**A2:- Loan Amount Prediction Using Linear Regression and**

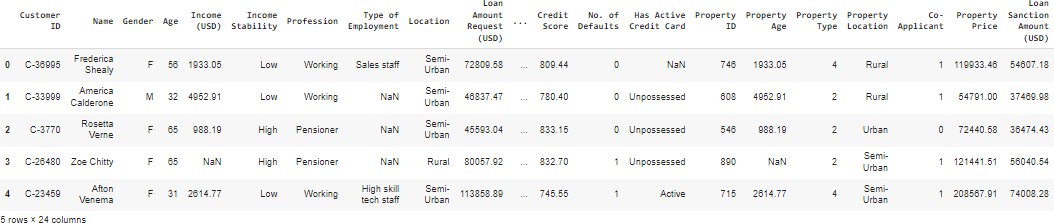
**Visualization**

**Aim**

To Develop a python program to predict the loan amount to be sanctioned using Linear Regression (LR) Model using Scikit-learn library. Visualize the features from the dataset and interpret the results obtained by the model using Matplotlib library.

## Code and output

Loading the dataset:



import numpy as np import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler,LabelEncoder from sklearn.linear\_model import LinearRegression

from sklearn.metrics import accuracy\_score,classification\_report,confusion\_matrix,mean\_squared\_error,r2

\_score

import matplotlib.pyplot as plt import seaborn as sns

#Load train dataset train\_df=pd.read\_csv('/content/train.csv') train\_df.head(5)

Pre-Processing the data (Handling missing values, Encoding, Normalization, Standardization).

#Preprocessing

indexes = train\_df[train\_df['Loan Sanction Amount (USD)'].isna()].index

#Loan Amount empty is removed(invalid) train\_df.drop(index=indexes, inplace=True) #Loan Amount < 0

indexes1 = train\_df[train\_df['Loan Sanction Amount (USD)'] < 0].index train\_df.drop(index=indexes1,inplace=True)

#Coapplicant < 0 is made 0

indexes10 = train\_df[train\_df['Co-Applicant']<0].index train\_df.drop(index=indexes10,inplace=True)

#property age empty is removed

indexes11 = train\_df[train\_df['Property Age'].isna()].index train\_df.drop(index=indexes11,inplace=True)

#credit score null is

indexes1 = train\_df[train\_df['Credit Score'].isna()].index train\_df.drop(index=indexes1,inplace=True)

indexes2 = train\_df[train\_df['Income (USD)'].isna()].index train\_df.drop(index=indexes2,inplace=True)

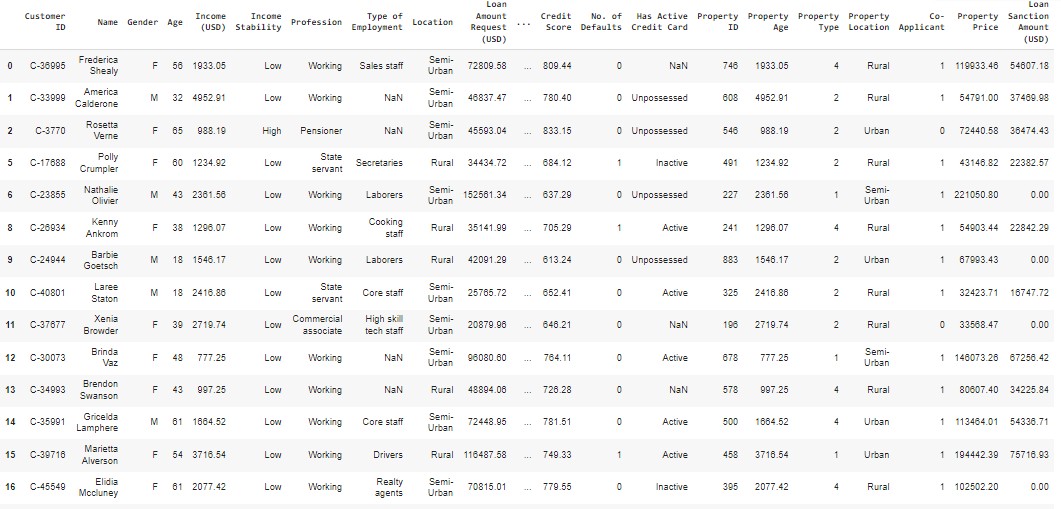
indexes4 = train\_df[train\_df['Current Loan Expenses (USD)'].isna()].index train\_df.drop(index=indexes4,inplace=True)

indexes5 = train\_df[train\_df['Current Loan Expenses (USD)'] < 0].index train\_df.drop(index=indexes5,inplace=True)

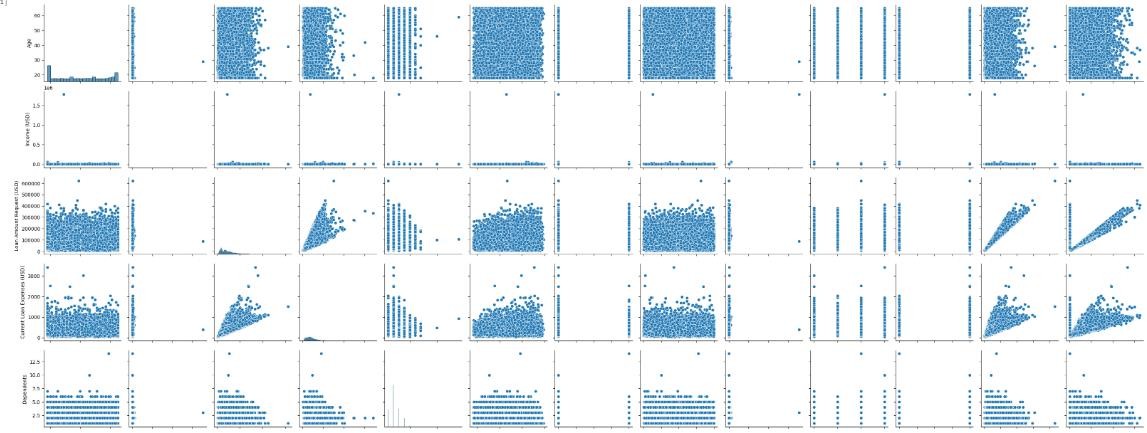
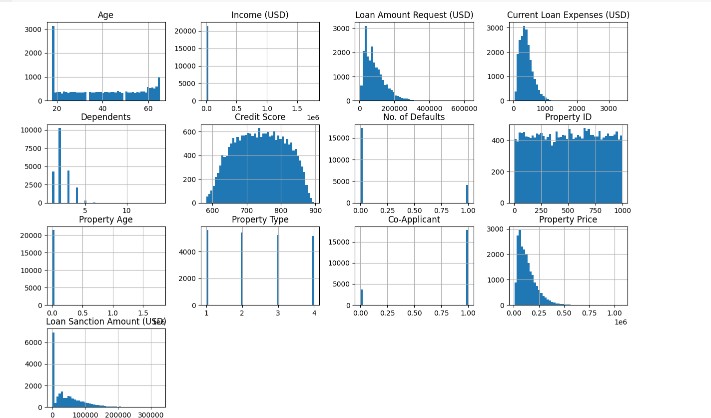
indexes14 = train\_df[train\_df['Dependents'].isna()].index train\_df.drop(index=indexes14,inplace=True)

indexes3 = train\_df[train\_df['Property Price']<0].index

train\_df.drop(index=indexes3,inplace=True)



Exploratory Data Analysis.

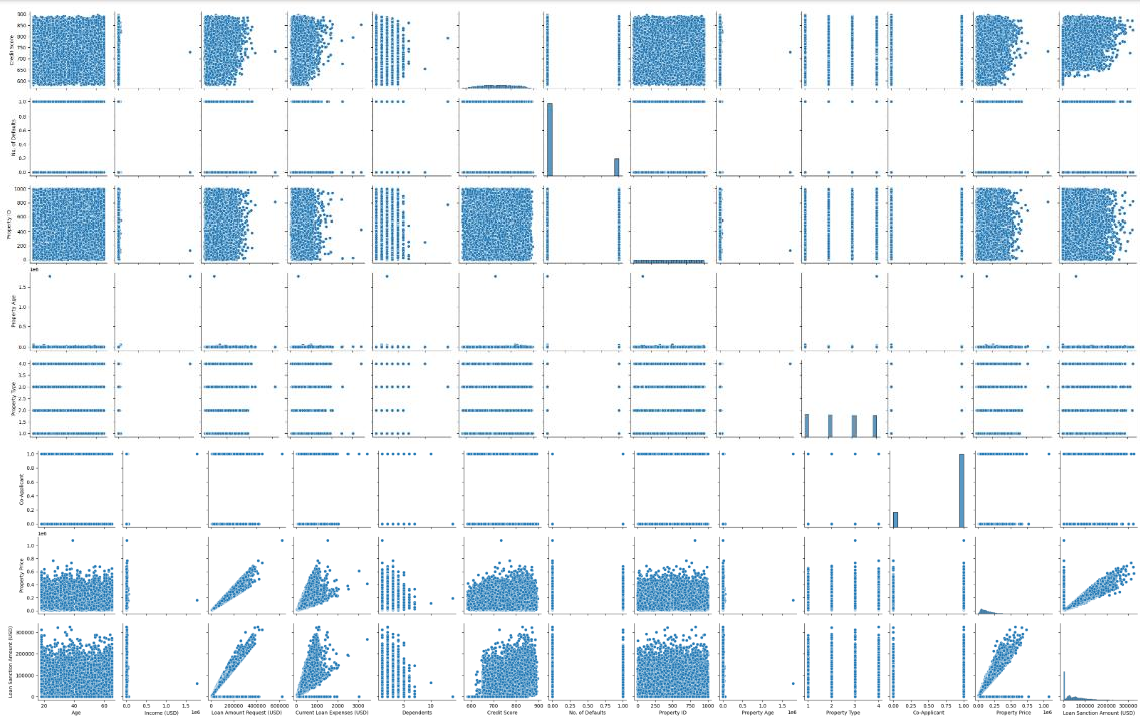


#Exploratory data analysis - Histogram train\_df.hist(bins=50,figsize=(15,10))

plt.show()

#Exploratory data analysis - Pairplot sns.pairplot(train\_df)

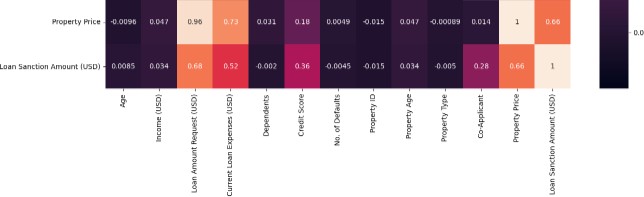
plt.show()



Feature Engineering techniques.

# Heatmap plt.figure(figsize=(15,15))

sns.heatmap(train\_df.corr(),annot=True)

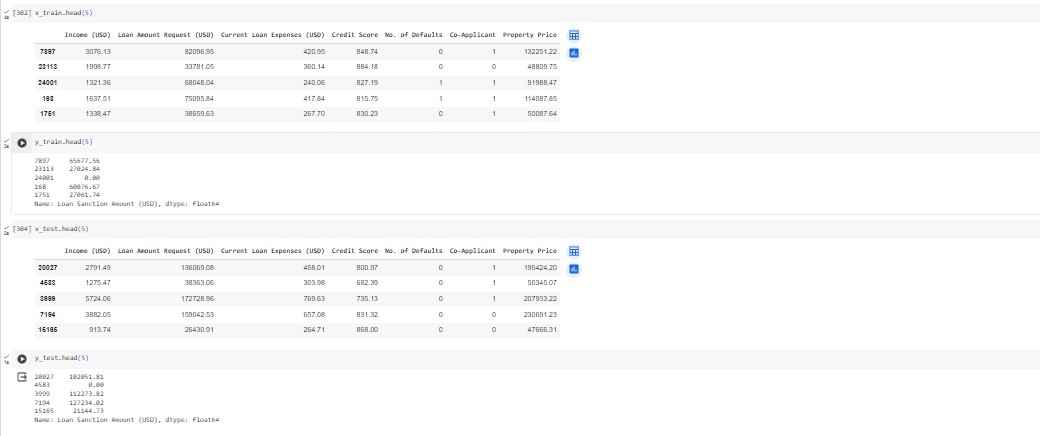


Split the data into training and testing sets.

y = train\_df['Loan Sanction Amount (USD)']

x = train\_df[['Loan Amount Request (USD)','Current Loan Expenses (USD)','Credit Score','No. of Defaults','Co-Applicant','Property Price']] x\_train,x\_test,y\_train,y\_test =

train\_test\_split(x,y,test\_size=0.2,random\_state=42)

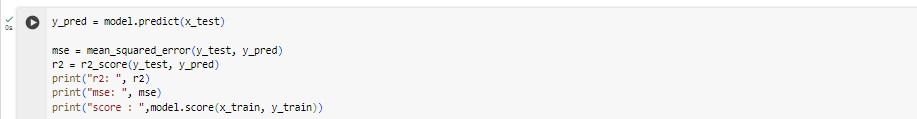


Train the model

Test the model.

y\_pred = model.predict(x\_test)

Measure the performance of the trained model.



#Measure the performance of the trained model mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred) print("Mean Squared Error:", mse) print("R-squared Score:", r2)



Represent the results using graphs.

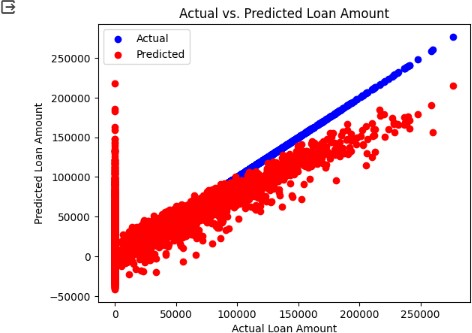
# Scatter plot for actual loan amounts plt.scatter(y\_test, y\_test, color='blue', label='Actual')

# Scatter plot for predicted loan amounts plt.scatter(y\_test, y\_pred, color='red', label='Predicted')

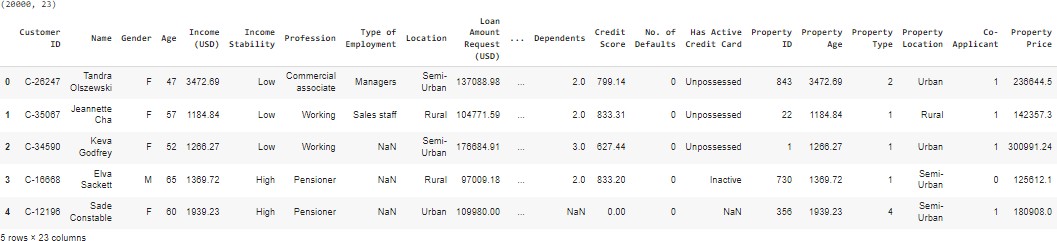
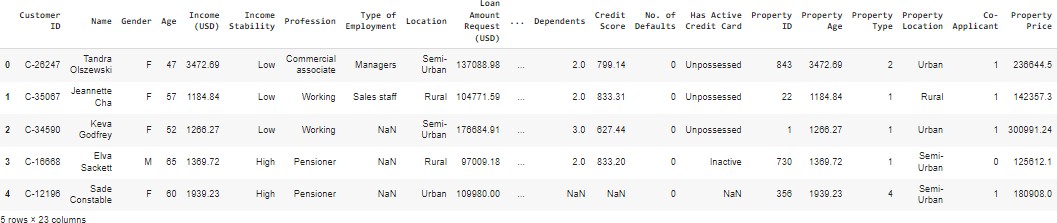
plt.xlabel("Actual Loan Amount")

plt.ylabel("Predicted Loan Amount") plt.title("Actual vs. Predicted Loan Amount")

plt.legend() plt.show()



Predicting loan amount for test dataset



#Load test dataset

test\_df = pd.read\_csv('/content/test.csv')

test\_df.head(5)

test\_df.loc[(test\_df['Co-Applicant'] == '?'),'Co-Applicant'] = 0 test\_df.loc[(test\_df['Co-Applicant'] == '0'),'Co-Applicant'] = 0

test\_df['Property Age'].fillna(train\_df['Property Age'].mean().round(2), inplace=True)

test\_df['Credit Score'].fillna(0, inplace=True)

test\_df1=test\_df[['Loan Amount Request (USD)','Current Loan Expenses (USD)','Credit Score','No. of Defaults','Co-Applicant','Property Price']] test\_pred = model.predict(test\_df1)

op = pd.DataFrame({'Customer ID' : CUST\_ID, 'Loan Sanction Amount (USD)':test\_pred

})

op.to\_csv('output.csv',index=False) op.head(10)

