Data Mining - Banking

Aprendizagem Computacional

...

How to make an awesome model

Project by:

Joana Mesquita
João Costa
Miguel Freitas

Business Understanding

What seems to be the problem?

A bank in Czech Republic wants to lend money, but to whom?

To solve this question we shall implement a predictive system in order to:

- To help bank employees in decision-making when assessing risk in a loan request.
- To speed up the loan application process, by making an automatic profile analysis and risk assessment.
- To reduce loan defaulting by denying credit to clients we do not expect to be able to pay off.
- To reduce loan defaulting by anticipating loan default, and negotiate a change of terms with the client.
- To reduce the losses associated to loan defaulting by adjusting interest in proportion to the risk of defaulting (a client with a higher risk of defaulting will be made to pay more interests because banks want to take as much interest out of the client before he/she defaults).
- To give more favorable conditions to good clients by reducing interests in their loans, thus making the bank more attractive and competitive for those customers when compared to other banks.

Definition of Business Goals

- A conceded loan that fails is considerably worse than a not conceded loan that could have potentially succeed, therefore we should pay close attention to False Positives in detriment of False Negatives by using tests following this same philosophy.
- Reduce defaulting significantly
- Do not significantly reduce credit to good clients
- Helping employees reduce credit analysis time.

Translation of business goals into data mining goals

The positive case is the one where the borrower can not pay back his loan.

We settled on the following goals:

- We will create a model that should yield an *Area Under the Curve* of at least 0.75
- Give credit to at least 95% of the good clients (keep true negative rate above 95%, or keep false positive rate below 5%)

Data Understanding

What exactly does our data represent?

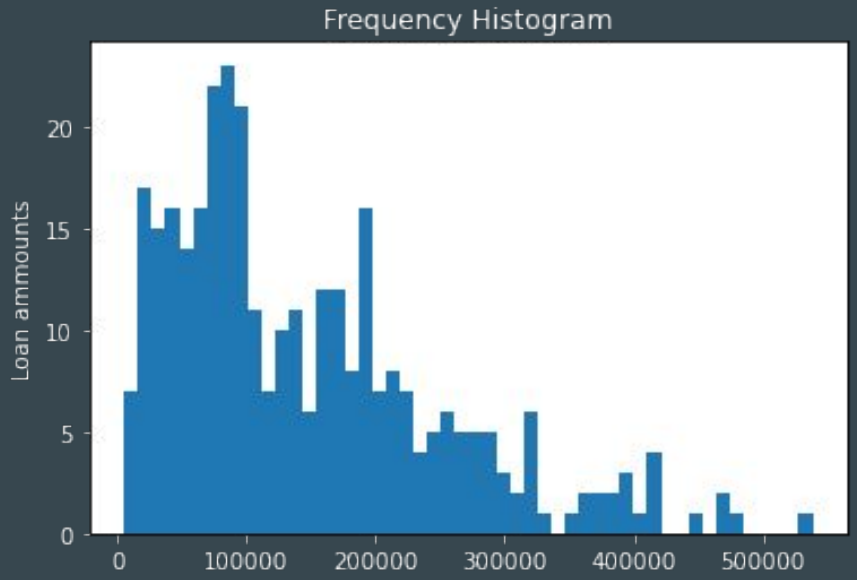
Data Size

Sample contains 328 loans

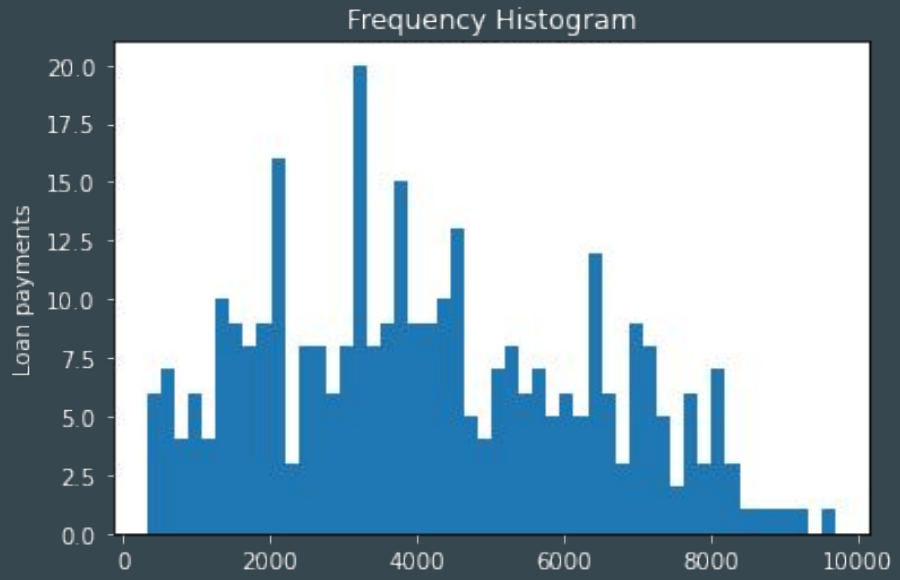
86% of the data refers to loans that were paid off.

We are dealing with a very imbalanced data set.

Data distribution - loan amounts

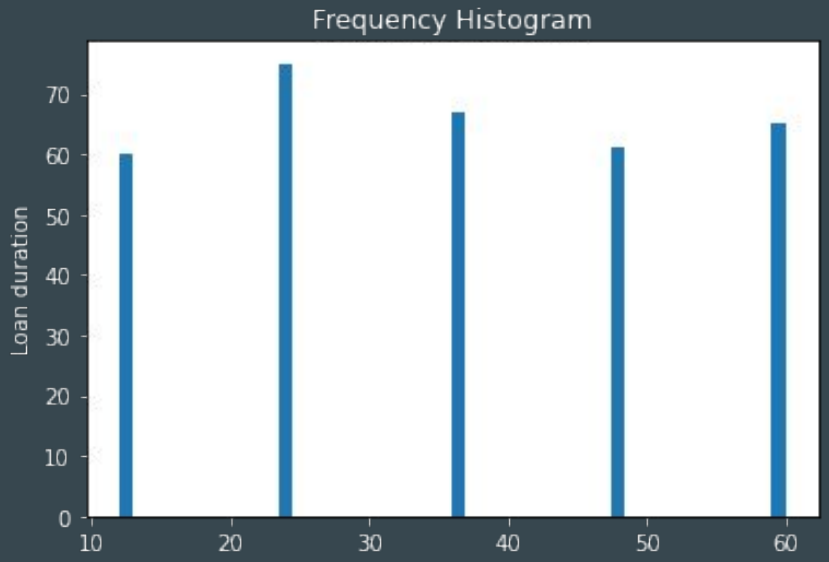


There are more loans with lower total amounts

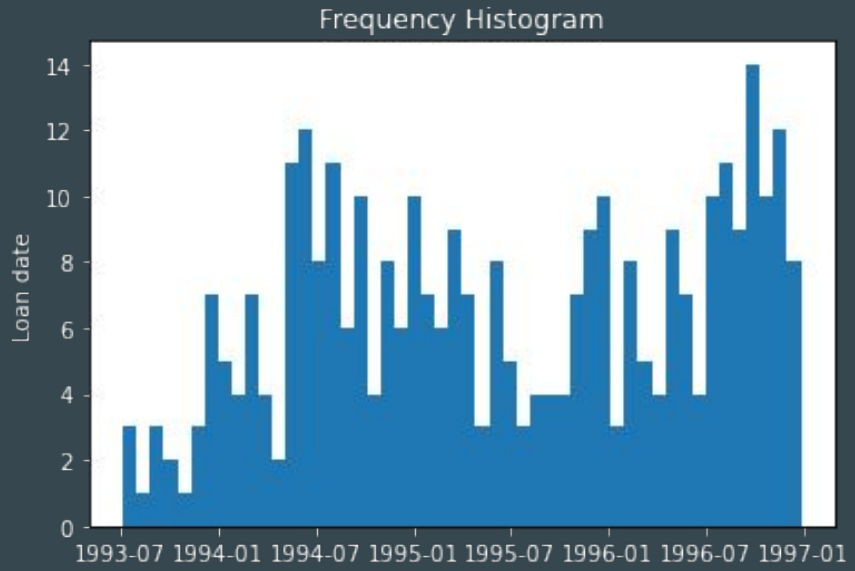


There are more loans with periodic payments 300 and 4000

Data distribution - loan dates

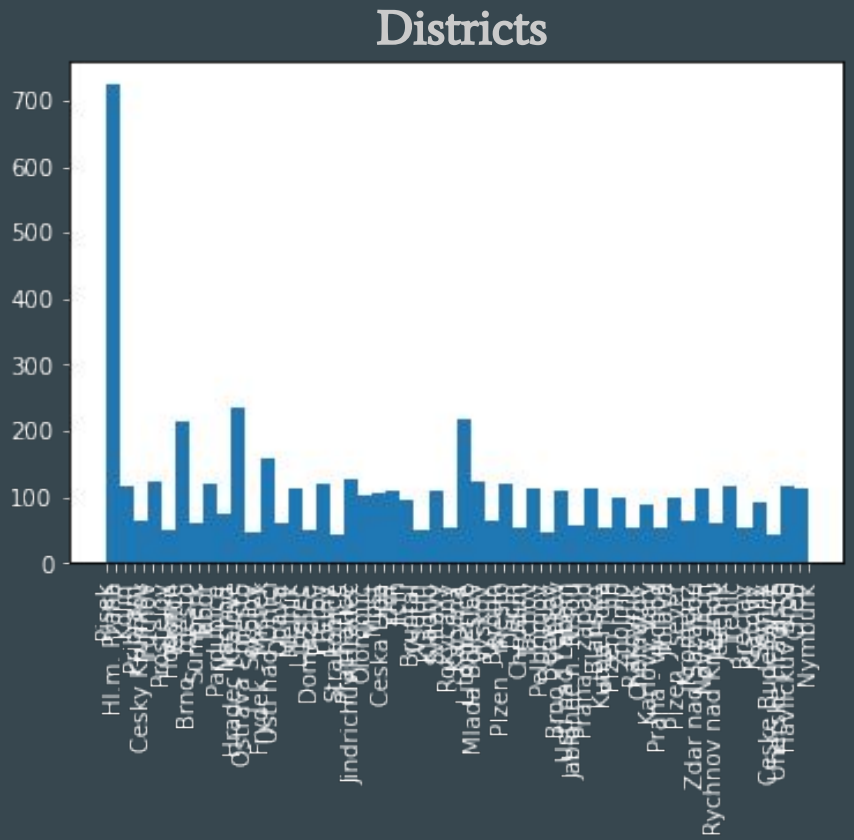


The amount of loans per duration seems to be mostly even



There seems to be two big peaks in loans in July 1994 and July 1996

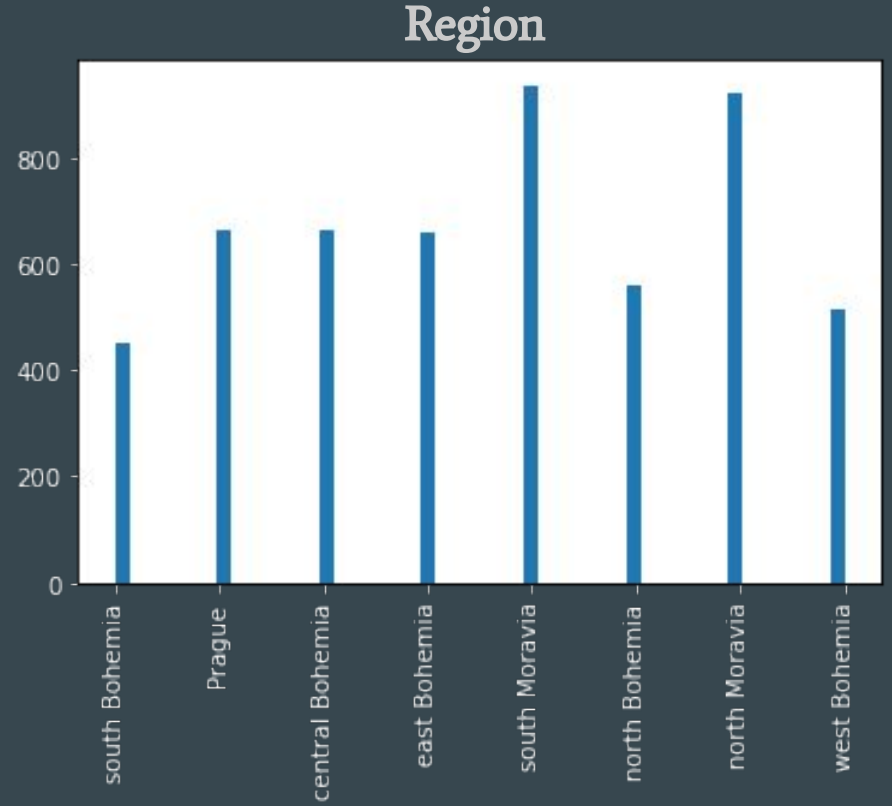
Data distribution - Demographic Information



Number of districts: 77

District with the most loans : Hl.m. Praha (Expected since this is the capital)

Data distribution - Demographic Information



While the regions of south Moravia and North Moravia have the most loans the amounts seem to be mostly even between regions

Others - Client age at the time of the loan

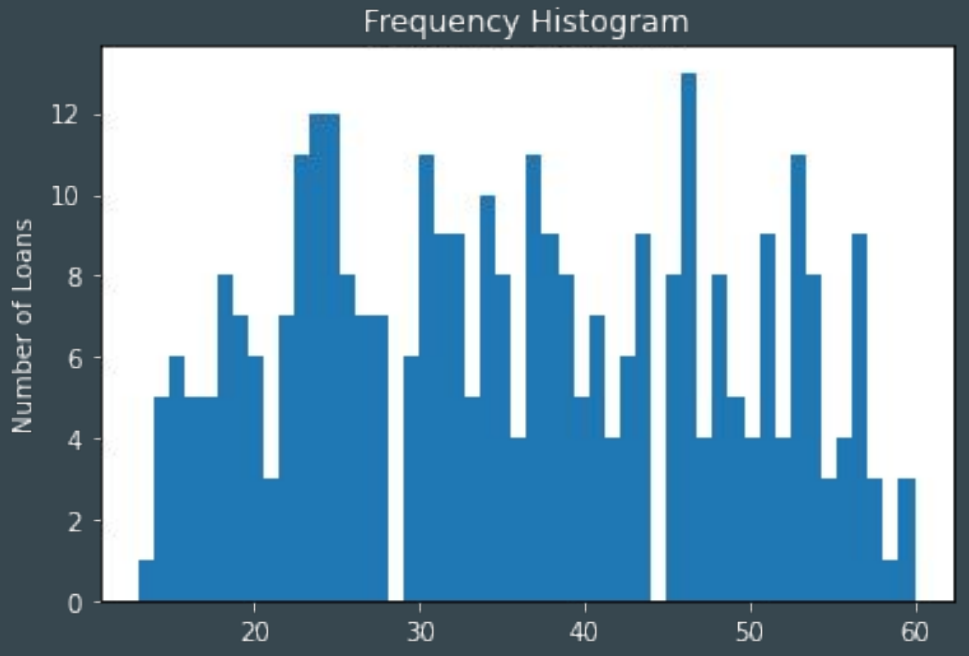
Youngest: 13

Oldest: 60

Average: 35.75609756097561

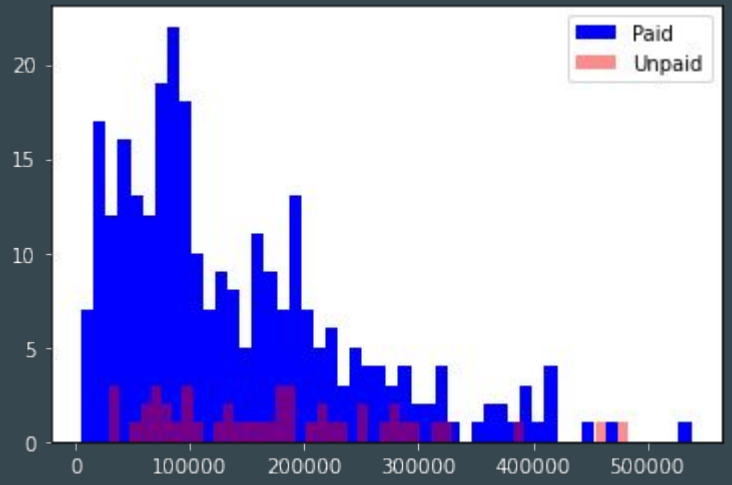
Number of people younger than 18: 22, making up 6.71% of loans

Age that asks for the most loans: 46 with 13 loans. Making up 3.96% of loans

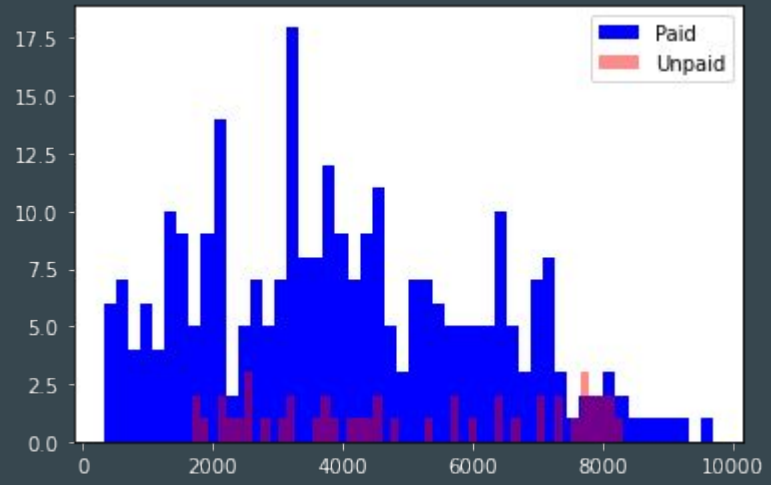


Paid vs Unpaid loans

Loan amount

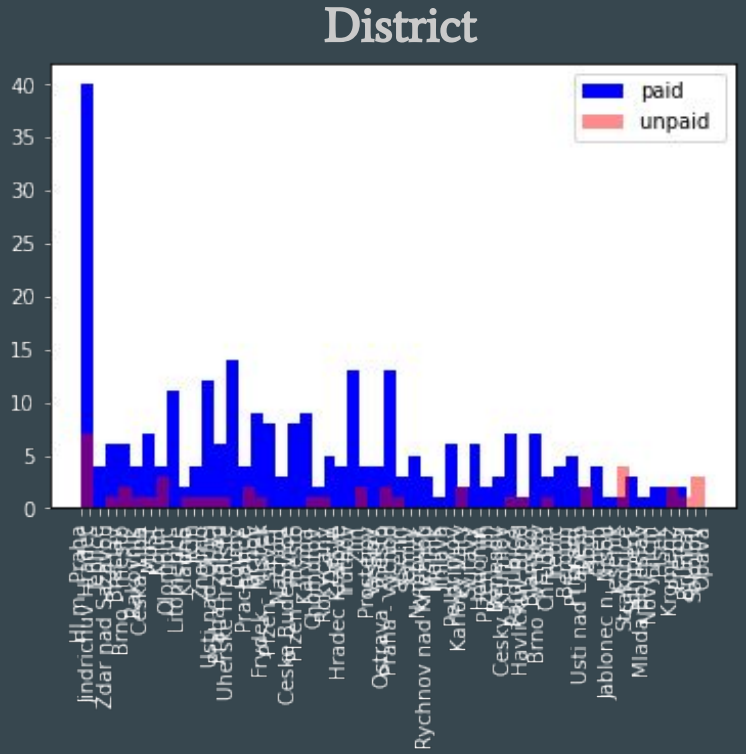


Loan payments



Neither seem to affect if loans are payed or not too much

Paid vs Unpaid loans



73 districts total

Tabor - 2/4 (50.0%)

Breclav - 1/2 (50.0%)

Jesenik - 1/3 (33.33%)

Strakonice - 2/4 (50.0%)

Sokolov - 0/1 (0.0%)

Bruntal - 1/2 (50.0%)

Opava - 0/2 (0.0%)

Kromeriz - 2/4 (50.0%)

Karlovy Vary - 2/4 (50.0%)

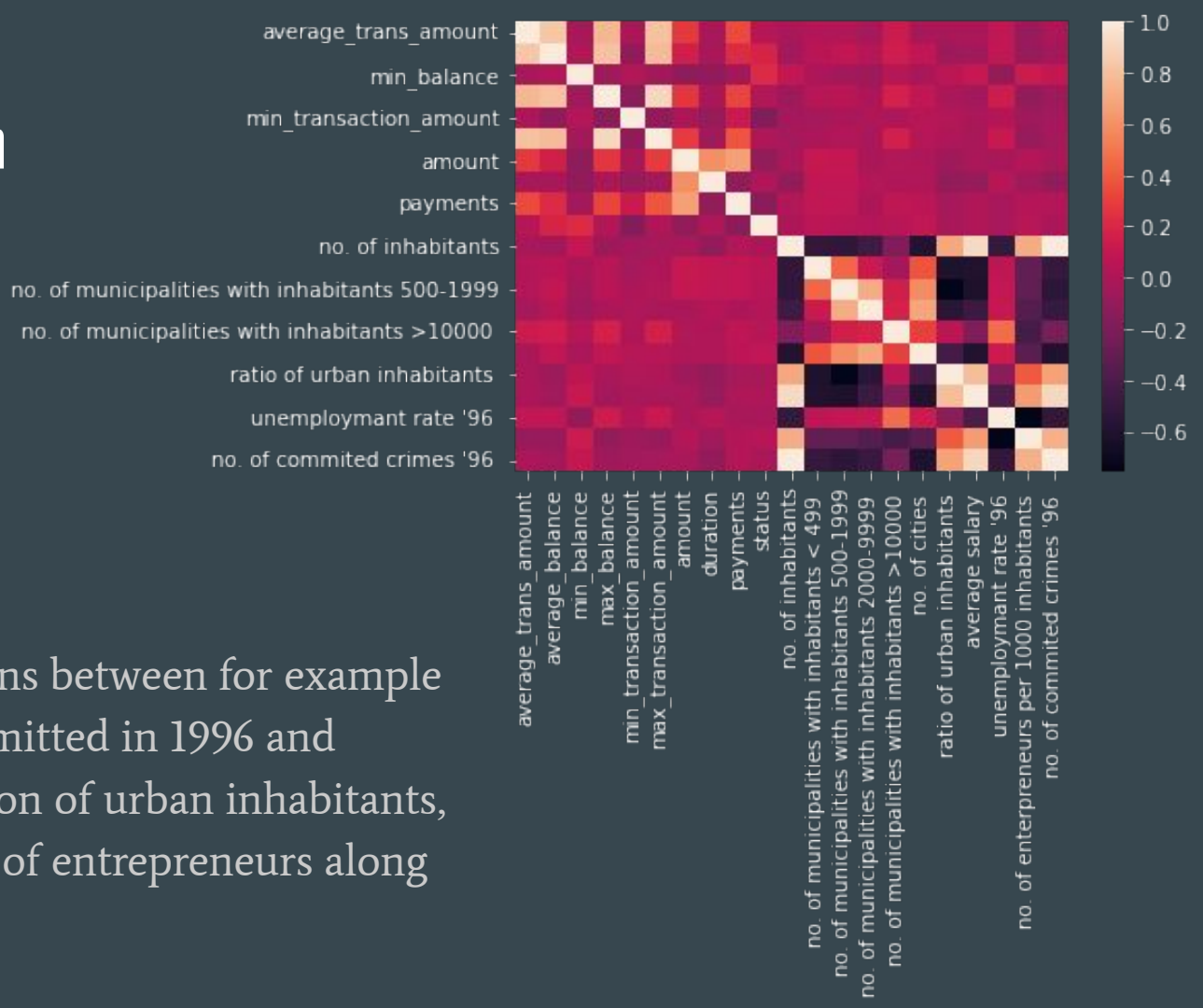
Havlickuv Brod - 1/2 (50.0%)

While some districts have very low percentages of loans paid the small amount of data brings into question if these values should be taken at face value.

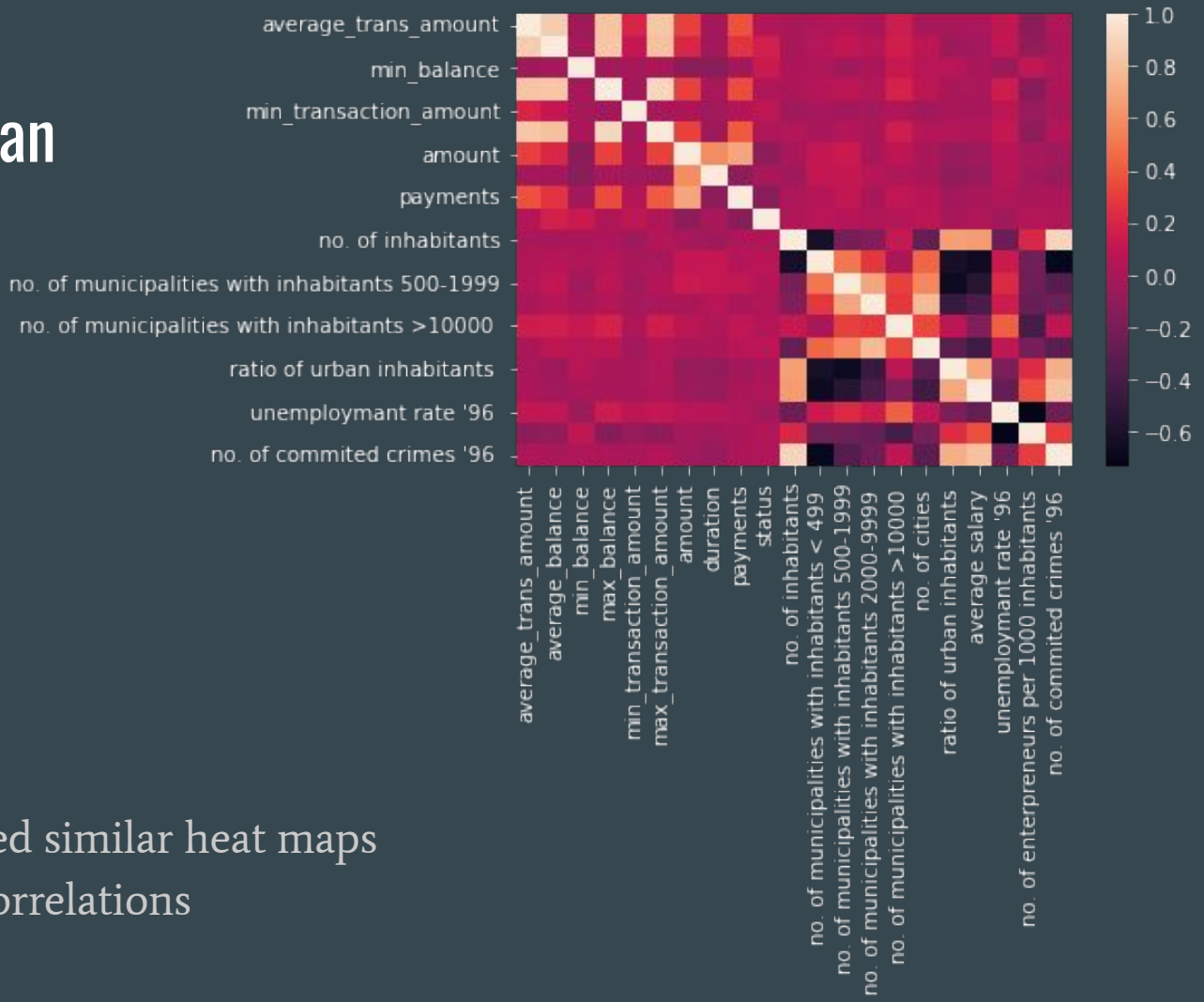
Correlation - Pearson

In order to get rid of redundant features, Pearson and Spearman correlations were calculated between most features.

We can see some correlations between for example the number of crimes committed in 1996 and number of inhabitants, ration of urban inhabitants, average salary and number of entrepreneurs along with many others.



Correlation - Spearman



The two algorithms produced similar heat maps hinting at similar variable correlations

Data Understanding: Some General Conclusions

- Analysis of our results revealed that gender had a near zero impact on the person's ability to pay back their loan.
- People's age distribution turned out sort of as expected: at the time of the loan most were between their 20's and 40's
- Maximum of a single loan per account
- The vast majority of people did not own a credit card
- No interest rate was applied by the bank, meaning that the loan amount equaled 'payments' * 'duration'

What to do with this information?

Our process of feature selection will correctly discard most of this useless information.

Data Processing

Making our data usable!

Data Integration and Preparation

The client's gender was extracted from the 'birth_number' feature and this feature was subsequently transformed in an actual date format of yyyy/mm/dd.

Additionally, other dates in the data were also converted to that format.

Only kept rows with type 'OWNER' in disposition table as these were the only ones who could take out loans.

Dropped all ids.

Process each table and merge them into one.

Redundancy : Limitations in applying our correlations

Because of the necessary assumptions that Pearson's correlation coefficient makes about our data namely that :

- Both variables are on an interval or ratio level of measurement
- Data from both variables follow normal distributions
- Data has no outliers
- Expecting a linear relationship between the two variables

We came to the conclusion that these assumptions could not be applied to our variables so we stuck to the Spearman correlation which has much less barriers to its application.

In the end, our criteria for removal of redundant features was a Spearman correlation superior to 0.7. This lead to very few features being discarded, however we found through trial and error that this value gave us the best overall metrics, as discarding too many features resulted in significantly worse performances.

Outliers

Transactions with 0€ ?

Children asking for hundreds of dollars of loans?

Discussion

Children asking for hundreds of dollars in loans could be explained by this simply being done by their guardians in their name however we still considered it a bit suspect.

Addressing the outliers

Although we considered initially that these outliers could be truthful deviations from the data norm, we ultimately decided to remove them because doing so resulted in an improvement of the overall metrics.

Dimensions of data quality

- Accuracy - ★★★★
  We found data that was clearly incorrect, but for the most part the information was accurate as it seemed to reflect valid real life situations but there was no way to verify this.
- Completeness - ★★★★★
  There is data available for each loan regarding the associated account and the person/people who administer it (with the exception of some missing values in the district dataset)
- Validity - ★★
  Validity of the data is a bit questionable because of the existence of loans to certain accounts with owners under the age of 18 (in some cases no other person is associated to the account), which was not legally possible in 90's czech republic
- Uniqueness - ★★★★★
  We found no repeated data
- Timeliness - ★★★★★
  The loans we had to predict were fairly close in date to the ones in the training dataset
- Relevance - ★★★★
  Some features were not very relevant, but for the most part there was some relevance to them

Feature Engineering

In the account table we added:

- Average Transaction Amount
- Minimum Transaction Amount
- Maximum Transaction Amount
- Average Balance
- Minimum Balance
- Maximum Balance
- Age at Loan

Problems

Generated Nulls

↓

Not compatible with algorithms!

Missing Values

Substitute them by the average of the column

To preserve statistical characteristics

If the data is missing in one year, get the data from the next year

Most likely will not change significantly between the two years

Data transformation for compatibility with algorithms

We used CategoricalOneHotEncoder for our categorical variables (this creates new features for each possible category which is then either 0 or 1 according to whether or not that category is selected)

- frequency
- gender
- region

And non standardized data as well!

Apply StandardScaler to numerical features as this is a major requirement of most predictive models

These transformations not only assured that our models could now digest our data, but also improved significantly certain metrics of our models, nominally their fit times were greatly reduced.

Feature selection: Filter-based methods (using SelectKBest)

Applied only to the training set.

Chi-squared

The feature selection is only suited to categorical features or features having discrete data contained within it. The continuous features are not taken into account and therefore were not used while performing this test.

Anova Test

A feature selection technique is most suited to filter features wherein categorical and continuous data is involved. We decided to apply it, even though it is a type of parametric test, which means it assumes a normal distribution of data forming a bell shaped curve, which we could not assure for all of our data as seen in the Data Understanding section.

Feature selection: Wrapper-based methods

Applied only to the training set.

We used two algorithms to achieve this:

- SelectFromModel - Meta-transformer for selecting features based on importance weights.
- SequentialFeatureSelector - adds (forward selection) or removes (backward selection) features to form a feature subset in a greedy fashion. At each stage, this estimator chooses the best feature to add or remove based on the cross-validation score of an estimator.

Passing different models through these algorithms also yielded different results, but we could still compose a collection of the most relevant features.

In the end, features selected from RandomForestClassifier's importance weights(using both the above algorithms) proved to improve performance across all our models.

We concluded that these methods yielded features that worked best with our models so, even though we experimented with filter-based methods, all our Kaggle submissions ended up being done with the wrapper-based methods.

Feature selection: most relevant features

- average_trans_amount
- average_balance
- min_transaction_amount
- max_transaction_amount
- min_balance
- max_balance
- age_at_loan
- duration
- payments
- no. of entrepreneurs per 1000 inhabitants
- no. of municipalities with inhabitants < 499
- no. of committed crimes ‘96

We were glad to see that all of our engineered features made it to the list of most relevant features, proving their relevancy.

What do we do about the imbalanced data?

We tried SMOTE

applied only to the training data set

SMOTE aims to generate elements on the minority side of the data by getting two elements from it and generating one with values in between the two. In our case we got two failed loans and generated a new one with the standard SMOTE algorithm, also generating a new client and attributing them a random district according to the percentage of failed loans in each district, this being the custom part of our algorithm

↓

Improved the results very slightly

Data Predictions

ML Pipeline

Problem definition

“Predict whether a client will take an unsuccessful loan so the bank can avoid lending money that it won’t be able to get back”

It’s a classification problem where the output variable is either 1 (loan was successful) or -1 (loan wasn’t successful)

Split the data into test and training

train_test_split(X, y, test_size=0.2, stratify=y, shuffle=True)

Use “stratify” to guarantee an equal distribution of the less common class in both training and test sets

Hyperparameter Tuning

Used sklearn’s **GridSearch with CV** with **StratifiedKFold** to test the best combination of parameters for each model

After acknowledging the Supervised Learning nature of this problem we settled on the following algorithms

- Logistic Regression

We fit a S shaped curve, called Sigmoid, to our observations. This sigmoid is then used to assign a probability to every entry. Depending on whether or not that probability is under or above a certain threshold, the model classifies it as one of two categories.

- KNeighborsClassifier

Calculate the Euclidean distance of K number of neighbors to the new point and count the number of neighbours in each category to decide which category it will assign

- RandomForestClassifier

Classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.

Used models

- SVC

The goal of the SVC algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

- GaussianNB

Assumes that the occurrence of a certain feature is independent of the occurrence of other features.

Ensemble classifier

Built with the models from before, with their optimized hyperparameters

- StackingClassifier → **best performing**

The output from multiple classifiers is passed as an input to a meta-classifier for the task of the final classification, which in our case was a Logistic Regression.

Model evaluation

Cross-validate the fitted model with **RepeatedStratifiedKFold** with 5 splits and extract metrics. AUC and F1-Score were given the most priority.

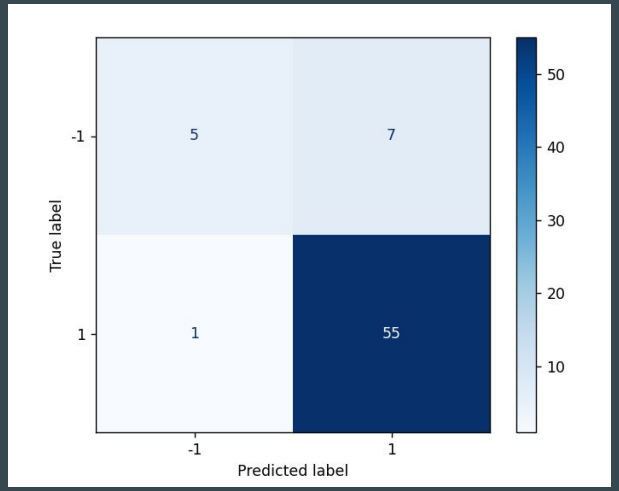
- **AUC**
  Area of the previous plot, the closest to one the better
- Accuracy
  The ratio of times our model got the loan outcomes right
- Balanced Accuracy
  The same as accuracy, but balanced according to the amount of failed and successful loans
- Recall
  How many failed loans the model identified from all the failed loans there are
- Precision
  How many loans were actually going to fail from the loans the model said were going to fail
- **F1 Score**
  A metric to express the accuracy and recall in one number only

Model evaluation - Results (for optimized hyperparameters)

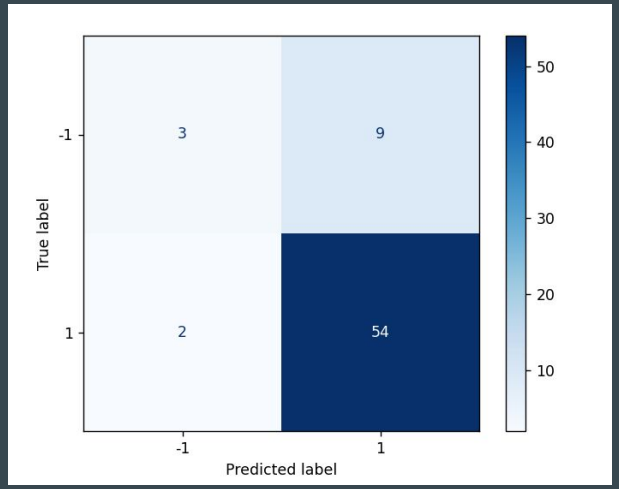
	AUC	Accuracy	Balanced Accuracy	F1 Score	Precision	Average Fit Times
Random Forest	0.688491	0.878089	0.565556	0.933796	0.875855	0.316283
GaussianNB	0.657498	0.844429	0.553204	0.913768	0.874009	0.00365334
KNeighborsClassifier	0.539797	0.847506	0.492888	0.917364	0.858007	0.00270581
LogisticRegression	0.746006	0.871888	0.608705	0.929028	0.88742	0.422747
SVC	0.693454	0.874918	0.626951	0.930405	0.892733	116.717
StackingClassifier	0.738989	0.875012	0.573158	0.931817	0.877949	2761.47

Confusion matrices (for optimized hyperparameters)

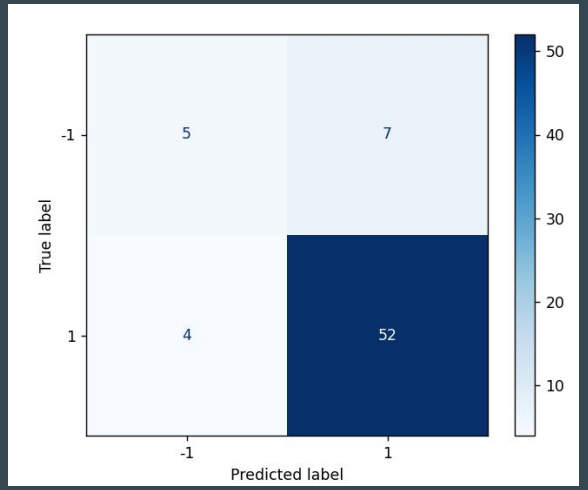
Random Forest



Logistic Regression

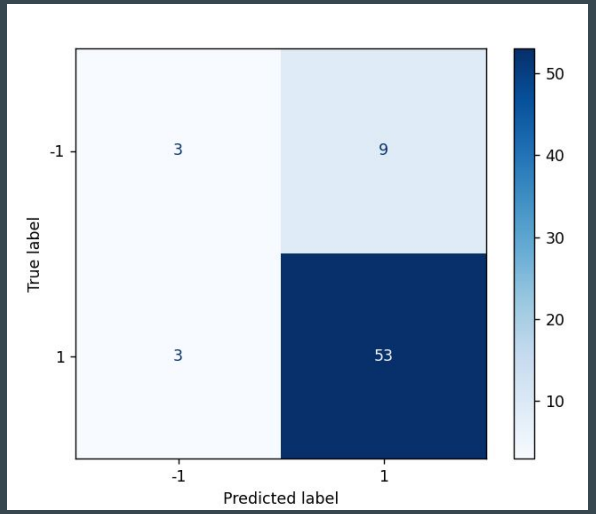


SVC

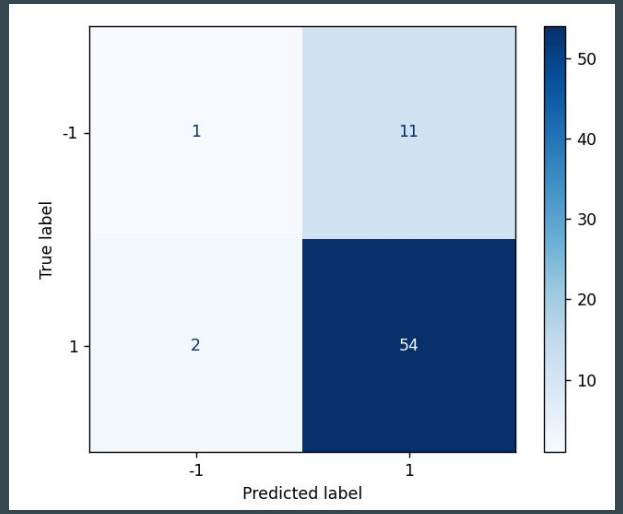


Confusion matrices (for optimized hyperparameters)

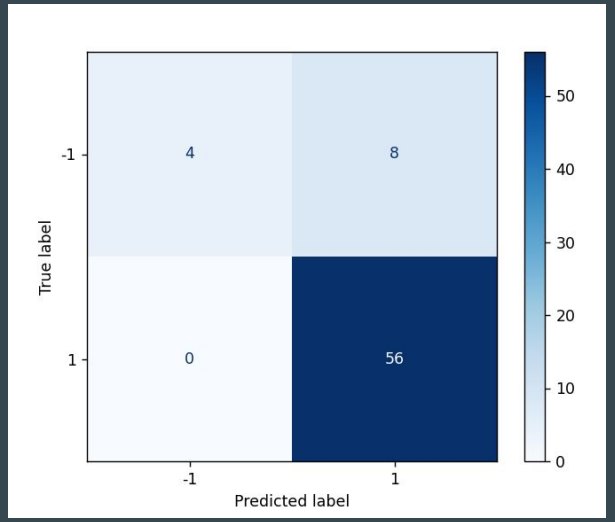
Gaussian NB



K Neighbours Classifier



Stacking Classifier



Data Predictions: Conclusions

- We concluded that feature selection and data standardization helped dramatically in reducing fit times for our models.

- Feature selection and SMOTE increased our F1-score and AUC scores, which were the ones we were most concerned about, but only slightly.

- In our given context, feature engineering and feature selection have a far greater impact on the overall metrics than hyperparameter tuning of an algorithm or its choice.

- Defined data mining goals (AUC > 0.75) were achieved.

Limitations and Future work

The main issues experienced by the group while working on this project were:

- The small amount of available data for training (328) when compared to the amount of data we had to predict (354).
- The data being very unbalanced (86% of data corresponded to paid loans).
- The lack of overall inexperience in machine learning pipelines.

Future work could involve experimenting with undersampling methods as opposed to SMOTE oversampling and trying out different algorithms.

Individual factor:
Joana Mesquita - 1/3
João Costa - 1/3
Miguel Freitas - 1/3