PREDICT BANKING DEFAULT/NON-DEFAULT USING LOAN DATA

A MINI PROJECT REPORT

18CSC305J - ARTIFICIAL INTELLIGENCE

Submitted by

Ninaad Arora

[RA2111032010014]

Bhuvana Reddy

[RA2111032010027]

Adeline

[RA2111032010031]

Under the guidance of
Dr.C.Fancy

Assistant Professor, Department of Computer Science and Engineering

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled **"PREDICT BANKING DEFAULT/NON-DEFAULT USING LOAN DATA"** is the bona fide work of **Ninaad Arora [RA2111032010014] ,Bhuvana Reddy[RA2111032010027] and Adeline[RA2111032010031]** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

DrC.Fancy
Assistant Professor
Department of Networking And Communications

Table of Contents:-

1)CONTENT:-

Here first of all we have a data set which is already given to us and it has beenattached with the report as well.The content has the use of predicting the faults during the completion of theloan.This is a classification problem and to solve it we have implemented two mainmethods which are:

- Logistic regression
- Decision tree

We here are using some advanced data science and Machine learning librarieswhich include:

- Pandas as pd
- Numpy as np
- Matplotlib.pyplot as plt
- Seaborn as sns
- Graphviz
- sklearn *

2.)INTRODUCTION:-

In today's world obtaining loans from financial institutions has become very common. Every day many people apply for loans, for a variety of purposes. But not all the applicants are reliable, and not everyone can be approved.

Based on the given dataset we predict banking default/non default duringloans.

The idea of this project is to gather loan data from the Lending Club website and use machine learning techniques on this data to extract important information and predict if a customer would be able to repaythe loan or not. In other words,the goal is to predict if the customer would be adefaulter or not. To solve such a classification problem we have logistic Regression and decision tree methodology.

Exploring the data set we have:

- current loan amount
- credit score
- annual income
- years in current job
- monthly debt
- years in credit history
- Months since last delinquent
- Number of open accounts
- Number of credit problems
- Current credit balance

- Maximum Open Credit
- Bankruptcies
- Tax Liens
- Defaulter
- Term_Long Term
- Term_Short Term
- Home Ownership_Home Mortgage
- Home Ownership_Own Home
- Home Ownership_Rent
- Purpose_Business Loan
- Purpose_Buy House
- Purpose_Buy a Car
- Purpose_Debt Consolidation
- Purpose_Educational Expenses
- Purpose_Home improvements
- Purpose_Medical Bills
- Purpose_Other
- Purpose_moving
- Purpose_renewable_energy
- Purpose_vacation
- Purpose_wedding

3.) PROBLEM STATEMENT:-

The dataset contains information about credit applicants. Banks, globally, use this kind of dataset and type of informative data to create models to help in deciding on who to accept/refuse for a loan.

After all the exploratory data analysis, cleansing and dealing with all the anomalies we might (will) find along the way, the patterns of a good/bad applicant will be exposed to be learned by machine learning models.

We're dealing with a supervised binary classification problem. The goal is to train the best machine learning model to maximize the predictive capability of deeply understanding the past customer's profile and minimizing the risk of future loan defaults.

4.) VARIABLE DEFINITION:-

So firstly we load the csv file in a pandas datafile, Which gets referred to as ‘df’ The csv file has 123 rows and 31 columns.

```
df = pd.read_csv("2_Banking_Default _Non_Default.csv")
```

```
[ ] df.shape
    # gives (row,column)

    (123, 31)
```

31 columns contain different types of 31 variables which are described using the df.info() method.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 123 entries, 0 to 122
Data columns (total 31 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Current Loan Amount             123 non-null    int64
 1   Credit Score                    123 non-null    int64
 2   Annual Income                   123 non-null    int64
 3   Years in current job            123 non-null    int64
 4   Monthly Debt                    123 non-null    float64
 5   Years of Credit History         123 non-null    float64
 6   Months since last delinquent    123 non-null    int64
 7   Number of Open Accounts         123 non-null    int64
 8   Number of Credit Problems       123 non-null    int64
 9   Current Credit Balance          123 non-null    int64
 10  Maximum Open Credit             123 non-null    int64
 11  Bankruptcies                    123 non-null    int64
 12  Tax Liens                       123 non-null    int64
 13  Defaulter                       123 non-null    int64
 14  Term_Long Term                  123 non-null    int64
 15  Term_Short Term                 123 non-null    int64
 16  Home Ownership_Home Mortgage    123 non-null    int64
 17  Home Ownership_Own Home         123 non-null    int64
 18  Home Ownership_Rent             123 non-null    int64
 19  Purpose_Business Loan           123 non-null    int64
 20  Purpose_Buy House               123 non-null    int64
 21  Purpose_Buy a Car               123 non-null    int64
 22  Purpose_Debt Consolidation      123 non-null    int64
 23  Purpose_Educational Expenses    123 non-null    int64
 24  Purpose_Home Improvements       123 non-null    int64
 25  Purpose_Medical Bills           123 non-null    int64
 26  Purpose_Other                   123 non-null    int64
 27  Purpose_moving                  123 non-null    int64
 28  Purpose_renewable_energy        123 non-null    int64
 29  Purpose_vacation                123 non-null    int64
 30  Purpose_wedding                 123 non-null    int64
dtypes: float64(2), int64(29)
```

5.) VARIABLE STRUCTURE:-

Different types of variables are described using the .describe() function which calculates mean, standard deviation. There are total 31 variables each having 123 count values.

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Current Loan Amount	123.0	2.974752e+05	171664.298553	21956.00	178013.00	265936.00	397540.00	772706.00
Credit Score	123.0	4.734472e+02	79.534837	350.00	410.00	466.00	515.50	708.00
Annual Income	123.0	1.315072e+06	540617.758376	188233.00	895603.00	1255691.00	1753424.50	2860260.00
Years in current job	123.0	5.723577e+00	3.704894	1.00	2.00	5.00	10.00	10.00
Monthly Debt	123.0	1.745325e+04	9551.579057	925.87	10876.17	16637.35	21928.28	45790.95
Years of Credit History	123.0	1.789106e+01	5.972231	6.90	14.00	16.50	20.30	34.00
Months since last delinquent	123.0	3.449593e+01	22.488385	1.00	14.00	33.00	53.00	81.00
Number of Open Accounts	123.0	1.135772e+01	4.725463	3.00	8.00	10.00	15.00	25.00
Number of Credit Problems	123.0	1.056911e-01	0.308699	0.00	0.00	0.00	0.00	1.00
Current Credit Balance	123.0	2.085449e+05	146767.279498	0.00	97318.00	183673.00	284572.50	641896.00
Maximum Open Credit	123.0	4.830658e+05	350074.211691	0.00	237402.00	409992.00	652630.00	1822128.00
Bankruptcies	123.0	7.317073e-02	0.261482	0.00	0.00	0.00	0.00	1.00
Tax Liens	123.0	0.000000e+00	0.000000	0.00	0.00	0.00	0.00	0.00
Defaulter	123.0	1.869919e-01	0.391500	0.00	0.00	0.00	0.00	1.00
Term_Long Term	123.0	2.682927e-01	0.444883	0.00	0.00	0.00	1.00	1.00
Term_Short Term	123.0	7.317073e-01	0.444883	0.00	0.00	1.00	1.00	1.00
Home Ownership_Home Mortgage	123.0	5.772358e-01	0.496019	0.00	0.00	1.00	1.00	1.00
Home Ownership_Own Home	123.0	4.878049e-02	0.216290	0.00	0.00	0.00	0.00	1.00
Home Ownership_Rent	123.0	3.739837e-01	0.485838	0.00	0.00	0.00	1.00	1.00
Purpose_Business Loan	123.0	2.439024e-02	0.154888	0.00	0.00	0.00	0.00	1.00
Purpose_Buy House	123.0	8.130081e-03	0.090167	0.00	0.00	0.00	0.00	1.00
Purpose_Buy a Car	123.0	8.130081e-03	0.090167	0.00	0.00	0.00	0.00	1.00
Purpose_Debt Consolidation	123.0	7.317073e-01	0.444883	0.00	0.00	1.00	1.00	1.00
Purpose_Educational Expenses	123.0	0.000000e+00	0.000000	0.00	0.00	0.00	0.00	0.00
Purpose_Home Improvements	123.0	5.691057e-02	0.232619	0.00	0.00	0.00	0.00	1.00
Purpose_Medical Bills	123.0	2.439024e-02	0.154888	0.00	0.00	0.00	0.00	1.00
Purpose_Other	123.0	1.300813e-01	0.337769	0.00	0.00	0.00	0.00	1.00
Purpose_moving	123.0	0.000000e+00	0.000000	0.00	0.00	0.00	0.00	0.00
Purpose_renewable_energy	123.0	0.000000e+00	0.000000	0.00	0.00	0.00	0.00	0.00
Purpose_vacation	123.0	8.130081e-03	0.090167	0.00	0.00	0.00	0.00	1.00
Purpose_wedding	123.0	8.130081e-03	0.090167	0.00	0.00	0.00	0.00	1.00

6.) EXPLORATORY DATA ANALYSIS (EDA):-

Exploratory Data Analysis (EDA) is a crucial step in any data analysis project as it enables data scientists to understand the data, identify patterns,relationships, and anomalies, and prepare the data for modelling.

```
df.head().T
```

	0	1	2	3	4
Current Loan Amount	452166.00	304634.00	396198.00	764302.0	398882.00
Credit Score	708.00	708.00	708.00	708.0	594.00
Annual Income	1952516.00	1597406.00	873639.00	1471037.0	1148322.00
Years in current job	1.00	2.00	3.00	10.0	2.00
Monthly Debt	34656.95	38470.63	12667.68	28807.8	19808.64
Years of Credit History	15.70	17.80	12.80	16.2	21.90
Months since last delinquent	1.00	2.00	2.00	4.0	5.00
Number of Open Accounts	14.00	16.00	13.00	8.0	12.00
Number of Credit Problems	0.00	0.00	0.00	0.0	0.00
Current Credit Balance	428925.00	329308.00	263891.00	386745.0	481137.00
Maximum Open Credit	546964.00	914408.00	411268.00	1408220.0	677732.00
Bankruptcies	0.00	0.00	0.00	0.0	0.00
Tax Liens	0.00	0.00	0.00	0.0	0.00
Defaulter	1.00	1.00	1.00	1.0	1.00
Term_Long Term	1.00	0.00	1.00	1.0	1.00
Term_Short Term	0.00	1.00	0.00	0.0	0.00
Home Ownership_Home Mortgage	1.00	1.00	0.00	1.0	1.00
Home Ownership_Own Home	0.00	0.00	0.00	0.0	0.00
Home Ownership_Rent	0.00	0.00	1.00	0.0	0.00
Purpose_Business Loan	0.00	0.00	0.00	0.0	0.00
Purpose_Buy House	0.00	0.00	0.00	0.0	0.00
Purpose_Buy a Car	0.00	0.00	0.00	0.0	0.00
Purpose_Debt Consolidation	1.00	1.00	0.00	0.0	1.00
Purpose_Educational Expenses	0.00	0.00	0.00	0.0	0.00
Purpose_Home Improvements	0.00	0.00	0.00	0.0	0.00
Purpose_Medical Bills	0.00	0.00	0.00	0.0	0.00
Purpose_Other	0.00	0.00	1.00	1.0	0.00
Purpose_moving	0.00	0.00	0.00	0.0	0.00
Purpose_renewable_energy	0.00	0.00	0.00	0.0	0.00
Purpose_vacation	0.00	0.00	0.00	0.0	0.00
Purpose_wedding	0.00	0.00	0.00	0.0	0.00

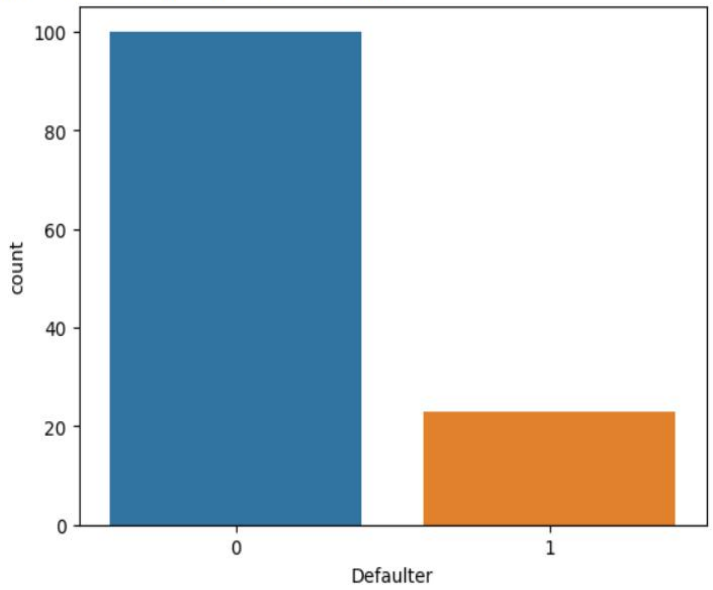
Analysing Target Variable

Variable Analysis and Plotting according to their data.

Defaulter: It is the most important variable as in the banking default file wehave to search for the person who has done the default. As it contains only two types of data 0 or 1 we have used a count plot.

```
sns.countplot(x="Defaulter",data=df)
```

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



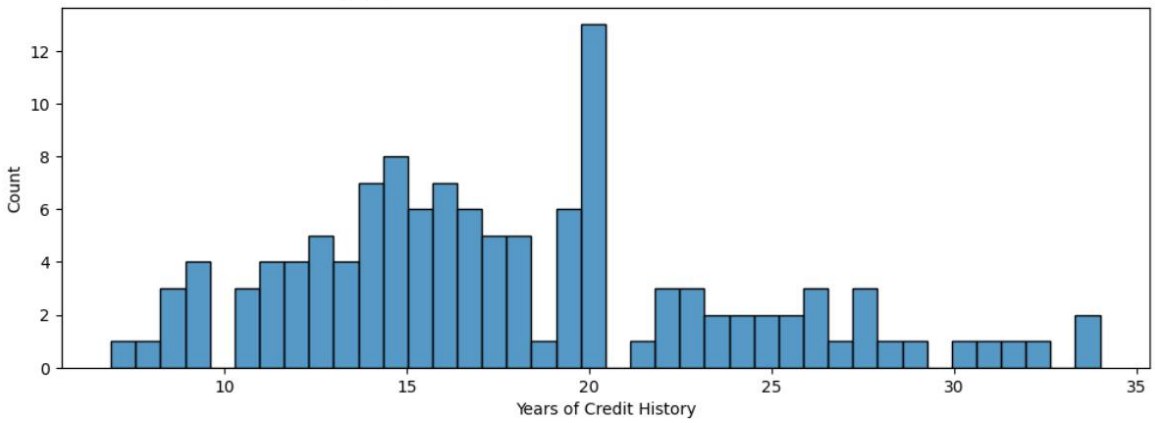
considering 0 for no and 1 for yes , i.e. 1 represent defaulter and 0 means non-defaulter

```
[ ] df['Defaulter'].value_counts()
    # shows the frequency of unique values in column
```

```
0    100
1     23
Name: Defaulter, dtype: int64
```

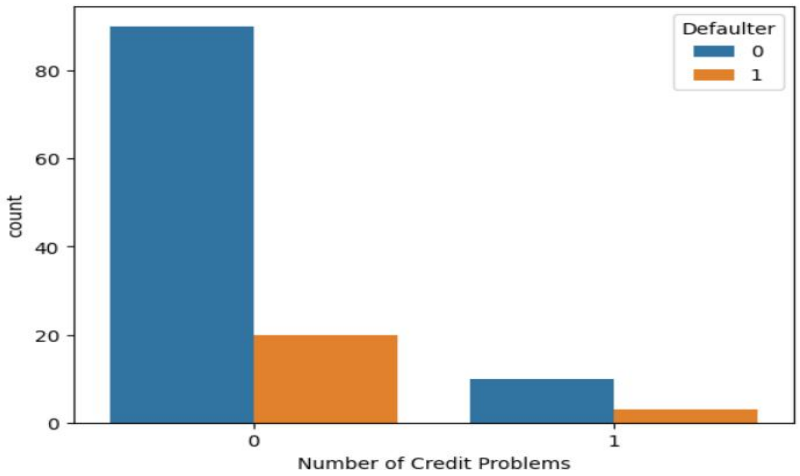
Years of Credit History: It basically tells about the tenure of the credit amount provided to an individual. It contains a lot of different values so Histogram plot is used.

```
[ ] df['Years of Credit History'].nunique()

    82

    plt.figure(figsize=(12,4))
    sns.histplot(df['Years of Credit History'],bins=40)

    <Axes: xlabel='Years of Credit History', ylabel='Count'>
```



Number of Credit Problems: It shows an individual who has a credit problem. It contains two different types of values 0 or 1. Count plot is used to describe the variable along with it shows the person who has defaulted if it has a credit problem or not.

```
[ ] sns.countplot(x="Number of Credit Problems",hue='Defaulter',data=df)

    <Axes: xlabel='Number of Credit Problems', ylabel='count'>
```



Now variables Term,Home Ownership and Purpose have been into dummy variables as :

Term (Short Term, Long Term)

Home Ownership(Home Mortgage,Own Home,Own Home ,Rent ,Loan)

Purpose (Business Loan,Buy House,Buy a Car Debt Consolidation,Educational Expenses,Home Improvements,Medical,Bills,Other ,

moving,renewable_energy,vacation,wedding)

```
df['Term_Long Term'].value_counts()
```

```
0    90
1    33
Name: Term_Long Term, dtype: int64
```

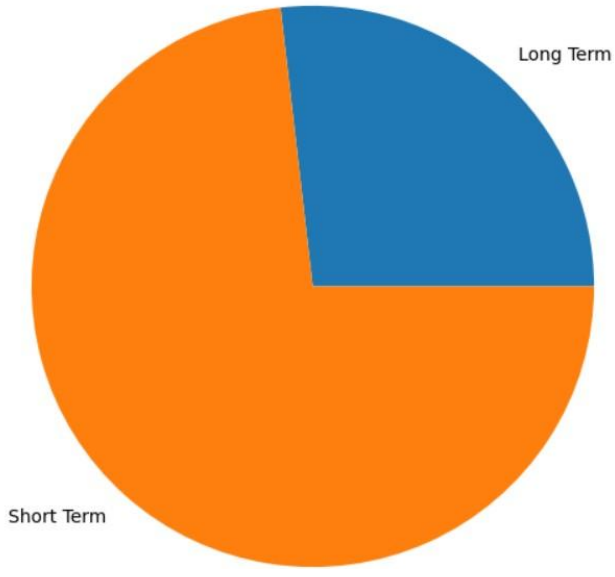
Considering 1 represents yes and 0 represents No, Thus 33 people have taken long_term_loan and rest 90 have taken short_term_loan

```
[ ]  df['Term_Short Term'].value_counts()
```

```
1    90
0    33
Name: Term_Short Term, dtype: int64
```

Long_term And Short_term: They are categorical values separated to provide clean data to predict. To represent them a pie chart is used.

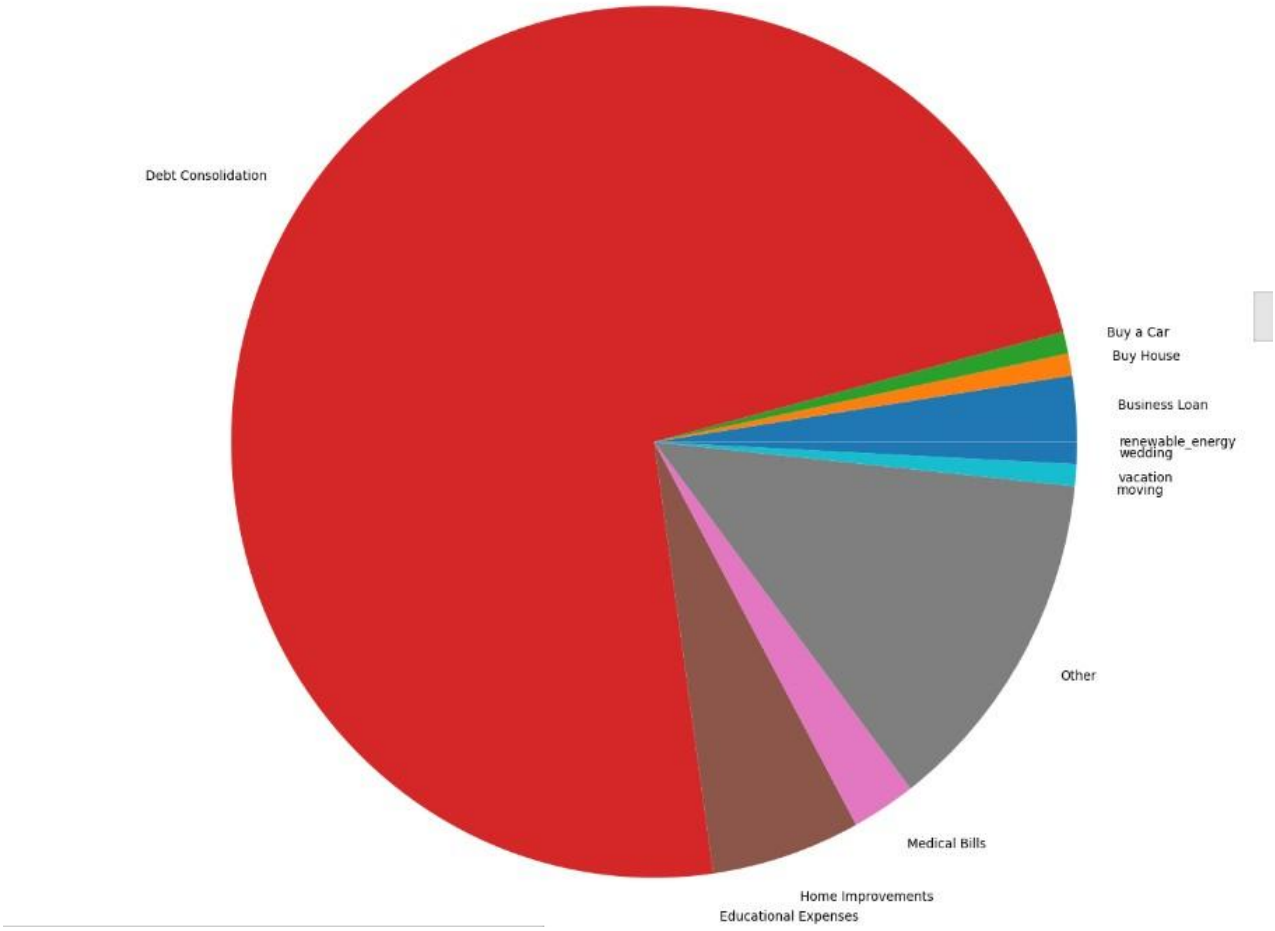
```
labels = ['Long Term', 'Short Term']
data = [33,90]
# exhaustive set thus not , also clearly can be seen by -1 correlation coefficient of these 2

fig = plt.figure(figsize =(10, 7))
plt.pie(data, labels = labels)
plt.show()
```



Purposes: The bank provides a loan to an individual for different types ofpurposes. Business Loan, Home Loan, Car Loan, etc.

To represent different purposes Pie Chart is used as it will provide a better picture.

```
labels = ['Business Loan','Buy House','Buy a Car','Debt Consolidation','Educational
data = [3,1,1,90,0,7,3,16,0,1,1,0]
fig = plt.figure(figsize =(15,15))
plt.pie(data,labels = labels)
plt.show()
```



6.)MISSING VALUE TREATMENT:-

Missing values in a dataset can impact the accuracy and effectiveness of a predictive model. The missing value treatment process involved identifying missing values, determining the type of missingness, and selecting an appropriate method for imputingthe missing values.

Overall, the missing value treatment process helped to ensure that the Default Prediction model was accurate and effective. By identifying and imputing missing values, the dataset was complete, and the model was based on accurate and complete data. This improved the accuracy of the model, making it more effective in predicting the likelihoodof default accurately.

There is no missing value in the dataset.

```
[ ]  df.isnull()
```

	Current Loan Amount	Credit Score	Annual Income	Years in current job	Monthly Debt	Years of Credit History	Months since last delinquent	Number of Open Accounts	Number of Credit Problems	Current Credit Balance	...
0	False	False	False	False	False	False	False	False	False	False	..
1	False	False	False	False	False	False	False	False	False	False	..
2	False	False	False	False	False	False	False	False	False	False	..
3	False	False	False	False	False	False	False	False	False	False	..
4	False	False	False	False	False	False	False	False	False	False	..
...	...	...	...	...	...	...	...	...	...	...	..
118	False	False	False	False	False	False	False	False	False	False	..
119	False	False	False	False	False	False	False	False	False	False	..
120	False	False	False	False	False	False	False	False	False	False	..
121	False	False	False	False	False	False	False	False	False	False	..
122	False	False	False	False	False	False	False	False	False	False	..

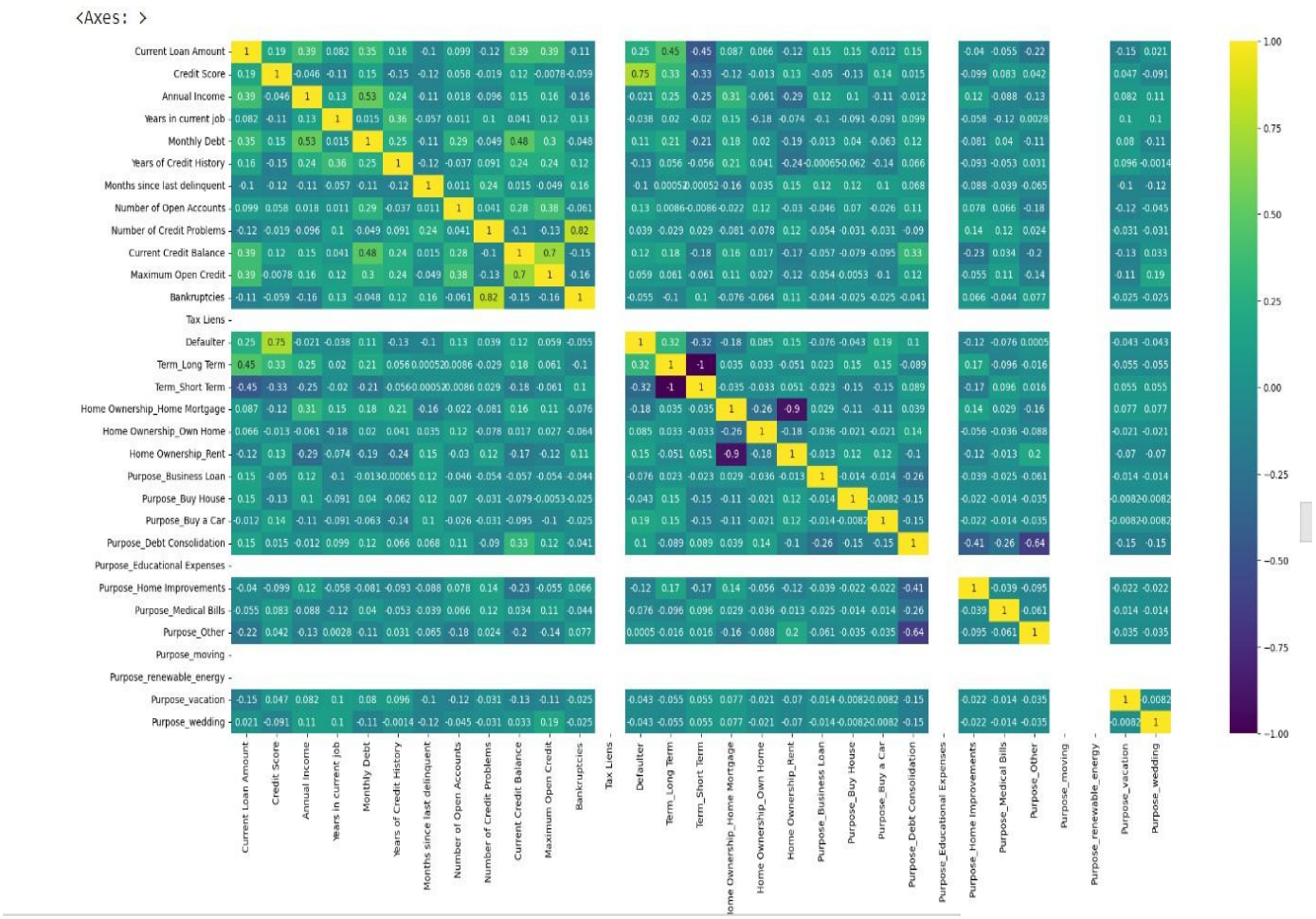
123 rows × 31 columns

7.) SIGNIFICANCE VARIABLE SELECTION:-

It is an essential step in building a predictive model as it involves identifying the mostsignificant variables that have a strong relationship with the target variable.

Correlation Analysis:df.corr() method is used for correlation and heat map is used for visualization.

```
[ ]  plt.figure(figsize=(25,12))
     sns.heatmap(df.corr(),annot=True,cmap="viridis")
```



Drawing conclusions from the graph , value > 0.5 is considered strong correlationDefaulter is highly positively correlated with credit score (0.75)

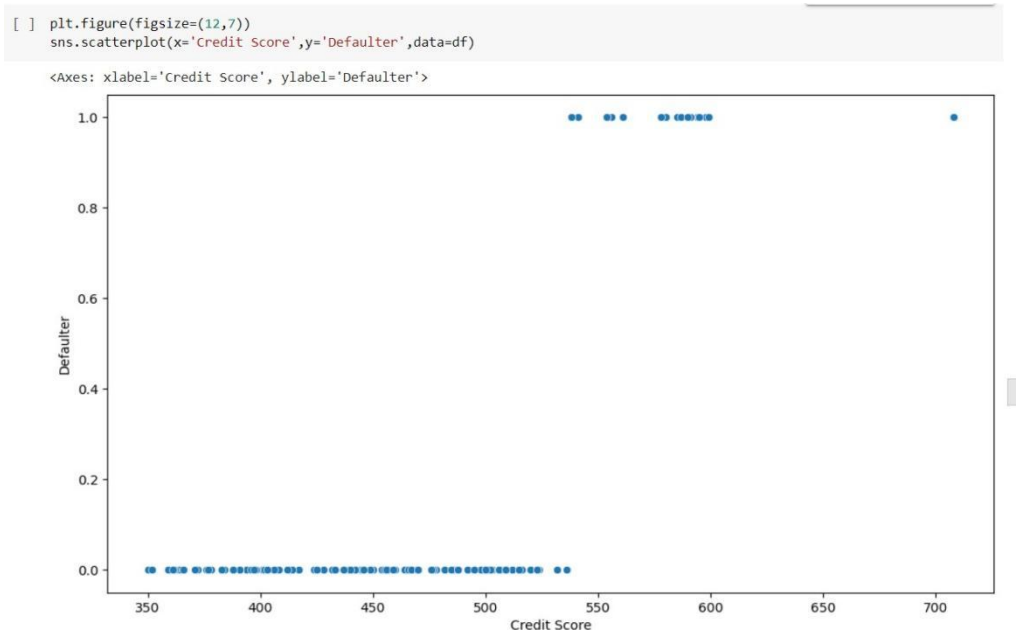
Annual income is positively correlated with credit score (0.53)

Current Credit Balance is positively correlated with credit score (0.48) Bankruptcies is highly positively correlated with Number of credit problem(0.82)Term_Short_Term and Term_Long_Term are absolutely negatively correlated (-1)

Home_ownership_rent is highly negatively correlated with home_onwneship_home_mortgage(-0.90)

Purpose_other is negatively correlated to Purpose_Debt_Consolidation (-0.64)

Scatterplot is the best method to show the relationship between two differentvariables. Scatterplot is used to display the relation between credit score and defaulter.



Separating of dependent and independent variables is done for the implementation of different models and prediction of defaulter andnon-defaulter in the data set.

Separating dependent and independent variables.

```
[ ] X=df.drop('Defaulter',axis=1)
    # axis = 1 -- column, 0 -- row
```

```
Y=df['Defaulter']
```

In Machine Learning algorithms using linear regression, logistic regression, KNN, neural network one has to apply feature scaling as they are distance based.

Random Forest, Decision Tree, Naive Bayes, ADAboost does not require scaling as they are notdistance based.

```
[ ] col1 = ['Current Loan Amount','Months since last delinquent','Credit Score','Annual Income','Maximum Open Credit',
            'Monthly Debt','Current Credit Balance','Years of Credit History','Number of Open Accounts']
    # select columns for feature scaling
```

```
[ ] from sklearn.preprocessing import StandardScaler
    st = StandardScaler()
    X[col1] = st.fit_transform(X[col1])
```

```
X.head().T
```

	0	1	2	3	4
Current Loan Amount	0.904809	0.041873	0.577444	2.730539	0.593143
Credit Score	2.961120	2.961120	2.961120	2.961120	1.521923
Annual Income	1.183925	0.524379	-0.819875	0.289673	-0.309706
Years in current job	1.000000	2.000000	3.000000	10.000000	2.000000
Monthly Debt	1.808504	2.209409	-0.503073	1.193624	0.247606
Years of Credit History	-0.368375	-0.015309	-0.855941	-0.284311	0.674009
Months since last delinquent	-1.495569	-1.450920	-1.450920	-1.361621	-1.316972
Number of Open Accounts	0.561444	0.986414	0.348959	-0.713466	0.136474
Number of Credit Problems	0.000000	0.000000	0.000000	0.000000	0.000000
Current Credit Balance	1.507703	0.826185	0.378643	1.219134	1.864905
Maximum Open Credit	0.183274	1.237184	-0.205932	2.653546	0.558346
Bankruptcies	0.000000	0.000000	0.000000	0.000000	0.000000
Tax Liens	0.000000	0.000000	0.000000	0.000000	0.000000
Term_Long Term	1.000000	0.000000	1.000000	1.000000	1.000000
Term_Short Term	0.000000	1.000000	0.000000	0.000000	0.000000
Home Ownership_Home Mortgage	1.000000	1.000000	0.000000	1.000000	1.000000

8.)MODEL PERFORMANCE:-

```
[ ] from sklearn.model_selection import train_test_split
    from sklearn.model_selection import cross_val_score
    from sklearn.metrics import accuracy_score
```

```
[ ] 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)}")

    # using cross validation in order to overcome the issue of selecting best areas for split
      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)
```

A user-defined function (model_val) is created to test for the accuracy of the different models for the given data of bank defaulter and non-defaulter provided. It encompasses the splitting of the data and prediction of the test_values, followed by checking accuracy and average cross validation score. To test for different models, one has tojust pass the model name object to the function.

Accuracy is the ratio of correctly predicted instances to the totalnumber of instances in the dataset, expressed as a percentage.

Cross Validation Score is a measure of a model's performance obtained through cross-validation.It is a technique used in machine learning to evaluate the performance of a model on multiple subsets of the data.

Here we have used k-fold cross-validation: The data is divided into k equal-sized folds, and the model is trained and evaluated k times, with each fold serving as the validation set once.

4 different models are used - **Logistic regression**:

Decision Tree:

Logistic Regression as binary classification problem

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model_val(model,X,Y)
```

```
LogisticRegression() accuracy is 0.96
LogisticRegression() avg cross val score is 0.9353333333333333
```

Decision Tree

```
[ ] from sklearn.tree import DecisionTreeClassifier
    model = DecisionTreeClassifier()
    model_val(model,X,Y)
```

```
DecisionTreeClassifier() accuracy is 1.0
DecisionTreeClassifier() avg cross val score is 1.0
```

Random Forest:-

Random Forest

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

```
RandomForestClassifier() accuracy is 0.96
RandomForestClassifier() avg cross val score is 0.9676666666666666
```

Gradient Boosting Classifier:-

Gradient Boosting classifier

```
from sklearn.ensemble import GradientBoostingClassifier
model = GradientBoostingClassifier()
model_val(model,X,Y)
```

```
GradientBoostingClassifier() accuracy is 1.0
GradientBoostingClassifier() avg cross val score is 1.0
```

10.) BEST MODEL ITERATIONS:-

So in total we have tried 4 different model whose accuracy scores we can follow as :

- Logistic regression:

accuracy: 0.96

average cross value score: 0.935

- Decision Tree

accuracy: 1.0

average cross value score: 1.0

- Random Forest

accuracy: 0.96

average cross value score: 0.97

- Gradient Boosting Classifier

accuracy: 1.0

average cross value score: 1.0

```
[ ]  model_df
```

```
{LogisticRegression(): 93.53,
 DecisionTreeClassifier(): 100.0,
 RandomForestClassifier(): 96.77,
 GradientBoostingClassifier(): 100.0}
```

Decision Tree is considered as the best model for the prediction of banking default and non-default.

Considering Decision Tree as best model

```
[ ] X=df.drop('Defaulter',axis=1)
    Y=df['Defaulter']
```

```
[ ] dt = DecisionTreeClassifier(min_samples_split=5,min_samples_leaf=5,max_features='sqrt',max_depth=5)
```

```
dt.fit(X,Y)
```

```
DecisionTreeClassifier(max_depth=5, max_features='sqrt', min_samples_leaf=5,
                       min_samples_split=5)
```

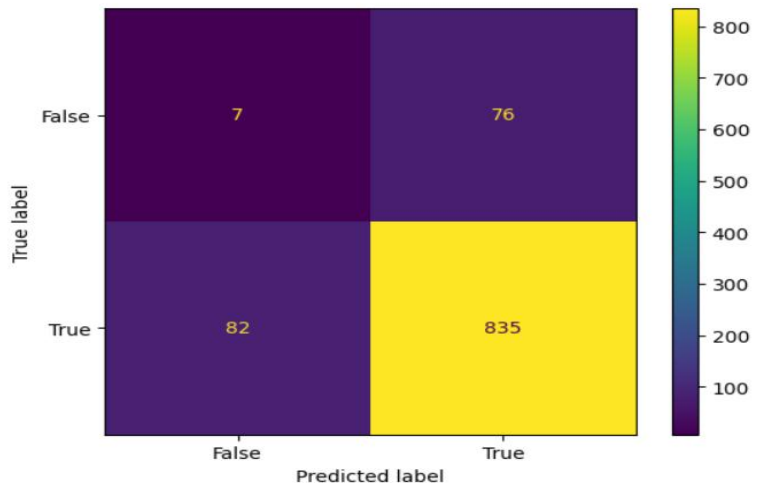
When dealing with imbalanced datasets or when different classes have different costs or consequences for misclassification , other metrics such as precision, recall, F1 score, and area under the receiver operating characteristic (ROC) curve may provide a more comprehensive and meaningful evaluation of a model's performance.

```
[ ] from sklearn import metrics
    confusion_matrix = metrics.confusion_matrix(actual, predicted)
```

```
[ ] cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [False, True])
```

```
[ ] import matplotlib.pyplot as plt
```

```
cm_display.plot()
plt.show()
```



The Confusion Matrix created has four different quadrants:

True Negative (Top-Left Quadrant) = 11 i.e. not defaulter predicted not defaulter

False Positive (Top-Right Quadrant) = 93 i.e. not defaulter predicted defaulter

False Negative (Bottom-Left Quadrant) = 93 i.e. defaulter predicted not defaulter

True Positive (Bottom-Right Quadrant) = 803 i.e. defaulter predicted defaulter

```
[ ]  Accuracy = metrics.accuracy_score(actual, predicted)
     print("Accuracy   :", Accuracy)
     Precision = metrics.precision_score(actual, predicted)
     print("Precision :", Precision)
     Sensitivity_recall = metrics.recall_score(actual, predicted)
     print("Recall    :",Sensitivity_recall)
     Specificity = metrics.recall_score(actual, predicted, pos_label=0)
     print("Specificity  :",Specificity )
     F1_score = metrics.f1_score(actual, predicted)
     print("F1-score  :", F1_score)
```

```
Accuracy   : 0.842
Precision : 0.9165751920965971
Recall    : 0.910577971646674
Specificity  : 0.08433734939759036
F1-score  : 0.913566739606127
```

11.)ACCURACY OF FINAL MODEL:-

The Decision Tree model is considered as the most accurate model because its accuracy is 100.0% and cross validation score is also 100%.

The predicted output for the given input using decision tree model:

```
[ ] df1 = pd.DataFrame({
        'Current Loan Amount':772706.00, 'Credit Score':599.00, 'Annual Income':1735061.00, 'Years in current job':10.00,
        'Monthly Debt':28050.08, 'Years of Credit History':23.40,'Months since last delinquent':63.00,
        'Number of Open Accounts':12.00, 'Number of Credit Problems':1.00, 'Current Credit Balance':151316.00,
        'Maximum Open Credit':395494.00 , 'Bankruptcies':1.00, 'Tax Liens':0.00,
        'Term_Long Term':1.00, 'Term_Short Term':0.00,
        'Home Ownership_Home Mortgage':0.00, 'Home Ownership_Own Home':0.00, 'Home Ownership_Rent':1.00,
        'Purpose_Business Loan':0.00, 'Purpose_Buy House':0.00, 'Purpose_Buy a Car':0.00, 'Purpose_Debt Consolidation':1.00,
        'Purpose_Educational Expenses':0.00, 'Purpose_Home Improvements':0.00, 'Purpose_Medical Bills':0.00,
        'Purpose_Other':0.00, 'Purpose_moving':0.00, 'Purpose_renewable_energy':0.00,
        'Purpose_vacation':0.00, 'Purpose_wedding':0.00
    },index=[0])
```

```
[ ] result = model.predict(df1)
    if result==0:
      print("Not defaulter")
    else:
      print("Defaulter")
```

```
Defaulter
```

correct prediction 24th row entry in sheet

```
[ ] df1 = pd.DataFrame({
        'Current Loan Amount':369908.00, 'Credit Score':520.00, 'Annual Income':1449491.00, 'Years in current job':10.00,
        'Monthly Debt':12924.75, 'Years of Credit History':19.30, 'Months since last delinquent':3.00,
        'Number of Open Accounts':4.00, 'Number of Credit Problems':0.00, 'Current Credit Balance':12882.00,
        'Maximum Open Credit':207042.00 , 'Bankruptcies':0.00, 'Tax Liens':0.00,
        'Term_Long Term':0.00, 'Term_Short Term':1.00,
        'Home Ownership_Home Mortgage':1.00, 'Home Ownership_Own Home':0.00, 'Home Ownership_Rent':0.00,
        'Purpose_Business Loan':0.00, 'Purpose_Buy House':0.00, 'Purpose_Buy a Car':0.00,
        'Purpose_Debt Consolidation':1.00, 'Purpose_Educational Expenses':0.00, 'Purpose_Home Improvements':0.00,
        'Purpose_Medical Bills':0.00, 'Purpose_Other':0.00,
        'Purpose_moving':0.00, 'Purpose_renewable_energy':0.00,
        'Purpose_vacation':0.00, 'Purpose_wedding':0.00
    },index=[0])
```

```
[ ] result = model.predict(df1)
    if result==0:
      print("Not defaulter")
    else:
      print("Defaulter")
```

```
Not defaulter
```

correct prediction 25th row entry in sheet

12.) MODEL INTERPRETATION:-

1. Decision Tree Model

Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.** In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiplebranches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset

- ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***

It is called a decision tree because, similar to a tree, it starts with theroot node, which expands on further branches and constructs a tree-like structure.

In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.** A decision tree simply asks a question, and based on theanswer (Yes/No), it further splits the tree into subtrees.

2)Random Forest:-

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

3)Gradient Boosting Classifier:-

This algorithm builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In eachstage n_classes regression trees fit on the negative gradient of the loss function, e.g. binary or multiclass log loss. Binary classification is a specialcase where only a single regression tree is induced.

4)Logistic regression:-

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independentvariables.

Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be eitherYes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0and 1, it gives the probabilistic values which lie between 0 and 1.

Logistic Regression is much similar to Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems. InLogistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not basedon its weight, etc. Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify newdata using continuous and discrete datasets.

13.)CONCLUSION:-

Considering decision tree as the best model as it is

- Simple to understand and to interpret. Trees can be visualized.
- Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
- Able to handle both numerical and categorical data. However, the scikit-learn implementation does not support categorical variables for now. Other techniques are usually specialized in analyzing datasets that have onlyone type of variable.
- Able to handle multi-output problems

- Uses a white box model. If a given situation is observable in a model,the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.
- Possible to validate a model using statistical tests. That makes it possibleto account for the reliability of the model.
- Performs well even if its assumptions are somewhat violated by the truemodel from which the data were generated.