Credit Card Approval Prediction - Fall 2024 (CIE 417)

Omar Awad 202100699

Supervision

Dr.Omar Fahmy
Eng. Youssef Mohamed
Eng. Engy Ayman Sayed

Communications and Information Engineering
Zewail City of Science and Technology

Contents

1 Introduction

The task of predicting whether a credit card applicant is a 'Good' or 'Bad' client is a significant aspect of the banking and financial industry. In this project, we work with a dataset containing both client information and their loan payment history to build a machine learning model capable of classifying applicants into one of the two categories: Good or Bad.

2 Dataset Overview

The dataset consists of two tables:

- **Table 1 (Client Information)**: Contains personal information about the clients such as marital status, gender, family size, and annual income.
- **Table 2 (Loan Payment History)**: Includes each client's loan payment history, which will be used to assign the "Good" or "Bad" label.

These two tables can be merged based on the 'ID' field for further analysis.

2.1 Dataset Source

The dataset is publicly available on Kaggle at the following link: Credit Card Approval Prediction Dataset.

3 Task Objective

The objective of this project is to build a machine learning model that can predict whether a client is likely to be a good or bad borrower. The challenge lies in generating the "Good" or "Bad" labels, as these are not explicitly defined in the dataset. You will need to create labels based on loan repayment behaviors and other client information.

4 Steps for the Project

4.1 Step 1: Understand the Dataset

Familiarize yourself with the two tables and their features:

- Table 1: Client Information.
- Table 2: Loan Payment History.

4.2 Step 2: Label Creation

- Using the loan payment history from Table 2, create a label for each client: "Good" if the client has no late payments, and "Bad" if they have missed one or more payments.

4.3 Step 3: Data Integration

- Merge Table 1 and Table 2 based on the ‘ID‘ field, and add the newly created labels as a column in the integrated dataset.

4.4 Step 4: Data Preprocessing

- Clean the data by handling missing values and outliers.
- Encode categorical variables using techniques like one-hot encoding.

4.5 Step 5: Criteria for ’Good’ and ’Bad’ Clients

In this step, I define the criteria for classifying clients as either ’Good’ or ’Bad’. The classification is based on the loan payment history, with the following scoring system:

Each loan payment status is assigned a score:

- ’C’ (No Credit) = -1 (negative score, indicating non-compliance).
- ’0’ (No Dues) = 0 (neutral score, indicating no issue).
- ’1’ (Paid on Time) = +1 (positive score, indicating timely payment).
- ’2’ (Paid with Delay) = +2 (moderate positive score, indicating slight delay).
- ’3’ (Major Delay) = +3 (more significant delay, indicating increasing risk).
- ’4’ (Critical Delay) = +4 (indicating serious risk of default).
- ’5’ (Default) = +5 (high risk of default).

The `calculate_score_avr` function calculates the average score for each client’s loan repayment history. The higher the score, the more likely the client is categorized as ’Good’. Conversely, a lower score suggests the client is ’Bad’.

4.5.1 Function Summary:

The `calculate_score_avr` function takes a list of loan payment statuses and calculates the average score for each status. For each status in the list:

- It assigns a score based on the criteria above.
- The scores are then summed and averaged by dividing by the number of statuses in the list.

Finally, a decision threshold is chosen to classify clients. After several iterations of testing different thresholds, I choose a threshold of 0.1 to determine the final classification:

- If the calculated average score is less than 0.1, the client is classified as ’Good’.
- If the average score is greater 0.1, the client is classified as ’Bad’.

4.6 Step 6: Model Training

- Train supervised models such as:

- Random Forest
- Logistic Regression
- knn nearest neighbor
- DecisionTreeClassifier

- Use these models to classify the clients as ”Good” or ”Bad”.

4.7 Step 7: Hyperparameter Tuning

- Use grid search and cross-validation to fine-tune the hyperparameters of each model to improve their performance.

4.8 Step 8: Model Comparison and Justification

- Compare the performance of the models. - Select the best-performing model based on evaluation metrics and provide a justification for the final selection. I found that Random Forest Model has the best fit for the data

- Accuracy 90%
- Precision(Good) 84%
- Precision(bad) 98%
- Recall(Good) 98%
- Recall(Bad) 82%
- F1-Score (Good) 91%
- F1-Score(Bad) 89%

5 Challenges to Consider

- **Imbalanced Data**: The dataset may have more ”Good” clients than ”Bad” clients. Techniques like oversampling, undersampling, or using balanced classifiers may be necessary.

6 Deliverables

- A Python or Jupyter notebook containing data preprocessing, model training, and evaluation code.
- A report that includes a confusion matrix and classification report for each model.
- A final report explaining the methodology, steps taken, results obtained, and justification for the final model selection.

7 References

- Over sampling method: YouTube

References