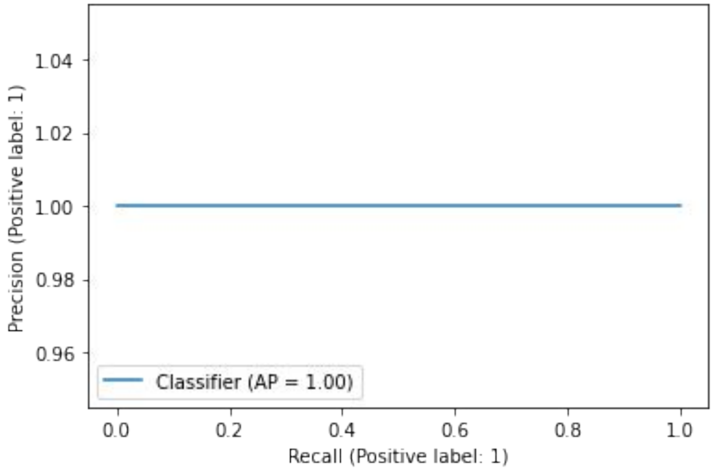


ROC curve for the Logistic Regression algorithm

Model Selection - KNearestNeighbors

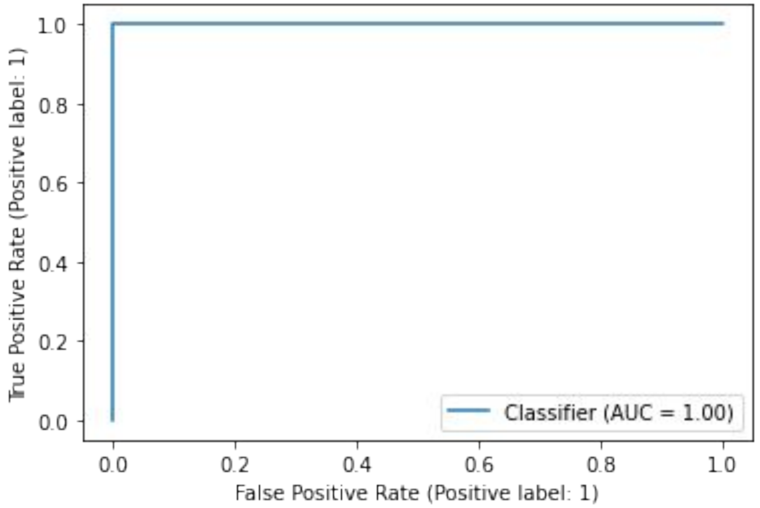


Confusion Matrix for the KNearestNeighbors algorithm



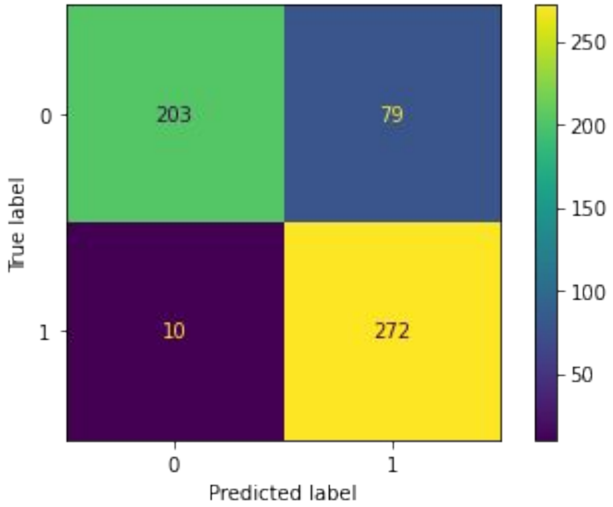
Precision-Recall curve for the KNearestNeighbors algorithm

Model Selection - KNearestNeighbors

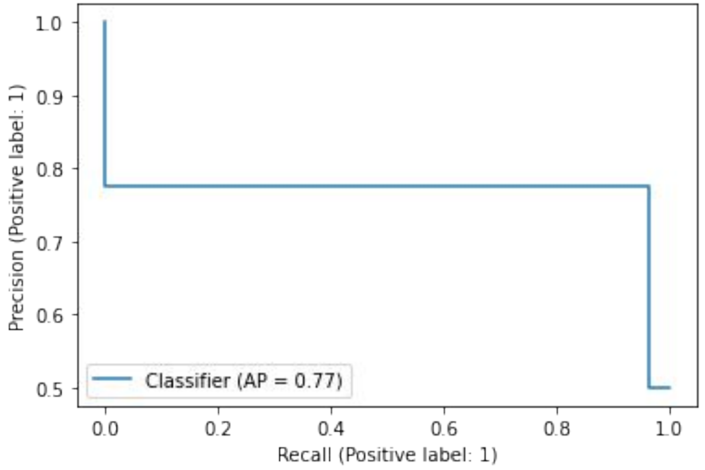


ROC curve for the KNearestNeighbors algorithm

Model Selection - MLPClassifier

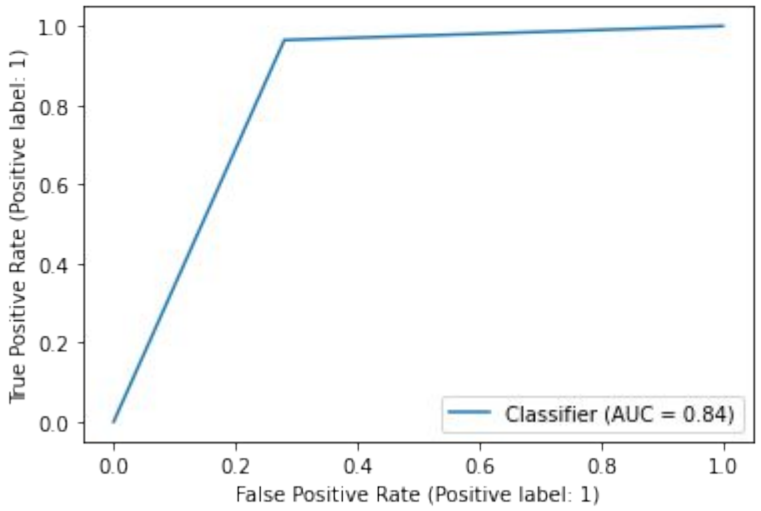


Confusion Matrix for the MLPClassifier algorithm



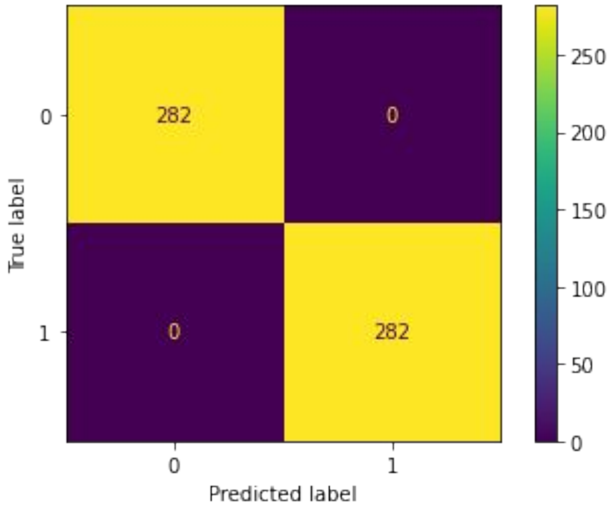
Precision-Recall curve for the MLPClassifier algorithm

Model Selection - MLPClassifier

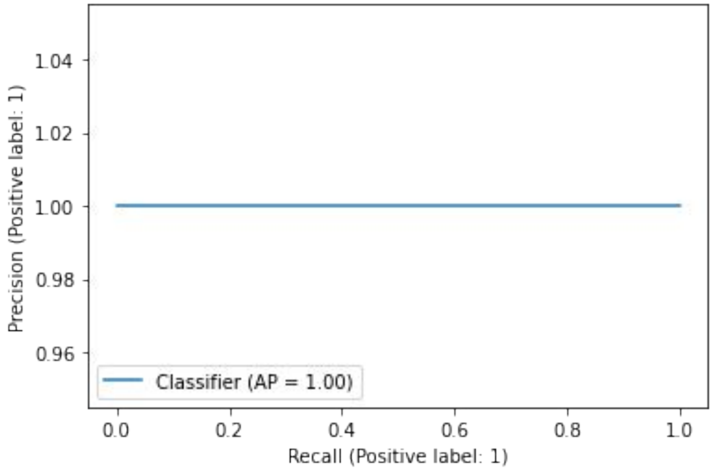


Confusion Matrix for the MLPClassifier algorithm

Model Selection - SVC (C-Support Vector Classification)

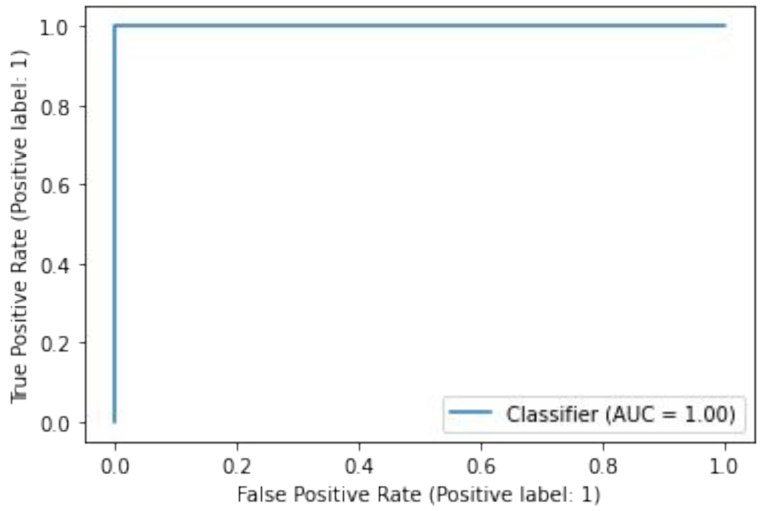


Confusion Matrix for the SVC algorithm



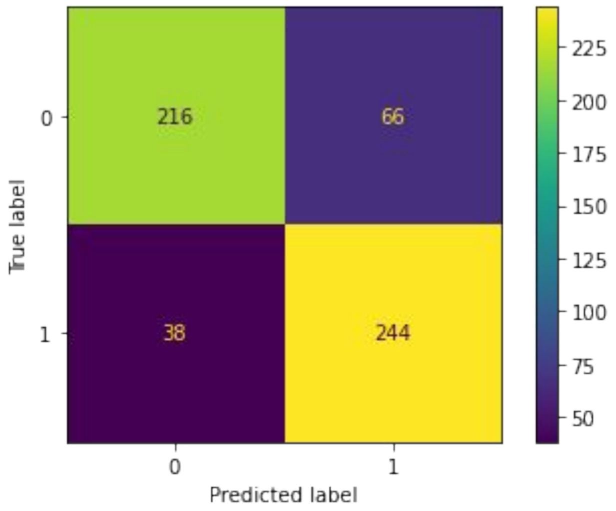
Precision-Recall curve for the SVC algorithm

Model Selection - SVC (C-Support Vector Classification)

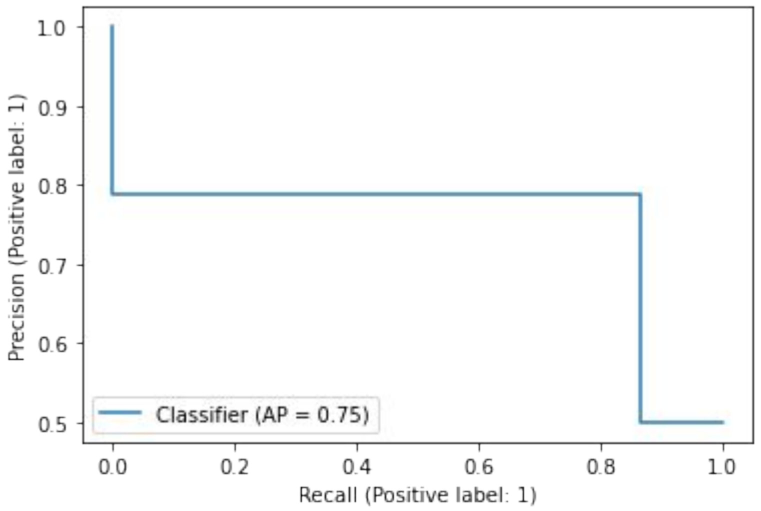


ROC curve for the KNearestNeighbors algorithm

Model Selection - Gaussian Naive Bayes

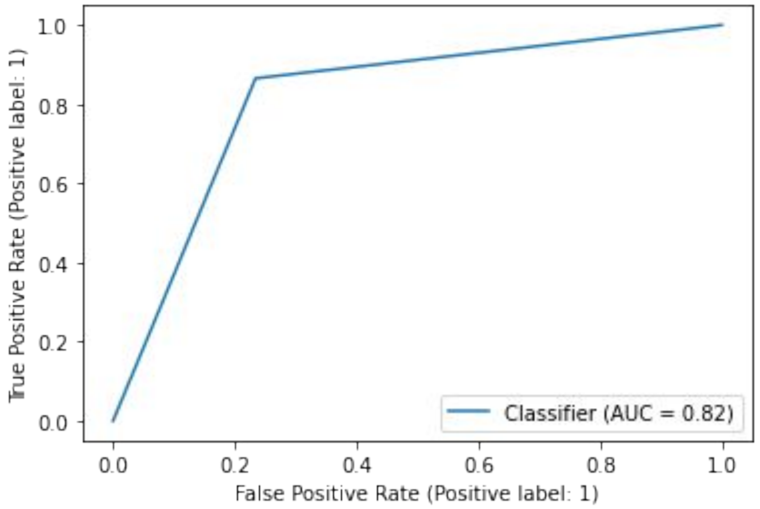


Confusion Matrix for the Gaussian Naive Bayes algorithm



Precision-Recall curve for the Gaussian Naive Bayes algorithm

Model Selection - Gaussian Naive Bayes


True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
Classifier (AUC = 0.82)