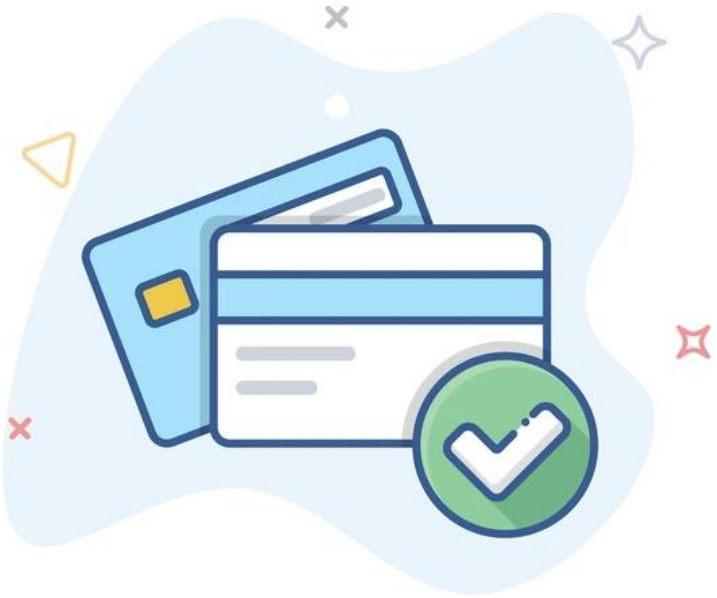
DS 861 Final Project:

Predicting Credit Card Approvals

Using Machine Learning

Spring 2024



Report by:

Vivid Liu

Ananyaa Shahi

Table of Contents

Introduction and Project Objectives

Commercial banks receive numerous credit card applications, many of which are rejected due to reasons such as high loan balances, low-income levels, or excessive inquiries on the applicant's credit report. Manually reviewing these applications is tedious, prone to errors, and time-consuming. This process can be automated using machine learning, a practice now common among commercial banks. In this project, we will use a credit card dataset containing features such as demographic information, financial history, and credit utilization to develop a machine learning model that predicts whether an applicant is a 'good' or 'bad' client for issuing a credit card.

About the Dataset

The dataset is sourced from Kaggle, containing 25128 rows and 21 columns. It is a credit card dataset aimed at predicting whether an applicant is eligible for a credit card. The data contains

various information such as demographic information, financial history, familial and marital status, employment, etc.

This dataset is interesting for several reasons. First, credit scoring is critical in the financial industry with significant implications for lenders and borrowers. Being able to accurately assess the creditworthiness of an applicant can help mitigate the risk of default and optimize lending decisions. Second, the dataset likely presents challenges such as imbalanced data, where the number of 'good' clients may far exceed the number of 'bad' clients or vice versa. This imbalance needs to be addressed to prevent the model from being biased towards the majority class. Finally, the task involves applying machine learning techniques such as correlation metrics, logistic regression, random forest, etc., which are fundamental concepts covered in our class. By working on this problem, we can gain practical experience in handling real-world datasets, dealing with imbalanced data, and building predictive models for binary classification tasks.

Literature Review

We looked at several solutions that have achieved the same objective of predicting credit card approvals using this dataset, employing a variety of machine learning techniques. Studies have utilized models such as Light Gradient Boosting Machine, Random Forest, and Naive Bayes to tackle this issue. Like our approach, these studies also focused on optimizing hyperparameters and have used methods like GridSearchCV for this purpose. Performance metrics like the AUC Curve and Confusion Matrix have been used to evaluate these models' effectiveness, measuring their ability to distinguish between 'good' and 'bad' credit applicants. These efforts highlight the dataset's ability in supporting the development of robust predictive models.

Aspect Explored

In this project, several aspects were explored to ensure a robust and effective model for classifying the target variable, 'Status'. The process began with Data Exploration, where we inspected the dataset for any missing values, duplicates, unique values, and the distribution of features. This step helped us identify potential issues and understand the dataset's structure.

During Data Preprocessing, the dataset was cleaned by handling outliers, converting categorical variables to numerical format, and dropping constant or irrelevant features. This step ensured that the data was in a suitable format for modeling. For instance, categorical variables such as job titles were converted using techniques like one-hot encoding.

Feature Engineering was another crucial aspect, where new categorical features were created by grouping job titles into broader categories. This step enhanced the model's ability to generalize and capture patterns across similar job titles.

To address the issue of imbalanced data, we employed the Synthetic Minority Over-sampling Technique (SMOTE). This technique oversampled the minority class to ensure a balanced distribution of the target variable, improving the model's performance on underrepresented classes.

Once SMOTE has been applied, we began working on our models, where logistic regression and random forest models were built and evaluated. GridSearchCV was used to tune hyperparameters and find the best model configurations. The models were then assessed based on their performance metrics. These steps ensured that the final models were well-built and optimized, leading to more accurate and reliable predictions for the target variable.

Techniques and Models Used

A combination of techniques and models were used to develop an effective classification system for the target variable. Before employing techniques and models, we performed Exploratory Data Analysis (EDA), where libraries such as pandas and visualization tools like matplotlib and seaborn were used to understand the data distribution and the characteristics of the various features. This step provided insights into the structure of the data and potential issues such as missing values and outliers.

Next, we moved on to Data Cleaning, removing constant columns, irrelevant columns like 'Applicant_ID', and handling spaces in categorical values. This ensured that the dataset follows a consistent format across all entries.

Then, we created box plots to detect outliers and extreme values were filtered out using percentile-based thresholds. This helped in maintaining a robust dataset by removing anomalies that could skew the model's performance.

To handle categorical variables, One-Hot Encoding was applied, converting these variables into a numerical format using pd.get_dummies. This step was essential so that the categorical data can be utilized by machine learning algorithms.

With the class imbalance in the dataset, we decided to use the Synthetic Minority Over-sampling Technique (SMOTE) to overcome the issue by oversampling. This technique was applied during the training phase to generate synthetic samples to balance class distribution, thereby enhancing the model's performance on the minority classes.

Once we have finished cleaning, preparing, and manipulating the dataset, we implemented two machine learning models: Logistic Regression and Random Forest Classifier. Logistic Regression was used with GridSearchCV to find the best hyperparameters. Similarly, a Random Forest Classifier was tuned using GridSearchCV to optimize model performance, considering various hyperparameters such as the number of estimators, maximum depth, and minimum samples split. To evaluate the performances of the two models, we assessed using a comprehensive set of metrics, including F1-score, precision, recall, accuracy, ROC-AUC, confusion matrix, and feature importance.

Logistic Regression

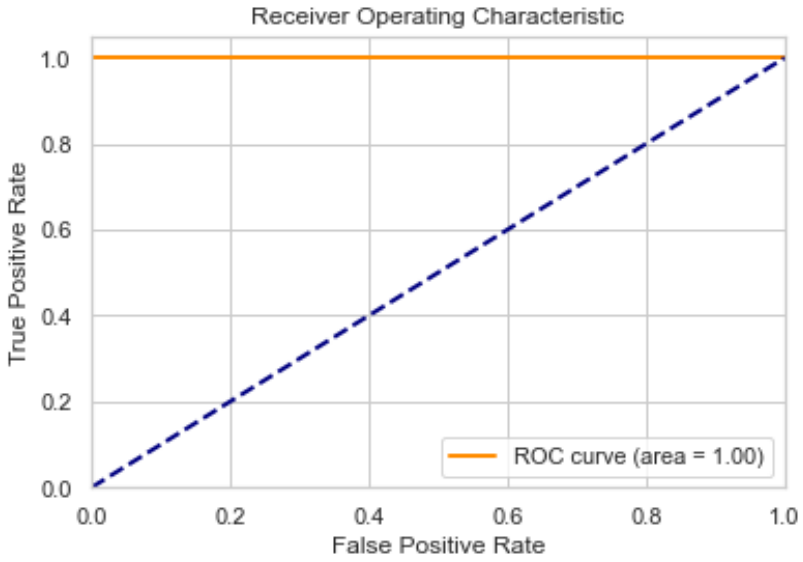
The main goal of using this method was to build a robust logistic regression model to predict the target variable 'Status'. The overall process included data preprocessing, handling class imbalance through oversampling (SMOTE), and hyperparameter tuning to optimize the model's performance. As stated earlier, we addressed the class imbalance using oversampling technique SMOTE, making sure there are equal instances of both classes in the training set, with each class having 19,260 instances.

To identify the best parameters which would lead to the best model, we conducted a grid search over several parameters, including the regularization parameter (C), penalty type (l1 and l2),

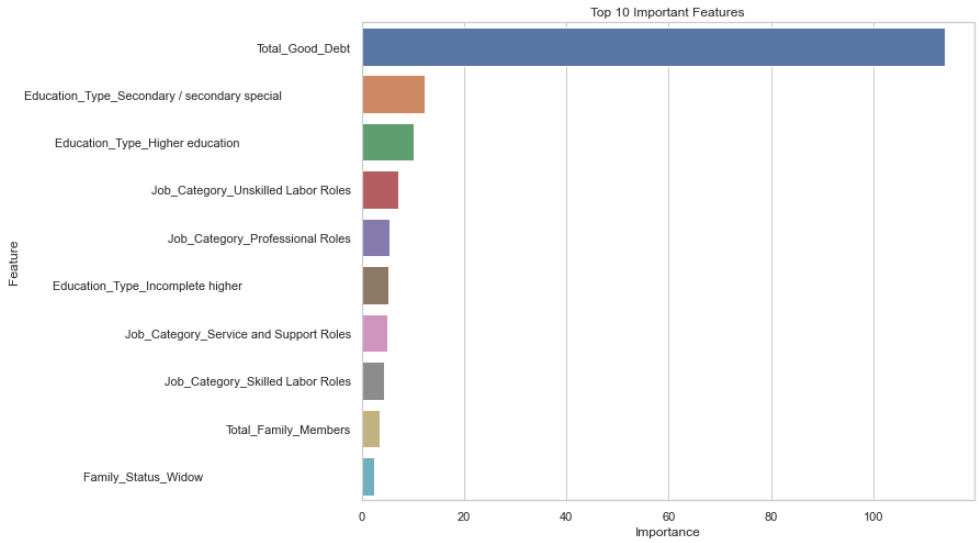
solver algorithm (liblinear), class weight (balanced and None), and maximum iterations (100, 200, and 500). The optimal hyperparameters identified by GridSearchCV were: **C=44668.35921509626, class_weight=balanced, max_iter=100, penalty=l2, and solver=liblinear**.

Results for Logistic Regression

We evaluated the model on a test set, which produced perfect scores across all metrics: an F1-score of 1.0, precision of 1.0, recall of 1.0, and accuracy of 1.0. The confusion matrix confirmed these results, indicating perfect classification with 4815 true positives, 11 true negatives, and no false positives or false negatives. The ROC curve demonstrated the model's ability to distinguish between classes, with an Area Under the Curve (AUC) of 1.0, indicating perfect performance.



Furthermore, the top 10 most important features identified by the logistic regression model included Total_Good_Debt, various education types, and job categories, with Total_Good_Debt being the most influential feature by a significant margin.

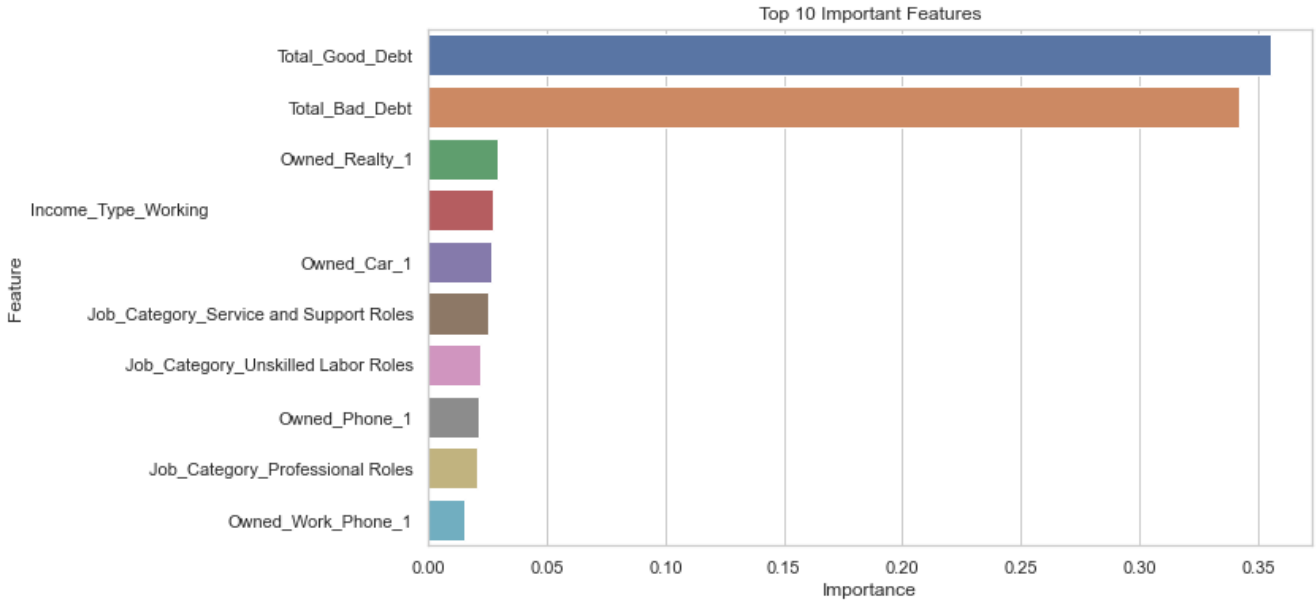


Random Forest

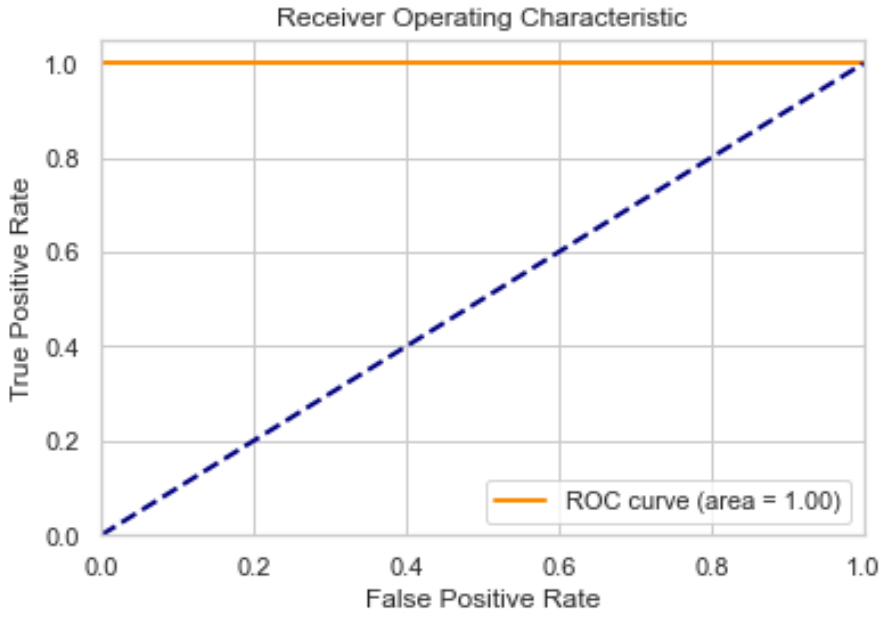
We utilized this analysis to develop an accurate Random Forest classification model to predict the target variable 'Status' effectively. We tuned the hyperparameters extensively to optimize the model's performance. We performed a grid search over a range of hyperparameters, including the number of estimators (n_estimators), maximum tree depth (max_depth), minimum samples required to split a node (min_samples_split), and the number of features to consider at each split (max_features). The optimal hyperparameters identified were max_depth=20, max_features='sqrt', min_samples_split=2, and n_estimators=200.

Results for Random Forest Model

We extracted the feature importances from the best estimator identified by GridSearchCV, revealing the top 10 most important features. The most significant features were Total_Good_Debt (0.355), Total_Bad_Debt (0.342), and Owned_Realty_1 (0.030), among others. We visualized these features in a bar plot to highlight their importance in the model's predictions.



The model's performance on the test set was perfect, achieving an F1-score of 1.0. The ROC curve demonstrated a perfect Area Under the Curve (AUC) of 1.0, indicating the model's outstanding ability to distinguish between classes. The precision-recall curve further supported the model's robustness, showing high precision and recall values. The classification report detailed the model's performance metrics, with precision, recall, and F1-scores all at 1.00 for both classes.



The confusion matrix confirmed these results, indicating perfect classification with 4815 true positives, 11 true negatives, and no false positives or false negatives.

Learnings

The data exploration and processing steps were crucial in identifying and handling data imbalance and feature importance. During these steps, we examined the dataset for missing values, outliers, and irrelevant features, ensuring that the data was clean and well-prepared for modeling. By converting categorical variables to numerical formats and applying techniques like one-hot encoding, we made the data suitable for machine learning algorithms. Addressing the data imbalance was especially important, as it allowed us to prevent the model from being biased towards the majority class, thereby improving the reliability of our predictions.

Both models, Logistic Regression and Random Forest, performed exceedingly well, achieving perfect scores across all metrics on the test set. However, the class imbalance brought challenges in interpreting these results, indicating that the models might be overfitting to the majority class. This suggests that while our models are technically proficient, the imbalanced nature of the dataset could have skewed the results. More balanced datasets could provide a clearer understanding of the models' true performance and ensure that they generalize better to new data.

Future Explorations

If we had more time, we would explore other methods for handling data imbalance, such as more advanced sampling methods, to further improve model performance. Additionally, we would experiment with other machine learning models, including gradient boosting machines and K-Means Clustering, to determine if they offer better predictive accuracy. Furthermore, we would conduct more in-depth feature engineering to capture complex relationships within the data. This would involve a more detailed analysis of feature importance and the incorporation of domain-specific knowledge to enhance the model’s ability to generalize and make accurate predictions.