Title of the Project

Loan Status Prediction Using Python with Machine Learning

A

Report submitted

in partial fulfillment for the Degree of

MSCSC: M.Sc. COMPUTER SCIENCE

By

Ankit Kumar (CUSB2102312010)

Under the Guidance of

Mr. Rakesh Kumar Jha

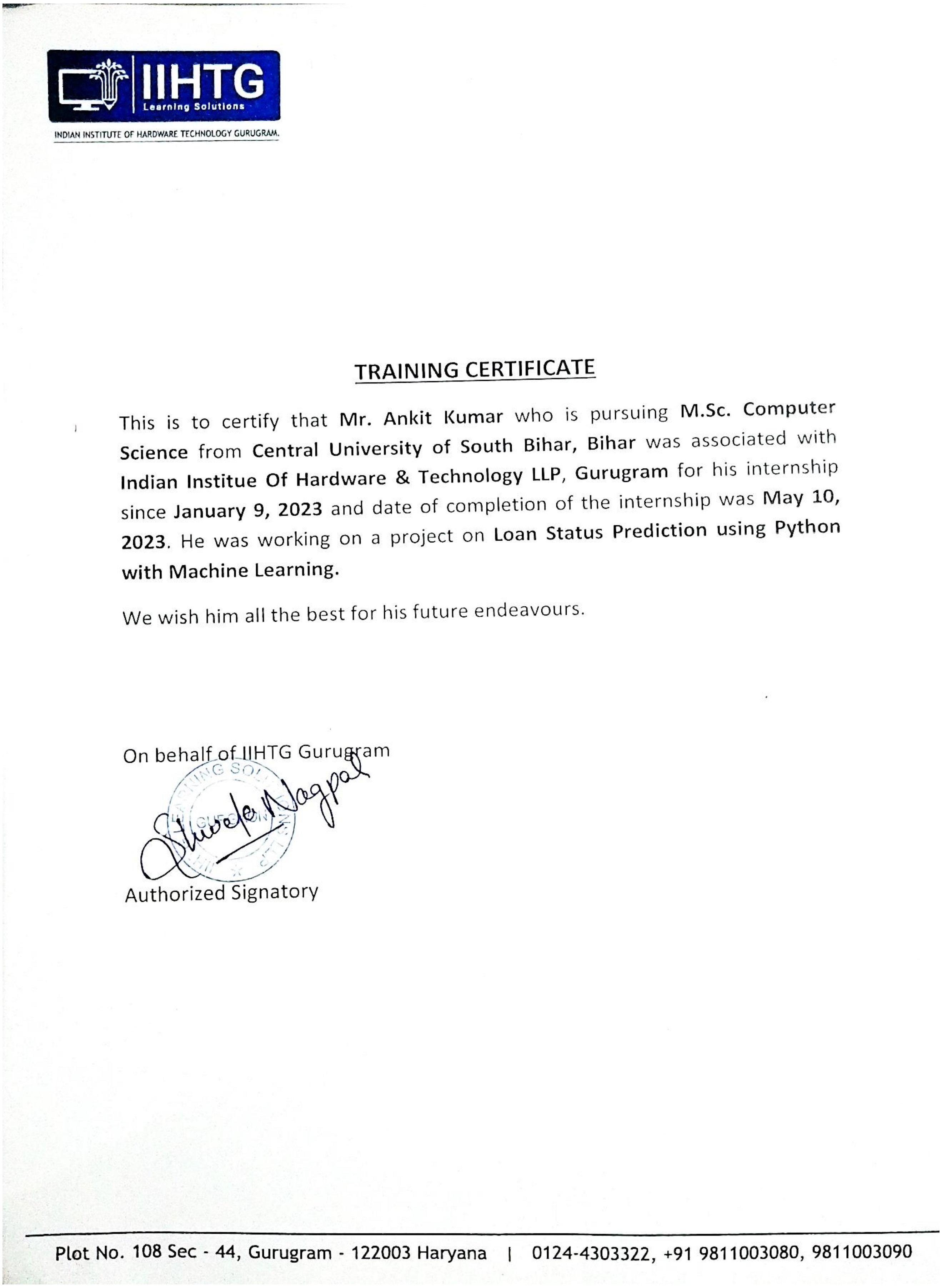
Indian Institute of Hardware & Technology- Gurgaon (Haryana)

To

Department of Computer Science

CENTRAL UNIVERSITY OF SOUTH BIHAR

PANCHANPUR- GAYA (BIHAR) - 824236



INDIAN INSTITUTE OF HARDWARE TECHNOLOGY GURUGRAM.

TRAINING CERTIFICATE

This is to certify that **Mr. Ankit Kumar** who is pursuing **M.Sc. Computer Science** from **Central University of South Bihar, Bihar** was associated with **Indian Institue Of Hardware & Technology LLP, Gurugram** for his internship since **January 9, 2023** and date of completion of the internship was **May 10, 2023**. He was working on a project on **Loan Status Prediction using Python with Machine Learning.**

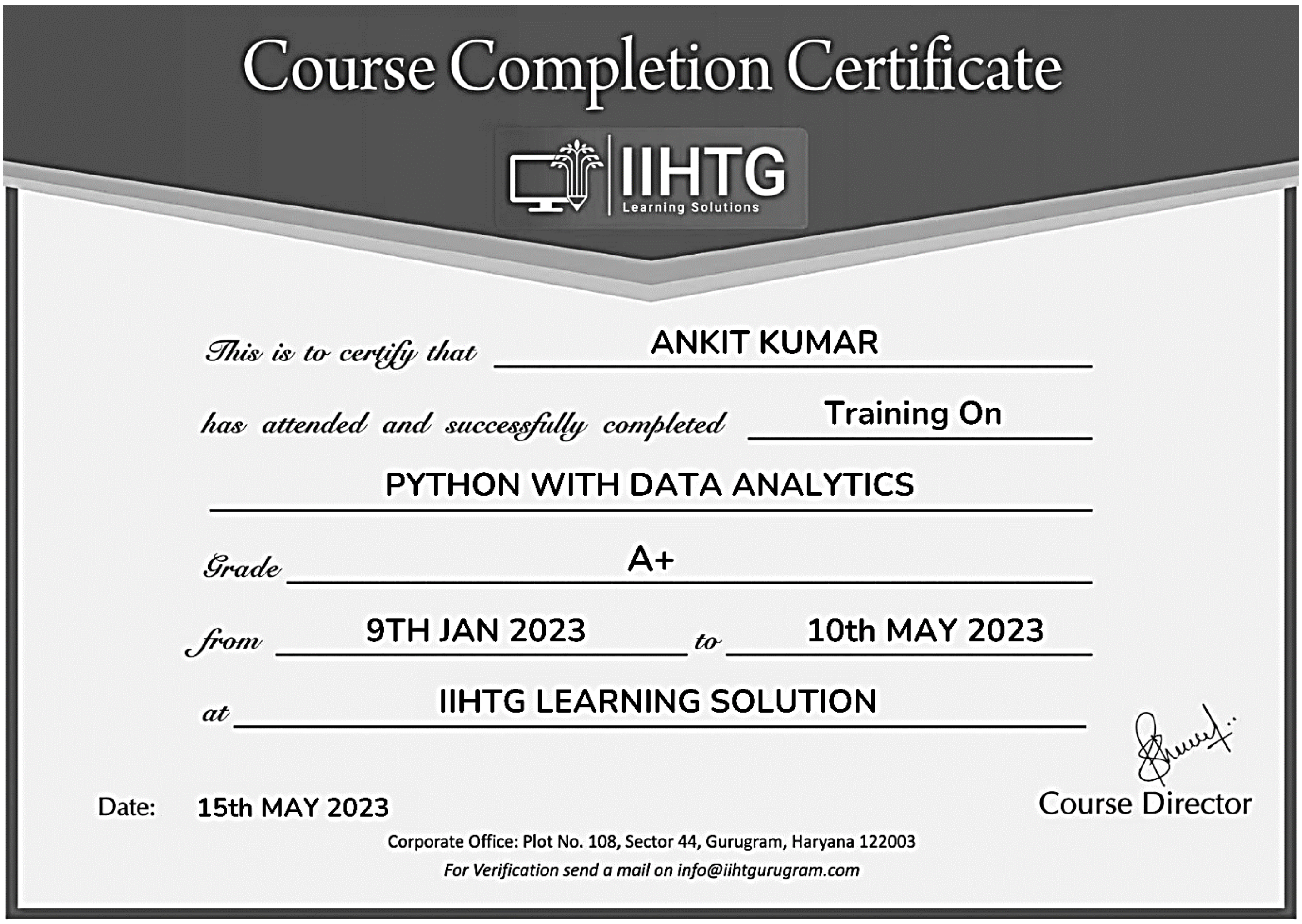
We wish him all the best for his future endeavours.

On behalf of IIHTG Gurugram

[illegible]

Authorized Signatory

Plot No. 108 Sec - 44, Gurugram - 122003 Haryana | 0124-4303322, +91 9811003080, 9811003090



Course Completion Certificate

IIHTG
Learning Solutions

This is to certify that ANKIT KUMAR

has attended and successfully completed Training On

PYTHON WITH DATA ANALYTICS

Grade A+

from 9TH JAN 2023 *to* 10th MAY 2023

at IIHTG LEARNING SOLUTION

[illegible]

Date: 15th MAY 2023

Course Director

Corporate Office: Plot No. 108, Sector 44, Gurugram, Haryana 122003

For Verification send a mail on info@iihtgurugram.com

CENTRAL UNIVERSITY OF SOUTH BIHAR

PANCHANPUR,GAYA,BIHAR- 824236

Certificate

This is to certify that the project work entitled **LOAN STATUS PREDICATION USING PYTHON WITH MACHINE LEARNING** submitted by **Mr. Ankit Kumar** to the Department of Computer Science, Central University of South Bihar(Gaya) in partial fulfillment for the award of the degree of MSCSC: M.Sc. Computer Science.

We certify further that to the best of my knowledge, the project work reported here in is not a part of any other project or software on the basis of which a degree or an award has been conferred earlier to any other candidate.

Head of Department | Internal Examiners

Dept. of Computer Science

Central University of South Bihar

DECLARATION

I **Ankit Kumar,** a student of department of Computer Science, Central University of south Bihar, hereby declare that the project work entitled "Loan Status Predication Using Python with Machine Learning" is being submitted to the Department of Computer Science for the fulfillment for reward of degree in M.Sc. Computer Science is a record of bonafide project work carried out by me.

I further declare that the work reported in this project has not been submitted and will not be submitted, either in the part or full, for the award of any degree or diploma in this university or any other university.

Ankit Kumar

M.Sc. CS (2021-2023)

Enroll: CUSB2102312010

ACKNOWLEDGEMENT

It is great happiness and privilege for me to represent this Project report. I would like to express my special thanks of gratitude to Dr. Prabhat Ranjan, Associate Professor and head of department, Department of Computer Science, for providing all the necessary facilities to us and gave extra support in all matters for the successful completion of this project.

With pleasure I also thank all the staff members of Department of Computer Science who provides me valuable guidance during my need.

I would like to express my special thanks of gratitude to my project lead Mr. Rakesh Kumar Jha (Senior trainer) IIHT GURGAON (Haryana) who availed this magnificent option to us to move ahead. He has always inspired and guided us for the right track to be followed for all the system analysis section of this project.

I'd like to be thankful to my colleague for the their valuable support and cooperation during my project.

Ankit Kumar

ABSTRACT

When any financial institution lends the money to the person, it is always been high risk. Today data is increasing with the rapid pace in the banks, therefore the bankers need to evaluate the person’s data before giving the loan. It can be a big headache to evaluate the data. This problem is solved by analyzing and training the data by using one of the Machine Learning algorithms. For this, we have generated a model for the prediction that the person will get the loan or not. The primary objective of this paper is to check whether the person can get the loan or not by evaluating the data with the help of decision tree classifiers which can gives the accurate result for the prediction.

Keywords—Loan, Machine-Learning, Data-training, Model Building.

Loan Status Predication Using Python with Machine Learning Page

Loan Status Predication Using Python with Machine Learning

1. Introduction

- ➢ Loan prediction is a common problem faced by financial institutions such as banks, credit unions, and other lending organizations.
- ➢ Predicting whether a loan applicant will default on their loan is an essential part of the lending process, and accurate predictions can help lenders minimize their risks and make better lending decisions.

1.1 Types of Loan:

There are several types of loans categorized as follows:

- ✓ Secured Loans: These are loans that are backed by collateral, such as a home or car.
  Eg:- Mortgage loans, auto loans, and home equity loans.
- ✓ Unsecured loans: These are loans that are not backed by collateral, and are approval based on the borrower's creditworthiness.
  Eg:-Personal loans, credit card loans and student loans.
- ✓ Fixed-rate loans: These are loans where the interest rate remains the same of entire loan period, which makes budgeting easier.
  Eg: Fixed-rate mortgages and fixed-rate personal loans.
- ✓ Variable-rate loans: These are loans where the interest rate fluctuates over the loan period, based on changes in the market interest rates.
  Eg: Adjustable-rate mortgages and variable rate personal loans.

- ✓ Business loans: These are loans by businesses to finance their operations, purchase equipment, or expand their operations. Business loans can be secured or unsecured, and may come with different terms and conditions depending on the lender and the borrower's credit worthiness.
- ✓ Government loans: These are loans offered by government agencies to support specific activities, such as home purchases, education, or small business development. Eg: MSME loans.

1.2 Goals:

- ✓ The goals of loan status prediction using machine learning techniques can depending on specific context and objectives of organization or individual employing model.
- ✓ These are some common goals associated with loan status prediction as below:
  - ❖ Risk Assessment:
    - i) It can analyze historical loan data, borrower information, and various financial indicators to predict the loan default.
    - ii) This helps lenders make informed decisions about loan approval and manage their overall risk exposure.
  - ❖ Decision Support:
    - i) ML models can provide decision support by automatically evaluating loan application.
    - ii) It provides recommendations or predication regarding the likelihood of loan approval.
    - iii) This assists loan officers or automated system in making consistent and efficient decision.

- ❖ Fraud Detection:
  - i) Loan status prediction models can be used to identify potentially fraudulent loan applications.
  - ii) By analyzing patterns, anomalies, and historical data, ML algorithms can flag suspicious activities or applications that exhibit high-risk characteristics. Etc.

2 Machine Learning

2.1 Introduction:

- ✓ Machine learning is a subset of artificial intelligence.
- ✓ Machine Learning(ML) is the scientific study of algorithms and statistical models that computer systems.
- ✓ It is use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead.

2.2 Importance of Machine Learning

- ✓ Fraud Detection: Machine learning can be used to detect fraudulent activity, such as credit card fraud or identity theft, by analyzing patterns and anomalies in large amounts of data.
- ✓ Better Decision Making: Machine learning algorithms can analyze vast amounts of data quickly and accurately to make predictions and decisions.
- ✓ Predictive Maintenance: Machine learning can be used to predict when equipment or machinery is likely to fail, allowing for preventative maintenance to be scheduled before a breakdown occurs.
- ✓ Medical Diagnosis and Treatment : Machine learning algorithms can be trained to analyze medical images and data to aid in diagnosis and treatment of diseases.
- ✓ This can help businesses and organizations make more informed decisions and improve their overall performance.
- ✓ Automation of tasks: Machine learning can automate tasks that would otherwise require human intervention, freeing up valuable time and resources.
- ✓ This can include tasks such as data entry, customer service, and quality control.

- ✓ Personalization: Machine learning can help personalize experiences for customers, such as recommending products based on their purchase history or providing personalized content based on their preferences.

2.3 Uses of Machine Learning

- ✓ Topic modeling: By analyzing the content of lectures or meetings, machine learning algorithms can be identify the main topics and theme discussed. It can be help note-takers focus their attention on the most important points and organize their notes accordingly.
- ✓ Named Entity recognition: It can be used to automatically identify and extract key information such as names, dates and locations from lecture or meeting notes.

2.4 Types of Machine Learning

There are some types of machine learning as below:

2.4.1 Supervised Learning:

- ✓ It is a type of machine learning where the algorithm learns to make predictions based on labeled training data.
- ✓ Labeled data usually knows as known data.
- ✓ Goal of supervised learning is to learn a function that can generalize well to unseen data.
- ✓ Some few examples of supervised learning as below:

i) Image classification: Given a set of images and their corresponding labels, the goal is to train a model that can correctly classify new images into their respective categories.

ii) Spam detection: Given a set of emails labeled as spam or non-spam, the goal is train a model that can accurately predict whether new emails are spam or not.

2.4.2 Unsupervised Learning

In unsupervised learning, the input data is not labeled, and the goal is discover meaningful patterns or grouping of data.

- ✓ Some few examples of unsupervised learning as below:
- ✓ Clustering: Given a set of unlabeled data points, the goal is to group similar data points together in clusters. Clustering is commonly used in market segmentation, image segmentation, and customer segmentation.
- ✓ Anomaly detection: Given a set of data points, the goal is to identify data points that are significantly different from the rest of the data. It is commonly used in fraud detection, intrusion detection, and fault detection. Etc.

2.4.3 Semi-supervised Learning

- ✓ As the name suggests, semi-supervised learning is a bit of both supervised and unsupervised learning and uses both labeled and unlabeled data for training.
- ✓ In a typical scenario, the algorithm would use a small amount of labeled data with a large amount of unlabeled data.

2.4.4 Reinforcement Learning

- ✓ Reinforcement learning is a type of machine learning where an agent learns to make decision by interacting with an environment.
- ✓ The agents receive on punish or rewards for each action it takes.
- ✓ It is often used in robotics, game playing, and control system.

2.5 Deep Learning

- ✓ At a very basic level, deep learning is a machine learning technique.
- ✓ It teaches a computer to filter inputs through layers to learn how to predict and classify information.
- ✓ Observations can be in the form of images, text, or sound.
- ✓ The inspiration for deep learning is the way that the human brain filters information.

2.5.1 Introduction of Deep Learning

- ✓ Imagine you are work for a loan company, and I need to build a model for predicting, whether a user (borrower) should get a loan or not? You have the features for each customer like age, bank balance, salary per annum, whether retired or not and soon.

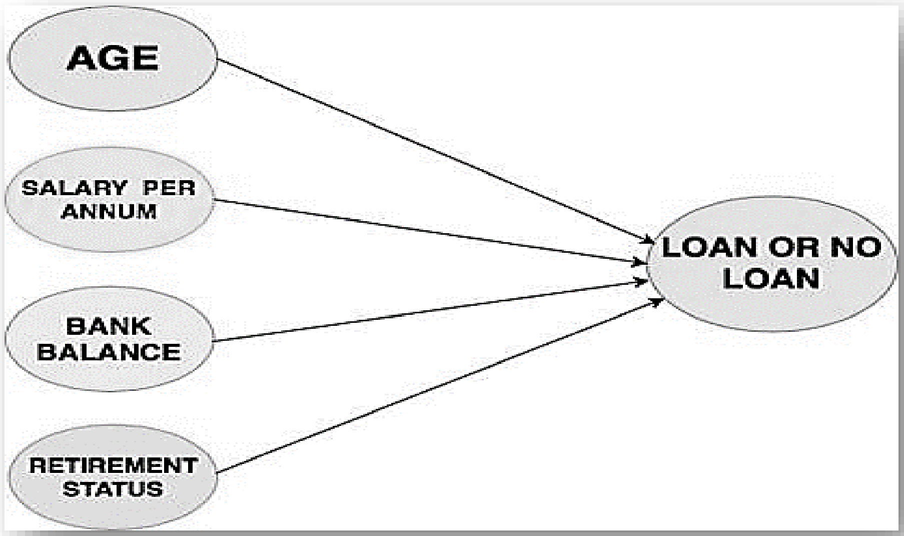
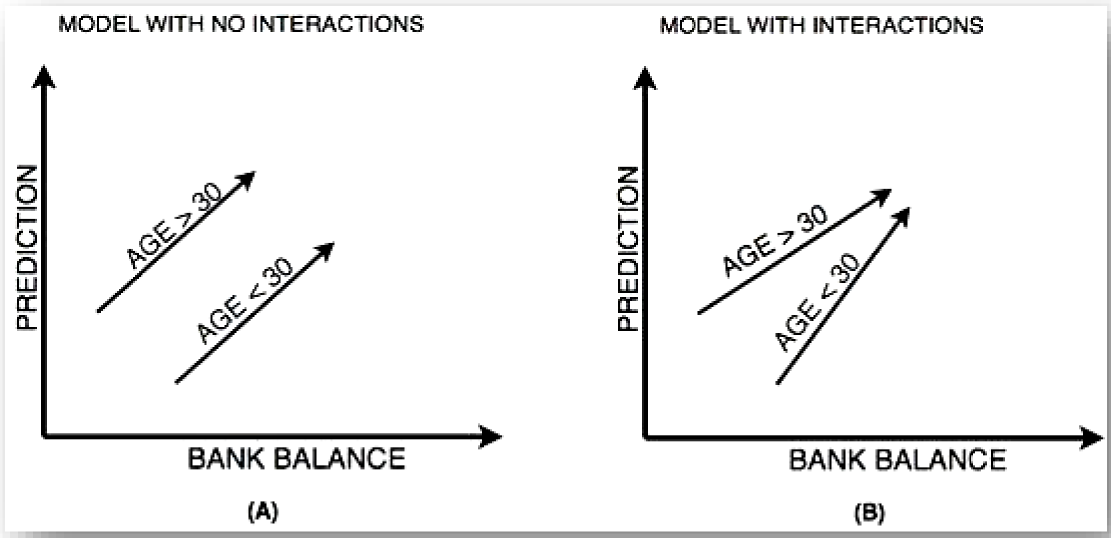


Fig: Model for Loan

- ✓ Consider if you want to solve this problem using a linear regression model, then the linear regression will assume that the outcome (whether a customer's loan should be sanctioned or not) will be the sum of all the features.

- ✓ It will take into account the effect of age, salary, bank balance, retirement status and so.
- ✓ So the linear regression model is not taking into account the interaction between these features or how they affect the overall loan process.



T **Fig: Bank Balance Predication**

In the above figure left (A) shows prediction from a linear regression model with absolutely no interactions in which it simply adds up the effect of age (30 < age < 30) and bank balance, you can observe from figure (A) that the lack of interaction is reflected by both lines being parallel that is what the linear regression model assumes.

On the other hand, figure right (B) shows predictions from a model that allows interactions in which the lines do not have to parallel. Neural Networks is a pretty good modeling approach that allows interactions like the one in figure (B) very well and from these neural networks evolves a term known as Deep Learning which uses these powerful neural networks. Because the neural network takes into account these type of interactions so well it can perform quite well on a well of prediction problems you have seen till now or possibly not heard.

Since neural networks are capable of handling such complex interactions gives them the power to solve challenging problems and do amazing things with Image, text, Audio, video.

This list is a subset of what neural networks are capable of solving, almost anything you can think of in data science field can be solved with neural networks.

2.5.2 Interactions in Neural Network

- ✓ The neural network architecture looks something similar to the above figure.
- ✓ On the far left you have the input layer that consists of the features like age, salary per annum, bank balance, etc. and on the far right, you have the output layer that outputs the prediction from the model which in your case is whether a customer should get a loan or not.

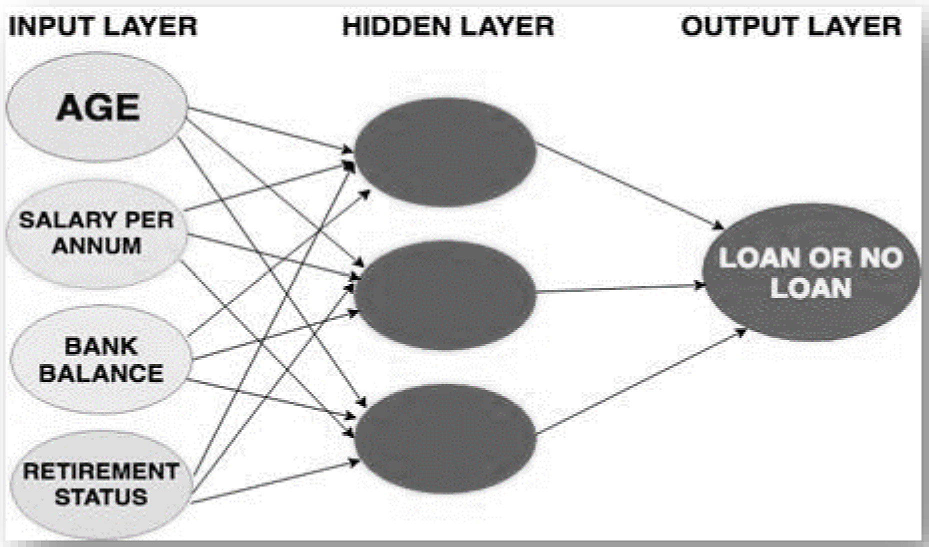


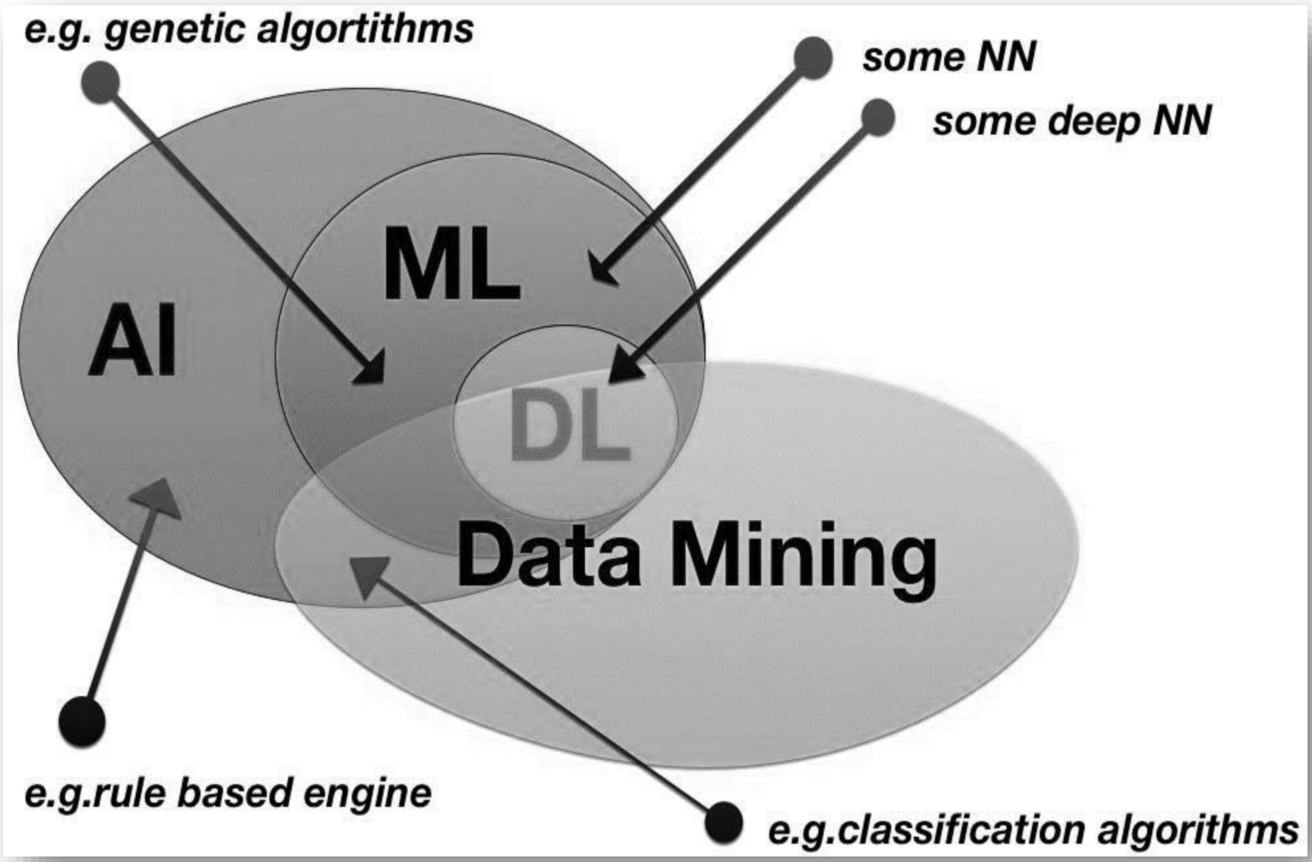
Fig: Interactions in Neural Network

- ✓ The layers a part from the input and the output layers are called the hidden layers.
- ✓ Technically, each node in the hidden layer represents an aggregation of information from the input data; hence each node adds to the model's capability to capture interactions between the data.

2.6 Relation between Data Mining, Machine Learning & Deep Learning:

- ✓ Data mining, machine learning, and deep learning are all related to the field of Artificial Intelligence(AI) are often used interchangeably, but they have different meaning and applications.
- ✓ Data mining is the process of discovering patterns in large datasets, often with the goal of finding correlations or associations between variables. It uses statistical and mathematical techniques to analyze data and extract insights that can be used for decision-making.
- ✓ Machine learning is a subset of AI that focuses on building algorithms that can learn from data without being explicitly programmed.
- ✓ It involves developing models that can make predictions or decisions based on patterns in data.
- ✓ These models can be supervised, unsupervised, or semi-supervised.
- ✓ Deep learning is a subset of machine learning that involves building neural networks with many layers, which can learn and extract features from data at multiple levels of abstraction.
- ✓ Deep learning has been very successful in areas such as image recognition, natural language processing, and speech recognition.

- ✓ It applies them to large amounts of data to learn, understand, and identify complicated patterns. Automatic language translation and medical diagnoses are examples of deep learning.



3 Python

3.1 Introduction of python

- ✓ Python is a high-level, interpreted language ie means that python code executed directly by the python interpreter without the need for compilation into machine code.
- ✓ When run a python program, the interpreter reads the code line by line and executes it in real time.
- ✓ Python is a general purpose programming language that is often applied in scripting roles.
- ✓ Python is Object-Oriented : Python supports the Object-Oriented style or technique of programming with object.

3.2 History of Python

- ✓ Python was developed by Guido van Rossum in the late 1980s and early 1990s.
- ✓ Python 3.11.2 is the newest major release of the python.

3.3 Feature of python

- ✓ Python is easy to learn and has a simple syntax that is easy to read and write.
- ✓ It is considered one of the most beginner-friendly programming language.
- ✓ Portal: Python can be run on a variety of platforms such as windows, Linux and Macintosh.
- ✓ This makes it a versatile language that can be used in many different contexts.
- ✓ Open Source: Python is an open-source language, which means that its source code is freely available and can be modified and distributed by anyone.

- ✓ GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Macintosh, Unix.

3.4 Python Variable Types

- ✓ Variables are reserved memory locations to store values. This means that when to create a variable reserve some space in the memory.

➔ Variables are reserved memory locations to store values.

- ✓ Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory.
- ✓ Python variables do not need explicit declaration to reserve memory space.
- ✓ The declaration happens automatically when you assign a value to a variable.

➔ There are six standard Data Types of Python:

- ➢ Numeric
- ➢ String
- ➢ List
- ➢ Tuple
- ➢ Set
- ➢ Dictionary

Numeric Data types:

- ✓ In Python, numeric data type represents the data that has a numeric value.
- ✓ The numeric value can be an integer, floating number, or even complex number.
- ✓ These values are defined as int, float, and complex classes in Python.

String Data Type:-

- ✓ The string is a sequence of Unicode characters.
- ✓ A string may be a collection of 1 or more characters put during a quotation mark, double-quote, or triple quote.
- ✓ It can be represented using an str class.

Example:-

```
string = “Ankit”
print(string)
output: Ankit
```

List Data Type:-

- ✓ A list is formed (or created) by placing all the items (elements) inside square brackets [ ], separated by commas.
- ✓ It can have any number of items and they may or may not be of different types (integer, float, string, etc.).
- ✓ A list is mutable, which suggests we will modify the list

Example:

```
List = [3,8.2,7.2,"Ankit"]
print("List[2] = ", List[2])
Output:  List[2] = 7.2
```

Tuple Data Type:-

- ✓ A tuple is defined as an ordered collection of Python objects.
- ✓ The only difference between tuple and list is that tuples are immutable i.e. tuples can’t be modified after it’s created.
- ✓ It is represented by tuple class. we can represent tuples using parentheses ( ).

Example:

```
Tuple = (5,1,2.5,"Ankit")
print("Tuple[1] = ", Tuple[1])      #[0,4]- Start index 0 and range on 4-1
Output: Tuple[1] =  1
print("Tuple[0:3] =", Tuple[0:3])
Output: Tuple[0:3] =  (5,1,2.5)
```

Set Data Type:-

- ✓ A set is a collection of unique and immutable elements.
- ✓ The elements in a set are unordered, meaning that they are not stored in any specific order.
- ✓ Sets can be created by enclosing a comma-separated sequence of elements inside curly braces ({}) or by using the built-in set() function.

Example:

```
# create a set using curly braces
my_set = {1, 2, 3, 4, 5}
# create a set using the set() function
my_set2 = set([6, 7, 8, 9, 10])
```

Exercise:

```
Set = {4,3,6.6,"Hello"}
print(Set) Output: {'Hello', 3, 4, 6.6}
```

Dictionary Data Type:-

- ✓ In Python, a dictionary is a collection of key-value pairs, where each key is unique and associated with a corresponding value.
- ✓ Dictionaries are enclosed in curly braces ({}) and each key-value pair is separated by a colon (:), with each pair separated by a comma.

Example: ages = {"Alice": 27, "Bob": 35, "Charlie": 19}

Output: print(ages["Bob"])   # Output: 35

4 Overview of Related Project

- ✓ Loan status prediction is a common machine learning problem in the banking and financial sector.
- ✓ The goal of loan status prediction is to determine whether a loan applicant is likely to default on their loan or not.
- ✓ This is important because banks and financial institutions need to manage their risk and ensure that they are lending money to borrowers who are likely to repay their loans.

In Python, loan status prediction can be performed using machine learning techniques such as logistic regression, decision trees, and random forests. The process typically involves the following steps:

1. Data collection and preprocessing: This involves collecting data on loan applicants and their loan history, and cleaning and preprocessing the data to prepare it for analysis.
2. Feature engineering: This involves selecting and transforming the relevant features (i.e., variables) that will be used to predict the loan status. Examples of features might include the applicant's income, credit score, employment history, and loan amount.
3. Model training: This involves training a machine learning model on a portion of the data. The model will learn to predict the loan status based on the selected features.
4. Model evaluation: This involves evaluating the performance of the trained model on a separate portion of the data. Metrics such as accuracy, precision, recall, and F1 score can be used to assess the model's performance.
5. Model deployment: Once the model has been trained and evaluated, it can be deployed in a production environment to make predictions on new loan applications.

In addition to the above steps, it is also important to perform data visualization and exploratory data analysis to gain insights into the data and identify any patterns or correlations that may exist between the features and the loan status.

4.1 Hardware & Software Used

Hardware Used

1. Windows 10

Software /Code Editor Used

1. JupyterLite(Online),Google Colab

Libraries Used

1. Pandas 2.NumPy 3.Seaborn 4.Matplotlib 5.Sklearn

Hardware Used

- Windows 10 is the latest version of the Windows operating system, released by Microsoft in 2015.
- It is designed to be a more unified and user-friendly experience across all devices, including desktops, laptops, tablets, and smartphones.
- Windows 10 features a range of improvements and new features, including a new start menu, virtual desktops, Microsoft Edge web browser, and the Cortana virtual assistant.

Software /Code Editor Used

- JupyterLab is a web-based interactive development environment (IDE) that allows users to create and share documents that contain live code, equations, visualizations, and narrative text.
- It is an extension of the popular Jupyter Notebook, which was originally designed for scientific computing, but has since been used for a wide range of applications.

- JupyterLab provides a flexible and powerful environment for data analysis, machine learning, and scientific research.
- It supports multiple programming languages, including Python many more. Users can install packages and libraries, run code cells, visualize data, and interact with various data sources.

Libraries Used

Pandas

- Pandas is a popular open-source library for data analysis and manipulation in Python.
- With Pandas, users can easily read and write data in a variety of formats, including CSV, Excel, SQL databases, and more.
- It also provides powerful tools for data cleaning, transformation, aggregation, and visualization

Numpy

- Numpy(short for Numerical Python) is a powerful open source library for scientific computing in python.
- It provides efficient and powerful tools for working with arrays and matrices of numerical data, and is widely used in data science, Machine learning and other scientific fields.
- Numpy's powerful set of functions, which include element-wise operations, linear algebra operations, statistical functions and more.

Matplot and Seaborn:

- **Matplot and Seaborn is** the best way to get insights is by visualizing the data.

- Matplotlib is a versatile library that provides a wide range of plot types, including line plots, scatter plots, bar charts, histograms, and more.
- Matplotlib is a low-level library, meaning that it provides a lot of control but may require more code to create complex plots.
- Seaborn is a visualization library in Python. It is built on top of Matplotlib.
- Seaborn also provides a set of default styles and color palettes that make it easy to create aesthetically pleasing plots
- Data can be visualized by representing it as plots which is easy to understand, explore and grasp.
- Such data helps in drawing the attention of key elements. To analyse a set of data using Python, we make use of Matplotlib, a widely implemented 2D plotting library.

<u>Important feature of seaborn</u>:

- ✓ Seaborn is a popular data visualization library in Python that is built on top of Matplotlib. Some of the key features of Seaborn include:
  - Statistical visualization: Seaborn provides a range of statistical visualizations that are commonly used in data analysis, such as scatter plots, line plots, bar plots and histograms.
  - Default styles and color palettes: it easy to create high-quality plots without having to manually adjust every aspect of the plot.
  - Integration with Pandas: Seaborn is designed to work seamlessly with Pandas, a popular data manipulation library in Python. This makes it easy to create

visualizations directly from data stored in Pandas data structures such as Data Frames.

Library of Seaborn And Matplot:

from matplotlib import pyplot as plt
import seaborn as sb

Sklearn

- Scikit-learn, also known as sklearn, is a popular open-source machine learning library for Python.
- It provides a wide range of tools and algorithms for machine learning tasks such as classification, regression, clustering, and dimensionality reduction.
- Scikit-learn provides a user-friendly and consistent interface for various machine learning algorithms, making it easy for users to experiment with different models and techniques.
- Preprocessing Tools: Scikit-learn provides various preprocessing tools for handling missing data, scaling features, encoding categorical variables, and more.
- Model Selection and Evaluation: Scikit-learn provides tools for model selection and evaluation, including cross-validation, hyperparameter tuning, and model selection metrics.
- Integration with other libraries: Scikit-learn integrates well with other Python libraries such as NumPy, Pandas, and Matplotlib, making it easy to manipulate and visualize data.

5 Case Study

5.1 Problem Statement

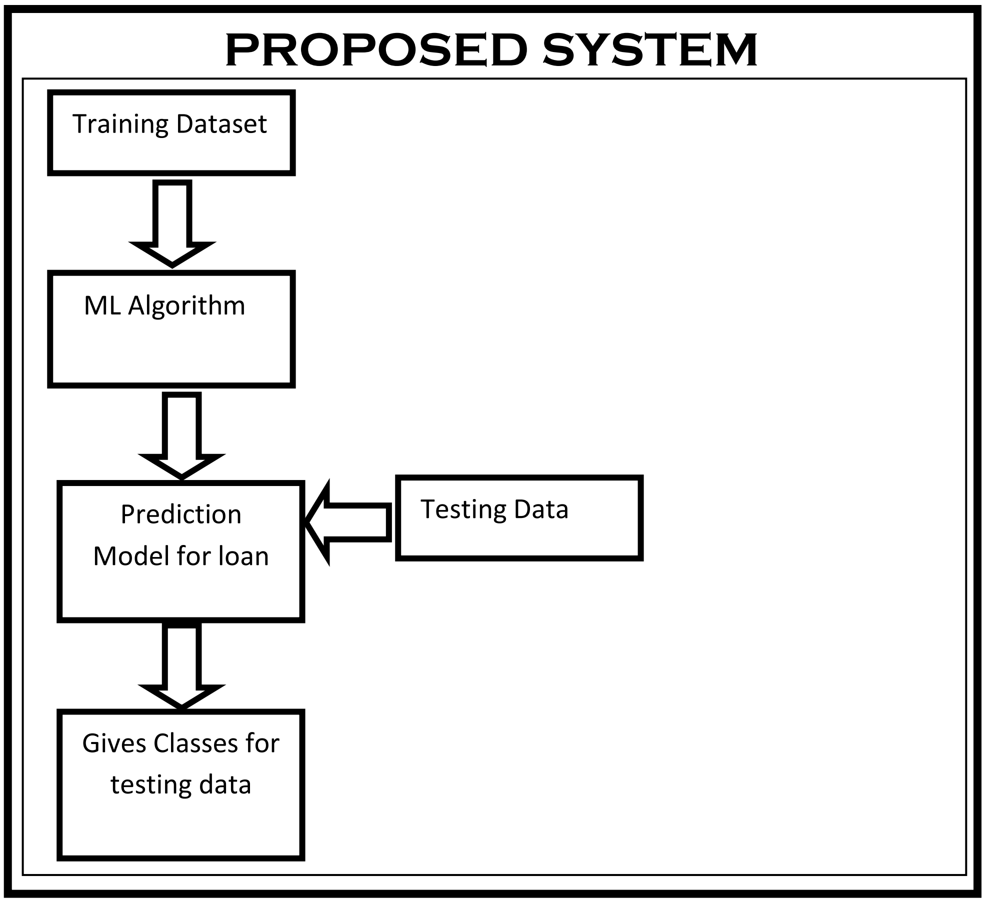
- ✓ Banks need to analyze for the person who applies for the loan will repay the loan or not.
- ✓ Sometime it happens that customer has provided partial data to the bank, in this case person may get the loan without proper verification and bank may end up with loss.
- ✓ Bankers cannot analyze the huge amounts of data manually, it may become a big headache to check whether a person will repay its loan or not.
- ✓ It is very much necessary to know the person getting loan is going in safe hand or not. So, it is pretty much important to have a automated model which should predict the customer getting the loan will repay the loan or not.

5.2 Proposed System

- ✓ I have developed a prediction model for Loan sanctioning which will predict whether the person applying for loan will get loan or not.
- ✓ The major objective of this project is to derive patterns from the datasets which are used for the loan sanctioning process and create a model based on the patterns derived in the previous step.
- ✓ This model is developed by using the one of the machine learning algorithms.
- ✓ In the proposed model for loan prediction, Dataset is split into training and testing data.
- ✓ After then training datasets are trained using the decision tree algorithm and a prediction model is developed using the algorithm.
- ✓ Testing datasets are then given to model for the prediction of loan.
- ✓ The motive of this paper is to predict the defaults who will

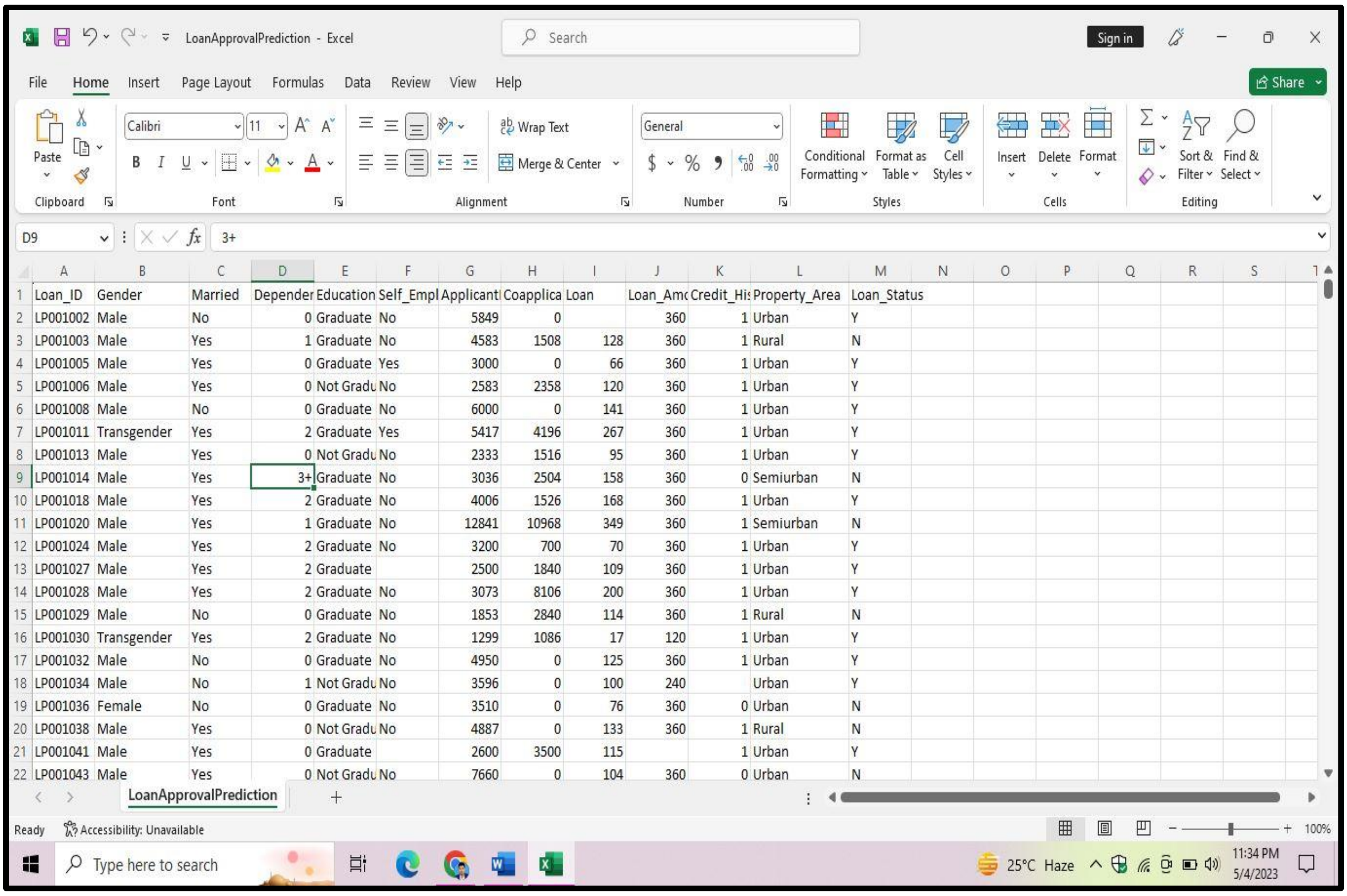
repay the loan or not.

- ✓ Various libraries like pandas, numpy have been used. After the loading of datasets, Data Preprocessing like missing value treatment of numerical and categorical is done by checking the values.
- ✓ Numerical and categorical values are segregated. Outliers and frequency analysis are done, outliers are checked by getting the boxplot diagram of attributes.



5.3 Dataset

- ✓ Datasets are gathered from Kaggle. Dataset is now provided to Machine learning models on the basis of this facts this version is trained.
- ✓ Data sets are divided into Existing and New Customers. Every new applicant info act as a fact test set. After the operation of testing, model expect whether the brand-new applicant is in case for approval of the loan or now not primarily based upon the inference it concludes on the idea of training information sets.



D9 3+

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Loan_ID	Gender	Married	Depender	Education	Self_Empl	Applicant	Coapplica	Loan	Loan_Amc	Credit_His	Property_Area	Loan_Status
2	LP001002	Male	No	0	Graduate	No	5849	0		360	1	Urban	Y
3	LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N
4	LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y
5	LP001006	Male	Yes	0	Not Gradu	No	2583	2358	120	360	1	Urban	Y
6	LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y
7	LP001011	Transgender	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y
8	LP001013	Male	Yes	0	Not Gradu	No	2333	1516	95	360	1	Urban	Y
9	LP001014	Male	Yes	3+	Graduate	No	3036	2504	158	360	0	Semiurban	N
10	LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y
11	LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N
12	LP001024	Male	Yes	2	Graduate	No	3200	700	70	360	1	Urban	Y
13	LP001027	Male	Yes	2	Graduate		2500	1840	109	360	1	Urban	Y
14	LP001028	Male	Yes	2	Graduate	No	3073	8106	200	360	1	Urban	Y
15	LP001029	Male	No	0	Graduate	No	1853	2840	114	360	1	Rural	N
16	LP001030	Transgender	Yes	2	Graduate	No	1299	1086	17	120	1	Urban	Y
17	LP001032	Male	No	0	Graduate	No	4950	0	125	360	1	Urban	Y
18	LP001034	Male	No	1	Not Gradu	No	3596	0	100	240		Urban	Y
19	LP001036	Female	No	0	Graduate	No	3510	0	76	360	0	Urban	N
20	LP001038	Male	Yes	0	Not Gradu	No	4887	0	133	360	1	Rural	N
21	LP001041	Male	Yes	0	Graduate		2600	3500	115		1	Urban	Y
22	LP001043	Male	Yes	0	Not Gradu	No	7660	0	104	360	0	Urban	N

LoanApprovalPrediction

5.4 Implementation of Model

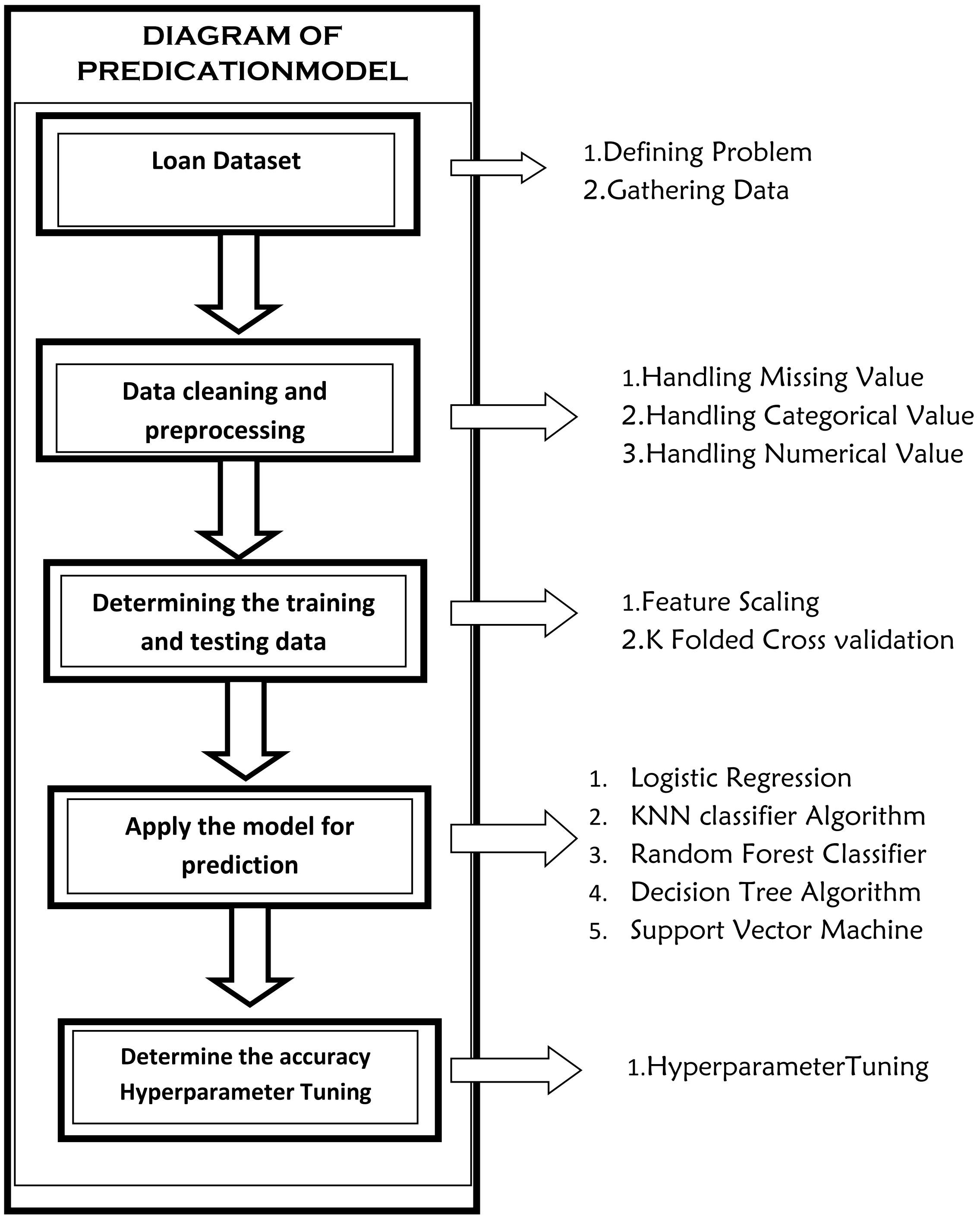


DIAGRAM OF PREDICATIONMODEL
Loan Dataset
Data cleaning and preprocessing
Determining the training and testing data
Apply the model for prediction
Determine the accuracy Hyperparameter Tuning
1.Defining Problem
2.Gathering Data
1.Handling Missing Value
2.Handling Categorical Value
3.Handling Numerical Value
1.Feature Scaling
2.K Folded Cross validation
1. Logistic Regression
2. KNN classifier Algorithm
3. Random Forest Classifier
4. Decision Tree Algorithm
5. Support Vector Machine
1.HyperparameterTuning

5.4.1 Data Exploratory Analysis

- ✓ Data Exploratory Analysis (DEA) is a process of analyzing and understanding data sets to gain insights into the underlying patterns, relationships, and trends.
- ✓ It involves examining the data from various angles, using different statistical and visualization techniques to identify patterns, trends, and outliers.
- ✓ DEA helps to identify the data quality issues, missing values, and anomalies that need to be addressed before proceeding with further analysis.
- ✓ The process of DEA involves performing descriptive statistics, visualizing the data using charts, histograms, scatter plots, and other visualization techniques to understand the data distribution, and dependencies among the variables.
- ✓ It helps to identify the data patterns that can help to build predictive models and gain insights into the underlying relationships between the variables.
- ✓ The ultimate goal of DEA is to gain insights into the data that can help to make informed decisions, identify new opportunities, and solve complex business problems.
- ✓ DEA is an important first step in the data analysis process, as it helps to understand the data better and can guide the selection of appropriate statistical and machine learning models.
- ✓ Data Exploratory Analysis is done through Bivariate Analysis by Numeric (T Test) or Categorical (Chisquare).
- ✓ Visualization of Attributes are also done by Bivariate Analysis.
- ✓ We use a Bivariate Analysis Plot which is used to analyze the impact of features on the target variables.

Bivariate Analysis:

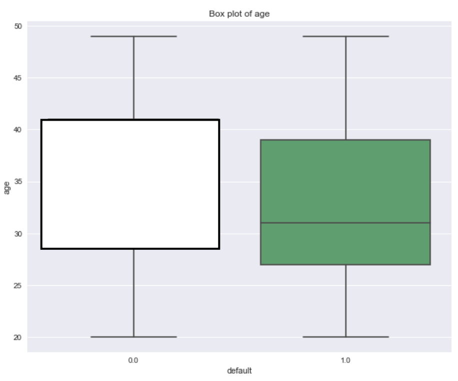
- Applicants with high incomes should have more chances of loan approval.
- Applicants who have repaid their previous debts should have higher chances of loan approval.
- Loan approval should also depend on the loan amount. If the loan amount is less, the chances of loan approval should be high.
- Lesser the amount to be paid monthly to repay the loan, the higher the chances of loan approval.

Boxplot:

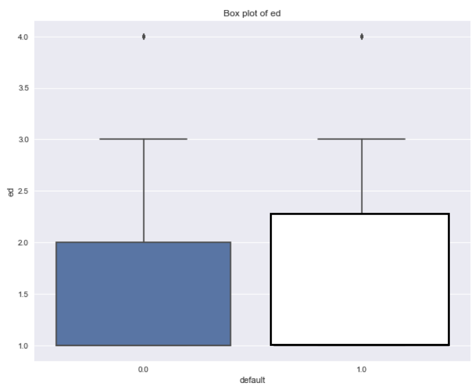
- Boxplot is basically used in find outliers.
- A boxplot is a graphical representation of a dataset that displays the distribution of data values based on their quartiles.

  <Quartiles are values that divide a dataset into four equal parts, each containing 25% of the data.>

  < The three quartiles, Q1, Q2 (also known as the median), and Q3, divide the data into four equal parts.>
- In a bivariate boxplot, two separate boxplots are displayed side-by-side on the same graph, with each boxplot representing the distribution of one variable.
- By visualizing the distribution of two variables simultaneously, a bivariate boxplot can provide insights into the relationship between the two variables, including any outliers, differences in variability, and potential patterns or trends.
- It can also help identify potential data quality issues, such as missing or inconsistent values.



Box Plot of Age



Box Plot of Education

5.5 Object of the case Study

- ➢ These are regression problem with clear outliers which cannot be predicted using any reasonable method.
- ➢ A comparison of the some methods has been done:
  - ➔ Linear Regression
  - ➔ Random Forest Regressor
  - ➔ KNN classifier
  - ➔ Decision Tree Algorithm
  - ➔ Support Vector Machine

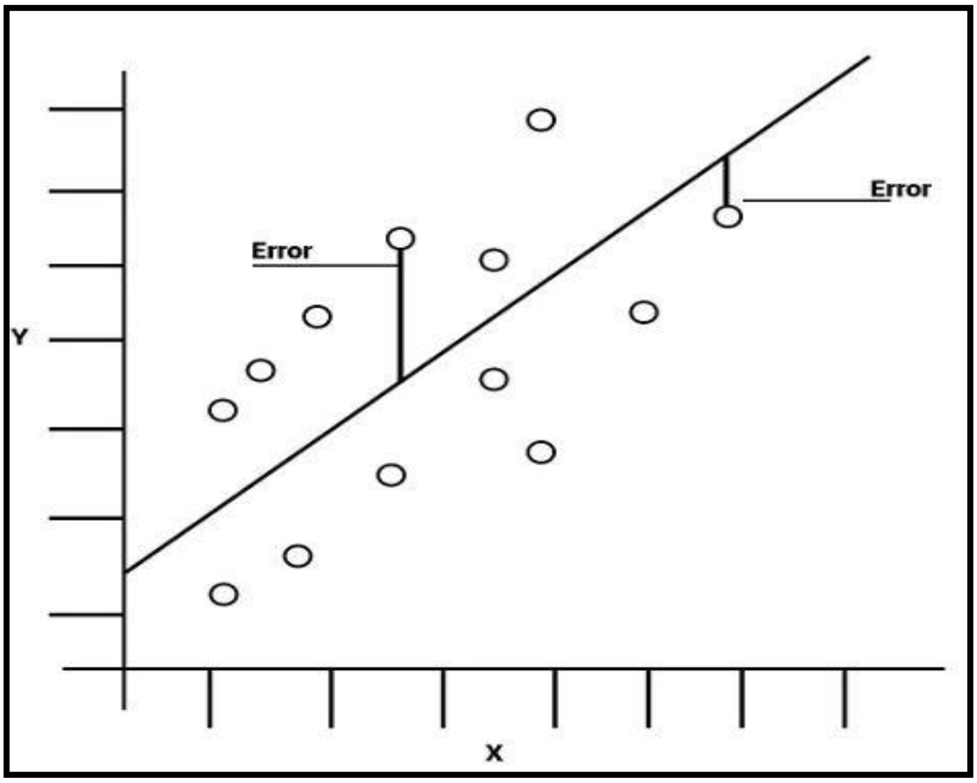
5.5.1 Linear Regression

- ✓ Linear regression is a statistical method commonly used for predicting loan status in machine learning.
- ✓ In a loan status prediction model using linear regression, several factors or independent variables are considered to predict the dependent variable, i.e., the loan status.
- ✓ Some of the variables that may be used in a loan status prediction model using linear regression include:

1. Credit Score: The credit score of the applicant is an important factor in determining their creditworthiness and their ability to repay the loan.
2. Debt-to-Income Ratio: This is the ratio of the applicant's debt to their income, which is used to assess their ability to repay the loan.
3. Loan Amount: The amount of the loan requested by the applicant is also an important factor in predicting loan status.
4. Employment Status: The employment status of the applicant, such as whether they are employed, self-employed, or unemployed, can also be used to predict their ability to repay the loan.
5. Age: The age of the applicant is also considered in predicting loan status, as it can be a factor in assessing their stability and

reliability.

- ✓ The model’s performance can be evaluated using metrics such as mean squared error or R-squared value.
- ✓ Squared Error or R-squared Value error is a function of Numpy.
- ✓ The Mean Squared Error measures how close a regression line is to a set of data points.
- ✓ It is a risk function corresponding to the expected value of the squared error loss.



.The Mean Squared Error is calculated as:

MSE = (1/n) * Σ(actual – forecast)2

where:

- Σ – a symbol that means “sum”
- n – sample size
- actual – the actual data value
- forecast – the predicted data value
- n- total row of csv file in term of dataset.

R-squared value, is also known as coefficient of determination, is a statistical measure used to evaluate the goodness of fit of a linear regression model.

The formula for calculating R-squared value is:

R-squared Error = 1 - (SSres / SStot)

- ➔ where SSres is the sum of squared residuals (the difference between the actual and predicted values of the dependent variable),
- ➔ SStot is the total sum of squares (the difference between the actual values of the dependent variable and the mean of the dependent variable).

The R-squared value ranges from 0 to 1, with a value of 1 indicating a perfect fit of the model to the data

5.5.2 Random Forest Regressor:

- ✓ It is a type of machine learning algorithm that is used for regression tasks, where the goal is to predict a continuous numerical value as the output.
- ✓ It is an ensemble learning method that combines multiple decision trees to produce a more accurate and robust prediction model.
- ✓ In Random Forest Regressor, a large number of decision trees are created using random subsets of the training data and random subsets of the input features.
- ✓ Each decision tree is trained on a different subset of the data, and the final prediction is made by averaging the predictions of all the trees.
- ✓ It is used in loan status prediction to build a model that can predict the loan status (i.e., whether the loan will be repaid or not) based on various factors such as credit score, debt-to-income ratio, loan amount, employment status, and age.
- ✓ To use Random Forest Regressor for loan status prediction, a dataset of past loan applications is collected and preprocessed.
- ✓ The dataset is split into training and testing subsets, and the Random Forest Regressor model is trained on the training subset.
- ✓ The model is then evaluated on the testing subset using metrics such as mean squared error or R-squared value to assess its performance.

6 Model Building

6.1 Preprocessing of Data:

- ✓ Data preprocessing is an essential step in data analysis and involves preparing data for analysis by cleaning, transforming, and reducing its dimensionality.

  < Dimensionality:It is the no of independent variable that are used to predict the dependent variable, which is loan status in this case.>

➔ Preprocessing of actual data involves of following steps as : CSV, SQL, Excel.

Introduction of CSV

- ✓ A CSV (Comma-Separated Values) file is a type of file format used to store and exchange tabular data, such as spreadsheets or databases.
- ✓ As the name suggests, the values in a CSV file are separated by commas, and each row represents a record or data point.
- ✓ CSV files are a popular way to exchange data between different applications and systems.
- ✓ It is easy to read and write in a text editor and can be easily imported into spreadsheet software such as Microsoft Excel or Google Sheets.
- ✓ In a CSV file, each line typically represents a single record or data point, and each field or column is separated by a comma.
- ✓ For example, a CSV file containing data on loan applications might have columns for the loan amount, the borrower's credit score, the loan status, and other relevant factors.

Introduction of SQL

- ✓ It stands for Structured Query Language.
- ✓ It is a programming language used for managing and manipulating relational databases.
- ✓ It is a standard language used by many relational database management systems (RDBMS), including MySQL, Oracle, Microsoft SQL Server, and PostgreSQL.
- ✓ SQL is used for a variety of tasks related to managing data in relational databases, including:
    - ➔ Creating and modifying database tables
    - ➔ Inserting and updating data:
    - ➔ Querying data: SQL can be used to retrieve data from one or more tables in a database using queries or searches.
    - ➔ Managing database transactions: SQL can be used to manage transactions, which are sets of database operations that need to be completed together in a consistent and reliable way.

Introduction of Excel

- ✓ Excel is a powerful spreadsheet software developed by Microsoft that is widely used for data analysis and modeling.
- ✓ It provides a range of tools and functions that make it easy to manage and analyze data in a structured way.
- ✓ In the context of loan status prediction, Excel can be used for a variety of tasks, including:

    a. Data Preprocessing
    b. Exploratory data analysis
    c. Statistical analysis
    d. Model building and evolution

- ❖ Reading data from CSV(comma separated values) is a fundamental necessity in Data Science. Often,
- ❖ we get data from various sources which can get exported to CSV format so that they can be used by other systems.
- ❖ The Panadas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

GETTING THE DATASET:

We can get the data from client. We can get the data from database like Sql, Excel And CSV.

https://www.kaggle.com/datasets/ninzaami/loan-predication?resource=download

6.1.1 Importing the Libraries

We have to import the libraries as per the requirement of the algorithm:

```
import pyodbc
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sb
```

Path of Sql Connectivity

```
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=DESKTOP-9LI0S2S\SQLEXPRESS;'
                      'Database=master;'
                      'Trusted_Connection=yes;')


    cursor = conn.cursor()
    cursor.execute('SELECT * FROM dbo.loan_Status_prediction')
```

- For row in cursor: print(row)
- Data Convert it into CSV And Excel File so that process is easily done.
- Now Rename a CSV file: LoanApprovalPrediction.csv

6.1.2 Importing the Data-set

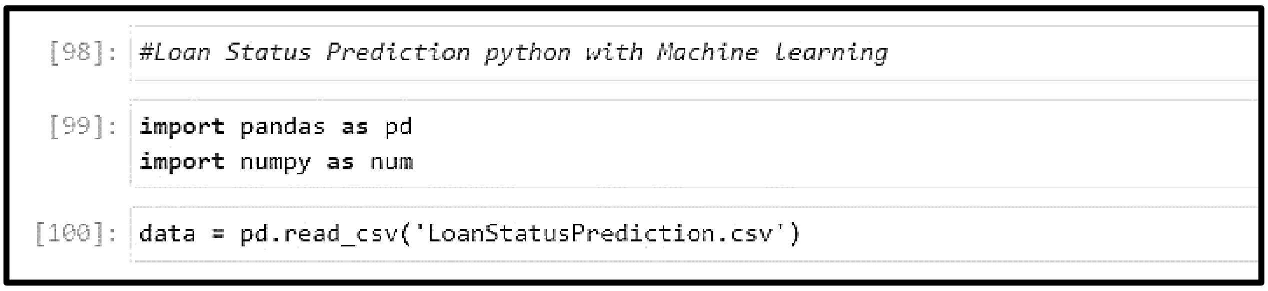
- Pandas in python provide an interesting method read_csv().
- The read_csv function reads the entire dataset from a comma separated values file.
- We can assign Pandas Data Frame to which all the operations can be performed. It helps us to access each and every row as well as columns and each and every value can be access using the data frame. Any missing value or NaN value have to be cleaned.

Attributes of a Data Set Export to Sql:

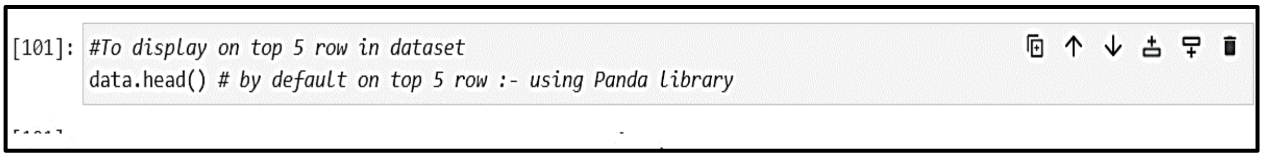
Variable Name	Description
# Loan_ID	Unique Loan ID
# Gender	Male/ Female/Transgender
# Married	Applicant married (Y/N)
# Dependents	Number of dependents (Member)
# Education	(Graduate/ Under Graduate)
# Self_Employed	Self employed(Y/N)
# ApplicantIncome	Applicant income(Monthly)
# CoapplicantIncome	Additional applicant’s income
# LoanAmount	Loan amount in thousands of dollars
# Loan_Amount_Term	Term of loan in months
# Credit_History	Credit history meets guidelines yes or no
# Property_Area	Urban/ Semi Urban/ Rural
# Loan_Status	Acceptable(Y) & Not-acceptable (N)

6.1.3 Reading the dataset Using Jupyter (Notebook)

❖ Loan Status Prediction python with Machine Learning

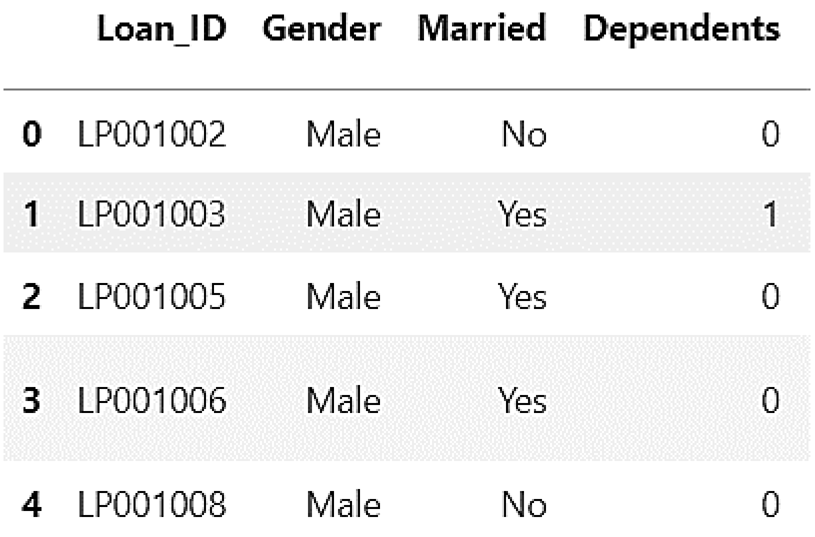


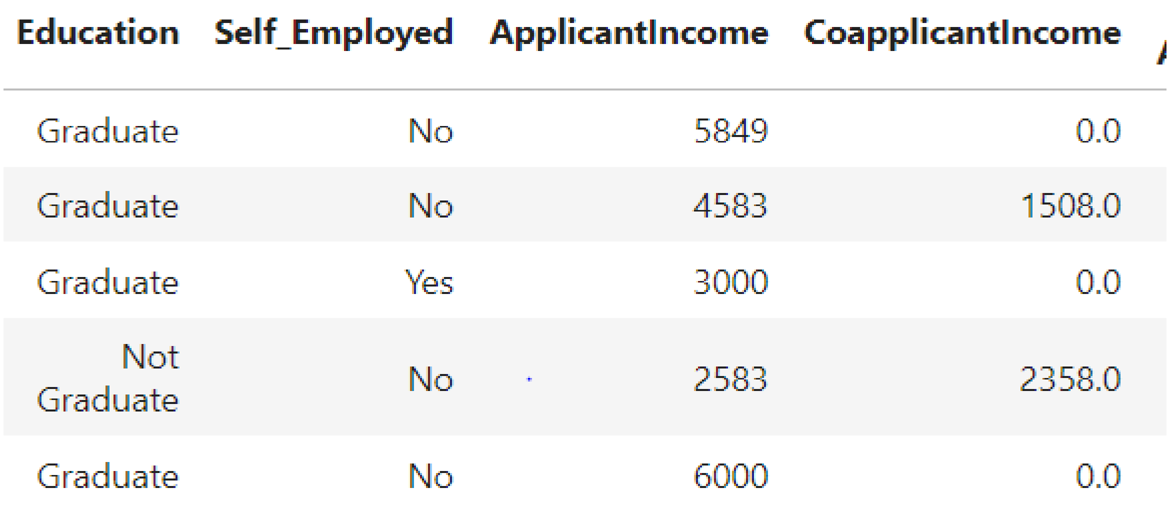
```
[98]:  #Loan Status Prediction python with Machine Learning

[99]:  import pandas as pd
       import numpy as num

[100]: data = pd.read_csv('LoanStatusPrediction.csv')
```



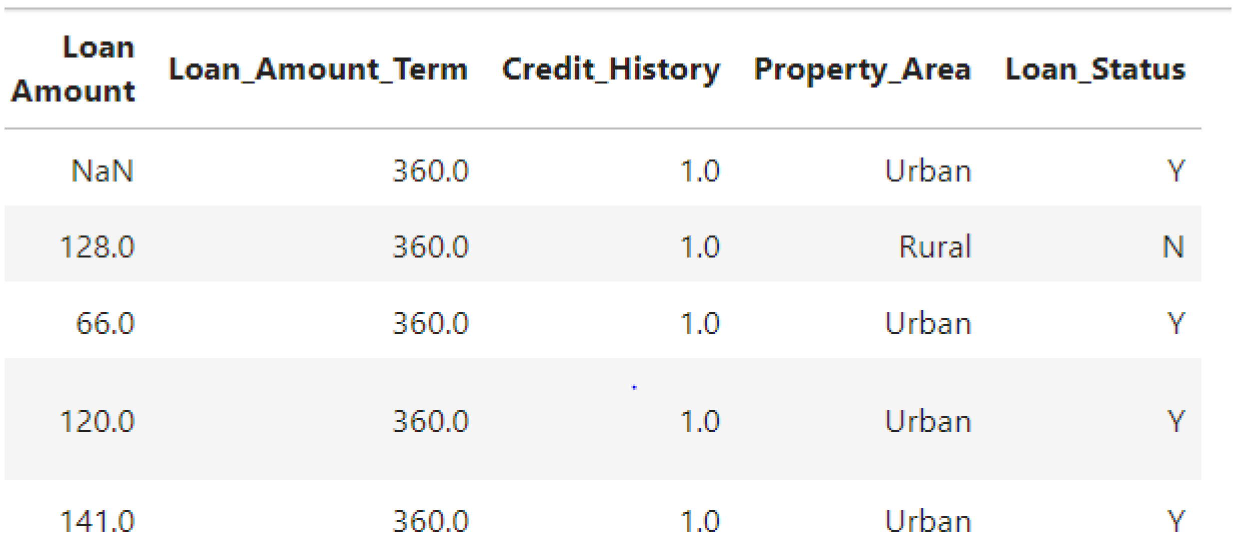
```
[101]: #To display on top 5 row in dataset
        data.head() # by default on top 5 row :- using Panda library
```

❖ Display on Top 5 Row in Dataset(default) : data.head( )



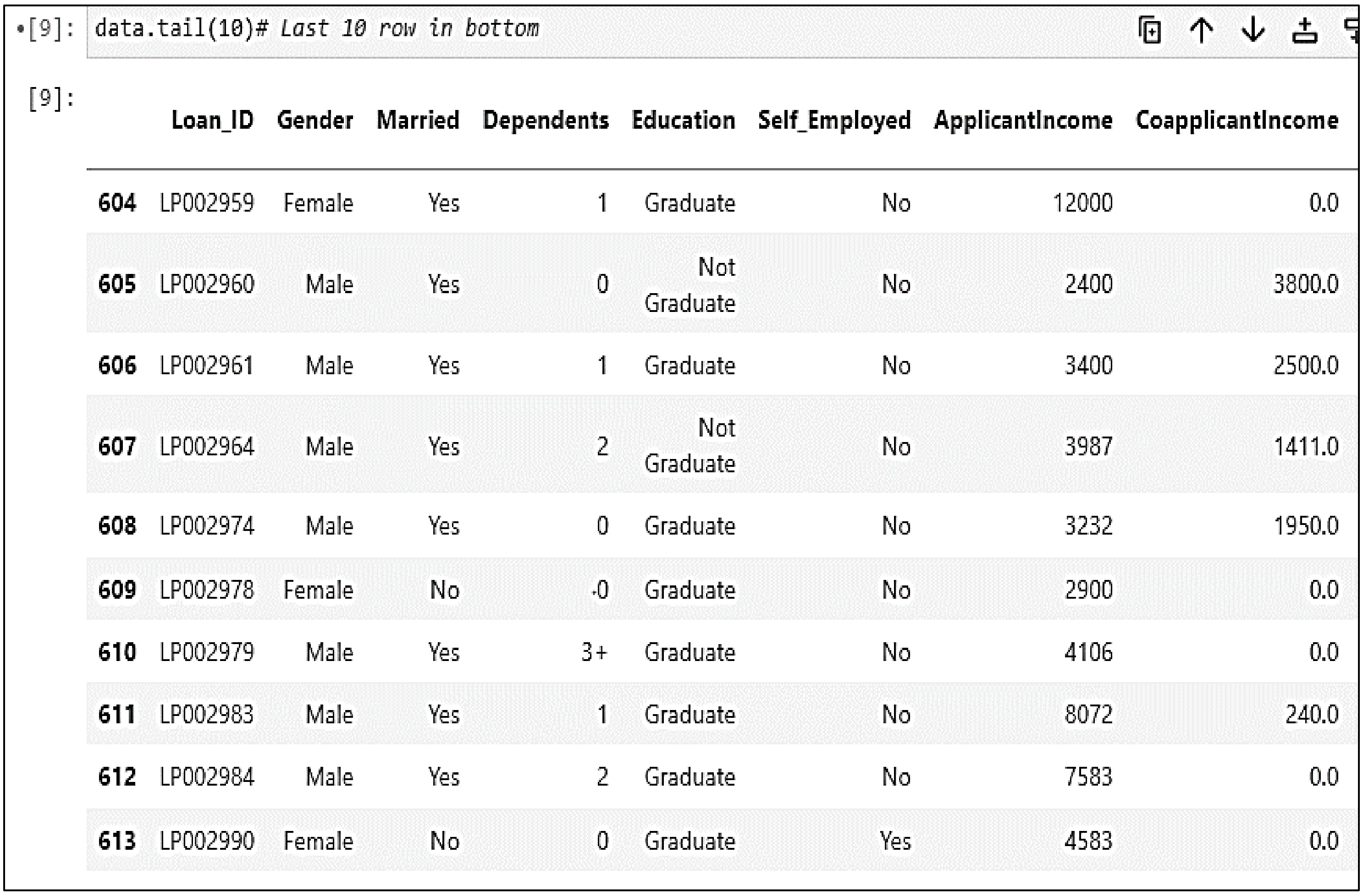


	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome
0	LP001002	Male	No	0	Graduate	No	5849	0.0
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0
4	LP001008	Male	No	0	Graduate	No	6000	0.0



Loan Amount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
NaN	360.0	1.0	Urban	Y
128.0	360.0	1.0	Rural	N
66.0	360.0	1.0	Urban	Y
120.0	360.0	1.0	Urban	Y
141.0	360.0	1.0	Urban	Y

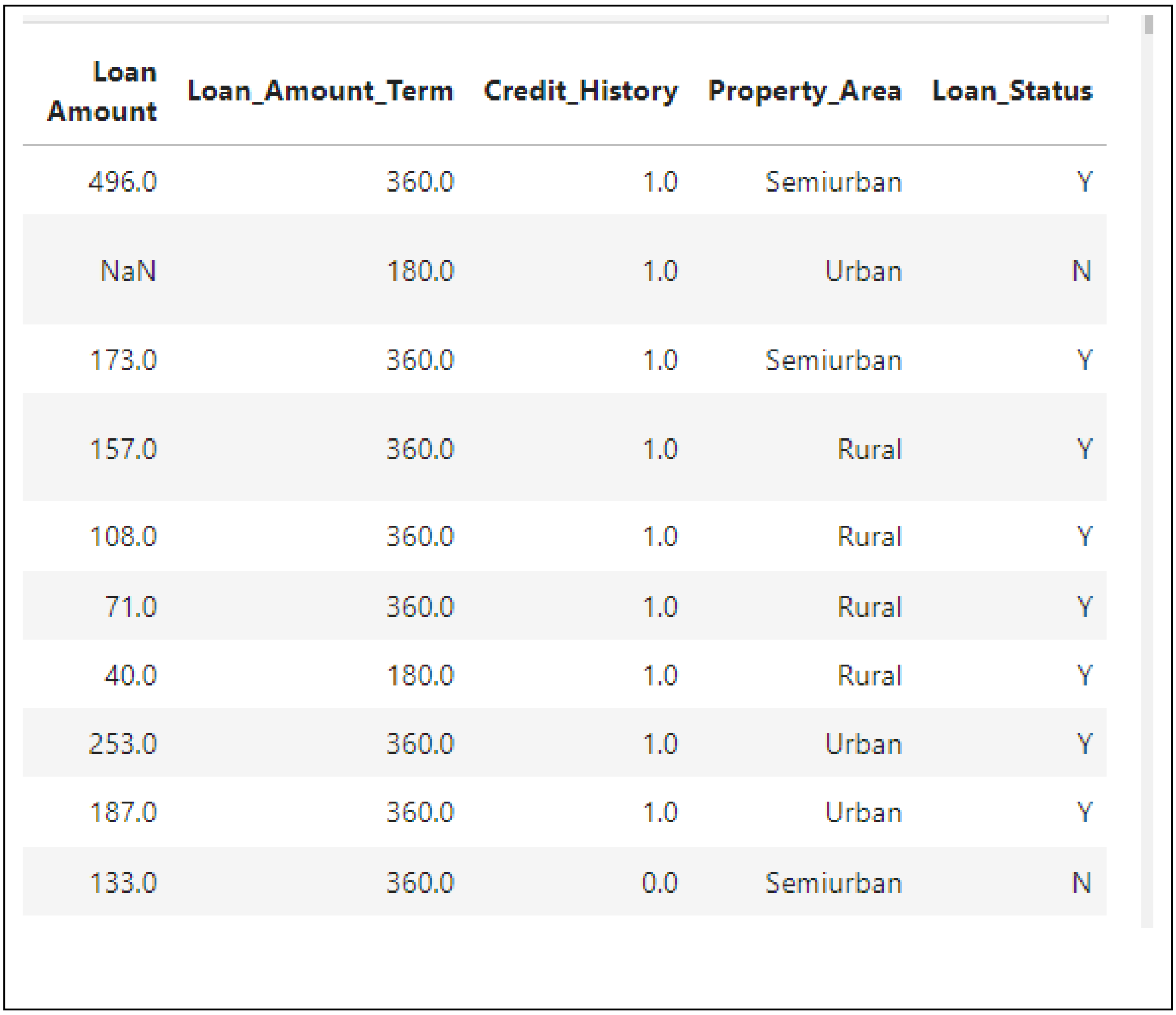
❖ Display on Bottom 10 Row in Dataset are used in - data.tail(10).



```
•[9]: data.tail(10)# Last 10 row in bottom
```

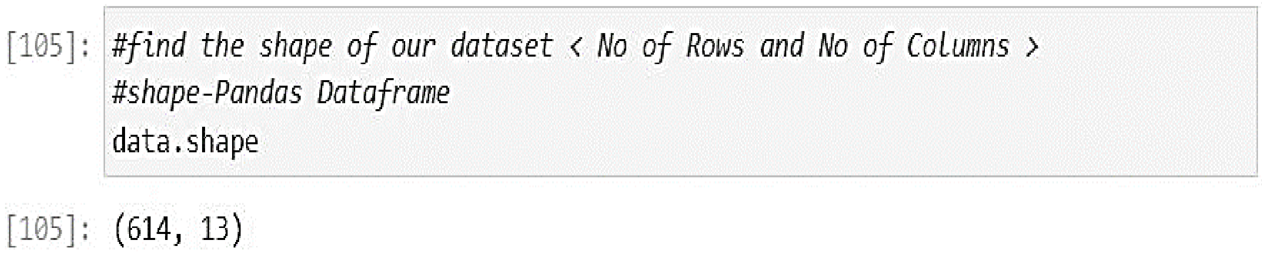
[9]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome
604	LP002959	Female	Yes	1	Graduate	No	12000	0.0
605	LP002960	Male	Yes	0	Not Graduate	No	2400	3800.0
606	LP002961	Male	Yes	1	Graduate	No	3400	2500.0
607	LP002964	Male	Yes	2	Not Graduate	No	3987	1411.0
608	LP002974	Male	Yes	0	Graduate	No	3232	1950.0
609	LP002978	Female	No	0	Graduate	No	2900	0.0
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0

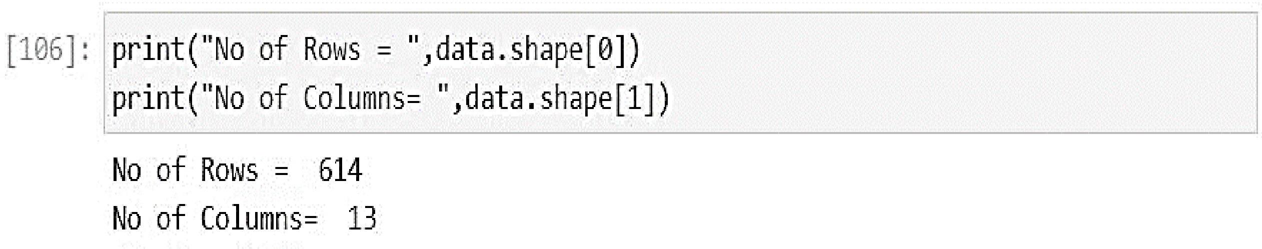


Loan Amount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
496.0	360.0	1.0	Semiurban	Y
NaN	180.0	1.0	Urban	N
173.0	360.0	1.0	Semiurban	Y
157.0	360.0	1.0	Rural	Y
108.0	360.0	1.0	Rural	Y
71.0	360.0	1.0	Rural	Y
40.0	180.0	1.0	Rural	Y
253.0	360.0	1.0	Urban	Y
187.0	360.0	1.0	Urban	Y
133.0	360.0	0.0	Semiurban	N

❖ Find the shape of our Dataset : data.shape

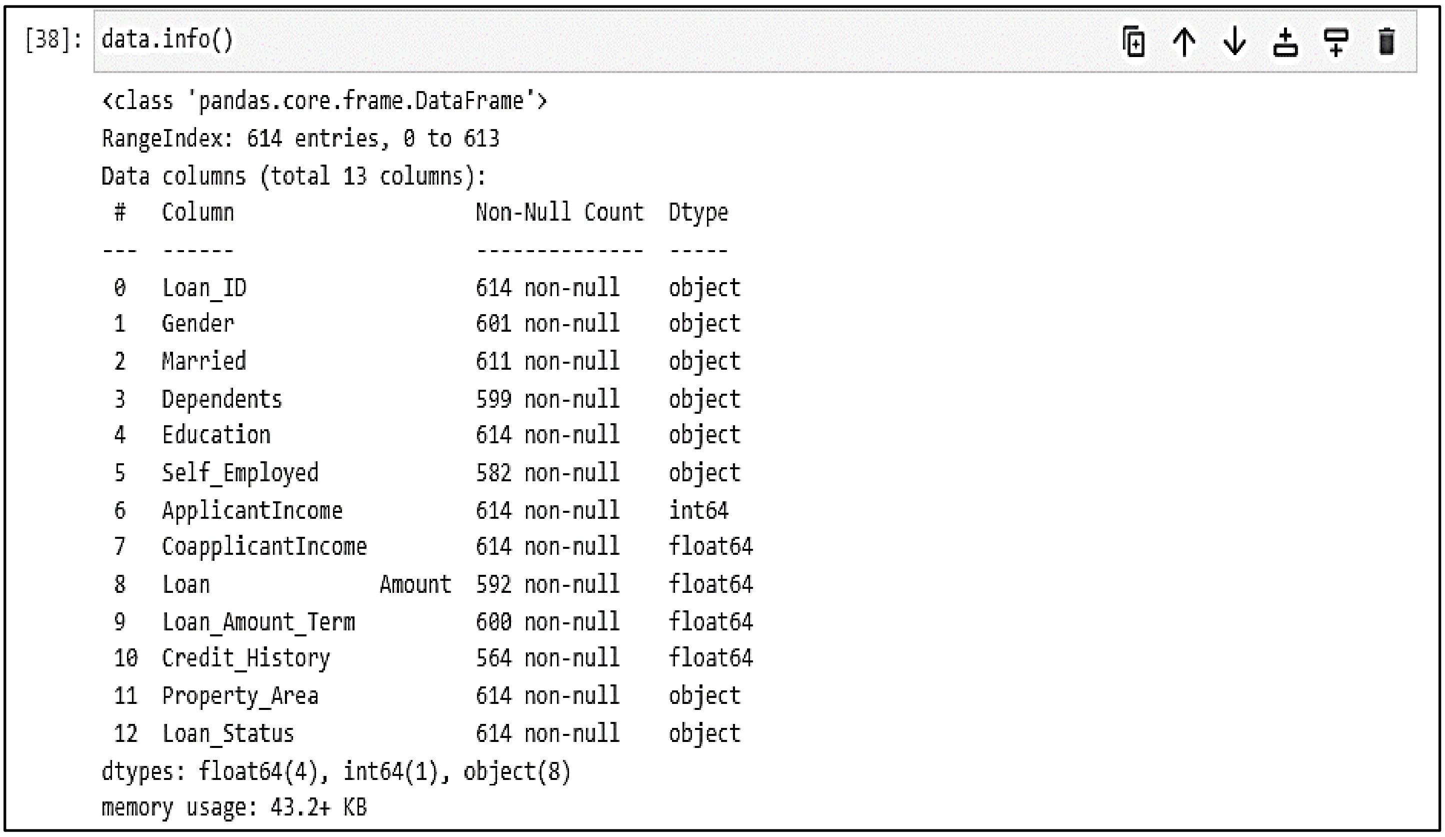


```
[105]: #find the shape of our dataset < No of Rows and No of Columns >
       #shape-Pandas Dataframe
       data.shape

[105]: (614, 13)
```



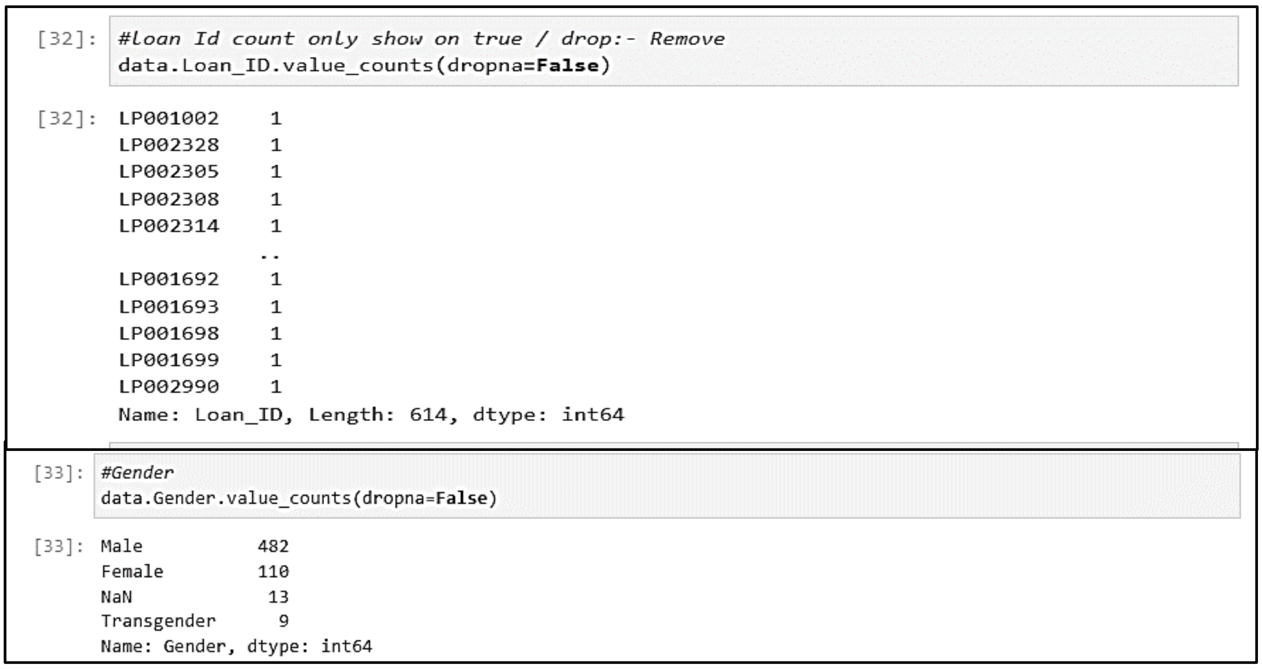
```
[106]: print("No of Rows = ",data.shape[0])
       print("No of Columns= ",data.shape[1])

       No of Rows =  614
       No of Columns=  13
```

❖ Find the Information of our Dataset



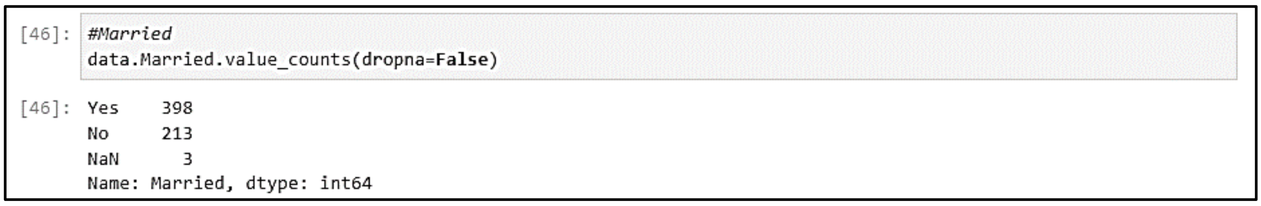
```
[38]: data.info()

      <class 'pandas.core.frame.DataFrame'>
      RangeIndex: 614 entries, 0 to 613
      Data columns (total 13 columns):
       #   Column             Non-Null Count  Dtype
      ---  ------             --------------  -----
       0   Loan_ID            614 non-null    object
       1   Gender             601 non-null    object
       2   Married            611 non-null    object
       3   Dependents         599 non-null    object
       4   Education          614 non-null    object
       5   Self_Employed      582 non-null    object
       6   ApplicantIncome    614 non-null    int64
       7   CoapplicantIncome  614 non-null    float64
       8   Loan        Amount 592 non-null    float64
       9   Loan_Amount_Term   600 non-null    float64
       10  Credit_History     564 non-null    float64
       11  Property_Area      614 non-null    object
       12  Loan_Status        614 non-null    object
      dtypes: float64(4), int64(1), object(8)
      memory usage: 43.2+ KB
```

❖ Count Loan_Id ,Gender value : **data.Loan_ID.value_counts(dropna=False)**



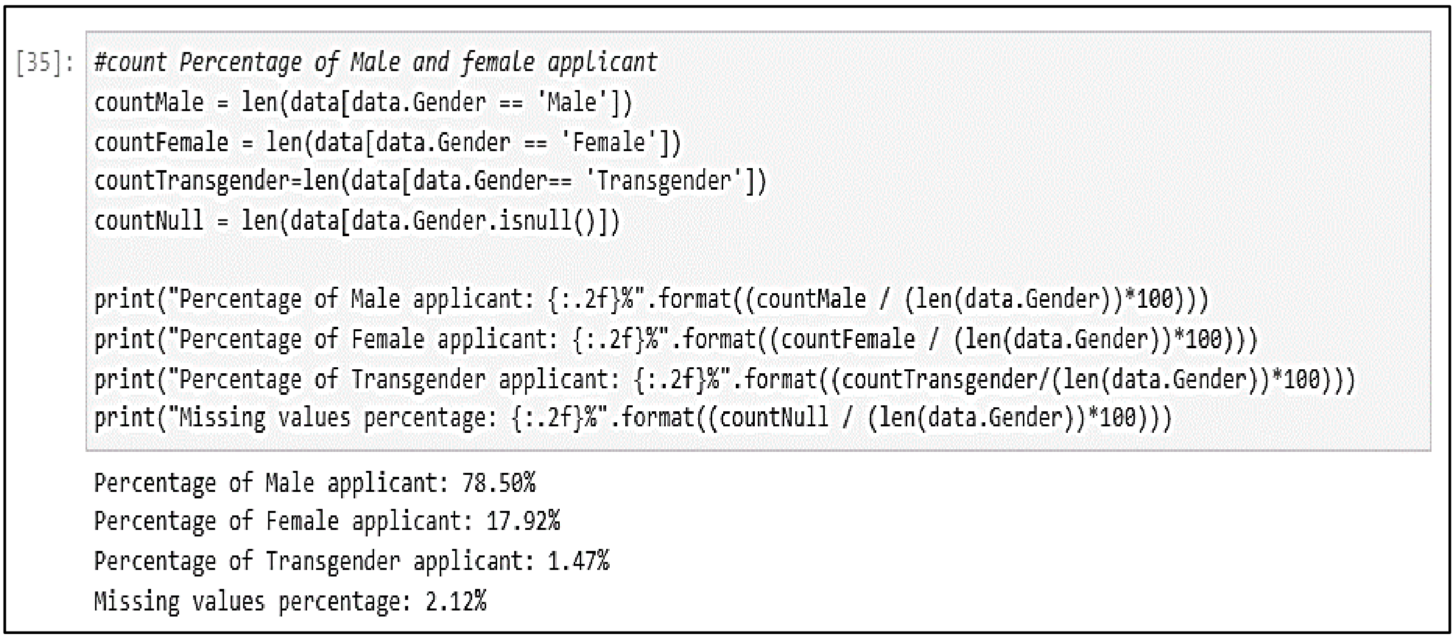
```
[32]: #loan Id count only show on true / drop:- Remove
      data.Loan_ID.value_counts(dropna=False)

[32]: LP001002    1
      LP002328    1
      LP002305    1
      LP002308    1
      LP002314    1
                 ..
      LP001692    1
      LP001693    1
      LP001698    1
      LP001699    1
      LP002990    1
      Name: Loan_ID, Length: 614, dtype: int64
```

```
[33]: #Gender
      data.Gender.value_counts(dropna=False)

[33]: Male           482
      Female         110
      NaN             13
      Transgender      9
      Name: Gender, dtype: int64
```



```
[46]: #Married
      data.Married.value_counts(dropna=False)

[46]: Yes    398
      No     213
      NaN      3
      Name: Married, dtype: int64
```

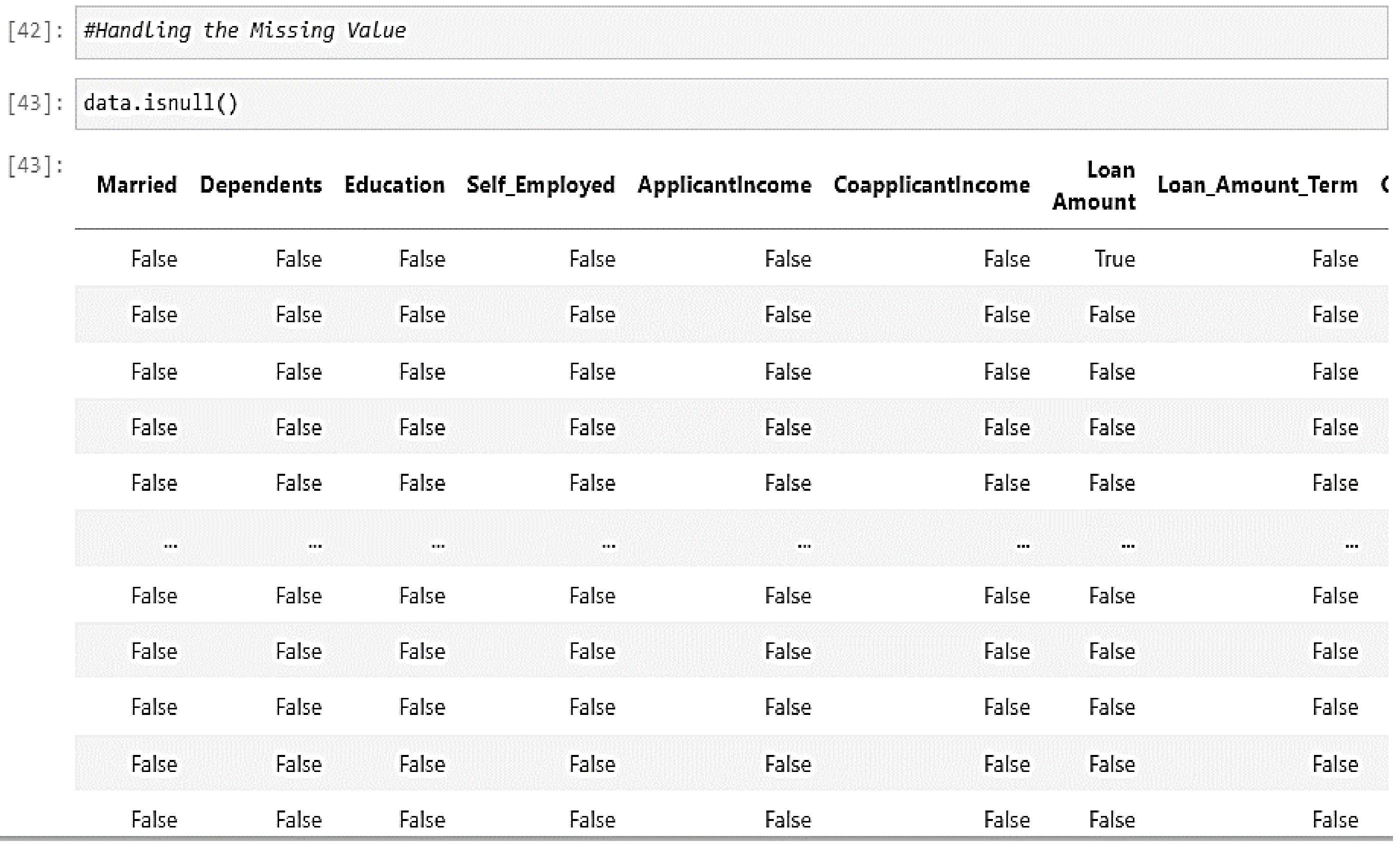
❖ Find the percentage of Male, Female and Transgender:



```
[35]: #count Percentage of Male and female applicant
      countMale = len(data[data.Gender == 'Male'])
      countFemale = len(data[data.Gender == 'Female'])
      countTransgender=len(data[data.Gender== 'Transgender'])
      countNull = len(data[data.Gender.isnull()])

      print("Percentage of Male applicant: {:.2f}%".format((countMale / (len(data.Gender))*100)))
      print("Percentage of Female applicant: {:.2f}%".format((countFemale / (len(data.Gender))*100)))
      print("Percentage of Transgender applicant: {:.2f}%".format((countTransgender/(len(data.Gender))*100)))
      print("Missing values percentage: {:.2f}%".format((countNull / (len(data.Gender))*100)))

      Percentage of Male applicant: 78.50%
      Percentage of Female applicant: 17.92%
      Percentage of Transgender applicant: 1.47%
      Missing values percentage: 2.12%
```

6.1.4 Handling the Missing values

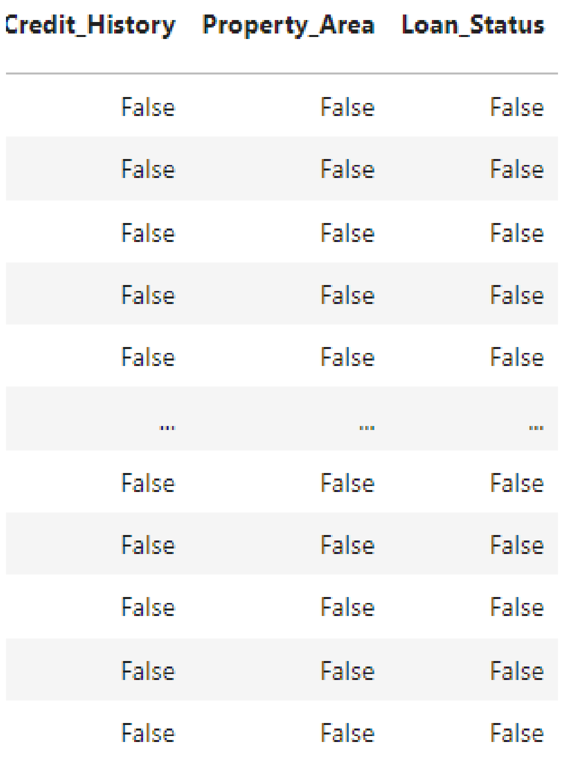
❖ Check null value of dataset of all columns :data.isnull()



```
[42]: #Handling the Missing Value
```

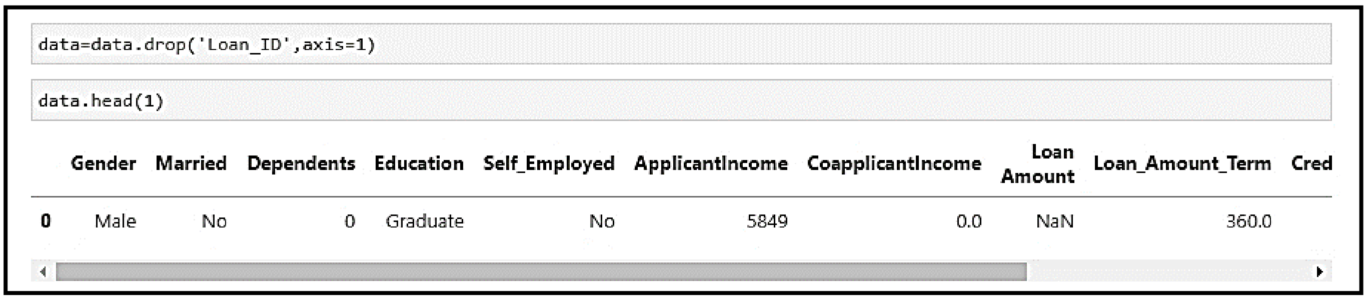
```
[43]: data.isnull()
```

[43]:



Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Loan Amount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
False	False	False	False	False	False	True	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False

❖ Drop of Loan_Id completely unique and not correlated with any of the other column, So we will drop it using .drop() function.

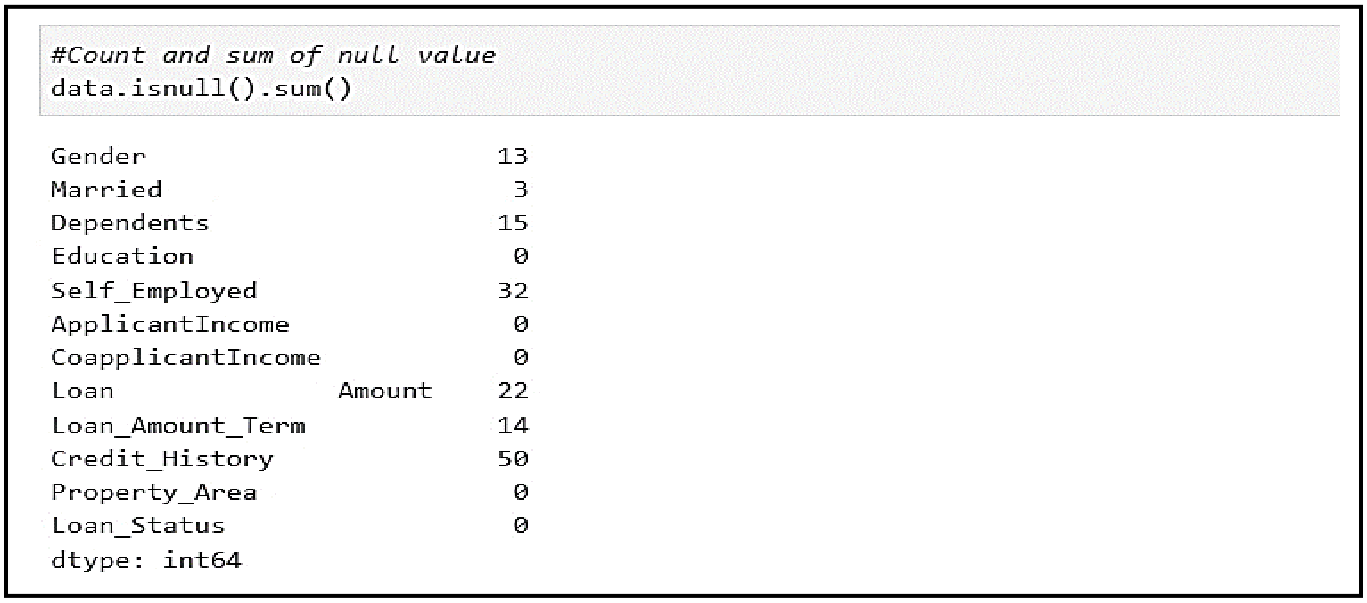


```
data=data.drop('Loan_ID',axis=1)
```

```
data.head(1)
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Loan Amount	Loan_Amount_Term	Cred
0	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	

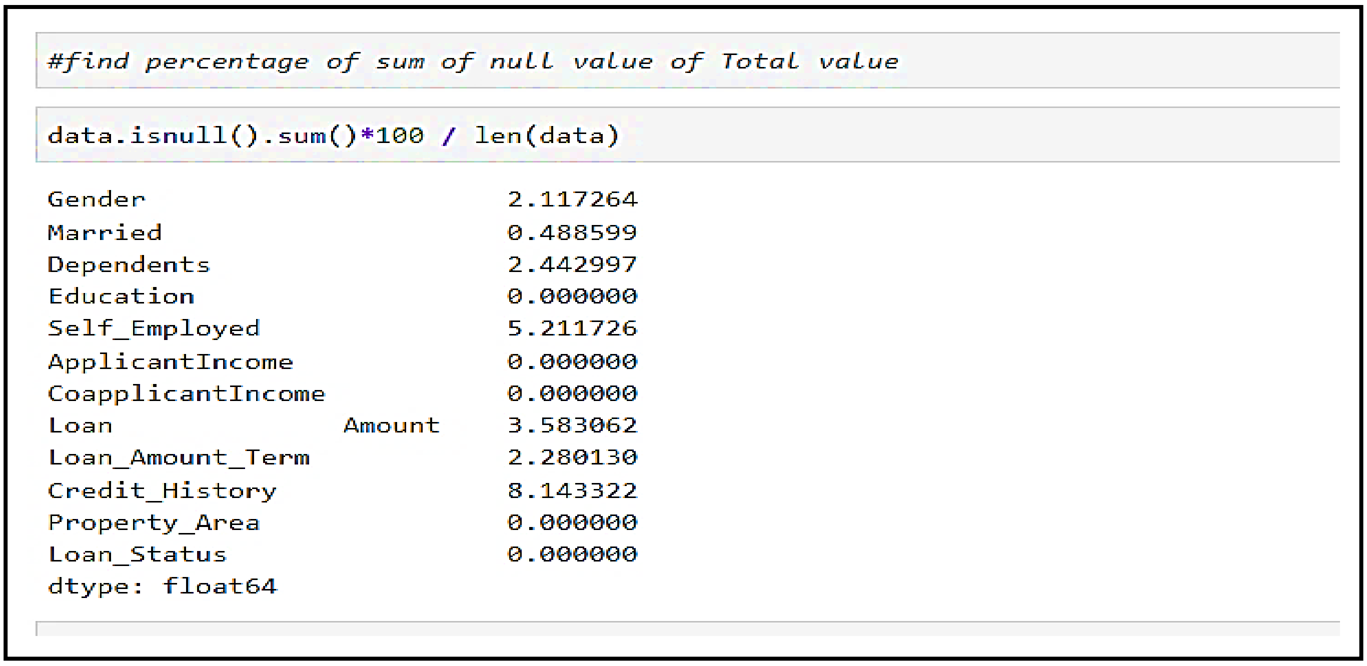
❖ Count and Sum of value in all columns



```
#Count and sum of null value
data.isnull().sum()
```

```
Gender                  13
Married                  3
Dependents              15
Education                0
Self_Employed           32
ApplicantIncome          0
CoapplicantIncome        0
Loan               Amount   22
Loan_Amount_Term        14
Credit_History          50
Property_Area            0
Loan_Status              0
dtype: int64
```

❖ Find percentage of sum of null value of Total value (Columns)

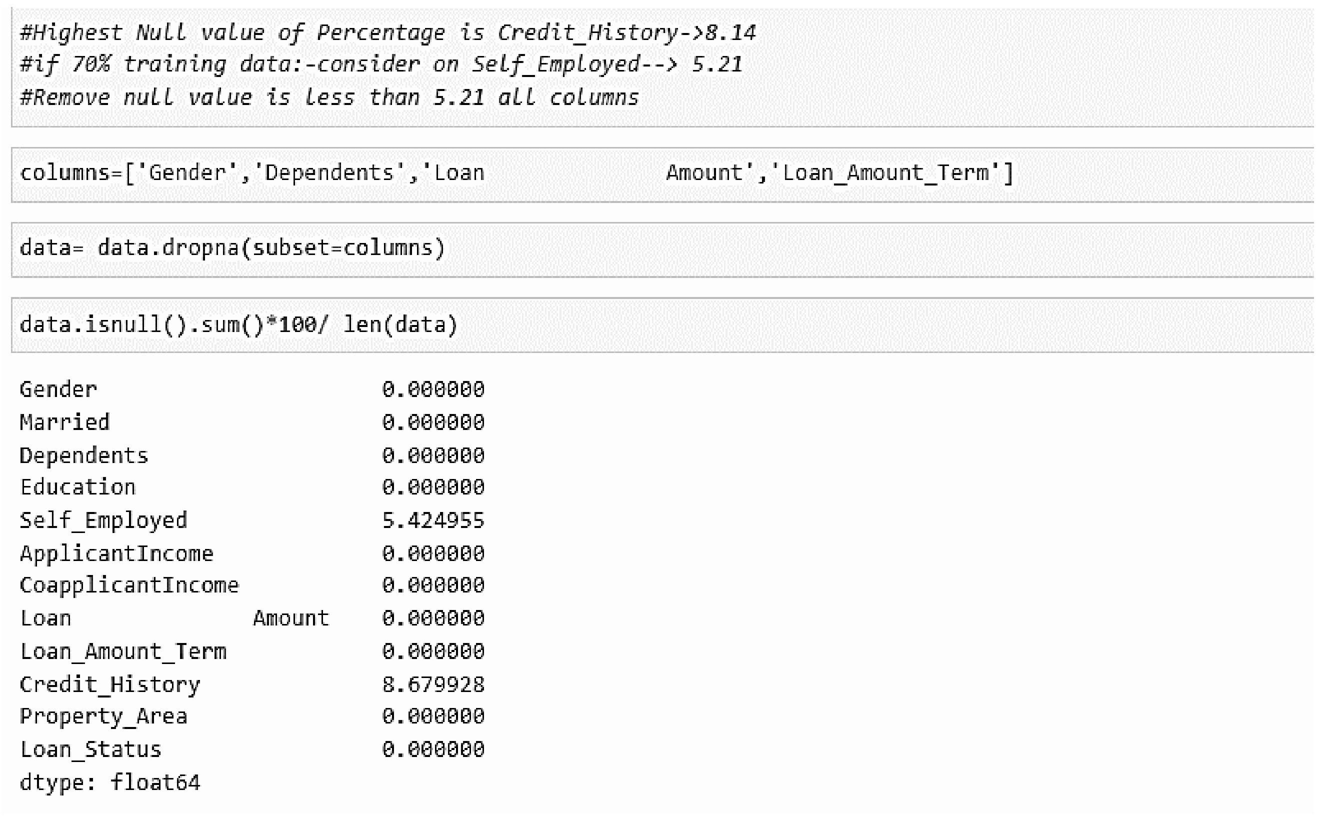


```
#find percentage of sum of null value of Total value
```

```
data.isnull().sum()*100 / len(data)
```

```
Gender                    2.117264
Married                   0.488599
Dependents                2.442997
Education                 0.000000
Self_Employed             5.211726
ApplicantIncome           0.000000
CoapplicantIncome         0.000000
Loan               Amount     3.583062
Loan_Amount_Term          2.280130
Credit_History            8.143322
Property_Area             0.000000
Loan_Status               0.000000
dtype: float64
```

❖ Remove null value for all columns (Training Data <75%)



```
#Highest Null value of Percentage is Credit_History->8.14
#if 70% training data:-consider on Self_Employed--> 5.21
#Remove null value is less than 5.21 all columns
```

```
columns=['Gender','Dependents','Loan             Amount','Loan_Amount_Term']
```

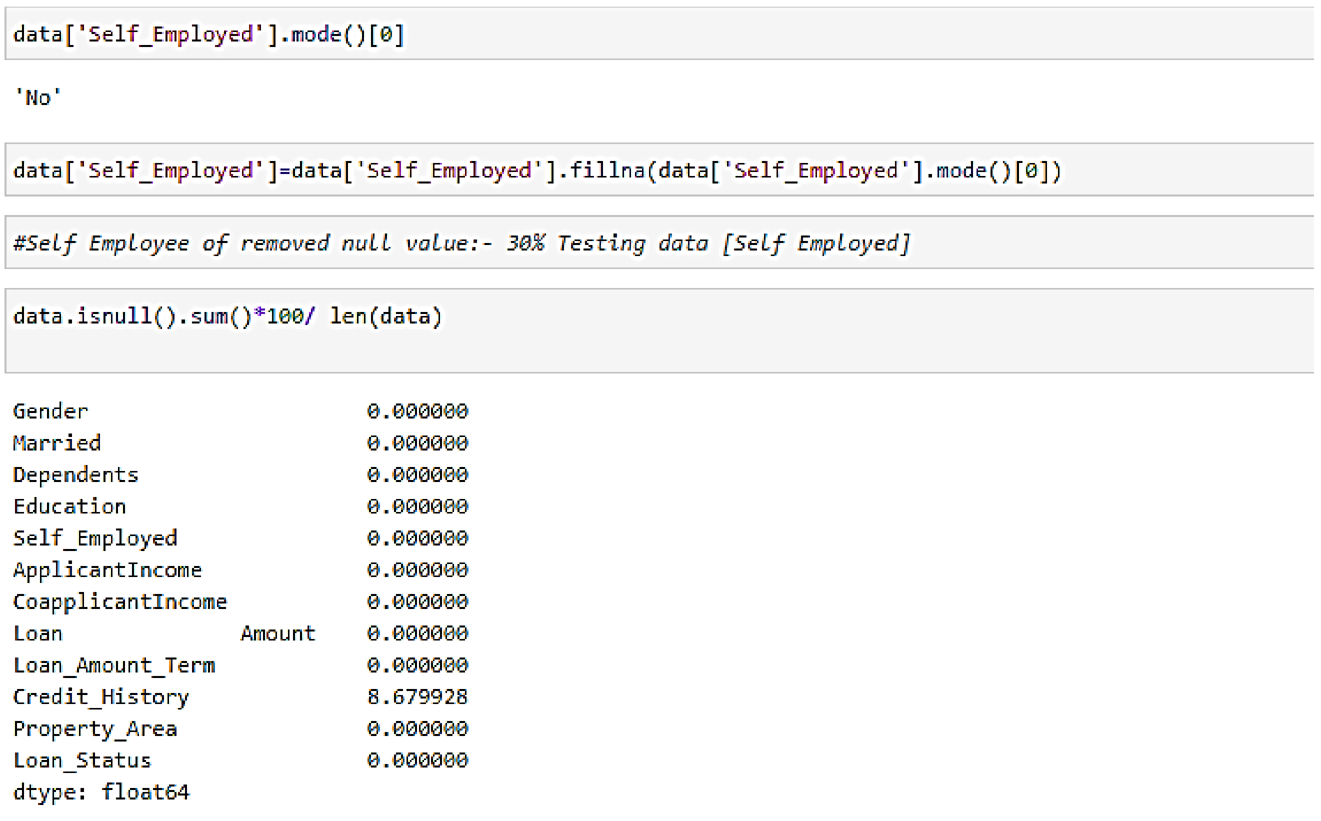
```
data= data.dropna(subset=columns)
```

```
data.isnull().sum()*100/ len(data)
```

```
Gender                      0.000000
Married                     0.000000
Dependents                  0.000000
Education                   0.000000
Self_Employed               5.424955
ApplicantIncome             0.000000
CoapplicantIncome           0.000000
Loan              Amount    0.000000
Loan_Amount_Term            0.000000
Credit_History              8.679928
Property_Area               0.000000
Loan_Status                 0.000000
dtype: float64
```

Remove null value: Gender, Dependents, LoanAmount, Loan_Amount_term

❖ Remove null value for Self_Employed, Credit_History(>75%)



```
data['Self_Employed'].mode()[0]
```

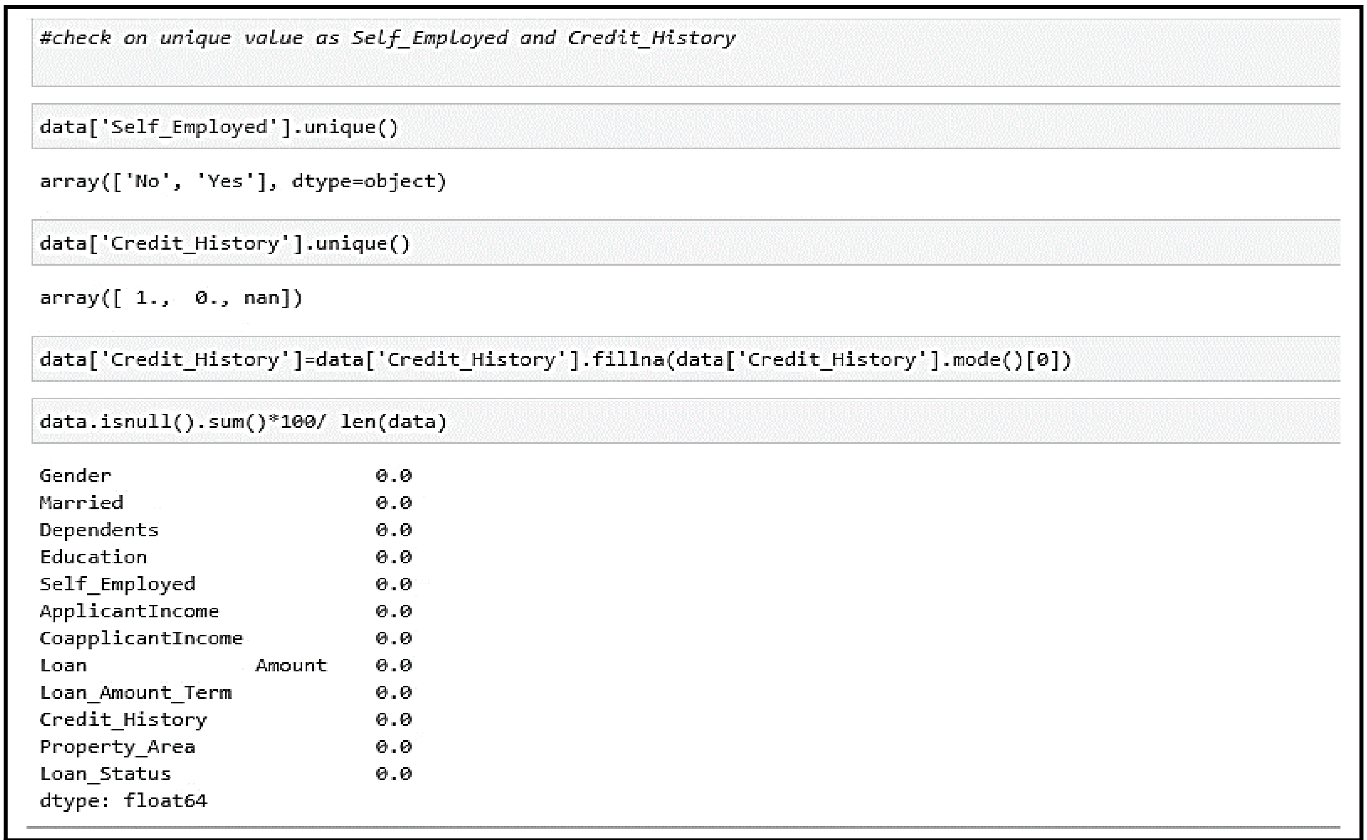
```
'No'
```

```
data['Self_Employed']=data['Self_Employed'].fillna(data['Self_Employed'].mode()[0])
```

```
#Self Employee of removed null value:- 30% Testing data [Self Employed]
```

```
data.isnull().sum()*100/ len(data)
```

```
Gender                     0.000000
Married                    0.000000
Dependents                 0.000000
Education                  0.000000
Self_Employed              0.000000
ApplicantIncome            0.000000
CoapplicantIncome          0.000000
Loan             Amount    0.000000
Loan_Amount_Term           0.000000
Credit_History             8.679928
Property_Area              0.000000
Loan_Status                0.000000
dtype: float64
```



```
#check on unique value as Self_Employed and Credit_History
```

```
data['Self_Employed'].unique()
```

```
array(['No', 'Yes'], dtype=object)
```

```
data['Credit_History'].unique()
```

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

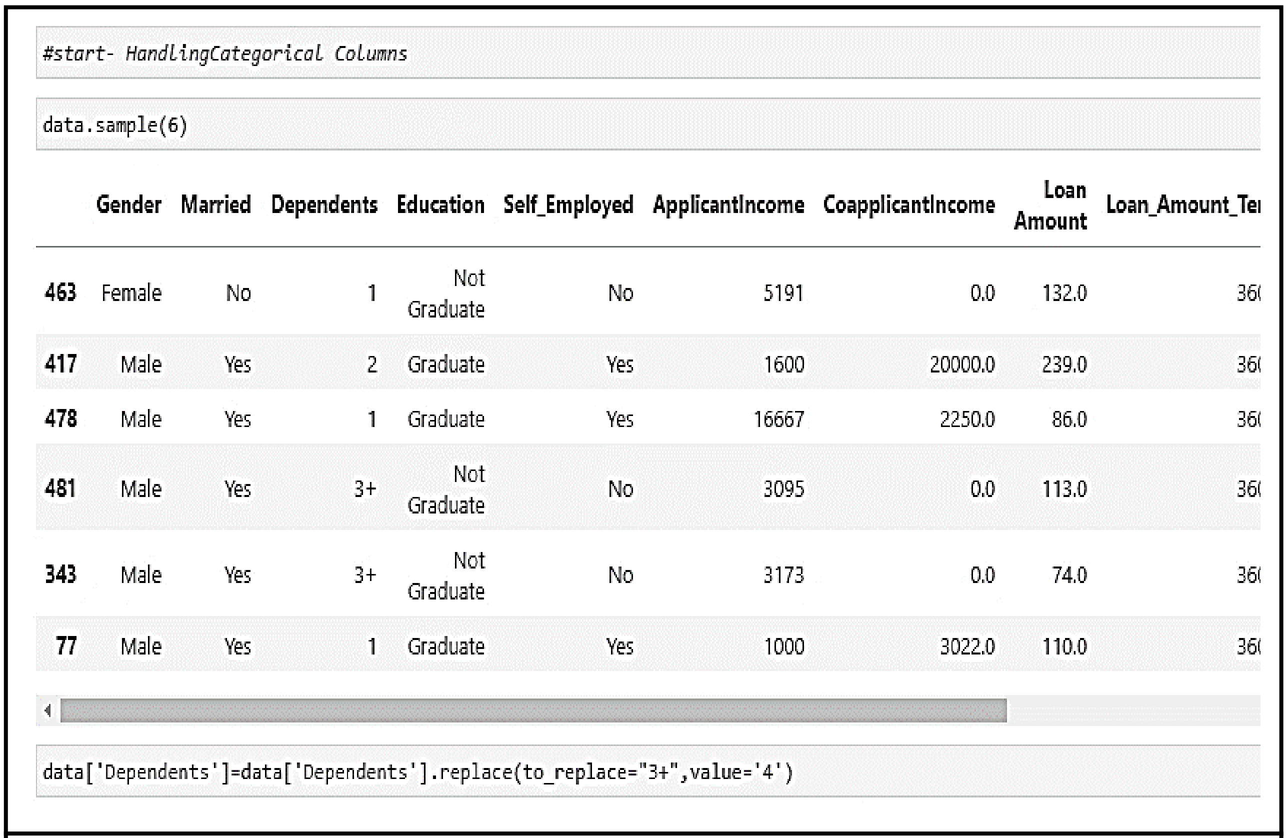
```
data['Credit_History']=data['Credit_History'].fillna(data['Credit_History'].mode()[0])
```

```
data.isnull().sum()*100/ len(data)
```

```
Gender                      0.0
Married                     0.0
Dependents                  0.0
Education                   0.0
Self_Employed               0.0
ApplicantIncome             0.0
CoapplicantIncome           0.0
Loan                 Amount 0.0
Loan_Amount_Term            0.0
Credit_History              0.0
Property_Area               0.0
Loan_Status                 0.0
dtype: float64
```

#Successfully Handling the Missing Value (Null Value)

6.1.5 Handling the Categorical Data

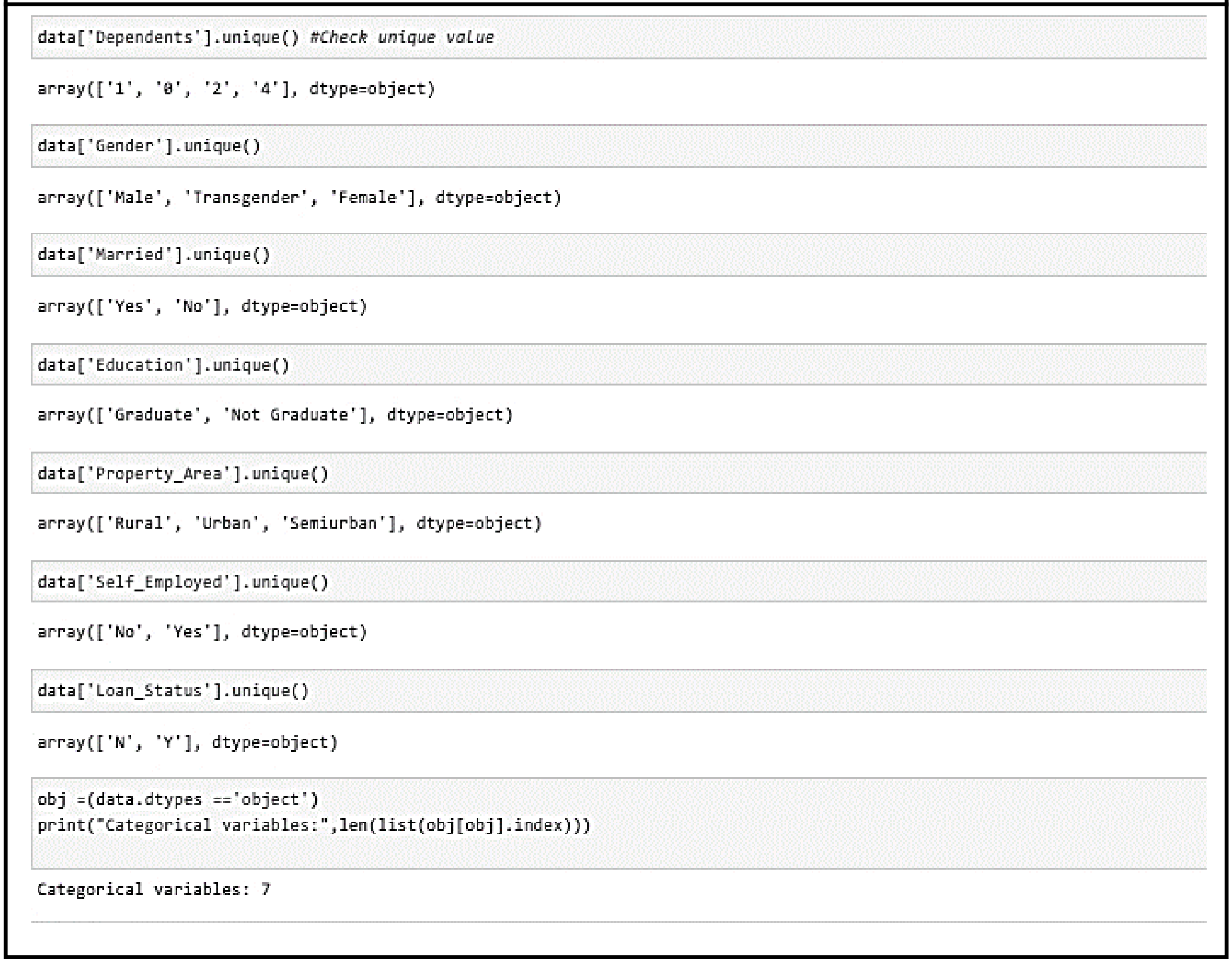


```
#start- HandlingCategorical Columns
```

```
data.sample(6)
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Loan Amount	Loan_Amount_Te
463	Female	No	1	Not Graduate	No	5191	0.0	132.0	36
417	Male	Yes	2	Graduate	Yes	1600	20000.0	239.0	36
478	Male	Yes	1	Graduate	Yes	16667	2250.0	86.0	36
481	Male	Yes	3+	Not Graduate	No	3095	0.0	113.0	36
343	Male	Yes	3+	Not Graduate	No	3173	0.0	74.0	36
77	Male	Yes	1	Graduate	Yes	1000	3022.0	110.0	36

```
data['Dependents']=data['Dependents'].replace(to_replace="3+",value='4')
```



```
data['Dependents'].unique() #Check unique value
```

```
array(['1', '0', '2', '4'], dtype=object)
```

```
data['Gender'].unique()
```

```
array(['Male', 'Transgender', 'Female'], dtype=object)
```

```
data['Married'].unique()
```

```
array(['Yes', 'No'], dtype=object)
```

```
data['Education'].unique()
```

```
array(['Graduate', 'Not Graduate'], dtype=object)
```

```
data['Property_Area'].unique()
```

```
array(['Rural', 'Urban', 'Semiurban'], dtype=object)
```

```
data['Self_Employed'].unique()
```

```
array(['No', 'Yes'], dtype=object)
```

```
data['Loan_Status'].unique()
```

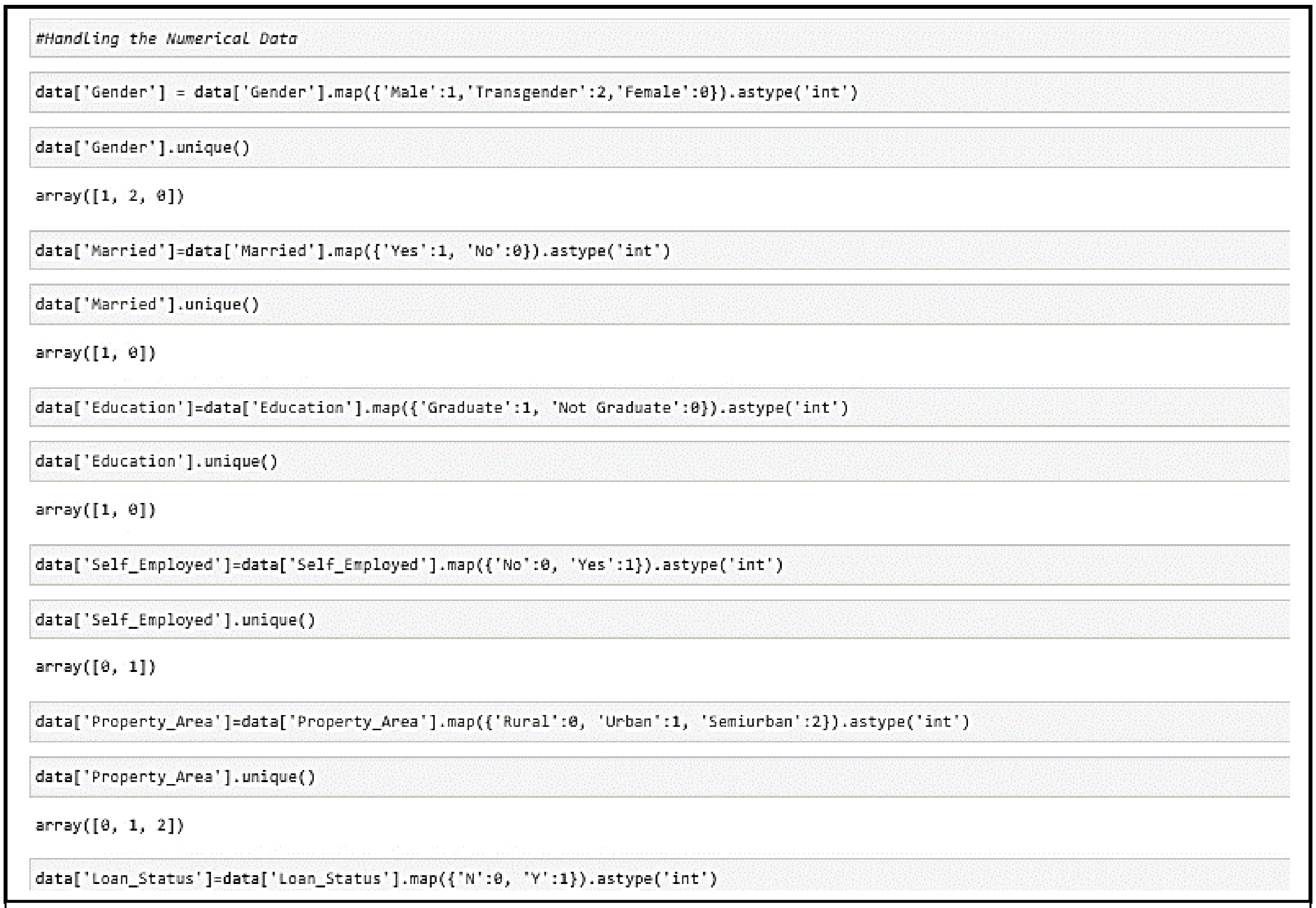
```
array(['N', 'Y'], dtype=object)
```

```
obj =(data.dtypes =='object')
print("Categorical variables:",len(list(obj[obj].index)))
```

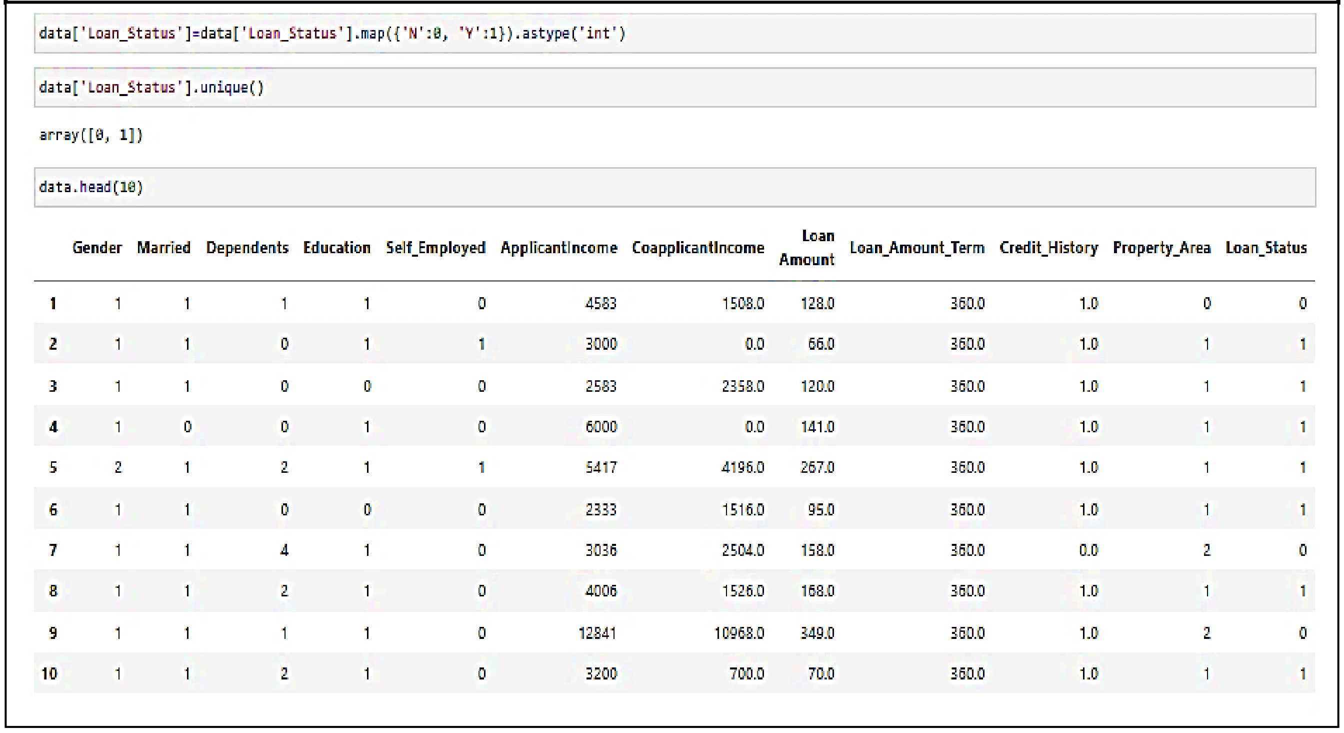
```
Categorical variables: 7
```

6.1.6 Handling the Numerical Data

<u>(Why: Machine Learning works as only numeric value)</u>



```
#Handling the Numerical Data

data['Gender'] = data['Gender'].map({'Male':1,'Transgender':2,'Female':0}).astype('int')

data['Gender'].unique()

array([1, 2, 0])

data['Married']=data['Married'].map({'Yes':1, 'No':0}).astype('int')

data['Married'].unique()

array([1, 0])

data['Education']=data['Education'].map({'Graduate':1, 'Not Graduate':0}).astype('int')

data['Education'].unique()

array([1, 0])

data['Self_Employed']=data['Self_Employed'].map({'No':0, 'Yes':1}).astype('int')

data['Self_Employed'].unique()

array([0, 1])

data['Property_Area']=data['Property_Area'].map({'Rural':0, 'Urban':1, 'Semiurban':2}).astype('int')

data['Property_Area'].unique()

array([0, 1, 2])

data['Loan_Status']=data['Loan_Status'].map({'N':0, 'Y':1}).astype('int')
```



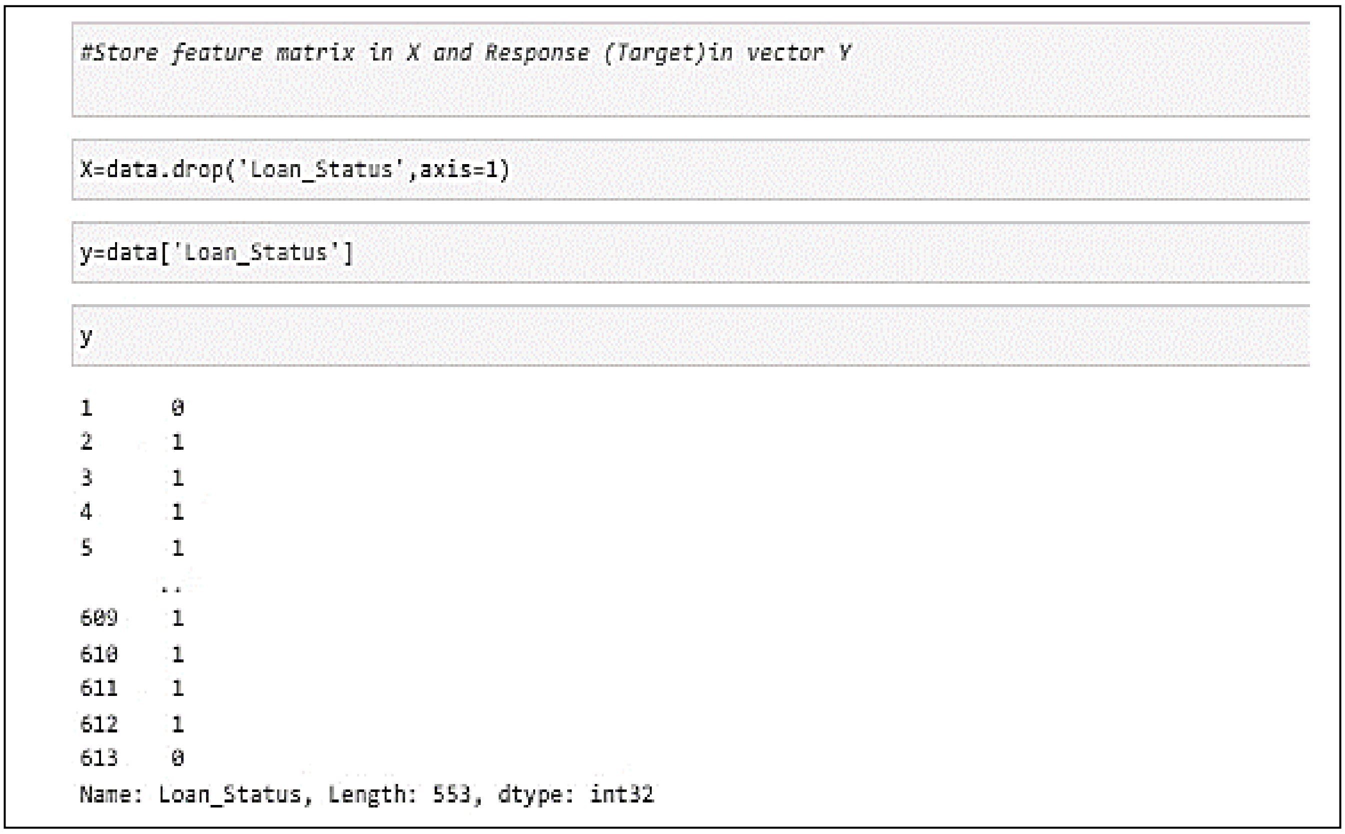
```
data['Loan_Status']=data['Loan_Status'].map({'N':0, 'Y':1}).astype('int')
```

```
data['Loan_Status'].unique()

array([0, 1])

data.head(10)
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Loan Amount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	1	1	1	1	0	4583	1508.0	128.0	360.0	1.0	0	0
2	1	1	0	1	1	3000	0.0	66.0	360.0	1.0	1	1
3	1	1	0	0	0	2583	2358.0	120.0	360.0	1.0	1	1
4	1	0	0	1	0	6000	0.0	141.0	360.0	1.0	1	1
5	2	1	2	1	1	5417	4196.0	267.0	360.0	1.0	1	1
6	1	1	0	0	0	2333	1516.0	95.0	360.0	1.0	1	1
7	1	1	4	1	0	3036	2504.0	158.0	360.0	0.0	2	0
8	1	1	2	1	0	4006	1526.0	168.0	360.0	1.0	1	1
9	1	1	1	1	0	12841	10968.0	349.0	360.0	1.0	2	0
10	1	1	2	1	0	3200	700.0	70.0	360.0	1.0	1	1

<u>(Successfully Handling Numeric Data)</u>

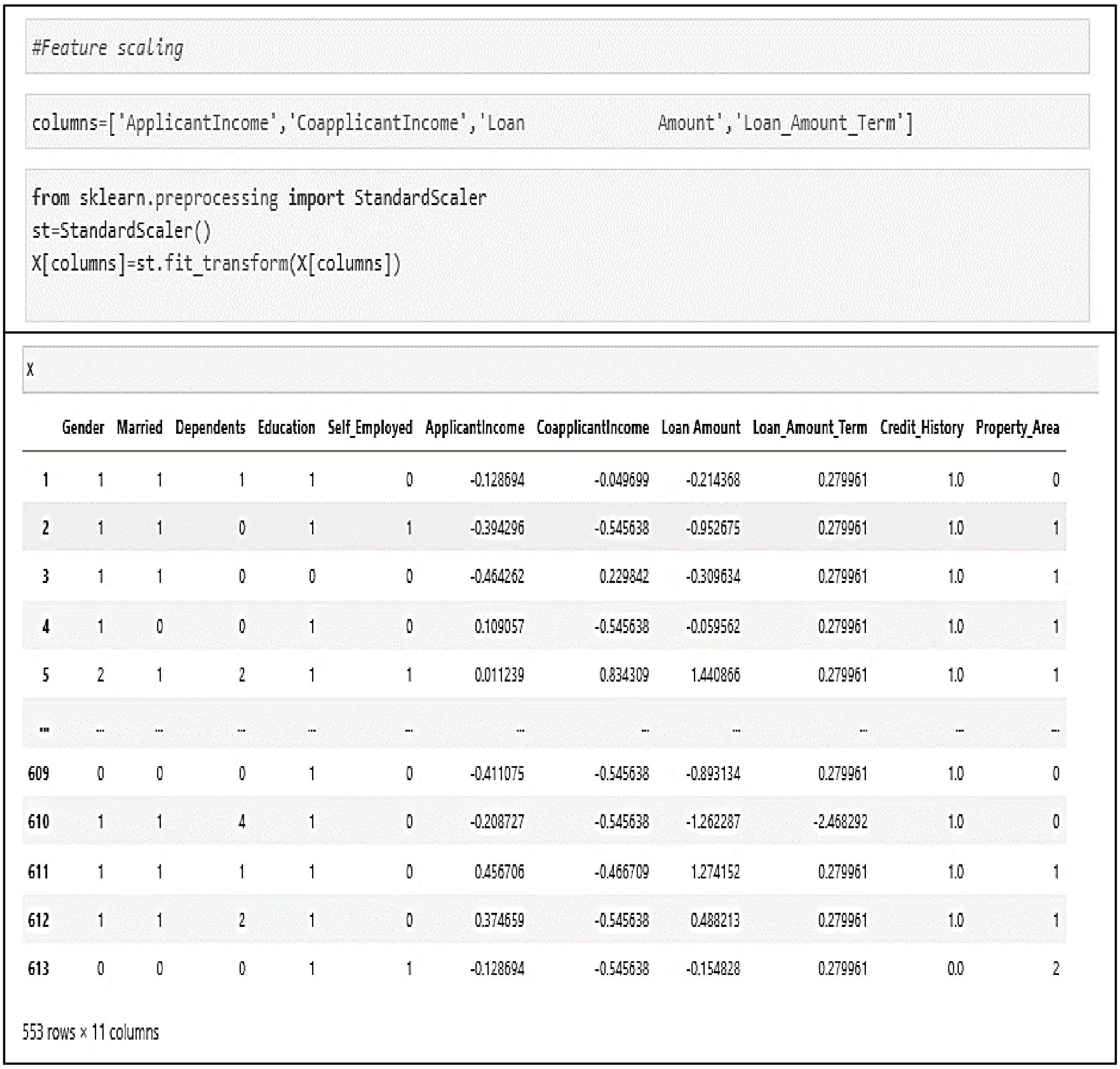
❖ Store feature matrix in X and Response (Target in vector y)


```
#Store feature matrix in X and Response (Target)in vector Y
```
```
X=data.drop('Loan_Status',axis=1)
```
```
y=data['Loan_Status']
```
```
y
```
```
1      0
2      1
3      1
4      1
5      1
      ..
609    1
610    1
611    1
612    1
613    0
Name: Loan_Status, Length: 553, dtype: int32
```

- ➔ How to predict a certain output based on some input data, we use a common approach of storing input data in matrix called X, where each row is different numeric value or observation and each column represents a different feature or characteristic of that observation.
- ➔ Using X and y, we can train a machine learning algorithm to learn the relationship between the input features in X and the target variable in y.
- ➔ The goal is to find a model that can accurately predict the target variable y for new, unseen examples in X.
- ➔ By training the algorithm on a labeled dataset of X and y, it can learn to make accurate predictions on new data that it has not seen before.

6.1.7 Feature Scaling

- ✓ Feature scaling is a preprocessing step in Machine learning where the input data is transformed to ensure that all features have similar scales or ranges.
- ✓ This is important because some machine learning algorithms are sensitive to scale of the input data, and features with larger scales can dominate the training process.



```
#Feature scaling
```

```
columns=['ApplicantIncome','CoapplicantIncome','Loan              Amount','Loan_Amount_Term']
```

```
from sklearn.preprocessing import StandardScaler
st=StandardScaler()
X[columns]=st.fit_transform(X[columns])
```

```
X
```

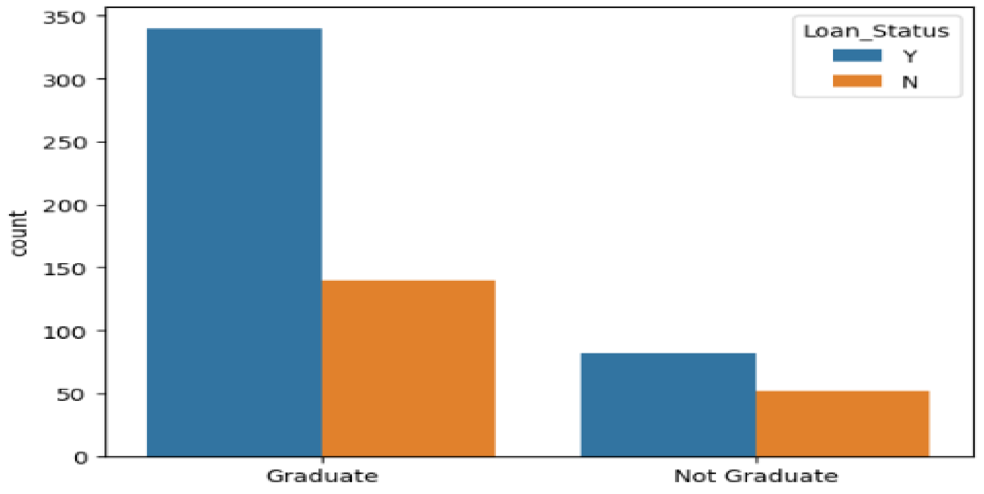
```
     Gender  Married  Dependents  Education  Self_Employed  ApplicantIncome  CoapplicantIncome  Loan Amount  Loan_Amount_Term  Credit_History  Property_Area
1         1        1           1          1              0        -0.128694          -0.049699    -0.214368          0.279961             1.0              0
2         1        1           0          1              1        -0.394296          -0.545638    -0.952675          0.279961             1.0              1
3         1        1           0          0              0        -0.464262           0.229842    -0.309634          0.279961             1.0              1
4         1        0           0          1              0         0.109057          -0.545638    -0.059562          0.279961             1.0              1
5         2        1           2          1              1         0.011239           0.834309     1.440866          0.279961             1.0              1
...     ...      ...         ...        ...            ...              ...                ...          ...               ...             ...            ...
609       0        0           0          1              0        -0.411075          -0.545638    -0.893134          0.279961             1.0              0
610       1        1           4          1              0        -0.208727          -0.545638    -1.262287         -2.468292             1.0              0
611       1        1           1          1              0         0.456706          -0.466709     1.274152          0.279961             1.0              1
612       1        1           2          1              0         0.374659          -0.545638     0.488213          0.279961             1.0              1
613       0        0           0          1              1        -0.128694          -0.545638    -0.154828          0.279961             0.0              2

553 rows × 11 columns
```

6.1.8 Data Visualization

- ✓ Seaborn and scikit-learn (sklearn) are both Python libraries used in machine learning and data analysis, but they serve different purposes.
- ✓ Seaborn is a data visualization library that provides a high-level interface for creating informative and attractive statistical graphics.

```
#Data Visualization

#Martial Status and loan status

sns.countplot(x='Education',hue='Loan_Status',data=data)
```

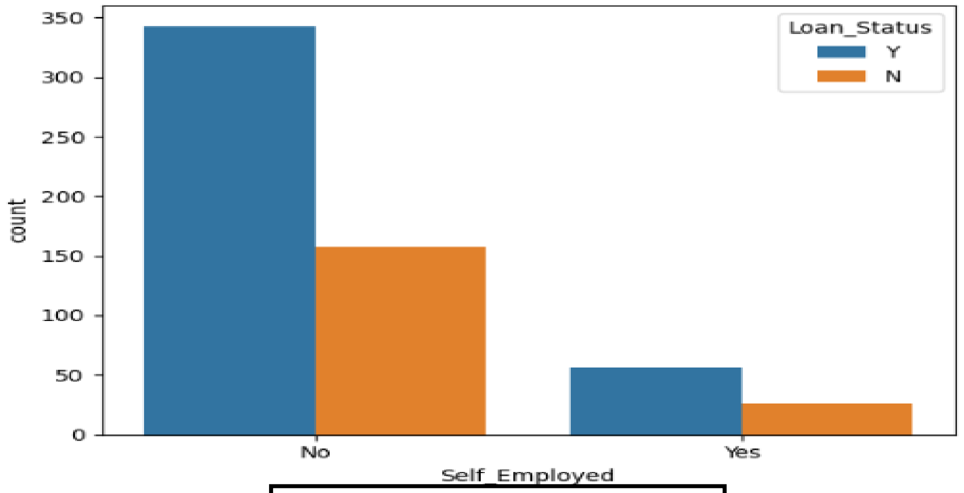
```
<Axes: xlabel='Education', ylabel='count'>
```



Here, x='Education' specifies the variable to be plotted on the x-axis.
Y= 'Loan_Status' specifies the variable to be plotted on the y-axis.
Hue='Loan_Status' specifies the variable to be used for grouping the data by color.
Data= The data parameter specifies the DataFrame to used for plotting.

```
sns.countplot(x='Self_Employed',hue='Loan_Status',data=data)
```

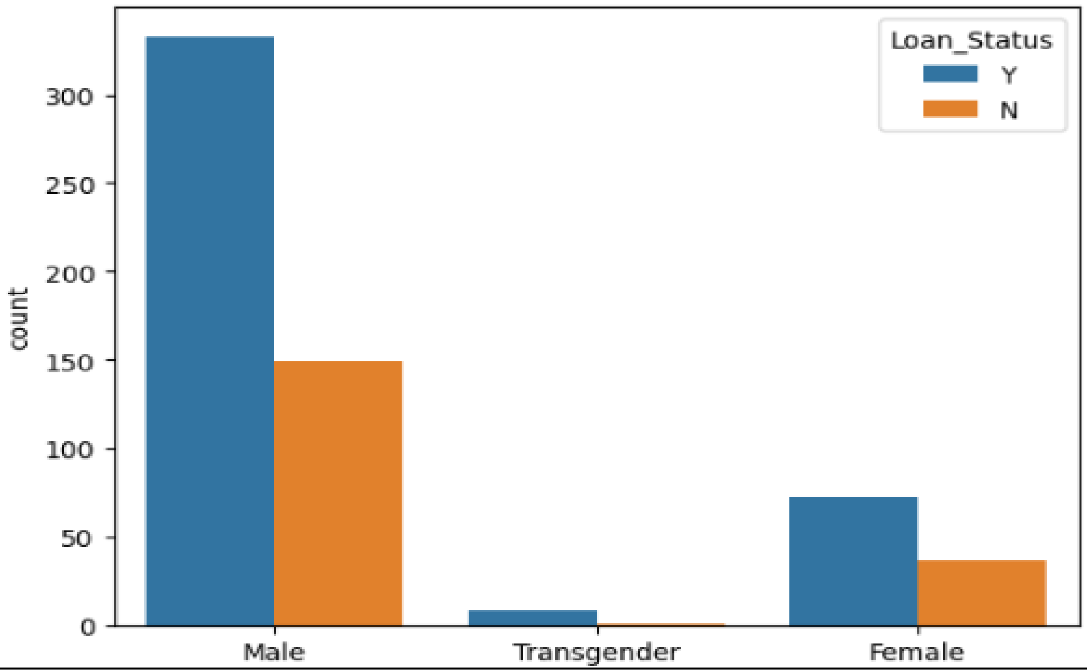
```
<Axes: xlabel='Self_Employed', ylabel='count'>
```



No-> Unemployed Yes-> Employed

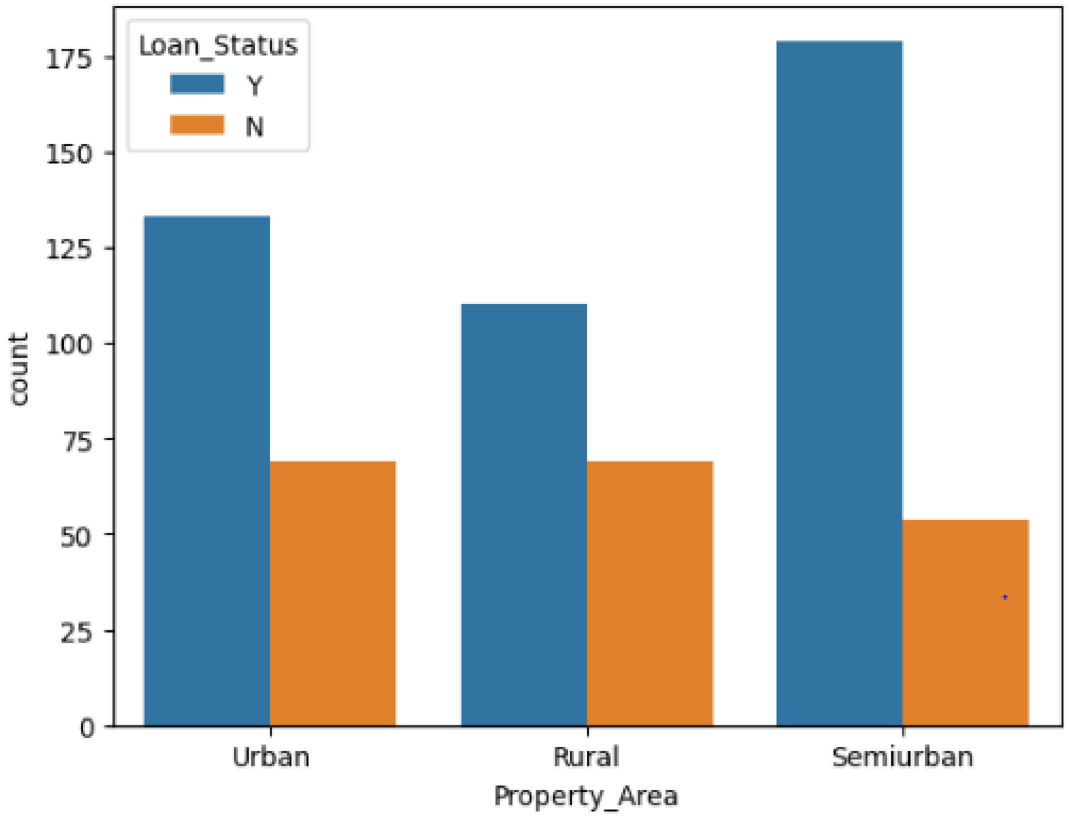
```
sns.countplot(x='Gender',hue='Loan_Status',data=data)
```

```
<Axes: xlabel='Gender', ylabel='count'>
```



```
sns.countplot(x='Property_Area',hue='Loan_Status',data=data)
```

```
<Axes: xlabel='Property_Area', ylabel='count'>
```



6.2 Training the Model

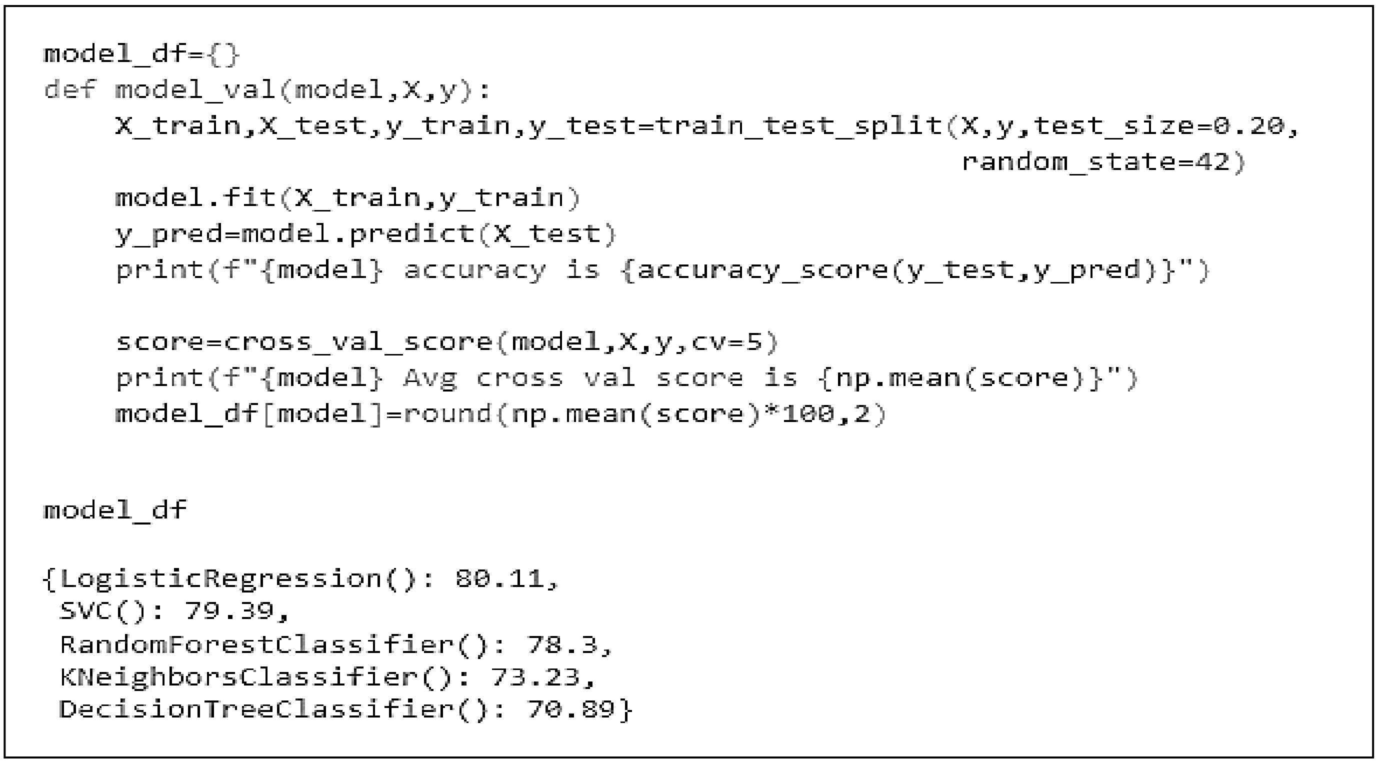
Splitting the dataset:

Splitting the dataset refers to dividing the available data into separate subsets for different purposes, such as training a model, validating its performance, and testing its generalization ability.

a) Training set: This subset is used to train the machine learning model. It contains the majority of the data and is used to optimize the model's parameters and learn the underlying patterns and relationships.

b) Validation set: A separate subset called the validation set is used to fine-tune the model's hyperparameters and evaluate its performance during the training process. validation set helps in selecting the best-performing model and avoiding overfitting.

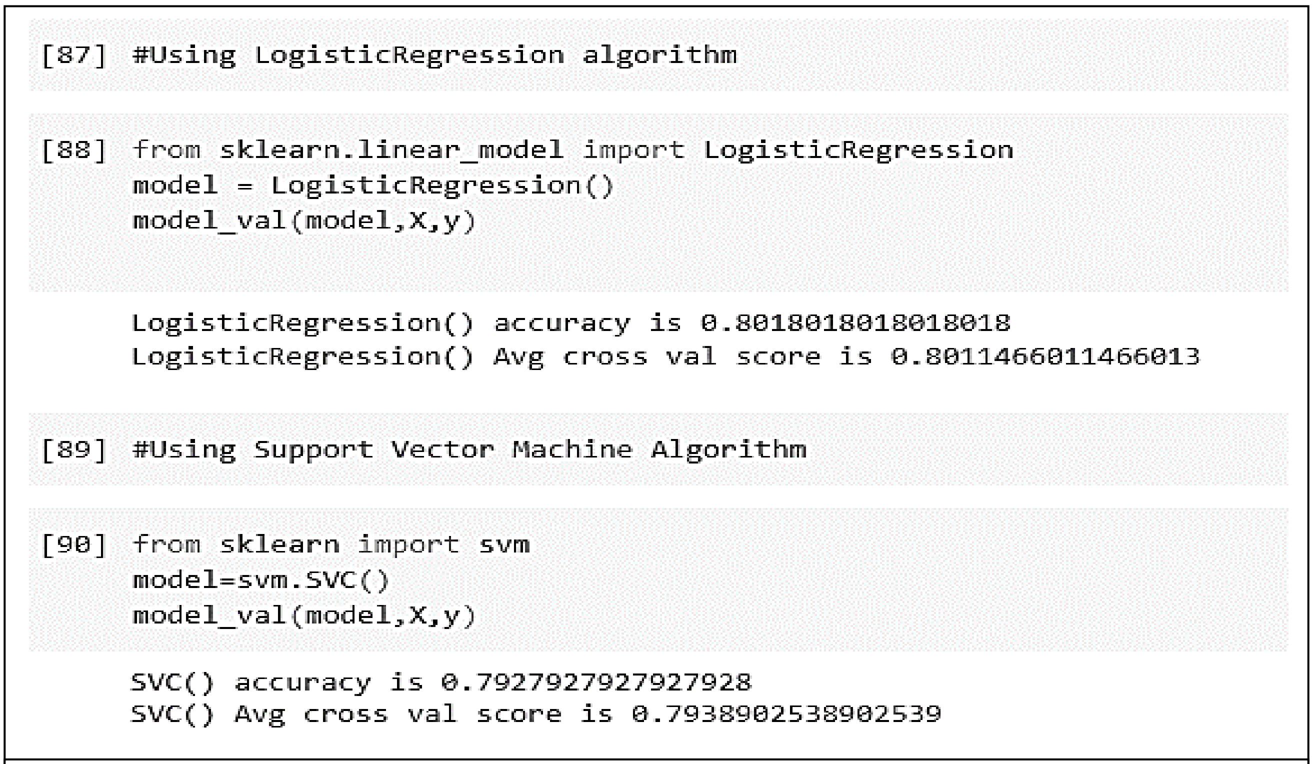
c) Test set: This subset is used to evaluate the final performance and generalization ability of the trained model. It is independent of the training process and provides an unbiased estimation of how well the model will perform on unseen data.



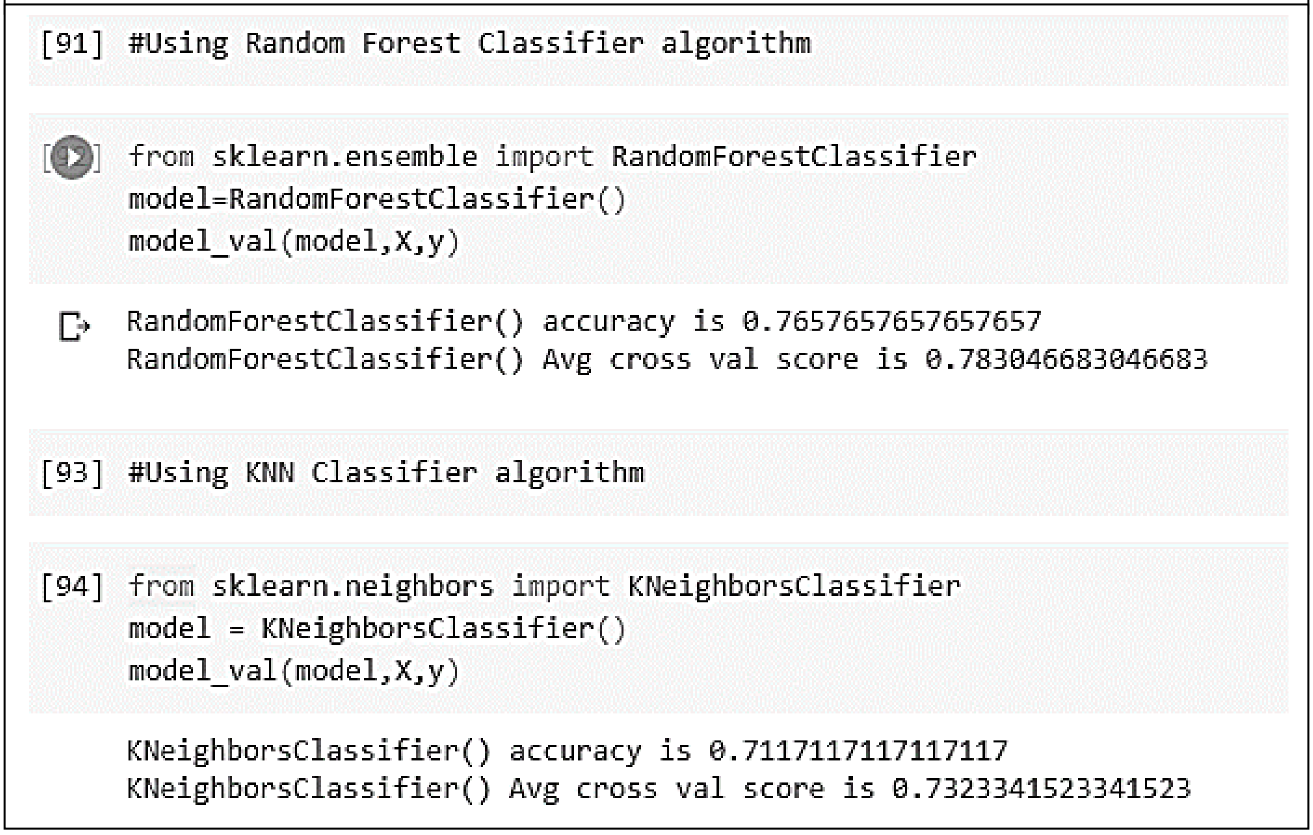
```
model_df={}
def model_val(model,X,y):
    X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.20,
                                                   random_state=42)
    model.fit(X_train,y_train)
    y_pred=model.predict(X_test)
    print(f"{model} accuracy is {accuracy_score(y_test,y_pred)}")

    score=cross_val_score(model,X,y,cv=5)
    print(f"{model} Avg cross val score is {np.mean(score)}")
    model_df[model]=round(np.mean(score)*100,2)


model_df

{LogisticRegression(): 80.11,
 SVC(): 79.39,
 RandomForestClassifier(): 78.3,
 KNeighborsClassifier(): 73.23,
 DecisionTreeClassifier(): 70.89}
```

As this is a classification problem so we will be using these models :

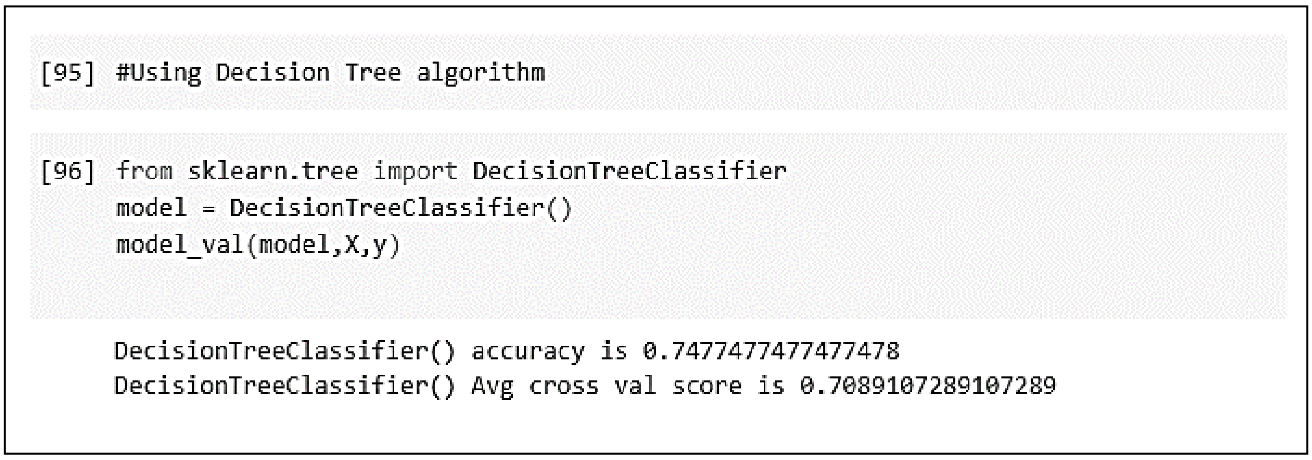
- ✓ Logistics Regression Algorithm
- ✓ Support Vector Machine Algorithm
- ✓ Random Forest Classifiers Algorithm
- ✓ KNN Classifier Algorithm
- ✓ Decision Tree Algorithm



```
[87] #Using LogisticRegression algorithm

[88] from sklearn.linear_model import LogisticRegression
     model = LogisticRegression()
     model_val(model,X,y)

     LogisticRegression() accuracy is 0.8018018018018018
     LogisticRegression() Avg cross val score is 0.8011466011466013

[89] #Using Support Vector Machine Algorithm

[90] from sklearn import svm
     model=svm.SVC()
     model_val(model,X,y)

     SVC() accuracy is 0.7927927927927928
     SVC() Avg cross val score is 0.7938902538902539
```



```
[91] #Using Random Forest Classifier algorithm

[ ]  from sklearn.ensemble import RandomForestClassifier
     model=RandomForestClassifier()
     model_val(model,X,y)

     RandomForestClassifier() accuracy is 0.7657657657657657
     RandomForestClassifier() Avg cross val score is 0.783046683046683

[93] #Using KNN Classifier algorithm

[94] from sklearn.neighbors import KNeighborsClassifier
     model = KNeighborsClassifier()
     model_val(model,X,y)

     KNeighborsClassifier() accuracy is 0.7117117117117117
     KNeighborsClassifier() Avg cross val score is 0.7323341523341523
```



```
[95] #Using Decision Tree algorithm

[96] from sklearn.tree import DecisionTreeClassifier
     model = DecisionTreeClassifier()
     model_val(model,X,y)

     DecisionTreeClassifier() accuracy is 0.7477477477477478
     DecisionTreeClassifier() Avg cross val score is 0.7089107289107289
```

- HyperParameter Tuning
  1) It refers to the process of selecting the optimal value for hyperparameters of machine learning model used for predicting loan status.
  2) The goal of find the combination of hyperparameter values that maximizes the predictive performance of the model in terms of accurately classifying whether a loan will be approved or not.

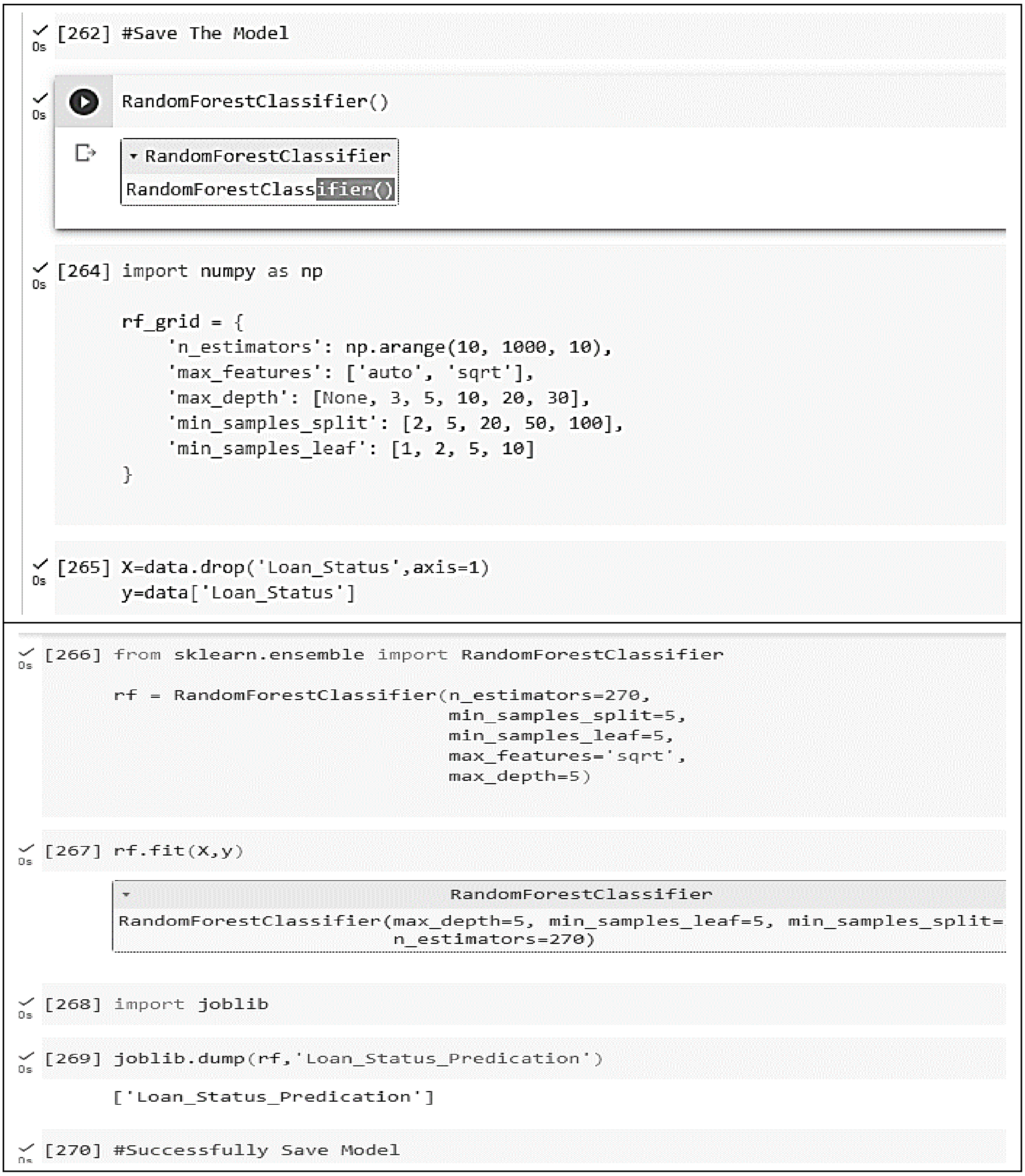
```
LogisticRegression score Before Hyperparameter Tuning: 80.11
LogisticRegression score after Hyperparameter Tuning: 80.48

SVC score Before Hyperparameter Tuning: 79.39
SVC score after Hyperparameter Tuning: 80.21

RandomForestClassifier score Before Hyperparameter Tuning: 78.3
RandomForestClassifier score after Hyperparameter Tuning: 80.66

KNeighborsClassifier score Before Hyperparameter Tuning: 73.23
KNeighborsClassifier score after Hyperparameter Tuning: 73.69

DecisionTreeClassifier score Before Hyperparameter Tuning: 70.89
DecisionTreeClassifier score after Hyperparameter Tuning: 75.66
```

Conclusion:

➔ Random Forest Classifier is giving the best accuracy with an accuracy score of 80.66% for the testing dataset.

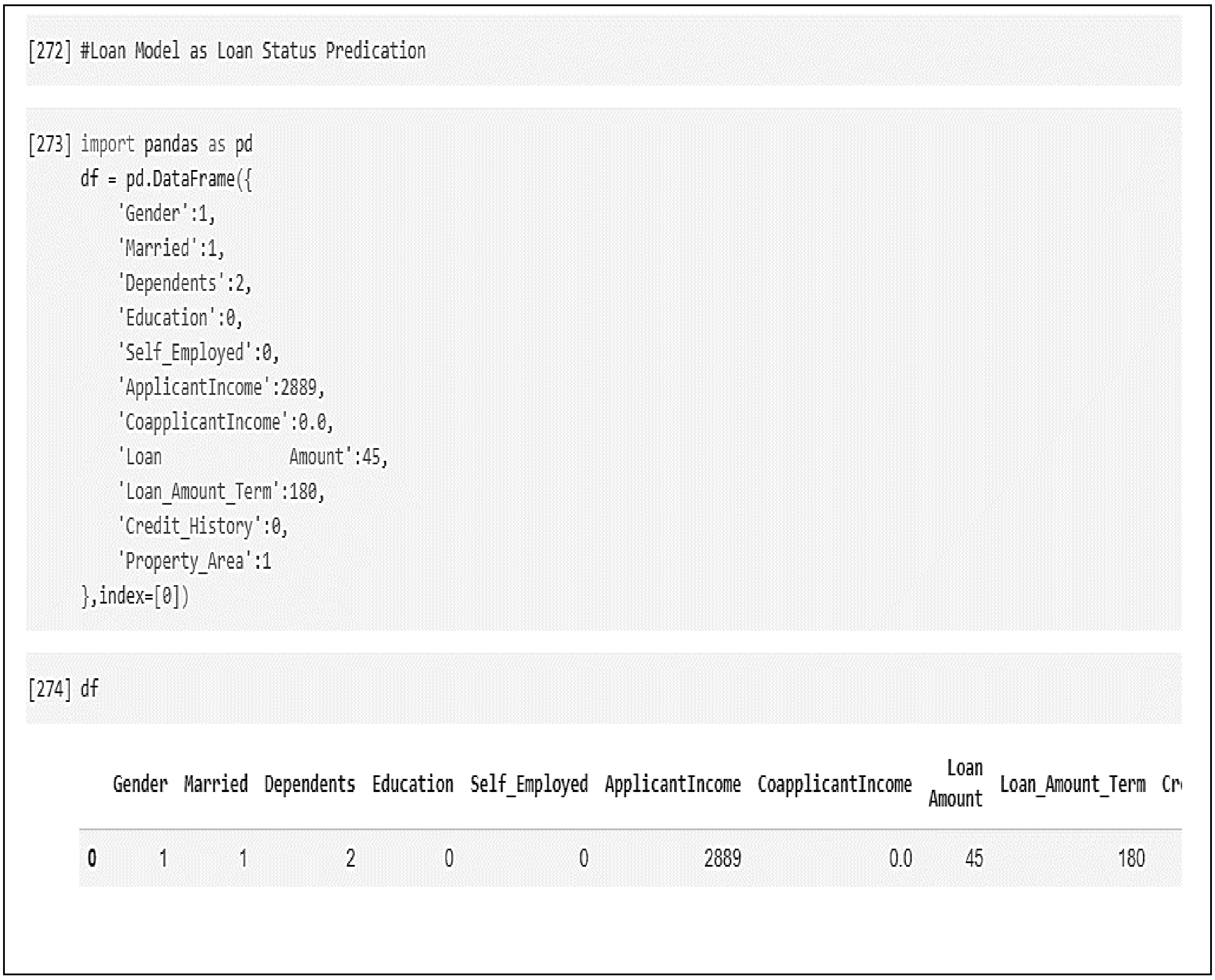
➔ It is used to best machine algorithm for Random Forest classifier.

6.3 Save the Model



```
[262] #Save The Model

RandomForestClassifier()

▾ RandomForestClassifier
RandomForestClassifier()

[264] import numpy as np

      rf_grid = {
          'n_estimators': np.arange(10, 1000, 10),
          'max_features': ['auto', 'sqrt'],
          'max_depth': [None, 3, 5, 10, 20, 30],
          'min_samples_split': [2, 5, 20, 50, 100],
          'min_samples_leaf': [1, 2, 5, 10]
      }

[265] X=data.drop('Loan_Status',axis=1)
      y=data['Loan_Status']

[266] from sklearn.ensemble import RandomForestClassifier

      rf = RandomForestClassifier(n_estimators=270,
                                  min_samples_split=5,
                                  min_samples_leaf=5,
                                  max_features='sqrt',
                                  max_depth=5)

[267] rf.fit(X,y)

      ▾                    RandomForestClassifier
      RandomForestClassifier(max_depth=5, min_samples_leaf=5, min_samples_split=
                             n_estimators=270)

[268] import joblib

[269] joblib.dump(rf,'Loan_Status_Predication')

      ['Loan_Status_Predication']

[270] #Successfully Save Model
```

6.4 Check the Model

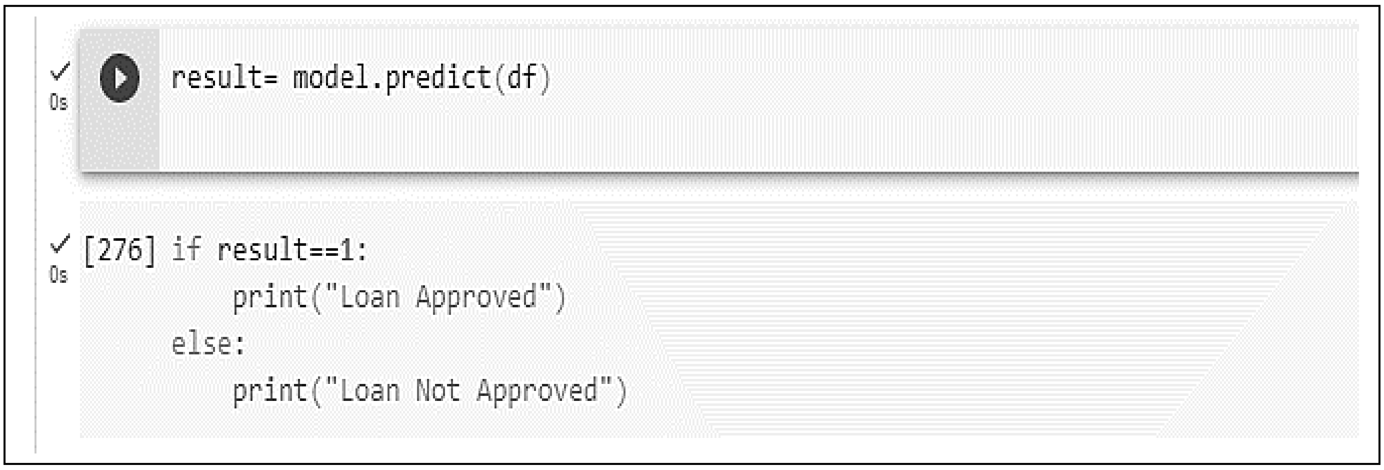


```
[272] #Loan Model as Loan Status Predication
```

```
[273] import pandas as pd
      df = pd.DataFrame({
          'Gender':1,
          'Married':1,
          'Dependents':2,
          'Education':0,
          'Self_Employed':0,
          'ApplicantIncome':2889,
          'CoapplicantIncome':0.0,
          'Loan              Amount':45,
          'Loan_Amount_Term':180,
          'Credit_History':0,
          'Property_Area':1
      },index=[0])
```

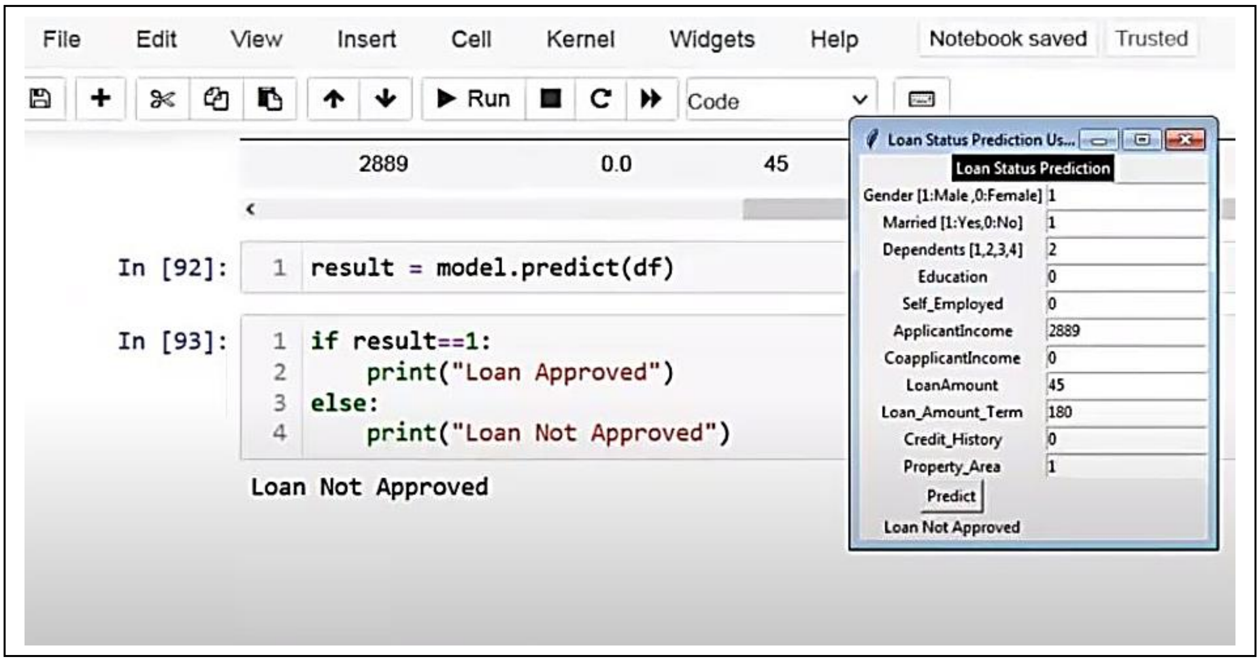
```
[274] df
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Loan Amount	Loan_Amount_Term	Cr
0	1	1	2	0	0	2889	0.0	45	180	

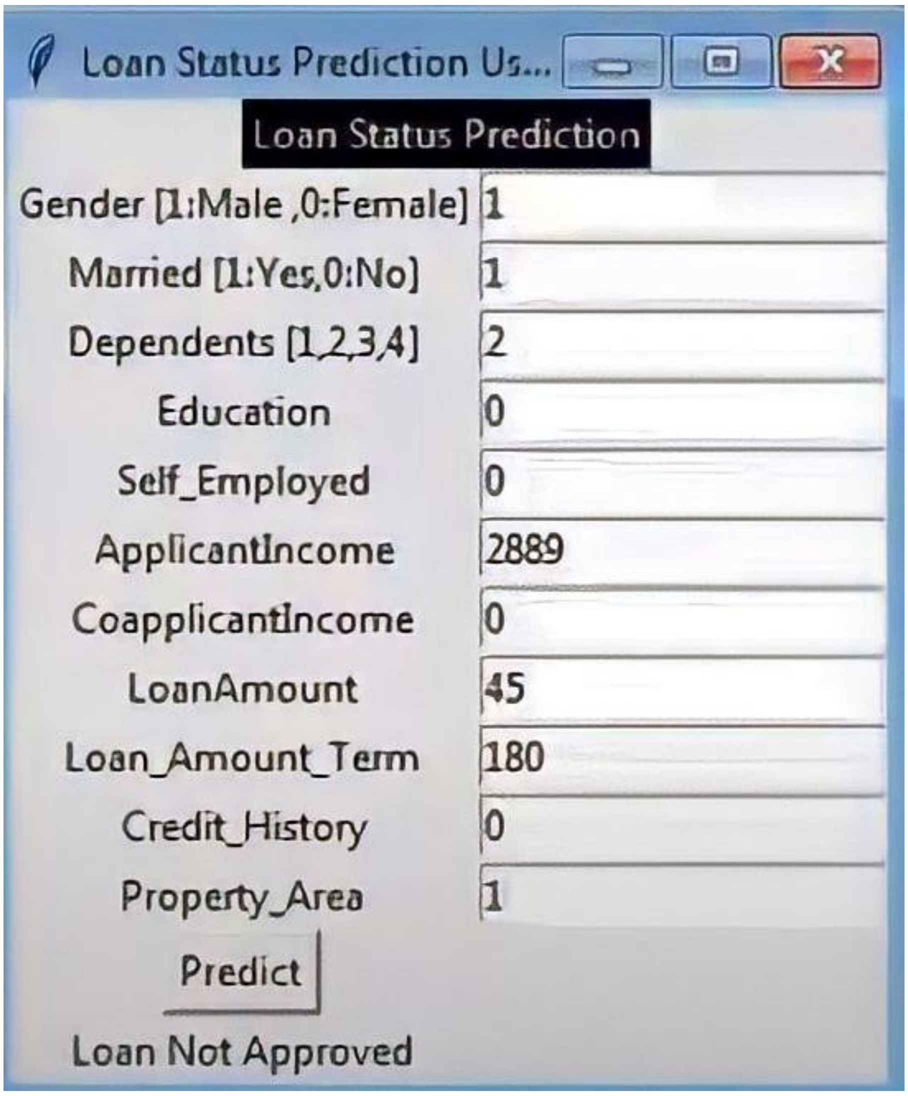


```
✓
0s    result= model.predict(df)
```

```
✓ [276] if result==1:
0s          print("Loan Approved")
        else:
            print("Loan Not Approved")
```



```
2889        0.0        45

In [92]:   result = model.predict(df)

In [93]:   if result==1:
               print("Loan Approved")
           else:
               print("Loan Not Approved")

           Loan Not Approved
```



Loan Status Prediction

Gender [1:Male ,0:Female]	1
Married [1:Yes,0:No]	1
Dependents [1,2,3,4]	2
Education	0
Self_Employed	0
ApplicantIncome	2889
CoapplicantIncome	0
LoanAmount	45
Loan_Amount_Term	180
Credit_History	0
Property_Area	1

Loan Not Approved

7 Conclusion

- ➔ The loan application system approves or rejects loan applications.
- ➔ Loan recovery is a significant parameter in the financial statements of loans, and predicting the likelihood of loan repayment by customers is challenging.
- ➔ Machine learning models are valuable for predicting outcomes with large datasets.
- ➔ In our project, We utilized five machine learning algorithms are used as Logistic regression, support vector machines, random forest classifiers, KNN classifiers, and decision trees to predict loan approval for customers.
- ➔ Based on experimental results, the random forest classifier algorithm exhibited the highest accuracy among the four algorithms.

8 References

https://www.simplilearn.com/tutorials/machine-learning-tutorial/supervised-and-unsupervised-learning

https://machinelearningmastery.com/machine-learning-with-python/

https://www.kaggle.com/datasets/ninzaami/loan-predication?resource=download

https://numpy.org/doc/stable/

https://seaborn.pydata.org/generated/seaborn.boxplot.html

https://colab.research.google.com/

https://jupyter.org/try-jupyter/lab/

THANK YOU