



LONE PREDICTION SYSTEM



AUGUST 2, 2024
R. P. C. THENUKA

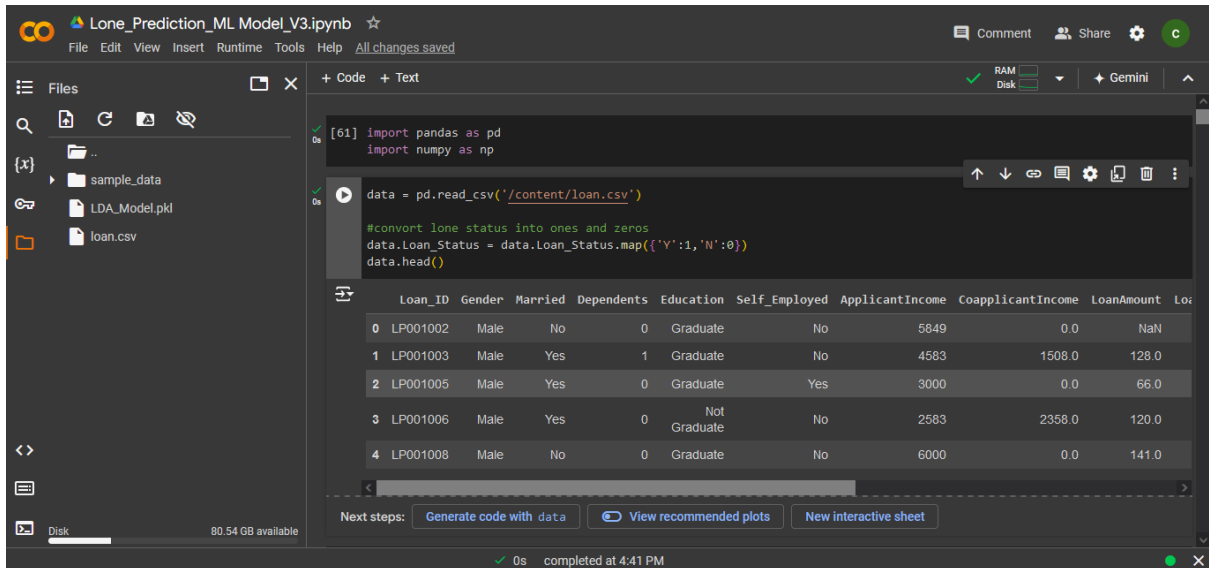
Introduction

Getting a Loan from the Bank is a challenging activity because of many factors. Such as the ability to re-pay, Current economic status, Loan repayment History, etc. Nowadays In Sri Lanka, eligibility of getting a loan is found manually or on paper. This mechanism of the procedure is time-consuming and costly. As a solution to this issue, I created a Machine Learning based Loan Prediction System. By using this Application users can easily find out whether the users can get a loan or not without time consuming. It is an efficient and practical solution. This solution is beneficial not only for users but also for banks. It provides a broader understanding of the factors affecting loan approval, helping both parties make informed decisions. For users, it clarifies the requirements and improves their chances of loan approval. For banks, it helps assess the risk and manage their lending processes more effectively.

I used different factors to train the machine learning model. Such as gender, Marital Status, dependents, education, employment status, income, co-applicant income, loan amount, loan-amount terms, credit history, and properties. I downloaded the dataset from Kaggle. To clean and preprocess the dataset I used Python language and google colab environment. Streamlit library used for creating web application.

Development

- ❖ To create a Machine Learning Model, Firstly I load the dataset on the google colab.



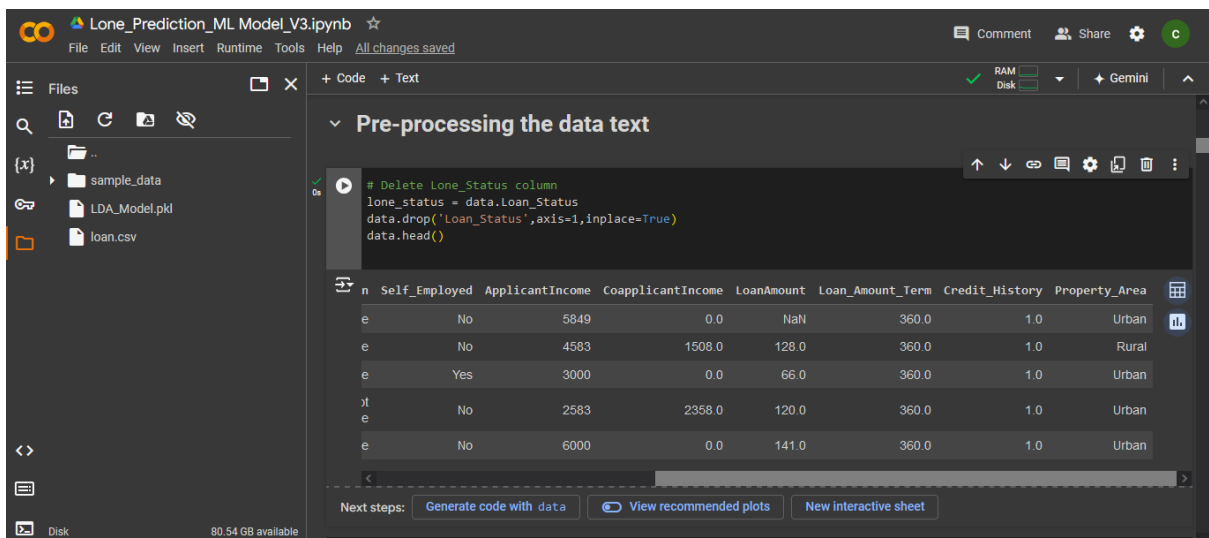
```
[61] import pandas as pd
import numpy as np

data = pd.read_csv('/content/loan.csv')

#convert lone status into ones and zeros
data.Loan_Status = data.Loan_Status.map({'Y':1,'N':0})
data.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	

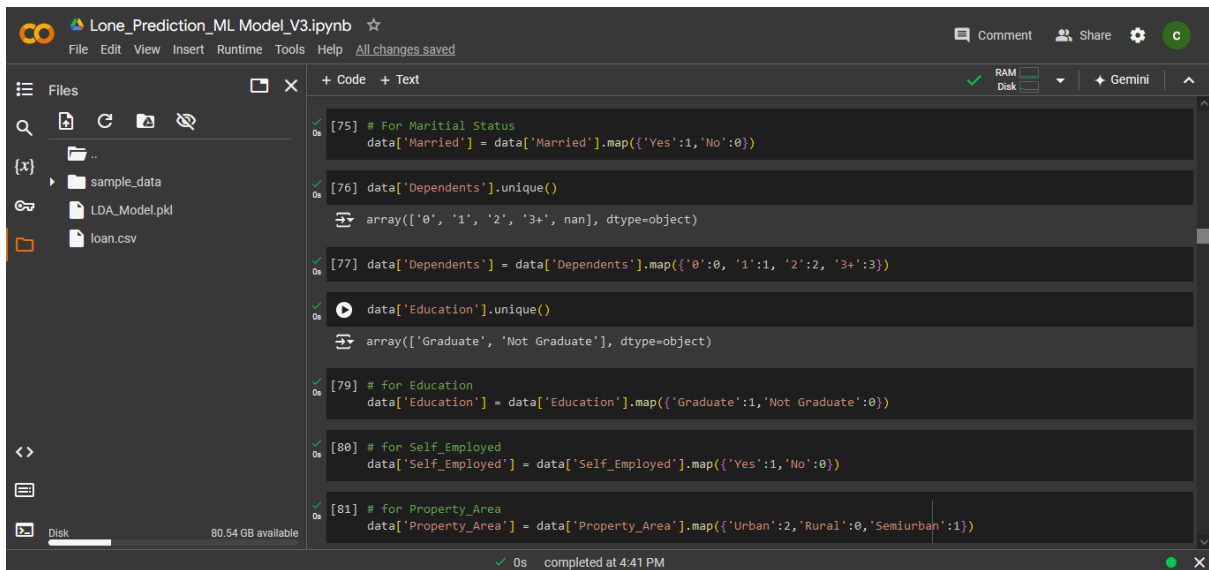
- ❖ After loading the dataset, I create a new variable called “lone_status” and assign the “Lone_Status” column’s data to the “lone_status” variable. Then delete Lone_Status column from the dataset because I do not need this column to train the model. Only I need this column for testing purposes.



```
# Delete Lone_Status column
lone_status = data.Loan_Status
data.drop('Loan_Status',axis=1,inplace=True)
data.head()
```

	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area
0	No	5849	0.0	NaN	360.0	1.0	Urban
1	No	4583	1508.0	128.0	360.0	1.0	Rural
2	Yes	3000	0.0	66.0	360.0	1.0	Urban
3	No	2583	2358.0	120.0	360.0	1.0	Urban
4	No	6000	0.0	141.0	360.0	1.0	Urban

❖ Then I did Label Encoding for the categorical column's data.



The screenshot shows a Jupyter Notebook titled 'Lone_Prediction_ML_Model_V3.ipynb'. The left sidebar displays a file explorer with a folder named 'sample_data' containing 'LDA_Model.pkl' and 'loan.csv'. The main code area contains several cells performing label encoding:

```
[75] # For Marital Status
data['Married'] = data['Married'].map({'Yes':1,'No':0})

[76] data['Dependents'].unique()
array(['0', '1', '2', '3+', nan], dtype=object)

[77] data['Dependents'] = data['Dependents'].map({'0':0, '1':1, '2':2, '3+':3})

[78] data['Education'].unique()
array(['Graduate', 'Not Graduate'], dtype=object)

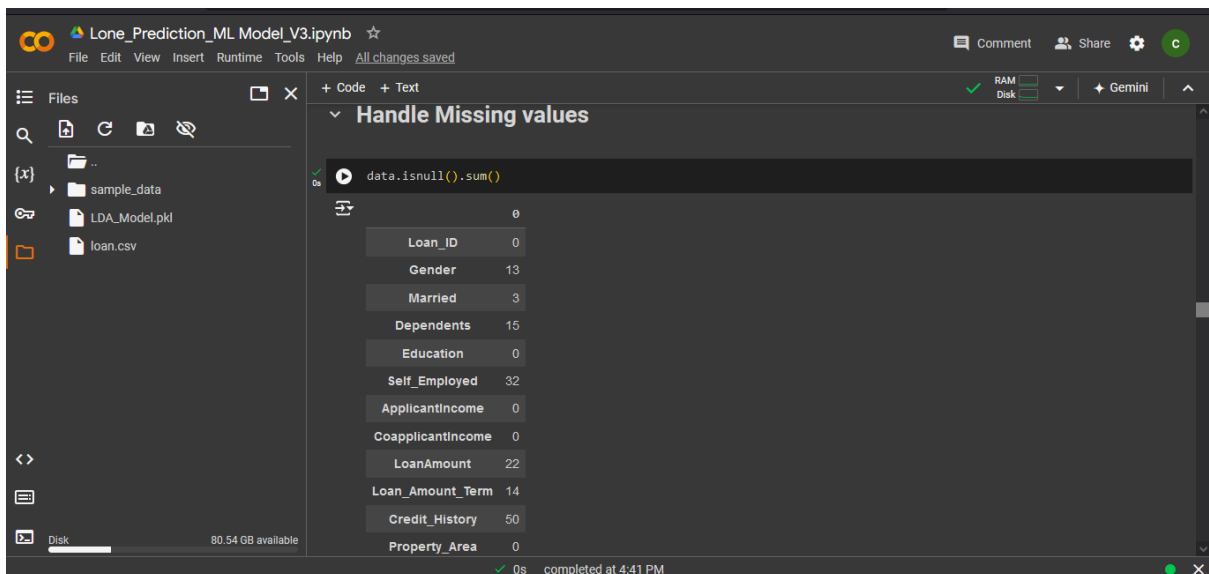
[79] # for Education
data['Education'] = data['Education'].map({'Graduate':1,'Not Graduate':0})

[80] # for Self_Employed
data['Self_Employed'] = data['Self_Employed'].map({'Yes':1,'No':0})

[81] # for Property_Area
data['Property_Area'] = data['Property_Area'].map({'Urban':2,'Rural':0,'Semiurban':1})
```

The status bar at the bottom indicates 'completed at 4:41 PM'.

❖ After that I check the missing values of my dataset.



The screenshot shows the same Jupyter Notebook with a new cell titled 'Handle Missing values' containing the following code:

```
data.isnull().sum()
```

The output of the code is displayed as a table:

Loan_ID	0
Gender	13
Married	3
Dependents	15
Education	0
Self_Employed	32
ApplicantIncome	0
CoapplicantIncome	0
LoanAmount	22
Loan_Amount_Term	14
Credit_History	50
Property_Area	0

The status bar at the bottom indicates 'completed at 4:41 PM'.

- ❖ After checking the missing values, I fill all those missing values with the most suitable data.

The screenshot shows a Jupyter Notebook interface with the following code cells:

```
[85]
dtype: int64

[86] data['Gender'].fillna(np.random.randint(0,2),inplace=True)

[87] data['Married'].fillna(np.random.randint(0,2),inplace=True)

[88] data['Dependents'].fillna(data['Dependents'].median(),inplace=True)

[89] data['Self_Employed'].fillna(np.random.randint(0,2),inplace=True)

[90] data['LoanAmount'].fillna(data['LoanAmount'].median(),inplace=True)

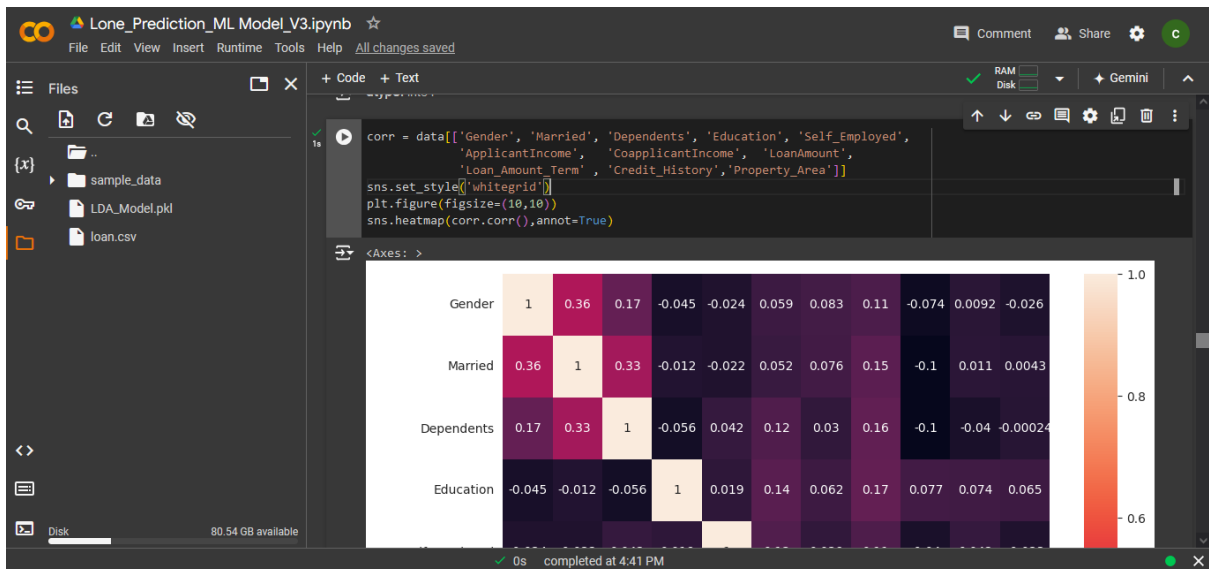
[91] data['Loan_Amount_Term'].fillna(data['Loan_Amount_Term'].mean(),inplace=True)

[92] data['Credit_History'].fillna(np.random.randint(0,2),inplace=True)

[93] data.isnull().sum()
```

The output of the last cell is 0, indicating no missing values remain.

- ❖ Then, I create a heatmap with the all-numeric columns. In this stage, I could find a strong co-relationship between the “Loan Amount” and the “Applicant's Income” it was 0.57.





❖ After the data preprocessing, I split data into X and Y for training and testing purposes.

```
Lone_Prediction_ML_Model_V3.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[99] !pip install scikit-learn
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.3.2)
Requirement already satisfied: numpy<2.0, >=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.2)
Requirement already satisfied: scipy<1.5.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3)

[100] train_X = data.iloc[:461] #first 500 rows
train_y = lone_status.iloc[:461]

[101] from sklearn.model_selection import train_test_split
train_X, test_X, train_y, test_y = train_test_split(train_X, train_y, random_state=0)

[102] train_X.head()
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term
7	1.0	1.0	3.0	1	0.0	3036	2504.0	158.0	
157	1.0	1.0	1.0	1	0.0	9538	0.0	187.0	
382	0.0	0.0	0.0	1	0.0	6000	0.0	156.0	
22	1.0	1.0	0.0	0	0.0	2600	1041.0	116.0	

- ❖ I check the accuracy of my model by using six machine learning models like Decision tree Classifier, Logistic Regression, Linear Discriminant Analysis, K – Nearest Neighbors, Naïve Bayes, SVM, and Random Forest Classifier.

The screenshot shows a Jupyter Notebook interface with the following code in the first cell:

```
[105] from sklearn.tree import DecisionTreeClassifier
      from sklearn.svm import SVC
      from sklearn.neighbors import KNeighborsClassifier
      from sklearn.naive_bayes import GaussianNB
      from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
      from sklearn.linear_model import LogisticRegression
      from sklearn.ensemble import RandomForestClassifier
```

The second cell is titled "Fit the all ML models" and contains the following code:

```
models = []
models.append(('Logistic Regression', LogisticRegression()))
models.append(('Linear Discriminant Analysis', LinearDiscriminantAnalysis()))
models.append(('K- Nearest Neighbors', KNeighborsClassifier()))
models.append(('Decision Tree Classifier', DecisionTreeClassifier()))
models.append(('Naive Bayes', GaussianNB()))
models.append(('Support Vector Machine', SVC()))
models.append(('Random Forest Classifier', RandomForestClassifier()))
```

The notebook interface shows a file explorer on the left with files like 'sample_data', 'LDA_Model.pkl', and 'loan.csv'. The status bar at the bottom indicates "completed at 4:41 PM".

- ❖ After that I used Linear Discriminant Analysis for my model training because this model performs 0.84 accuracy.

The screenshot shows a Jupyter Notebook interface with the following code in the first cell:

```
[109] RandomForestClassifier()
      Random Forest Classifier 0.773866
```

The second cell contains the following code:

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

LDA = LinearDiscriminantAnalysis()
LDA.fit(train_X, train_y)
pred = LDA.predict(test_X)
print("Model Accuracy: ", accuracy_score(test_y, pred))
print(confusion_matrix(test_y, pred))
print(classification_report(test_y, pred))
```

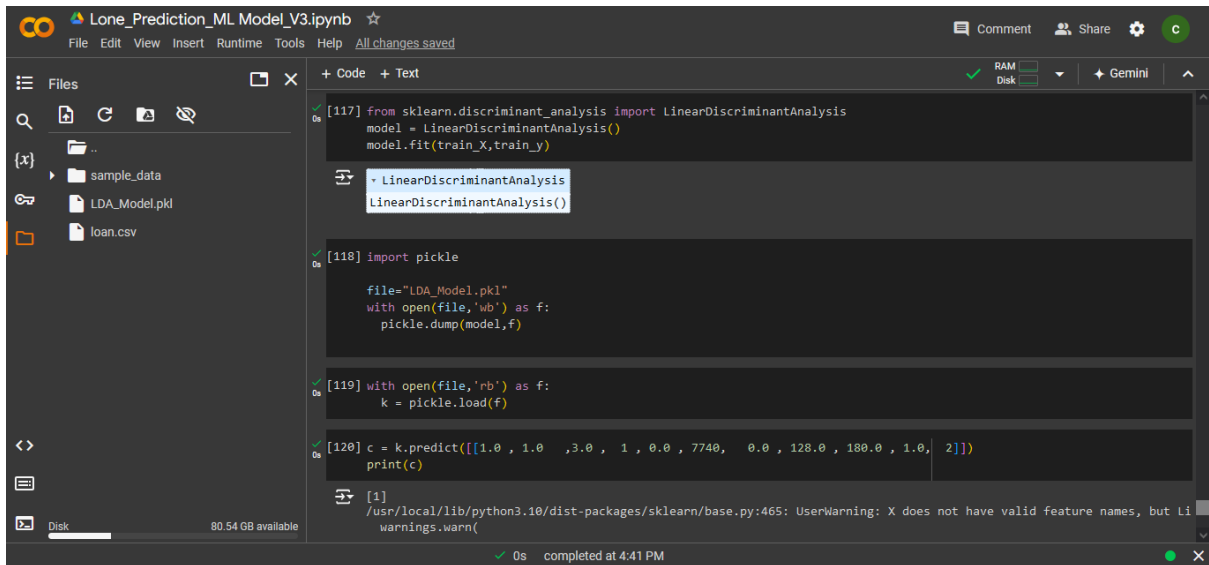
The output of the second cell is as follows:

```
Model Accuracy: 0.8362068965517241
[[11 17]
 [ 2 86]]
```

	precision	recall	f1-score	support
0	0.85	0.39	0.54	28
1	0.83	0.98	0.90	88
accuracy			0.84	116
macro avg	0.84	0.69	0.72	116
weighted avg	0.84	0.84	0.81	116

The notebook interface shows a file explorer on the left with files like 'sample_data', 'LDA_Model.pkl', and 'loan.csv'. The status bar at the bottom indicates "completed at 4:41 PM".

❖ Then I create a model and download to my pc.



```
[117] from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
model = LinearDiscriminantAnalysis()
model.fit(train_X, train_y)

LinearDiscriminantAnalysis
LinearDiscriminantAnalysis()

[118] import pickle

file = "LDA_Model.pkl"
with open(file, 'wb') as f:
    pickle.dump(model, f)

[119] with open(file, 'rb') as f:
    k = pickle.load(f)

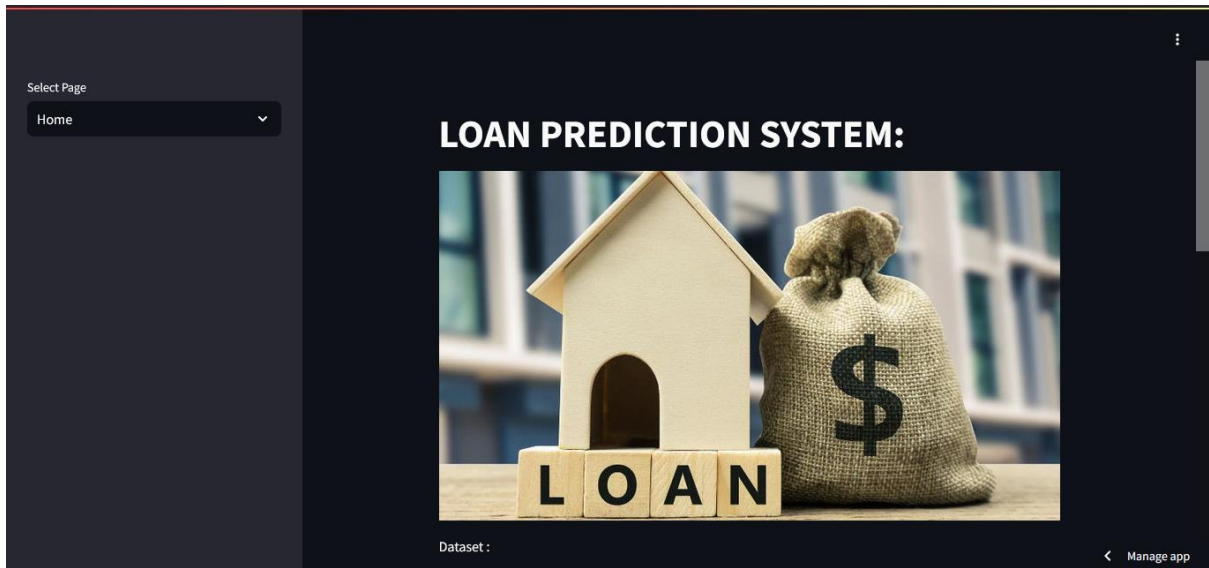
[120] c = k.predict([[1.0, 1.0, 3.0, 1, 0.0, 7740, 0.0, 128.0, 180.0, 1.0, 2]])
print(c)

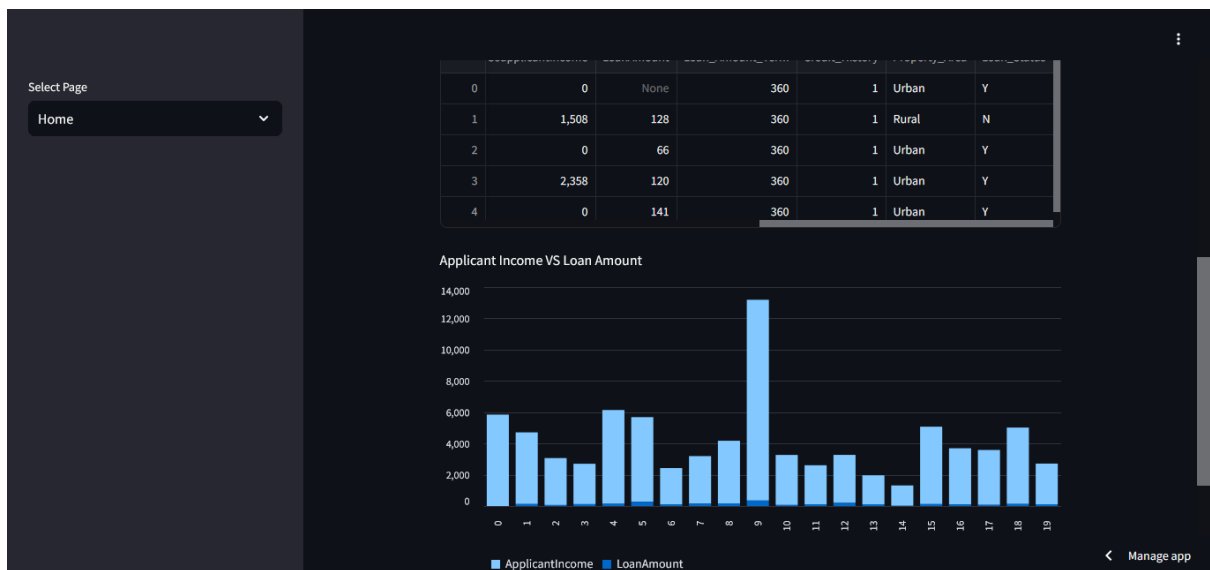
[1]
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:465: UserWarning: X does not have valid feature names, but Li
warnings.warn(

0s completed at 4:41 PM
```

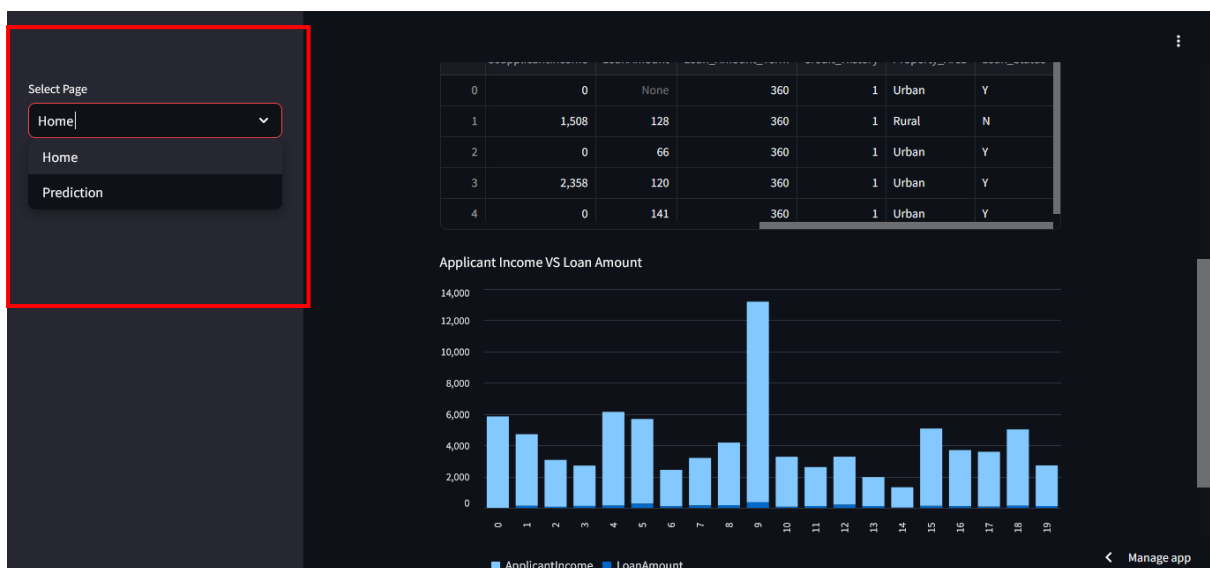
❖ Finally, I created a streamlit application by using streamlit library and deploy my application to the streamlit cloud. My streamlit Application you can see in below,

Home Page





❖ users can click the left-top corner drop-down list to redirect to the prediction page.



Prediction Page

Select Page
Prediction

Predict Your Status of Bank Loan

Your Full Name

Gender

Male

Marriage

Yes

Dependencies

No

Education

Graduate

Self-Employed

Manage app

Select Page
Prediction

Rural

Credit History

Yes

Applicant's Monthly Income(\$)

0

Co-Applicant's Monthly Income(\$)

0

Loan Amount

0

Loan Duration

2 months

Predict

Manage app

Prediction

Select Page
Prediction

No

Applicant's Monthly Income(\$)
500000

Co-Applicant's Monthly Income(\$)
0

Loan Amount
500

Loan Duration
8 months

Predict

Hello! Charana Congratulations! You can get a loan.

Manage app

Select Page
Prediction

No

Applicant's Monthly Income(\$)
50

Co-Applicant's Monthly Income(\$)
0

Loan Amount
500

Loan Duration
8 months

Predict

Hello! Charana You cannot get a loan, sorry.

Manage app

Links

❖ My Git hub link and Streamlit URL you can see in below box,

Git-Hub Link:

<https://github.com/Thenuka09/Lone-Prediction-System>

Streamlit URL:

<https://lone-prediction-system.streamlit.app/>