Loan Prediction Using Machine Learning Algorithms

Sanika Narayanpethkar[1], S. Chandana[2], M. Rishitha[3], Preethi Jeevan[4]

[1,2,3]Student, Sreenidhi Institute of Science and Technology
[4]Associate Professor, Sreenidhi Institute of Science and Technology

ABSTRACT

Loans are a major prerequisite of the present-day world. By this as it were, Banks get a major portion of the entire benefit. It is advantageous for understudies to oversee their instruction and living costs, and for individuals to purchase any kind of extravagance like houses, cars, etc. But when it comes to choosing whether the applicant's profile is pertinent to being allowed a credit or not. Banks must see after many aspects. Giving credit is the most commerce of banks. A noteworthy parcel of the bank's income comes straightforwardly from benefits from credits. Indeed, on the off chance that the bank favors the advance after the confirmation and certification relapse prepare, it cannot be beyond any doubt that the chosen candidate is the genuine candidate. Doing this handle physically requires unused time. Able to anticipate whether a specific candidate is secure, and the whole confirmation handle is robotized with machine fashion. After the presentation of innovation to the world everything began to be automatized. With the introduction of machine learning, the world has seen a modern side of innovation. When the diversion of forecast is played, machine learning gives a few of the most excellent models to test the information for exactness of result. A few of them are broadly utilized. Machine learning calculations are Decision trees, support vector machine, logistic regression, random forest, etc.

I. INTRODUCTION

Nowadays who doesn’t depend on loans. Every person must have taken a loan for some purpose in their life. We take loans for various purposes like home loans, vehicle loans, education loans, etc. Banks provide us with loans based on a certain criterion. The criterion depends on various factors like income, area of living, dependents, job, lifestyle, credit score, etc. Now if we through it manually and check each attribute for every individual it might take up to months to issue the loan. Hence, we use the existing machine learning algorithms to find out which model works best and automatically provide us with a system which tells us whether a certain individual is given the loan or not. Out of all the models used the one with highest accuracy is chosen to be the best fit for prediction problems. We use decision trees to try and predict the accuracy first and then compare the other 5 models to check whether they are better compared to decision trees. This difference is shown in the result after we apply each model to the dataset given. The one which surpasses the accuracy of decision trees is declared best. If not, decision trees is chosen as the best used model.

II. EXISTING SYSTEM

Bank employees manually check the application documents and issue loans to eligible applicants. Viewing all documents requests takes too much time. Therefore, we use an algorithm such as a decision tree to check the correctness of the model. Decision trees help us see how the supervised learning algorithm produces results. Decision trees in machine learning provide a good way to make decisions once they have identified the problem and all possible outcomes. It allows developers to analyze the consequences of a decision, and as the algorithm accesses more data, it can predict the future outcome of the data.

III. PROPOSED SYSTEM

In the planning process, we use advanced algorithms/models to check the accuracy of the results. We use six different machine learning techniques such as Logistic Regression, Support Vector Machine Models, Random Forests, K-nearest Neighbors and Gaussian Naive Bayes Models. We will use the previous data So the machine can analyze and understand the process. The machine will then check for eligible applicants and provide the results to us. We use the most accurate reporting model to create loan applications.

Advantages

- The lending period will be shortened.
- All processes will be automated to avoid people.
- Credits are granted immediately to eligible applicants.

IV. SYSTEM ARCITECHTURE

Considering each dataset each and each information set has both input as well as yield. Taking models into thought a few information models may require colossal sum of information in arrange to urge exact and exact comes about. Whenever we bolster the information, we'll discover it categorical, so we ought to utilize a wide extent of pre-processing procedures in arrange to form all the categorical information and convert it into nonstop. We should check the information moreover after the pre-processing and we ought to envision the dataset to induce curious designs. After this the information is put away within the database and it is recovered from the database and sent for the arrangement handle. The strategies that include pre-processing extricates information at that point it changes over the selected set of qualities into persistent traits and makes the information valuable. Now the experiences given from here are utilized develop a model which is scientific by making utilize of machine learning methods.

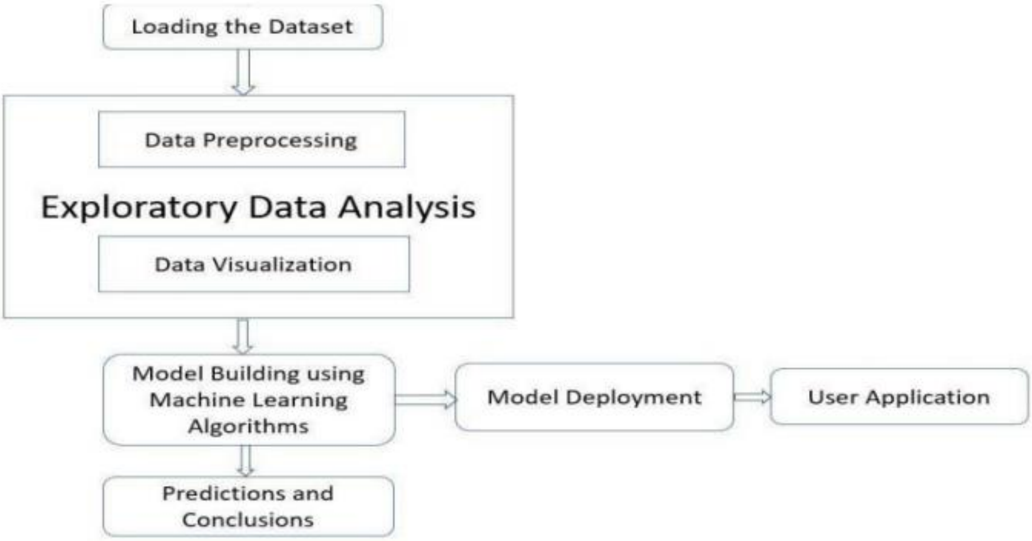

Loading the Dataset
Data Preprocessing
Exploratory Data Analysis
Data Visualization
Model Building using Machine Learning Algorithms
Model Deployment
User Application
Predictions and Conclusions

Figure 1.1 steps

V. IMPLEMENTATION

MODULES

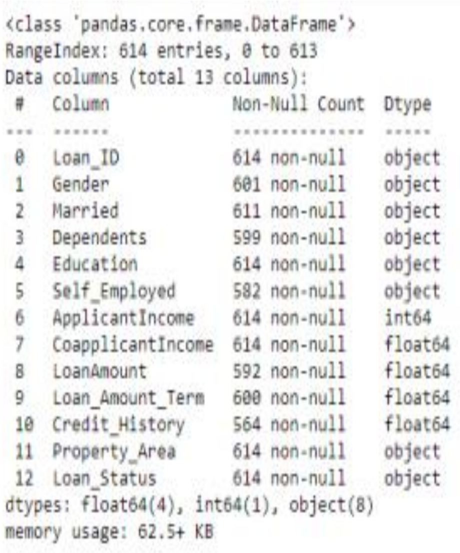
THE IMPLEMENTATION CONSISTS OF:

- Import the libraries
- Read the dataset
- Clean the data
- Check the missing values
- Plot the graphs • Data visualization
- Split the data into training and testing
- Apply the algorithms on the trained and tested data
- Check the accuracy

DATASET

The dataset collected for prognosticating advance disappointment clients is anticipated into Preparing set and testing set. By and large 8020 extent is connected to separate the preparing set and testing set. Test set determining is done. There are various attributes which are considered when collecting dataset. These attributes play a vital role in determining the loan given to an individual.

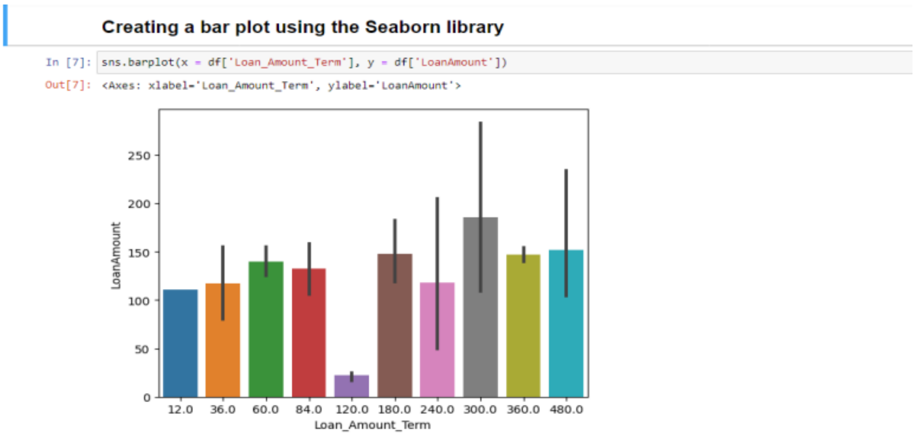
Attributes are



```
<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   LoanAmount         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: 62.5+ KB
```

Figure 1.2 attributes in the dataset

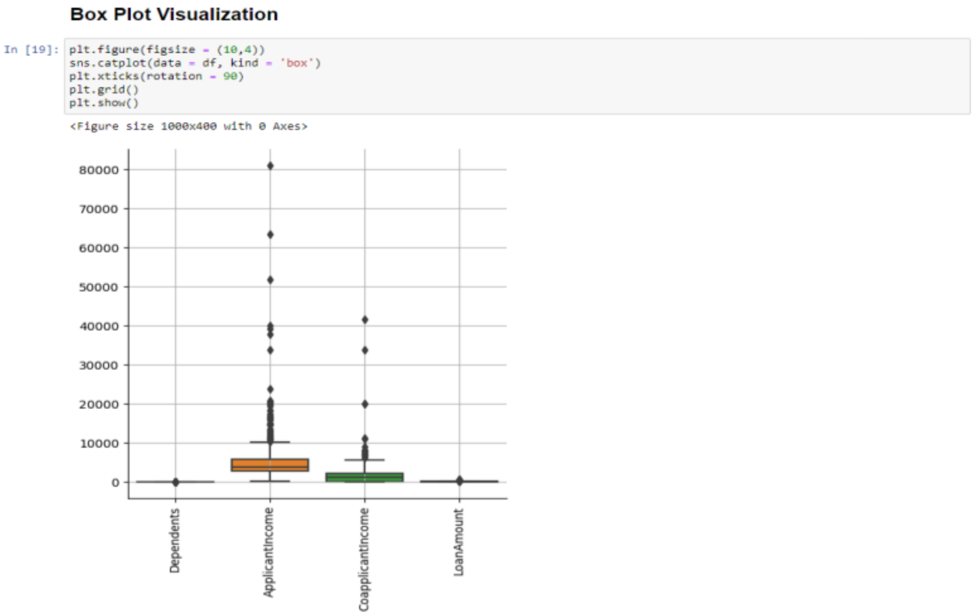
PLOTTING GRAPHS AND VISUALIZATION



Creating a bar plot using the Seaborn library

```
In [7]: sns.barplot(x = df['Loan_Amount_Term'], y = df['LoanAmount'])
Out[7]: <Axes: xlabel='Loan_Amount_Term', ylabel='LoanAmount'>
```

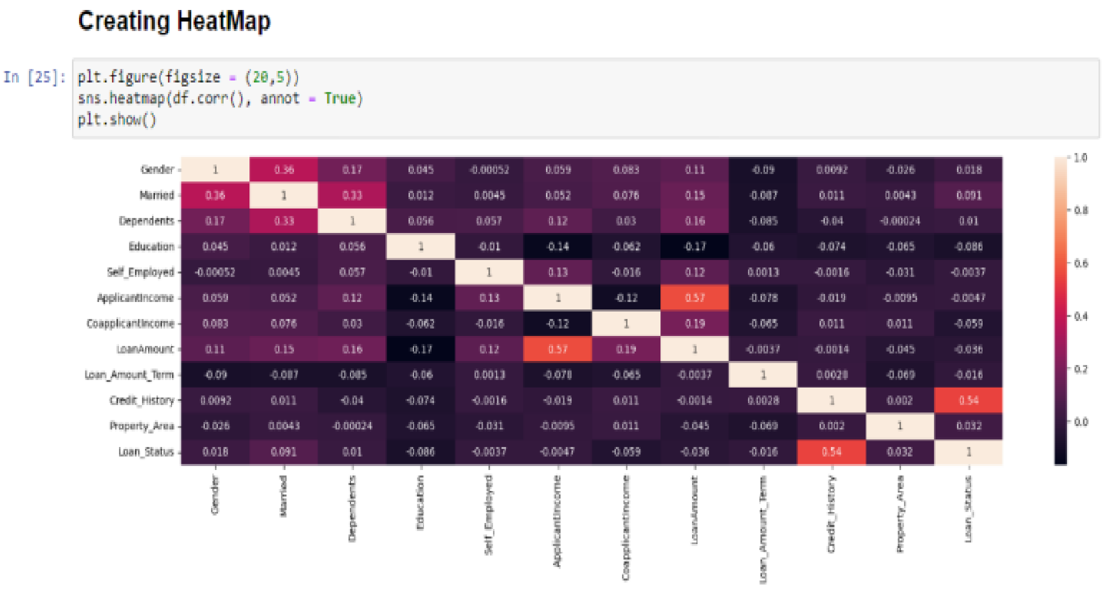
Figure 1.3 visualization using seaborn



Box Plot Visualization

```
In [19]: plt.figure(figsize = (10,4))
         sns.catplot(data = df, kind = 'box')
         plt.xticks(rotation = 90)
         plt.grid()
         plt.show()

<Figure size 1000x400 with 0 Axes>
```

Figure 1.5 box plot visualization



Creating HeatMap

```
In [25]: plt.figure(figsize = (20,5))
         sns.heatmap(df.corr(), annot = True)
         plt.show()
```

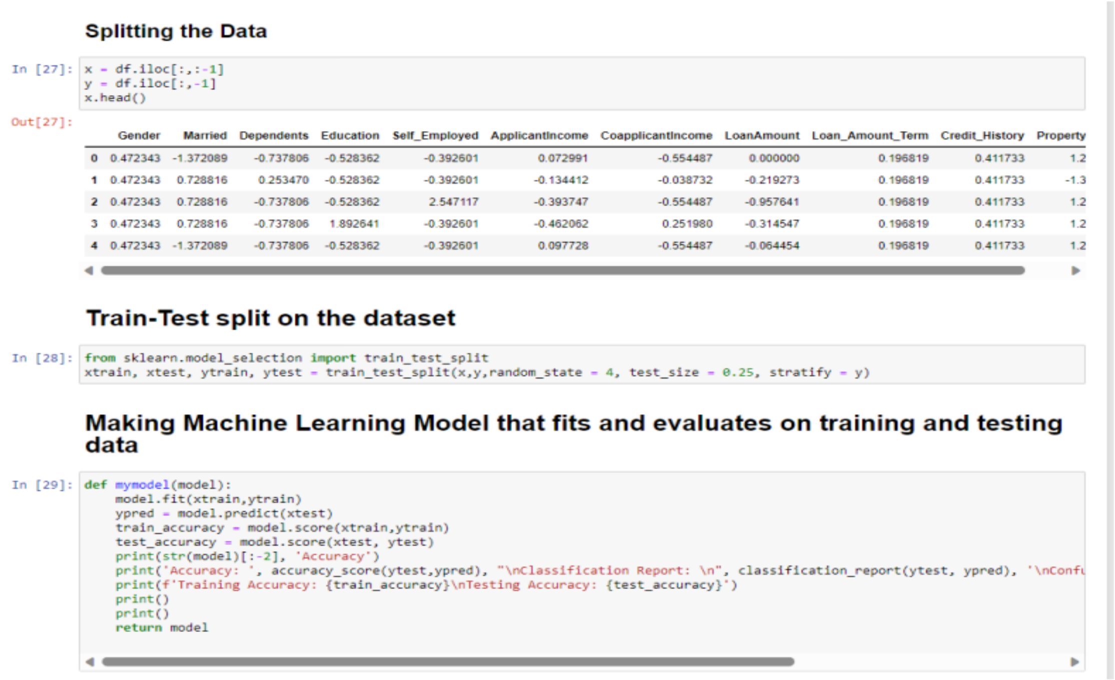
Gender 1 0.36 0.17 0.045 -0.00052 0.059 0.083 0.11 -0.09 0.0092 -0.026 0.018
Married 0.36 1 0.33 0.012 0.0045 0.052 0.076 0.15 -0.087 0.011 0.0043 0.091
Dependents 0.17 0.33 1 0.056 0.057 0.12 0.03 0.16 -0.085 -0.04 -0.00024 0.01
Education 0.045 0.012 0.056 1 -0.01 -0.14 -0.062 -0.17 -0.06 -0.074 -0.065 -0.086
Self_Employed -0.00052 0.0045 0.057 -0.01 1 0.13 -0.016 0.12 0.0013 -0.0016 -0.031 -0.0037
ApplicantIncome 0.059 0.052 0.12 -0.14 0.13 1 -0.12 0.57 -0.078 -0.019 -0.0095 -0.0047
CoapplicantIncome 0.083 0.076 0.03 -0.062 -0.016 -0.12 1 0.19 -0.065 0.011 0.011 -0.059
LoanAmount 0.11 0.15 0.16 -0.17 0.12 0.57 0.19 1 -0.0037 -0.0014 -0.045 -0.036
Loan_Amount_Term -0.09 -0.087 -0.085 -0.06 0.0013 -0.078 -0.065 -0.0037 1 0.0028 -0.069 -0.016
Credit_History 0.0092 0.011 -0.04 -0.074 -0.0016 -0.019 0.011 -0.0014 0.0028 1 0.002 0.54
Property_Area -0.026 0.0043 -0.00024 -0.065 -0.031 -0.0095 0.011 -0.045 -0.069 0.002 1 0.032
Loan_Status 0.018 0.091 0.01 -0.086 -0.0037 -0.0047 -0.059 -0.036 -0.016 0.54 0.032 1
Gender
Married
Dependents
Education
Self_Employed
ApplicantIncome
CoapplicantIncome
LoanAmount
Loan_Amount_Term
Credit_History
Property_Area
Loan_Status
1.0
0.8
0.6
0.4
0.2
0.0

Figure 1.6 heatmap

SPLITTING DATA INTO TRAINING AND TESTING

TRANING SET: The information collection utilized to prepare our show some time recently making an application is alluded to as the preparing set. The preparation set is regularly physically developed, and your demonstration follows the same rules and definitions as those expressed within the hone set.

TESTING SET: A testing set may be an information set on which you apply your demonstration to see in the event that it is redressed and creating the required comes about. A test set is comparative to a model's test.



Splitting the Data

```
In [27]: x = df.iloc[:,:-1]
         y = df.iloc[:,-1]
         x.head()
```

Out[27]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property
0	0.472343	-1.372089	-0.737806	-0.528362	-0.392601	0.072991	-0.554487	0.000000	0.196819	0.411733	1.2
1	0.472343	0.728816	0.253470	-0.528362	-0.392601	-0.134412	-0.038732	-0.219273	0.196819	0.411733	-1.3
2	0.472343	0.728816	-0.737806	-0.528362	2.547117	-0.393747	-0.554487	-0.957641	0.196819	0.411733	1.2
3	0.472343	0.728816	-0.737806	1.892641	-0.392601	-0.462062	0.251980	-0.314547	0.196819	0.411733	1.2
4	0.472343	-1.372089	-0.737806	-0.528362	-0.392601	0.097728	-0.554487	-0.064454	0.196819	0.411733	1.2

Train-Test split on the dataset

```
In [28]: from sklearn.model_selection import train_test_split
         xtrain, xtest, ytrain, ytest = train_test_split(x,y,random_state = 4, test_size = 0.25, stratify = y)
```

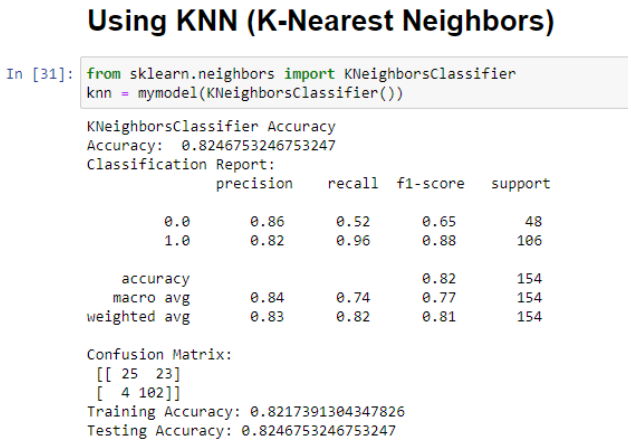
Making Machine Learning Model that fits and evaluates on training and testing data

```
In [29]: def mymodel(model):
             model.fit(xtrain,ytrain)
             ypred = model.predict(xtest)
             train_accuracy = model.score(xtrain,ytrain)
             test_accuracy = model.score(xtest, ytest)
             print(str(model)[:-2], 'Accuracy')
             print('Accuracy: ', accuracy_score(ytest,ypred), "\nClassification Report: \n", classification_report(ytest, ypred), '\nConfu
             print(f'Training Accuracy: {train_accuracy}\nTesting Accuracy: {test_accuracy}')
             print()
             print()
             return model
```

Figure 1.7 splitting the data

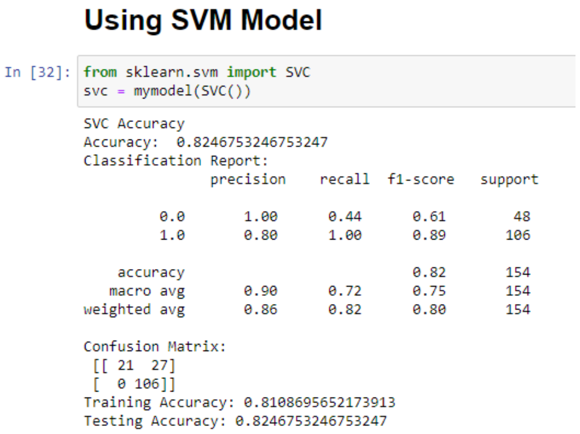
VI. RESULTS

We used 6 algorithms, and each gave a different accuracy:



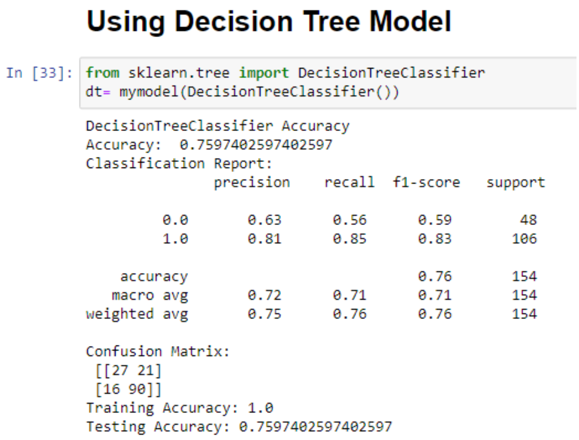
Using KNN (K-Nearest Neighbors)

```
In [31]: from sklearn.neighbors import KNeighborsClassifier
         knn = mymodel(KNeighborsClassifier())

         KNeighborsClassifier Accuracy
         Accuracy:  0.8246753246753247
         Classification Report:
                       precision    recall  f1-score   support

                  0.0       0.86      0.52      0.65        48
                  1.0       0.82      0.96      0.88       106

             accuracy                           0.82       154
            macro avg       0.84      0.74      0.77       154
         weighted avg       0.83      0.82      0.81       154

         Confusion Matrix:
          [[ 25  23]
          [  4 102]]
         Training Accuracy: 0.8217391304347826
         Testing Accuracy: 0.8246753246753247
```



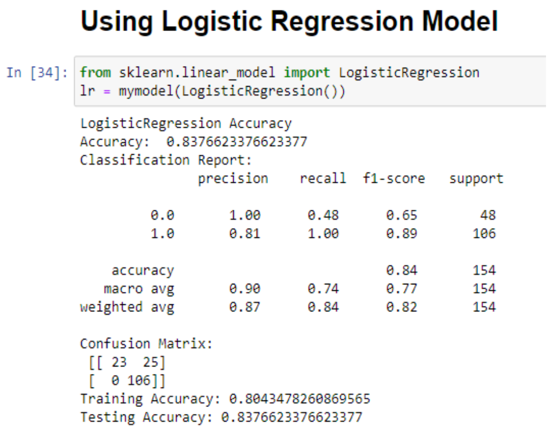
Using SVM Model

```
In [32]: from sklearn.svm import SVC
         svc = mymodel(SVC())

         SVC Accuracy
         Accuracy:  0.8246753246753247
         Classification Report:
                       precision    recall  f1-score   support

                  0.0       1.00      0.44      0.61        48
                  1.0       0.80      1.00      0.89       106

             accuracy                           0.82       154
            macro avg       0.90      0.72      0.75       154
         weighted avg       0.86      0.82      0.80       154

         Confusion Matrix:
          [[ 21  27]
          [  0 106]]
         Training Accuracy: 0.8108695652173913
         Testing Accuracy: 0.8246753246753247
```



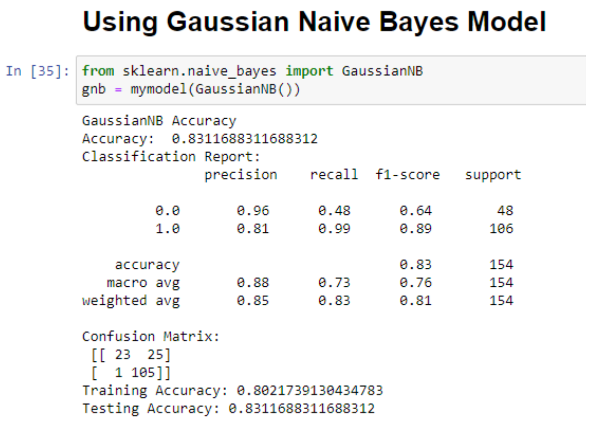
Using Decision Tree Model

```
In [33]: from sklearn.tree import DecisionTreeClassifier
         dt= mymodel(DecisionTreeClassifier())

         DecisionTreeClassifier Accuracy
         Accuracy:  0.7597402597402597
         Classification Report:
                       precision    recall  f1-score   support

                  0.0       0.63      0.56      0.59        48
                  1.0       0.81      0.85      0.83       106

             accuracy                           0.76       154
            macro avg       0.72      0.71      0.71       154
         weighted avg       0.75      0.76      0.76       154

         Confusion Matrix:
          [[27 21]
          [16 90]]
         Training Accuracy: 1.0
         Testing Accuracy: 0.7597402597402597
```



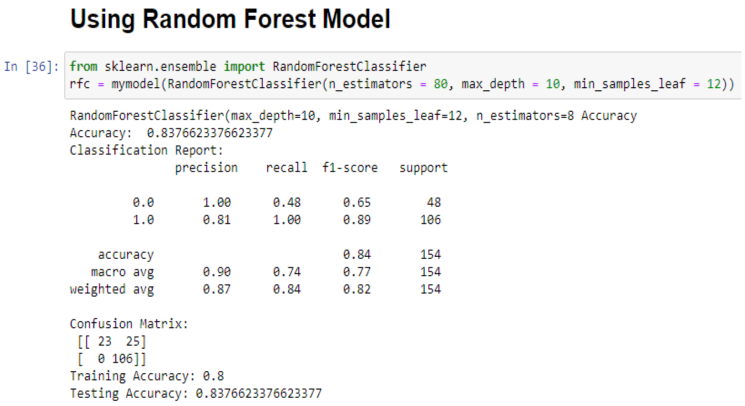
Using Logistic Regression Model

```
In [34]: from sklearn.linear_model import LogisticRegression
         lr = mymodel(LogisticRegression())

         LogisticRegression Accuracy
         Accuracy:  0.8376623376623377
         Classification Report:
                       precision    recall  f1-score   support

                  0.0       1.00      0.48      0.65        48
                  1.0       0.81      1.00      0.89       106

             accuracy                           0.84       154
            macro avg       0.90      0.74      0.77       154
         weighted avg       0.87      0.84      0.82       154

         Confusion Matrix:
          [[ 23  25]
          [  0 106]]
         Training Accuracy: 0.8043478260869565
         Testing Accuracy: 0.8376623376623377
```



Using Gaussian Naive Bayes Model

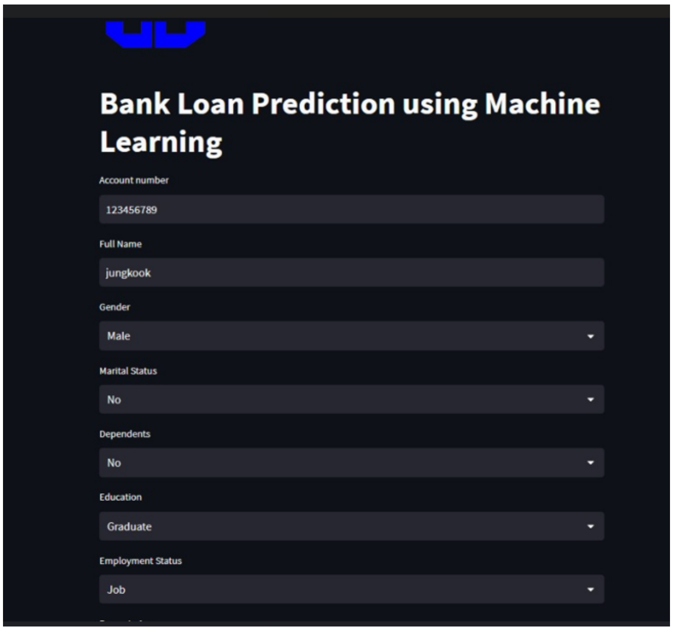
```
In [35]: from sklearn.naive_bayes import GaussianNB
         gnb = mymodel(GaussianNB())

         GaussianNB Accuracy
         Accuracy:  0.8311688311688312
         Classification Report:
                       precision    recall  f1-score   support

                  0.0       0.96      0.48      0.64        48
                  1.0       0.81      0.99      0.89       106

             accuracy                           0.83       154
            macro avg       0.88      0.73      0.76       154
         weighted avg       0.85      0.83      0.81       154

         Confusion Matrix:
          [[ 23  25]
          [  1 105]]
         Training Accuracy: 0.8021739130434783
         Testing Accuracy: 0.8311688311688312
```



Using Random Forest Model

```
In [36]: from sklearn.ensemble import RandomForestClassifier
         rfc = mymodel(RandomForestClassifier(n_estimators = 80, max_depth = 10, min_samples_leaf = 12))

         RandomForestClassifier(max_depth=10, min_samples_leaf=12, n_estimators=8 Accuracy
         Accuracy:  0.8376623376623377
         Classification Report:
                       precision    recall  f1-score   support

                  0.0       1.00      0.48      0.65        48
                  1.0       0.81      1.00      0.89       106

             accuracy                           0.84       154
            macro avg       0.90      0.74      0.77       154
         weighted avg       0.87      0.84      0.82       154

         Confusion Matrix:
          [[ 23  25]
          [  0 106]]
         Training Accuracy: 0.8
         Testing Accuracy: 0.8376623376623377
```

GUI RESULTS



Bank Loan Prediction using Machine Learning

Account number: 123456789

Full Name: jungkook

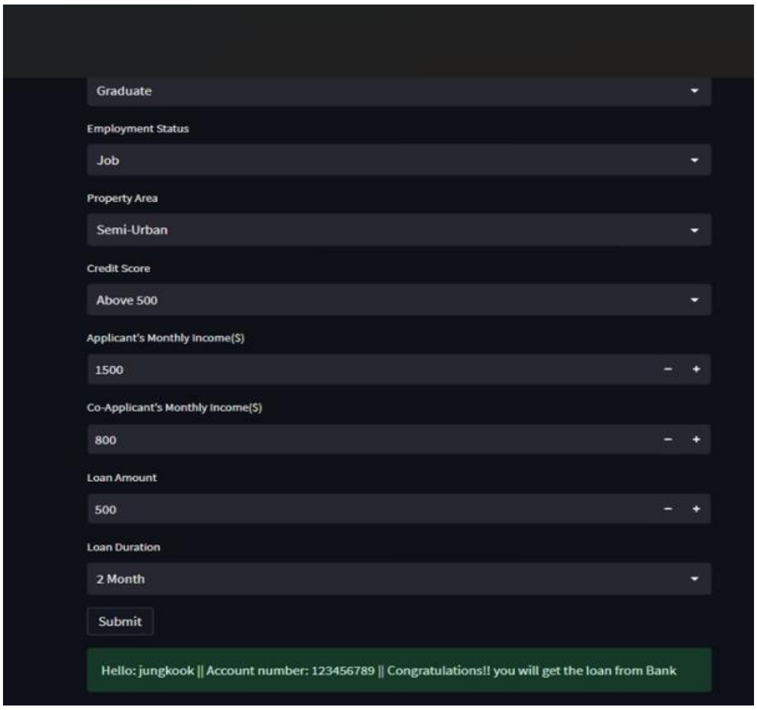
Gender: Male

Marital Status: No

Dependents: No

Education: Graduate

Employment Status: Job



Graduate

Employment Status: Job

Property Area: Semi-Urban

Credit Score: Above 500

Applicant's Monthly Income($): 1500

Co-Applicant's Monthly Income($): 800

Loan Amount: 500

Loan Duration: 2 Month

Hello: jungkook || Account number: 123456789 || Congratulations!! you will get the loan from Bank

VII. CONCLUSION

With cautious investigation of dynamic substance and members' limitations, it can be securely done that things are qualified individuals. This app works well and meets all the requirements of an investor. This part can be openly associated in numerous other frameworks. Computer glitches, there are numerical information around the breach points, and the weight of the foremost important highlights has been settled in prophet builder, within the future the program title can be more secure, more solid and more dependable. Energetic overwhelming adaptation. Within the future, this prophet module can be combined with the structure of the computerized process. The framework employs ancient preparing information to prepare future program, so the unused test information ought to be coordinates with the preparing information after a few times.

VIII. REFERENCES

1. Kumar Arun, Garg Ishan, Kaur Sanmeet, May-Jun. 2016. Loan Approval Prediction based on Machine Learning Approach, IOSR Journal of Computer Engineering (IOSR-JCE)
2. Wei Li, Shuai Ding, Yi Chen, and Shanlin Yang, Heterogeneous Ensemble for Default Prediction of Peer-to-Peer Lending in China, Key Laboratory of Process Optimization and Intelligent Decision-Making, Ministry of Education, Hefei University of Technology, Hefei 2009, China
3. Short-term prediction of Mortgage default using ensembled machine learning models, Jesse C.Sealand on July 20, 2018.
4. Clustering Loan Applicants based on Risk Percentage using K-Means Clustering Techniques, Dr. K. Kavitha, International Journal of Advanced Research in Computer Science and Software Engineering.