Domain Oriented Case Study

Case Study Group:
Akshay Athawale
Vatsala Shukla
Dhyey Ratanpara

1.0 Project Background

In this assignment, you would face real world data of applications and bureau as shared by Home Credit, to practice the end-to-end process of model development in Credit Risk for Banks, Financial Institutions and NBFCs. You would build a bank's internal end-to-end scoring mechanism, based on the application information, clubbed with the raw bureau information.

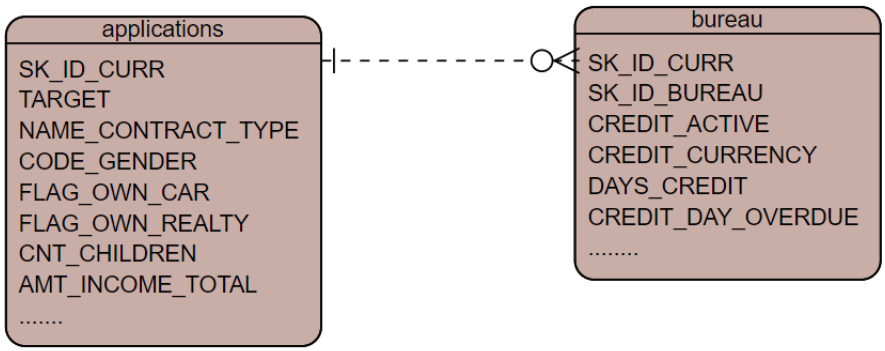
The primary objective of this study is to assist Home Credit in deciding which loan applications should be disbursed, and which should be rejected, based on the applicant's past behavior and application information.

2.0 Problem Statement and Solution Strategy

The dataset for this binary classification challenge: predicting loan repayment or default based on financial and behavioral data of applicants. Our approach involved five stages: Data Preparation, Exploratory Data Analysis, Feature Engineering, Classifier Model Training and Prediction, and Hyperparameter Tuning. We addressed missing data, encoded categorical features, and utilized feature engineering to link disparate data sources. The classifier models, including logistic regression and random forest, were trained using various techniques, with the aim of identifying the most effective model.

3.0 Datasets and Inputs

Datasets were sourced from Kaggle, comprising multiple CSV files with relational data structures. The main dataset included 307511 samples, with additional data from related sources like bureau records and previous applications. Our analysis focused on identifying correlations between these features and the target variable, loan default risk.



4.0 Benchmark Model and Evaluation Metrics

Our evaluation prioritized the ROC curve and AUC metrics, given their effectiveness in assessing binary classification models, especially with imbalanced datasets. A receiver operating characteristic (ROC) curve can summarize the performance of a binary classification model on the positive class where the x-axis shows False Positive Rate and the y-axis shows the True Positive Rate. A ROC curve does not have bias towards the majority or minority class since it uses the actual predicted probability, instead of the probability class, making it favorable when using imbalanced data with equal importance for the both classes5 . The area under the ROC curve can be calculated to find a score between 0 and 1 for a classifier for all threshold values, called the ROC area under curve or AUC. Since the AUC was also used as the evaluation metric in the Kaggle competition, it has been used as the primary evaluation metric, but other metrics like accuracy and F1-score are also calculated for comparing the different models. The accuracy6 is defined as number of correct predictions made, divided by the total number of predictions and does not work well for imbalanced datasets.

The F1 score6 is a metric which is calculated by taking the harmonic mean of the precision

$$\left(\frac{True\ Positive}{True\ Positive+False\ Positive}\right) \text{ and the recall } \left(\frac{True\ Positive}{True\ Positive+False\ Negative}\right).$$

However, since we had limited domain knowledge to create more features, and since we did not have access to the actual test dataset, my best results (test data AUC: 0.7860) did not exceed this higher-end benchmark. On the lower end, as a sanity check, we trained a base case model using logistic regression with only the main training dataset as a benchmark to improve on, which gave an AUC of 0.7454 on my test dataset

5.0 Project Design and Solution

The project, executed in Python3 within a Jupyter Notebook environment, employed various machine learning libraries and techniques. We focused on connecting relational databases through automated feature engineering and comparing classifier models on imbalanced data. Our analysis revealed significant insights, particularly regarding the impact of employment duration, income type, and other socio-economic factors on loan default risk.

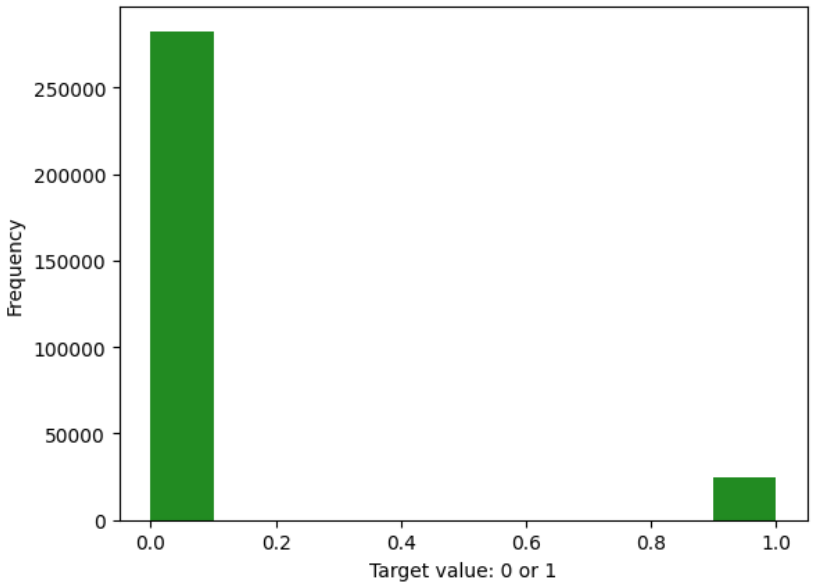
5.1 Data Preparation

As the first step, the necessary libraries and the datasets are imported. Since there are more than one files, all need to be imported before looking at the feature types and number of rows/columns in each file. For the main training data, there are 307511 total samples (each row a separate loan) with 122 features of types 41 integer, 65 float and 16 object datatypes. Out of these, the feature (SK_ID_CURR) serves as the index and TARGET is the response feature to be predicted. The dataset file names, number of rows and columns is summarized in Table 1.

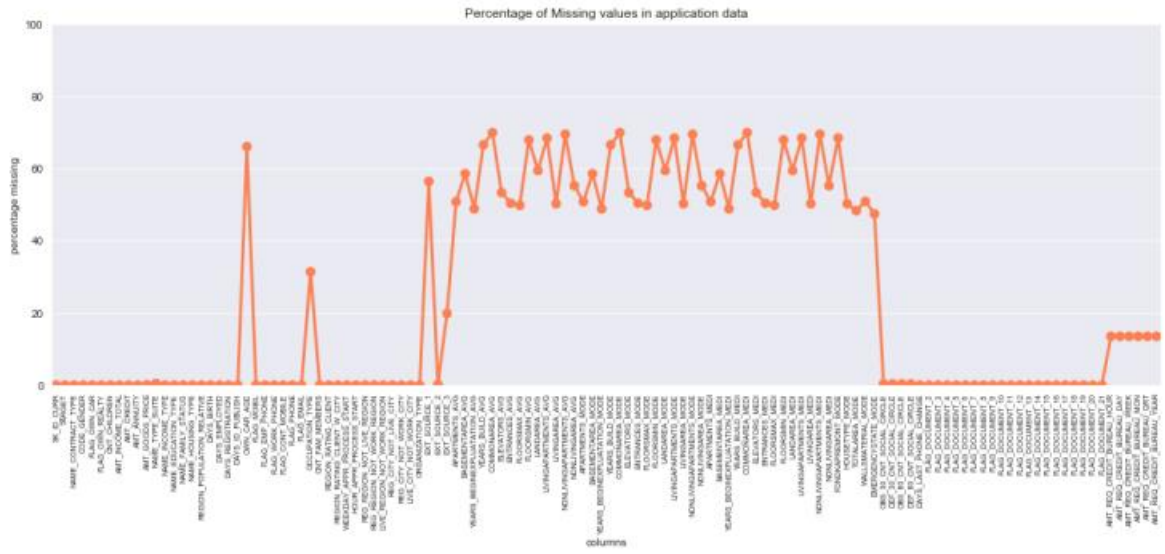
Dataset name	Rows	Columns
application_train	307511	122
bureau	1716428	17

5.2 Exploratory Data Analysis

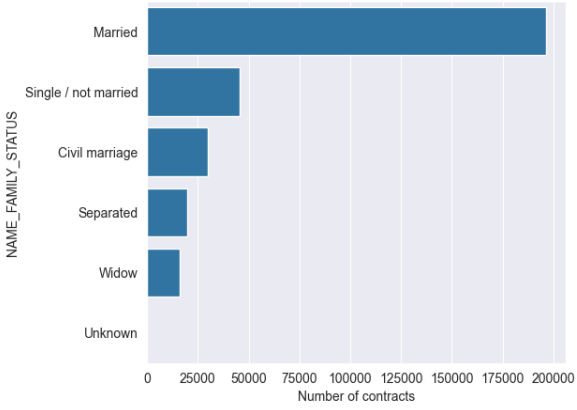
After data importing, we can investigate the data to check data quality and find trends. On plotting the distribution of the predictor variable, we can see in respective Figure that the dataset is imbalanced with the number of samples where loan is repaid (282,686) more than 10 times the number of samples where loan is defaulted (24,825). This imbalance should be considered during model training

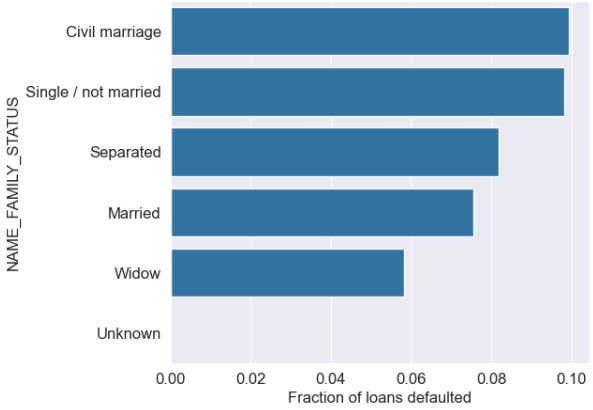


On checking for missing data, it is seen that a number of features in the main dataset (application) have missing values almost 50%. The features with high fractions of missing data need to be discarded, and those with some missing values need to be imputed before training any model. Similar checks are performed on the other datasets.

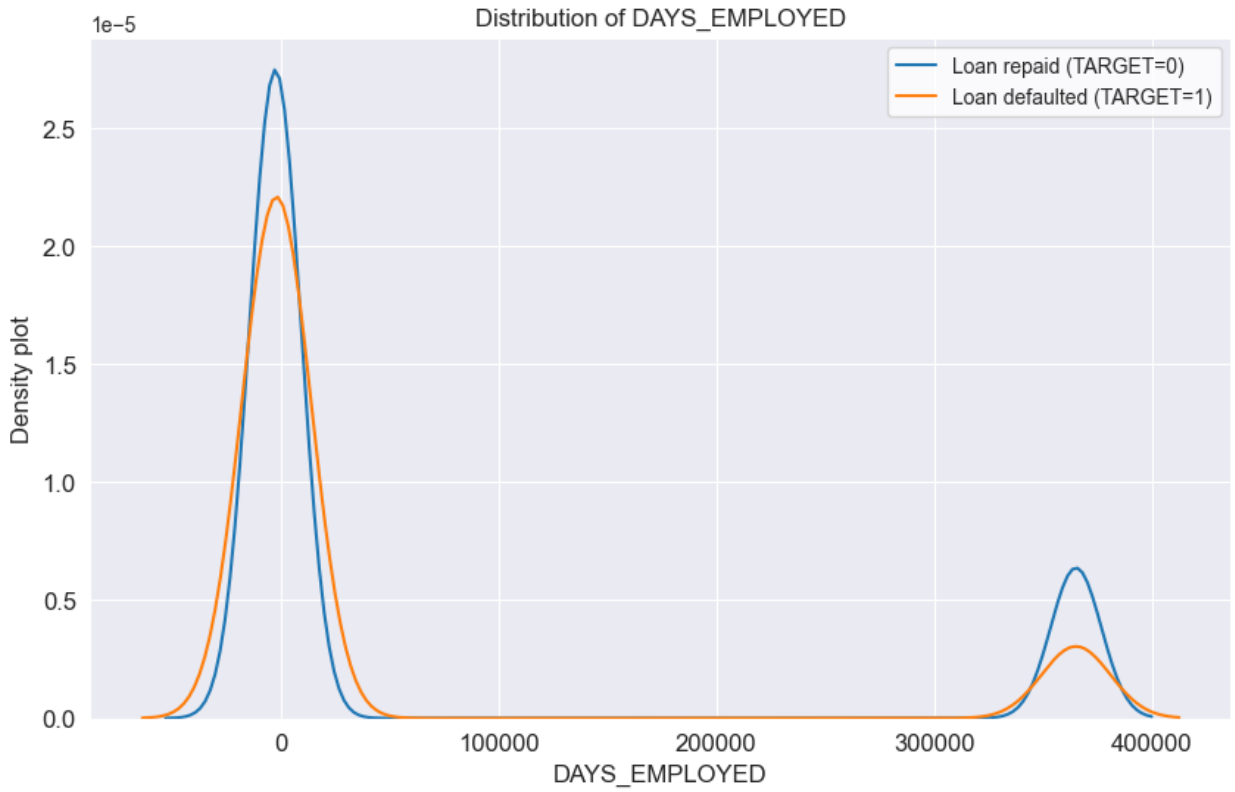


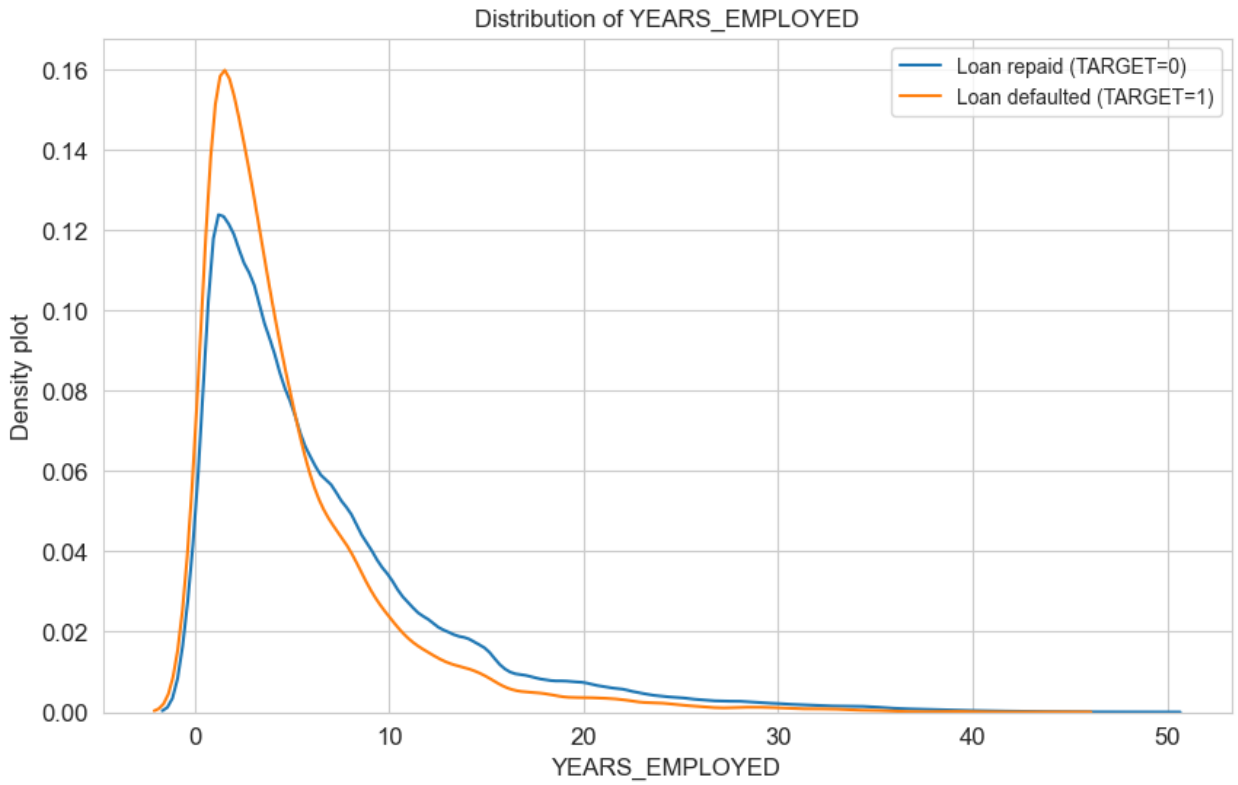
The next step entailed continuing the exploratory data analysis to see if any response features have significant differences for cases when loans are repaid as opposed to when loans are defaulted. A number of figures were created for categorical features with bar plots for each type, with each figure containing, a) the total number of categories for each response feature, b) the fraction of each category with loan defaulted. For example, for the feature income type (NAME_INCOME_TYPE) in respective Figure, we see that there are more loans taken by those who are working and they are more defaulted by those who are on maternity leave or unemployed. A comprehensive list of bar plots is available in the Jupyter notebook.





For the continuous features, the distribution of a feature for the cases when loans are repaid and defaulted, were plotted to examine differences and check data quality. For example, for the feature 'DAYS_EMPLOYED' in respective Figure, we can see that the distribution is hard to perceive and has some anomalous values of days employed more than 350000 days (958 years), which is impossible and needs to be corrected. On removing the outliers, and converting the days to years, we can see a more interpretable distribution in respective Figure, where there are a greater number of loans defaulted by people who are employed for fewer years.





A comprehensive list of such distribution plots is available in the Jupyter notebook. Such an analysis helps us gain an insight into the features which need to be corrected, and those which may help identify the likelihood of loan default and can turn out to be important features for model training.

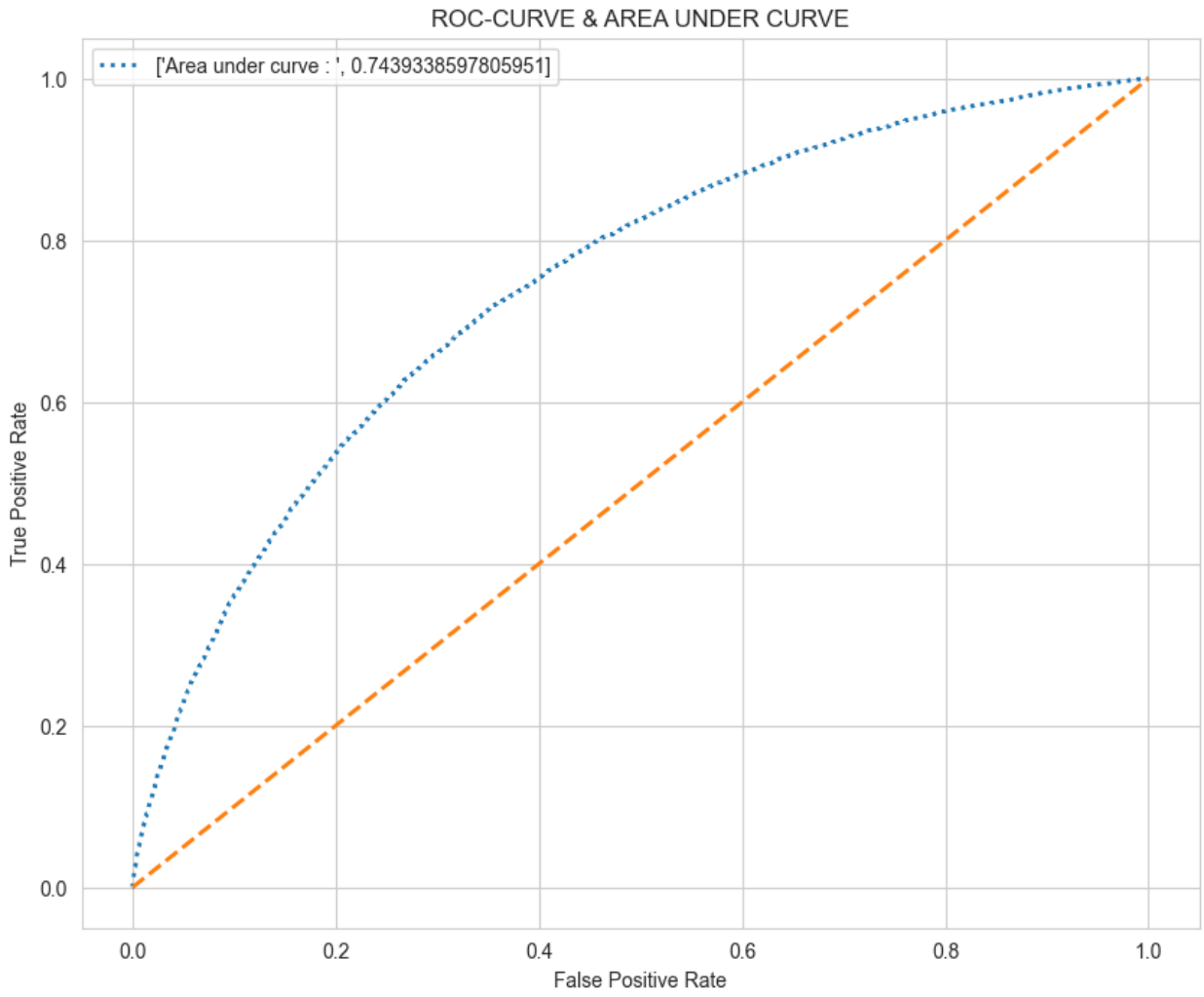
5.3 Feature Engineering

After exploring the data distributions, we can conduct feature engineering to prepare the data for model training. This includes operations like replacing outliers, imputing missing values, one-hot encoding categorical variables, and rescaling the data. The process of replacing outliers involves removing values from data which are greater than three standard deviations from the mean (eg. number of days in respective Figure). Since categorical variables cannot be directly interpreted by most classifiers, they need to be encoded as numbers. This can be done by label encoding or one-hot encoding. Label encoding assigns each category in a feature with an integer without creating new columns. One-hot encoding10 creates a new column for each category where each observation is assigned as 1, while the other category columns are assigned value of 0. It is preferred in this project since it is possible that the label numbers in the label encoding are wrongly interpreted by the model as holding some significance, while one-hot encoding does not have that issue. For tackling missing values, a two-step approach is followed. The features which have more than 60% data missing are removed, while the remaining features have data imputed, i.e., categorical features are filled with the most frequent column, and other features are filled with the median value. Rescaling of the features involves transforming each feature to a range of 0-1. I have also applied my limited domain knowledge to create few more variables,

specifically ratios accounting for the credit income %, annuity income %, credit term, and fraction of years employed. Since there are number of relational databases, we can use extract, transform, load (ETL) processes using automated feature engineering to connect the datasets. Since manual feature engineering cannot create all possible combinations of features from all the available datasets, automated feature engineering helps in building thousands of features using different operations like merge, group, sum, etc. Featuretools11 is an open-source library for making features from related datasets using deep feature synthesis. A table/dataframe called entity is created with a unique index for all the datasets. The relationships between the datasets are defined as shown in respective Figure using an index such as SK_ID_CURR or by creating a new index when needed. Care is taken to prevent cross-connecting relationships. This allows the multiple entities to come together to form an entity-set. Then, feature primitives are created for the datasets, where primitives are operations conducted on tables to create a feature. The two common types are aggregation and transformation. An aggregation groups together values from child dataset for each parent and then calculates a feature such as mean, min, max, or standard deviation. A transformation can be applied to one or more columns in a single table such as the difference between two columns or the absolute value of one column. Some common primitives in Featuretools are count, mean, median, trend, maximum, minimum, number of words, cumulative sum, difference etc. These primitives are used for deep feature synthesis (DFS), which is the process Featuretools uses to make new features. In my project, the choice of primitives resulted in a total of 3534 total features. However, before moving forward towards training, the features created from Featuretools needed to be cleaned up, since many features had high missing values or had a high degree of correlation with each other. The low information features (only one unique value) and those that had more than 60% missing values were removed. Then, from the predictor features which had a high degree of correlation with each other, one feature from each feature pair over a set threshold (0.8) were removed. These steps reduced the number of features, which helps reduce the impact of the curse of dimensionality12 . Thereafter, operations like imputing missing values, one-hot encoding of categorical variables, and rescaling were performed on the remaining features to prepare the dataset for model training.

5.4 Classifier Models:

Training, Prediction and Comparison Before building the full-fledged classification models, I trained a logistic regression model13 using only the main application dataset, to serve as a low-end benchmark and work as a sanity check. This base case - logistic regression model, had a high accuracy score of 0.9184. It worked well when predicting the cases when loan is repaid (F1-score = 0.96) but did not perform well to predict when loans are defaulted (F1-score = 0.02), since this is an imbalanced dataset. Hence, I preferred to use AUC ROC which works with data imbalance, since it works on prediction values rather than classes. As shown in respective Figure, the AUC ROC gave a value of 0.7454 for the base case on the test data



The final created dataset is first split into training and testing parts in the ratio 75:25. To reduce the data imbalance where the ratio of the majority class (TARGET=0) to minority class (TARGET=1) is > 10, a technique called random undersampling5 was implemented on the training dataset. This reduces the skew in data by selecting fewer number of majority class samples, which helps balance the dataset. To prevent losing a large volume of data due to discarding samples from the majority class, a limit was set to consider majority class samples equal to twice the minority class samples. While this does not completely resolve the data imbalance, it makes it more tolerable so that the model is better trained to predict when loans may be defaulted (TARGET=1). For finding the best classifier models, I trained numerous different models on the undersampled training dataset.

5.4.1 Logistic Regression

Logistic regression is a supervised classification algorithm that uses the sigmoid function (σ) (equation 1B) to convert the linear regression equation (t) (equation 1A) formed by the predictor features (X) into binary classes, 0 or 1 using decision thresholds.

$$t = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

$$\sigma(t) = \frac{1}{1+e^{-t}}$$

The model from the Scikit-learn14 package was implemented, and when applied with the default hyperparameters (entire dataset), it gave an accuracy of 0.9184, F1-score of 0.0 and AUC of 0.5079.

5.4.2 Random Forest

Random forest15 is an ensemble tree-based learning algorithm that uses multiple decision trees from randomly selected subsets of training data for classification. It uses averaging from the ensembles to build class predictions, while improving the predictive accuracy and control over-fitting. The model from the Scikit-learn package was implemented, and when applied with the default hyperparameters, it gave an accuracy of 0.8451, F1-score of 0.2208 and AUC of 0.6670.

5.4.3 Decision Tree

Decision tree16 is an algorithm where the data is iteratively split into partitions using all the features of the dataset for classification. While the accuracy increases with more splits, it can lead to overfitting when used without cross-validation. The model from the Scikit-learn package was implemented, and when applied with the default hyperparameters, it gave an accuracy of 0.6879, F1-score of 0.1912 and AUC of 0.5806.

5.4.4 Gaussian Naïve Bayes

Gaussian Naïve Bayes17 is a supervised classification algorithm based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features and the likelihood is assumed to be Gaussian. The model from the Scikit-learn package was implemented, and when applied with the default hyperparameters, it gave an accuracy of 0.9184, F1-score of 0.0 and AUC of 0.6691.

5.4.5 XGBoost

XGBoost8 is an optimized distributed gradient boosting library, which uses Gradient Boosting algorithm for efficient classification. XGBoost provides parallel tree boosting to build multiple weak prediction decision tree models to perform fast and accurate predicition. The model from the XGBoost package was implemented, and when applied with the default hyperparameters, it gave an accuracy of 0.8292, F1-score of 0.3103 and AUC of 0.7604.

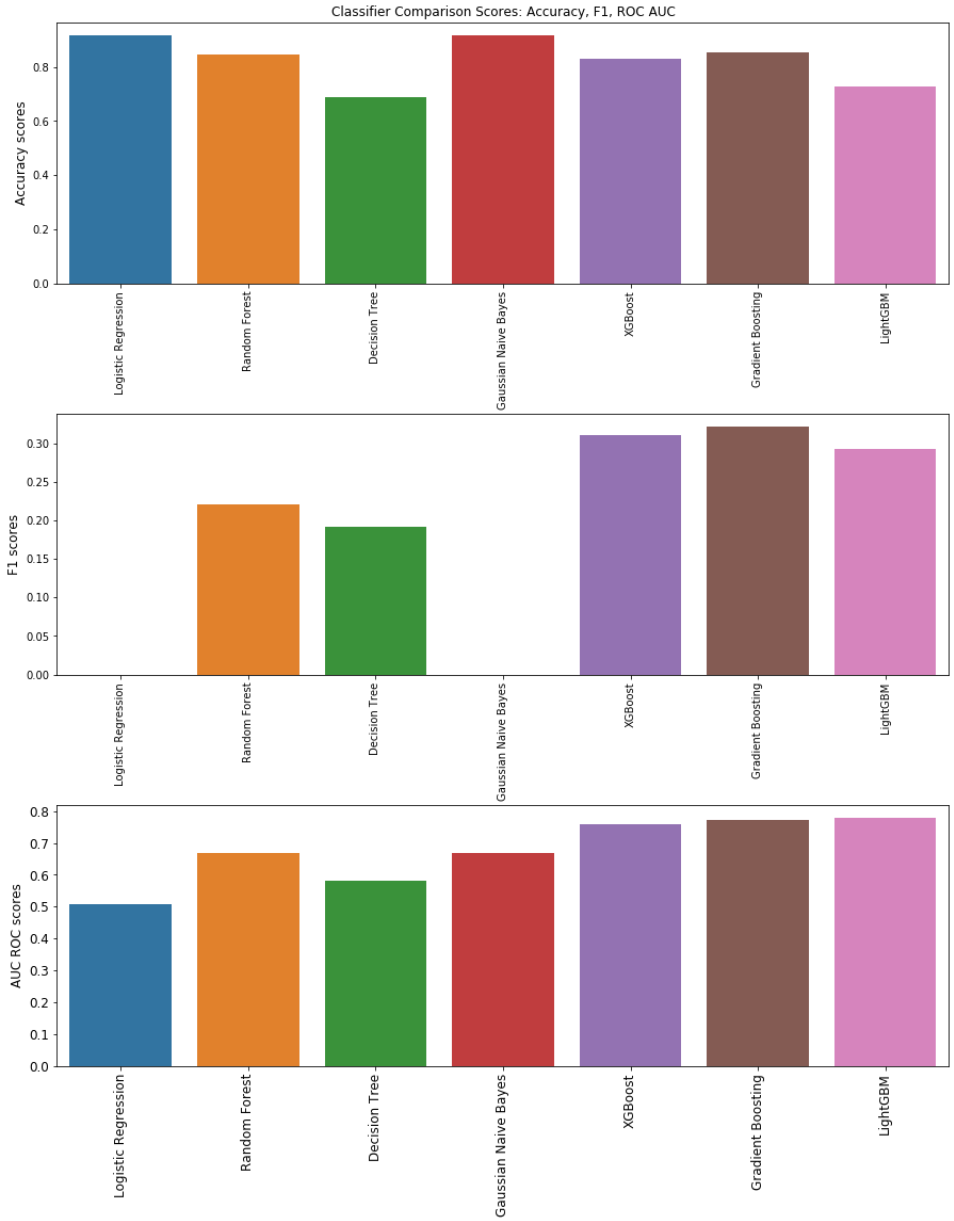
5.4.6 Gradient Boosting

Gradient Tree Boosting18 is a generalization of boosting to arbitrary differentiable loss functions. It uses multiple weak learners (regression trees) additively in a forward stage-wise fashion and generalizes them by allowing optimization of a differentiable loss function. The model from the Scikitlearn package was implemented, and when applied with the default hyperparameters, it gave an accuracy of 0.8548, F1-score of 0.3221 and AUC of 0.7735.

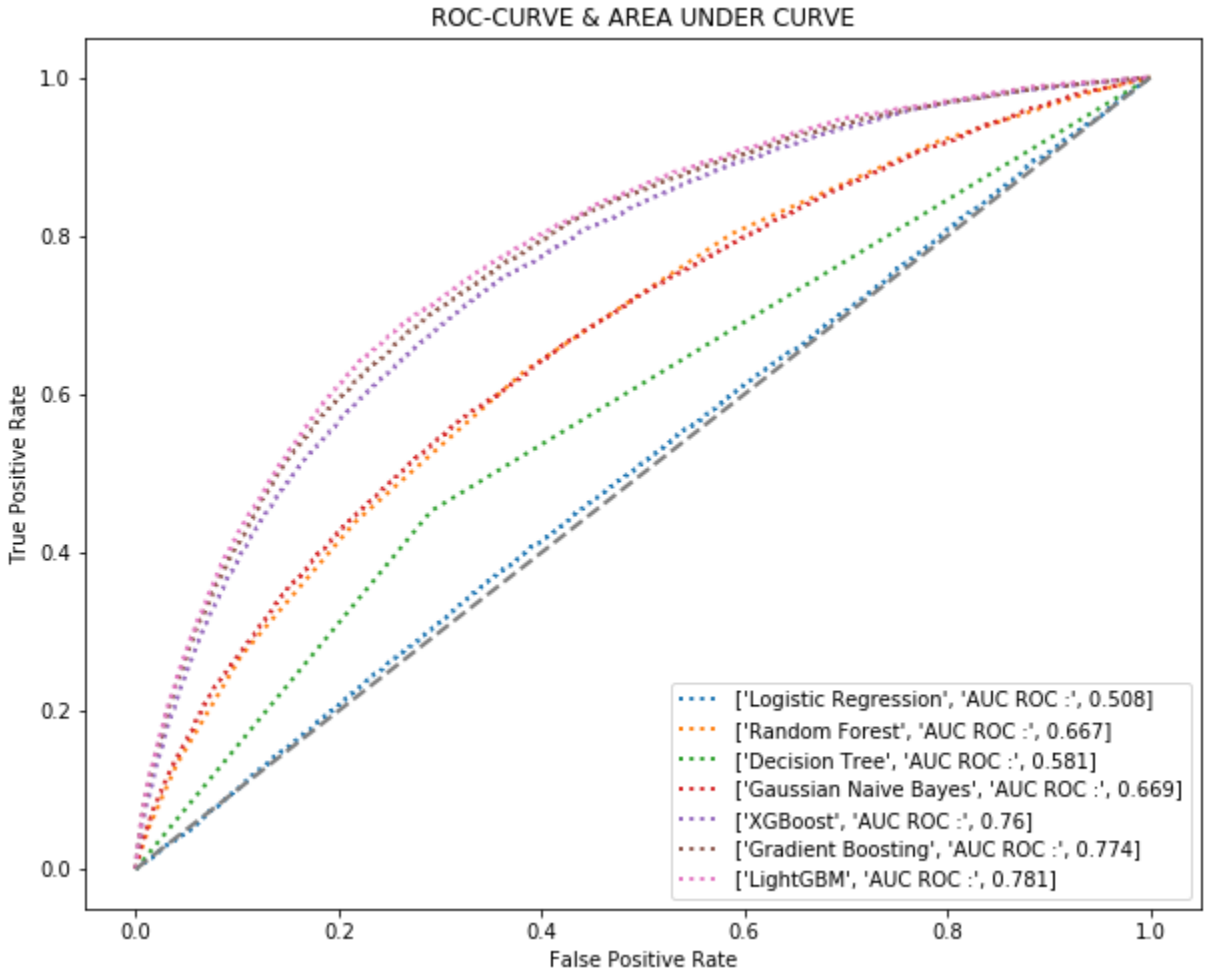
5.4.7 LightGBM

LightGBM9 is a gradient boosting framework that uses tree-based learning algorithm and can be used for supervised classification. It uses histogram-based algorithms, which bucket continuous feature (attribute) values into discrete bins, which decreases training time and memory usage. The model from the LightGBM package was implemented, and when applied with the default hyperparameters and validation split (early stopping), it gave an accuracy of 0.7264, F1-score of 0.2929 and AUC of 0.7806.

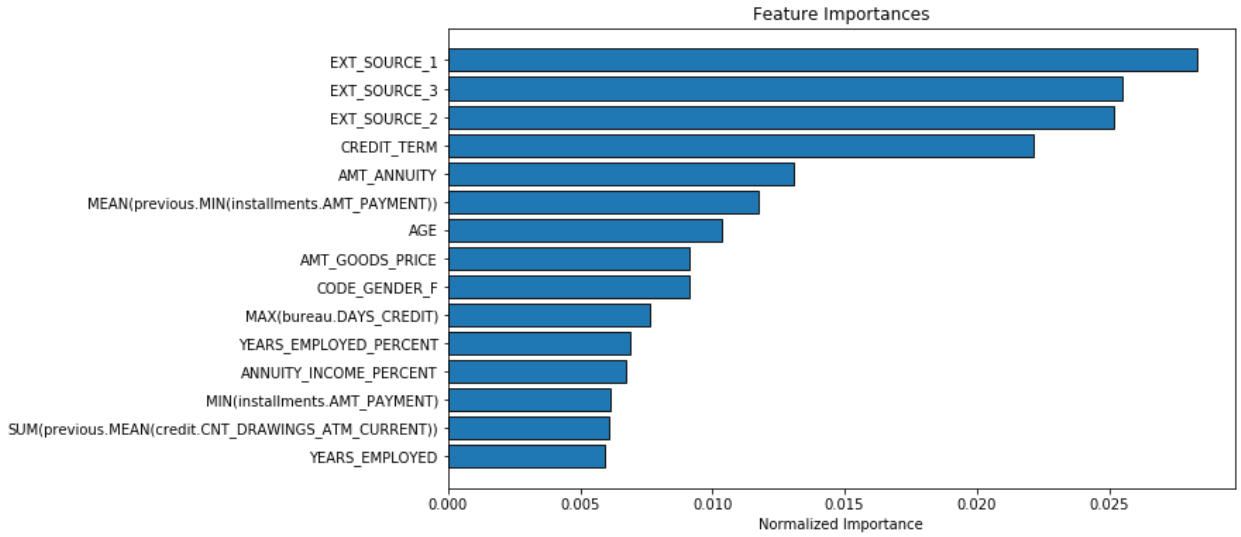
The performance of the different classifiers on the test data can be better compared using metrics like accuracy, F1-score, and ROC AUC, We can see from the comparisons of accuracy, F1 score and AUC ROC scores that all models have different rankings.

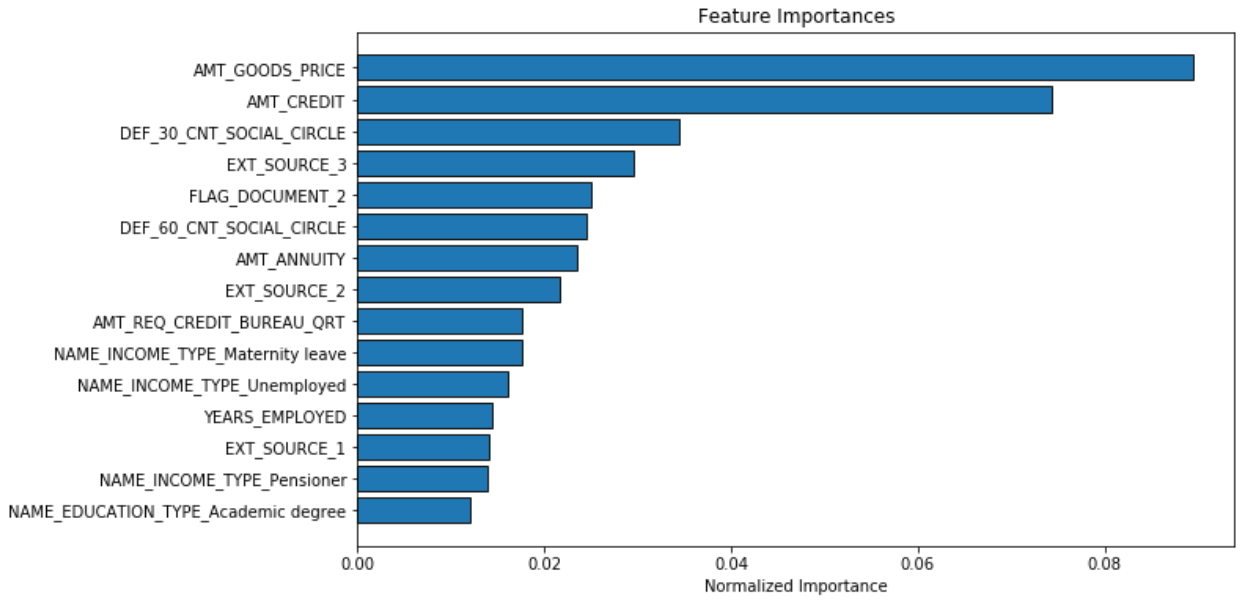


The Logistic Regression and Gaussian Naive Bayes models have highest accuracy, but the lowest F1 scores. This shows that they do not work well for imbalanced data. The XGBoost, Gradient Boosting and LightGBM classifiers give good F1 and AUC results. The AUC scores can be better visualized using the ROC curves respective Figure The AUC scores improve as the curves rise higher above the diagonal. Since, the LightGBM classifier has the best score for AUC (0.781), I selected it moving forward.



After choosing the best classifier, We used K-fold cross validation to select the best model. K-fold cross validation partitions the samples into K subsamples, where in each case one subsample is used for validation while other partitions are used for training. This allowed us to choose parameters corresponding to the best performance (AUC score: 0.7834) without creating a separate validation dataset.





From the top 15 features of the base case model , I observed that it contains a number of features that had a high loan default rate during exploratory data analysis. From the top 15 features of the LightGBM model, it is seen that it contains a number of features that were created from domain knowledge and through the automated feature engineering (DFS) process. Among the original features, many are those that had a high loan default rate during exploratory data analysis.

5.5 Hyperparameter Tuning

After choosing the binary classifier, we can tune the hyperparameters for improving the model results through grid search, random search, and Bayesian optimization (Hypertopt library)20. The hyperparameter tuning process will use an objective function on the given domain space, and an optimization algorithm to give the results. The domain space is the range of hyperparameters over which the tuning process will be carried out. The domain space is the range of hyperparameters over which the tuning process will be carried out.

5.5.1 Grid search

The grid search algorithm iterates over each hyperparameter incrementally, one at a time to explore the entire domain space. However, the exhaustive search can be very time consuming and cannot be completed with finite computing resources. Hence, the maximum evaluations were limited to 20 due to time and computational constraints. The best model from grid search gave an AUC score of 0.7840 on the test data set.

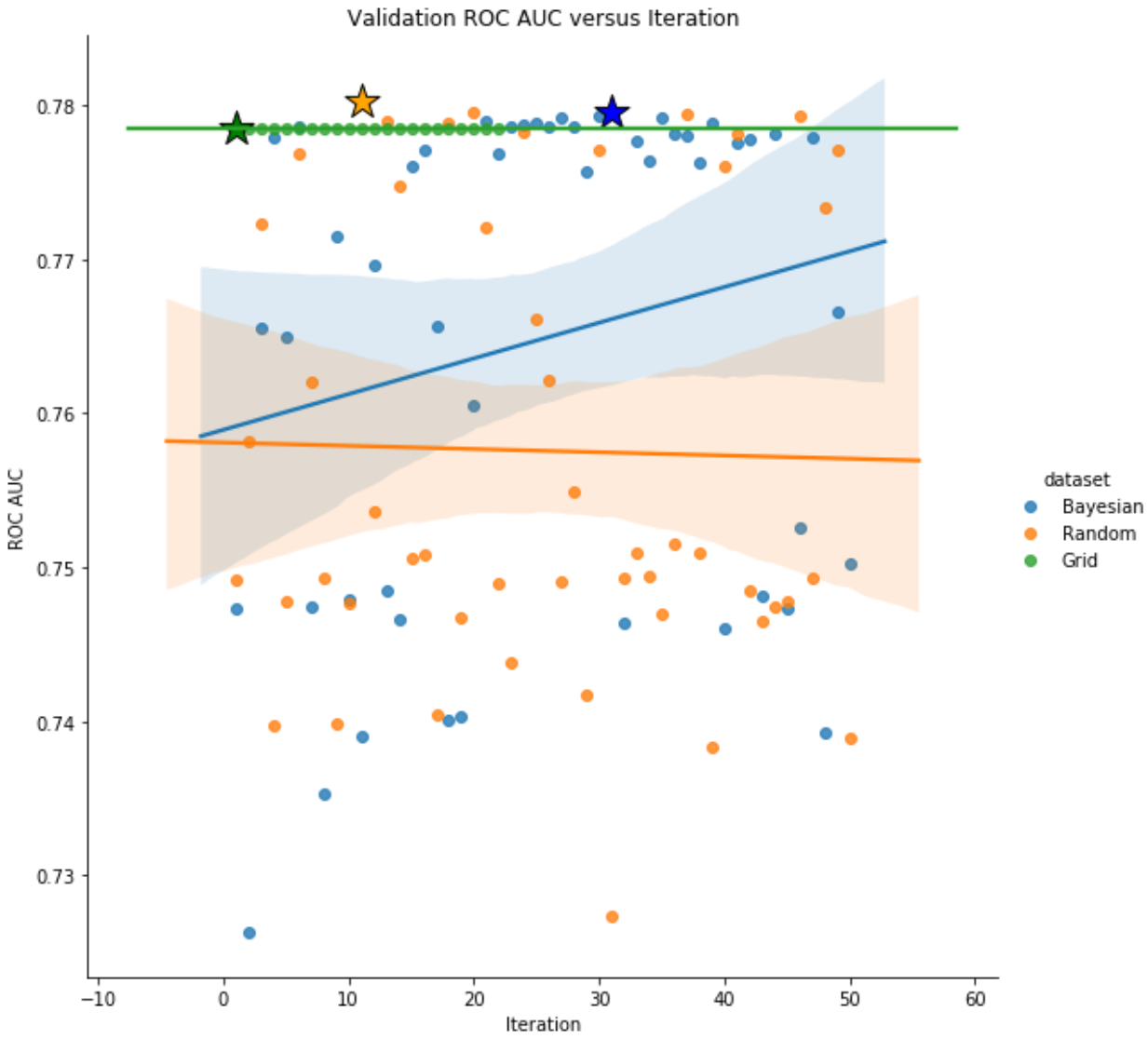
5.5.2 Random search

The random search algorithm selects hyperparameters from the domain space in a random manner. With limited resources, it can act as a preferred strategy due to a less exhaustive search and high possibility of finding good hyperparameter combinations. The maximum evaluations were limited to 50 due to time and computational constraints. The best model from random search gave an AUC score of 0.7860 on the test data set.

5.5.3 Bayesian optimization

The Bayesian method, unlike the previous uninformed methods uses the results of the previous iteration to decide the next hyperparameter combination in the domain space. It uses the Hyperopt library using the Tree Parzen Estimator algorithm20 to select the next hyperparameter values for evaluation in the objective function. The maximum evaluations were limited to 50 due to time and computational constraints. The best model from Bayesian search gave an AUC score of 0.7836 on the test data set. The ROC AUC validation scores from all three methods for different iterations can be compared to see the trends respective Figure. The grid search scores (green) do not differ much since during each iteration, only one hyperparameter is incrementally changed with highest validation AUC of 0.7785.

The random search scores (orange) are scattered with no noticeable trend but have the highest validation ROC AUC score (0.7802) simply by finding a favorable combination. The Bayesian search (blue) can be observed to have progressively better scores with increasing iterations as expected with highest validation AUC of 0.7795, and if the number of iterations were increased could have surpassed the score from random method.



The hyperparameters for the LightGBM model with best validation AUC (random search) are as shown in Table 2. Finally, the predictions on test dataset from the best model (true/predicted labels and index) were saved

Hyperparameter	Value
boosting_type	gbdt
colsample_bytree	0.6
is_unbalance	False
learning_rate	0.0068724
min_child_samples	445
n_estimators	2897
num_leaves	24
reg_alpha	0.8163265
reg_lambda	0.7346939
subsample	0.6969697
subsample_for_bin	160000

6.0 Conclusion(On Business Impact)

- Based on the analysis of various features related to loan defaults, several key indicators and actionable recommendations can be identified for a finance entity:

- CREDIT_TYPE Impact: The prevalence of defaults among those with consumer credit/credit card suggests a need for stricter credit checks or tailored financial products for these customers. For equipment purchase or microloan users, who show higher defaults, consider implementing specialized risk assessment strategies.
- CREDIT_CURRENCY Insight: The higher default rate with currency 3 loans indicates potential economic or stability issues with this currency. Enhancing currency risk assessment and adjusting interest rates or credit terms for loans in this currency could mitigate risks.
- CREDIT_ACTIVE Analysis: A significant number of defaults in bad debt or sold credit categories highlights the importance of rigorous monitoring of credit activity and proactive debt recovery strategies.
- Income Band Observations: The slight increase in repaid loans for higher income bands suggests tailoring loan products and marketing strategies towards higher-income customers, possibly offering them more favorable terms.
- Years of ID Publishing and Registration: More defaults with lower years of ID publishing and registration indicate that individuals with a shorter financial history may pose a higher risk. Enhancing the vetting process for newer customers could be beneficial.

- Age-Related Risk: Younger individuals defaulting more suggests the need for different lending models for younger demographics, potentially incorporating educational initiatives about financial management.
- Geographical Factors: Higher defaults by those not living or working in the same city/region as the loan origination point to the need for location-based risk assessment. Tailoring loan products to account for regional economic variations could be effective.
- Housing, Education, and Occupation: More defaults among those with rented housing, lower education, or in labor categories point to socio-economic factors influencing loan repayment. Consider developing financial products or assistance programs tailored to these demographics.
- Loan Type and Contract: Higher defaults on cash loans compared to revolving loans suggest revisiting the terms and conditions of these loan types, possibly incorporating more flexible repayment options.
- Demographic Insights: The higher default rates among males, those without cars, and those not owning realty indicate demographic segments that might require more thorough credit assessments or targeted financial education programs.
- Family Status: The higher default rate among those who are in a civil marriage or single suggests that marital status should be a consideration in the risk assessment process.

6.1 Recommendations:

- Implement Enhanced Credit Scoring Models: Incorporate these insights into credit scoring models to better assess individual risk.
- Develop Tailored Financial Products: Offer products that cater to the specific needs and risk profiles of different customer segments.
- Increase Financial Literacy Initiatives: Especially for younger and lower-income customers to improve their understanding of financial obligations.
- Strengthen Debt Recovery Strategies: Particularly for high-risk segments to mitigate potential losses.
- Utilize Data Analytics: Continuously analyze loan performance data to identify emerging trends and adjust strategies accordingly.