LOAN DEFAULTER ML PROJECT

A DESCRIPTIVE PROCEDURE

Ahmed Abdalrahiem Mahmoud
a.abdalrahiem@gmail.com
+201148639332

About the Author

Ahmed Abdalrahiem is an ML Engineer with hands-on experience in supply chain and waste management applications. He has a great understanding and is skilled in exploratory data analysis, Model Training, and evaluation.

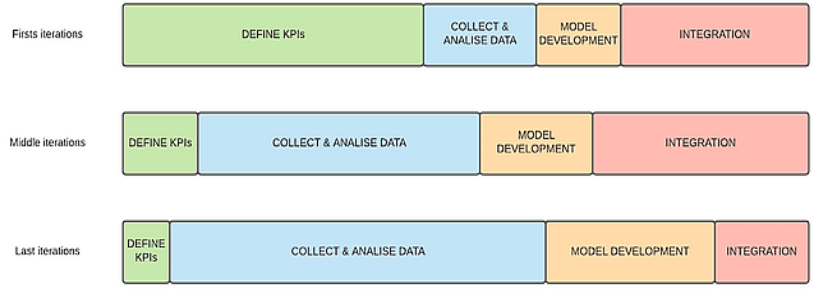
Email: a.abdalrahiem@gmail.com

Phone: +201148639332

Contents

1. Business KPIs

While machine learning projects are challenging and complex, it's always advisable to start them by identifying the goal of the project and the way it serves the business container in which it lives. One of the ways is defining the business goals along with the Key Performance Indicators (KPI).



For a financial company, there are plenty of KPIs that can be utilized for the purpose of measuring the company's performance. Here, we are listing some of them (Most of them are related to loan applications):

KPI 1. **Gross Profit Margin:** It measures the profitability of the company by quantifying the net income.
KPI 2. **Average Cycle Time:** This KPI is used to optimize the loan process. It measures the average time needed from when a client applies an application until the loan/credit is approved.
KPI 3. **Application Approval Rate:** This KPI measures the performance of the application process. It represents the number of approved applications compared to the total number of valid applications.
KPI 4. **Profit Per Loan:** This KPI gives an intuition about how each loan comes with a benefit to the company. It measures the total profit compared to the number of Disbursed loans.
KPI 5. **Incomplete Application Rate:** It gives an understanding of how good and easy the application process is. It measures the number of incomplete applications compared to the total number of processed applications.
KPI 6. **Customer Acquisition Cost:** This KPI measures the cost of getting new customers to have approved loan applications. It compares the total costs spent on marketing to the total number of new customers.

2. Problem Statement

In order to achieve a high value in the predefined KPIs for the loan application process, we have to come up with a solution that can improve most of them.

When the company receives a loan application, It has to decide either to approve the application or decline it. When a client defaults on paying his loan, this may introduce a decrease in the company's profit (KPI 1).

The most valuable solution is to design a data-driven model that is able to identify if an applicant will repay his loan or will default. During the application process, if we can quickly classify if the client is a defaulter or a repair, this will optimize different aspects of the process (KPI 2:KPI 5).

This way, we will improve 5 KPIs out of the previously defined 6 KPIs (KPIs A, B, C, D)!

3. Dataset

3.1 Data Description:

As for most of the ML classification tasks, there should be labeled data for the model to be trained and tested on.

The dataset was found on Kaggle and was proposed to apply exploratory data analysis on it and identify patterns of clients who apply to a lending company or a company that offers credit services.

The data is combined of three CSV files:

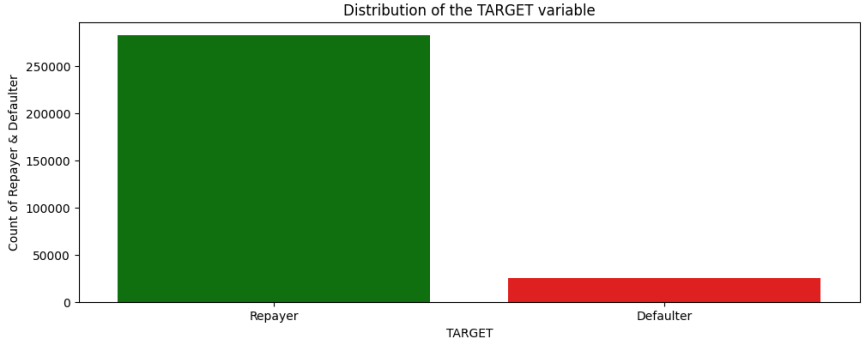
CSV File	Description
application_data.csv	• This file has data columns that contain information captured from the loan application filled in with the client at the time of applying for the loan/credit. • Every row is a unique loan application and has a unique ID SK_ID_CURR. • One of the columns has the target variable "TARGET" which contains a binary value. If the value is 1, it means that the client was a defaulter; if the value is 0, he repaid the loan. • The data contains 307511 rows/applications and 122 variables (including the target variable). • The data contains mixed variable types (categorical and continuous types). • Some of the categorical variables in the data are multi-class.
previous_data.csv	• This file contains data of previous applications for clients who have IDs in the application_data file. • Every row is a unique loan application and has a unique ID SK_ID_PREV. • Each current loan application in the application_data file may have **more than** one previous application for the same client. • Each current loan application in the application_data file may have **no** previous application for the same client. • Columns (variables) in previous applications are not identical to those in the application data. • The data contains 1670214 rows/applications and 37 variables (including the target variable).
columns_description.csv	This file contains a description for each column/variable in both the application data and previous data.

3.2 Dataset problems:

The dataset has some problems that shall be assessed, those problems are as follows:

1- Data imbalance

- **Problem**: The number of applicants who repaid their loans in the dataset is way greater than the number of people who repaid. This



- **Effect:** This may result in the classifier being naïve and giving too many predictions as Repayer (majority class) for instances that should have been classified as Defaulter (minority class).
- **Solutions:** In our work, we will try to penalize the cost function for wrong classifications to the minority class. We will also try to shift the classification threshold.

2- Data missingness

- **Problem:** The dataset is found to be having missing values in 3 forms: NAN, strange classes (XNA, Unknown, others). Around 97% of the application data instances have missing values!
- **Effect:** Some patterns in the data will not be present for the learning algorithm to learn from.
- **Solutions:** In the project's IPYNP notebook, we have explored the validity of some imputation methods for both categorical and continuous variables. Some of the classes possess missingness.

are kept, and some variables are removed. Some are filled with a relevant value.

3- Unknown Data Collection Process

- **Problem:** The process of collection, processing, and aggregation of the dataset is unknown.
- **Effect:** If we had known the process of data collection, we would have known why some data were missing. We can answer some questions like: Are the missing data missing at random (MAR) or missing completely at random (MCAR) or missing not at random (MNAR)? If we could identify the process of missingness, we would have realized the most reasonable way of data imputation.
- **Solution:** Effectiveness of imputation methods shall be verified against the real-life performance of the model (i.e., we cannot know if the imputation method is reasonable by using model performance on the test dataset. Because the test dataset is imputed the same way as for the training dataset).

4- Unknown Data Origin

- **Problem:** The geographical coverage area of the data is not defined. Note: The application data set has a variable with the name "FONDKAPREMONT"; which translates to "Capital Repair Fund" in Russian; this may mean that the data was collected in Russia.
- **Effect:** Learning from data that has patterns related to people from a specific geographical area may be tricky, especially in financial problems. People from different countries may have different philosophies when it comes to money.
- **Solutions:** We may try to collect data from the geographical area in which the model will be deployed. Since data collection is complex and takes much time, we may try to study variables that reflect the client's behavior and compare them to any data (even if small) related to the same geographical area.

5- Missing Data Description

- **Problem:** Despite that the data is sourced with a description of its columns' meaning, there are some vague descriptions as in: [FONDKAPREMONT, YEARS_BEGINEXPLUATATION_MODE, EMERGENCYSTATE_MODE].
- **Effect:** Missing description of data variables may lead to missing some patterns understanding during the exploratory data analysis stage.

6- Old Coverage Date

- **Problem:** According to Kaggle, the data is collected in the time between the start of 2018 and mid-2020**.**

- **Effect:** Since we are in late 2023, some of the variables may have had a drift in their distribution. which shall be given more attention.
- **Solutions:** Newer data shall be collected to study if there is a drift in any of the variables' distribution.

4. Plan

As discussed in the previous section, the dataset has a great degree of missingness. Also, as per the construction of the application data and previous data sets: it's been decided to disregard the previous applications dataset for the following reasons:

1. Aggregating both datasets will add another level of missingness because not each column in previous data will have a row in application data. Handling this missingness in addition to the existing level of missingness in the application data will be tricky and will require a lot of effort and analysis.
2. The dataset will become larger, which will make training more computationally expensive. This is not wise given the shallow timeframe of the project.

5. Data preprocessing

In the data preprocessing step, I tried to clean the data as much as possible and also made it ready for ML model training. As you may know, cleaning the data is a critical step and may lead to a dramatic enhancement in the model's performance. Following is the description of some of the steps carried out during cleaning, but one should return to the cleaning stage after training to enhance the performance.

5.1 Handling Missing Values

After inspection of the application data columns, it seems to have 2 forms of missing data: NAN values, and strange classes (XNA, Unknown, Others).

It was important to study the way data was collected and aggregated. This way, we can be sure if data was Missing Not at Random (MNAR), Missing Completely at Random (MCAR), or Missing at Random (MAR). Example: In the application process, is it possible that a client doesn't answer some questions (optional)? If so, this means that (assuming the aggregation method brought no missingness) maybe some variables are MNAR (e.g.: people with small age don't tell house area because they don't know)

Following is our strategy for missing data imputation:

1- Strange Classes (XNA, Unknown, Others):

- If the number of rows having a strange/missing class is very small (4 or 5), we apply listwise deletion (delete the rows).
- For larger numbers, I assume that the strange class may be carrying important information and shall be kept in the dataset. Example: the "Other" class in the ORGANIZATION_TYPE variable may represent all clients with odd organization types (isn't easily categorized).

2- NAN values (General):

- I didn't apply listwise deletion (the easiest) because 97% of the rows have missing values.
- Pairwise deletion is applied to variables with missingness more than 54%, because imputing those variables may lead to teaching the model incorrect information and patterns.
- After the missingness correlation test using dendrograms, the missingness of some variables was found to be correlated with the existence of some other variables. This leads us to suspect that those variables are missed at random (MAR) because their missingness is related to the value of other variables in the multivariate space.

3- NAN values (for Categorical Variables):

- Mode imputation is applied to categorical variables with low missing value occurrence in comparison to other classes.
- The missing value is considered as an additional class for categorical variables with high missing value occurrence in comparison to other classes.

4- NAN values (for Continuous Variables):

- A multivariate imputation algorithm (KNN Imputation) was preferred to account for the multivariate space while filling in the missing entries. But, due to limited computational resources, another simple method was used.
- Median imputation is considered for continuous variables. While this may not be the best way.

5.2 Duplicated Rows Elimination

Duplicated instances will result in the model giving more attention to the duplicated row. The application data was found to contain more duplicated rows.

5.3 Handling Negative Values

Some variables of the application data have negative values. Those variables represent days before the application date, which makes sense for them to have only negative values. Here is my strategy for handling negative variables:

- Unreasonable values (e.g.: 1000 years) we removed.
- All those variables are converted to positive values and divided by 365 (converted to years.)

5.4 Encoding Categorical Variables

In addition to continuous variables, the dataset has 2 types of categorical data: binary and nominal. Categorical variables shall be encoded before model training.

One-hot encoding was applied to all categorical variables. **But this was a mistake**, encoding shall have been only applied to categorical variables with nominal values and the binary categorical variables (0/1, Y/N) shall be kept in a Boolean format.

6. Exploratory Data Analysis

We applied data analysis to some variables, but time was not enough to analyze all the important variables.

7. Feature Selection

While our dataset is of mixed variables (Categorical and Continuous), I had two options:

1- Use two algorithms, one to select important categorical variables (e.g.: Chi-square Test), and the other to select important continuous variables (e.g.: Pearson Correlation).
2- Use an algorithm that accepts both variables: Random Forest.
3- Use features as hyperparameters (i.e., cross-validation training with different sets of features to know what features are considered important). In our case, this would be the best but the most time-consuming.

I decided to use random forest because it seems reasonable, **but option 1 may be examined to check if it improves the model performance.**

Note: Random forests will mark correlated features to be equally important. At first glance, removing all correlated variables (having the same information) and leaving one of them may seem reasonable.

But, when dropping any of the correlated variables, its importance power is divided among all other variables, and this may cause a decrease in model performance. One solution to this is using **recursive feature elimination**. I haven't tried this option because of time, but it is worth trying.

8. Model training

People may consider model training as the last step in a ML project, **but it's not!** Machine learning is an iterative process, and one may need to go back to any stage (especially Data Preprocessing) and tweak it in order to enhance model performance.

I trained a random forest model on the data. The model showed bad performance (low recall, high precision) because the data is imbalanced. I tried to tweak/tune the model in order to enhance its performance by penalizing the cost function for the minority class and decreasing the classification threshold.

I wanted to try training SVM and ANN, but given the time frame, I stopped the training process.

9. Evaluation

Since the data has a high level of missingness, the imputation we have done may drift variable distribution or reflect a pattern that doesn't exist in reality. Thus, Evaluating the model using a test dataset that is a subset of the original imputed data will tell nothing about the real model performance. The model shall be tested using other real-life data.

10. Future Enhancement:

1- Newer data shall be collected so that we can identify variables that have a drift in their distribution. This way we can have a clear decision on model retraining.
2- Data preprocessing:
    - Some dropped variables (with missing values) may be having important information, further examination shall be brought to those variables. If they are important, more data for them shall be collected.
    - Encoding shall be applied to nominal categorical variables only (correcting the mistake).
3- Train another model (SVM for example).
4- To increase model performance, we may tweak our feature selection procedure:
    - Assess the effect on performance when using **recursive feature elimination.**
    - Assess the effect on performance when using two algorithms, one to select important categorical variables (e.g.: Chi-square Test), and the other to select important continuous variables (e.g.: Pearson Correlation).
    - Try to decrease the number of selected features. Hence, we can try other computationally expensive training models.
5- Design pipelines to load the data, preprocess the data and make predictions.