

**APPROVAL PREDICTION**

**TO GET A LOAN**

Carlos Santana Bobadilla

Mark Christian Albinto

Table of Contents

[1. Introduction 3](#_Toc134698482)

[Phase 1: Business Understanding 3](#_Toc134698483)

[Phase 2: Data Understanding 3](#_Toc134698484)

[Phase 3: Data Preparation 3](#_Toc134698485)

[Phase 4: Modeling 4](#_Toc134698486)

[Phase 5: Evaluation 4](#_Toc134698487)

[Phase 6: Deployment 4](#_Toc134698488)

[2. Import Necessary Libraries 4](#_Toc134698489)

[3. Import the Data 6](#_Toc134698490)

[4. Answering Research Questions - Descriptive Statistics 8](#_Toc134698491)

[4.1 Data pre-processing 9](#_Toc134698492)

[**4.2 EDA of Categorical Variables (qualitative variables)** 17](#_Toc134698493)

[**4.3 EDA of Numerical Variables** 21](#_Toc134698494)

[4.4 What is the gender distribution of loan applicants 23](#_Toc134698495)

[**4.5 What is the marital status distribution of loan applicants?** 24](#_Toc134698496)

[**4.6** What is the education level of loan applicants? 25](#_Toc134698497)

[4.7 What is the income distribution of loan applicants? 25](#_Toc134698498)

[4.8 What is the loan amount distribution of loan applicants? 26](#_Toc134698499)

[4.9 What is the credit history distribution of loan applicants? 27](#_Toc134698500)

[4.10 What is the loan approval status distribution? 28](#_Toc134698501)

[4.11 What is the income distribution of loan applicants? 29](#_Toc134698502)

[5 Development of a Machine Learning Model End to End Process 30](#_Toc134698503)

[5.1 Correlation Matrix Heatmap 30](#_Toc134698504)

[5.2 Separate the variables into feature matrix (X) and target variable (y) 32](#_Toc134698505)

[5.3 Scaling Using Standard Scaler 32](#_Toc134698506)

[5.4 Logistic Regression 34](#_Toc134698507)

[6 References 41](#_Toc134698508)

Roles and Responsibilities

Carlos Santana Bobadilla

1. Business Understanding (SCRS Business Analytical Tool)
2. Data Understanding
3. Descriptive Statistics
4. Data Pre-processing
5. Exploratory Data Analysis

Combined Effort

1. Data Preparation
2. Filling Missing Values
3. Checking outliers
4. Encoding categorical variables
5. Poster presentation

Carlos

1. Abstract
2. Research questions
3. CRISP-DM framework
4. Methods

Mark

1. Methods
2. Findings
3. Conclusion

Mark Christian Albinto

1. Model selection
2. Input feature selection
3. Splitting dataset
4. Scaling
5. Modelling (Logistic Regression)
6. Confusion Matrix and Accuracy Score

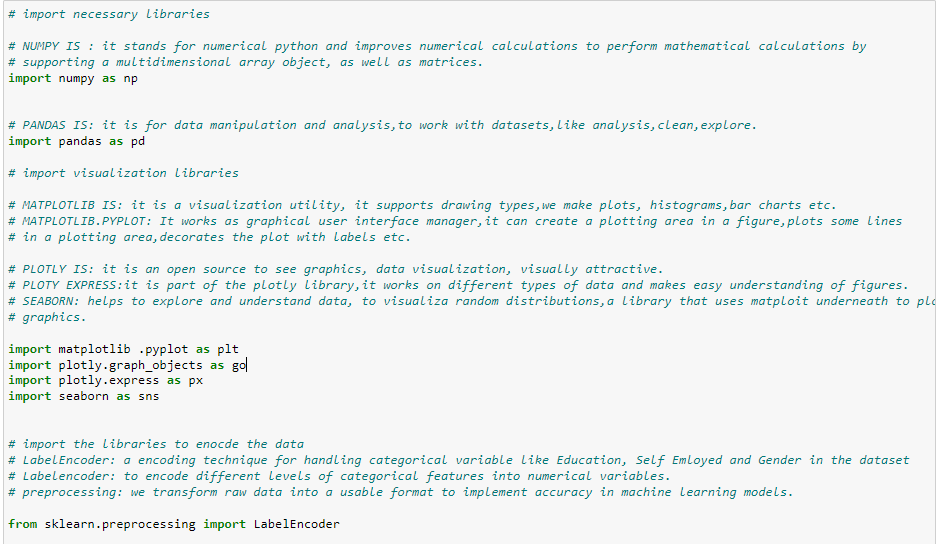
Please note that even if we have our own part to focus on the project, we often communicate to each other our findings and ask each other opinions regarding the part we are working on.

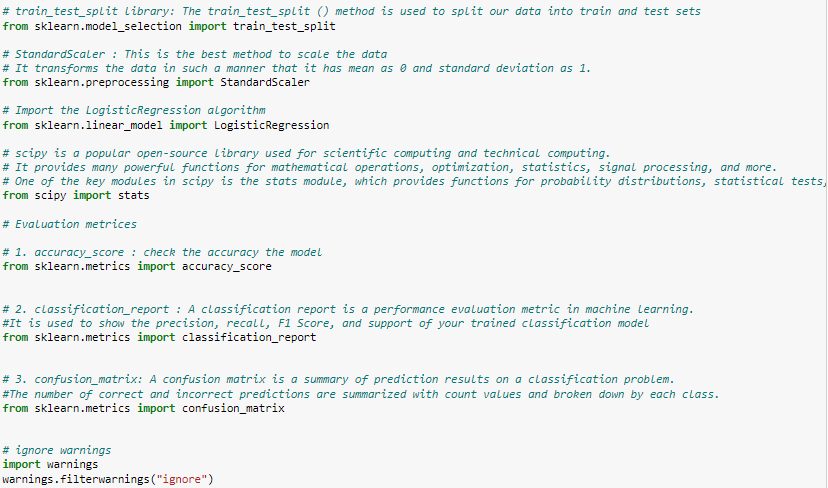
# 1. Introduction

In this case study, our objective is to create a model that can forecast whether or not a loan application will be approved. Banks and other financial organizations must complete this important activity in order to make decisions regarding whether or not to approve a loan for an applicant. Accurately predicting loan approval can lower the risk of default and guarantee that loans are only given to borrowers who have a higher likelihood of repaying them. We will use the CRISP-DM methodology, which offers a structured approach to data science initiatives, to do this.

# 2. Import Necessary Libraries

We must import the required libraries for data processing, visualization, and modelling before we can begin our investigation. The following libraries will be used:





**Numpy:**  is a library that supports multidimensional arrays and matrices in numerical computations.

**Pandas**: We may deal with datasets, conduct analyses, and clean data using Pandas, a tool for data manipulation and analysis.

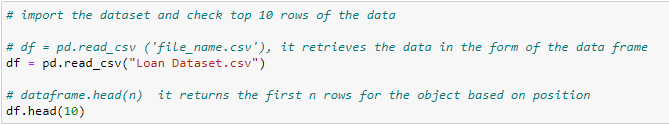
**Matplotlib and Seaborn:**  are visualization libraries that offer a range of plot and graphic kinds.

**Plotly:** is an open-source data visualization library that offers interactive and aesthetically pleasing visualizations.

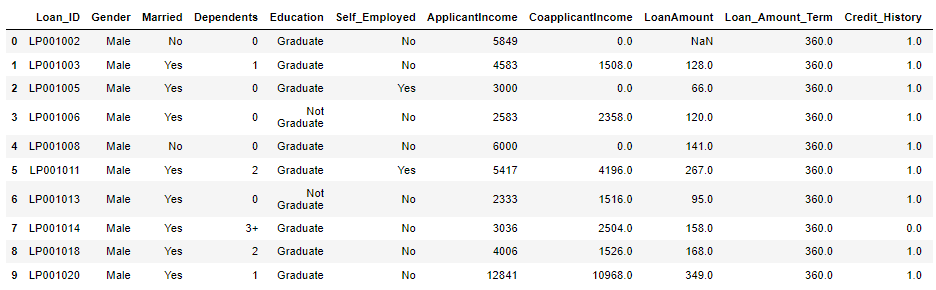
**Scikit-learn** : A machine learning library called Scikit-learn offers a number of tools for data preprocessing and modelling.

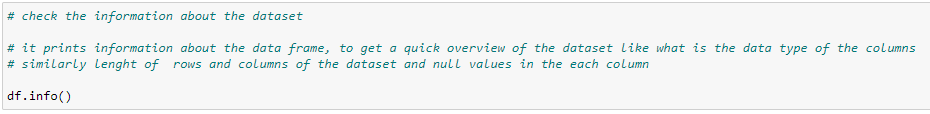
# 3. Import the Data

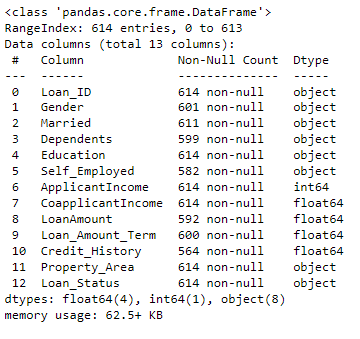
Once we have imported the necessary libraries, we can load the loan\_data.csv file, which contains information about loan applications This dataset contains information about the applicant's income, the amount of the loan, their credit history, and other elements that could influence the loan's approval. This dataset will be used to create our prediction model for loan approval. To get the data ready for modelling, we'll do some data cleaning and investigation in the following steps.

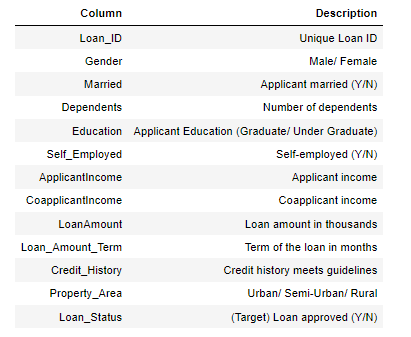


Data Understanding



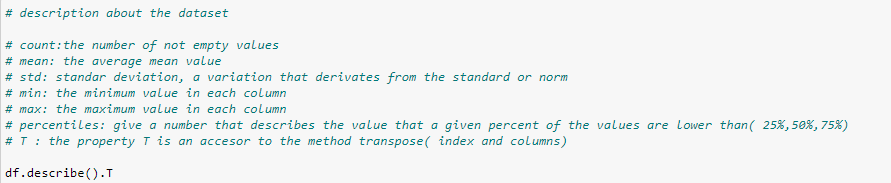


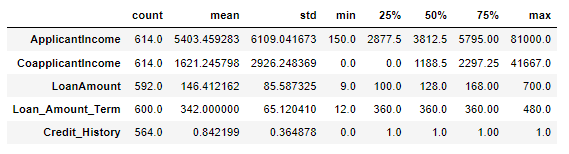




* The dataset for predicting loan approval has 614 entries and 13 columns.
* Loan\_ID, Gender, Married, Dependents, Education, Self-Employed, ApplicantIncome, CoapplicantIncome, LoanAmount, Loan\_Amount\_Term, Credit\_History, Property\_Area, and Loan\_Status are the names of the columns.
* Each column contains a variable amount of non-null entries,
* Each column has a unique Dtype. Eight columns have the dtype object, four columns have the dtype float64, one column has the dtype int64.
* The dataset takes up 62.5+ KB of RAM.

# 4. Answering Research Questions - Descriptive Statistics





* The dataset contains information on 614 loan applicants, with details such as their income, loan amount, loan term, credit history, etc.
* The minimum income of an applicant is 150, while the maximum is 81,000.
* The co-applicant income has a mean of 1,621.25 and a standard deviation of 2,926.25, with a minimum of 0 and a maximum of 41,667.
* The loan amount has a mean of 146.41 and a standard deviation of 85.59, with a minimum of 9 and a maximum of 700.
* The loan amount term has a mean of 342 and a standard deviation of 65.12, with a minimum of 12 and a maximum of 480.
* The credit history has a mean of 0.84 and a standard deviation of 0.36, with a minimum of 0 and a maximum of 1.

## 4.1 Data pre-processing

Data Preparation

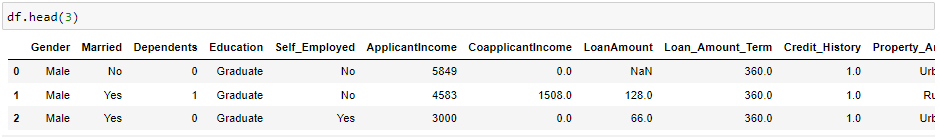
#### 4.1.1 Drop Irrelevant Columns

One frequent data cleaning method to remove extraneous data and streamline the dataset is to drop unneeded columns. Additionally, it can aid in lowering memory usage and enhancing model performance.

In Python, we can use the drop() function from pandas to remove the "Loan\_ID" column:



Here, 'Loan\_ID' refers to the column name, axis=1 indicates that we want to drop the column and not a row, and inplace=True updates the dataframe in place rather than creating a new copy.

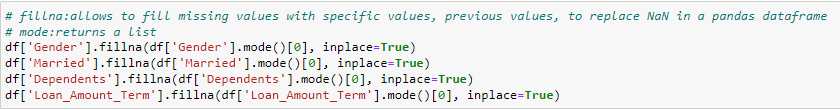


#### **4.1.2 Fill Missing Values**

Using the isnull().sum() method, count the number of missing values in each column as the first step. This will provide us with a total of the columns' missing values.



* Depending on the kind of the column, we employ various approaches to fill in the missing values. For instance, we use the mode to fill in the missing values for the "Gender", "Married", Loan\_Amount\_Term, and "Dependents" columns. The value that appears the most frequently in a data collection is called the mode, which is a statistical measure of central tendency.
* Similar to this, we assume that if the "Self\_Employed" column has no data, the person is not self-employed and fill in the blank values with "No".
* We use the median to fill in the missing values for the "LoanAmount" column. When a dataset contains outliers or extreme values, the median provides the middle value within the data set, making it a useful indicator of central tendency.
* Finally, we assume that if there are any missing values in the "Credit\_History" column, the person's credit history does not satisfy the company's requirements and replace them with 0.



In this part, we'll use the pandas fillna() function to replace the missing values in the 'Gender', 'Married', 'Dependents', and 'Loan\_Amount\_Term' columns with the corresponding columns' modes. The most frequent value or item in a dataset is represented by the mode, a statistical metric.

The fillna() function allows us to replace missing values in the supplied columns with the value that appears the most frequently in that column. Instead of making a copy, the original dataframe is altered when the inplace=True parameter is used.

Maintaining the distribution of the data in the column and reducing the impact of missing values on our analysis can both be accomplished by filling in the missing values with the mode.



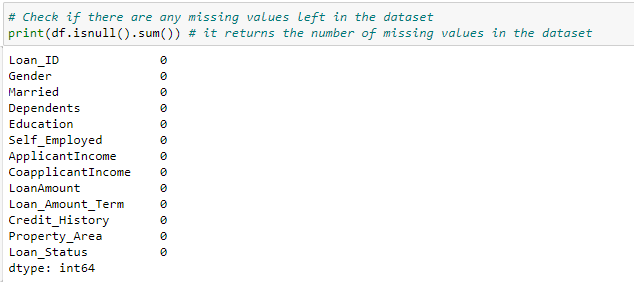
The 'fillna' method of the pandas library is being used in this phase to replace any missing values in the dataframe's 'Self\_Employed' column with the word 'No'. If the 'Self\_Employed' data is absent, we erroneously conclude that the subject is not self-employed and enter 'No' instead. When we can make a solid assumption about the values that are missing, this is a standard way to handle the missing data. Instead of making a copy of the original dataframe, the 'inplace=True' option modifies it.



In this step, we use the column's median value to fill in the missing values in the "LoanAmount" column of the pandas dataframe. Here, the missing values in the column are estimated using the median, a statistical measure that depicts the middle value of a dataset. The NaN (missing) values in the column are replaced with the median value using pandas' "fillna" method.

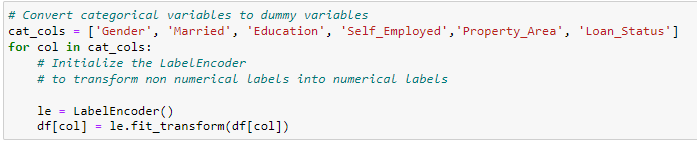


In this step, we use the fillna() method to replace the blank entries in the "Credit\_History" column with the value 0. This is due to the fact that a person's credit history will not be approved for a loan if it does not fulfil the company's requirements. Therefore, it makes sense to assume that missing values are equal to 0.



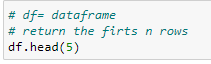
#### **Now you can see there is no single missing value in the dataset.**

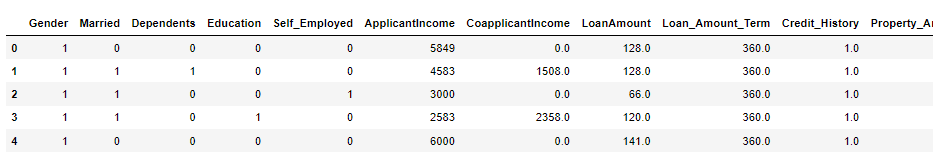
#### 4.1.3 Encode categorical variables



The category variables are being transformed into numerical values in this step. Variables with discrete values such as "Gender," "Married," "Education," etc. are categorical variables. Since most machine learning algorithms only work with numerical data, these variables must be transformed into numerical values.

The scikit-learn library's LabelEncoder is used to transform categorical data into numerical values. Each category in the variable is given a distinct integer value by the LabelEncoder.





The first five rows of the changed dataset are shown using the head(5) function.

#### 4.1.3 Check the outliers

#### Outliers

**Formula for removing outliers using z-score**:

The number of standard deviations from the mean is depicted statistically by the z-score. We can compute the z-score for each data point using the method below in order to eliminate outliers from a dataset:

z = (x - μ) / σ

**where**:

* z is the z-score
* x is the data point
* μ is the mean of the dataset
* σ is the standard deviation of the dataset

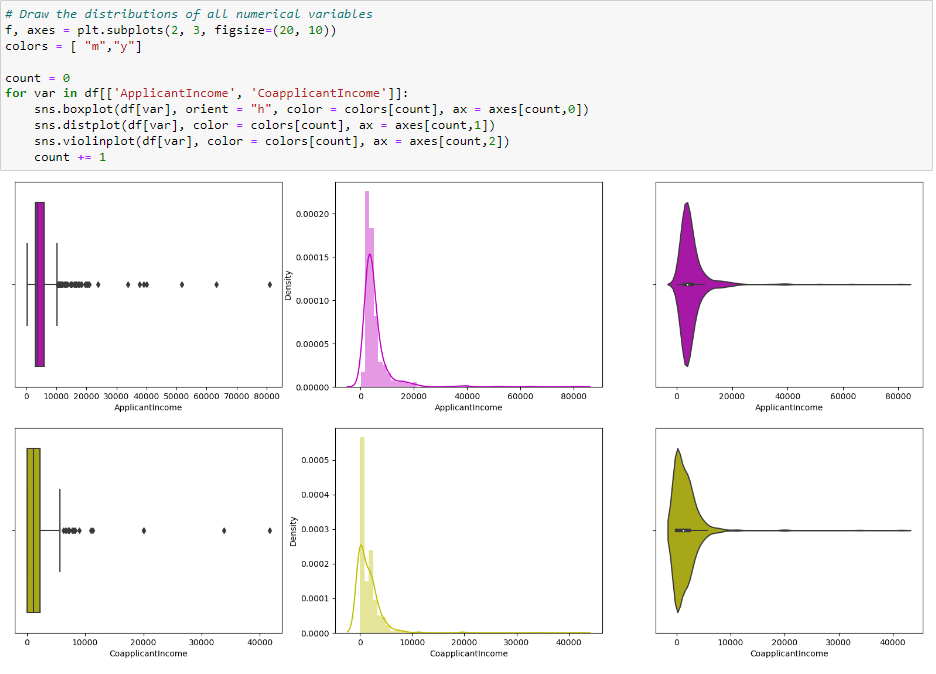
Any data point with a z-score greater than 3 or lower than -3 can then be eliminated. Extreme outliers, which are extremely uncommon and highly unlikely to happen by chance, are often identified using this criteria.

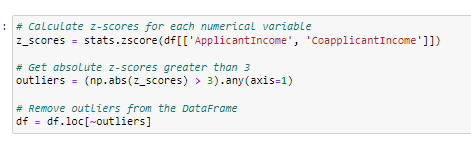
Logic for removing outliers using z-score:

* Calculate the mean (μ) and standard deviation (σ) of the dataset.
* Calculate the z-score for each data point using the formula: z = (x - μ) / σ
* Identify any data points with a z-score greater than 3 or less than -3.
* Remove the identified outliers from the dataset.

The logic behind removing outliers using z-score is that any data point with a z-score greater than 3 or less than -3 is considered an extreme outlier and is very unlikely to occur by chance. Therefore, removing these extreme outliers can help to improve the accuracy and reliability of statistical analysis and machine learning algorithms. However, it's important to note that removing too many outliers can also result in biased results and should be done with caution.

**Figure 1. Visualizing Outliers**

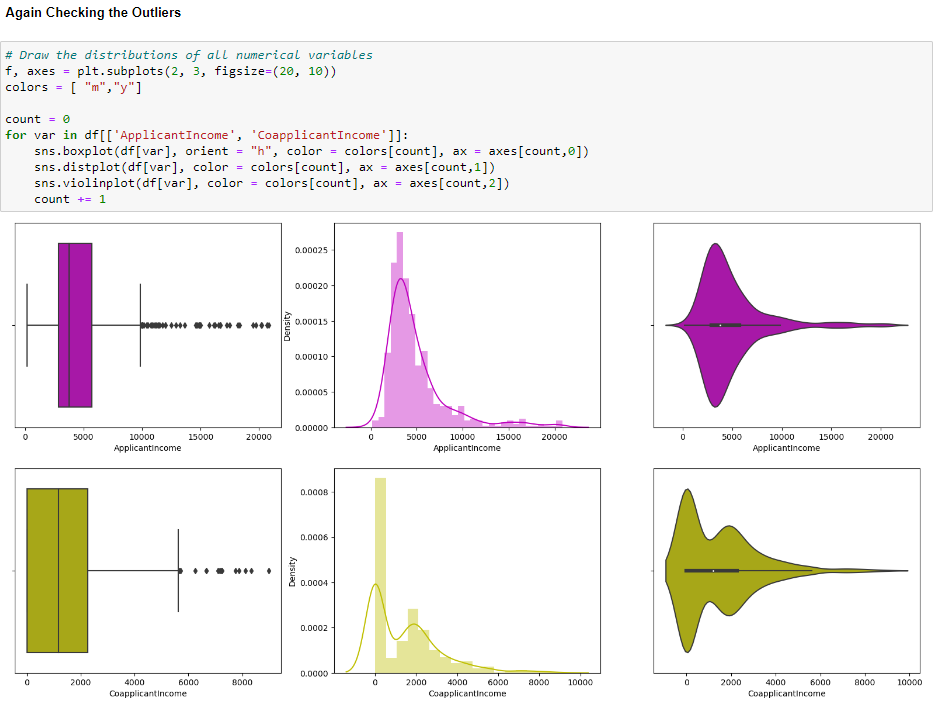




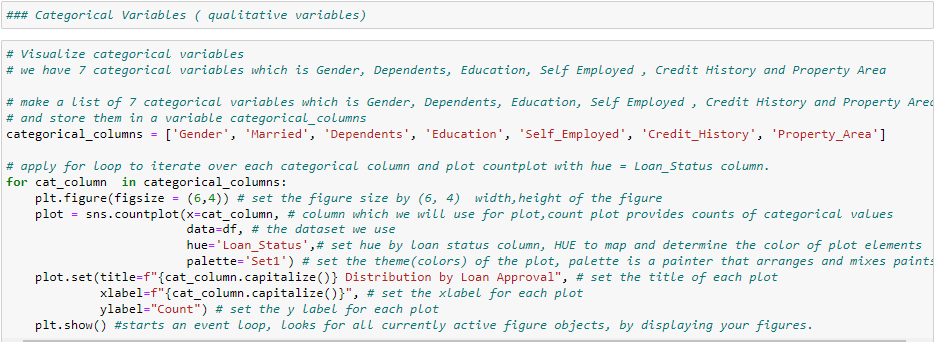
The 'ApplicantIncome' and 'CoapplicantIncome' columns of the dataframe are checked by the code block for outliers. Data points known as outliers are very distinct from other observations in the dataset and can drastically affect the study.

First, boxplots, density plots, and violin plots are used to visualise the distribution of the two columns. Then, it determines any observations with z-scores greater than 3 (signifying extreme outliers), which are considered extreme outliers, by calculating the z-scores for the two columns. In order to see the impact of the outlier removal, it eliminates the outliers from the dataframe and shows the distribution of the two columns once more.

**Figure 2. Visualizing Outliers**

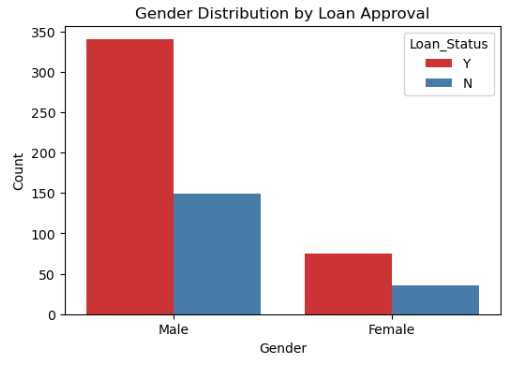


## [**4.2**](https://michael-fuchs-python.netlify.app/2020/08/21/the-data-science-process-crisp-dm/#research-question-1) **EDA of Categorical Variables (qualitative variables)**



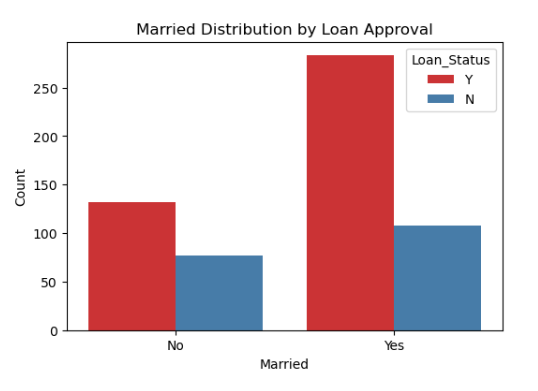
Data visualisation for categorical variables in a dataset is carried out by the provided code. It takes a list of categorical columns, iterates over each column, and then produces a countplot with the colour of the "Loan\_Status" column for each column. The hue defines the colour of the plot parts based on loan approval, and the countplot displays the number of each category in the column.

**Figure 4. Visualizing Gender Distribution by Loan Approval**



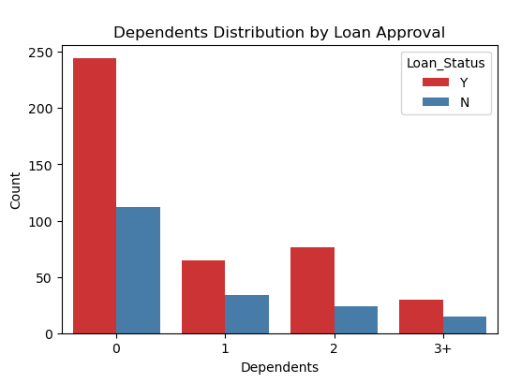
**Gender: The majority of applicants are men, and acceptance rates for male and female applicants are comparable.**

**Figure 5. Visualizing Married Distribution by Loan Approval**



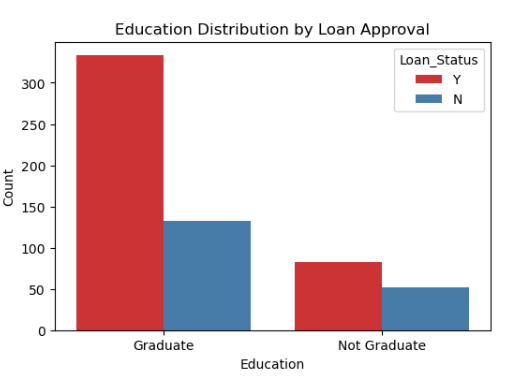
• **Married: Married applicants have a better acceptance rate and make up the majority of applicants.**

**Figure 6. Visualizing Dependents Distribution by Loan Approval**



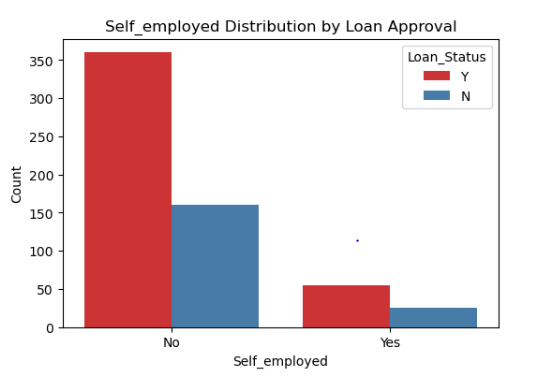
**Dependents: The majority of applicants don't have any children, and those who do tend to get approved more often.**

**Figure 7. Visualizing Education Distribution by Loan Approval**



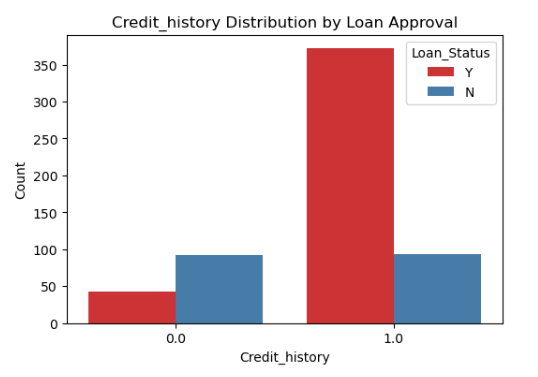
**Education: Graduates are more likely to be accepted than non-graduates, and they make up the majority of applicants.**

**Figure 8. Visualizing Self Employed Distribution by Loan Approval**



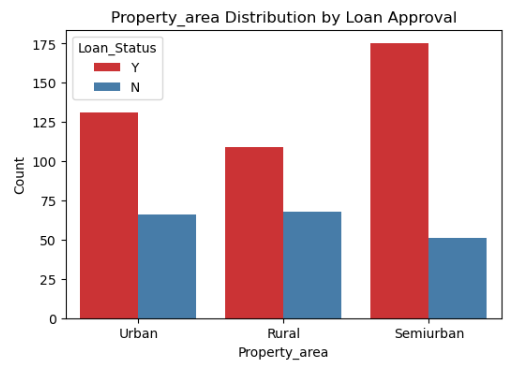
**Self-Employed: The majority of applicants are not independent contractors, and those who identify as self-employed have a little higher acceptance rate.**

**Figure 9. Visualizing Credit History Distribution by Loan Approval**



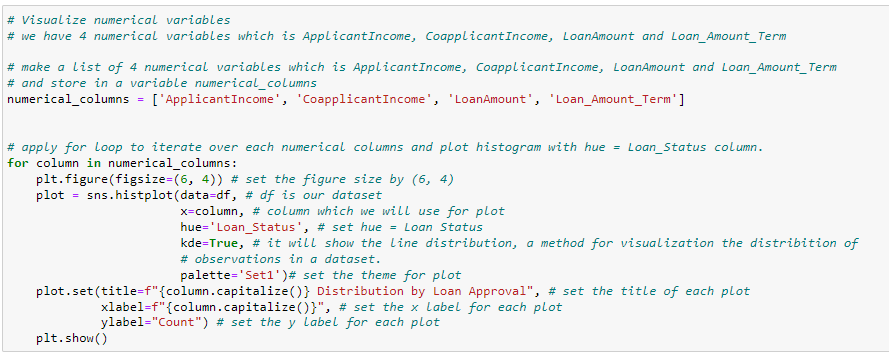
**Credit\_History: Applicants with a credit history of 1 have a greater acceptance rate, and the majority of applicants have credit histories of 1.**

**Figure 10. Visualizing Property Area Distribution by Loan Approval**



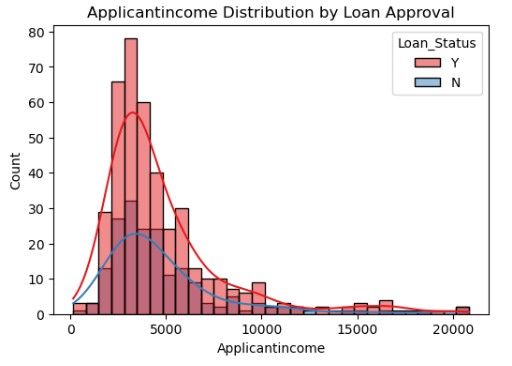
**Property\_Area: Semi-urban areas, followed by urban and rural areas, account for the majority of applicants. Semi-urban applicants are more likely to be accepted**.

## [**4.3**](https://michael-fuchs-python.netlify.app/2020/08/21/the-data-science-process-crisp-dm/#research-question-1) **EDA of Numerical Variables**



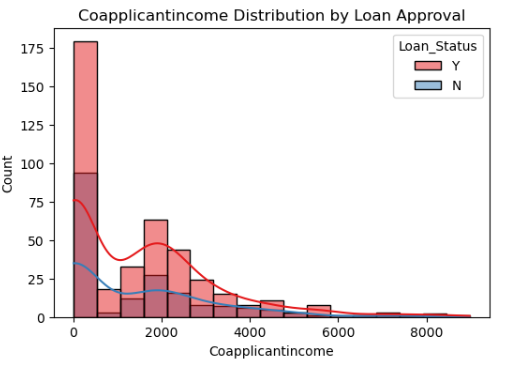
The provided code snippet employs histograms to display four numerical variables, including ApplicantIncome, CoapplicantIncome, LoanAmount, and Loan\_Amount\_Term. The code generates a histogram with a line distribution for each variable and sets the hue to Loan\_Status. Additionally defined are the figure size, title, x-label, y-label, and palette.

**Figure 11. Visualizing Applicant Income Distribution by Loan Approval**



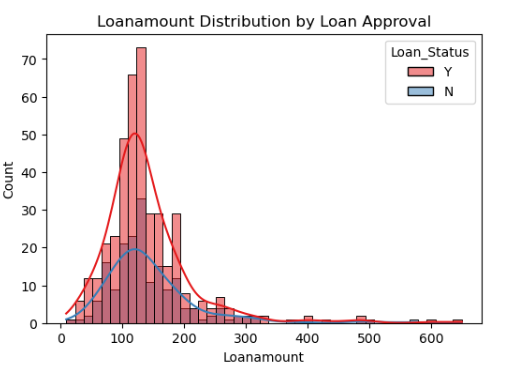
**ApplicantIncome: Since the distribution is right-skewed, the majority of applicants make less than 10,000 annually. For applicants with greater incomes, the approval rate is marginally higher.**

**Figure 12. Visualizing Co-Applicant Income Distribution by Loan Approval**



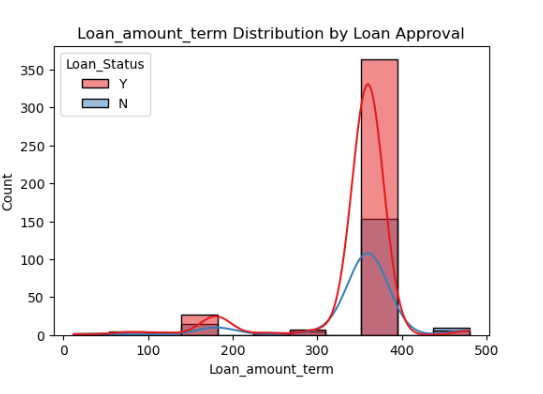
**CoapplicantIncome: The distribution is also skewed to the right, with the majority of coapplicants having incomes under $5,000. For applicants with lower co-applicant income, the approval rate is marginally higher.**

**Figure 13. Visualizing Loan Amount Distribution by Loan Approval**



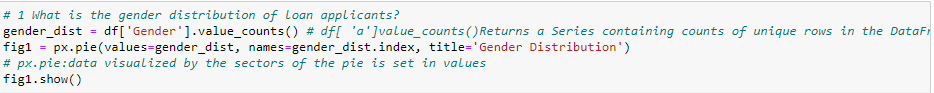
**LoanAmount: With the majority of loans falling between $100 and $200, the distribution is essentially standard. All loan amounts have a comparable approval rate.**

**Figure 14. Visualizing Married Distribution by Loan Approval**

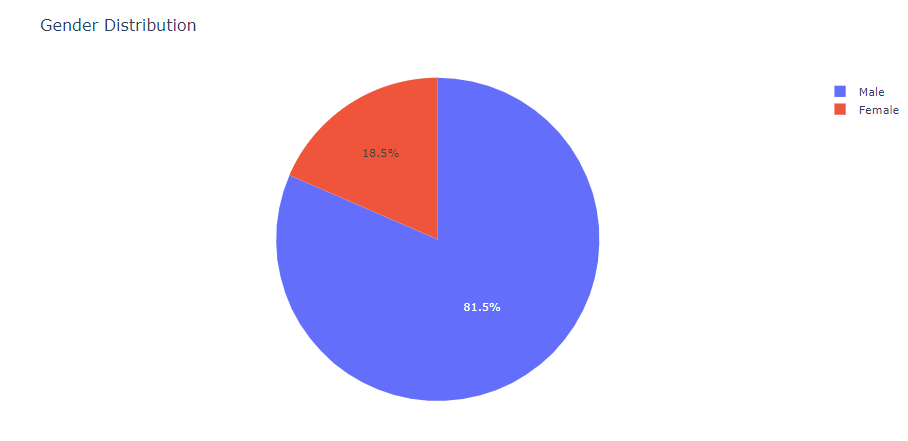


**The majority of loans have a period of 360 months (30 years) and a loan amount of up to $100,000.**

## [4.4](https://michael-fuchs-python.netlify.app/2020/08/21/the-data-science-process-crisp-dm/#research-question-1) What is the gender distribution of loan applicants

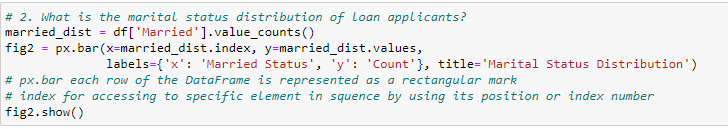


**Figure 15. Visualizing Gender Distribution**

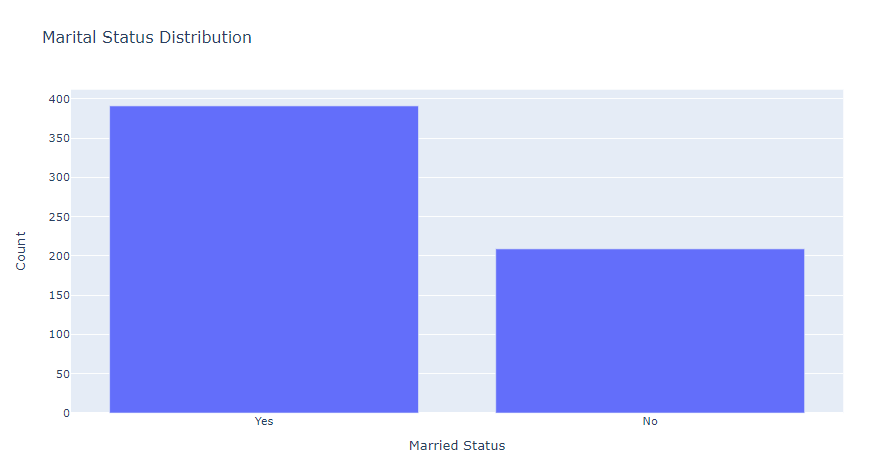


The code is utilized to display a dataset's gender breakdown of loan applicants. The DataFrame's 'Gender' column has a number of distinct values, which are counted using the value\_counts() method. The gender distribution count and gender distribution index are used as the values and labels, respectively, in a pie chart made using the px.pie() method. According to the ensuing pie chart, men make up 81.5% of loan applications while women make up 18.5%.

## [**4.5**](https://michael-fuchs-python.netlify.app/2020/08/21/the-data-science-process-crisp-dm/#research-question-1) **What is the marital status distribution of loan applicants?**

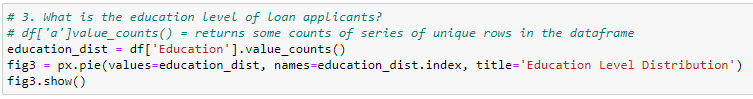


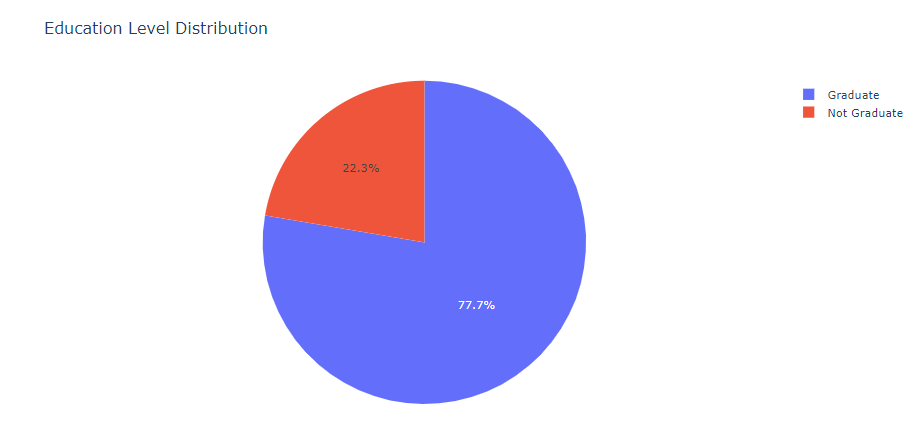
**Figure 16. Visualizing Marital Status Distribution**



The visualization displays how married or single loan applicants are distributed. A greater percentage of candidates are married, whereas a lesser proportion are single.

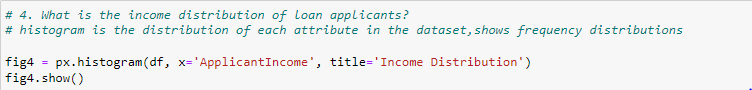
## [**4.6**](https://michael-fuchs-python.netlify.app/2020/08/21/the-data-science-process-crisp-dm/#research-question-1) What is the education level of loan applicants?

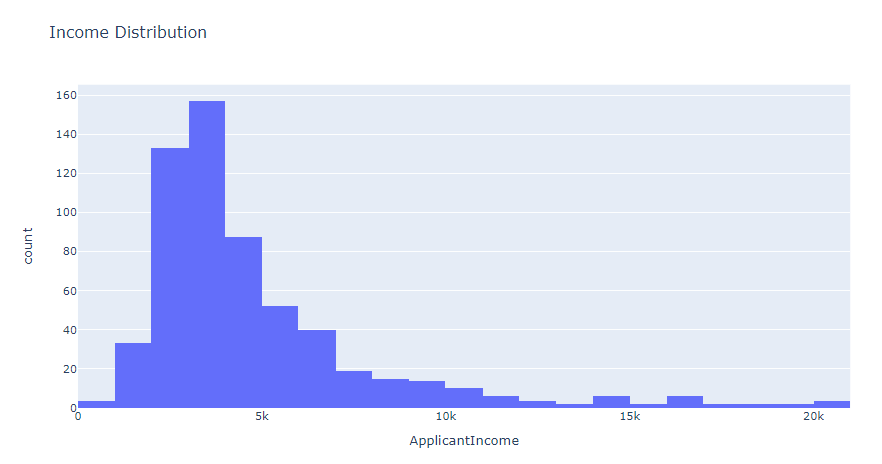


**Figure 17. Visualizing Gender Distribution**

Using the Plotly Express module, the code above computes the distribution of loan applicants' educational levels and generates a pie chart. A graduate degree is required for the majority of loan applicants, who make up around 77.7% of the total. The remaining 22.3% of loan applicants lack a graduate degree. This data can be utilized to make choices about loan approval and to better understand the educational history of loan applicants.

## 4.7 What is the income distribution of loan applicants?

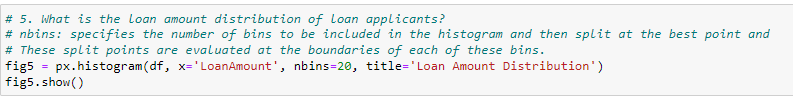


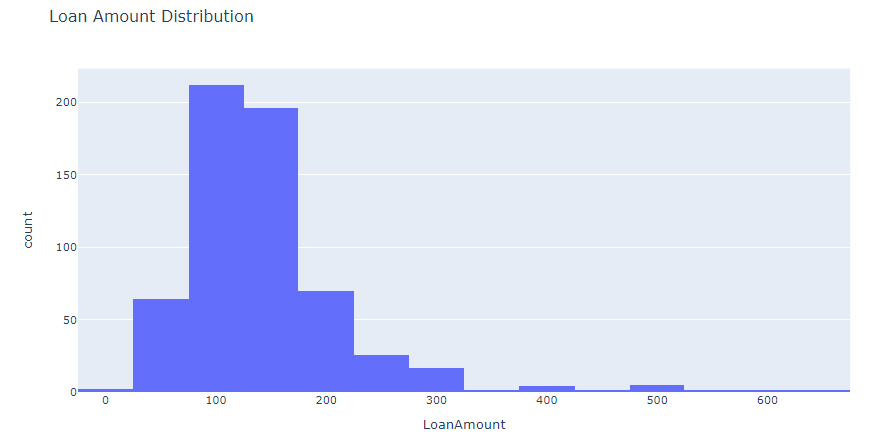


**Figure 18. Visualizing Income Distribution**

The income distribution of the loan application is displayed in this visualization. The y-axis displays the number or frequency of applications falling within each category, and the x-axis displays the income ranges. The histogram reveals that while there are a few outliers with high salaries, the majority of applicants have lower incomes. The distribution is positively skewed, which suggests that the majority of applicants earn less than the outliers.

## 4.8 What is the loan amount distribution of loan applicants?

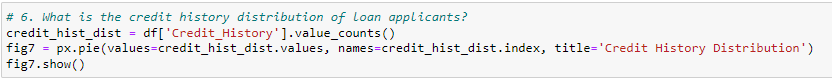


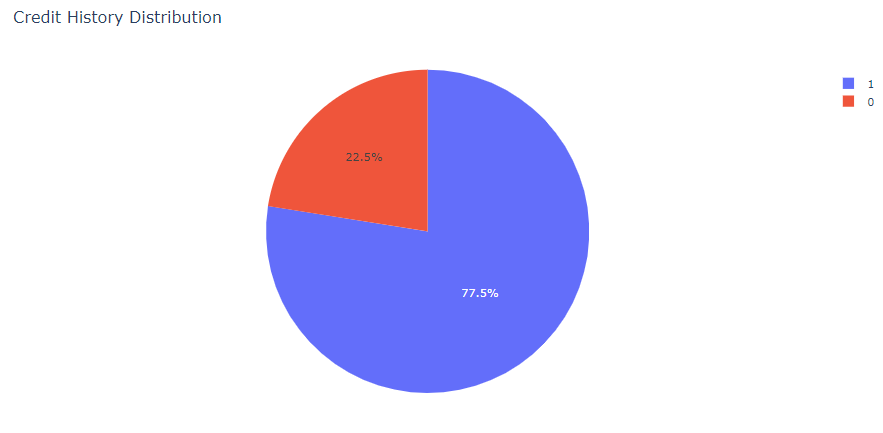


**Figure 19. Visualizing Loan Amount Distribution**

A few applicants requested high loan amounts, but the majority of loan amounts are concentrated in the lower range, according to the histogram of the loan amount distribution. Given that the distribution is favorably skewed, the bulk of loans fall below the median amount. The visualization aids in identifying the common loan amount range among loan applicants and draws attention to any loan amounts that fall outside of this range.

## 4.9 What is the credit history distribution of loan applicants?

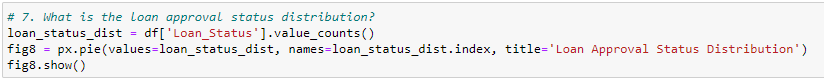


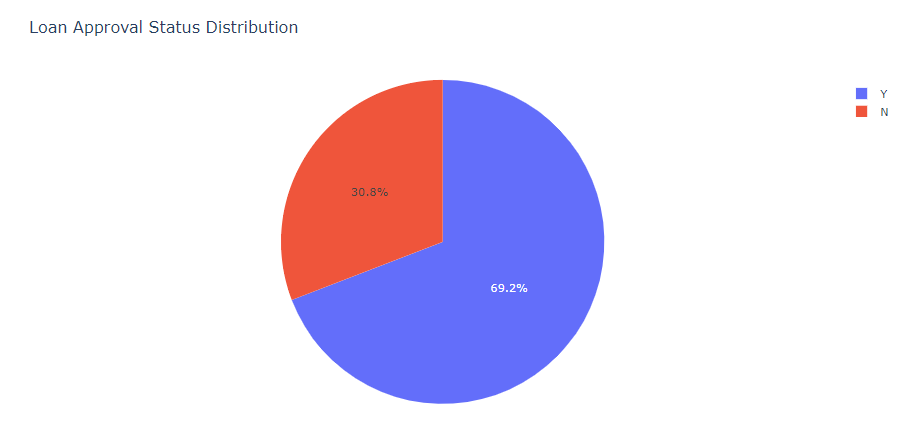


**Figure 20. Visualizing Credit History Distribution**

The distribution of credit histories reveals that the majority of loan applicants have credit histories of 1, which indicates that they have paid off their prior debts. This is a crucial consideration for lenders as it shows how likely it is that the applicant will pay back their loan.

## 4.10 What is the loan approval status distribution?

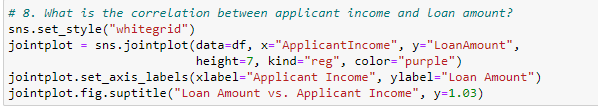


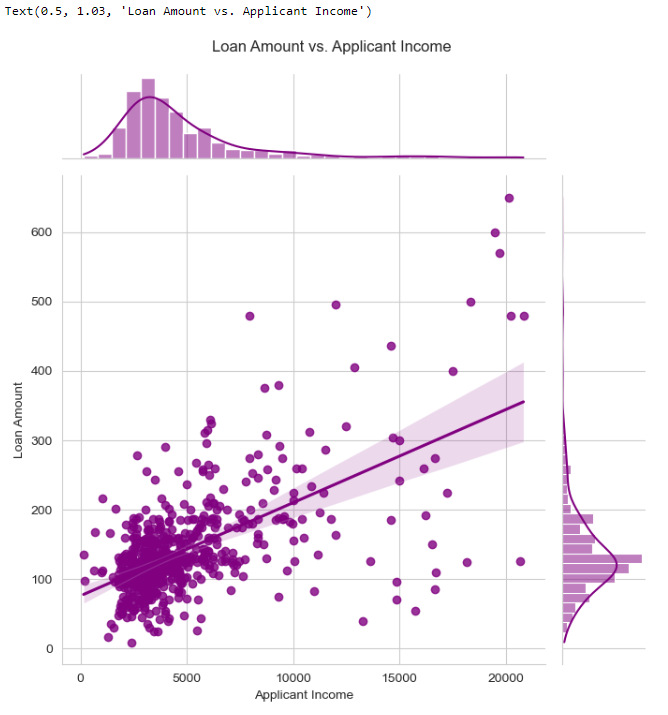


**Figure 21. Visualizing Loan Approval Status Distribution**

The analysis of the pie chart reveals an uneven distribution of loan approval statuses. Few loan applications were turned down, and most loan applicants had their requests approved. This can mean that the loan approval procedure is less stringent than it might be, or it might mean that the majority of loan applicants are financially secure and satisfy the standards for receiving a loan.

## 4.11 What is the income distribution of loan applicants?





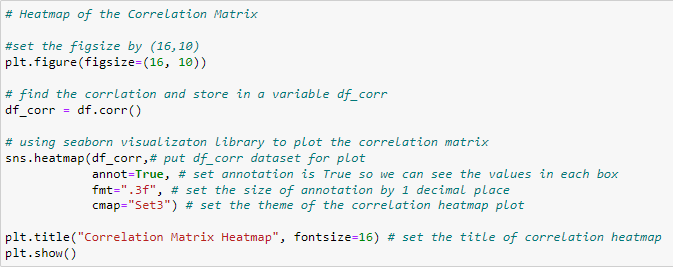
**Figure 22. Visualizing Correlation between Loan Amount and Applicant Income**

The scatterplot with the regression line shows that the loan amount tends to increase along with the applicant's income. The degree and direction of the linear link between the two variables are measured using the correlation coefficient. If the correlation coefficient is positive, there is a positive association between the two variables, meaning that if one variable rises, the other also tends to rise.

# 5 Development of a Machine Learning Model End to End Process

Modelling

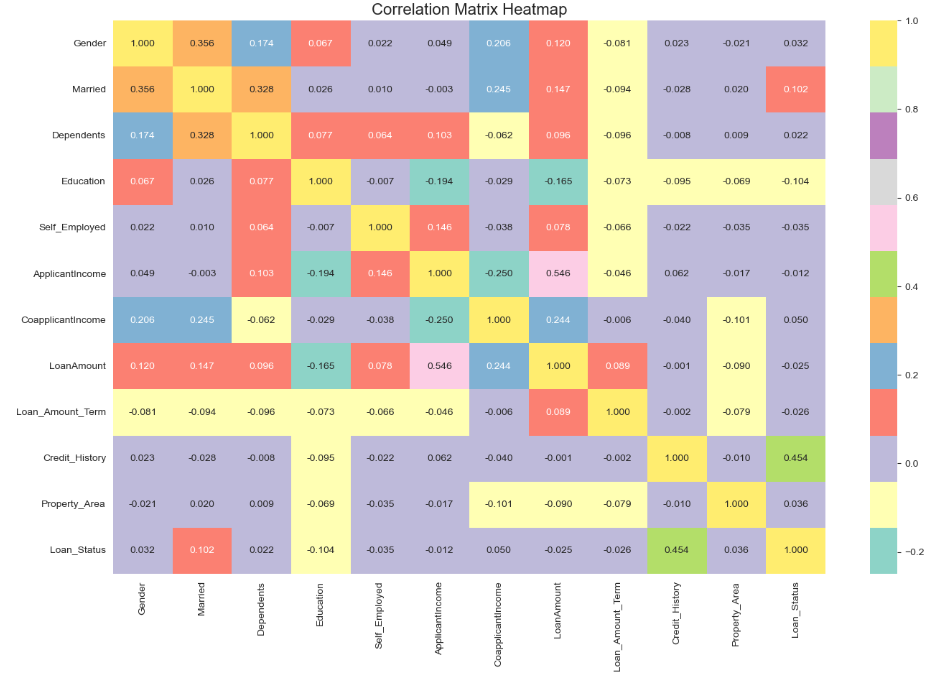
## 5.1 Correlation Matrix Heatmap



To determine the correlation between various columns in the dataset, the code generates a correlation matrix heatmap using the df.corr() method. The seaborn visualisation library is used to plot the heatmap. The correlation heatmap makes it easier to see how the input features and the target variable are related.

#### Correlation Matrix

To find patterns and connections between variables in a dataset, exploratory data analysis (EDA) frequently uses the correlation matrix. It can aid in feature selection, dimensionality reduction, and the detection of multicollinearity—a situation in which two or more variables exhibit strong correlations with one another.



**Figure 23. Visualizing Correlation between all variables**

There are a number of variables in our dataset, including Loan\_Status, Dependents, Gender, Married, Education, Self-Employed, ApplicantIncome, CoapplicantIncome, LoanAmount, Loan\_Amount\_Term, and Property\_Area. To investigate the connections between these variables, we can construct a correlation matrix.

As we can see from the heatmap, some of the attributes have a strong association with the target variable (Loan\_Status), while others have a weak or nonexistent correlation. Features like Dependents, Gender, Property area, and Married show essentially no correlation with the goal variable, while features like CoapplicantIncome, Education, LoanAmount, Loanamountterm, applicant income, and self-employed have negative relationships with the target variable.

Therefore, As a result, Credit\_History will be the input feature and Loan\_Status will be the target variable for our ML model.

## 5.2 Separate the variables into feature matrix (X) and target variable (y)



In this code snippet, the variables are being separated into a feature matrix (X) and a target variable (y). The feature matrix (X) is a subset of the original dataframe that contains the feature(s) that will be used to predict the target variable (y). Here, the feature is "Credit\_History" which is the credit history of the loan applicant.

The target variable (y) is the variable we are trying to predict, which is "Loan\_Status".



To assess a model's performance in machine learning, it's crucial to divide the data into training and testing sets. This code snippet splits the data into a feature matrix (X) and target variable (y) for loan approval status using the train\_test\_split() method from the scikit-learn library. Since the test size is set to 20% of the data, 80% of the data are used to train the model and 20% are used to assess how well it performs. The stratify parameter makes sure that the classes in the training and testing sets of the target variable (Loan Status) are balanced. The random\_state option is set to 0 to ensure that the findings can be replicated.

## 5.3 Scaling Using Standard Scaler

#### Standard Scaler

To standardise or normalise the scale of numerical characteristics or variables in a dataset, the Standard Scaler approach is used. The data is rescaled to have a 0 mean and 1 standard deviation. Machine learning algorithms that are sensitive to the magnitude of the input variables frequently employ this strategy.

Formula:

The formula for Standard Scaler is:

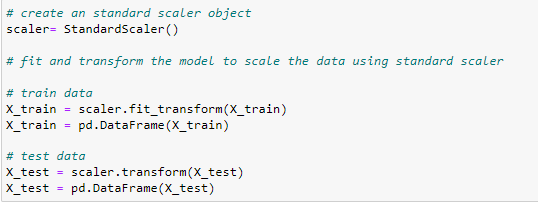
**z = (x - mean) / standard deviation**

where:

* z is the standardized value of the variable x
* x is the original value of the variable

We first determine the mean and standard deviation for each numerical feature in the dataset before applying Standard Scaler to convert each value of that feature into a standardized value. Python machine learning packages like scikit-learn make it simple to apply this approach.

Standard Scaler is a straightforward yet effective method that can enhance the efficiency of machine learning algorithms, particularly those relying on gradient descent optimisation or distance measures. By preventing the dominance of larger scale variables over smaller scale variables, it helps to ensure that all variables contribute evenly to the model.



Scaling is the process of altering the data in machine learning so that it can be more easily compared and the differences between the features can be seen more clearly. A well-liked technique for scaling data in machine learning is StandardScaler. The data is standardised to have a mean of 0 and a standard deviation of 1.

In the given code snippet, an object of the StandardScaler is created, and it is then used to fit and transform the training data. The same scaler is then used to transform the testing data. The fit\_transform() method scales the data and fits the scaler to the data at the same time. After scaling, the training and testing data are converted into Pandas DataFrame. The StandardScaler is applied to only the input feature matrix X, which is the Credit\_History column in this case.

## 5.4 Logistic Regression

#### Logistic Regression

Logistic regression is a supervised machine learning algorithm that is used for binary classification problems. It is a sort of regression analysis where the outcome (or dependent variable) is categorical in nature and is used to forecast the likelihood that an event will occur, such as the likelihood that a consumer will make a purchase or a patient would receive a diagnosis of an illness. In order to convert the results of a linear regression model into a number between 0 and 1, which represents the likelihood that an event will occur, logistic regression is used.



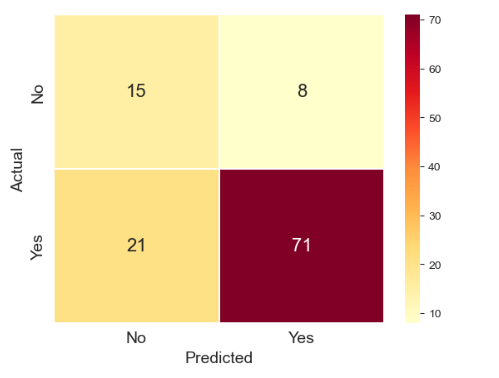
A classification approach called logistic regression is used to predict binary target variables like loan approval status. This code snippet uses evaluation metrics like accuracy, confusion matrix, precision, recall, and f1-score to train and evaluate the logistic regression algorithm on loan data**.**

Evaluation

#### Confusion Matrix

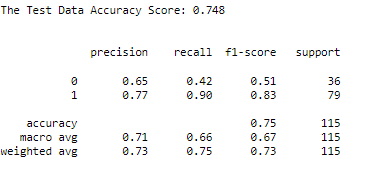
A confusion matrix is a table that displays the quantity of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) for a binary classification model.

* True Positive (TP): The prediction of the positive class by the model was accurate.
* True Negative (TN): The negative class was successfully predicted by the model.
* False Positive (FP): When the class was actually negative, the model incorrectly predicted the positive class.
* False Negative (FN): When the class was actually positive, the model predicted the negative class.



**Figure 24. Visualizing Confusion Matrix of the Model**

The confusion matrix demonstrates that for the majority of the test dataset, the model correctly anticipated the loan approval outcome. While recall measures the proportion of true positives to all actual positives, precision measures the ratio of true positives to all positive predictions. The harmonic mean of precision and recall is the f1-score.



#### Precision:

A performance indicator called precision counts the proportion of occurrences that are relevant out of all the instances that are classed as positive. In other words, precision assesses the reliability of the optimistic predictions.

Formula:

Precision = True Positives / (True Positives + False Positives)

#### Recall:

Recall is a performance statistic that quantifies the proportion of relevant occurrences out of all relevant instances that are successfully detected. Recall, then, assesses how accurately the classifier can isolate each successful case.

Formula:

Recall = True Positives / (True Positives + False Negatives)

#### F1 Score

A performance indicator called the F1 score combines recall and precision into one number. It is the precision and recall harmonic mean.

Formula:

F1 Score = 2 \* (Precision \* Recall) / (Precision + Recall)

#### Accuracy

A performance statistic called accuracy counts the proportion of accurate forecasts among all guesses. The ability of the classifier to categorise both positive and negative examples is measured by this metric.

Formula:

Accuracy = (True Positives + True Negatives) / (True Positives + False Positives + False Negatives + True Negatives)

These metrics are frequently used in classification tasks to assess how well a machine learning model is performing. They can aid in the selection and optimisation of models by offering information about how well a model is working.

#### Test Accurcay

The test data aacuracy is 0.748 which means that, the model correctly predicts that, loan approval outcome for 74.8% of the test dataset.

#### Precision, Recall and f1-score for class 0

The precision for class 0 (rejected loan) is 0.65, which means that, out of all the loan applications predicted as rejected, 65% were actually rejected, The recall for class 0 is 0.42, which means that out of all the loan application that were actually rejected, the model predicted 42% correctly, The f1=score is 0.51, which is the harmonic mean of precision and recall for claas 0.

#### Precision, Recall and f1-score for class 1

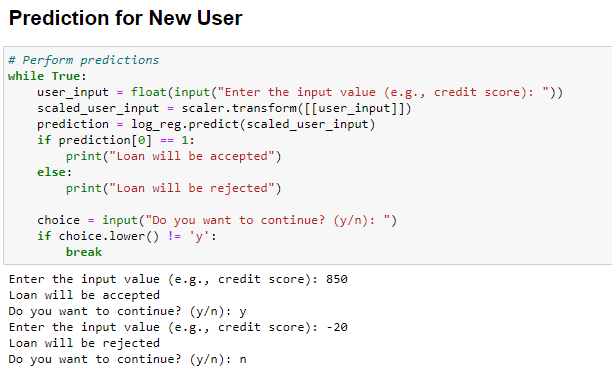
The precision for class 1 (accepted loan) is 0.77, which means that, out of all the loan applications predicted as accepted, 77% were actually accepted. The recall for class 1 is 0.90, which means that out of all the loan applications that were actually accepted, the model predicted 90% correctly. the f1-score is 0.83, which is the harmonic mean of precision and recall for class 1.

#### Support for Class 0 and 1

The support for class 0 is 36, which means that there were 36 loan applications that were actually rejected in the test dataset, and the support for class 1 is 79, which means that there were 79 loan that were actually accepted in the test dataset.

#### Confusion Matrix

The confusion matrix shows that out of 36 rejected loan applications, 15 were correctly predicted as rejected (true negative), and 21 were incorrectly predicted as accepted (false positive). Out of 80 accepted loan applications, 71 were correctly predicted as accepted (true positive), and 8 were incorrectly predicted as rejected (false negative).



1. The code begins with a while True loop, which allows the user to make multiple predictions until they choose to stop by entering a value other than 'y'.
2. Inside the loop, the user is prompted to enter an input value, such as a credit score, using the input() function. The value is then converted to a float using float() since the input function returns a string.
3. The user input is then scaled using the previously fitted scaler object. The scaler.transform() method takes the user input as a 2D array [[user\_input]] and returns the scaled value.
4. The scaled input value is passed to the log\_reg.predict() method to obtain the loan approval prediction. The predict() method predicts the class label (0 or 1) based on the scaled input.
5. The code checks the prediction value. If it is equal to 1, it prints "Loan will be accepted". Otherwise, it prints "Loan will be rejected".
6. After displaying the prediction, the user is asked whether they want to continue making predictions by entering 'y' or 'n'. The input() function is used to obtain the choice, and choice.lower() converts the choice to lowercase for easier comparison.
7. If the user does not enter 'y', the break statement is executed, which breaks out of the while loop, ending the prediction process.

This code allows the user to input a value (e.g., credit score) and predicts whether a loan will be accepted or rejected based on that input. The user can continue making predictions until they choose to stop.

Deployment

Deployment is the CRISP-DM's last stage. In this phase, we'll prepare the results so we can present them to the client and use them as input for their decision-making. To forecast whether fresh loan applications will be approved, we will utilize the model we have established.

In conclusion, we use a logistic regression model to predict whether a loan application will be granted or not using the CRISP-DM methodology. By doing this, we can make certain that our model accurately and successfully predicts loan approvals, lowering the risk of default and ensuring that loans are only offered to borrowers who have a greater chance of repaying them.

Reflection

**Carlos Santana Bobadilla**

1 **For Business Understanding**: I first decided to set the objectives, from a business perspective, second produce project plan, to achieve business goals, third inventory of resources like list the resources available for the project.

2 **For Data Understanding**: First l initiated by collecting. A list of the data sources acquired. Second data description report to evaluate that data satisfies our expectations.

3 **For Data Preparation**: First l selected my data for my analysis, second clean my data, To raise the data quality to the level required by the analysis techniques, third construct, The data preparation operations.

**Mark Christian Albinto**

During the whole process of working this project, we have faced several key phases that are vital in achieving accurate and reliable results. These steps are data preparation, model selection, input feature selection, splitting the dataset, scaling, modelling, and evaluating the performance using Confusion Matrix and Accuracy Score.

Data preparation is the most fundamental step in this project. It involves cleaning and pre-processing the data to make sure it’s suitable for analysis. The methods used includes handling missing values, dealing with outliers, and performing data transformation.

The model selection part was a crucial decision-making point during this project. It involved testing different algorithms and selecting the one that best suited for the problem. In our case, we used Logistic Regression as the classifier. Logistic Regression is a popular choice for binary classification. The decision to use Logistic Regression were the availability of a vast range of resources to study.

Input feature selection also played a significant role in optimizing the performance of the model. By carefully examining the features and identifying the most relevant ones for the task. By keeping relevant features and removing irrelevant ones like the ID, we improves the model's efficiency and not over fit it.

Scaling the data was also another important step to make sure that the features are within similar range, which prevented other variables from dominating the model's learning process due to their larger range.

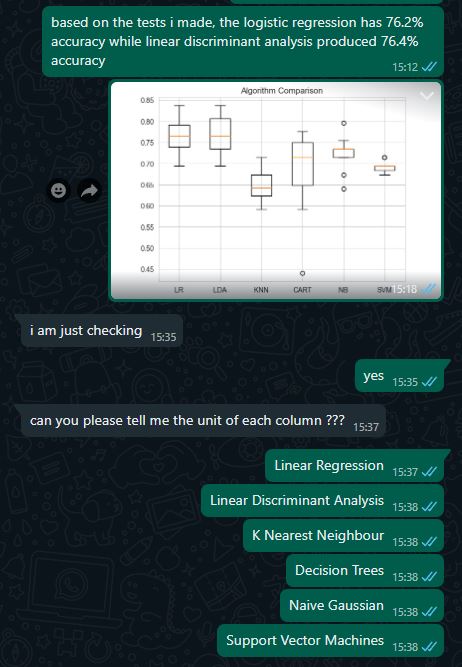
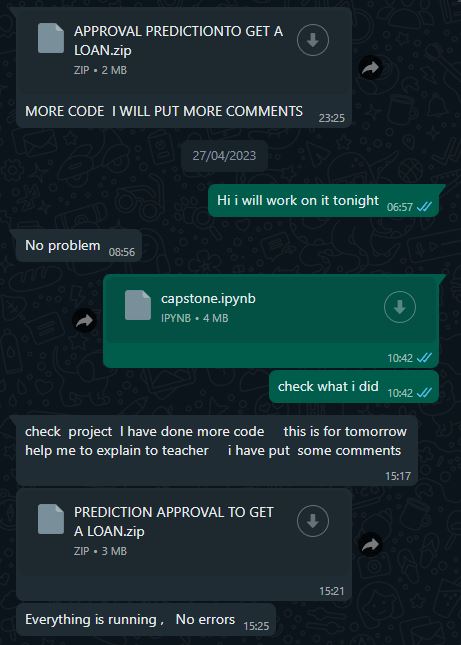
The modelling phase involves the implementation and training the model on the training dataset.

To evaluate the model's performance, we used Confusion Matrix and Accuracy Score. The Confusion Matrix provided information about the model's predictions, allowing us to assess its performance in terms of true positives, true negatives, false positives, and false negatives. By analyzing these metrics, we gained an understanding of the model’s correct and wrong predictions. The Accuracy Score taken from the Confusion Matrix, provided a single measure to identify the model's accuracy.

In conclusion, the steps mentioned above contributed a huge help to the success of our project. Through this experience, we have gained a deeper understanding of the importance of each step, which helped us to improve our ability to approach similar problems in the future.

Evidence of Group Work

Since we have different working time where I (Mark) work nights and he (Carlos) work days, we often find our schedules conflicting. We primarily use WhatsApp as communication and file sharing platform for our project. Below are some of the screenshots containing transfer of files and asking each other’s opinions regarding what we are working on at the moment.



# 6 References

“(PDF) the LOAN PREDICTION USING MACHINE LEARNING.” *ResearchGate*, www.researchgate.net/publication/357449126\_THE\_LOAN\_PREDICTION\_USING\_MACHINE\_LEARNING.

Agarwal, Sumit, and Itzhak Ben-David. *The Effects of Loan Officers’ Compensation on Loan Approval and Performance: Direct Evidence from a Corporate Experiment*. 2011.

“ANALYSIS and FORECASTING of BANK LOAN APPROVAL DATA USING MACHINE LEARNING ALGORITHMS.” *International Research Journal of Modernization in Engineering Technology and Science*, 15 Mar. 2023, <https://doi.org/10.56726/irjmets34316>. Accessed 6 May 2023.

Anand, Mayank, et al. “Prediction of Loan Behaviour with Machine Learning Models for Secure Banking.” *Journal of Computer Science and Engineering (JCSE)*, vol. 3, no. 1, 15 Feb. 2022, pp. 1–13, <https://doi.org/10.36596/jcse.v3i1.237>.

Bhargav, P., and P. Rama Parvathy L. “Comparing Random Forest with the Naive Bayes Algorithm with Improved Accuracy: An Effective Machine Learning Method for Loan Prediction.” *Journal of Survey in Fisheries Sciences*, vol. 10, no. 1S, 8 Mar. 2023, pp. 2018–2029, sifisheriessciences.com/journal/index.php/journal/article/view/436, <https://doi.org/10.17762/sfs.v10i1S.436>. Accessed 11 May 2023.

Boddepalli, Rajitha. “Loan Eligibility Criteria Using Machine Learning.” *International Journal of Research Publication and Reviews Journal Homepage: Www.ijrpr.com*, vol. 3, no. 11, 2022, pp. 407–410, ijrpr.com/uploads/V3ISSUE11/IJRPR7738.pdf. Accessed 11 May 2023.

Calcagnini, Giorgio, et al. “Hierarchy of Bank Loan Approval and Loan Performance.” *Economia Politica*, vol. 35, no. 3, 28 Mar. 2018, pp. 935–954, <https://doi.org/10.1007/s40888-018-0109-3>. Accessed 9 Dec. 2020.

“Classification and Estimation of High-Risk Factors to Low-Risk Factors in Approving Loan through Creditworthiness of Bank Customers Using SVM Algorithm and Analyze Its Performance over Logistic Regression in Terms of Accuracy.” *Journal of Pharmaceutical Negative Results*, vol. 13, no. SO4, 1 Jan. 2022, <https://doi.org/10.47750/pnr.2022.13.s04.212>. Accessed 7 Nov. 2022.

Fati, Suliman Mohamed, and Suliman Mohamed Fati. “Machine Learning-Based Prediction Model for Loan Status Approval.” *Journal of Hunan University Natural Sciences*, vol. 48, no. 10, 2021, jonuns.com/index.php/journal/article/view/783.

Gupta, Anshika, et al. “Bank Loan Prediction System Using Machine Learning.” *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*, 4 Dec. 2020, <https://doi.org/10.1109/smart50582.2020.9336801>.

Kulothungan, A., and Et Al. “Loan Forecast by Using Machine Learning.” *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 7, 19 Apr. 2021, pp. 894–900, www.turcomat.org/index.php/turkbilmat/article/view/2673, <https://doi.org/10.17762/turcomat.v12i7.2673>. Accessed 11 May 2023.

Kumar, Narra Rahul, and L. Rama Parvathy. “Higher Accuracy on Loan Eligibility Prediction Using Random Forest Algorithm over Decision Tree Algorithm.” *Baltic Journal of Law & Politics*, vol. 15, no. 4, 14 Dec. 2022, pp. 241–251, versita.com/menuscript/index.php/Versita/article/view/826. Accessed 11 May 2023.

“Loan Prediction Using Machine Learning Project Source Code.” *ProjectPro*, www.projectpro.io/article/loan-prediction-using-machine-learning-project-source-code/632.

Niu, Beibei, et al. “Credit Scoring Using Machine Learning by Combing Social Network Information: Evidence from Peer-To-Peer Lending.” *Information*, vol. 10, no. 12, 17 Dec. 2019, p. 397, <https://doi.org/10.3390/info10120397>.

Rahul, Tulluri, et al. “BANK LOAN PREDICTION USING MACHINE LEARNING.”

Serengil, Sefik Ilkin, et al. “A Comparative Study of Machine Learning Approaches for Non Performing Loan Prediction.” *IEEE Xplore*, 1 Sept. 2021, ieeexplore.ieee.org/document/9558894. Accessed 29 Apr. 2022.

Sheikh, Mohammad Ahmad, et al. “An Approach for Prediction of Loan Approval Using Machine Learning Algorithm.” *IEEE Xplore*, 1 July 2020, ieeexplore.ieee.org/abstract/document/9155614.

Shinde, Anant, et al. “Loan Prediction System Using Machine Learning.” *ITM Web of Conferences*, vol. 44, 2022, p. 03019, <https://doi.org/10.1051/itmconf/20224403019>.

Sreesouthry, S., et al. “Loan Prediction Using Logistic Regression in Machine Learning.” *Annals of the Romanian Society for Cell Biology*, 11 Apr. 2021, pp. 2790–2794, www.annalsofrscb.ro/index.php/journal/article/view/2818.

Sripriya, S. Varshaa Sai, et al. “Predictive Model to Compute Eligibility Test for Loans.” *IEEE Xplore*, 1 Oct. 2022, ieeexplore.ieee.org/abstract/document/9951727/. Accessed 11 May 2023.

# 