Machine Learning Assignment Report

Application of Machine Learning on Loan Risk Analysis

BIRMINGHAM CITY
UNIVERSITY

Submitted By: Shubham Singh

Student ID: 21192874

Submission Date: 12th January 2023

Birmingham City University

Module: CMP7228 Machine Learning A S1 2022/3

Table of Contents

Table of Figures

Introduction / Domain Description

Financial institutions receive a lot of applications for loans. One of the main challenges for financial institutions when it comes to giving loans to borrowers is determining the risk of default. Lending money carries inherent risks, and if a borrower defaults on a loan due potential failure in making timely repayment of principal, interest and fees, the lender can incur significant losses (Apostolik, 2009). Therefore, it is important for lenders to carefully assess the risk of lending to a particular borrower before approving a loan.

There are several factors that can impact a borrower's risk of default, including their credit score, debt-to-income ratio, employment history, and financial stability (Serrano-Cinca, et al., 2015). Assessing these factors manually can be time-consuming and may not always provide an accurate assessment of risk.

Hence, machine learning algorithms can help financial institutions determine the risk of lending to a particular borrower by analysing large amounts of data and identifying patterns and trends that may indicate an increased risk of default. By training machine learning models on historical data, financial institutions can create models that can accurately predict the likelihood of default by determining loan application being high risk or low risk for anew loan applicant (Shan & Nilsson, 2018). This can help lenders make more informed lending decisions and reduce the risk of loan defaults.

Problem Definition

The goal of this project is to use data analysis and machine learning techniques, utilizing historical loan data, to make predictions about credit risk for future loan applicants. The project aims to solve credit risk-related problems by utilizing statistics and machine learning algorithms:

1. Machine learning algorithms can be used to predict the likelihood of a borrower defaulting on their loan by analysing their financial risk attributes after they have applied for the loan and has been part of several previous research work (Liu, et al., 2022). Hence, it is possible to predict the likelihood that a borrower will default on a loan, and individuals with a high probability of default can be identified as potential defaulters or high-risk loans. Financial institutions can use this information to decide whether to approve or reject a new loan application based on the predicted risk associated with it. To do this, **binary classification machine learning models can be trained on data from past low-risk and high-risk loans, and then used to predict the risk of default for future loan**

applicants based on their financial information and other loan attributes (Li & Han, 2018). It is called Probability of Default (PD) modelling (Kenton, 2021).

2. In case the loan borrower gets *defaulted* - failure to make required interest or principal repayments on a debt (Investopedia, 2022) or *charged off* - debt that a creditor does not expect to be able to collect because the borrower has not made payments on it for a long time (Investopedia, 2021), the financial institution is exposed to certain amount of loss on proportion of total funded amount (Georgiev, 2023). Hence, to determine exposed loss in such situation, BASEL II accord (bis.org, 2001) suggested the use of credit risk metric tool called *Credit Conversion Factor* (CCF) - percentage of an unpaid debt that will be taken out at the time of default, the value of which ranges from 0 to 1 and can be expressed in percentage (Koulafetis, 2017). Therefore, **regression machine learning algorithms can be utilized to predict the CCF and loan attributes contributing towards estimation of CCF** (Tong, 2016).

3. **Using unsupervised machine learning algorithms, loans based on the financial information of previous borrowers, such as annual income and loan amount requested, could be grouped together into clusters**. These clusters can then be associated with a specific risk or probability of default using their loan status outcomes. Therefore, In order to assess the risk or probability of default for a new loan applicant, their requested loan amount and financial attributes can be used to assign them to a specific cluster. This allows for the determination of the corresponding risk or probability of default for the loan based on the cluster.

Dataset Description

This project uses Lending Club dataset for analysis and tackling the challenges, as mentioned in previous section, using data analysis and machine learning algorithms. The Lending Club dataset is a collection of data on loans made through the online USA lending platform Lending Club (LendingClub, 2022). The dataset includes information on the borrower, the loan, and the loan performance.

The Lending Club dataset has **total 142 columns and 2925493 rows**, which includes a wide range of variables, including information on the borrower's credit history, debt-to-income ratio, employment status, and loan purpose. It also includes information on the loan, such as the loan amount, interest rate, and loan term. Additionally, the dataset includes information on the past loan status, such as whether the loan was fully paid off, charged off, or defaulted between 2007 to 2020.

In the context of loan risk analysis, (LendingClub, 2020) "fully paid off" refers to a loan that has been fully repaid by the borrower according to the terms of the loan agreement. "Charged off" refers to a loan that has been deemed uncollectible by the lender, usually because the borrower has defaulted on the loan and the lender has given up on collecting the debt. "Defaulted" refers to a loan on which the borrower has failed to make the required payments and is in default of the loan agreement.

The dataset and its description file can be downloaded from table given below.

Item	Download Link	Comment
Dataset	Loan_status_2007-2020Q3.gzip	1.65 GB original dataset in compressed .gzip format
Description File	Loan Dataset Columns and Descriptions	Excel file containing descriptions of all columns in the dataset

Dataset Exploration using Exploratory Data Analysis (EDA)

Exploratory data analysis (EDA) is a process of analyzing and understanding a dataset by using statistical techniques and visualizations to uncover patterns, trends, and relationships in the data (Tukey, 1977). This project uses two types of EDA –

1. **Univariate EDA**: This involves analyzing a single variable in a dataset to understand its distribution, identify patterns and trends, and identify any unusual or extreme values (Fuentes, 2018).

2. **Bivariate EDA**: This involves analyzing the relationship between two variables in a dataset to understand how they are related and identify any patterns or trends (Fuentes, 2018).

As there are more than 2.5 million records of loan in the given dataset, **EDA is done using first 1 million records**. All the EDA analysis is done using python in colab notebook, which can be accessed using the link below:

EDA Colab Notebook Link

Univariate EDA

Univariate exploratory data analysis involves analyzing a single variable in a dataset. To conduct a univariate exploratory data analysis, it is typically necessary to first select a single variable from the dataset and calculate some basic statistics, such as the mean, median, and standard deviation (Fuentes, 2018).

Part A: Exploring Data Type of each column

[Code Section Link]

As the dataset has several types of columns – numerical, date and categorical, it is important to know the data type of each column in a dataframe for univariate analysis because different types of data require different types of analysis.

Due to the massive number of columns, an excel file prepared using Python code that shows the data type of each column, can be downloaded from the link:

[Excel Sheet Link for Dataset Columns & corresponding Datatypes]

Overall, there are 106 float type columns, 35 object type columns and 1 integer type column in the dataset, as can be seen in below bar-chart:

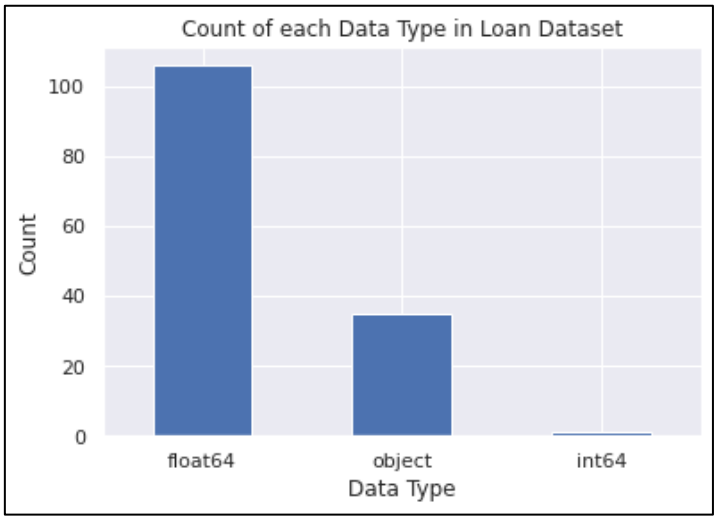

Figure 1: Count of each Datatype in Dataset

Part B: Univariate EDA on key loan and borrower-related columns

1. Loan Amount

[Code Section Link]

As per the historical loan dataset, loan amount is around 15262 USD ± 9670, minimum amount is 500 USD and maximum amount is 40000 USD, lend to borrowers between 2007 to 2020.

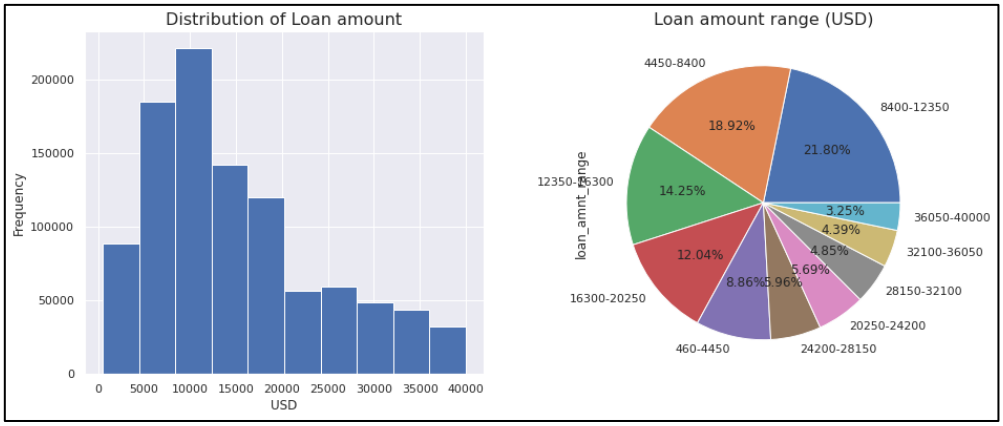

Figure 2: Distribution of Loan Amount (Histogram) and Loan Amount Range (Pie-Chart)

The most common loan amount range, according to the histogram and pie-chart, is between 8400 and 12350 USD, representing 21.8% of loans. The second most common range is between 4450 and 8400 USD, representing 18.92% of loans. The least common range is between 36050 and 40000 USD, representing only 3.25% of loans.

2. Interest Rates

[Code Section Link]

The average interest rate given on loans is around 13% ± 5, with minimum interest rate around 5.3% and maximum interest rate around 30%.

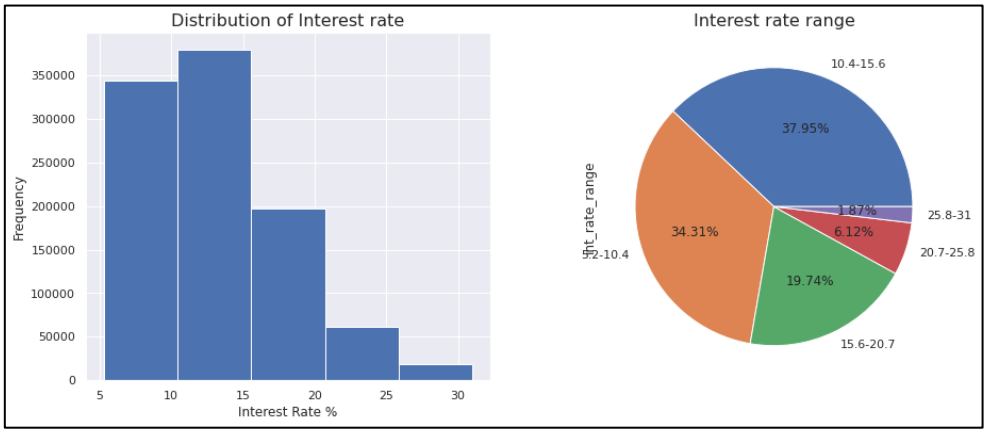

Figure 3: Distribution of Interest Rate (Histogram) and Interest Rate Range (Pie-Chart)

According to the visualization given above, **the most common interest rate range for loans is between 10.4% and 15.6%, accounting for approximately 38% of loans**. The second most common range is between 5.2% and 10.4%, accounting for approximately 34% of loans. The least common range is between 25.8% and 31%, representing only 1.8% of loans.

3. FICO Scores

[Code Section Link]

(FICO®score, 2022) FICO scores are credit scores that are used to evaluate the creditworthiness of an individual. In general, higher FICO scores indicate a lower credit risk, while lower FICO scores indicate a higher credit risk. Lenders typically consider a FICO score of 700 or higher to be good, while a score below 600 is generally considered poor.

The dataset includes both high and low FICO scores for each loan record, so an **average FICO score using high and low FICO scores has been calculated for analysis**. The average score is approximately 706 with a standard deviation of 35. **The highest historical FICO score in the dataset is 847, and the lowest is 612.**

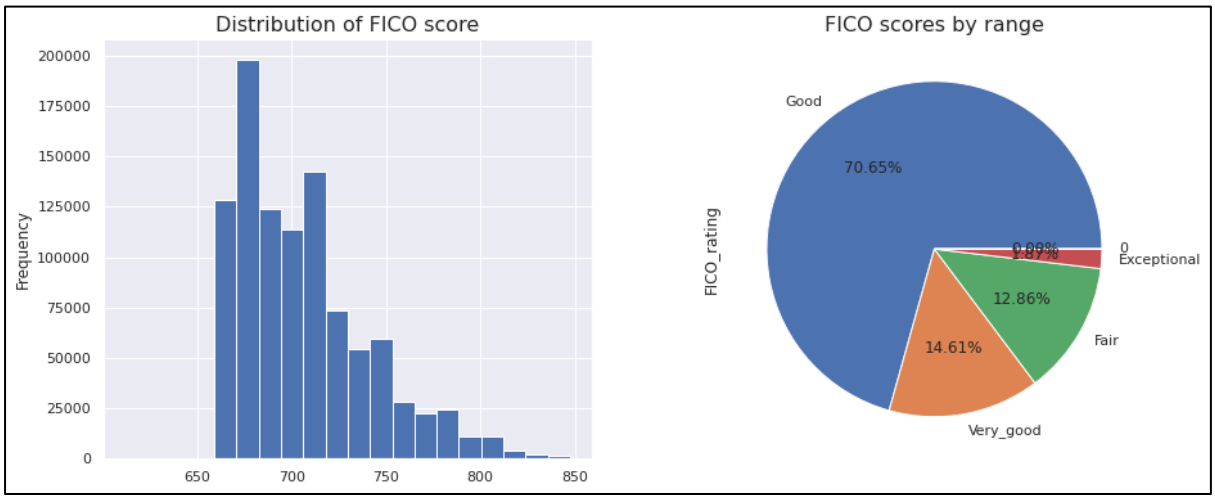

Figure 4: FICO Scores Distribution (Histogram) and FICO Scores Range (Pie-Chart)

The FICO scores are categorized into 'Bad', 'Good', 'Fair', 'Very Good' and 'Exceptional' and their respective proportions, as shown in above pie-chart. **About 70.65% loans applicants have 'Good' FICO scores**.

According to the FICO score histogram, the most common FICO score among loan applicants is around 670. The number of applicants with exceptional scores above 800 is rare, representing only 1.8% of the total number of applicants.

4. Loan Purpose and Term

[Code Section Link]

The given loan dataset has two separate columns for loan purpose and term. The loan term is the length of time in which borrower must repay the loan.

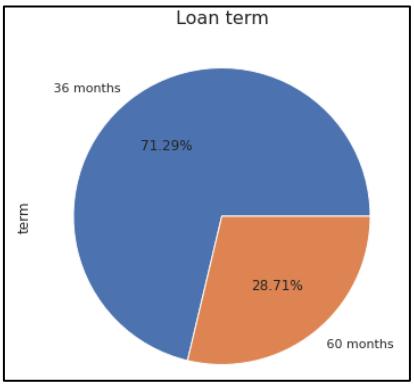

Figure 5: Loan Terms (Pie-Chart)

There are two types of loan terms available: 36 months and 60 months. The majority of loans, approximately 71.29%, have a term of 36 months, while 28.71% have a term of 60 months. In other words, 36-month loans are much more common than 60-month loans.

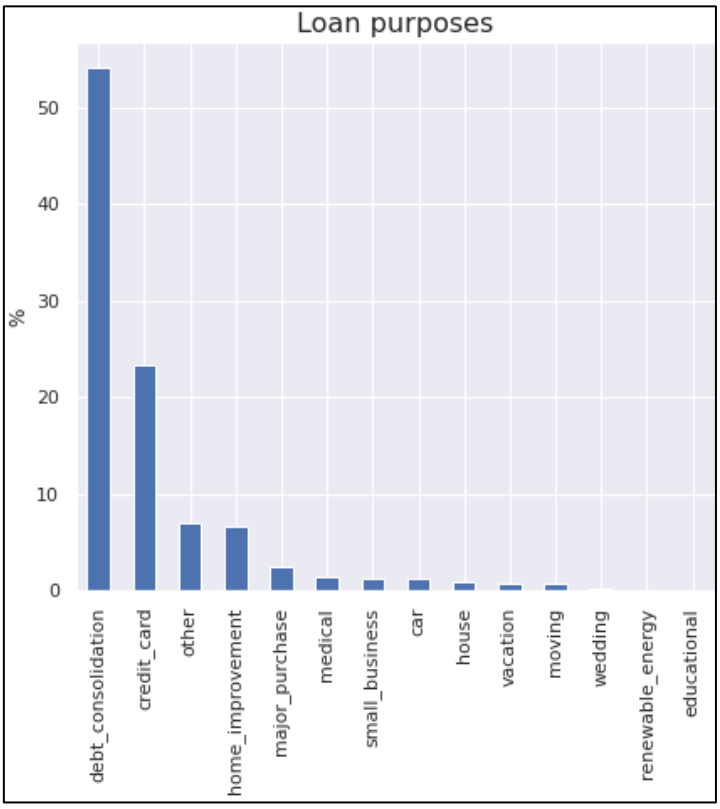

Figure 6: Percentage Distribution of each Loan Purpose (Bar Graph)

The above given bar graph displays the distribution of loan data based on the purpose of the loan. **The primary reason that borrowers are issued loans is for debt consolidation, which represents around 54% of loans**. Debt consolidation is a financial strategy that involves taking out a new loan to pay off multiple smaller debts (Investopedia, 2022). The second most common reason is credit card debt, accounting for approximately 23% of loans. **In contrast, loans for education, renewable energy, or weddings are relatively rare, representing a small percentage of the total number of loans.**

5. Grade and Sub-Grade

[Code Section Link]

Grades and Sub-Grades are typically used by lenders to determine the interest rate and other terms of a loan (LendingClub, 2019). The higher the grade, the lower the credit risk, and the more favorable the loan terms are likely to be. Sub-grades are used to further refine the assessment of credit risk within a particular grade. For example, within the "A" grade, there may be several sub-grades, such as A1, A2, and A3, with each sub-grade representing a different level of credit risk.

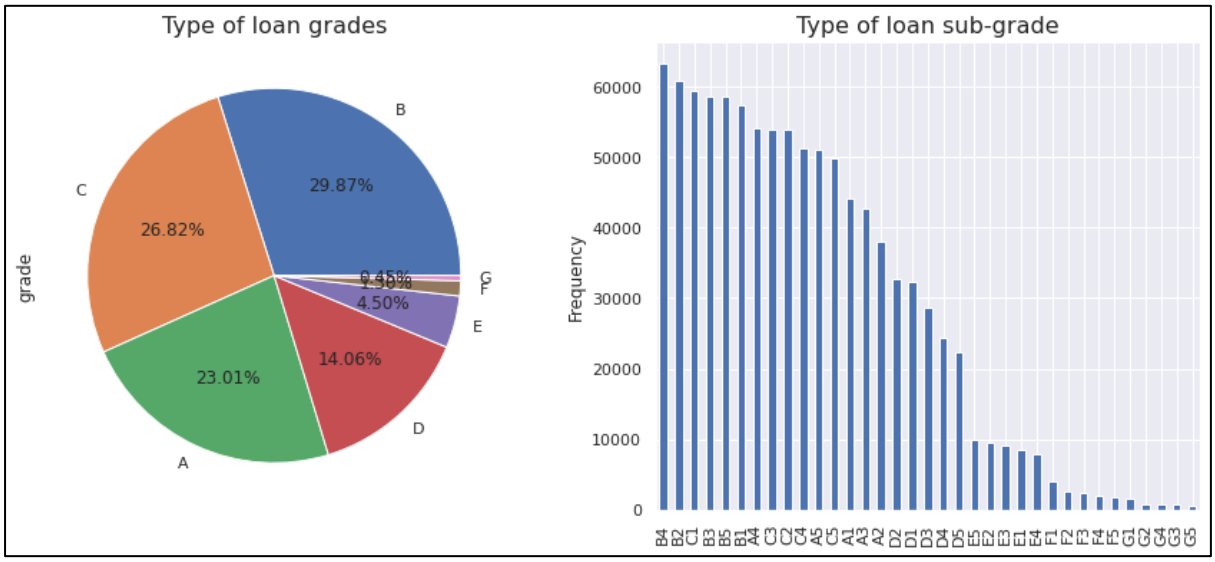

Figure 7: Proportion of each Loan Grade (Pie-Chart) and Total Count of each Loan Sub-Grade (Bar Graph)

The majority loan Grade types are A, B and C, which respectively accounts for about 23.01%, 29.87% and 26.82% of total number of grade types. There are 35 sub-grades in total. Rated loans B4, B2, C1 and B3 the most common one compared to the rest of the sub-grades.

6. Employment Length, Home Ownership and Verification Status

[Code Section Link]

Employment length is often seen as an indicator of financial stability and can affect the risk of a loan. Borrowers with a longer employment history may be seen as having a lower risk of defaulting on a loan, while borrowers with a shorter employment history may be seen as having a higher risk (Croux, et al., 2020).

Home ownership can also affect the risk of a loan. Borrowers who own their own home may be seen as having a lower risk of defaulting on a loan, since they have a stake in the property and may be more motivated to make timely payments. Borrowers who rent their home or do not have a stable living situation may be seen as having a higher risk (Croux, et al., 2020).

Verification status refers to the process of verifying the information provided by the borrower, such as their income and employment history. Borrowers who can provide documentation to verify their information may be seen as having a lower risk of defaulting on a loan, while borrowers who are unable to provide verification may be seen as having a higher risk (Croux, et al., 2020).

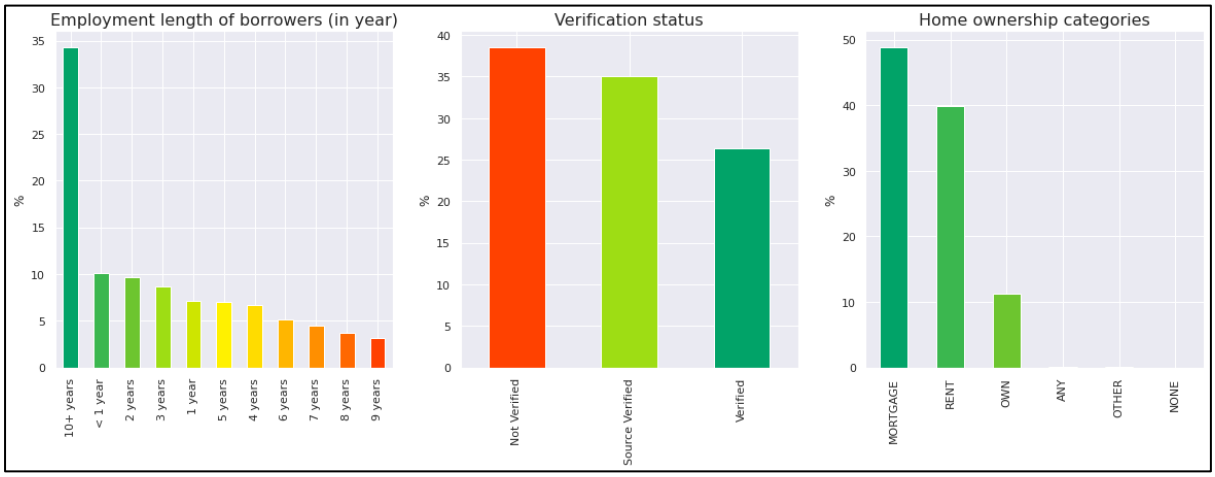

Figure 8:Percentage Distribution of Employment Length of Borrowers (Bar Graph), Verification Status (Bar Graph) and House Ownership Status (Bar Graph)

A significant portion (nearly 35%) of applicants have been employed for more than 10 years. The second largest group (around 10%) consists of applicants with no job or less than 1 year of employment. The smallest group (less than 5%) includes applicants with 9 years of employment.

The next graph, which shows the verification status of applicants, indicates that the majority of applicants' income information is verified either by the lender itself or from a verified source. **Around 38% of applicants' incomes were not verified at the time the data was exported.**

According to the third bar graph, the most common type of home ownership is mortgage, representing 49% of the loan applicants. The second most common type is rent, representing 40% of the loan applicants in USA.

7. Loan Status and Application Type

[Code Section Link]

In the context of loan risk, there are several different loan statuses that can be used to describe the status of a loan. (LendingClub, 2020) These statuses can indicate whether a borrower is current on their payments, if they are in a grace period, or if they are late on their payments. Here is a brief explanation of each loan status:

- **Current**: A loan is considered current if the borrower is making timely payments according to the terms of the loan.

- **In Grace Period**: A loan is in the grace period if the borrower is allowed a certain amount of time after the due date of a payment to make the payment without incurring a late fee or penalty.

- **Late (16-30)**: A loan is considered late if the borrower misses a payment and is within 16 to 30 days of the due date.

- **Late (31-120)**: A loan is considered late if the borrower misses a payment and is more than 30 days past the due date but within 120 days.

- **Fully Paid**: A loan is considered fully paid if the borrower has made all of the required payments and the loan has been fully repaid.

- **Default**: A loan is considered in default if the borrower has failed to make the required payments and the lender has initiated collection proceedings or taken other legal action.

- **Charged Off**: A loan is considered charged off if the lender has determined that the borrower is unlikely to repay the loan and has written off the debt as a loss.

Overall, these loan statuses are used to describe the status of a loan and the borrower's progress in repaying the debt.

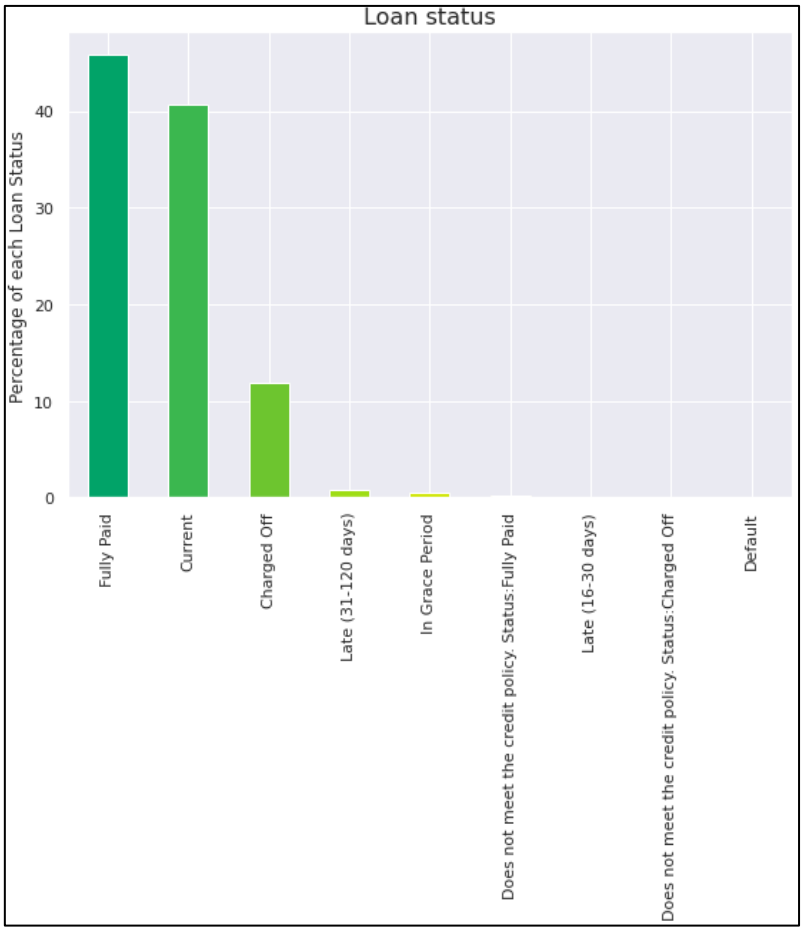

Figure 9: Percentage Distribution of each Loan Status (Bar Graph)

According to analysis from above loan-status bar chart, a significant number of borrowers have a loan status of "Current" (about 41%) or "Fully Paid" (47%). This means that about 47% of loan applicants were able to successfully repay their loans. The "Charged Off" status represents about 12% of borrowers, while the percentage of borrowers who are "Late" or in "Default" is extremely low, less than 1%.

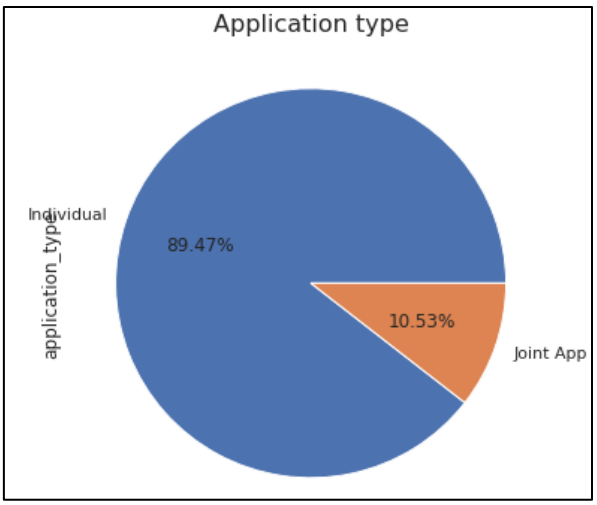

Figure 10: Proportion of Each Application Type (Pie-Chart)

According to the above application-type pie chart, **approximately 89.47% of loan applications are made by a single borrower** (individual application type), while only 10.53% are made by two or more borrowers (joint application type).

Part C: Analyzing Missing values in each column

[Code Section Link]

It is essential to address missing values in the dataset as they can significantly affect the accuracy and trustworthiness of the analysis performed using the data. The following horizontal bar graph shows the columns in the loan dataset that have a high number of missing values (in decreasing order):

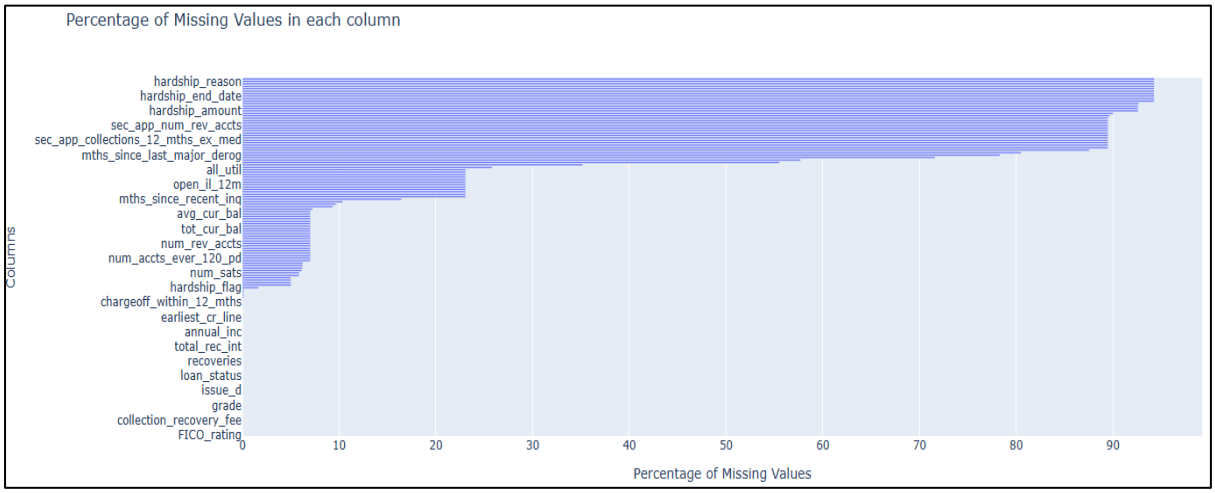

Figure 11: Percentage of missing values in each column (Horizontal Bar Graph)

It is evident that majority of columns contains high percentage of missing values in them. An interactive horizonal bar graph and a table with the column name and percentage of missing values for columns with more than 80% missing values is available at this [link] for more detailed visibility.

Bivariate EDA

Bivariate analysis is a statistical methodology used to identify the relationship between two variables in a dataset (Nordhausen, 2009). The following key bivariate analysis can be used to understand the relationships using important loan and borrower attributes in a loan dataset.

Part A: Bivariate Analysis between important loan attributes

1. Loan Status with Term

[Code Section Link]

According to the below given stacked bar graph between loan status and term, **there are more than twice as many loans with a term of 36 months as there are loans with a term of 60 months for borrowers** with a loan status of "Charged Off," "Current," "Default," "In Grace Period," or "Late." However, **for borrowers with a status of "Does not meet the credit policy. Status: Charged Off" or "Does not meet the credit policy. Status: Fully Paid," the number of loans with a term of 36 months is significantly higher than the number of loans with a term of 60 months**.

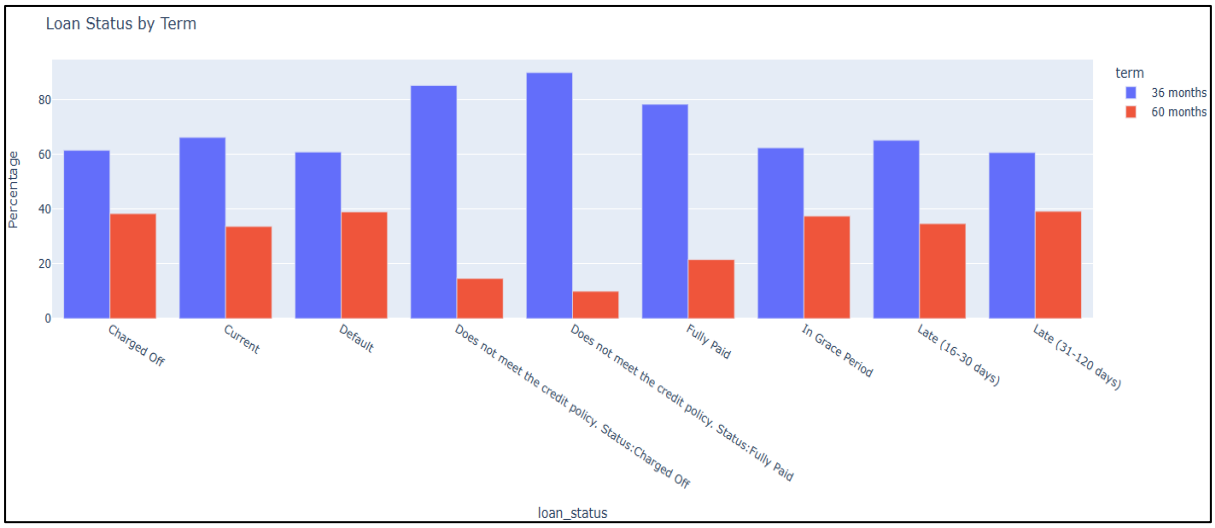

Figure 12: Percentage Distribution of Loan Status by each Loan term (Grouped Bar Chart)

2. Loan Status with Employment Length

[Code Section Link]

Employment length of a loan applicant seems to a very important consideration for any financial institution before lending loans. Borrowers who have a longer employment history may have had more time to build up their credit history and may have a better credit score, which could make them less risky for lenders. Similarly, borrowers with a shorter employment history may have a less established credit history and may be seen as being riskier for lenders.

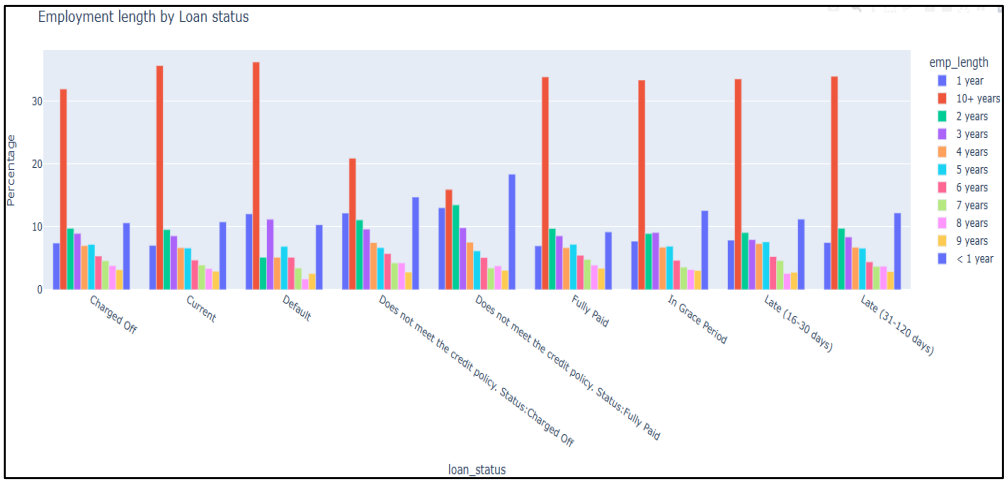

Figure 13: Percentage Distribution of each employment length by Loan Status (Grouped Bar Chart)

According to the historical loan data, **borrowers who have more than 10 years of employment make up approximately 33% on average across all loan statuses (excluding the "Does not meet the credit policy" status).** Borrowers who do not have a job or have been employed for less than 1 year make up around 10% of all loan applications across all statuses. This information suggests that a **large percentage of loans were issued to borrowers who had been employed for 10 years or more**. This may be because borrowers with a longer employment history are generally seen as being more financially stable and less likely to default on their loans.

3. Loan Amount and Purpose of loan

[Code Section Link]

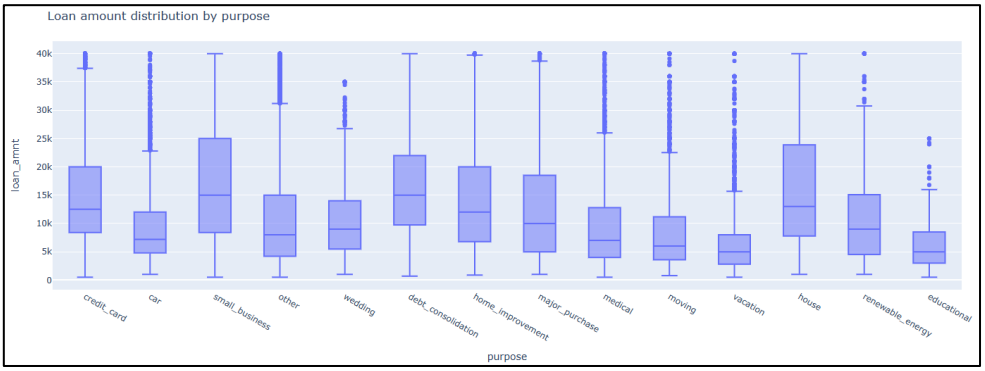

Figure 14: Distribution of Loan Amount by each Loan Purpose (Box Plot)

According to analysis from loan amount boxplot for each purpose, borrowers typically request a median loan amount of around 15,000 USD for the purposes of small business and debt consolidation. However, for vacation and educational purposes, the typical loan amount requested is around 5,000 USD. **This suggests that borrowers may need larger loan amounts for certain purposes, such as small business and debt consolidation, compared to other purposes such as vacation and education**.

4. Loan Status with Interest Rates and Loan Amount

[Code Section Link]

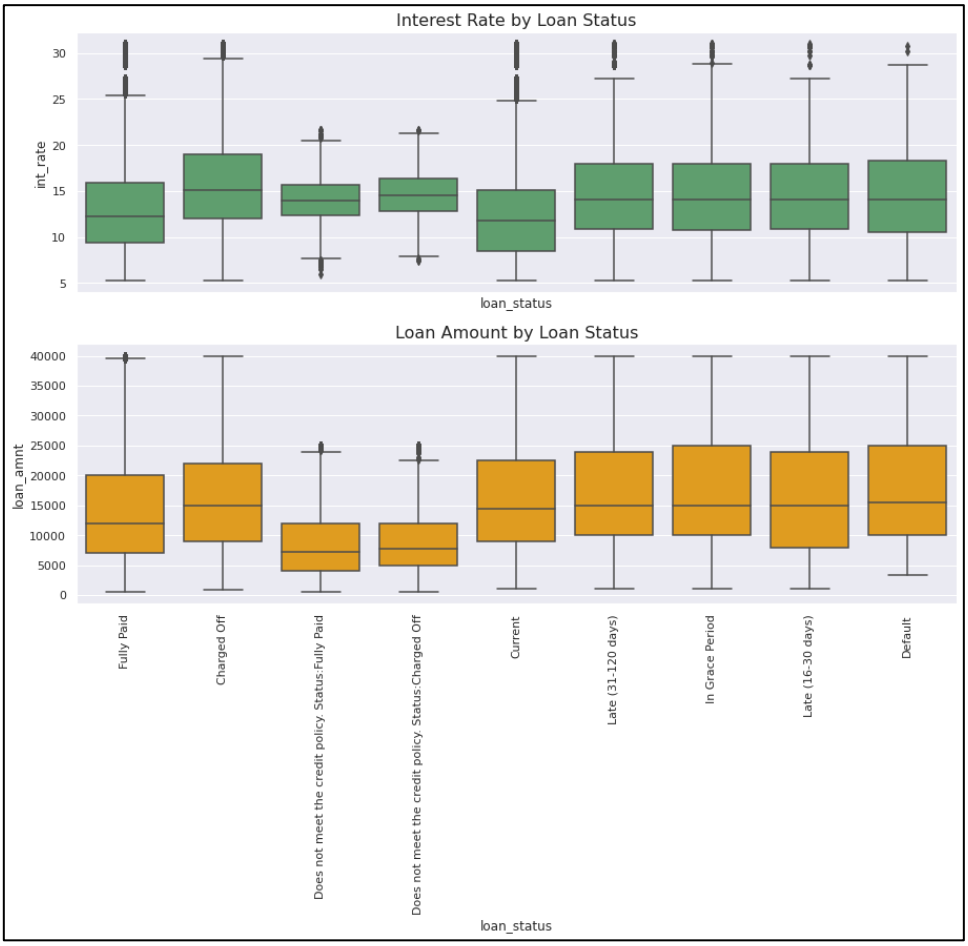

Figure 15: Distribution of Interest Rate by Loan Status (Box Blot) and Loan Amount by Loan Status (Box Plot)

According to the information from the above interest rates and loan amount boxplots for each loan status, **borrowers who have a loan status of "Charged Off" or "Default" tend to have higher average interest rates (around 15% on median average).** The median average loan amount for these two statuses (around 15,000 USD) is similar to the median average loan amount for other statuses, with the exception of "Fully Paid." This suggests that **borrowers with a "Charged Off" or "Default" status tend to have higher interest rates and loan amounts.**

5. Interest Rates and Loan Grades

[Code Section Link]

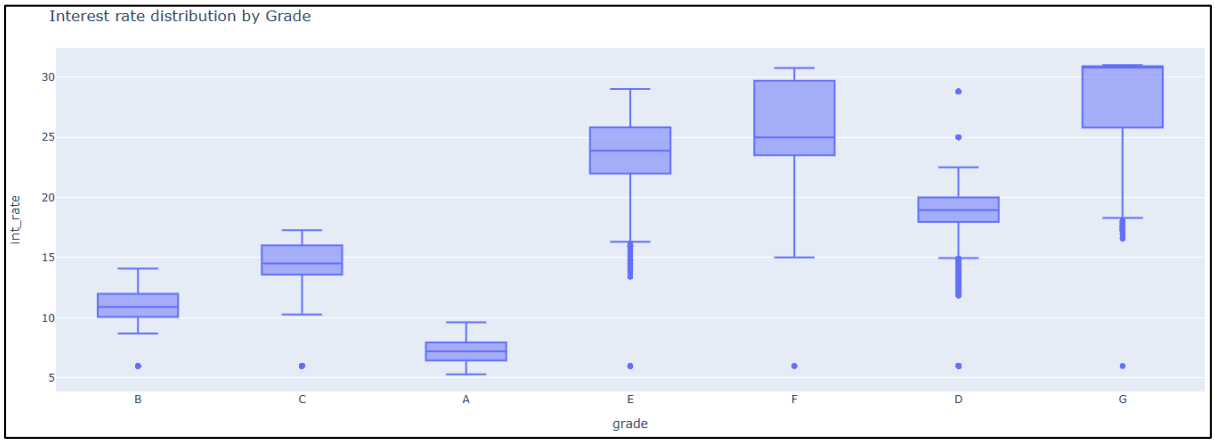

Figure 16: Distribution of Interest Rates by Grade (Box Plot)

According to the analysis from interest rates boxplot for each grade, grades G, F, and E (in descending order) have the highest median interest rates at 30.79%, 24.99%, and 23.88%, respectively. These loan grades are considered to be riskier and therefore have higher interest rates assigned to them. On the other hand, grades A, B, and C (in descending order) have the lowest median interest rates at 7.21%, 10.91%, and 14.52%, respectively. (LendingClub, 2019) Lending Club has a base interest rate of around 5.05% for all loans, so there are no interest rates below 5%.

Part B: Correlation Analysis

Correlation is useful for statistically analysing the relationship between two variables by determining strength and direction of the relationship between them (365DataScience, 2021).

[Code Section Link]

1. Correlation between numerical columns

One of the ways to determine the correlation between numerical or continuous columns is by using Pearson's correlation. (Kotu & Deshpande, 2019) It could be calculated using formula given below:

$$r = \sum (x - \bar{x})(y - \bar{y}) / (\sum (x - \bar{x})^2 \sum (y - \bar{y})^2)^{0.5}$$

where:

- x and y are the values of the two variables being correlated
- $\bar{x}$ and $\bar{y}$ are the means of the two variables
- Σ represents the sum of the values
- r is the correlation coefficient

(Kotu & Deshpande, 2019) The strength of the correlation is usually measured using a correlation coefficient, which is a numerical value that ranges from -1 to 1. **A value of 1 indicates a perfect positive correlation, a value of -1 indicates a perfect negative correlation, and a value of 0 indicates no correlation**. A correlation coefficient close to 1 or -1 indicates a strong correlation, while a correlation coefficient close to 0 indicates a weak correlation.

Pearson's correlation only measures the strength of a linear relationship between two variables (David M. Levine, 2020) and hence, may not be appropriate for categorical variables.

The following heatmap shows the correlation between selected numerical loan and borrower attributes from the loan dataset.

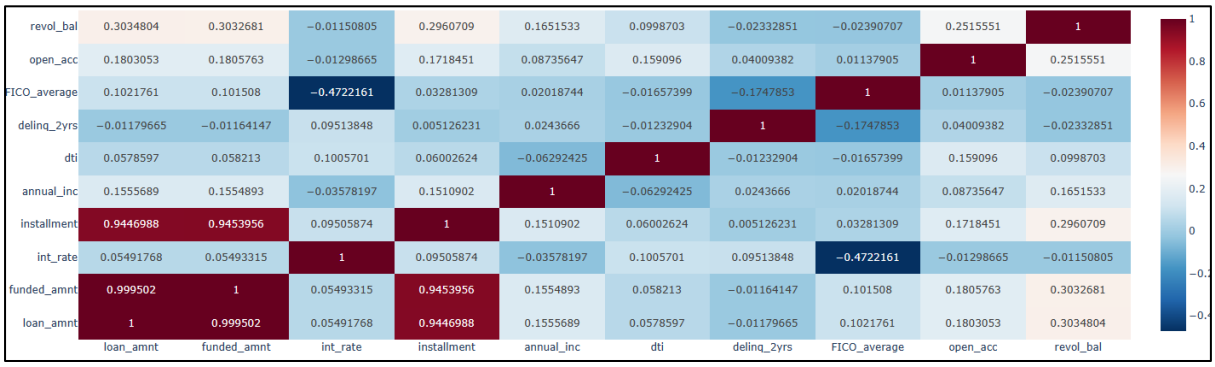

Figure 17: Correlation heatmap between important numerical columns using Pearson's Correlation

Key analysis from the above heatmap is as follows:

a. The loan amount and funded amount have a strong positive correlation of 0.99, meaning that the higher the loan amount requested from the bank, the more likely it is that a larger amount will be approved.
b. There is a strong positive correlation of 0.94 between the funded amount and the number of instalments, indicating that it is more likely to require a larger number of payments to pay back a larger funded amount.
c. The interest rate and the average FICO rating have a negative correlation of 0.47, indicating that a higher FICO credit score is generally associated with a lower interest rate, and vice versa.

2. Correlation between categorical columns

To determine correlation between the categorical columns, 'chi-squared test of association' is used by comparing the frequencies of categories from two categorical variables (Turney, 2022). If the chi-squared statistic (calculated from formula given below in chi-squared test) is large, it indicates that the observed frequencies are significantly different from the expected frequencies, suggesting that there is an association between the two categorical columns.

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

The following heatmap shows the correlation (in terms of chi-squared statistic) between selected categorical loan and borrower attributes from the loan dataset:

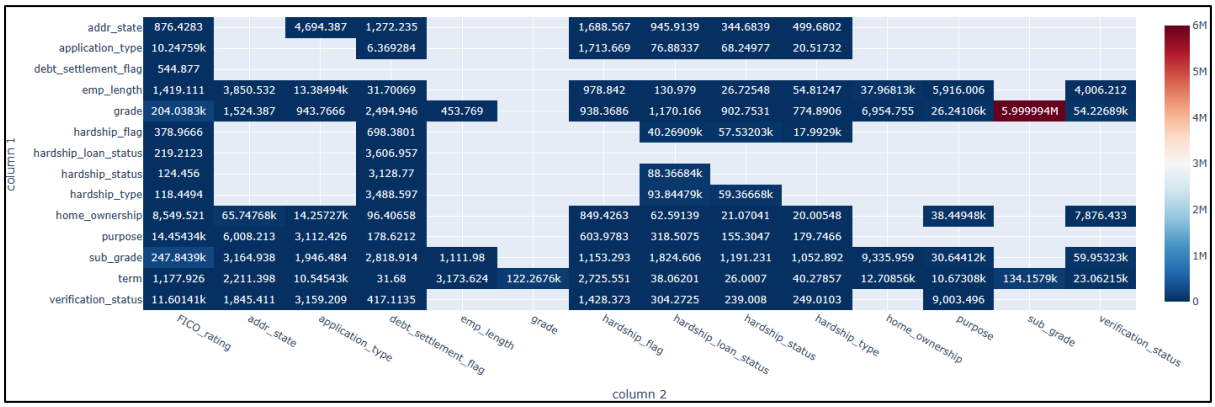

Figure 18: Correlation Heatmap between important categorical columns using 'chi-squared test of association'

As expected, there is a significant relationship between the borrower's FICO ratings, grade, and sub-grade of a loan, as well as between the borrower's loan application type and their home ownership status and loan purpose. Additionally, the data suggests that borrowers who have been employed for a longer period of time are more likely to receive a higher sub-grade for their loan, indicating that it is perceived as less risky to lend loan to them.

Problem 1 – Prediction of Loan approval and Probability of Default (PD): Classification Problem

Predicting loan approval and Probability of Default (PD) is a common classification problem in financial industry. **The objective is to use statistical analysis along with models using machine learning algorithms to predict whether a new application should be approved or not based on model's prediction of its risk level (high-risk or low-risk) and if approved, what the likelihood is that borrower will default in future**. In addition, machine learning techniques can also be used to identify the attributes that are most relevant to the risk of default, helping lenders to better understand and manage their risk.

The code, experiments, and analysis for this classification problem statement, including detailed comments, are available in a Google Colab notebook and can be accessed using the following link:

[PD or Loan Approval Prediction Google Colab Link]

Data Preparation

To prepare the Lending Club dataset for predicting the loan risk along with probability of default using classification machine learning algorithms, first 1 million historical records are used. Given below are the steps taken for preparing the dataset by removing the irrelevant columns for further analysis and pre-processing. **For code reference, each step is hyper-linked to their respective code section in google colab notebook**.

1. Remove the columns which are not related to the purpose

There are certain columns in the dataset, which are not important contributors towards analysis and prediction of loan defaults. This includes URL, zip codes, titles of loan, employee titles and loan inquiry related columns. Hence, these columns are simply removed in dataset for further analysis. Refer appendix section to check the column names removed.

2. Remove the categorical columns which has high proportion of only one category

Upon careful inspection, it was discovered that some of the categorical columns had a very high proportion of one category (more than 97%), while the other categories were rare (about 2-3%) in their respective columns. **This indicates that other categories do not contribute significantly to the variation in the data and are not useful for analysis.** The donut pie chart below illustrates the proportions of each category and supports the decision to remove these columns from the analysis. Refer appendix section to check the column names removed.

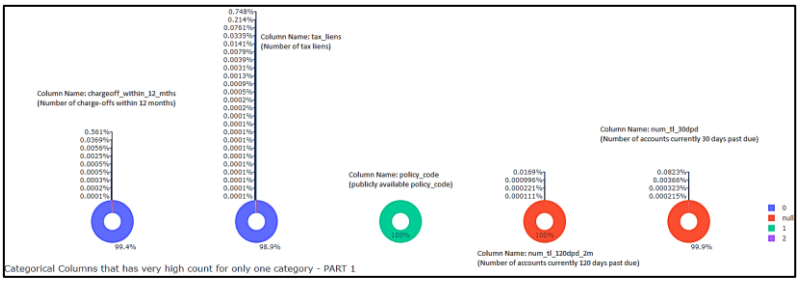

Figure 19: Categorical Columns with high proportion of only one category – PART 1

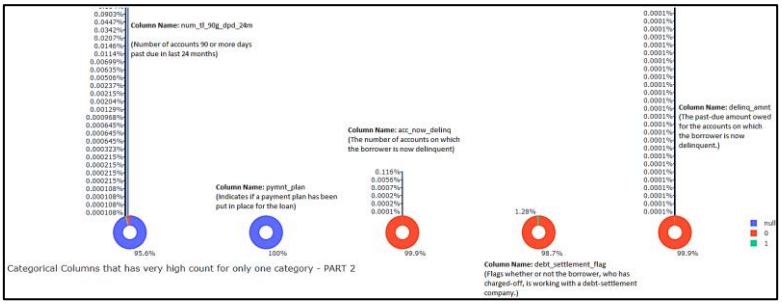

Figure 20: Categorical Columns with high proportion of only one category – PART 2

3. Remove the columns which have values from future, i.e., when loan has been granted

There are columns in the dataset that contain information about the loans after they have been granted. However, **to predict whether a loan applicant will default or not, it is necessary to use only those attributes that are relevant at the time when the loan application is made and before the loan is granted**. Therefore, these columns such as funded amount, recoveries, remaining outstanding principal, last payment amount, loan issued date, next and previous payment date, installments etc. must be removed from the data.

Using columns that contain information about the loans after they have been granted could lead to overfitting of the machine learning model. **This is because the model may learn patterns in the data that are not relevant for predicting default and may perform poorly on new loan data**. To avoid this issue, it is important to use only data that is available at the time the loan application is made and before the loan is granted.

Refer appendix section to check the column names removed.

4. Remove the columns which has more than 80% missing values

Based on the missing value analysis, there are columns with more than 80% missing values, and some even have around 94% missing values. **While there are techniques to impute missing values, filling in a large number of missing values in these columns could potentially compromise the quality of the analysis**. Hence, in order to maintain the quality of the analysis, columns with more than 80% missing values are removed. Refer appendix section to check the column names removed.

5. Remove the numerical columns which are highly correlated

Multi-collinearity occurs when two or more features/columns are highly correlated, which can negatively impact a machine learning model, including overfitting, the curse of dimensionality, and reduced interpretability.:

- **Overfitting**: Multi-collinearity can cause a machine learning model to overfit by relying too heavily on correlated variables and fitting the training data too closely, resulting in poor performance on new data (J.A Cook, 2016).
- **Curse of dimensionality**: Multi-collinearity can cause the curse of dimensionality in some machine learning algorithms, resulting in poor performance due to sensitivity to noise in the data when the number of variables is large (Banks, 2010).
- **Interpretability**: Multi-collinearity can reduce interpretability by making it difficult to understand the individual contribution of each independent variable to the dependent variable, hindering the understanding of relationships in the data and informed decision-making based on model predictions (Ayantola, 2020).

To address these issues, **one of the columns from the high-column pair (more than threshold of 0.8) is removed**. The decision to remove one of columns from the correlated pair is based on its relative importance, as explained here in the Colab notebook..

Although multi-collinearity cannot be completely avoided in real-world datasets, **one exception has been made for the columns *'pub_rec_bankruptcies'* and *'pub_rec'* (correlation: 0.81)**, as they provide important information about loan applicant's bankruptcy history and could be important information for financial institution regarding the risk of default based on past bankruptcy.

Below correlation heatmap, is generated **after removing the correlated columns (with one exception explained above)** for further analysis.

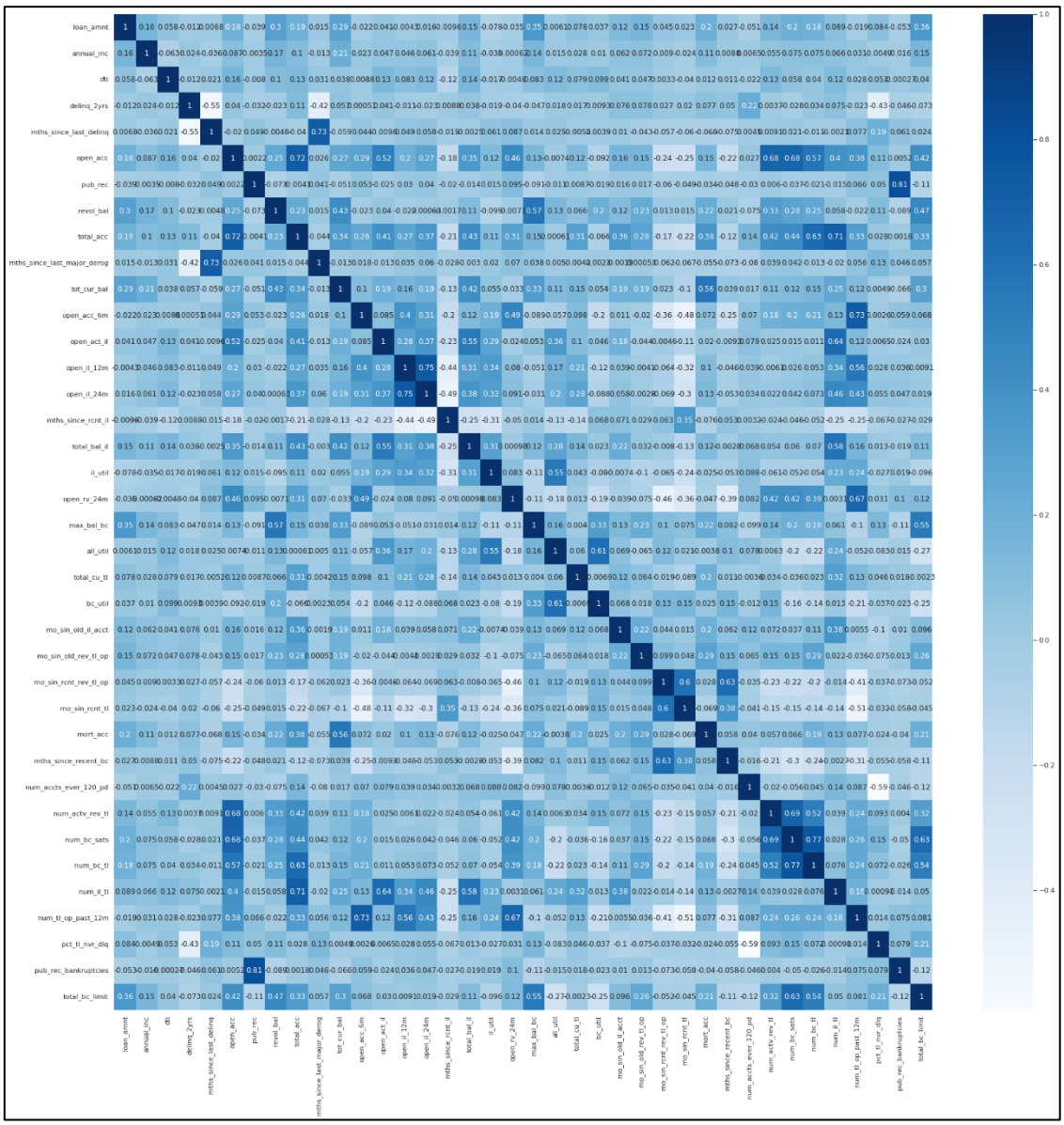

Figure 21: Correlation heatmap among all numerical columns after removing correlated numerical columns

As seen in the above correlation heatmap**, correlation coefficient between almost all of the numerical columns are now less than 0.8.** Refer appendix section to check the column names removed for handling multi-collinearity.

6. Remove the Sub-Grade column and retain only Grade column

The Sub-Grade column was removed, while the Grade column was retained because both are used to convey information about the creditworthiness and risk profile of a borrower in terms of ratings. However, Grade is a broader categorical rating assigned to a borrower based on their credit history, while Sub-Grade is a more specific rating within the Grade category (LendingClub, 2019). Refer appendix section to check the column names removed.

Feature Engineering & Feature Selection

(Data Pre-Processing)

This section covers the process of preparing and selecting relevant features for predicting loan default and using them in machine learning models, as well as identifying the factors that contribute to default. This is accomplished through **feature engineering, which involves techniques for converting raw data into features that can be effectively utilized by machine learning models for improved prediction accuracy** (Fuentes, 2018). **Feature selection involves selecting the features or columns in the dataset that are most useful for predicting default and transforming them for use in machine learning models** (Brownlee, 2020)

Below are the sequential steps taken to prepare the dataset for performing data preprocessing including feature selection and feature engineering:

1. Defining the target / dependent variable

The *'loan_status'* column in the dataset can be used as a target variable for predicting whether a borrower will default on their loan application. As mentioned before, there are several categories in the *'loan_status'* column, but only the categories 'Fully Paid', 'Charged Off', and 'Default' are relevant for this analysis. **Records with other categories like 'Current', 'Late (16-30) days' or '(31-120) days', 'In Grace Period' are not included, as these borrowers are still in the process of making loan payments and the final outcome of their loan payment has not yet been recorded**. Therefore, **only records with a final loan status outcome of 'Fully Paid', 'Charged Off', and 'Default' are selected for further analysis**.

Below pie-chart shows the proportion of 'Fully Paid', 'Charged Off', and 'Default' categories in *'loan_status'* column in percentage.

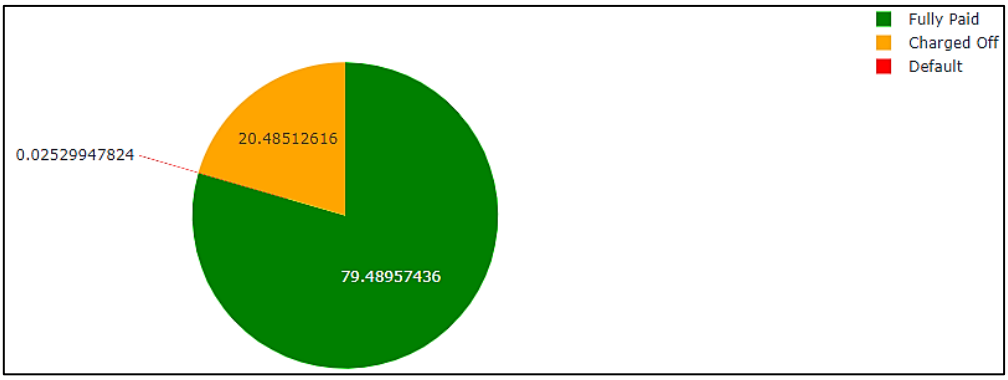

Figure 22: Proportion of 'Fully Paid', 'Charged-Off' and 'Default' Loans (Pie-Chart)

Finally, 'Charged Off' and 'Default' categories are encoded as target value **1**, **representing high risk loans** and 'Fully Paid' category is encoded as target value **0**, **representing low risk loans**. These binary values are stored in another column called *'HighRisk_LowRisk'*

2. Splitting data into 70% training, 10% validation and 20% testing data

After selecting the target variable and feature variables, the dataset is divided into three sets: a training set (461669 rows - 70%), a validation set (57709 rows - 10%), and a test set (115418 rows - 20%). Machine learning algorithms are used to train various models on the training set. The validation set is then used to evaluate, fine-tune and optimize these models. The model that performs best is selected and evaluated on the test set to ensure that it can generalize to new, unseen loan application records to predict their loan risk and probability of default.

It is important to note that the dataset is split using **stratified sampling**. (Klosterman, 2021) **This ensures that the proportions of both classes (high-risk loans and low-risk loans) are the same in the training, validation, and test sets, ensuring that each set is representative of the entire dataset.**

3. Convert data type of certain columns into numerical

In the dataset, there are some columns with object data types that should be numerical (such as integer or float). These columns are converted to numerical data types as explained below:

- The *'emp_length*' column, which originally contained values like '<1 year', '1 year', '2 year', etc., has been converted to a numerical data type by replacing these string values with the corresponding number of years as integers.

- The '*earliest_cr_line*' column contains a month and year in string format. It has been converted to a numerical format by calculating the number of months between **January 2022** and the date given in the column.

- The '*term*' column contains two unique values as strings: '36 months' and '60 months'. These values have been converted to a numerical column by replacing the string values with the corresponding number of months.

- The '*int_rate*' column contains interest rates in string format, with the '%' symbol. These values have been converted into a numeric column by removing the '%' symbol.

4. Feature Selection

In this project, **statistical feature selection techniques** are used to identify the subset of features that best explain the selected **target variable, i.e., loan risk**. These techniques involve using statistical tests such as **ANOVA (analysis of variance) F-test** (Brownlee, 2020)and **Chi-square test** (Brownlee, 2020) to evaluate the relationship between different features and the target variable.

(Brownlee, 2020) **ANOVA F-test is also used for feature selection in classification tasks with one numerical and one categorical feature to determine the significance of their relationship using p-value. A p-value less than 0.05 indicates a significant relationship**. (scikit-learn.org, 2022) The scikit-learn library has a function, **f_classif**(), in the **SelectKBest** class for performing ANOVA F-test feature selection. It can be used to select the top k features with significant relationships, as determined by p-values less than 0.05. Using this method, **41 numerical features were selected** as having a significant relationship with the binary target column *'HighRisk_LowRisk'*. See appendix section for more details.

(Brownlee, 2020) **The Chi-Squared test is used to check for significant relationship between two categorical variables. It can be used in feature selection to identify which categorical features are independent of the target variable, by comparing p-values to 0.05. Features with a p-value less than 0.05 are included in the dataset, others are removed.** (scikit-learn.org, 2022) The **SelectKBest** class in the scikit-learn library can be used for this, using the **chi2()** function to calculate p-values and select the top k features with the lowest p-values. Using this method, **7 categorical features were selected** as having a significant relationship with the binary target column *'HighRisk_LowRisk'*. See appendix section for more details.

5. Missing value imputation

The process of replacing missing values in a dataset with calculated values is called missing value imputation. It is commonly done by replacing missing numerical values with the mean or median of the column or replacing missing categorical values with the mode of the corresponding column. However, this can change the distribution of the data and potentially lead to inaccurate results. To avoid this, numerical features can be imputed using random sampling from the column to preserve the distribution of the data for more accurate analysis and predictive modeling. (Galli, 2021) The **RandomSampleImputer()** function in the **feature-engine** python library is used to replace missing values with random samples from the corresponding column, so that distribution of numerical before and after imputation is same for more accurate analysis.

6. Removing outliers

To identify and remove outliers in the dataset, **the z-score method is used**. This involves calculating the z-score of each value in the dataset, which represents the distance of the value from the mean in terms of standard deviations. Outliers are typically defined as values that are more than a certain number of standard deviations away from the mean, such as 3 or more standard deviations. **By setting a threshold of 3 standard deviations, rows in the dataset with z-scores greater than 3 can be identified and removed** (Shiffler, 1988) (Frost, 2019).

After removing the rows that contain outliers, there are **338807** remaining rows in the training dataset, with a total of **48** columns.

7. Encoding categorical columns

Categorical encoding is the process of converting categorical data, which consists of labels or categories, into numerical form so that it can be input to a machine learning model. One method for encoding categorical variables is **Weight of Evidence** (WOE) encoding, which **utilizes the proportion of both target labels (high risk and low risk loan) for encoding each unique category in a categorical column** (Galli, 2020). In formula below 'Target=1' represents high risk loan labels and 'Target=0' represent low risk loan labels:

$$\text{WOE} = \ln(\text{Proportion of Target} = 1 \,/\, \text{Proportion of Target} = 0).$$

A higher WOE value indicates a stronger relationship between the categorical variable and the target variable, while a lower WOE value indicates a weaker relationship (Galli, 2020).

Other techniques such as one-hot encoding or label encoding are often used to encode categorical columns, but they may not be as effective. One-hot encoding can increase the dimensions of the dataset by adding new columns for each category (Hastie, et al., 2009), while label encoding may not accurately reflect the relationship between the target variable and the categories in a categorical feature, potentially leading to inaccurate results in predictive modeling (Raschka & Mirjalili, 2015).

Therefore, the primary motivation for using Weight of Evidence (WOE) encoding in this project is that it maintains the relationship between the categorical columns and the target variable after encoding, based on the proportion of each binary target for each category in the categorical column. This can make it simpler to examine the factors that may lead to defaults from categorical columns. By better capturing the relationship between the predictor and target variables, WOE encoding can potentially improve the performance of the machine learning models.

8. Fixing the dataset imbalance

One of key issues in any real-world dataset is imbalanced dataset. an imbalanced dataset is one where the number of samples belonging to different classes is significantly unequal. For example, a dataset might have a large number of samples belonging to one class (the majority class) and only a few samples belonging to another class (the minority class). When a model is trained, a machine learning algorithm may become biased towards majority class while learning the pattern in classification task. Hence, it is crucial to balance the dataset. **In the training dataset, 79.49% consists of low-risk loan records and 20.51% consists of high-risk loan records.**

The technique used for balancing the dataset is under-sampling (Ferreira, et al., 2017), which is straightforward to implement given the dataset is large in size for both majority and minority class. Under-sampling is a technique for balancing a dataset that involves reducing the number of samples in the majority class. **The goal of under-sampling is to create a more balanced dataset by reducing the number of samples in the majority class to be more in line with the number of samples in the minority class**.

The bar graph and pie chart below show the size and proportions, in percentage, of low-risk and high-risk loan records (the majority and minority classes, respectively) **before and after under-sampling is performed**. The charts provide information on the size of the low-risk and high-risk loan records and how their proportions change after under-sampling.

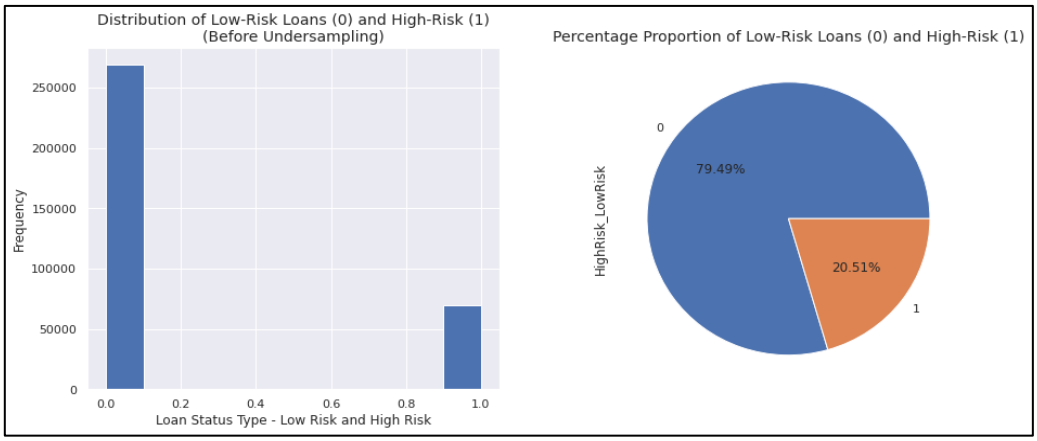

Figure 23: Before Under-Sampling - Distribution and Proportion of Low Risk (0) and High Risk (1) in training dataset

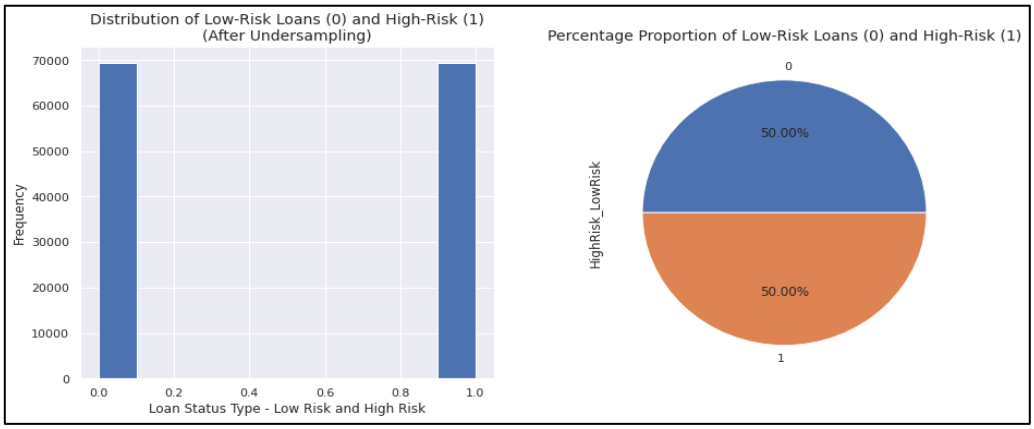

Figure 24: After Under-Sampling - Distribution and Proportion of Low Risk (0) and High Risk (1) in training dataset

Finally, there are **138982 records in training dataset containing 50% high-risk and 50% low-risk loan records for training** machine learning model and analysis.

9. Scaling Numerical Columns

To train classification models effectively using machine learning algorithms, numerical columns are recommended to be brought into common scale. Without scaling, columns with a large range of values (e.g., columns with values ranging from 0 to 1 million) can dominate the analysis and make it difficult to see patterns in columns with smaller ranges (e.g., columns with values ranging from 0 to 100). Scaling helps to equalize the influence of all the columns, which can lead to more accurate and meaningful results. (Brownlee, 2020)

For classification task, project uses **Robust Scaler**, as it is resistant to the influence of the outliers or extreme values in the dataset (Brownlee, 2020) (scikit-learn.org, 2022).

Classification Experiments and 3 Algorithms Selection Decision

(On Sample of Training Dataset)

[Code Section Link]

The goal of the classification experiments is to find the machine learning algorithms that are most effective at predicting high risk loans and probability of their default, by training multiple base models (models which are not optimized) on a sample of the training data and comparing their performance. The top 3 best performing models will be selected for training those models on full training dataset.

To compare the performance of different classification algorithms**, a subset of 40,000 records is selected from the training data**. (PyCaret, 2022)The **PyCaret library** is used to train multiple classification models on this subset and compare their performance. Optionally, the PyCaret library is also used to track classification experiments using the **MLflow** platform in google colab.

The KPI (**K**ey **P**erformance **I**ndicator or evaluation metrics) results of each classification model are presented in table below, with the top-performing model shown at the top.

Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
CatBoost Classifier	0.6525	0.7112	0.6630	0.6489	0.6559	0.3050	0.3050	15.7900
Ridge Classifier	0.6521	0.0000	0.6608	0.6491	0.6549	0.3041	0.3042	0.0700
Linear Discriminant Analysis	0.6521	0.7095	0.6609	0.6491	0.6549	0.3042	0.3042	0.3000
Gradient Boosting Classifier	0.6520	0.7101	0.6668	0.6472	0.6569	0.3040	0.3042	11.1500
Logistic Regression	0.6519	0.7095	0.6577	0.6498	0.6537	0.3039	0.3039	1.7500
Light Gradient Boosting Machine	0.6496	0.7071	0.6591	0.6464	0.6527	0.2992	0.2993	1.2867
Ada Boost Classifier	0.6460	0.7023	0.6568	0.6425	0.6495	0.2920	0.2920	2.4467
Random Forest Classifier	0.6445	0.6984	0.6469	0.6434	0.6452	0.2891	0.2891	7.3200
Extra Trees Classifier	0.6408	0.6960	0.6387	0.6409	0.6398	0.2815	0.2815	4.6133
Naive Bayes	0.6378	0.6892	0.6615	0.6312	0.6459	0.2756	0.2760	0.1300
SVM - Linear Kernel	0.6323	0.0000	0.7380	0.6095	0.6670	0.2647	0.2716	0.4900
Quadratic Discriminant Analysis	0.6315	0.6752	0.6352	0.6307	0.6325	0.2630	0.2634	0.1867
K Neighbors Classifier	0.5791	0.6073	0.5755	0.5792	0.5773	0.1581	0.1581	73.8400
Decision Tree Classifier	0.5641	0.5641	0.5655	0.5635	0.5645	0.1281	0.1281	0.7500
Dummy Classifier	0.5004	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1000

Figure 25: Performance Comparison of Base Classification Machine Learning models on sample of training dataset

The selection of three models based on classification experiment is done based on accuracy, AUC, recall and F1 scores (scikit-learn.org, 2013), which are defined as below.

- **Accuracy**: The proportion of correct predictions made by the model.
- **AUC-ROC**: The Receiver Operating Characteristic (ROC) curve is a measure of the trade-off between the true positive rate and the false positive rate. The area under this curve, known as the AUC-ROC, is used to evaluate the performance of a classifier. A value of 0.5 represents a random classifier while a value of 1.0 indicates a perfect classifier.
- **Recall**: The proportion of actual positive cases that are correctly predicted by the model. Recall is particularly important in situations where it is important to minimize false negatives (e.g., in loan default prediction).
- **F1-score**: The F1-score is the harmonic mean of precision and recall that combines precision and recall. It is often used as a single metric to compare the performance of different classification models, as it takes both precision and recall into account.

The **CatBoost classifier**, which is an algorithm for gradient boosting using decision trees (Yandex, 2017), appears to be a strong performer for predicting high risk loans based on its evaluation metrics. **It has an accuracy of 65.25%, an AUC of 0.711, a recall of 0.663, and an F1-score of 0.655**. These metrics indicate that the model has a good balance of precision and recall and can accurately classify a large proportion of the data.

Another optimal choice, based on time taken and simplicity to train the classification model, is **Logistic Regression**. **It has an accuracy of 65.19%, an AUC of 0.709, a recall of 0.657, and an F1-score of 0.6537 in predicting high risk loans.** (Stoltzfus, 2011)The logistic regression is a linear model and is often used in finance industry due its simplicity and interpretability.

Third potential choice of model could be **Random Forest**, which is an ensemble machine learning algorithm that uses bootstrap aggregation to train and average the predictions of multiple decision trees on different subsets of the data (Sarkar & Natarajan, 2019). **This model is selected to avoid overfitting the classification with a complex model like CatBoost, which has the potential to overfit the training dataset with a large number of columns. On a subset of the training data, Random Forest has achieved an accuracy of 64.45%, an AUC of 0.698, a recall of 0.647, and an F1-score of 0.645 in predicting high risk loans.**

In the next step, the three models that were selected in the previous step are trained and optimized on the entire training dataset. The model that performs the best in evaluation using validation dataset is then chosen as the final model for further evaluation and prediction on the test dataset.

Training and Optimizing selected Classification Models

(Using whole Training and Validation Dataset)

[Code Section Link]

Selected classification algorithms - Logistic Regression, Random Forest and CatBoost algorithms are trained on whole **70% training dataset** and their performance is compared and optimized using **10% validation dataset**.

Machine learning model are optimized using a process called **hyperparameter tuning** (Raschka & Mirjalili, 2019). Each machine learning algorithm has certain parameters that are not learned from the training data but are set prior to model training. Hyperparameter tuning involves selecting the optimal combination of these parameters from the parameter search space, which is the defined range of values for each parameter for a specific model. The combination of hyperparameter values that leads to the highest scores on evaluation metrics is selected as the optimal set of hyperparameters for the model. Hence, this process helps to improve the performance of the model and make it more robust.

The ROC-AUC score is selected as the evaluation metric for hyperparameter tuning (scikit-learn.org, 2013), therefore, the optimal combination of hyperparameters is determined using hyper-parameter tuning that gives the highest ROC-AUC score among all the other combinations.

A ROC curve is a plot that shows the relationship between the true positive rate and false positive rate **as the threshold for classifying a loan as high risk is varied**. The area under the ROC curve (AUC) is a single number that summarizes the performance of the classifier across all possible thresholds. A classifier that performs well will have a high AUC, which is equal to 1. (Bhandari, 2020)

The main reason for using the ROC-AUC score as the evaluation metric for hyperparameter tuning is that when predicting loan defaults, which are considered high-risk loans, it is important to have as many true positives (correctly predicted high-risk loans) as possible and as few false positives (incorrectly predicted high-risk loans) as possible. If the ROC-AUC score is high, it indicates that the trained model has a better ability to correctly predict high-risk loans. Also, evaluation using ROC-AUC score is insensitive to class imbalance (Fawcett, 2004)**.**

When tuning the hyperparameters of a model, precision or accuracy could also be considered as evaluation metrics to optimize the number of true positives and minimize false positives. But, these metrics are more effective only when the dataset is balanced, as it is not the case for validation and test sets, given that they are imbalanced while only the training set is balanced by under-sampling, these metrics are not the ideal choice for detecting and predicting high-risk loans in validation or unseen test dataset (Menardi & Torelli, 2014)**.**

While performing hyper-parameter tuning, **all three models are evaluated using stratified 5-fold cross validation technique** (Raschka & Mirjalili, 2019). In this technique, the data is first split into 5 equally sized folds. For each fold, the model is trained on the remaining 4 folds and validated on the current fold using given evaluation metrics. The performance of the model is then averaged across all 5 folds. This process is repeated 5 times, with each fold serving as the validation set once.

After performing hyper-parameter tuning for all three models using **random-search** (search using random selection of parameter combination) or **grid-search** (search using every possible parameter combination) algorithms, optimized values of each model parameter and effect on model is given below (scikit-learn.org, 2023) (scikit-learn.org, 2012) (Dorogush, et al., 2017):

Model Name	Hyper-Parameters	Optimized Value	Parameter Explaination
Logistic Regression	C	**1.5**	C parameter is inversely related and controls the strength of the regularization applied to the model
Random Forest	max_leaf_nodes	**500**	Controls the maximum number of leaf nodes in each tree, affecting the tree's bias and variance
	min_samples_split	**300**	Controls the minimum number of samples required to split an internal node in the tree, affecting the tree's depth and variance.
	max_depth	**15**	Controls the maximum depth of each tree in the forest, affecting the tree's variance and bias
	max_features	**20**	Controls the number of features considered when choosing the best split at each node, affecting the tree's bias and variance.
	n_estimators	**200**	Controls the number of trees in the forest, affecting the model's accuracy and variance.
	criterion	**gini**	Controls the function used to measure the quality of a split in the tree, affecting the tree's bias and variance
Catboost	learning_rate	**0.1**	Controls the step size at which the model adjusts the weights of the features to minimize the loss function
	depth	**10**	Controls the depth of each tree in the model.
	l2_leaf_reg	**9**	Controls the strength of the regularization applied to the model.
	iterations	**50**	Controls the number of iterations (or rounds) that the model will run to minimize the loss function.
	border_count	**64**	Controls the number of splits (or borders) that are created in the model.
	loss_function	**Logloss**	Controls the loss function that is used to evaluate the model's performance.

Figure 26: Hyperparameters, their optimized values for ROC-AUC and their explanations for each classification model after Hyperparameter Tuning

Evaluation Scores Comparison for Final Trained Model Selection

(On Training and Validation Dataset)

[Code Section Link]

To evaluate the optimized models, **key performance indicators (KPIs) on both the training and validation datasets are used with stratified 5-fold cross-validation. Stratified 5-fold cross-validation allows to assess the models' performance on an equal mix of high and low risk loans** (Raschka & Mirjalili, 2019). The chart below shows the average 5-fold cross-validated KPI scores for all three models on **both the training and validation datasets** for comparison:

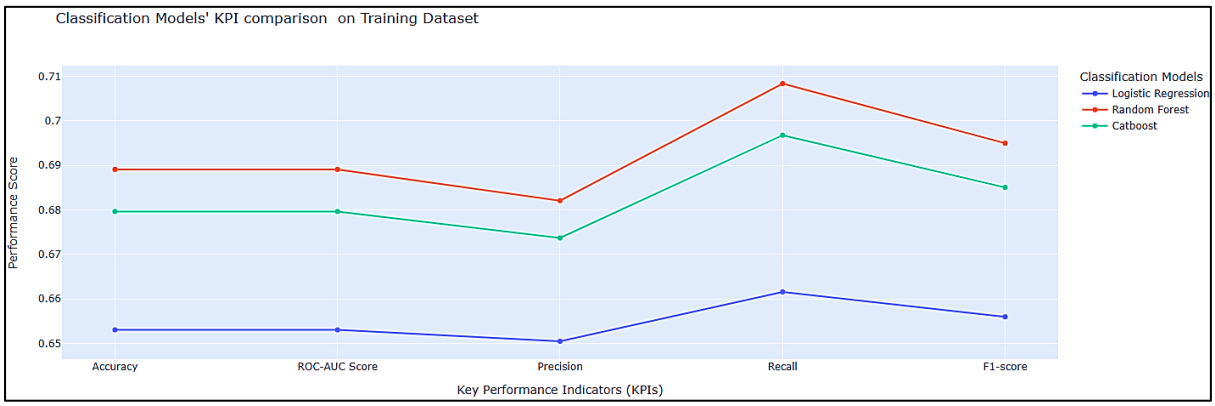

Figure 27: Performance score comparison score of three classification models on selected important Key Performance Indicators (KPI) for training dataset

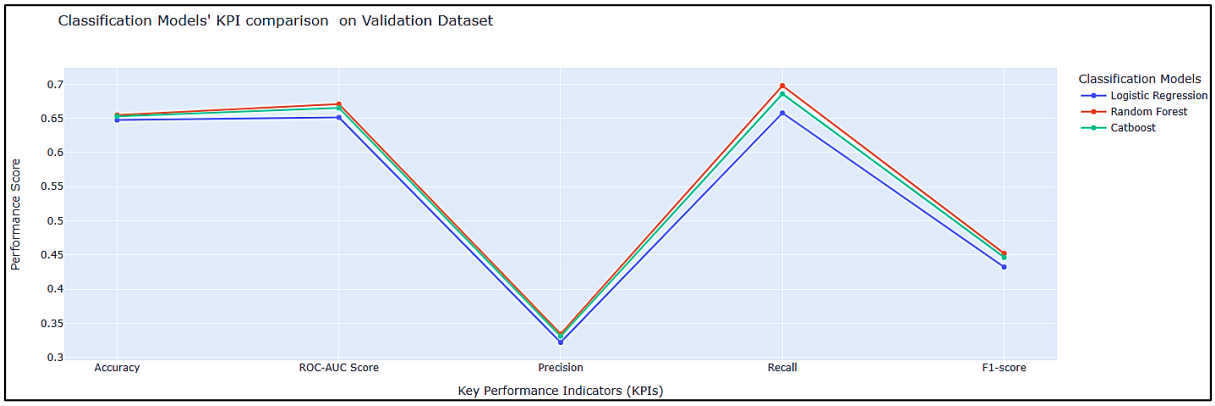

Figure 28: Performance score comparison score of three classification models on selected important Key Performance Indicators (KPI) for Validation dataset

Based on the comparison of KPI scores in above charts, **the random forest model performed the best** among all three models on both the training and validation datasets. Therefore, **random forest model is selected as the final classification model**.

Training and validation dataset's evaluation scores / KPI scores for all three models are:

- Random Forest

	Accuracy	**ROC-AUC**	**Recall**
Training Scores	68.90%	0.689	0.708
Validation Scores	65.48%	0.670	0.697

- CatBoost

	Accuracy	**ROC-AUC**	**Recall**
Training Scores	67.96%	0.679	0.696
Validation Scores	65.30%	0.665	0.686

- Logistic Regression

	Accuracy	**ROC-AUC**	**Recall**
Training Scores	65.30%	0.653	0.661
Validation Scores	64.75%	0.651	0.657

As per observation from above tables, **none of the above given models are suffering from overfitting** as the performance on training and validation dataset is almost similar. Also, evaluation metrics - **good accuracy, ROC-AUC and recall scores indicate good true positive predictions, which are most desirable in terms of predicting the loan risk** (Cooper, 2013). High ROC-AUC score of random forest model means that it is most effective in predicting high-risk loan when random forest's output probability is more than threshold probability of 0.5 and its high recall means that it is most effective in minimizing the predicting false negatives (i.e., predicting low risk loan which are actually high risk)

Finally, all three models' performance is compared using Area Under Curve (AUC) on training and validation dataset, which are given below. Among all three classification models, random **forest model has highest area under the curve (0.76 on training dataset and 0.73 on validation dataset) than Logistic Regression and CatBoost model, which means that random forest model did better job in classifying the high-risk loans on both training and validation datasets** (Bhandari, 2020)**.** (Brownlee, 2014).

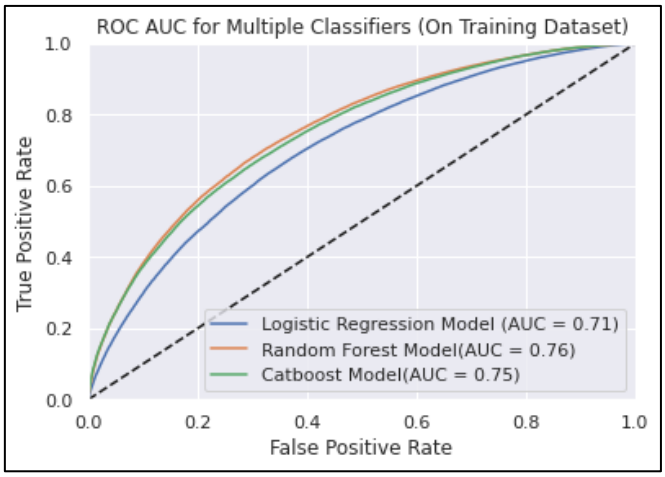

Figure 29: ROC-AUC Curve Comparison of selected classification models on Training Dataset

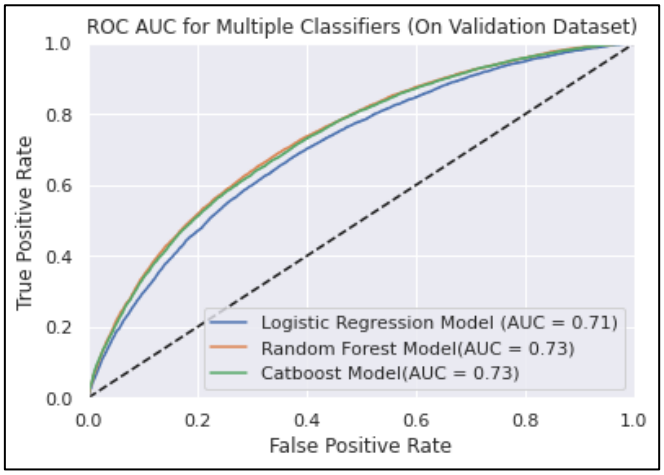

Figure 30: ROC-AUC Curve Comparison of selected classification models on Validation Dataset

According to the **feature importance scores (in descending order) produced by the random forest model**, **the most important loan attributes for financial institutions to consider are loan grade, interest rate, term, loan amount, debt-to-income ratio, and the total current balance of the borrower**.

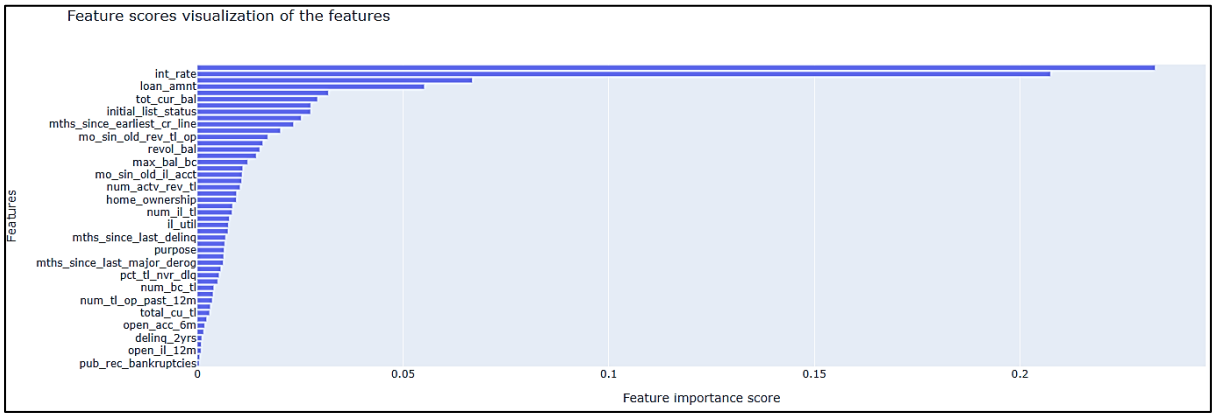

Figure 31: Feature Importance of Independent Variables for predicting Loan Risk using trained Random Forest model

Prediction and Classification Model Evaluation with Result Analysis

(On Test Dataset)

[Code Section Link]

Finally, predictions are done on remaining 20% test dataset (**115418 new loan applications**) using **finally selected random forest model**. Before performing predictions, the test dataset is pre-processed using steps and transformation same as done previously on training and validation dataset.

The classification report and confusion matrix given below shows the performance of the random forest model in predicting **low-risk loans (label 0)** and **high-risk loans (label 1)** on the unseen test dataset.

Classification Report

	precision	recall	f1-score	support
0	0.882196	0.640754	0.742337	91745.000000
1	0.324361	0.668399	0.436768	23673.000000
accuracy	0.646424	0.646424	0.646424	0.646424
macro avg	0.603279	0.654576	0.589552	115418.000000
weighted avg	0.767780	0.646424	0.679662	115418.000000

Confusion Matrix

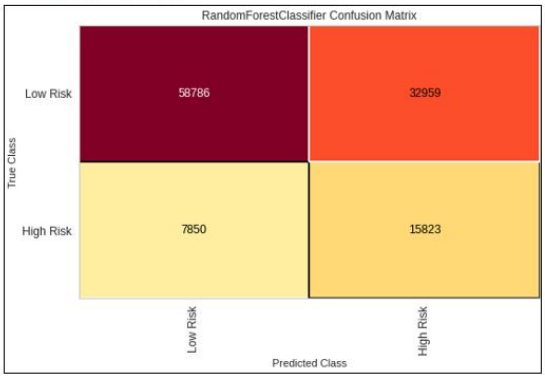

Figure 32: Classification Report and Confusion Matrix on Test Dataset using trained and optimized Random Forest Classification Model

In loan risk analysis, the primary objective for financial institutions is to accurately identify and predict as many high-risk loans (label 1) as possible while minimizing the number of incorrectly predicted low-risk loans (label 0) as much as possible. Therefore, **in addition to the accuracy of the trained model, it is crucial for the final model to have a high recall score** (Turiel & Aste, 2020). High recall indicates a low number of false negatives (missed high-risk loans) and a high number of true positives (correctly identified high-risk loans).

Therefore, expressing the result of random forest model in terms of accuracy and recall, it has achieved an **overall accuracy of 64.64% in predicting new loan applications as low-risk (label 0) or high-risk (label 1). Its recall score of 66.83% for high-risk (label 1), means that it correctly identified 15823 out of 23673 high-risk loans on test dataset as new loan application. It also correctly identified 58786 out of 91745 low-risk loans, giving it a recall score of 64.07% for low-risk loans (label 0).**

In terms of missed predictions, out of the 91745 low-risk loan applications, 32959 were incorrectly classified as high-risk (false positives). These cases can be manually investigated by financial institutions. **Out of all 23673 actual high-risk loan applications, only 7850 were predicted as low risk (false negatives)**, which could be a concern for financial institutions. However, **the loss incurred is low due to the low number of predicted false negatives from random forest model**, as these high-risk loan applications are likely to default but were wrongly predicted as low risk.

The AUC visualization, given below, on the test dataset shows that **the final random forest model is effective at distinguishing between low-risk and high-risk loans as it achieves 0.71 AUC**. This means that the trained random forest model performs significantly better in predicting and distinguishing low-risk and high-risk loans than a dummy model (indicated by diagonal black dotted line in below ROC-AUC visualization) that predicts the loan risk randomly (Brownlee, 2014) (Bhandari, 2020).

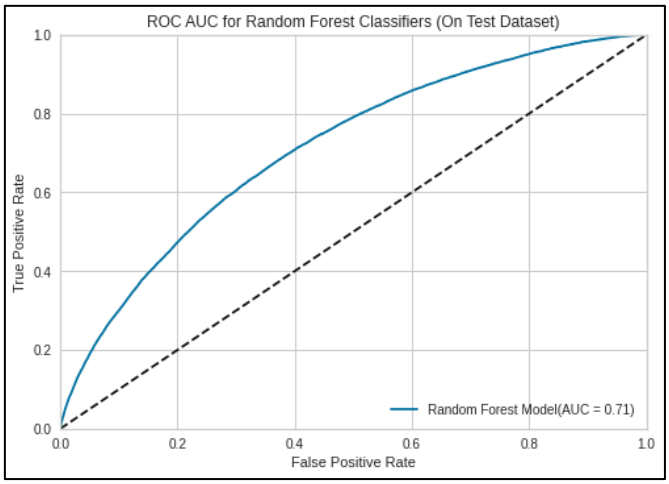

Figure 33: ROC-AUC Curve of Random Forest on Test Dataset

Prediction for new Loan Approval & its Probability of Default with Example and Interpretation

[Code Section Link]

Suppose an individual in Florida has applied for a loan of $14,000 to be paid back over a period of 36 months at an interest rate of 9.67%. The applicant has a B credit grade, has been employed for 8 years, and owns a mortgage. Applicant's annual income is $93,000 and intend to use the loan to pay off credit card debt. Applicant has no delinquencies on their record, 13 open accounts, a credit utilization rate of 39.9% and current total balance is $158,400. The financial institution reviewing the loan

application must assess its strength and risk level (low risk or high risk), as well as the applicant's probability of default (PD) if the loan is approved.

Therefore, the financial institution can utilize the same trained random forest model and same pre-processing steps to get the output from it in terms of probability, which is same as probability of default.

Using same trained random forest model and same steps for pre-processing the data from new loan application, it is found that probability of default for this loan application is 0.291 and since, this probability of default is less 0.5, it can be predicted as low-risk loan. Hence, Financial Institution can approve the loan.

Problem 2 – Prediction of Credit Conversion Factor: Regression Problem

The credit conversion (CCF) is a financial measure used by lenders to indirectly estimate the amount of money they may lose if a borrower defaults or declared as charged-off on their loan. It indirectly represents the fraction of the remaining balance on a loan that is expected to be withdrawn by the lender in the event of borrower's loan default (Tong, 2016). In other words, the CCF helps lenders estimate the amount of money lender may lose if a borrower defaults on their loan. The Basel II/III banking regulations have suggested using historical data to analyze the CCF (Valvonis, 2008).

Regression models are the established and effective approach to predict the credit conversion factor (Brown, 2011). Hence, By using regression models that consider a borrower's most relevant financial attributes, it is possible to predict the credit conversion factor (CCF) for that borrower after they have been declared as charged-off on their loan. This can help financial institutions estimate their potential losses using the CCF.

The value of CCF ranges from 0 to 1 (in terms of ratio) or depicted in percentage (Koulafetis, 2017)**. From the above given definition of CCF, if its value is 0, it means that all money was paid to lender on time and if its value is 1, then no money was paid back to lender till charged-off date.**

The code, experiments, and analysis for this regression problem statement, including detailed comments, are available in a Google Colab notebook and can be accessed using the following link:

[Credit Conversion Factor (CCF) Prediction Google Colab Link]

Data Preparation and Pre-Processing Steps for Regression

Below are steps taken to preparing the data for CCF regression modeling. **Each step is hyper-linked to its corresponding code section on Google Colab Notebook**.

1. Reading Data and Data preparation

The credit conversion factor (CCF) may be influenced by various financial attributes of borrowers after borrower is declared charged off such as, funded amount, annual income, credit history, the amount of principal recovered, credit revolving balance, debt to income ratio etc. Therefore, it is important to select the attributes that best reflect a borrower's financial situation at the time when they have been declared as charged-off on their loan. **In order to model the CCF, a sample of 118978 past loan records out of 1 million records are chosen for analysis by filtering the dataset, where the borrowers were considered to have charged-off and the debt was unlikely to be collected**

Below is the subset of columns, which are selected for the CCF regression model:

Selected Columns for modeling CCF using Regression Models	
Column Name	**Description**
grade	loan grade
total_rec_prncp	Principal received to date
home_ownership	home_ownership status of the borrower
verification_status	Indicates if income was verified by LC
purpose	Purpose of the loan request
initial_list_status	The initial listing status of the loan
term	The number of payments on the loan
emp_length	Employment length in years.
issue_d	The month which the loan was funded
earliest_cr_line	The month the borrowers earliest reported credit line was opened
funded_amnt	The total amount committed to that loan at that point in time.
int_rate	Interest Rate on the loan
installment	The monthly payment owed by the borrower if the loan originates.
annual_inc	The self-reported annual income
dti	debt to income ratio
delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrowers credit file for the past 2 years
mths_since_last_delinq	The number of months since the borrowers last delinquency
mths_since_last_record	The number of months since the last public record
open_acc	The number of open credit lines
pub_rec	Number of derogatory public records
total_acc	The total number of credit lines currently
acc_now_delinq	The number of accounts on which the borrower is now delinquent
total_rev_hi_lim	Total revolving high credit/credit limit
revol_bal	Total credit revolving balance
out_prncp	Remaining outstanding principal for total amount funded
recoveries	post charge off gross recovery

Figure 34: Select columns for modelling Credit Conversion Factor using regression models

2. Target variable definition and Data pre-processing (feature engineering)

Below are the steps followed for **defining dependent variable** for modeling 'credit conversion factor' (CCF) and for pre-processing the borrowers' financial attributes after they are declared charged off by the lender.

- ### Create new CCF column as dependent column for regression

  As per the definition of CCF as ratio or percentage of outstanding amount (Qi, 2009), a new column called CCF is calculated by using the funded amount and recovered principal amount received from the borrower until the charge-off date. The formula for estimating CCF is shown below:

  CCF = (funded amount – principal received from the borrower) / funded amount

  The distribution of CCF is given below and its value varies between 0 and 1, with average value around 0.73 ± 0.20. The distribution is left-skewed, indicating that most of the time, lender expects to recover around 73% of the funded amount in cases when the borrower becomes unlikely to pay back the loan.

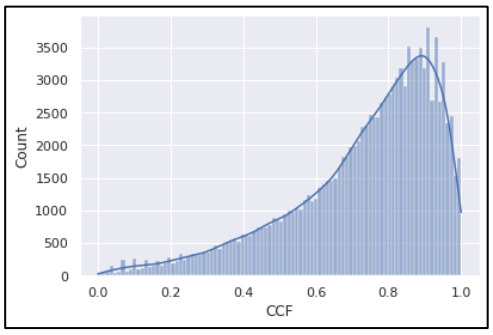

Figure 35: Distribution of Credit Conversion Factor (CCF) in dataset

- ### Splitting dataset into almost 70% training, 10% validation and 20% testing dataset

  The dataset is split into three parts: 70% for training (95182 rows), 10% for validation (11898 rows), and 20% for testing (23796 rows). The training data is used to train machine learning models for regression, specifically for modelling CCF. The performance of three trained regression models is optimized and evaluated using the validation data and regression evaluation metrics. The model that performs the best on the validation data is chosen for final evaluation and prediction using the remaining 20% test data.

- Data type conversion to numerical format

  For a regression machine learning model to produce accurate results, certain object datatype columns, including employment length, interest rates, loan issued date, delinquencies in the past two years, and months since the last delinquency etc., are converted into numerical data types.

- Missing value imputation

  The missing values are replaced with random samples from the respective numerical columns, so that distribution of before and after imputation does not change significantly. Hence, The **RandomSampleImputer()** function in the **feature-engine** python library is used for this purpose (Galli, 2021).

- Encoding categorical columns

  The categorical columns are encoded as numerical data by creating new binary columns for each unique category in a column. This technique is known as **one-hot encoding** (Raschka & Mirjalili, 2019). For example, categorical '*purpose*' column has been encoded using one-hot encoding by creating separate binary columns (like, '*purpose_medical*', '*purpose_education*' etc) for each purpose category. So, any row in '*purpose*' column has '*medical*', it would have "1" in '*purpose_medical*' column and "0" in '*purpose_education*' and other purpose-related column.

- Scaling numerical columns

  In order to effectively train a regression model, it is often necessary to scale numerical columns with different units to a common scale and therefore could potentially help to improve the model's performance. This process, called standardization, involves rescaling the distribution of values so that the mean of the observed values is 0 and the standard deviation is 1. Hence, The function **StandardScaler** in the scikit-learn library is used to standardize numerical columns in data. (Brownlee, 2020) (scikit-learn.org, 2023).

- Final check with training dataset

Before starting the regression modelling experimentation, it is important to check the overall training dataset after pre-processing and to ensure that there are no missing values, and all columns are numerical datatype format.

Total training dataset finally has **95182 rows with 53 independent variables** to model and predict CCF, with no missing values in any columns related to borrowers' financial situation after charge-off.

The top five rows of the training dataset after pre-processing are shown below:

	total_rec_prncp	term	emp_length	funded_amnt	int_rate	installment	annual_inc	dti	delinq_2yrs	mths_since_last_delinq	...	purpose_medical	purpose_major_purchase
113539	-0.333092	-0.786496	1.261438	-0.573261	0.315375	-0.360977	0.036747	-0.916607	-0.332692	0.969509	...	0	0
218306	0.602348	1.271462	1.261438	0.820601	-0.016694	0.347537	0.214183	-0.461222	-0.332692	1.476510	...	0	0
326173	-0.699958	1.271462	1.261438	-0.513338	0.410252	-0.705571	-0.015929	-0.415515	-0.332692	-0.182764	...	0	0
383280	0.164241	-0.786496	1.261438	-0.680080	0.778373	-0.449789	0.277949	-0.197697	-0.332692	-0.136673	...	0	0
770949	-0.925054	-0.786496	-0.533844	-0.096482	-1.103981	0.027171	-0.004840	-0.944257	0.888248	-0.735855	...	0	0

5 rows × 53 columns

Figure 36: Training Dataset after performing pre-processing steps for training regression model

---

Regression Experiments and 3 Algorithms Selection Decision

(On Sample of Training Dataset)

[Code Section Link]

To identify which regression models might be suitable for predicting the credit conversion factor (CCF) of loans, an experiment is performed using portion of training dataset, i.e., 40,000 past loan records and the PyCaret auto-ml library in Python (PyCaret, 2022). The experiment trained a range of base regression models (which are not optimized yet) to determine which regression model had the best fit to the data in terms of the R2 score. **The R2 score is a measure of how well the regression model explains the variance in the dependent variable, with a higher score indicating a better fit. A model with an R2 score of 1 perfectly explains the variance, while a model with an R2 score of 0 does not explain any of the variance** (Frost, 2018). The results of the experiment are shown in tabular form, with the top-performing models listed based on their R2 scores:

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
Random Forest Regressor	0.0021	0.000000e+00	6.000000e-03	9.991000e-01	0.0049	0.0554	36.014
Extra Trees Regressor	0.0026	1.000000e-04	7.700000e-03	9.985000e-01	0.0064	0.0968	24.876
CatBoost Regressor	0.0039	1.000000e-04	8.000000e-03	9.984000e-01	0.0062	0.0911	13.364
Light Gradient Boosting Machine	0.0050	1.000000e-04	9.500000e-03	9.978000e-01	0.0073	0.1119	0.738
Decision Tree Regressor	0.0046	1.000000e-04	1.150000e-02	9.967000e-01	0.0089	0.0539	0.594
Gradient Boosting Regressor	0.0117	4.000000e-04	1.880000e-02	9.914000e-01	0.0141	0.1582	9.513
AdaBoost Regressor	0.0893	1.010000e-02	1.003000e-01	7.542000e-01	0.0597	0.4263	6.261
Bayesian Ridge	0.0699	1.010000e-02	1.005000e-01	7.532000e-01	0.0650	0.6080	0.157
Ridge Regression	0.0699	1.010000e-02	1.005000e-01	7.531000e-01	0.0650	0.6083	0.039
Linear Regression	0.0699	1.010000e-02	1.005000e-01	7.531000e-01	0.0650	0.6083	0.551
Orthogonal Matching Pursuit	0.0720	1.040000e-02	1.021000e-01	7.453000e-01	0.0659	0.5867	0.041
Passive Aggressive Regressor	0.0889	1.370000e-02	1.171000e-01	6.642000e-01	0.0726	0.6326	0.096
K Neighbors Regressor	0.0937	1.640000e-02	1.281000e-01	5.988000e-01	0.0808	0.6049	14.586
Lasso Regression	0.1605	4.100000e-02	2.024000e-01	-8.000000e-04	0.1276	1.1420	0.040
Elastic Net	0.1605	4.100000e-02	2.024000e-01	-8.000000e-04	0.1276	1.1420	0.038
Lasso Least Angle Regression	0.1605	4.100000e-02	2.024000e-01	-8.000000e-04	0.1276	1.1420	0.039
Dummy Regressor	0.1605	4.100000e-02	2.024000e-01	-8.000000e-04	0.1276	1.1420	0.027
Least Angle Regression	121210.5160	1.271244e+14	3.565451e+06	-3.131963e+15	1.3860	211649.7662	0.048

Figure 37: Performance Comparison of Base Regression Machine Learning models on sample of training dataset

From the performance comparison from the above experiment, 3 regression models are selected based on two following factors:

- It is important for financial institutions to choose a regression model that is both interpretable and efficient, as this can help them to understand the variables that are most important for predicting the credit conversion factor (CCF) and make accurate predictions on large datasets without overfitting. Complex models may be less interpretable and take longer to train, making them more susceptible to overfitting. Therefore, it is important to select a model that is simple enough to be interpretable, yet able to explain a good number of the variables that contribute to predicting the CCF, and that can quickly make predictions on large numbers of loan records without overfitting.
- The regression model should be able to accurately explain the target variable using predicted values, as measured by the R2 score. While a high R2 score is generally desirable, a model with a very high R2 score (~0.99) may not always be the best choice. This is because a model with a very high R2 score may not generalize well to new, unseen data, or may be too complex and prone to overfitting. Therefore, it is generally preferred to choose a regression model with a decent R2 score that is able to accurately explain the target variable while also avoiding issues such as overfitting or complexity.

Based on above given reasoning, below three regression models are selected from the experiment for further regression model training and validation:

- **Decision Tree Regressor**: It is a tree-based model utilized for regression that constructs a tree-like structure to predict a continuous target variable by utilizing the features of the data, with internal nodes representing decisions based on feature values and leaf nodes representing predictions(Gordon, et al., 1984). **This model was selected due to its high R2 value of 0.967 on a sample of the training data**, **as well as its interpretability and potential to accurately predict the credit conversion factor (CCF) between 0 and 1**.
- **AdaBoost Regressor**: It is an ensemble technique that merges the predictions of multiple weak regression models to form a more robust and accurate model. It trains a sequence of weak models and generates a final prediction by taking into account the predictions of the individual models (Freund & Schapire, 1997). **Despite achieving a lower R2 score of 0.754, this model was considered as a potential candidate due to its potential to generalize well to predict the CCF between 0 and 1.**
- **Linear Regressor**: It is a linear model used for regression tasks. It fits a linear equation to the data to predict a continuous target variable (Jobson, 1995). **This model was chosen as a potential candidate due to its high interpretability, short training time, and albeit R2 value of 0.753 on a sample of the training dataset.**

Training and Optimizing selected Regression Models

(Using whole Training and Validation Dataset)

[Code Section Link]

The regression models were trained using 70% of the available data as the training dataset and evaluated using 10% of the data as the validation dataset. **The objective is to optimize the models' hyperparameters to minimize the mean squared error (MSE) by maximizing the negative MSE as regression scoring metric** (scikit-learn.org, 2013). To achieve this, 3-fold cross-validation was used to assess the models' performance and tune the hyperparameters. The optimized values for the hyperparameters of the three regression models were determined using either random-search or grid-search algorithms.

The results of the hyperparameter tuning process, including the parameter combinations that resulted in the minimum MSE and the impact of the parameters on the model's performance, are presented below.

Model Name	Hyper-Parameters	Optimized Value	Parameter Explaination
Linear Regressor	fit_intercept	**TRUE**	A boolean parameter that determines whether to include an intercept term in the linear regression model
	normalize	**FALSE**	A boolean parameter that determines whether to normalize the input features before fitting the model.
AdaBoost Regressor	n_estimators	**50**	Controls the number of weak learners (e.g., decision trees) used in the boosting process.
	learning_rate	**0.1**	Controls the impact of each weak learner on the final model.
	loss	**linear**	Controls the loss function used to handle errors made by the weak learners. This loss function determines how the model updates itself in response to these errors.
Decision Tree Regressor	max_depth	**30**	Controls the maximum depth of the decision tree
	min_samples_leaf	**2**	Controls the minimum number of samples that can be in a leaf node. This can be used to prevent overfitting by requiring that a leaf node have a minimum number of observations.
	min_samples_split	**10**	Controls the minimum number of samples required to split an internal node. This can be used to prevent overfitting by requiring that a node have a minimum number of observations before it can be split
	max_features	**auto**	Controls the maximum number of features considered when splitting a node. This can be used to prevent overfitting by limiting the number of features that the model can use to make decisions.
	max_leaf_nodes	**40**	Controls the maximum number of leaf nodes in the decision tree. This can be used to prevent overfitting by limiting the number of leaf nodes and thereby limiting the model's complexity.

Figure 38: Hyperparameters, their optimized values and their explanation for each regression model after Hyperparameter Tuning

---

Evaluation Scores Comparison for Final Trained Model Selection

(On Training and Validation Dataset)

[Code Section Link]

The performance of the trained and optimized versions of the three regression models is evaluated using both the training and validation datasets. **3-fold cross-validation is utilized in the evaluation process, along with several evaluation visualizations for regression including residual plots, mean squared error heatmaps, and feature importance graphs**. These regression evaluation scores were used to select the best regression model for predicting the CCF on the remaining 20% of the test dataset in the next section.

Three regression models are evaluated using metrics and visually for training and validation dataset as given below:

1. Average 3-fold cross validation score for R2 and MSE:
    - Linear Regression

	R2 score	**MSE**
Training Dataset	0.746 +/- 0.008	0.010 +/- 0.000
Validation Dataset	0.746 +/- 0.006	0.010 +/- 0.000

- AdaBoost Regression

	R2 score	MSE
Training Dataset	0.817 +/- 0.004	0.007 +/- 0.000
Validation Dataset	0.829 +/- 0.007	0.007 +/- 0.000

- Decision Tree Regression

	R2 score	MSE
Training Dataset	0.923 +/- 0.002	0.003 +/- 0.000
Validation Dataset	0.918 +/- 0.005	0.003 +/- 0.000

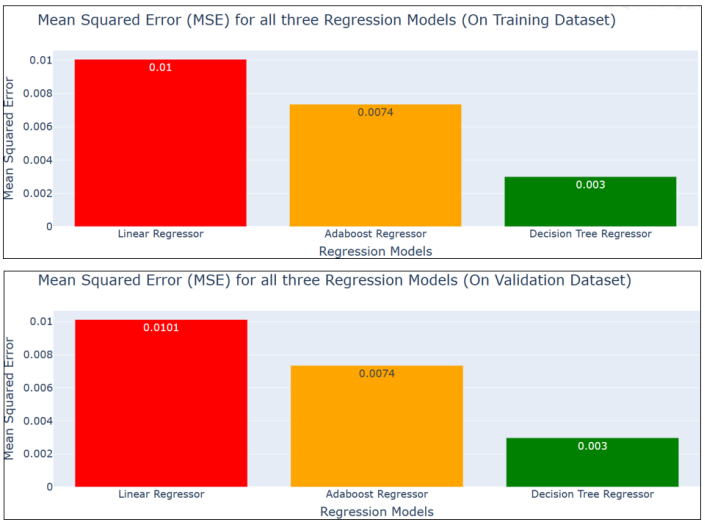

Figure 39: MSE comparison of three regression models on training and validation dataset

Mean squared error (MSE) (Raschka & Mirjalili, 2019)is a measure of prediction error in regression models. It **is calculated as the average of the squared differences between the predicted and true values. MSE is more sensitive to large errors, making it suitable for identifying and reducing large errors in tasks such as predicting CCF, a continuous variable between 0 and 1**. The MSE scores for linear regression, decision tree regression, and AdaBoost regression models on training and validation datasets suggest none of the models are overfitting. The linear regression model had the highest MSE score of 0.01, while the decision tree regression model had the lowest score of 0.003 and the AdaBoost regression model had an intermediate score of 0.0074.

2. Residual Plot: A residual plot is a graphical representation of the prediction errors, or residuals, of a regression model (Raschka & Mirjalili, 2019). **It is used to assess the variance of the residuals and to determine if the errors are randomly distributed according to a normal distribution.** A plot with residuals close to zero and minimal variance suggests a strong regression model, while a plot with greater variance and a larger spread of residuals indicates a weaker model. **The normal**

distribution of the residuals is also an important consideration, as it indicates that the errors are random and not systematic. In an ideal scenario, the residuals should be normally distributed with a mean of zero.

Given below are the residual plots for selected 3 regression models.

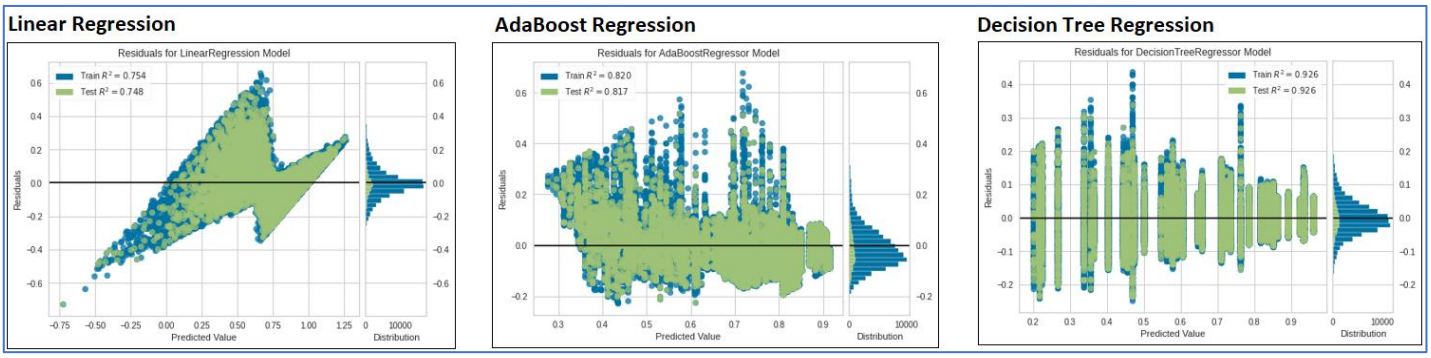

Figure 40: Residual Plot for all trained and optimized three regression models for comparison of residual variance

In linear regression, the residual variance is high because many residuals are spread out, as shown by the thin bell curve of the normal distribution. In contrast, in **AdaBoost and decision tree regression, most of the residuals are centred around y=0**, as indicated by the thick bell curve of the residual normal distribution. **Hence, AdaBoost and decision tree regression models are much better in predicting the CCF in range 0 and 1**.

3. Feature Importance plot: Feature importance is a measure of how each feature in a machine learning model affects the predictions made by the model. It can help to identify the most important features in the dataset and understand how they influence the model's predictions. The bar graphs below show the influence of the top 10 independent variables on the prediction of CCF from the relevant regression models.

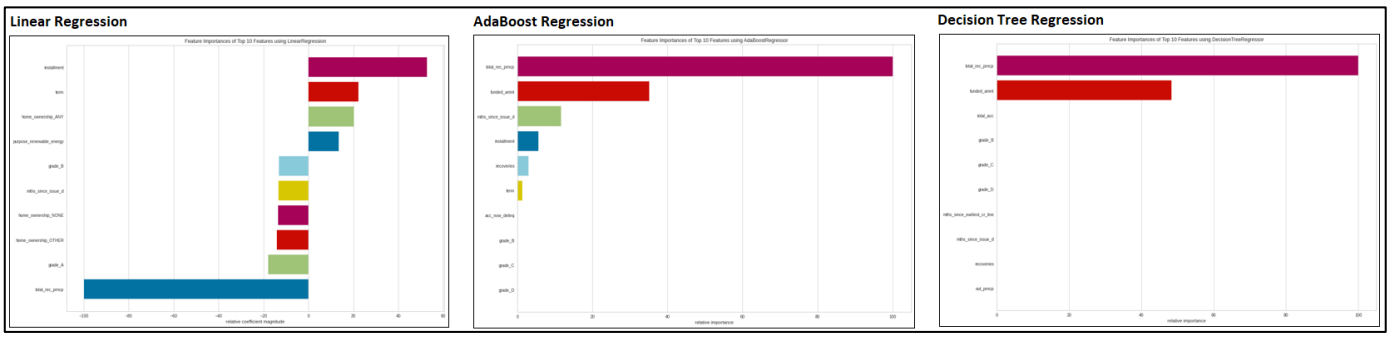

Figure 41: Important contributing variables in prediction of credit conversion factor for respective regression models

The feature importance graph shows that the prediction of CCF using linear regression is influenced by multiple independent variables, including loan instalments, term, home ownership status, loan grade, and principal recovered. **In contrast, the prediction of CCF using a decision tree model is only influenced by two variables: total recovered principal and funded. This is a significant limitation of the decision tree regression model, as CCF is likely to be influenced by other loan features as well. This suggests that the decision tree model may be prone to overfitting on an unseen dataset, as it may be too closely tied to the specific characteristics of the training data.**

However, AdaBoost regression model's CCF prediction is influenced by enough attributes such as total principal recovered till charged-off date, funded amount, months since amount was funded, instalments, recoveries, and loan term. Therefore, in terms of importance of features for predicting CCF, either linear regression model or AdaBoost regression model are better choice in comparison with Decision tree regression model.

One limitation of linear regression for modelling CCF is that it can produce output values that are beyond the range of the training data. This is because linear regression models use a linear equation to make predictions, and the output of this equation can range from negative infinity to positive infinity. In contrast, **tree-based models such as AdaBoost and decision tree regression can produce output values that are limited to a specific range or domain**. For example, if the training data for CCF consists of values between 0 and 1, these tree-based models will also produce output values that fall within this range. However, **decision tree regression model will be not used further because, as per the feature importance graph above, the model's output is only influenced mainly by two independent variables, which may not be sufficient to accurately predict the CCF on unseen dataset**.

Based on the analysis described above, **the AdaBoost regression model is chosen to make predictions on the unseen test dataset for CCF**.

Prediction and Regression Model Evaluation with Result Analysis

(On Test Dataset)

[Code Section Link]

CCF predictions are done on remaining 20% test dataset using **finally selected AdaBoost model**. Before performing predictions, the test dataset is pre-processed using steps and transformation same as done previously on training and validation dataset.

The AdaBoost Regression model was utilized to predict unseen test data, **achieving an R2 score of 0.82 and a mean squared error (MSE) of 0.01**. The residual plot on the test dataset suggests that most of the residuals are centered around y=0, as evidenced by the thicker bell curve of its normal distribution around its mean equal to 0.

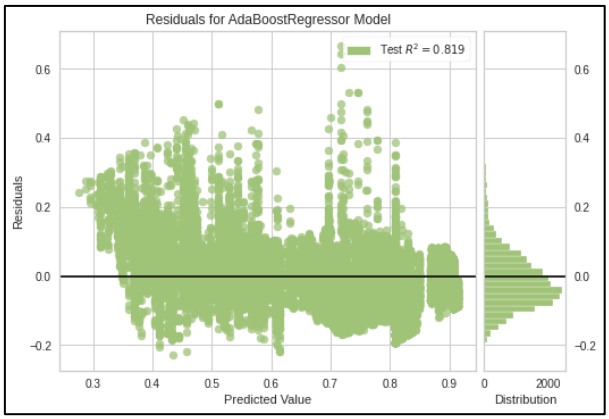

Figure 42: Residual Plot on Test Data using trained and Optimized AdaBoost Regression model

In conclusion, The AdaBoost Regression model performs exceptionally well on the test dataset, as indicated by its high R2 score of 0.82 and low MSE of 0.01. Additionally, the normal distribution of residuals around 0 on the test dataset demonstrates that the model is able to accurately fit the data and that the errors made by the AdaBoost Regression model are randomly distributed. These results suggest that the trained AdaBoost Regression model is highly effective at predicting credit conversion factors between 0 and 1.

Prediction for CCF on a new charged-off Loan with Example and Interpretation

[Code Section Link]

Suppose there was a loan that was issued in September 2017 for an amount of 13000 USD. The interest rate on this loan was 9.93% and the borrower was required to make monthly payments of 419.05 USD. Unfortunately, the loan ended up being charged-off, meaning that the borrower was unable to pay it back. Despite this, the lender was able to recover some of the money, specifically 10061 USD. Of this recovered amount, 625.53 USD was the principal that was originally borrowed. The loan had a term of 36 months, or 3 years.

Using the same trained AdaBoost regression model and same pre-processing steps on the data from this charged-off loan record, the Credit Conversion Factor (CCF) is predicted to be 0.907. It means lender is expected to recover 90.7% of funded amount. It indicates that lender is exposed to loss of 11792.261 USD (13000*0.907)

Problem 3 – Unsupervised Machine Learning Approach to Grouping Loans by Default Risk: Clustering Problem

Financial institutions may use past loan records to segment borrowers into clusters or groups in order to determine the risk and probability of default associated with each group. This process involves using unsupervised machine learning algorithms and probability concepts on past records that include loan attributes and borrower financial information. By analysing the cluster that a new loan applicant belongs to, a financial institution can determine the risk and probability of default for that applicant.

This approach is interesting because it allows financial institutions to assess the risk associated with new loan applicants using clustered groups of previous loans and risk associated with it. By analysing the cluster to which a new loan application belongs to, the financial institution can estimate the probability of default for that application. This information can then therefore be used to make informed lending decisions.

The code, experiments, and analysis for this clustering problem statement, including detailed comments, are available in a Google Colab notebook and can be accessed using the following link:

[Loan Application Clustering based on Risk Google Colab Link]

Data Preparation and Pre-Processing

1. Read Loan Dataset and Select Relevant Rows & Columns

For preparing the dataset, the first 1 million previous loan records from Lending Club dataset are considered. **The loan records are filtered based on the loan status i.e., "Full Paid", "Charged Off" and "Defaulted". A new column called "*HighRisk_LowRisk*" is created, with a label of 0 assigned to "Full Paid" records (indicating low risk) and a label of 1 assigned to "Charged Off" and "Defaulted" records (indicating high risk).**

Column related to loan amount and other attributes indicating the loan applicant's financial situation are selected for clustering. The list of columns and their descriptions are given below:

Column Name	Description
loan_amnt	Loan Amount requested by loan applicant
emp_length	Employment Length of the loan applicant
avg_fico_range	Average FICO score of the loan applicant
int_rate	Interest Rate on the loan
annual_inc	Annual Income of the loan applicant
dti	Debt to Income ratio of the loan applicant
open_acc	Number of Open accounts (having a high number of open accounts can be seen as a sign of financial responsibility and may indicate a lower credit risk)
revol_bal	Revolving Balance (A high revolving balance can be a sign of financial stress and may increase credit risk)
HighRisk_LowRisk	Column indicating loan status if the loan was high-risk (defaulted) or low-risk (non-defaulted)

Figure 43: Important selected variables for creating cluster based on Loan Risk

Also, any row containing any missing values are also removed leaving the overall 539721 loan records with 9 columns for pre-processing and clustering.

The *'HighRisk_LowRisk'* column is only used to interpret the clusters in terms of probability at last and is not used for creating clusters using unsupervised learning algorithms

2. Data Pre-Processing

The employment length and interest rate columns are converted to numerical data types. Outliers in these columns are identified by calculating the z-score of each value and removing rows where the z-score exceeds a threshold of 3 (Frost, 2019). **This step is necessary for the use of unsupervised machine learning algorithms, as they are sensitive to outliers and require numerical input data**.

A subset of the clustering dataset is chosen, comprising randomly sampled 15000 rows of high-risk loans and 15000 rows of low-risk loans, totalling 30000 rows. This is done for the following reasons:

- Unsupervised algorithms are computationally expensive because they require the calculation of a distance matrix that includes the distance between data points. A large number of rows in the dataset could lead to the use of all available RAM memory, which could cause issues.
- To ensure that the probability assigned to each cluster is fair and rational, it is important that cluster groups are created from the equal number of high-risk and low-risk loans within clustering dataset.

Finding Optimal Number of Clusters

[Code Section Link]

In order to determine the optimal number of clusters for a given clustering dataset (Nwanganga & Chapple, 2020), the "within-cluster sum of squares" (WCSS) metric is often used. **WCSS measures the compactness of the clusters and is calculated as the sum of the squared distances between the points in a cluster and the centroid (mean) of the cluster. The lower the WCSS score, the more compact and cohesive the cluster is.**

The determination of the optimal number of clusters can be visualized using an elbow curve, which plots the WCSS or distortion score on the y-axis and the number of clusters on the x-axis (Shi, et al., 2021). As the number of clusters increases, the WCSS score decreases, but at a diminishing rate. **The optimal number of clusters is typically chosen at the point where the decrease in the WCSS score levels off, which is represented by an "elbow" in the plot** (Sugar & James, 2003).

In the elbow curve given below, which was generated using the 'YellowBrick' Python library (scikit-yb.org, 2019), **the elbow is located at 4 clusters, as indicated by the vertical black line**. This suggests that the optimal number of clusters for the given dataset is 4, as the WCSS score decreases slowly after this point on the x-axis.

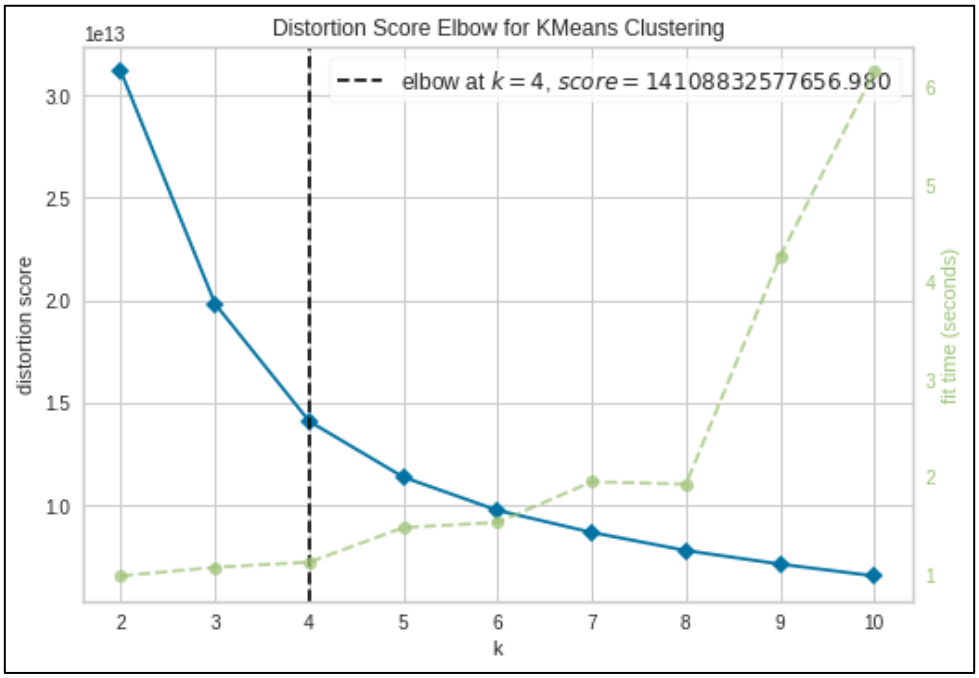

Figure 44: Elbow Curve showing optimal number of Clusters using K-Means Algorithm. Optimal Number of Cluster is found using elbow location at k=4

Hence, the number of optimal clusters for the given clustering dataset is 4.

Creating and Visualizing Clusters using K-Means Algorithm

[Code Section Link]

K-means is an unsupervised algorithm that is used to group data points into clusters. It begins by randomly selecting K points in the data called "centroids", then it assigns each data point to the nearest centroid based on a distance measure such as Euclidean distance. The centroids are then moved to the average of the points that are assigned to them. The process of reassigning the points to the nearest centroid and moving the centroids to the average of the points assigned to them is repeated until the assignment of points to centroids no longer changes. At this point, the algorithm has "converged" and the centroids represent the centre of the points in their respective clusters. (Patel, 2019)

For clustering the previous loans, K-means algorithm is used with following parameters (scikit-learn.org, 2022):

- ***max_iter=1000***: It specifies that the k-means clustering algorithm will run for a maximum of 1000 iterations or until the assignment of data points to centroids no longer changes and the centroids stop moving, whichever comes first.
- ***Init='k-means++'***: It is the improved method for selecting the initial centroids to speed up the convergence and improve the quality of clusters produced by K-means algorithm (Arthur & Vassilvitskii, 2007).
- ***n_clusters=4***: It specifies the number of clusters to be created using K-means algorithm. Hence, **it is given value as optimal number of clusters** found earlier using elbow curve.

After 4 clusters are created using K-means algorithm, a cluster evaluation metric called **silhouette score** is used to evaluate the quality of clusters created (Raschka & Mirjalili, 2019). The silhouette score is **a measure of how well each data point is assigned to its respective cluster in a clustering algorithm**. It is calculated by taking the mean of the silhouette coefficient for all the data points in the dataset. **The silhouette coefficient** for an individual data point is calculated by comparing the mean distance to other points in the same cluster ('a') with the mean distance to points in the next closest cluster ('b'). The silhouette coefficient is then calculated as follows:

$$(b - a) / \max(a, b)$$

where, 'a' also refers to cohesion or intra-cluster distance and 'b' also refers to separation between two clusters

The silhouette coefficient can range from -1 to 1, with values close to 1 indicating a strong assignment to the correct cluster, values close to 0 indicating a weak assignment to the cluster, and values close to -1 indicating a misassignment to the cluster.

The average **silhouette score** for all 4 clusters using K-means **is 0.43**.

It could also be analysed visually using silhouette plot for each cluster. Given below are the silhouette analysis diagram and 2-dimensional visualization of the clustering dataset (achieved using dimensionality reduction called PCA with 2 components). Each colour label denotes distinct respective cluster.

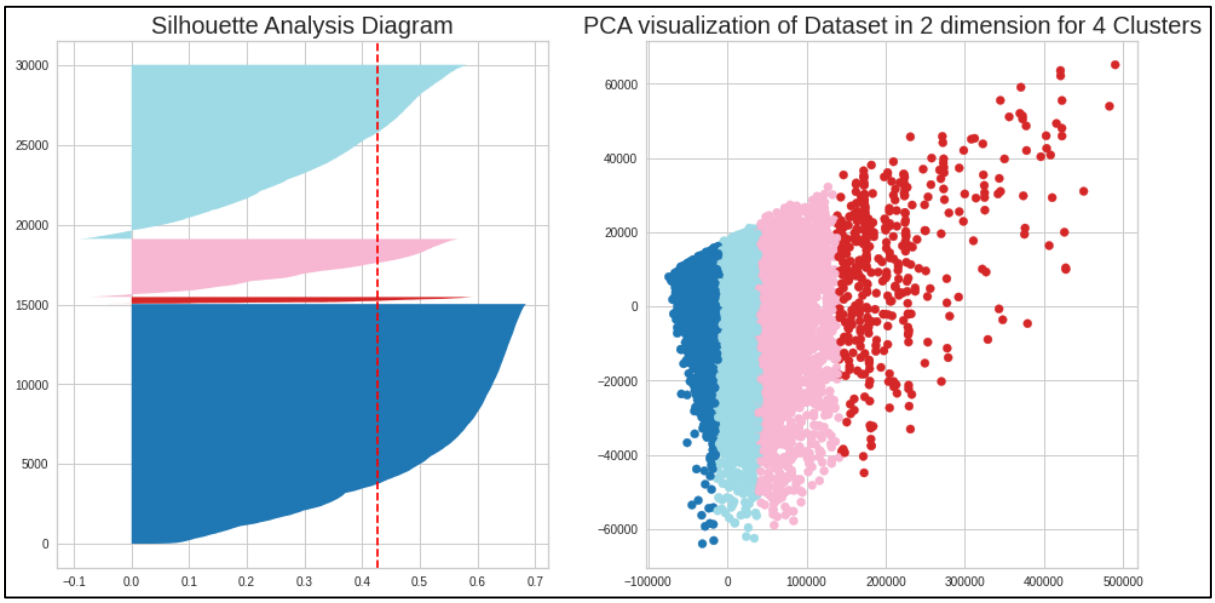

Figure 45: Silhouette Plot (Using K-Means Clustering Algorithm) and 2- Dimensional PCA Visualization of 30000 Loan data with color-labeled clusters *(scikit-learn.org, 2019)*

In the above given silhouette plot, the **y-axis represents the 30,000 data points related to previous loan records, which have been arranged in descending order of calculated silhouette coefficient within each cluster** (indicated by different colours). **The x-axis shows the silhouette coefficient for each data point**. The plot includes a vertical red dotted line at 0.43, which is the average silhouette score for the four clusters produced by the K-means algorithm.

The overall average **silhouette score as 0.43** from silhouette diagram indicates that the samples in the dataset are generally well-matched to their own clusters and poorly matched to other clusters. It suggests that there is a good amount of cohesion within each cluster and distance between the different clusters. The majority of the silhouette coefficients are located beyond the vertical line at 0.43, which supports the idea that the clusters are well-defined. **There are a few exceptions where the silhouette coefficients are negative, but overall, the score is positive, indicating that the clusters are well-separated and well-defined.**

Below is a visualization of the clusters after applying principal component analysis (PCA) to reduce the dimensions of the dataset to two dimensions. **The plot shows the cluster assignments with distinct and respective colour codes using K-means algorithm for each data point**.

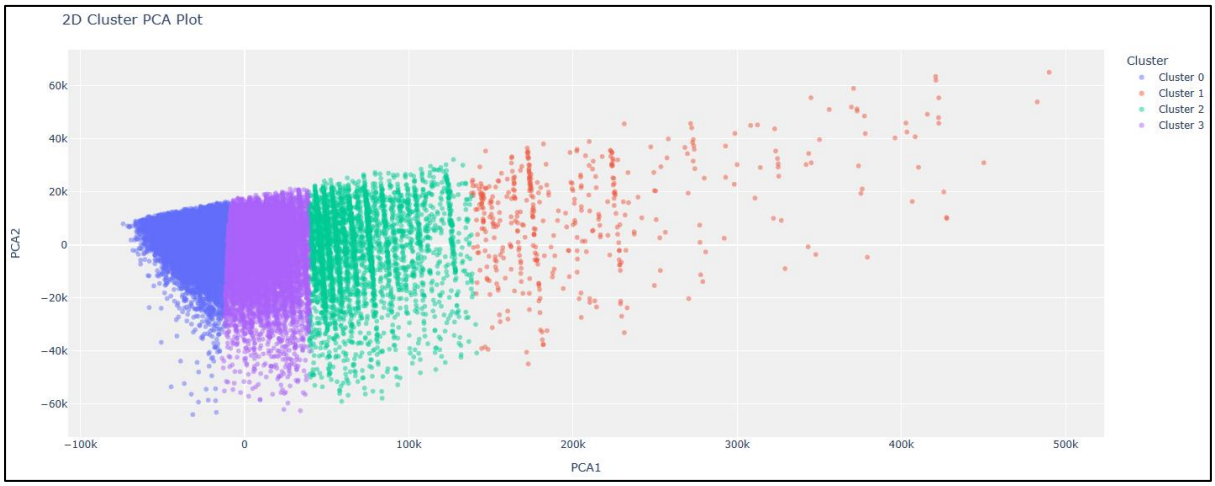

Figure 46: 2-Dimensional PCA plot for 30000 Loan Data with cluster specific color labels

Given below is distribution of data points for each cluster **using K-means algorithm**.

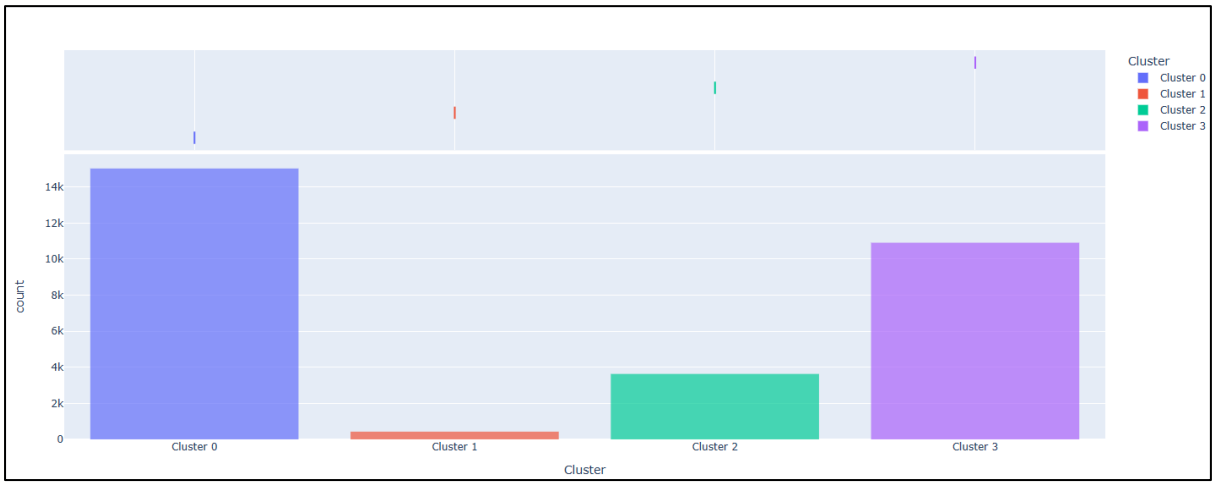

Figure 47: Distribution of 30000 loans in each cluster (using K-Means Clustering Algorithm)

The table below shows the number of loans that have been assigned to each cluster, **based on the cluster distribution shown above**.

Cluster number using K-Means	Distribution of Previous 30,000 Loans Among Clusters
Cluster 0	15,029 out of 30000 loans
Cluster 1	429 out of 30000 loans
Cluster 2	3631 out of 30000 loans
Cluster 3	10,911 out of 30000 loans

In summary, the K-Means algorithm was employed to classify a significant number of loans, with 15029 loans falling into cluster 0, which made up 50.09% of the total loans. Similarly, 10911 loans were assigned to cluster 3, accounting for 36.37% of the total loans. A relatively small proportion of loans, 1.43%, were allocated to cluster 1, with only 429 loans. Lastly, cluster 2 was found to include 3631 loans, representing 12.10% of the total loans.

The interpretation of each cluster in terms of probability of loan default is done later. Click here to reach the section about interpretation of cluster.

Creating and Visualizing Clusters using Hierarchical Clustering Algorithm

[Code Section Link]

Hierarchical clustering is an unsupervised machine learning technique used to create clusters. It starts by treating each data point as an individual cluster and repeatedly merges the two closest clusters together until all the data points are merged into a single cluster. (Patel, 2019).

One key limitation of using hierarchical clustering is that it is computationally very expensive, as the algorithm must find the two closest clusters and merge them at each iteration, which requires calculating the distance between all pairs of data points in the two clusters. This process must be repeated until all the data points are merged into a single cluster (Du & Lin, 2005).

Hierarchical clustering is done using following parameters (scikit-learn.org, 2018):

- ***affinity='euclidean'***: indicates the distance measure as Euclidean distance for calculating distance between two clusters.
- ***linkage='ward'***: indicates the method of clustering that tries to minimize the variance (or maximize the homogeneity) within each cluster.
- ***n_clusters=4***: indicates the optimal number of clusters determined by the elbow curve.

Hierarchical clustering was slower than k-means and produced less satisfactory results when applied to a dataset of 30,000 loan records. **Hierarchical clustering took 43 seconds ± 2.71 seconds to identify 4 clusters, while k-means took only 1.2 seconds ± 118 milliseconds**. **The silhouette score for the 4 clusters found by hierarchical clustering was 0.39, which suggests that k-means is more effective method for this particular dataset.**

The decision to use k-means instead of hierarchical clustering for this dataset is also supported by the below given silhouette plot for hierarchical clustering:

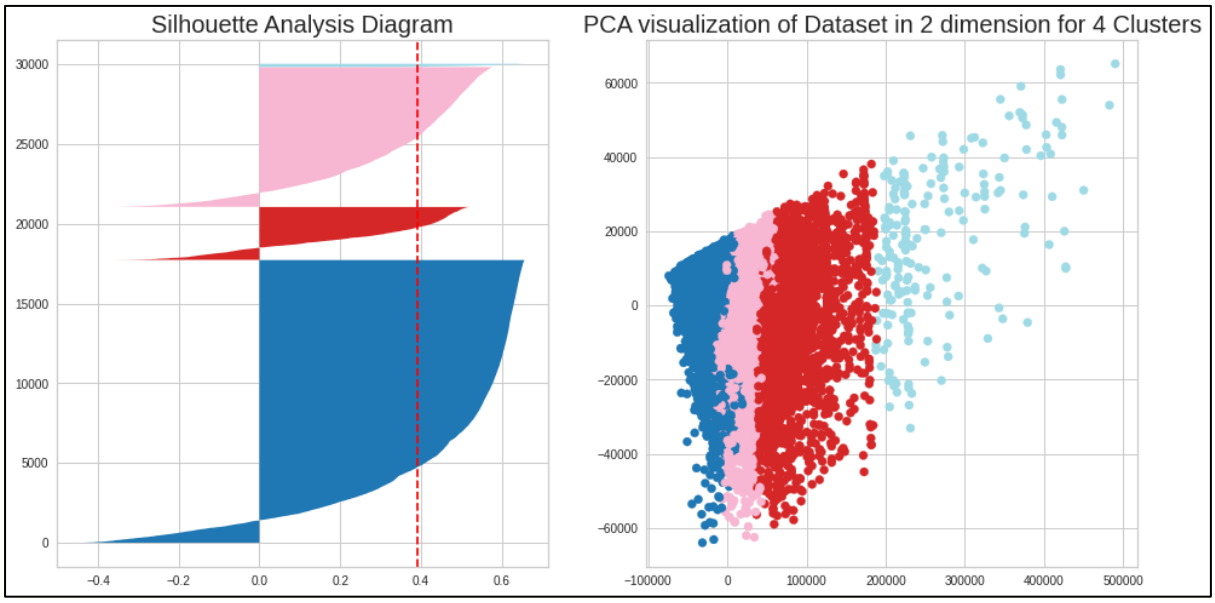

Figure 48: Silhouette Plot (Using Hierarchical Clustering Algorithm) and 2- Dimensional PCA Visualization of 30000 Loan data with color-labeled clusters. *(scikit-learn.org, 2019)*

The silhouette plot for the clusters created using hierarchical clustering shows **that many of the data points have a negative silhouette coefficient**, which contributes to a lower overall silhouette score of 0.39 for the clusters. This indicates that some of the data points were incorrectly assigned to the wrong clusters (Rousseeuw, 1987).

Hence, due to above mentioned reasons and limitation of hierarchical clustering algorithm, final interpretation of the clusters is given using only K-means clusters.

Interpretation of Clusters based on Probability of Loan Default

[Code Section Link]

After using the k-means algorithm to assign each of the previous 30,000 loan records to a cluster label, the final data with the cluster labels is shown below

	loan_amnt	emp_length	avg_fico_range	int_rate	annual_inc	dti	open_acc	revol_bal	HighRisk_LowRisk	kmeans_cluster_labels
0	5000.0	10	697.0	5.32	70000.0	11.14	6.0	7538.0	0	3
1	6000.0	5	777.0	5.42	50508.0	2.42	7.0	5794.0	0	0
2	13000.0	7	702.0	13.56	100000.0	13.37	12.0	11839.0	0	3
3	1500.0	0	682.0	15.49	30000.0	34.96	6.0	1228.0	0	0
4	21350.0	0	692.0	12.35	90000.0	21.64	9.0	23224.0	0	3
...	...	...	...	...	...	...	...	...	...	...
29995	2500.0	3	677.0	16.14	28000.0	9.17	5.0	4787.0	1	0
29996	3000.0	0	697.0	12.42	15600.0	3.46	10.0	3600.0	1	0
29997	9000.0	0	712.0	16.14	48000.0	32.35	7.0	18493.0	1	0
29998	8400.0	0	702.0	17.09	25000.0	34.42	6.0	6009.0	1	0
29999	12000.0	2	682.0	17.77	42000.0	30.26	10.0	13640.0	1	0

Figure 49: Assignment of Cluster Labels to each loan record using K-Means clustering algorithm

The above highlighted *'kmeans_cluster_labels*' column indicates the cluster label (0, 1, 2, or 3) assigned to each past loan record.

The probability of default for each of the 4 clusters in the '*kmeans_cluster_labels'* column of the *'clustering_data'* dataframe is calculated. The probability of default for a cluster is calculated by dividing the number of borrowers in that cluster who defaulted (indicated by a value of 1 in the *'HighRisk_LowRisk'* column) by the total number of borrowers in that cluster.

The mathematical calculation using conditional probability (Barone, 2022) for determining probability of default for each high risk and low risk loan records belong to respective clusters is given as follow:

P(D|C0) = Number of borrowers in C0 who defaulted / Total number of borrowers in C0

P(D|C1) = Number of borrowers in C1 who defaulted / Total number of borrowers in C1

P(D|C2) = Number of borrowers in C2 who defaulted / Total number of borrowers in C2

P(D|C3) = Number of borrowers in C3 who defaulted / Total number of borrowers in C3

where, '*HighRisk_LowRisk'* column is denoted as D, where D = 1 indicates default and D = 0 indicates no default.

C0 is cluster 0, C1 is cluster 1, C2 is cluster 2 and C3 is cluster 3

The resulting probability of high risk (**indicated by a label of 1**, which signifies that the borrower defaulted on this loan record) and low risk (**indicated by a label of 0**, which signifies that the borrower did not default on this loan record) for each cluster is shown below:

	kmeans_cluster_labels	HighRisk_LowRisk	Probability of Default
0	0	1	0.261600
1	0	0	0.239367
2	1	0	0.008700
3	1	1	0.005600
4	2	0	0.065900
5	2	1	0.055133
6	3	0	0.186033
7	3	1	0.177667

Figure 50: Probability of defaults associated with High Risk and Low Risk Loans in each individual Cluster

Finally, the average probability of default per cluster is determined and compared it to the average loan amount and annual income of loan applicants in each cluster, using probability formula given below:

- Mean loan amount for C_n = (Sum of loan amounts for all borrowers in C_n) / (Number of borrowers in C_n)

  where, C_n denotes cluster label and n denotes cluster 0, 1,2 or 3

- Mean annual income for C_n = (Sum of annual incomes for all borrowers in C_n) / (Number of borrowers in C_n)

  where, C_n denotes cluster label and n denotes cluster 0, 1,2 or 3

- Average probability of default for C_n = (Sum of probabilities of default for all borrowers in C_n) / (Number of borrowers in C_n)

  where, C_n denotes cluster label and n denotes cluster 0, 1,2 or 3

The resulting average probabilities of default in each cluster is normalized, so that sum of probability across the cluster is 1. The average probability of default for each cluster, along with the average loan amount requested by borrowers and their average annual income in each cluster, is shown below:

	kmeans_cluster_labels	loan_amnt	annual_inc	Probability of Default
0	0	11372.723	45324.274	0.501
1	1	24721.970	283687.961	0.014
2	2	21666.442	142220.742	0.121
3	3	17759.415	83495.597	0.364

Figure 51: Aggregated Probability of Default with Average Loan Amount and Average Annual Income for Individual Cluster using K-Means Clustering Algorithm

Therefore, each cluster can be interpreted based on the probability of default as follows:

- **Interpretation of Cluster 3 (Less Loan amount and Low Annual Income cluster – <u>36.37% of total loans</u>):**
  i. Loans in Cluster 3 have an average loan amount of 17,759.415 USD and an average annual income of 83,495.597 USD.
  ii. The probability of default for this group is 0.364, indicating that loan applicants with similar income and loan amount to those in Cluster 3 are likely to default on their loans.
  iii. It can be inferred that individuals in Cluster 3 are likely to be lower-income borrowers who are requesting lower loan amounts.
  iv. Given this group's high probability of default, lenders may be less likely to approve loans for this group or may approve them with higher interest rates.

- **Interpretation of Cluster 2 (Less Loan amount and High Annual Income cluster – <u>12.10% of total loans</u>):**
    i. Loans in Cluster 2 have an average loan amount of 21,666.442 USD and an average annual income of 142,220.742 USD.
    ii. The probability of default for this group is 0.121, indicating that loan applicants with similar income and loan amount to those in Cluster 2 have a moderate likelihood of defaulting on their loans.
    iii. It can be inferred that individuals in Cluster 2 are likely to be middle-income borrowers who are requesting moderate loan amounts.
    iv. Lenders may be more likely to approve loans for this group, but may still exercise caution with regards to their creditworthiness.
- **Interpretation of Cluster 0 (High Loan Amount and Low Annual Income cluster – <u>50.09% of total loans</u>):**
    i. Loans in Cluster 0 have an average loan amount of 11,372.723 USD and an average annual income of 45,324.274 USD.
    ii. The probability of default for this group is 0.501, indicating that loan applicants with similar income and loan amount to those in Cluster 0 are highly likely to default on their loans.
    iii. It can be inferred that individuals in Cluster 0 are likely to be lower-income borrowers who are requesting low loan amounts.
    iv. Given this group's high probability of default, lenders may be very cautious about approving loans for this group or may approve them with higher interest rates.
- **Interpretation of Cluster 1 (High Loan Amount and High Annual Income cluster – <u>1.43% of total loans</u>):**
    i. Loans in Cluster 1 have an average loan amount of 24,721.970 USD and an average annual income of 283,687.961 USD.
    ii. The probability of default for this group is 0.014, indicating that loan applicants with similar income and loan amount to those in Cluster 1 have a very low likelihood of defaulting on their loans.
    iii. It can be inferred that individuals in Cluster 1 are likely to be higher-income borrowers who are requesting large loan amounts.
    iv. Lenders may be more likely to approve loans for this group and may approve them with lower interest rates, given the low probability of default.

Predicting Cluster for new Loan Application and Evaluating Default Risk in terms of Probability of Default

[Code Section Link]

Suppose a new loan applicant, who has been employed for 9 years and in need of 7500 USD loan. The applicant has an average FICO score of 717, an annual income of $53,000, and a manageable debt-to-income ratio of 12.75 with 16 open credit lines with a revolving balance of $4890. After assessment, the financial institution is ready to give loan at 7.35% interest rate.

With the data from above information, the financial institution assigns loan application to a cluster by predicting using above model of K-Means algorithm. After prediction from K-means algorithm, it is predicted that this loan application belonged to cluster 0, which has 0.501 probability of loans in past. Therefore, financial institution may be very cautious about approving loan for this application.

Conclusion

In summary, this project effectively demonstrated the utilization of various machine learning techniques to address credit risk in the financial sector. Through the use of classification methods, the project was able to predict the likelihood of loan default. Regression models were employed to predict the Credit Conversion Factor, which is a crucial financial measure used to calculate expected loss in the event of loan default. Additionally, clustering techniques were used to group loans based on financial information and assign them with a specific risk or probability of default.

The project findings indicate that it is possible to predict the probability of loan default and identify high-risk borrowers using classification machine learning models. The final selected random forest classification model had an accuracy of 64.64% and recall of 66.83% on new loans, with an incorrect prediction rate of only 33.16% for high-risk loans. It is important to note that the results of the study were based on historical data and assumed that it is representative of future loan applications. The analysis also only used a limited set of attributes, and other factors that may influence the risk of default were not considered. To improve the predictions, further research should focus on enhancing the prediction performance of the models and optimizing feature engineering for classification models and feature selection for loan risk analysis.

The project also found that it is possible to use regression algorithms to predict the important financial measure, Credit Conversion Factor (CCF) and determine the exposed loss in case of a loan default. The final selected regression model AdaBoost achieved the R2 score of 0.82 and mean squared error of 0.01. This implies that the model is able to predict the CCF accurately and with a small error, which could be useful for financial institutions to manage their credit risk by estimating their expected

loss in case of default on a loan. However, the main limitation with prediction of CCF is that it requires much more in-depth financial domain knowledge and other considerations with much larger dataset and this project approaches the solution with very limited finance knowledge, which was outside the scope of this project.

Finally, Unsupervised machine learning algorithms were used to group loans based on financial information and they were associated with a specific risk or probability of default. K-Means and Hierarchical Clustering were used for the clustering experiments, and it was found that K-Means had a higher silhouette score of 0.42, indicating that the cluster assignments were more coherent and well-defined. The probability of defaults was assigned to loans in each cluster, and the potential of unsupervised machine learning algorithms to group loans based on financial information and associate them with a specific risk or probability of default was demonstrated. However, in terms of limitations, a limited set of financial attributes were used for the analysis and other factors that may also impact the risk of default. Additionally, the optimal number of clusters was determined using one specific method, and different methods or evaluation criteria may yield different results.

Overall, the project highlights the potential of these techniques for managing credit risk in the financial industry and serves as a starting point for further research to improve the prediction performance of the models and account for additional factors that may impact the risk of default.

References

365DataScience, 2021. *Correlation Analysis: All the Basics You Need to Know.* [Online]
Available at: https://365datascience.com/trending/correlation-analysis/
[Accessed 9 January 2023].

Apostolik, R., 2009. *Foundations of banking risk : an overview of banking, banking risks, and risk-based banking regulation.* Hoboken, N.J.: John Wiley & Sons.

Arthur, D. & Vassilvitskii, S., 2007. K-Means++: The Advantages of Careful Seeding. *Proc. of the Annu. ACM-SIAM Symp. on Discrete Algorithms,* pp. 1027-1035.

Ayantola, A., 2020. Minimizing Credit Risk In Peer-to-Peer Lending Business Using Supervised Machine Learning Techniques. *Doctoral dissertation, Dublin, National College of Ireland.*

Banks, D., 2010. Statistical data mining. *Wiley Interdisciplinary Reviews: Computational Statistics,* 2(1), pp. 9-25.

Barone, A., 2022. *Conditional Probability: Formula and Real-Life Examples.* [Online]
Available at: https://www.investopedia.com/terms/c/conditional_probability.asp
[Accessed 10 January 2023].

Bhandari, A., 2020. *Guide to Auc-Roc Curve in Machine Learning.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/
[Accessed 10 January 2023].

bis.org, 2001. *The Standardised Approach to Credit Risk.* [Online]
Available at: https://www.bis.org/publ/bcbsca04.pdf
[Accessed 6 January 2022].

Brown, I., 2011. *ResearchGate.* [Online]
Available at:
https://www.researchgate.net/publication/296596808_Regression_Model_Development_for_Credit_Card_Exposure_At_Default_EAD_using_SASSTATR_and_SASR_Enterprise_Miner_53
[Accessed 8 January 2023].

Brownlee, J., 2014. *Assessing and Comparing Classifier Performance with ROC Curves.* [Online]
Available at: https://machinelearningmastery.com/assessing-comparing-classifier-performance-roc-curves-2/
[Accessed 10 January 2023].

Brownlee, J., 2020. Feature Selection. In: *Data Cleaning, Feature Selection,and Data Transforms in Python.* s.l.:s.n., pp. 110-115.

Brownlee, J., 2020. How to Scale Numerical Data. In: *Data Preparation for Machine Learning.* s.l.:s.n., pp. 213-229.

Brownlee, J., 2020. How to Select Features for Numerical. In: *Data Cleaning, Feature Selection,and Data Transforms in Python.* s.l.:s.n., pp. 156-170.

Brownlee, J., 2020. How to Select Numerical Input. In: *Data Cleaning, Feature Selection,and Data Transforms in Python.* s.l.:s.n., pp. 119-135.

Cooper, P. M., 2013. *Credit Risk Management: A Practitioner's Guide to the Regulation of Credit Risk.* s.l.:John Wiley & Sons.

Croux, C., Jagtiani, J., Korivi, T. & Vulanovic, M., 2020. Important factors determining Fintech loan default: Evidence from a lendingclub consumer platform. *Journal of Economic Behavior & Organization,* Volume 173, pp. 270-296.

David M. Levine, K. A. S. D. F. S., 2020. The Covariance and the Coefficient of Correlation. In: *Business Statistics A First Course.* s.l.:s.n., pp. 187-190.

Dorogush, A. V., Ershov, V. & Gulin, A., 2017. *Parameter Tuning.* [Online]
Available at: https://catboost.ai/docs/concepts/parameter-tuning.html
[Accessed 10 1 2023].

Du, Z. & Lin, F., 2005. A novel parallelization approach for hierarchical clustering. *Parallel Computing,* 31(5), pp. 523-527.

Fawcett, T., 2004. ROC Graphs: Notes and Practical Considerations for Researchers. Volume 31, pp. 1-38.

Ferreira, L. E. B., Barddal, J. P., Gomes, H. M. & Enembreck, F., 2017. Improving Credit Risk Prediction in Online Peer-to-Peer (P2P) Lending Using Imbalanced Learning Techniques. *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI),* Volume 105.

FICO®score, 2022. *FREQUENTLY ASKED QUESTIONS ABOUT FICO® SCORES.* [Online]
Available at: https://www.ficoscore.com/ficoscore/pdf/Frequently-Asked-Questions-About-FICO-

Scores.pdf
[Accessed 9 January 2023].

Freund, Y. & Schapire, R. E., 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences,* 55(1), pp. 119-139.

Frost, J., 2018. *How To Interpret R-squared in Regression Analysis.* [Online]
Available at: https://statisticsbyjim.com/regression/interpret-r-squared-regression/
[Accessed 10 January 2023].

Frost, J., 2019. *5 Ways to Find Outliers in Your Data.* [Online]
Available at: https://statisticsbyjim.com/basics/outliers
[Accessed 10 January 2023].

Fuentes, A., 2018. A brief introduction to feature engineering. In: *Hands-On Predictive Analytics.* s.l.:Packt Publishing, pp. 69-71.

Fuentes, A., 2018. Bivariate EDA. In: *Hands-On Predictive Analytics.* s.l.:Packt Publishing, pp. 86-108.

Fuentes, A., 2018. Univariate EDA. In: *Hands-On Predictive Analytics with Python.* s.l.:Packt Publishing, pp. 76-85.

Galli, S., 2020. Encoding with the Weight of Evidence. In: *Python Feature Engineering Cookbook.* s.l.:Packt Publishing, pp. 125-128.

Galli, S., 2021. *RandomSampleImputer.* [Online]
Available at: https://feature-engine.readthedocs.io/en/1.1.x/imputation/RandomSampleImputer.html
[Accessed 9 January 2023].

Georgiev, N., 2023. *Credit Risk Modeling in Python Course - 365 Data Science.* [Online]
Available at: https://365datascience.com/courses/credit-risk-modeling-in-python/
[Accessed 9 January 2023].

Gordon, A. D. et al., 1984. Classification and Regression Trees.. *Biometrics,* 40(3), p. 874.

Hastie, T., Tibshirani, R. & Friedman, J., 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* s.l.:Springer.

Investopedia, 2021. *Charge-Off: Definition, Effect on Credit Score, and How to Remove.* [Online]
Available at: https://www.investopedia.com/terms/c/chargeoff.asp
[Accessed 9 January 2023].

Investopedia, 2022. *Debt Consolidation: Here's What You Need To Know.* [Online]
Available at: https://www.investopedia.com/terms/d/debtconsolidation.asp
[Accessed 9 January 2023].

Investopedia, 2022. *Default: What It Means, What Happens When You Default, Examples.* [Online]
Available at: https://www.investopedia.com/terms/d/default2.asp#:~:text=A%20default%20occurs%20when%20a,their%20future%20access%20to%20credit.
[Accessed 9 January 2023].

J.A Cook, J. R., 2016. Overfitting. *Journal of British Surgery,* 103(13), pp. 1814-1814.

Jobson, J. D., 1995. *Applied multivariate data analysis. Vol. 1 Regression and experimental design [Hauptbd.].* s.l.:Editorial: New York Springer.

Kenton, W., 2021. *Default Probability Definition for Individuals & Companies.* [Online]
Available at: https://www.investopedia.com/terms/d/defaultprobability.asp
[Accessed 10 January 2023].

Klosterman, S., 2021. Splitting The Data: Training And Test Sets. In: *Data Science Projects with Python.* s.l.:Packt Publishing, pp. 86-88.

Kotu, V. & Deshpande, B., 2019. Data Science Concepts and Practice. In: *Data Exploration.* s.l.:Jonathan Simpson, pp. 47-48.

Koulafetis, P., 2017. Modern Credit Risk Management. In: *Theory and Practice.* s.l.:Palgrave Macmillan UK, p. 134.

LendingClub, 2019. *Lending Club's Interest rate detail.* [Online]
Available at: https://www.lendingclub.com/foliofn/rateDetail.action
[Accessed 9 January 2023].

LendingClub, 2020. *What Do the Different Note Statuses Mean?.* [Online]
Available at: https://help.lendingclub.com/hc/en-us/articles/216109367-What-Do-the-Different-Note-Statuses-Mean-
[Accessed 9 January 2023].

LendingClub, 2022. *LendingClub.* [Online]
Available at: https://www.lendingclub.com/
[Accessed 9 January 2023].

Li, P. & Han, G., 2018. *LendingClub Loan Default and Profitability Prediction.* [Online]
Available at: https://cs229.stanford.edu/proj2018/report/69.pdf
[Accessed 9 January 2023].

Liu, Y. et al., 2022. Applying machine learning algorithms to predict default probability in the online credit market: Evidence from China. *International Review of Financial Analysis,* Volume 79, p. 101971.

Menardi, G. & Torelli, N., 2014. Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery,* Volume 28, p. 92–122.

Nordhausen, K., 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition by Trevor Hastie, Robert Tibshirani, Jerome Friedman. *International Statistical Review,* 77(3), pp. 482-482.

Nwanganga, F. & Chapple, M., 2020. Choosing the Right Number of Clusters. In: *Practical Machine Learning in R.* s.l.:Wiley, pp. 409-412.

Patel, A. A., 2019. Clustering. In: *Hands-On Unsupervised Learning Using Python : How to Build Applied Machine Learning Solutions from Unlabeled Data.* s.l.:O'Reilly Media, pp. 127-136.

Patel, A. A., 2019. Hierarchical Clustering. In: *Hands-On Unsupervised Learning Using Python : How to Build Applied Machine Learning Solutions from Unlabeled Data.* s.l.:O'Reilly Media, pp. 138-142.

PyCaret, 2022. *pycaret.org.* [Online]
Available at: https://pycaret.org/
[Accessed 10 1 2023].

Qi, M., 2009. *Exposure at Default of Unsecured Credit Cards.* s.l.:Economics working paper 2009-2.

Raschka, S. & Mirjalili, V., 2015. *Python Machine Learning. Third Edition | Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2.* s.l.:Packt Publishing.

Raschka, S. & Mirjalili, V., 2019. Evaluating the performance of linear regression models. In: P. Publishing, ed. *Python Machine Learning. Third Edition | Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2.* s.l.:s.n., pp. 334-337.

Raschka, S. & Mirjalili, V., 2019. Learning Best Practices for Model Evaluation and Hyperparameter Tuning. In: *Python Machine Learning. Third Edition | Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2.* s.l.:Packt Publishing, pp. 191-220.

Raschka, S. & Mirjalili, V., 2019. Performing one-hot encoding on nominal. In: *Python Machine Learning. Third Edition | Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2.* s.l.:Packt Publishing, pp. 118-120.

Raschka, S. & Mirjalili, V., 2019. Quantifying the quality of clustering via silhouette plots. In: *Python Machine Learning. Third Edition | Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2.* s.l.:Packt Publishing, pp. 363-367.

Rousseeuw, P. J., 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics,* Volume 20, pp. 53-65.

Sarkar, D. & Natarajan, V., 2019. Introduction to random forests. In: s.l.:Packt Publishing, Limited, pp. 136-145.

scikit-learn.org, 2012. *sklearn.ensemble.RandomForestClassifier.* [Online]
Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[Accessed 10 1 2023].

scikit-learn.org, 2013. *Metrics and scoring: quantifying the quality of predictions.* [Online]
Available at: https://scikit-learn.org/stable/modules/model_evaluation.html
[Accessed 10 January 2023].

scikit-learn.org, 2018. *sklearn.cluster.AgglomerativeClustering.* [Online]
Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html
[Accessed 10 1 2023].

scikit-learn.org, 2019. *Selecting the number of clusters with silhouette analysis on KMeans clustering.* [Online]
Available at: https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html
[Accessed 12 January 2023].

scikit-learn.org, 2022. *RobustScaler.* [Online]
Available at: https://scikit-

learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html
[Accessed 10 January 2023].

scikit-learn.org, 2022. *sklearn.cluster.KMeans.* [Online]
Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[Accessed 10 1 2023].

scikit-learn.org, 2022. *sklearn.feature_selection.SelectKBest.* [Online]
Available at: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html
[Accessed 9 January 2023].

scikit-learn.org, 2023. *sklearn.linear_model.LogisticRegression.* [Online]
Available at: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[Accessed 10 January 2023].

scikit-learn.org, 2023. *sklearn.preprocessing.StandardScaler.* [Online]
Available at: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
[Accessed 10 January 2023].

scikit-yb.org, 2019. *Elbow Method.* [Online]
Available at: https://www.scikit-yb.org/en/latest/api/cluster/elbow.html
[Accessed 10 January 2023].

Serrano-Cinca, C., Gutiérrez-Nieto, B., López-Palacios, L. & Bask, M., 2015. Determinants of Default in P2P Lending. *PLOS ONE,* 10(10), p. e0139427.

Shan, Q. & Nilsson, M., 2018. *Credit risk analysis with machine learning techniques in peer-to-peer lending market,* Sweden: Stockholm Business School.

Shi, C. et al., 2021. A quantitative discriminant method of elbow point for the optimal number of clusters in clustering algorithm. *EURASIP Journal on Wireless Communications and Networking,* pp. 1-16.

Shiffler, R. E., 1988. Maximum Z Scores and Outliers. In: *The American Statistician.* s.l.:s.n., pp. 79-80.

Stoltzfus, J. C., 2011. Logistic Regression: A Brief Primer. *Academic emergency medicine,* 18(10), p. 1099–1104.

Sugar, C. A. & James, G. M., 2003. Finding the Number of Clusters in a Dataset. *Journal of the American Statistical Association,* 98(463), pp. 750-763.

Tong, E. N., 2016. Exposure at default models with and without the credit conversion factor. *European Journal of Operational Research,* 252(3).

Tukey, J., 1977. Exploratory data analysis. In: s.l.:s.n., pp. 131-160.

Turiel, J. D. & Aste, T., 2020. Peer-to-peer loan acceptance and default prediction with artificial intelligence. *Royal Society Open Science,* 7(6), p. 191649.

Turney, S., 2022. *Chi-Square Test of Independence | Formula, Guide & Examples.* [Online]
Available at: https://www.scribbr.com/statistics/chi-square-test-of-independence/
[Accessed 9 January 2023].

Valvonis, V., 2008. Estimating EAD for retail exposures for Basel II purposes. *The Journal of Credit Risk,* 4(1), pp. 79-109.

Yandex, 2017. *CatBoost.* [Online]
Available at: https://catboost.ai/en/docs/concepts/algorithm-main-stages
[Accessed 10 January 2023].

Appendix

List of additional Python Libraries Installed in Google Colab Environment

The following are the additional python libraries that were installed with specific versions in the Google Colab Notebook environment. It is crucial to use these specific versions of each library to ensure compatibility with other libraries and successful execution of the code in Colab Notebook.

pycaret==2.3.5
mlflow==1.22.0
mlxtend==0.19.0
numba==0.53.0
numexpr==2.8.1
catboost==1.0.3
feature-engine==1.1.2
lightgbm==3.3.1
matplotlib==3.5.1
matplotlib-inline==0.1.3
pandas==1.3.5
pandas-profiling==3.1.0
pandocfilters==1.5.0
plotly==5.5.0
scikit-learn==0.23.2
scipy==1.5.4
seaborn==0.11.2
shap==0.40.0
librosa
optuna
category_encoders
scikit-optimize

Link to the Shared Colab Notebooks for solving each specific problem

Problem Name	Colab Notebook Link
Exploratory Data Analysis	https://colab.research.google.com/drive/1YBUhHVEDNMukKuQ46TV1YPXh2ugf4wr9?usp=sharing
Classification Problem	https://colab.research.google.com/drive/1a5E2s0o7wH3Q4nNaNsZnJDbrJKYhNPUr?usp=sharing
Regression Problem	https://colab.research.google.com/drive/1RbD639F3YKQtIm12DEm2OqOZTN_u4WHG?usp=sharing
Clustering Problem	https://colab.research.google.com/drive/1HCtdNIeAhUfCcsgcQf3HBzuvhX7Hh8Lv?usp=sharing

Columns removed and their description for Data Preparation for classification

In section related to Data Preparation for Classification problem, following columns were removed. The list of columns and its description along with respective reasons to remove are given below for quick reference:

	Column Name	Column Description
Reason To Remove	Remove the columns which are not related to the purpose	
	Unnamed: 0	Row Number starting from 0
	url	URL for the LC page with listing data.
	id	A unique LC assigned ID for the loan listing.
	zip_code	The first 3 numbers of the zip code provided by the borrower in the loan application.
	title	The loan title provided by the borrower
	emp_title	The job title supplied by the Borrower when applying for the loan.
	inq_fi	Number of personal finance inquiries
	mths_since_recent_inq	Months since most recent inquiry.
	inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
	inq_last_12m	Number of credit inquiries in past 12 months

Reason To Remove	Remove the categorical columns which has high proportion of only one category	
	chargeoff_within_12_mths	Number of charge-offs within 12 months
	tax_liens	Number of tax liens
	policy_code	publicly available policy_code=1innew products not publicly available policy_code=2
	num_tl_120dpd_2m	Number of accounts currently 120 days past due (updated in past 2 months)
	num_tl_30dpd	Number of accounts currently 30 days past due (updated in past 2 months)
	pymnt_plan	Indicates if a payment plan has been put in place for the loan
	acc_now_delinq	The number of accounts on which the borrower is now delinquent.
	debt_settlement_flag	Flags whether or not the borrower, who has charged-off, is working with a debt-settlement company.
	delinq_amnt	The past-due amount owed for the accounts on which the borrower is now delinquent.
	num_tl_90g_dpd_24m	Number of accounts 90 or more days past due in last 24 months

Reason To Remove	Remove the columns which have values from future i.e. when loan has been granted	
	funded_amnt	The total amount committed to that loan at that point in time.
	funded_amnt_inv	The total amount committed by investors for that loan at that point in time.
	total_pymnt	Payments received to date for total amount funded
	recoveries	post charge off gross recovery
	hardship_flag	Flags whether or not the borrower is on a hardship plan
	collections_12_mths_ex_med	Number of collections in 12 months excluding medical collections
	out_prncp	Remaining outstanding principal for total amount funded
	total_pymnt_inv	Payments received to date for portion of total amount funded by investors
	collection_recovery_fee	post charge off collection fee
	installment	The monthly payment owed by the borrower if the loan originates.
	next_pymnt_d	Next scheduled payment date
	out_prncp_inv	Remaining outstanding principal for portion of total amount funded by investors
	total_rec_prncp	Principal received to date
	tot_coll_amt	Total collection amounts ever owed
	last_fico_range_low	The lower boundary range the borrower's last FICO pulled belongs to.
	total_rec_int	Interest received to date
	last_pymnt_amnt	Last total payment amount received
	total_il_high_credit_limit	Total installment high credit/credit limit
	revol_util	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.
	total_rec_late_fee	Late fees received to date
	last_fico_range_high	The upper boundary range the borrower's last FICO pulled belongs to.
	issue_d	The month which the loan was funded
	last_pymnt_d	Last month payment was received
	last_credit_pull_d	The most recent month LC pulled credit for this loan

Reason To Remove	Remove the columns which has more than 80% missing values	
	mths_since_last_record	The number of months since the last public record.
	annual_inc_joint	The combined self-reported annual income provided by the co-borrowers during registration
	dti_joint	A ratio calculated using the co-borrowers' total monthly payments on the total debt obligations, excluding mortgages and the requested LC loan, divided by the co-borrowers' combined self-reported monthly income
	verification_status_joint	
	mths_since_recent_bc_dlq	Months since most recent bankcard delinquency
	revol_bal_joint	Sum of revolving credit balance of the co-borrowers, net of duplicate balances
	sec_app_fico_range_low	FICO range (high) for the secondary applicant
	sec_app_fico_range_high	FICO range (low) for the secondary applicant
	sec_app_earliest_cr_line	Earliest credit line at time of application for the secondary applicant
	sec_app_inq_last_6mths	Credit inquiries in the last 6 months at time of application for the secondary applicant
	sec_app_mort_acc	Number of mortgage accounts at time of application for the secondary applicant
	sec_app_open_acc	Number of open trades at time of application for the secondary applicant
	sec_app_revol_util	Ratio of total current balance to high credit/credit limit for all revolving accounts
	sec_app_open_act_il	Number of currently active installment trades at time of application for the secondary applicant
	sec_app_num_rev_accts	Number of revolving accounts at time of application for the secondary applicant
	sec_app_chargeoff_within_12_mths	Number of charge-offs within last 12 months at time of application for the secondary applicant
	sec_app_collections_12_mths_ex_med	Number of collections within last 12 months excluding medical collections at time of application for the secondary applicant
	hardship_type	Describes the hardship plan offering
	hardship_reason	Describes the reason the hardship plan was offered
	hardship_status	Describes if the hardship plan is active, pending, canceled, completed, or broken
	deferral_term	Amount of months that the borrower is expected to pay less than the contractual monthly payment amount due to a hardship plan
	hardship_amount	The interest payment that the borrower has committed to make each month while they are on a hardship plan
	hardship_start_date	The start date of the hardship plan period
	hardship_end_date	The end date of the hardship plan period
	payment_plan_start_date	The day the first hardship plan payment is due. For example, if a borrower has a hardship plan period of 3 months, the start date is the start of the three-month period in which the borrower is allowed to make interest-only payments.
	hardship_length	The number of months the borrower will make smaller payments than normally obligated due to a hardship plan
	hardship_dpd	Account days past due as of the hardship plan start date
	hardship_loan_status	Loan Status as of the hardship plan start date
	orig_projected_additional_accrued_interest	The original projected additional interest amount that will accrue for the given hardship payment plan as of the Hardship Start Date. This field will be null if the borrower has broken their hardship payment plan.
	hardship_payoff_balance_amount	The payoff balance amount as of the hardship plan start date
	hardship_last_payment_amount	The last payment amount as of the hardship plan start date

Reason To Remove	Remove the numerical columns which are highly correlated	
	num_sats	Number of satisfactory accounts
	num_rev_tl_bal_gt_0	Number of revolving trades with balance >0
	tot_hi_cred_lim	Total high credit/credit limit
	bc_open_to_buy	Total open to buy on revolving bankcards.
	percent_bc_gt_75	Percentage of all bankcard accounts > 75% of limit.
	total_bal_ex_mort	Total credit balance excluding mortgage
	num_rev_accts	Number of revolving accounts
	num_actv_bc_tl	Number of currently active bankcard accounts
	avg_cur_bal	Average current balance of all accounts
	open_rv_12m	Number of revolving trades opened in past 12 months
	acc_open_past_24mths	Number of trades opened in past 24 months.
	mths_since_recent_revol_delinq	Months since most recent revolving delinquency.
	num_op_rev_tl	Number of open revolving accounts
	total_rev_hi_lim	Total revolving high credit/credit limit
Reason To Remove	Remove the Sub-Grade column and retain only Grade column	
	sub_grade	LC assigned loan subgrade

Selected Features for Classification problem using ANOVA F-Test and Chi-Squared Test

Result of ANOVA F-Test:

The table below lists column names, F-Score statistics, and p-values. These values are calculated by comparing each numerical column to the binary target column ('HighRisk_LowRisk') using statistical testing. Columns highlighted in green are selected because their p-value is less than 0.05, while columns highlighted in red are removed.

Numerical_Feature	F-Score	p values
int_rate	28033.128	0.00E+00
term	11688.934	0.00E+00
avg_fico_range	7014.881	0.00E+00
loan_amnt	4129.2369	0.00E+00
mort_acc	2894.8121	0.00E+00
open_rv_24m	2869.4004	0.00E+00
tot_cur_bal	1867.1265	0.00E+00
num_tl_op_past_12m	1795.1463	0.00E+00
num_actv_rev_tl	1791.3999	0.00E+00
mths_since_earliest_cr_line	1778.2481	0.00E+00
total_bc_limit	1402.2998	0.00E+00
all_util	1325.0843	0.00E+00
dti	1094.1953	0.00E+00

open_il_12m	1055.3824	0.00E+00
mo_sin_old_rev_tl_op	985.14459	0.00E+00
bc_util	888.59246	0.00E+00
total_acc	799.8624	0.00E+00
emp_length	783.43607	0.00E+00
mths_since_recent_bc	669.20384	0.00E+00
open_acc_6m	649.38169	0.00E+00
mo_sin_old_il_acct	639.86485	0.00E+00
mo_sin_rcnt_rev_tl_op	621.24097	0.00E+00
pct_tl_nvr_dlq	427.62089	0.00E+00
mo_sin_rcnt_tl	409.85522	0.00E+00
num_bc_tl	346.48223	0.00E+00
mths_since_rcnt_il	279.177	0.00E+00
pub_rec	275.08368	0.00E+00
pub_rec_bankruptcies	253.96267	0.00E+00
open_il_24m	246.92336	0.00E+00
num_bc_sats	232.11053	0.00E+00
num_il_tl	212.33674	0.00E+00
num_accts_ever_120_pd	165.29036	0.00E+00
delinq_2yrs	153.62754	0.00E+00
revol_bal	106.79113	0.00E+00
annual_inc	60.399235	0.00E+00
total_cu_tl	51.036601	0.00E+00
il_util	50.697194	0.00E+00
mths_since_last_major_derog	49.664433	0.00E+00
open_acc	49.151459	0.00E+00
mths_since_last_delinq	47.748012	0.00E+00
max_bal_bc	32.683458	1.09E-08
open_act_il	3.455897	6.30E-02
total_bal_il	0.162263	6.87E-01

Result of Chi-Squared Test:

Similarly, Result of Chi-Squared test by comparing categorical columns with binary target column('HighRisk_LowRisk'):

Feature	p-value
grade	0.00E+00
home_ownership	0.00E+00
verification_status	0.00E+00
purpose	0.00E+00
addr_state	0.00E+00
initial_list_status	0.00E+00
application_type	0.00E+00