Loan Default Prediction

Youssef Ashraf Kandil
Computer Engineering
The American University in Cairo
youssefkandil@aucegypt.edu

Abdullah Mohamed Kassem
Computer Engineering
The American University in Cairo
Abdullahkassem@aucegypt.edu

Dataset description and why we choose it:
https://www.kaggle.com/yasserh/loan-default-dataset

This dataset contains many features, both numeric and categorical, about the loan and the person requesting the loan. It has around 32 features and 149k samples.

We chose this dataset because it had a good amount of features that are relevant to whether the loan will be defaulted or not. There are also over 130k sample points which is enough for training and testing our models. Also the dataset has a diverse range of features, some are numerical and categorical. By considering the datasets presented in the project proposal, the number of features in the chosen dataset compared to other datasets is the most reasonable, as some datasets had over 200 features, which would make feature analysis and engineering very challenging. This dataset had some randomly missing values and outliers, unlike other datasets, which means that it is realistic as data collected from real life scenarios usually have missing data and outliers. According to the kaggle author, this dataset is subject to strong multicollinearity, which would be a good challenge for us to handle.

Data preprocessing and cleaning steps:
First we read the csv file into a pandas dataframe. Then our next step was to encode the data. We noticed that all our string data had discrete values, so we performed label encoding on all the string columns. And kept track of the encoded previous names and their corresponding code. Then we plotted our first heat map to show the correlation between the features and each other. What we did there will be discussed later in this paper. Then we did a lot of data visualization which was followed by dropping features we deemed insignificant. After that we did data normalization on the columns who needed it. Then we drew Box plots of the data before and after scaling to ensure that nothing went wrong. Then null values were handled as it will be explained later. Finally, we split the dataset into 4 partitions. 2 of them were for training, X_train had the features and y_train had the label and another 2 for testing X_test and y_test.

Dataset features analysis:

Identifying correlations:

For the sake of analyzing the dataset more deeply and excluding irrelevant features, we decided to use Pandas library to manifest the correlations lying within the dataset. To study the correlation between all features and specifically with the focus label “Status”, we created a correlation table using Pandas, and plotted a heatmap. By observation, some features showed very significant correlation with each other, in terms of having significant correlation indices on the heatmap. Thus, excluding one of each pair of features was our methodology to avoid any bias effects that would happen. The following features were dropped for such reason and they can be referred to in the “dropping insignificant features” section :

- year
- Interest_rate_spread
- property_value

Figure 1

Dealing with missing values:

As mentioned before, having null values was part of our reasoning to choose the dataset, since it resembles the genuinity of the data, however, these missing values could be very problematic during the training phases. So first, we looked for the frequency of null occurrence in each feature (column), then sorted them descendingly, and the following was observed:

Upfront_charges	26.664425
rate_of_interest	24.509989
LTV	10.155378
income	6.154571
loan_limit	2.249277
approv_in_adv	0.610749

Figure 2

As the largest frequency is only 26%, no specific feature could be dropped without really affecting the integrity of the data. Thus, we decided to focus on separate data points instead of full features. And we had to follow one of the following approaches to handle NULLs:

- Drop all rows with existing null values
- Replace Nulls with averages of features
- Replace Nulls with a driven formula for each features

Starting with the first approach, we luckily got the following results, noting that the remaining dataset is still large enough for training.

income	7.057820
loan_limit	2.186594
approv_in_adv	0.596177
submission_of_application	0.140331
age	0.140331
loan_purpose	0.097223
Neg_ammortization	0.077962
term	0.022013
Secured_by	0.000000

Figure 3

As illustrated in figure three, when the rows having null values associated with the feature of highest null frequency (Upfront charges) were dropped, most of the null values of the data set were dropped as well. Which means that this successfully excluded the corrupted data from the dataset. It is obvious that using the other two approaches was useless, because applying induced values for too many features of the same row will highly bias the dataset.

Semantic importance / relevance:

By researching in the banking field, we created an induced set of features that will not be dropped or manipulated significantly, as they have very high theoretical relevance to the focus label, and high probability to be driving features for the machine learning models we will apply. Such features are:

- Loan_limit
- Loan_amount
- Rate_of_interest
- LTV
- Income
- Upfront charges
- Credit score / type
- interest_only
- Credit_Worthiness
- loan_purpose

Feature analysis:

We drew the histograms of all the features to view their distributions. There we found how some features like the year only had one value, and we learned about the distributions of the features.

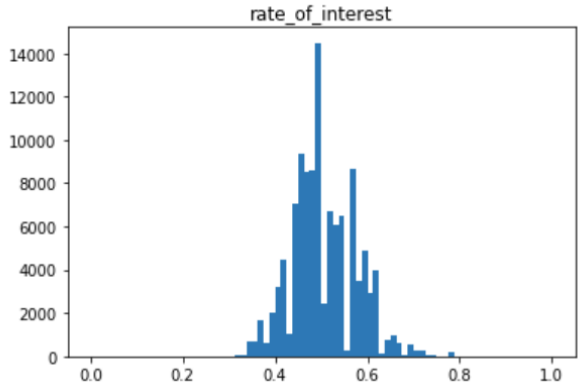

Figure 4

Here you can see that the rate_of_interest follows the trend of normal distribution curves, as well as a few other ones. Which means that standardization could be a valid option for data scaling, however since there were not many features which showed similar patterns we decided to go with min-max normalization instead.

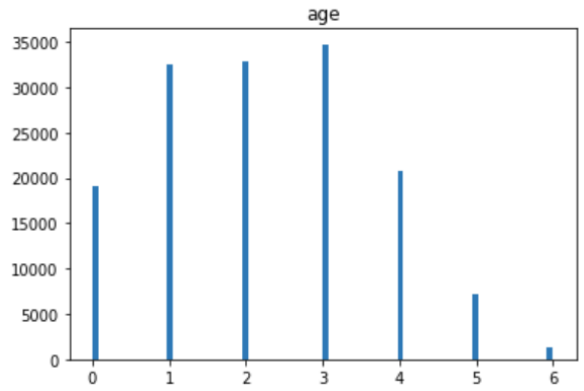

Figure 5

The Histograms made us realize how underrepresented some age groups are, which is something that we need to take into consideration when looking at our results.

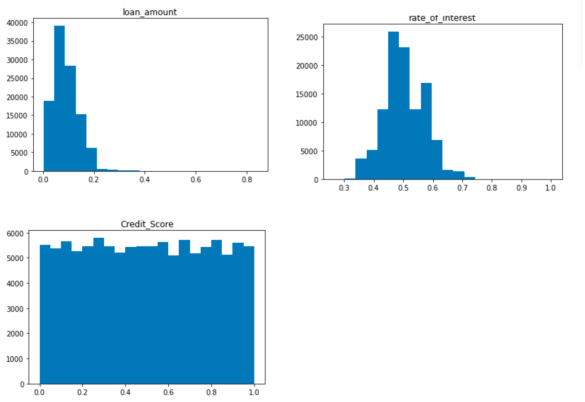

Figure 6

Here, for instance, some of the predicted to be relevant features show reasonably diverse distributions of values, which supports our intention of preserving them.

Summary:

As described in the above section, the preprocessing is crucial to have reasonable results in the training and testing phases. Here is a list of all the data preprocessing and cleaning steps which were applied for the data set. A full insight can be found in the attached notebook.

- reading csv file
- Creating Encoding function
- Creating correlation matrix
- plotting heatmap
- Dropping highly correlated (isomorphic) features
- Dropping insignificant features (based on the criteria described in the previous section)
- Label encoding for categorical features
- Normalization of features' domains
- Data visualization (to infer relations)
- Analyzing the null value distribution in the dataset
- Dropping rows with significant null values
- Reassess the null value distribution in the dataset
- Plotting relations with respect to Status label
- Splitting Training and Testing Data

Post cleaning:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 109028 entries, 2 to 148669
Data columns (total 29 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   loan_limit                106644 non-null  object
 1   Gender                    109028 non-null  object
 2   approv_in_adv             108378 non-null  object
 3   loan_type                 109028 non-null  object
 4   loan_purpose              108922 non-null  object
 5   Credit_Worthiness         109028 non-null  object
 6   open_credit               109028 non-null  object
 7   business_or_commercial    109028 non-null  object
 8   loan_amount               109028 non-null  float64
 9   rate_of_interest          109028 non-null  float64
 10  Upfront_charges           109028 non-null  float64
 11  term                      109004 non-null  float64
 12  Neg_ammortization         108943 non-null  object
 13  interest_only             109028 non-null  object
 14  lump_sum_payment          109028 non-null  object
 15  construction_type         109028 non-null  object
 16  occupancy_type            109028 non-null  object
 17  Secured_by                109028 non-null  object
 18  total_units               109028 non-null  object
 19  income                    101333 non-null  float64
...
 26  Region                    109028 non-null  object
 27  Security_Type             109028 non-null  object
 28  Status                    109028 non-null  int64
dtypes: float64(7), int64(1), object(21)
memory usage: 25.0+ MB
```

Figure 7

The dataset is now ready for training and the features have been decided and prepared.

The list of the final chosen features is:

['loan_limit', 'Gender', 'approv_in_adv', 'loan_type', 'loan_purpose','Credit_Worthiness','open_credit', business_or_commercial', 'loan_amount', 'rate_of_interest', 'Upfront_charges','term','Neg_ammortization','interest_only' ,'lump_sum_payment','construction_type','occupancy_type', 'Secured_by','total_units','income','credit_type','Credit_Scor e','co-applicant_credit_type','age','submission_of_applicatio n', 'LTV', 'Region', 'Security_Type']

Why these features were chosen and others were dropped was discussed in previous sections. Although features and samples were dropped the total dataset size is still a little over 109k and still has 28 features which is over the minimum of 10 features. This leaves us with 87k for training and 22k for testing.

Dataset reference:

- https://www.kaggle.com/yasserh/loan-default-dataset