CS-5820-100 – Artificial Intelligence

Credit Risk Evaluation using Machine Learning

By
Surya Vaddhiparthy

Table of Contents

GitHub Repository of the Project: https://github.com/SriSuryaSV/creditrisk

Table of Figures

I. Abstract

The project aims to enable access to credit for individuals with limited or no credit histories using alternative data sources to assess their creditworthiness. The data like annual income, goods price (product for which loan is applied), the population of the client's region, age of phone number, etc. were used to assess the creditworthiness. The data obtained is checked for the presence of inconsistent data that can interfere with the performance of the machine learning model using data visualization and exploratory data analysis techniques. Relevant features were identified by computing the correlation of all the features with the target variable which is a binary variable that indicates whether the individual has defaulted on credit repayments or not. Basic and advanced data wrangling techniques were used to fix the identified problems at various stages. Machine learning model evaluation techniques like cross-validation score, accuracy, confusion matrix, precision, and recall were used to evaluate the model's ability in predicting the credit default variable. Basic data visualization techniques revealed that there are more female applicants, the applicant was unaccompanied at the time of loan application and a major proportion of the applicants were married. Further analysis of the data revealed that there is a class imbalance in the target variable i.e., the target variable had an unequal distribution of 0s and 1s which interfered with the Machine Learning model's performance, and this was subsequently addressed by fixing the class imbalance. The model was recomputed to have a high degree of accuracy in predicting the target variable using extreme gradient boosting (XGBoost) Machine Learning algorithm.

II. Introduction

The primary objective of this project is to predict an applicant's ability to repay the loan using machine learning. The focus is primarily on the section of applicants without any credit history, where alternative sources of data was used to evaluate the applicant's creditworthiness.

1. Data

The data set contains information on applicants' and below are the various categories of information:

1.1 The family attributes: Number of family members, children, marriage status, and if the family members accompanied the applicant during the time of loan application.
1.2 Geospatial information: Location of the loan application. Attributes of the residential complex in which the applicant lives like the age of the building, area of the building, number of entrances, number of floors, rating of the region, and population of the region.
1.3 Personal attributes: Number of days the applicant has been employed, age of the applicant at the time of application, income, car information, and education.
1.4 Document flag: Certain attributes indicate whether a document was provided or not. Also, to indicate whether a certain piece of information like phone number, e-mail, or work phone number was provided.
1.5 Application information: The information is provided when the application form is filled.
1.6 Normalized variables: These are certain numbers extracted from external databases for each applicant that are normalized.
1.7 Target variable: This is a very important variable that indicates if a client would default on their payments, and this is the variable that is to be computed by the ML model.

2. Problem Statement

The primary goal of this project is to compute a machine learning model that is accurate enough to identify a risky applicant who is defined as a person having issues with making payments on time as per the data set.

This problem is primarily approached by analyzing the input data set and performing data cleansing on the data, identifying trends in the data, selecting relevant features, and computing the machine learning model. Where certain challenges have been encountered, the steps taken to address these challenges successfully, and the metrics used for measuring the machine learning model performance are described.

III. Methods

The methods used can be summarized broadly under the below categories.

1. Exploratory data analysis.
2. Data wrangling.
3. Data visualization.
4. Feature engineering for machine learning.
5. Machine learning

1. Exploratory data analysis

This step is intended to understand the Dataset in terms of its dimensions and content. The following are the Python methods used to perform the exploratory data analysis.

1.1 Basic Exploratory data analysis

a) *pd.read_csv:* To load the data set in CSV format into Python.
b) *df.head():* To view the first few rows of the datasets.
c) *df.rename(columns=str.lower):* To change the column names to lowercase.
d) *pd.set_option:* This prevents Python from truncating the output, to view all the rows in the output.
e) *df.dtypes:* To check the data type of each row.

1.2 Statistical exploratory data analysis

This step is used to compute various summary statistics of all the features in the data set.

a) *df.target.unique():* Prints unique values present in the selected variable.
b) *df.describe().T:* It is used to compute the summary statistics like the count, mean, standard deviation, minimum, percentiles, and maximum of each variable.

2. Data Wrangling

2.1 Missing values and their imputation

The accuracy of the machine learning model is heavily dependent on the availability of the data and also the quality of data, which implies that it should not have any missing values.

a) *df_missing=df.isna().sum():* To retrieve the number of missing values.
b) *df_missing.sort_values(ascending=False):* Output the variable with the highest number of missing values first.
c) *df[df_missing[df_missing<100000].index]:* Select those variables that have more than 100,000 missing values.
d) *df[df_missing2[df_missing2==True].index]:* Select all the variables with missing values, Which were visualized in the next step.
e) *df['housetype_mode'].value_counts():* Get the number of rows in each category.
f) *df[['occupation_type']].fillna(value='Unknown'):* the categorical variable occupation type, missing values are imputed with one of the frequently occurring category “Unknown”.
g) *df.isnull().any():* This function is used to verify the presence of any missing values after all the necessary imputations have been performed to eliminate the missing values. The output of this function would be “True” against the feature name if there is any missing value.

2.2 Outliers

Machine learning models can get affected by the presence of outliers and hence it is important to identify the outliers and take the necessary steps.

A custom function has been defined to identify the 25th and the 75th quantiles of the values and which were further used to compute the interquartile range to define an acceptable range of values and any value that doesn't fall in between these values is considered an outlier.

*outliers = df[((df<(q1-1.5*IQR)) | (df>(q3+1.5*IQR)))]*

3. Data Visualization

This step was used to identify various trends in the data, the frequency of the categorical variables and, to identify the outliers' using boxplots and scatterplots.

a) *df.plot(kind='bar'):* Generates a bar plot.
b) *df.plot(kind='pie',legend=True,figsize=(12,12)):* Generates a Pie plot.
c) *plt.boxplot:* Generates a boxplot.
d) *df.plot(kind='scatter',x==' ',y='):* Generates a scatter plot.

4. Feature Engineering

In this step, relevant features are identified to compute the machine learning algorithms using correlation values between all the features and the target variable (credit default indicator). The features that have high degree of correlation with the target variable are filtered in to be processed further.

a) *df.corr():* used to compute correlation between all variables in the data frame.
b) *sns.heatmap():* used to generate a heat map of the correlation values.
c) *df_01.drop('target', axis=1):* Removes a column from the data frame. Here, in the context of this project, this function was used to drop the target variable from the data to separate the Target variable and the predictors.

5. Machine learning model generation and evaluation

5.1 Machine learning model used

a) *Logistic regression (baseline):* This is a case of binary logistic regression as the target variable can be either zero or one, log odds to probability is called the logistic function. This model is chosen as the baseline model.
b) *Stochastic gradient descent:* It is a stochastic approximation of gradient descent optimization. The actual gradient is replaced by an estimate from a randomly selected subset of data useful in large datasets to reduce the computational burden.
c) *Extreme gradient boosting (XGBoost):* This is a scalable and accurate implementation of gradient boosting techniques that involve boosting/improving a weaker model by combining it with many other weak models iteratively. Multiple shallow decision trees are trained, and, in each iteration, the error residuals of previous models are used to improvise the current model.

d) *Random forest classifier:* It is an ensemble learning method for classification or regression by fitting several decision tree classifiers, the output of the random forest is the value selected by most trees.

5.2 Metrics used to evaluate the machine learning models

a) *Confusion matrix:* This is a matrix of the count of correct and wrong predictions for each class. Using this the values in this matrix, accuracy, precision, and recall are computed.

Confusion Matrix

	Predicted YES	Predicted NO
Actual Yes	True Positive (TP)	False Negative (FN)
Actual No	False Positive (FP)	True Negative (TN)

Figure 1: Confusion Matrix

b) *Accuracy:* This is the ratio of correct decisions out of total decisions.

c) *Precision:* Correct positive predictions out of all the positive predictions. It is the accuracy of the minority class predictions. In this case, it is the indicator of the client being a defaulter.

d) *Recall:* Correct positive predictions made from all possible positive productions. It considers the missed positive predictions by the model.

5.3 Python methods used in machine learning model generation process

a) xtrain, xtest, ytrain, ytest=train_test_split(X, y, test_size=0.15): used to split the given data set into training and testing data set with the selected test size (15% in this case).

b) Model.fit: Computes the machine learning model.

c) cross_val_score(): Generates K fold cross-validation score of the model.

d) Model.predict(): Generates the model predictions based on the test data set.

e) confusion_matrix(): Generates a confusion matrix to summarize the performance of the classification algorithm. The higher the values on the principal diagonal of the matrix, the better the model.

f) classification_report: Generates machine learning model evaluation metrics like precision, recall, F1 score, and accuracy.

5.4 Sampling methods used to fix the class imbalance

The SMOTE (Synthetic Minority Oversampling Technique) was used to resolve the imbalanced classification issue to improve the model accuracy. This technique is used to synthesize new samples from the minority class to fix the class imbalance.

```
sample_technique=SMOTE()
X, y = sample_technique.fit_resample(X_df, y_df)
```

IV. Results

1. Exploratory data analysis results

Exploratory data analysis revealed that the data has 307,511 rows and 122 columns. The attribute names were converted into lowercase. The data types are int64, float64, and object.

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDRI
0	100002	1	Cash loans	M	N	Y	
1	100003	0	Cash loans	F	N	N	
2	100004	0	Revolving loans	M	Y	Y	
3	100006	0	Cash loans	F	N	Y	
4	100007	0	Cash loans	M	N	Y	

5 rows × 122 columns

Figure 2: First five rows of the data set

```
df.target.unique()
```

```
array([1, 0])
```

Figure 3: Distinct values in the target variable

```
df['occupation_type'].value_counts()
```

```
Laborers                 55186
Sales staff              32102
Core staff               27570
Managers                 21371
Drivers                  18603
High skill tech staff    11380
Accountants               9813
Medicine staff            8537
Security staff            6721
Cooking staff             5946
Cleaning staff            4653
Private service staff     2652
Low-skill Laborers        2093
Waiters/barmen staff      1348
Secretaries               1305
Realty agents              751
HR staff                   563
IT staff                   526
```

Figure 4: Number of applicants in each occupation category

2. Data wrangling results

An analysis of missing values revealed that many variables have missing values in more than 100,000 data points, these attributes with missing values, if used in the machine learning model computation, would severely affect the accuracy of the model, and hence were eliminated. Bibles having missing values of fewer than 100,000 were further isolated and their first few rows were visualized using the head function.

	amt_annuity	amt_goods_price	name_type_suite	occupation_type	cnt_fam_members	ext_source_2	ext_source_3
0	24700.5	351000.0	Unaccompanied	Laborers	1.0	0.262949	0.139376
1	35698.5	1129500.0	Family	Core staff	2.0	0.622246	NaN
2	6750.0	135000.0	Unaccompanied	Laborers	1.0	0.555912	0.729567
3	29686.5	297000.0	Unaccompanied	Laborers	2.0	0.650442	NaN
4	21865.5	513000.0	Unaccompanied	Core staff	1.0	0.322738	NaN
5	27517.5	454500.0	Spouse, partner	Laborers	2.0	0.354225	0.621226
6	41301.0	1395000.0	Unaccompanied	Accountants	3.0	0.724000	0.492060
7	42075.0	1530000.0	Unaccompanied	Managers	2.0	0.714279	0.540654
8	33826.5	913500.0	Children	NaN	2.0	0.205747	0.751724
9	20250.0	405000.0	Unaccompanied	Laborers	1.0	0.746644	NaN

Figure 5: First 10 rows of features with missing values

```
Unknown                 96391
Laborers                55186
Sales staff             32102
Core staff              27570
Managers                21371
Drivers                 18603
High skill tech staff   11380
Accountants              9813
Medicine staff           8537
Security staff           6721
Cooking staff            5946
Cleaning staff           4653
Private service staff    2652
Low-skill Laborers       2093
Waiters/barmen staff     1348
Secretaries              1305
Realty agents             751
HR staff                  563
IT staff                  526
```

Figure 6: Variable "occupation type" imputed with frequent value "Unknown"

region_rating_client	80527
region_rating_client_w_city	78027
days_employed	72217
reg_city_not_work_city	70867
flag_work_phone	61308
flag_emp_phone	55386
live_city_not_work_city	55215
amt_req_credit_bureau_qrt	50575
amt_req_credit_bureau_mon	43759
def_30_cnt_social_circle	35166
flag_document_6	27078
def_60_cnt_social_circle	25769
flag_document_8	25024
target	24825
reg_city_not_live_city	24039
obs_30_cnt_social_circle	19971
obs_60_cnt_social_circle	19564
flag_email	17442
reg_region_not_work_region	15612
amt_goods_price	14728
amt_income_total	14035
live_region_not_work_region	12503

Figure 7: Features with more than 10,000 outliers are isolated

3. Data visualization results

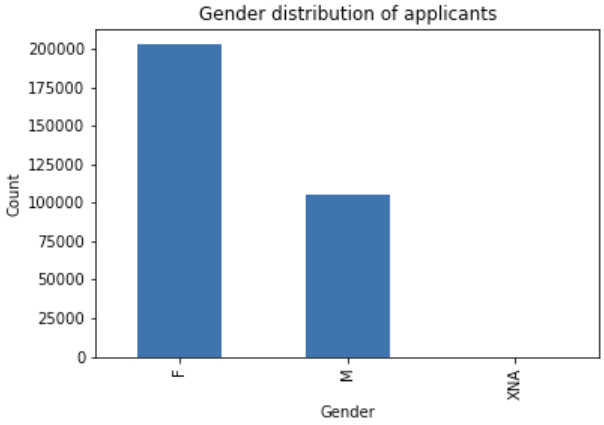

Figure 8: Gender distribution in the applicant data set

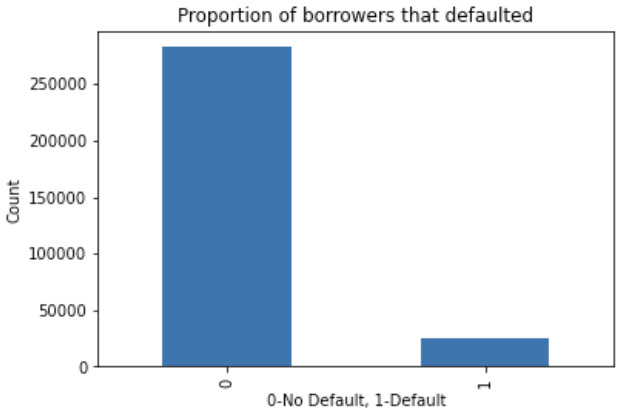

Figure 9: Clients that defaulted versus those that did not default on their payments.

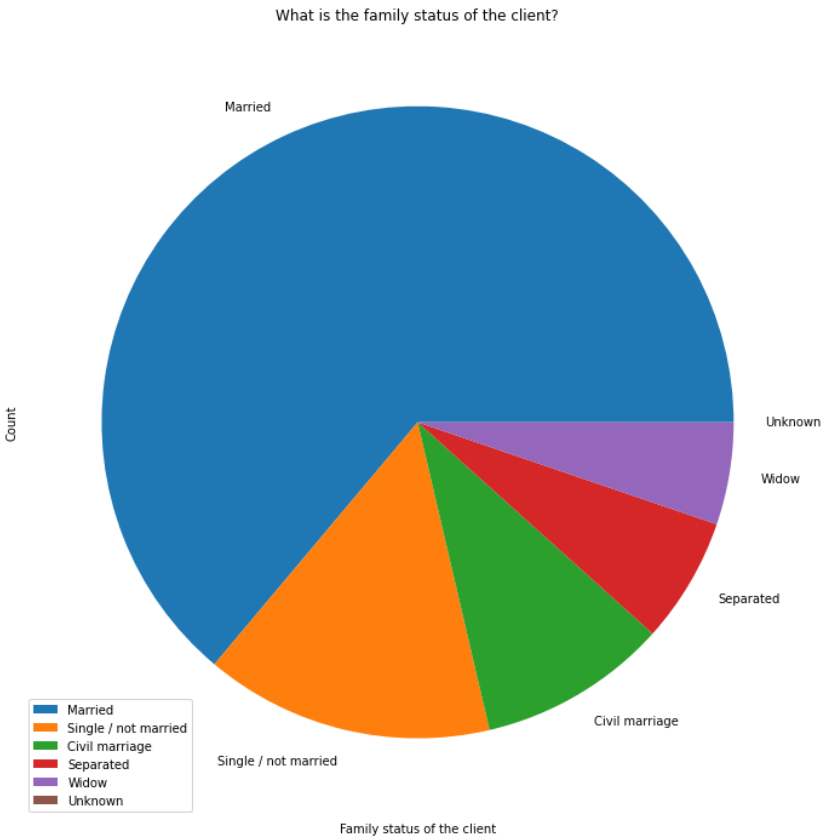

Figure 10: Family status of the applicant

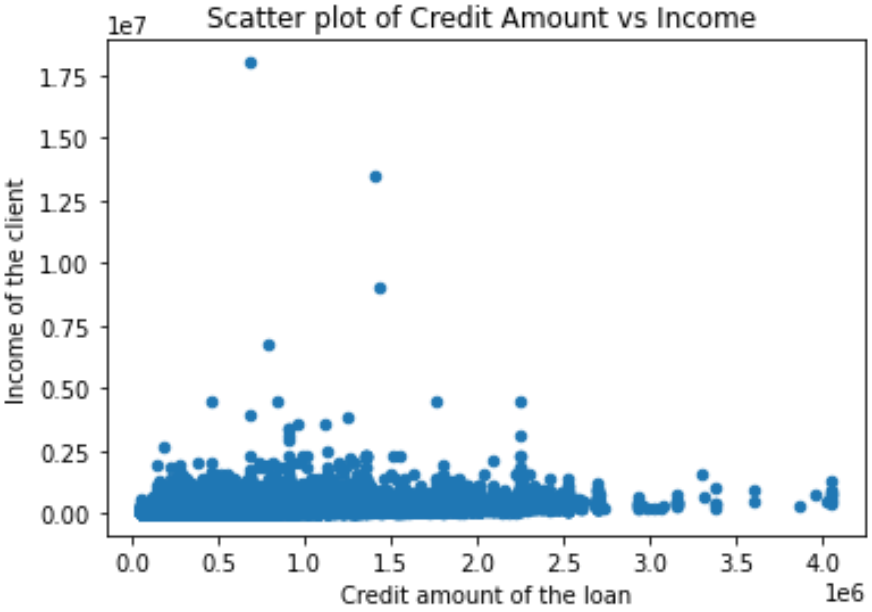

Figure 11: Scatter plot to visualize the credit amount sanctioned and income levels

4. Feature engineering results

Feature engineering was used to identify relevant features that had a high correlation with the outcome variable which is the target variable. The features which had a correlation above 0.01 were selected and the correlation was visualized below.

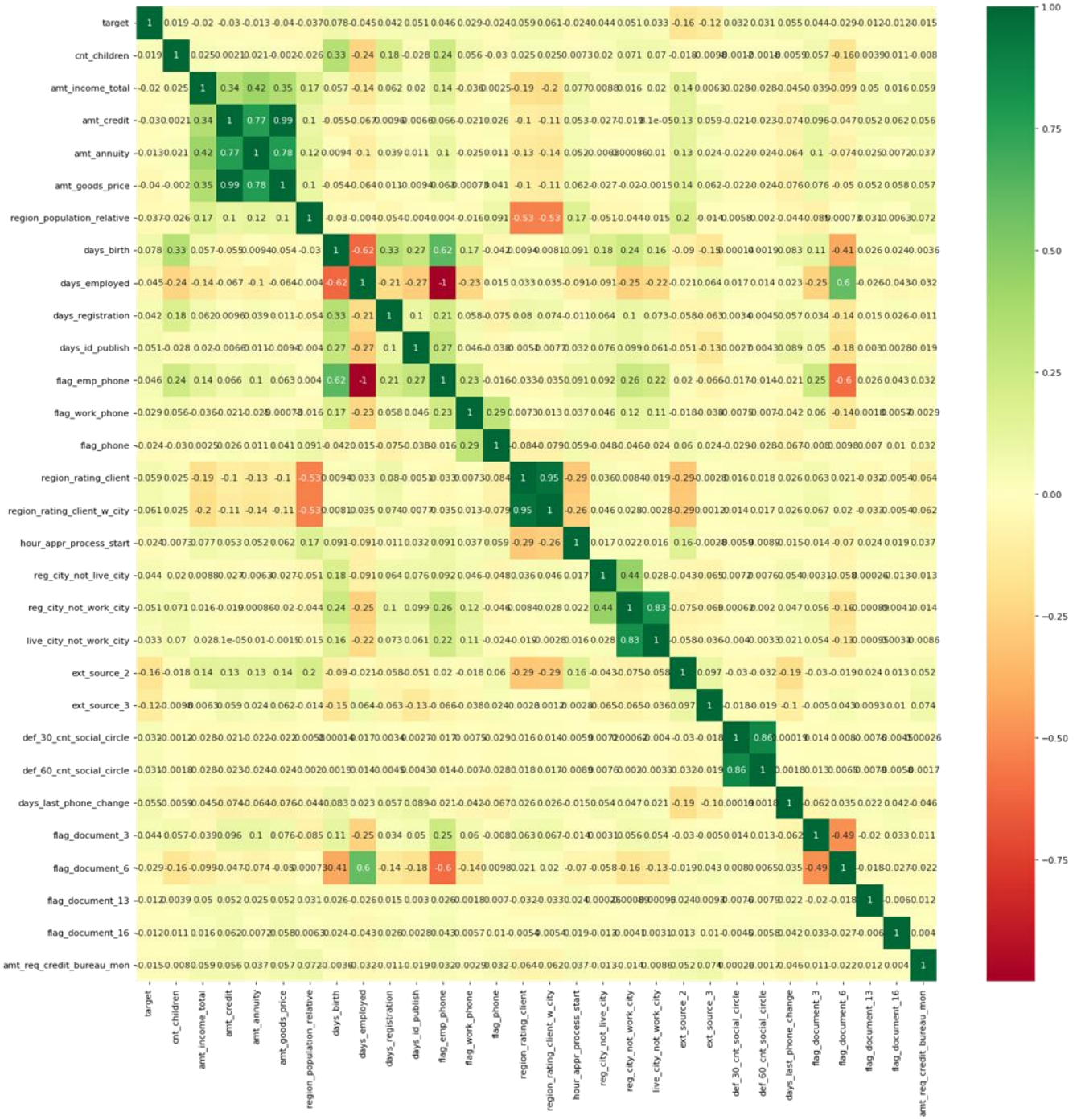

Figure 12: Features with a correlation greater than 0.01 with the target variable

5. Machine learning results

Four different machine learning algorithms were utilized to generate a predictive model and the model with the best accuracy, recall, and precision is chosen to be the final model.

The challenge of minority class imbalance has directed the focus towards optimizing the recall value, it is the model's ability to accurately identify all the true positives of the target variable, which is nothing but improving the model's ability to predict if the client is a defaulter.

5.1 Logistic Regression (without SMOTE)

The overall accuracy score of the logistic regression model is 0.92. And. 38% of the model's predictions were correct, but the model suffered a poor recall score of 0.

```
For the model: LogisticRegression(multi_class='ovr', random_state=0) , the training accuracy
is 0.9191205164059706 and the test accracy is 0.9196936684985854
[[56560     5]
 [ 4934     3]]
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     56565
           1       0.38      0.00      0.00      4937

    accuracy                           0.92     61502
   macro avg       0.65      0.50      0.48     61502
weighted avg       0.88      0.92      0.88     61502
```

Figure 13: Performance of logistic regression model on the imbalanced data set

5.2 Stochastic gradient descent (without SMOTE)

The overall accuracy for this module is 0.92 and just 5% of the model's predictions were correct and the model suffered a poor recall score of 0.

```
SGDClassifier(loss='log_loss', tol=0.01)
The training score for SGDC-SMOTE is  0.9183458755925213
[[42367    56]
 [ 3701     3]]
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     42423
           1       0.05      0.00      0.00      3704

    accuracy                           0.92     46127
   macro avg       0.49      0.50      0.48     46127
weighted avg       0.85      0.92      0.88     46127
```

Figure 14: Performance of Stochastic gradient descent model on the imbalanced data set

5.3 Extreme gradient boosting -XGBoost (without SMOTE)

The overall accuracy of this model is once again at 0.92 And the model's positive predictions were correct 55% of the time and had a low recall score of 0.02.

```
Mean cross-validation score: 0.92
K-fold CV average score: 0.92
[[42263    65]
 [ 3718    81]]
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     42328
           1       0.55      0.02      0.04      3799

    accuracy                           0.92     46127
   macro avg       0.74      0.51      0.50     46127
weighted avg       0.89      0.92      0.88     46127
```

Figure 15: Performance of the XGBoost model on the imbalanced data set

5.4 SMOTE Results

The above models suffered poor scores in predicting the positive values of the target variable, primarily due to the presence of an imbalanced Number of values for the target variable. There were only 24,824 values for the “1”-Class versus 282,686 values for the “0”-Class in the dataset.

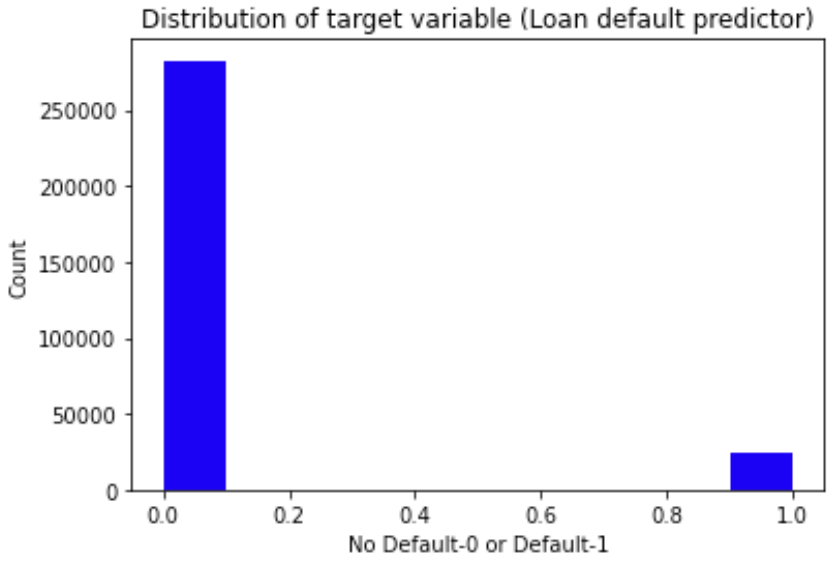

Figure 16: Bar plot of target variable with class-imbalance

```
count=Counter(y_df)
Counter(y_df)
```

```
Counter({1: 24824, 0: 282686})
```

Figure 17: Count of “1 and "0" in target variable with class-imbalance

The synthetic minority oversampling technique is applied to this data set to fix the class imbalance and hence after the class imbalance has been fixed there is equal distribution of both classes in the target variable.

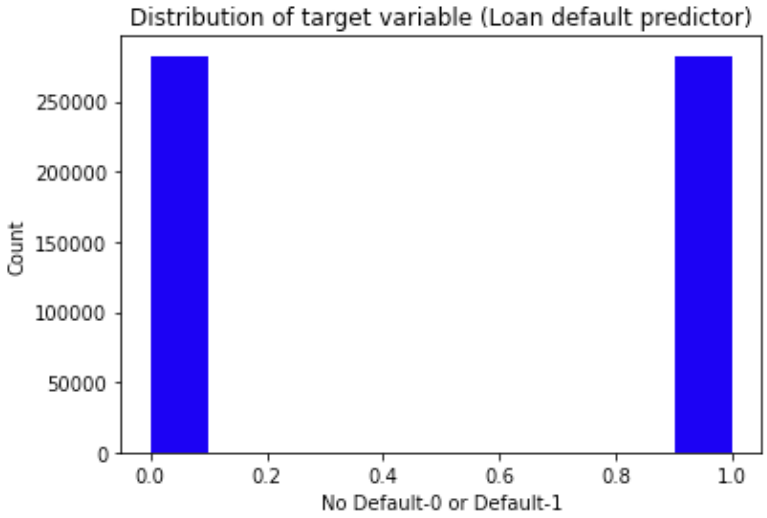

Figure 18: Bar plot of target variable without class-imbalance

```
sample_technique=SMOTE()
X, y = sample_technique.fit_resample(X_df, y_df)
Counter(y)
```

```
Counter({1: 282686, 0: 282686})
```

Figure 19: Count of "1" and "0" in target variable without class-imbalance

5.5 Stochastic gradient descent (with SMOTE)

After fixing the class imbalance. Overall accuracy for this model fell to 0.50 and although the model still suffered the same poor recall score of 0, that model's positive predictions were correct 50% of the time.

```
SGDClassifier(loss='log_loss', tol=0.01)
The training score for SGDC-SMOTE is  0.49991468393519306
[[42402    34]
 [42336    34]]
              precision    recall  f1-score   support

           0       0.50      1.00      0.67     42436
           1       0.50      0.00      0.00     42370

    accuracy                           0.50     84806
   macro avg       0.50      0.50      0.33     84806
weighted avg       0.50      0.50      0.33     84806
```

Figure 20: Performance of Stochastic gradient descent model with SMOTE

5.6 Extreme Gradient Boosting-XGBoost (with SMOTE)

After fixing the class imbalance, the overall accuracy of this model increased from 0.92 to 0.95, and the % of the model's predictions that were correct rose from 55% to 98%. Also, the model was able to identify 91% of the possible defaulters in the training data set.

```
Mean cross-validation score: 0.94
K-fold CV average score: 0.94
[[41720   677]
 [ 3931 38478]]
              precision    recall  f1-score   support

           0       0.91      0.98      0.95     42397
           1       0.98      0.91      0.94     42409

    accuracy                           0.95     84806
   macro avg       0.95      0.95      0.95     84806
weighted avg       0.95      0.95      0.95     84806
```

Figure 21: Performance of XGBoost model with SMOTE

5.7 Random Forest (with SMOTE)

After fixing the class imbalance, the overall accuracy of this model is 0.92, and % of correct predictors was 94%, and predicted 89% of possible defaulters in the test dataset.

```
RandomForestClassifier(n_estimators=50)
Mean cross-validation score: 0.91
K-fold CV average score: 0.92
[[40157  2256]
 [ 4586 37807]]
              precision    recall  f1-score   support

           0       0.90      0.95      0.92     42413
           1       0.94      0.89      0.92     42393

    accuracy                           0.92     84806
   macro avg       0.92      0.92      0.92     84806
weighted avg       0.92      0.92      0.92     84806
```

Figure 22: Performance of Random Forest model with SMOTE

The best machine learning model is the one generated with extreme gradient boosting algorithm after applying synthetic minority oversampling technique to fix the class imbalance of the target variable.

VI. Conclusions

1. The data set has 307511 rows and 122 columns.
2. The variable to predict Credit default is named “target” in the data set.
3. The target variable has two unique values zero and one. the value of 1 indicates that the client is having repayment programs
4. Some of the attributes in the data set have more than 100,000 missing values, which were excluded from the machine learning computational process, and the ones that were included were imputed with Either zero or frequently occurring values in that column.
5. There are more female applicants in the data set than males, as per the visualization (Figure 7) in the data set.
6. The proportion of borrowers with credit default is low in the data set (Figures 16 and 17).
7. Most of the applicants are married (Figure 10).
8. Features having correlation greater than 0.01 correlation with the target variable were selected for computing the machine learning models.
9. Machine learning models without fixing the imbalance in target variable distribution performed poorly in identifying the applicant with repayment difficulties.
10. After analyzing the performance of five different machine learning models, before and after resolving the class imbalance of the target variable. The best machine learning model to predict the minority class of the target variable which is used to identify applicants who can potentially default on their payments is the model generated with extreme gradient boosting algorithm after applying the synthetic minority oversampling technique (SMOTE).

VII. Challenges

1. Simplicity sometimes is preferred over accuracy due to regulatory compliances that require the model predictions to be interpretable and attributable to a specific feature, this would result in choosing the interpretability of the model over accuracy.
2. Privacy concerns related to gathering data for individuals with limited or no credit history where Adhoc data is used to assess the creditworthiness

VIII. Future Scope

1. Use of Adhoc data from bureau will further improvise the model
2. A dynamic model based on geo-spatial information.
3. Use of data scraped from bank statements.

IX. Bibliography

1. SMOTE-TOMEK Technique
https://towardsdatascience.com/imbalanced-classification-in-python-smote-tomek-links-method-6e48dfe69bbc

2. SMOTE Technique
https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification

3. Project on “Insurance Management System” by Surya Vaddhiparthy
https://vaddhiparthy.com/ims/

4. Kaggle
https://www.kaggle.com/competitions/home-credit-default-risk

5. One main Financial’s resources.
https://www.onemainfinancial.com/resources

6. GM Financial’s resources
https://www.gmfinancial.com/en-us/financial-resources.html

7. Dataset Source:
https://www.kaggle.com/competitions/home-credit-default-risk