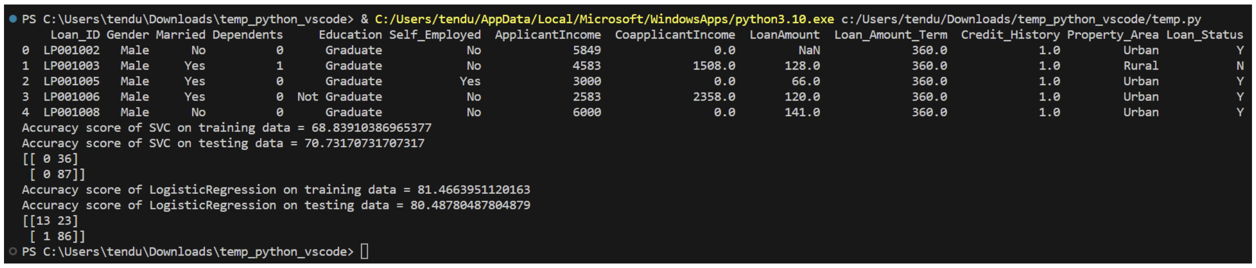
Loan Sanction

BUDIDA TENDULKAR

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn import preprocessing
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

# Load the dataset from an Excel file
data = pd.read_excel("LoanApprovalPrediction.xlsx")

# Display the first 5 rows of the dataset
print(data.head(5))

# Identify categorical columns and drop unnecessary 'Loan_ID' column
obj = (data.dtypes == 'object')
data.drop(['Loan_ID'], axis=1, inplace=True)

# Initialize a LabelEncoder for encoding categorical variables
label_encoder = preprocessing.LabelEncoder()
obj = (data.dtypes == 'object')

# Encode categorical variables using LabelEncoder
for col in list(obj[obj].index):
    data[col] = label_encoder.fit_transform(data[col].astype(str))

# Fill missing values with the mean of each column
for col in data.columns:
    data[col] = data[col].fillna(data[col].mean())
```

```python
# Split the dataset into features (X) and target variable (status)
X = data.drop(['Loan_Status'], axis=1)
status = data['Loan_Status']

# Split the data into training and testing sets
X_train, X_test, status_train, status_test = train_test_split(X, status,
test_size=0.2, random_state=10)

# Initialize Support Vector Classifier (SVC) and Logistic Regression (LR)
models
svc = SVC()
lc = LogisticRegression(max_iter=1000)

# Loop through each classifier and evaluate on training and testing data
for clf in (svc, lc):
    # Train the classifier and predict on training data
    clf.fit(X_train, status_train)
    status_pred = clf.predict(X_train)

    # Print accuracy score on training data
    print("Accuracy score of", clf.__class__.__name__, "on training data
=", 100 * metrics.accuracy_score(status_train, status_pred))

    # Predict on testing data and calculate accuracy
    status_pred = clf.predict(X_test)
    accuracy_test = 100 * metrics.accuracy_score(status_test, status_pred)

    # Print accuracy score on testing data
    print("Accuracy score of", clf.__class__.__name__, "on testing data
=", accuracy_test)

    # Calculate confusion matrix for testing data
    cm_test = confusion_matrix(status_test, status_pred)
    print(cm_test)
```

OUTPUT:



```
PS C:\Users\tendu\Downloads\temp_python_vscode> & C:/Users/tendu/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/Users/tendu/Downloads/temp_python_vscode/temp.py
    Loan_ID Gender Married Dependents     Education Self_Employed  ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  Credit_History Property_Area Loan_Status
0  LP001002   Male      No          0      Graduate            No             5849                0.0         NaN             360.0             1.0         Urban           Y
1  LP001003   Male     Yes          1      Graduate            No             4583             1508.0       128.0             360.0             1.0         Rural           N
2  LP001005   Male     Yes          0      Graduate           Yes             3000                0.0        66.0             360.0             1.0         Urban           Y
3  LP001006   Male     Yes          0  Not Graduate            No             2583             2358.0       120.0             360.0             1.0         Urban           Y
4  LP001008   Male      No          0      Graduate            No             6000                0.0       141.0             360.0             1.0         Urban           Y
Accuracy score of SVC on training data = 68.83910386965377
Accuracy score of SVC on testing data = 70.73170731707317
[[ 0 36]
 [ 0 87]]
Accuracy score of LogisticRegression on training data = 81.4663951120163
Accuracy score of LogisticRegression on testing data = 80.48780487804879
[[13 23]
 [ 1 86]]
PS C:\Users\tendu\Downloads\temp_python_vscode>
```