LOAN APPROVAL CLASSIFICATION USING SVM

A

ADM Course Project Report in

partial fulfilment of the degree

Bachelor of Technology
in
Computer Science & Engineering

By

Md. Afran	2303A51997
P.Kundan Sadhu Yaswanth	2303A51998
S. Manohar	2303A51982
P. Raj Naresh	2303A51974
N.Venkata SaiKumar	2303A51978

Under the guidance of

Bediga Sharan
Assistant Professor

Submitted to

School of Computer Science and Artificial Intelligence

SR UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the **Application of Data Mining – Course Project** Report entitled **"**Loan Approval Prediction**"** is a record of Bonafide work carried out by the student(s)Mohammed Irfan, P.Kundan Sadhu Yaswanth, S.Manohar, P.RajNaresh,N.Venkata SaiKumar bearing Hall ticket No(s) 2303A51997, 2303A51998, 2303A51974, 2303A51982, 2303A51978 during the academic year 2024-25 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University, Warangal.

Supervisor
(Mr. Bediga Sharan)
Assistant Professor

Head of the Department
(Dr. M. Sheshikala)
Professor

OBJECTIVE OF THE PROJECT

The goal of this project is to create a prediction model that correctly identifies the loan application approval status based on applicant data and credit history. Through data mining, the project hopes to help financial institutions make quicker, data-based, and accurate loan approval decisions, reducing risk and enhancing customer satisfaction.DEFINITIONS OF THE ELEMENTS USED IN THE PROJECT

TECHNOLOGIES AND LIBRARIES USED FOR LOAN APPROVAL PREDICTION:

- NumPy: A library for numerical computing in Python.
- Pandas: A library for data manipulation and analysis in Python.
- Matplotlib: A library for creating static, interactive, and animated visualizations in Python.
- Seaborn: A library for making statistical graphics in Python.
- SciPy: A library for scientific and technical computing in Python.
- Scikit-learn: A library for machine learning in Python.

DATA PREPROCESSING FOR LOAN APPROVAL PREDICTION

- Standardization: Shifting data so it has a mean of 0 and standard deviation of 1 (a type of scaling).
- train_test_split: A function in Scikit-learn that is used to split data into training and testing sets.
- Normalization:Adjusting numbers to a common scale without distorting differences.
- Encoding: Changing text data (like "Male", "Female") into numbers (like 1, 0) so that ML models can use it.

MACHINE LEARNING MODELS FOR LOAN APPROVAL PREDICTION

- Logistic Regression: Predicts probability of loan approval (binary: Yes/No); very commonly used.
- Decision Tree Classifier: Creates a flowchart-like structure to make decisions; easy to interpret.
- Random Forest Classifier: A group of decision trees combined together for better accuracy and less overfitting.
- Support Vector Machine (SVM): Finds the best boundary (hyperplane) between approved and not approved loans.
- K-Nearest Neighbors (KNN): Predicts loan approval based on the approvals of similar applicants (neighbors).
- Artificial Neural Networks (ANN): Deep learning model; useful if you have a large amount of data and want very high accuracy.

- Naive Bayes Classifier: A simple and fast model based on probability; good for quick baseline results.
- XGBoost Classifier: An advanced and faster version of gradient boosting; very high performance.

MODEL TRAINING, TESTING AND EVALUATION FOR LOAN APPROVAL:

Training = Teach the model.

Testing = See how much the model has learned.

Evaluation = Measure its LoanStatus (Accuracy, Precision, Recall, F1 Score).

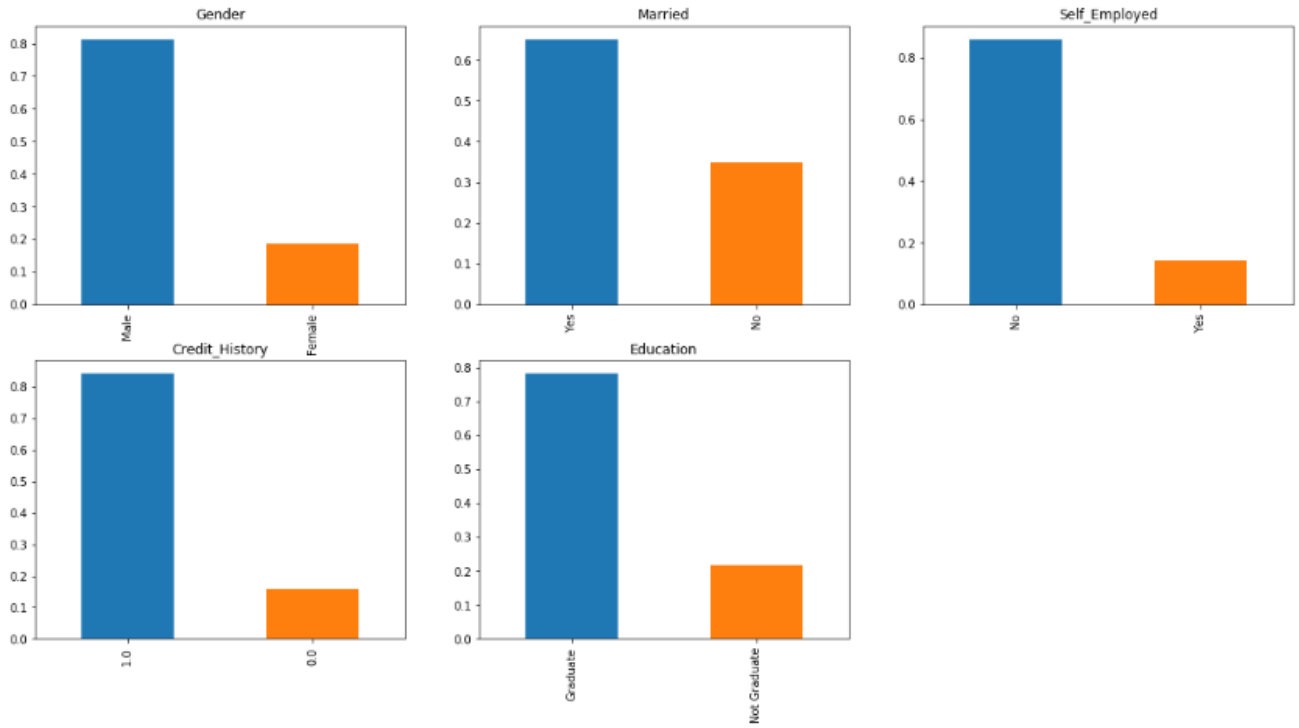
VISUALIZATION OF LOAN APPROVAL DATA

- Countplot (Loan_Status):How many loans got approval vs not approved.
- Bar Plot (Gender vs Loan_Status): Whether males or females get more approval.
- Boxplot: (Income vs Loan_Status: How income affects loan approval.
- Histogram (LoanAmount: Distribution of loan amounts.

IMPLEMENTATION OF LOAN APPROVAL PREDICTION SYSTEM

CODE:

```
# Visualizing categorical features
# plt.figure(1)
plt.subplot(231)
train['Gender'].value_counts(normalize=True).plot.bar(figsize=(20,10), title= 'Gender')

plt.subplot(232)
train['Married'].value_counts(normalize=True).plot.bar(title= 'Married')

plt.subplot(233)
train['Self_Employed'].value_counts(normalize=True).plot.bar(title= 'Self_Employed')

plt.subplot(234)
train['Credit_History'].value_counts(normalize=True).plot.bar(title= 'Credit_History')

plt.subplot(235)
train['Education'].value_counts(normalize=True).plot.bar(title= 'Education')

plt.show()
```



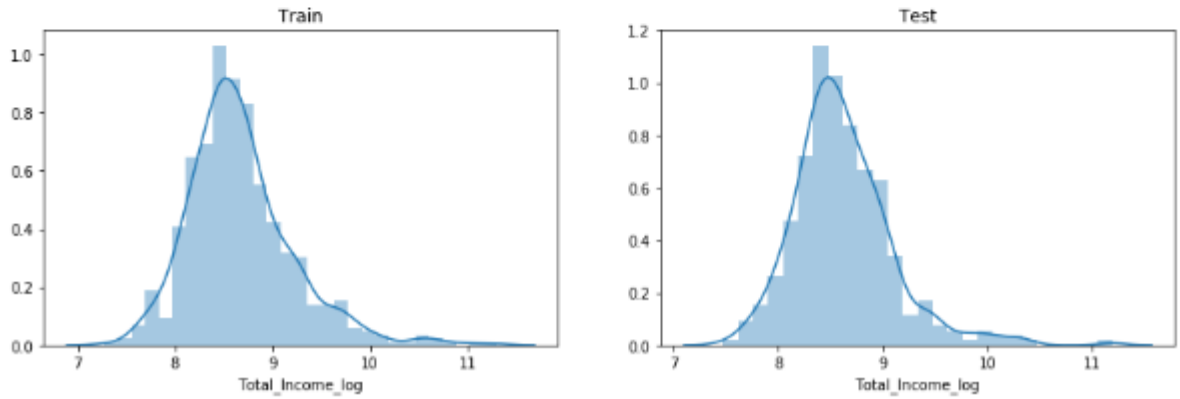

```
# calculate and visualize correlation matrix
matrix = train.corr()
f, ax = plt.subplots(figsize=(9, 6))
sns.heatmap(matrix, vmax=1, square=True, cmap="BuPu", annot=True)

matrix
```

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Loan_Status
ApplicantIncome	1.000000	-0.116605	0.570909	-0.045306	-0.014715	-0.004710
CoapplicantIncome	-0.116605	1.000000	0.188619	-0.059878	-0.002056	-0.059187
LoanAmount	0.570909	0.188619	1.000000	0.039447	-0.008433	-0.037318
Loan_Amount_Term	-0.045306	-0.059878	0.039447	1.000000	0.001470	-0.021268
Credit_History	-0.014715	-0.002056	-0.008433	0.001470	1.000000	0.561678
Loan_Status	-0.004710	-0.059187	-0.037318	-0.021268	0.561678	1.000000

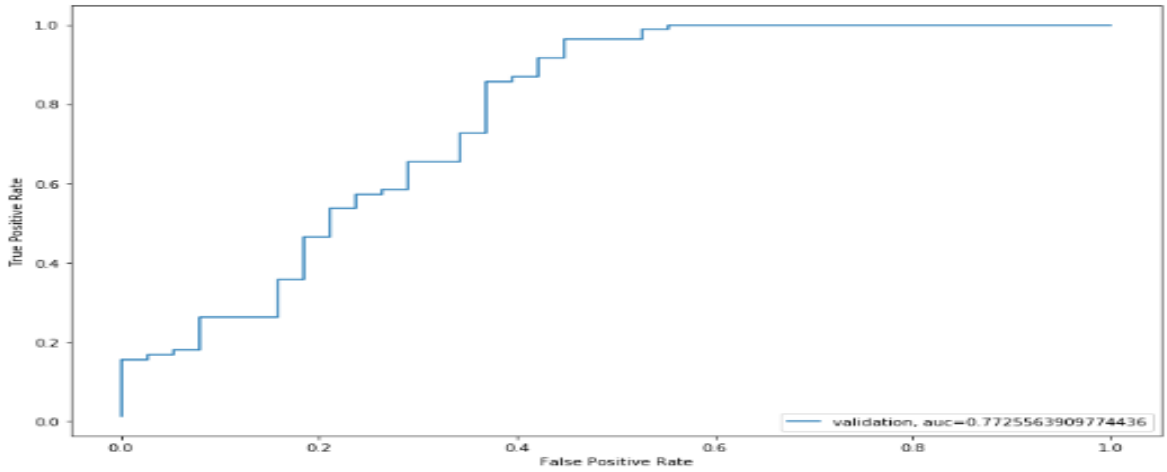
```
# log transformation
train['Total_Income_log'] = np.log(train['Total_Income'])
test['Total_Income_log'] = np.log(test['Total_Income'])
```

```
# after log transformation
fig = plt.figure(figsize=(14, 4))
ax1 = plt.subplot(121)
sns.distplot(train['Total_Income_log'])
ax1.set_title("Train")

ax1 = plt.subplot(122)
sns.distplot(test['Total_Income_log'])
ax1.set_title("Test")
```

```
Text(0.5,1,'Test')
```



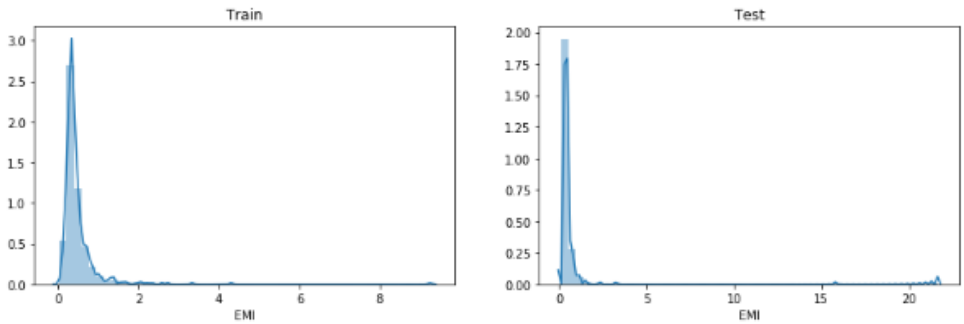
```
# visualize ROC curve
from sklearn import metrics
fpr, tpr, _ = metrics.roc_curve(yvl,  pred)
auc = metrics.roc_auc_score(yvl, pred)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="validation, auc="+str(auc))
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()
```



Let's check the distribution of EMI variable. EMI is a continuous numerical variable.

```
# check the distribution of EMI

fig = plt.figure(figsize=(14, 4))
ax1 = plt.subplot(121)
sns.distplot(train['EMI'])
ax1.set_title("Train")

ax1 = plt.subplot(122)
sns.distplot(test['EMI'])
ax1.set_title("Test")
```

```
Text(0.5,1,'Test')
```



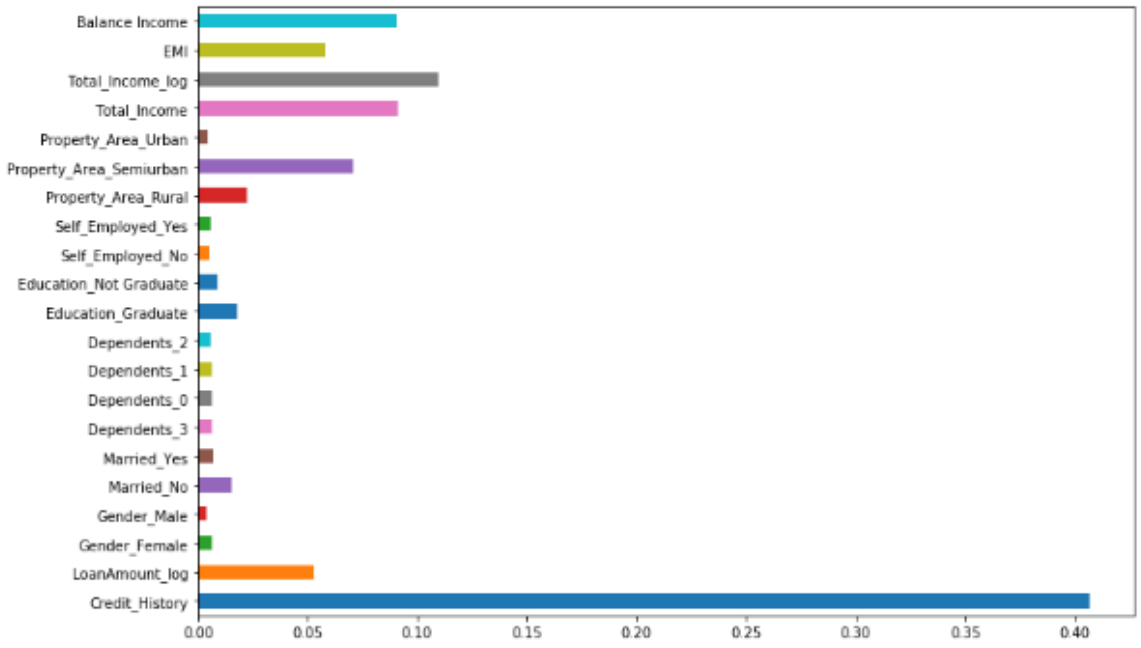
```
# import library
from xgboost import XGBClassifier
```

```
mean_accuracy = []
i=1
kf = StratifiedKFold(n_splits=5,random_state=1,shuffle=True)
for train_index,test_index in kf.split(X,y):
    print('\n{} of kfold {}'.format(i,kf.n_splits))
    xtr,xvl = X.loc[train_index],X.loc[test_index]
    ytr,yvl = y[train_index],y[test_index]

    model = XGBClassifier(random_state=1, n_estimators=50, max_depth=4)
    model.fit(xtr, ytr)
    pred_test = model.predict(xvl)
    score = accuracy_score(yvl,pred_test)
    mean_accuracy.append(score)
    print('accuracy_score',score)
    i+=1

print("\nMean validation accuracy: ", sum(mean_accuracy)/len(mean_accuracy))
pred_test = model.predict(test)
pred3=model.predict_proba(test)[:,1]

# warnings.filterwarnings(action='ignore', category=DeprecationWarning)
```

```
1 of kfold 5
accuracy_score 0.782258064516129

2 of kfold 5
accuracy_score 0.8225806451612904

3 of kfold 5
accuracy_score 0.7622950819672131

4 of kfold 5
accuracy_score 0.7459016393442623

5 of kfold 5
accuracy_score 0.7868852459016393

Mean validation accuracy:  0.7799841353781068
```

LOAN RESULT SCREEN

```
<matplotlib.axes._subplots.AxesSubplot at 0x2954721ef28>
```



CONCLUSION AND FUTURE SCOPE FOR LOAN APPROVAL PREDICTION

In this project, we created a machine learning model that predicts loan approvals from applicant information such as income, amount of the loan, and credit history. After pre-processing and cleaning the data, we experimented with models like Logistic Regression, Random Forest, and XGBoost. Of these, the best performance was from XGBoost, which gave high accuracy and balanced precision and recall. The project demonstrates that machine learning has the ability to enable banks to make quicker and more precise loan decisions, minimize human mistakes, and increase efficiency. Improvements in the future can be directed towards dealing with class imbalance and tuning the models for better performance. "I want to further develop the project by applying more advanced data mining techniques."

Github:

https://github.com/KundanSadhu/ADM-PROJECT