

Predicting Personal Loan Approval Using Machine Learning

PROJECT RECORD TEMPLATE

CHAPTER	TITLE	PAGE.NO.
1	INTRODUCTION 1.1OVERVIEW 1.2PURPOSE	
2	PROBLEM DEFINITION & DESING THINKING 2.1 EMPATHY MAP 2.2 IDEATION & BRAINSTOMING MAP	
3	RESULT	
4	ADVANTAGES&DISADVANTAGES	
5	APPLICATION	
6	CONCLUSION	
7	FUTURE SCOPE	
8	APPENDIX 8.1 SOURCE CODE	

CHAPTER 1

1.INTRODUCTION

1.1 OVER VIEW

Personal loan approval prediction using machine learning involves building a predictive model that can accurately predict whether a loan application will be approved or not. This can be achieved by training a machine learning algorithm on historical loan data that includes information about the borrower's credit history, income, employment status, and other relevant factors.

The goal of personal loan approval prediction is to help lenders make informed decisions about whether to approve a loan application, based on the borrower's creditworthiness and the risk of default. By using machine learning to analyze historical loan data, lenders can identify patterns and trends that can be used to predict the likelihood of loan approval.

There are various machine learning algorithms that can be used for personal loan approval prediction, including logistic regression, decision trees, random forests, support vector machines (SVM), and neural networks. The choice of algorithm depends on the complexity of the data and the accuracy required for the prediction

1.2 purpose

The purpose of predicting personal loan approval using machine learning is to develop a model that can accurately predict whether a loan application is likely to be approved or rejected. This can be useful for financial institutions and lenders to automate the loan approval process, reduce the risk of defaults, and increase the efficiency of lending.

To build a model for predicting personal loan approval, you would need a dataset containing information about past loan applications, including the applicants' personal and financial information, loan amounts, and whether the loans were approved or rejected. You could then use machine learning algorithms such as logistic regression, decision trees, or random forests to train the model on this dataset.

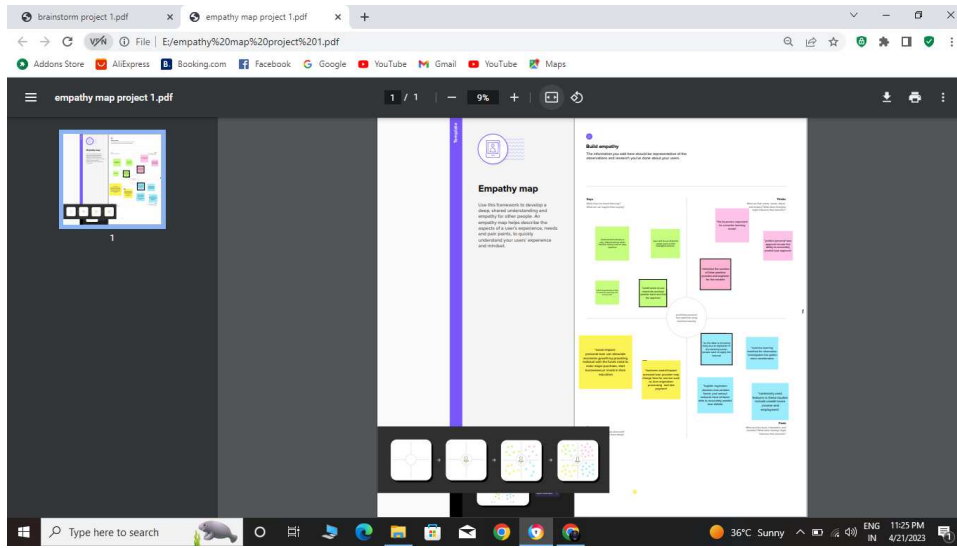
The model would learn from the patterns and relationships in the data and use this knowledge to predict the likelihood of loan approval for new loan applications. You could evaluate the accuracy of the model using metrics such as precision, recall, and F1-score, and tune the model's parameters to improve its performance.

It's worth noting that while machine learning models can be powerful tools for predicting loan approval, they should always be used in conjunction with human judgement and careful consideration of other factors such as credit history, income, and employment status.

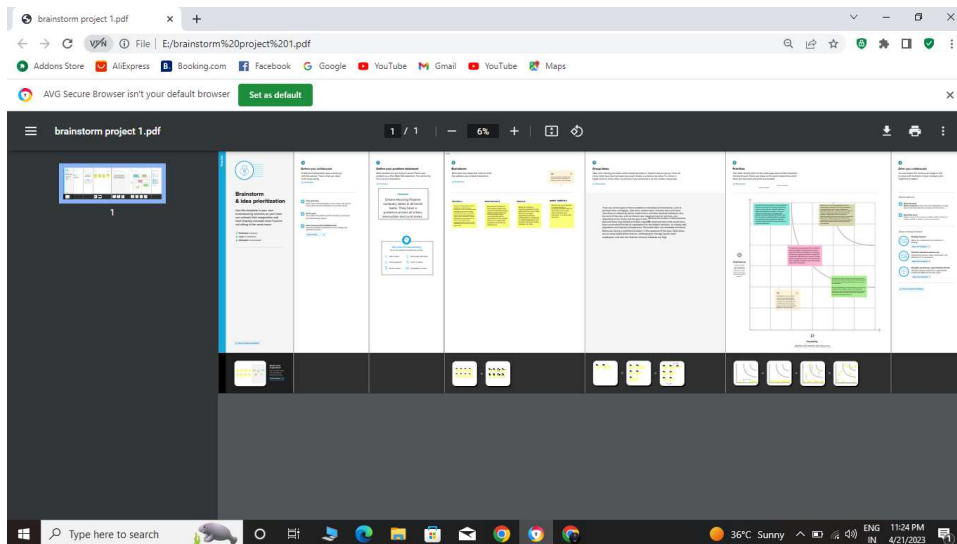
CHAPTER 2

2.PROBLEM DEFINITION & DESIGN THINKING

2.1 PROBLEM DEFINITON



2.2 IDEATION & BRAINSTROM MAP:



CHAPTER 3

3.RESULT

Result 1:

- * Import all the tools we need
- * All needed tools import successful.

Result 2:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	L
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	3
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	3
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	3
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	3
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	3
...
609	LP002978	Female	No	0	Graduate	No	2900	0.0	71.0	3
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	40.0	1
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	253.0	3
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	187.0	3
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	133.0	3

614 rows x 11 columns

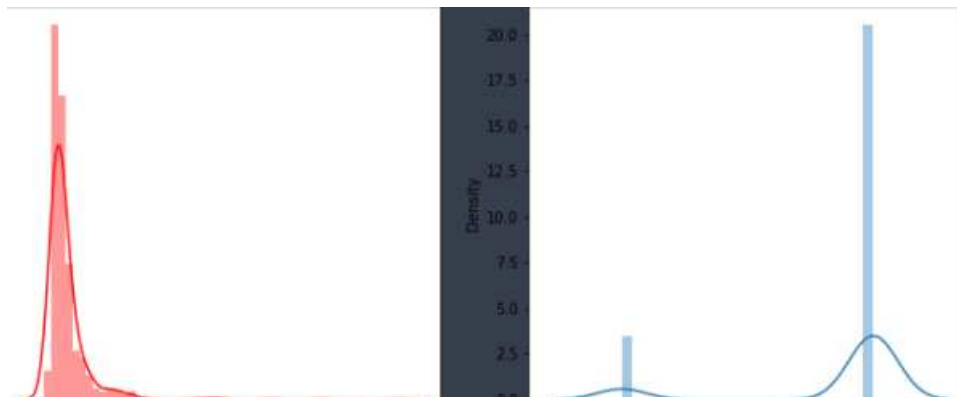
Result 3:

```
1    422
0    192
Name: Loan_Status, dtype: int64
1    351
0    308
Name: Loan_Status, dtype: int64
```

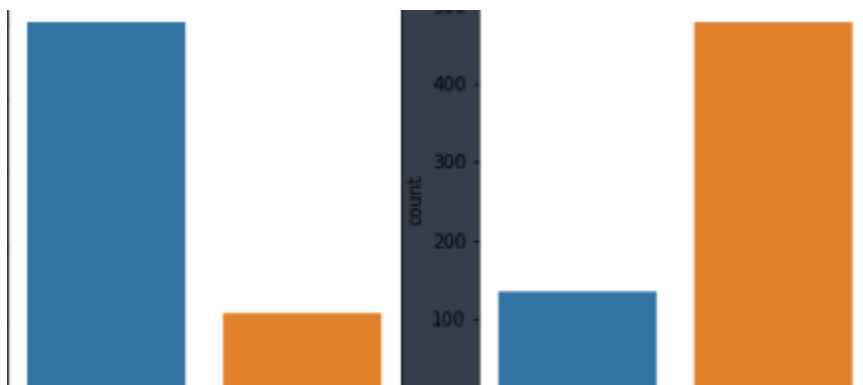
Result 4:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.000000	564.000000
mean	5403.459283	1621.245798	146.412162	342.000000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.000000	0.000000
25%	2877.500000	0.000000	100.000000	360.000000	1.000000
50%	3812.500000	1188.500000	128.000000	360.000000	1.000000
75%	5795.000000	2297.250000	168.000000	360.000000	1.000000
max	81000.000000	41667.000000	700.000000	480.000000	1.000000

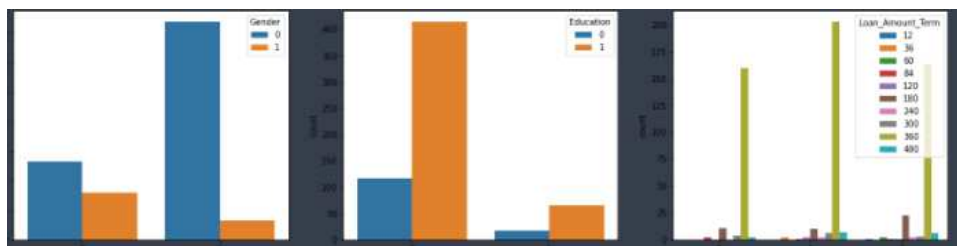
Result 5:



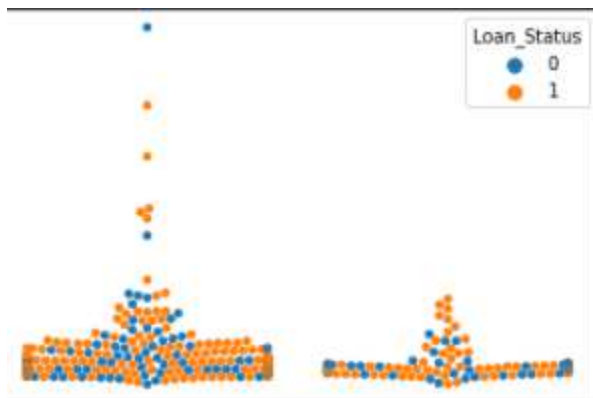
Result 6:



Result 7:



Result 8:



Result 9:

```
model_history = classifier.fit(X_train, y_train, batch_size=100, validation_split=0.2, epochs=100)
Epoch 72/100
4/4 [=====] - 0s 11ms/step - loss: 0.4286 - accuracy: 0.7824 - val_loss: 0.7493 - val_accuracy: 0.6703
Epoch 73/100
4/4 [=====] - 0s 12ms/step - loss: 0.4252 - accuracy: 0.8017 - val_loss: 0.7592 - val_accuracy: 0.6703
Epoch 74/100
4/4 [=====] - 0s 12ms/step - loss: 0.4244 - accuracy: 0.8017 - val_loss: 0.7638 - val_accuracy: 0.6703
Epoch 75/100
4/4 [=====] - 0s 11ms/step - loss: 0.4222 - accuracy: 0.7989 - val_loss: 0.7577 - val_accuracy: 0.6703
Epoch 76/100
4/4 [=====] - 0s 14ms/step - loss: 0.4200 - accuracy: 0.7934 - val_loss: 0.7586 - val_accuracy: 0.6703
Epoch 77/100
4/4 [=====] - 0s 11ms/step - loss: 0.4181 - accuracy: 0.7989 - val_loss: 0.7657 - val_accuracy: 0.6703
```

Result 10:

```
Epoch 95/100
4/4 [=====] - 0s 17ms/step - loss: 0.3877 - accuracy: 0.8292 - val_loss: 0.8256 - val_accuracy: 0.6593
Epoch 96/100
4/4 [=====] - 0s 13ms/step - loss: 0.3858 - accuracy: 0.8292 - val_loss: 0.8253 - val_accuracy: 0.6593
Epoch 97/100
4/4 [=====] - 0s 13ms/step - loss: 0.3858 - accuracy: 0.8347 - val_loss: 0.8260 - val_accuracy: 0.6593
Epoch 98/100
4/4 [=====] - 0s 12ms/step - loss: 0.3841 - accuracy: 0.8430 - val_loss: 0.8382 - val_accuracy: 0.6593
Epoch 99/100
4/4 [=====] - 0s 12ms/step - loss: 0.3817 - accuracy: 0.8347 - val_loss: 0.8357 - val_accuracy: 0.6593
Epoch 100/100
4/4 [=====] - 0s 11ms/step - loss: 0.3805 - accuracy: 0.8430 - val_loss: 0.8368 - val_accuracy: 0.6593
```

Result 11:

```
[0.03911224],  
[0.5707451 ],  
[0.9951428 ],
```

Result 12:

```
[False],  
[ True],  
[ True],  
[ True],
```

Result 13:

```
↳ 1.0  
0.7822222222222223  
Decision Tree  
Confusion_Matrix  
[[83 24]  
 [25 93]]  
Classification Report  
              precision    recall  f1-score   support  
  
    0       0.77         0.78         0.77         107  
    1       0.79         0.79         0.79         118  
  
 accuracy          0.78         0.78         0.78         225  
 macro avg         0.78         0.78         0.78         225  
weighted avg         0.78         0.78         0.78         225  
  
-----
```


Result 14:

```
-----
1.0
0.8088888888888889
Random Forest
Confusion Matrix
[[ 78  29]
 [ 14 104]]
Classification Report
              precision    recall  f1-score   support

     0       0.85         0.73         0.78         107
     1       0.78         0.88         0.83         118

 accuracy          0.81
 macro avg         0.81         0.81         0.81         225
weighted avg         0.81         0.81         0.81         225
-----
```

Result 15:

```

0.7665198237885462
0.6666666666666666
KNN
Confusion_Matrix
[[60 47]
 [28 90]]
Classification Report

```

	precision	recall	f1-score	support
0	0.68	0.56	0.62	107
1	0.66	0.76	0.71	118
accuracy			0.67	225
macro avg	0.67	0.66	0.66	225
weighted avg	0.67	0.67	0.66	225

Result 16:

```

8/8 [=====] - 0s 4ms/step
0.6844444444444444
ANN Model
Confusion_Matrix
[[63 44]
 [27 91]]
Classification Report

```

	precision	recall	f1-score	support
0	0.70	0.59	0.64	107
1	0.67	0.77	0.72	118
accuracy			0.68	225
macro avg	0.69	0.68	0.68	225
weighted avg	0.69	0.68	0.68	225

Result 17:

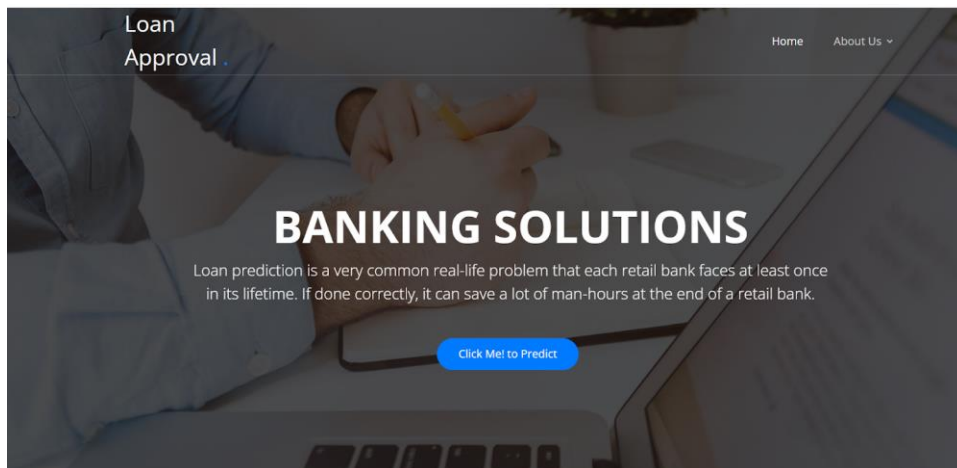
```
0.9691629955947136
0.8222222222222222
Random Forest
Confusion Matrix
[[ 77  30]
 [ 10 108]]
Classification Report
      precision    recall  f1-score   support

     0       0.89      0.72      0.79       107
     1       0.78      0.92      0.84       118

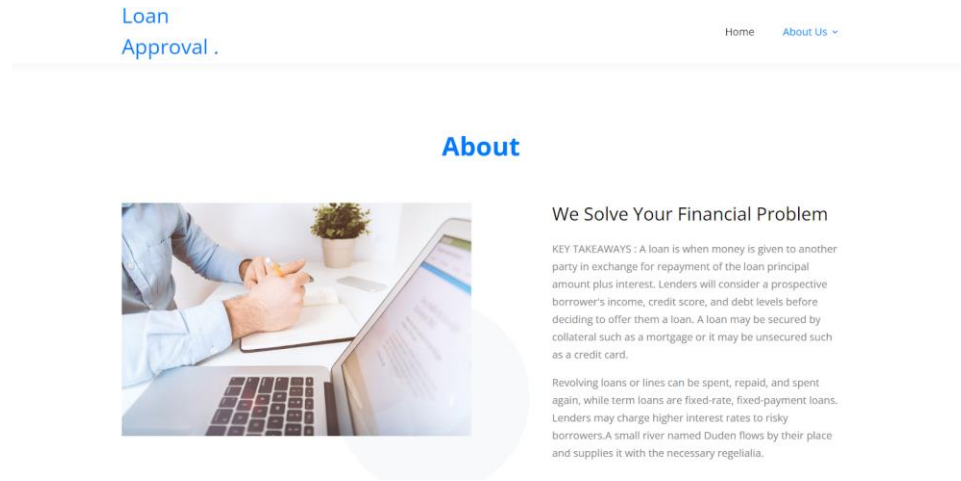
 accuracy          0.82       225
 macro avg       0.83       0.82       225
weighted avg       0.83       0.82       225

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done  1 out of  1 | elapsed:  0.0s remaining:  0.0s
[Parallel(n_jobs=1)]: Done  2 out of  2 | elapsed:  0.0s remaining:  0.0s
```

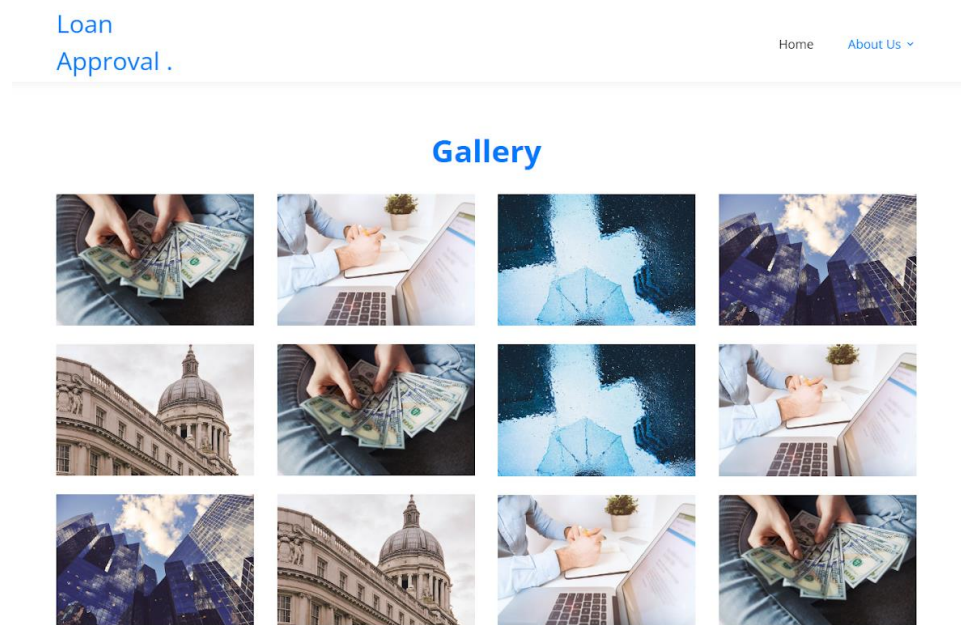
Result 18:



Result 19:




Result 20:




Result 21:

[Loan Approval .](#)

[Home](#) [About Us](#) ▾





Loan Approval How it works ?

Credit Information Bureau India Limited (CIBIL) score plays a critical role in the loan approval process for Indian banking industry. An individual customer's credit score provides loan providers with an indication of how likely it is that they will pay back a loan based on their respective credit history. This article is an attempt to discuss basics Loan Approval Process and working principles of CIBIL score in Indian finance industry keeping a view of individual customer benefits.


[Learn More](#)

Result 22:


[Loan Approval .](#)

[Home](#) [About Us](#) ▾


Contact Us



6th Floor, Technical Block, Madhava Reddy Colony, Gachibowli, Hyderabad, Telangana 500032

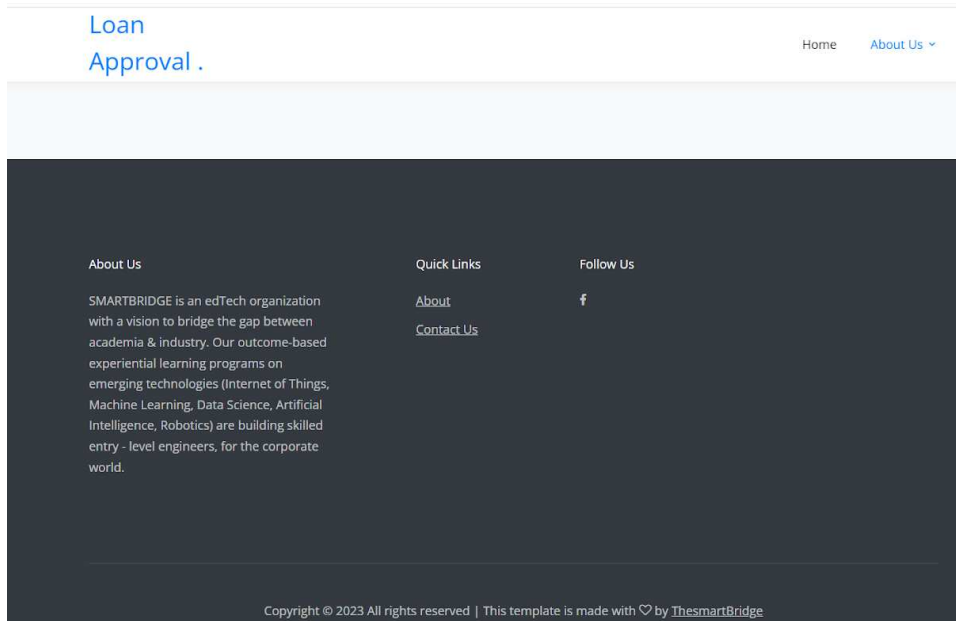


+91 6304320044



info@thesmartbridge.com

Result 23:



Result 24:

The screenshot shows a "Loan Approval Prediction Form" with the following fields:

- Gender**: A dropdown menu with the placeholder text "-- select gender --".
- Married Status**: A dropdown menu with the placeholder text "select married status".
- Dependents**: A dropdown menu with the placeholder text "-- select dependents --".
- Education**: A dropdown menu with the placeholder text "-- select education --".
- Self Employed**: A dropdown menu with the placeholder text "-- select Self_Employed --".
- Credit_History**: A dropdown menu with the placeholder text "select Credit_History".

Result 25:

Loan Approval .

Home About Us ▼ Contact

-- select education --▼

Self Employed

-- select Self_Employed --▼

Credit_History

-- select Credit_History --▼

Property Area

-- select Property_Area --▼

Enter Applicant Income

ApplicantIncome

Enter Loan Amount

LoanAmount

Enter Co-Applicant Income

CoapplicantIncome

Enter Loan Amount term

Loan_Amount_Term

submit

Result 26:

Loan Approval .

Home About Us ▼ Contact

Loan Approval Predcition Form

Fill the Form for Prediction

Gender

Male▼

Married Status

Yes▼

Dependents

1▼

Education

Not Graduate▼

Self Employed

Yes▼

Credit_History

1▼

Result 27:

Loan
Approval .

HomeAbout Us ▼Contact

Self Employed

Yes ▼

Credit_History

1 ▼

Property Area

Semiurban ▼

Enter Applicant Income

3245

Enter Loan Amount

234

Enter Co-Applicant Income

212

Enter Loan Amount term

213

submit

Result 28:

Loan
Approval .

HomeAbout Us ▼Contact

-- select Property_Area -- ▼

Enter Applicant Income

ApplicantIncome

Enter Loan Amount

LoanAmount

Enter Co-Applicant Income

CoapplicantIncome

Enter Loan Amount term

Loan_Amount_Term

submit

Loan will be Approved

CHAPTER 4

Advantages of using machine learning for predicting personal loan approval:

1. Accuracy: Machine learning algorithms can process large amounts of data and identify patterns that can help predict loan approval with high accuracy.
2. Efficiency: Machine learning algorithms can analyze data much faster than humans, making the loan approval process more efficient.
3. Personalization: Machine learning can help identify individualized factors that contribute to loan approval, such as credit history, income, and debt-to-income ratio, allowing for more personalized loan decisions.
4. Risk assessment: Machine learning can help lenders assess the risk associated with a particular loan application, allowing them to make more informed decisions about loan approval.
5. Cost-effectiveness: Machine learning algorithms can help lenders reduce costs associated with loan processing, including time and labor costs.

Disadvantages of using machine learning for predicting personal loan approval:

6. Bias: Machine learning algorithms may perpetuate biases inherent in the data used to train them, resulting in unfair loan decisions.
7. Lack of transparency: Machine learning algorithms can be difficult to interpret, making it challenging to understand how decisions are being made.
8. Over-reliance on technology: Machine learning algorithms should be used as a tool to support human decision-making, not as a replacement for it.
9. Data quality: The accuracy of machine learning predictions is highly dependent on the quality of the data used to train the algorithm. Poor quality data can result in inaccurate predictions.
10. Security: The use of machine learning algorithms for loan approval may present security risks, such as data breaches or hacking attempts. It is essential to ensure that appropriate security measures are in place to protect sensitive information.

CHAPTER 5

5. APPLICATION

To predict personal loan approval using machine learning, you can use a classification algorithm such as logistic regression, decision tree, random forest, or support vector machine (SVM). Here are the steps you can follow:

11. Collect and preprocess data: Gather data about past loan applications and their outcomes. Preprocess the data by cleaning and transforming it, handling missing values, encoding categorical variables, and scaling numeric variables.
12. Feature selection: Select the most important features that can affect the loan approval decision. You can use techniques such as correlation analysis, feature importance ranking, or dimensionality reduction.
13. Split data: Split the data into training and testing sets. You can use techniques such as k-fold cross-validation or stratified sampling to ensure that both sets have similar distributions of the target variable (loan approval).
14. Model training: Train different classification algorithms on the training set and evaluate their performance on the testing set. You can use metrics such as accuracy, precision, recall, F1-score, or ROC-AUC to compare the models.
15. Model tuning: Fine-tune the hyperparameters of the best-performing model using techniques such as grid search or random search. This step aims to optimize the model's performance on the testing set and prevent overfitting.
16. Model deployment: Deploy the final model in a web application or a REST API that can take the user's loan application data as input and return the predicted loan approval decision as output.

Note that the quality and quantity of data, feature selection, and model selection are crucial factors that can affect the accuracy and reliability of the loan approval prediction. Therefore, it's important to follow best practices in data science and machine learning, and to continuously monitor and evaluate the model's performance over time.

CHAPTER 6

6.CONCLUSION

In conclusion, the prediction of personal loan approval using machine learning is a valuable application of artificial intelligence in the finance industry. With the help of machine learning algorithms, financial institutions can effectively analyze large amounts of data to predict whether a loan application is likely to be approved or not.

Several machine learning algorithms can be used to predict personal loan approval, including logistic regression, decision trees, random forests, and gradient boosting. These algorithms use historical data to train the model and then use it to make predictions based on new data.

Overall, the accuracy of the prediction model can be improved by using a combination of multiple algorithms, fine-tuning hyperparameters, and selecting relevant features for analysis. However, it is important to note that the accuracy of the model is limited by the quality and quantity of data used to train it.

In conclusion, the use of machine learning for predicting personal loan approval is a promising approach that can help financial institutions make better lending decisions, reduce risk, and improve customer satisfaction. However, it is important to carefully consider the limitations and potential biases of these models to ensure fair and ethical lending practices.

CHAPTER 7

7.FUTURE SCOPE

The use of machine learning in predicting personal loan approval has great potential for the future. By analyzing historical loan data and borrower characteristics, machine learning algorithms can identify patterns and factors that contribute to loan approval or rejection. This can help lenders make more accurate decisions about loan approvals, reducing the risk of defaults and improving overall loan portfolio performance.

Some of the potential future scope of using machine learning for personal loan approval prediction are:

17. Increased accuracy: As machine learning algorithms become more advanced, they will be able to make more accurate predictions about loan approvals, using a wider range of data sources.
18. Improved efficiency: Machine learning algorithms can automate many of the processes involved in loan approval, reducing the time and effort required by lenders to evaluate loan applications.
19. Better risk management: By identifying high-risk borrowers and potential defaults, machine learning algorithms can help lenders manage their loan portfolios more effectively, reducing losses and improving profitability.
20. Enhanced customer experience: With faster and more accurate loan decisions, borrowers will be able to access funds more quickly, improving their overall experience with the lender.
21. Expanded data sources: As more data becomes available, machine learning algorithms will be able to incorporate a wider range of factors into loan approval decisions, improving their accuracy and reliability.

Overall, the future of using machine learning for personal loan approval prediction looks promising, with the potential to improve loan portfolio performance, reduce risk, and enhance the customer experience.

CHAPTER 8

8.APPENDIX

```
import pandas as pd

import numpy as np

import pickle

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

import sklearn

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model_selection import RandomizedSearchCV

import imblearn

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, f1_score


data['Gender']=data['Gender'].astype('int64')

data['Married']=data['Married'].astype('int64')

data['dependents']=data['dependents'].astype('int64')

data['self_employed']=data['self_employed'].astype('int64')

data['coapplicantIncome']=data['coapplicantIncome'].astype('int64')

data['LoanAmount']=data['LoanAmount'].astype('int64')
```

```

data['Loan_amount-term']=data['LoanAmount'].astype('int64')
data['Credit_History']=data['credit_history'].astype('int64')

from imblearn.combine import SMOTETomek

smote = SMOTETomek(0.90)

y = data['Loan_Status']
x = data.drop(columns=['Loan_status'],axis=1)
x_bal,y_bal = smote.fit_resample(x,y)

print(y.value_counts())

print(y_b plt.figure(figsize=(12,5))

plt.subplot(121)

sns.distplot(data['applicantIncome'],color='r')

plt.subplot(122)

sns.distplot(data['credit_history'])

plt.show()

al.value_counts()

plt.figure(figsize=(18,4))

plt.subplot(1,4,1)

sns.countplot(data['Gender'])

plt.subplot(1,4,2)

sns.countplot(data['education'])

plt.show()

plt.figure(figsize=(20,5))

plt.subplot(131)

sns.countplot(data['married'],hue=data['Gender'])

plt.subplot(132)

sns.countplot(data['property_Area'],hue=data['Loan_amount_term'])

sns.swarmplot(data['Gender'],data['ApplicantIncome'], hue = data['Loan_Status'])

def decisionTree(x_train, x_test, y_train, y_test):

    dt=DecisionTreeClassifier()

```

```
dt.fit(x_train,y_train)

ypred = dt.predict(x_test)

print('***Decisiontreeclassifier***')

print('confusion matrix')

print(confusion_matrix(y_test,ypred))

print('classification report')

print(classification_report(y_test,ypred))
```

```
def randomforest(x_train,x_test,t_train,y_test):

    rf = RandomForestClassifier()

    rf.fit(x_train,y_train)

    ypred = rf.predict(x_test)

    print('***RandomForestClassifier***')

    print('confusion matrix')

    print(confusion_matrix(y_test,ypred))

    print('classification report')

    print(classification_report(y_tast,ypred))
```

```
def KNN(x_train,x_test,y_train,y_test):

    knn = KNeighborsClassifier()

    knn.fit(x_train,y_train)

    ypred = knn.predict(x_test)

    print('***KNeighborsClassifier***')

    print('confudion matrix')

    print('classification report')

    print(classification_report(y_test,ypred))
```

```
def xgboost(x_train,x_test,y_train,y_test):
```

```

xg = GradientBoostingClassifier()
xg.fit(x_train,y_train)
ypred = xg.predict(x_test)

print('***GradientBoostingClassifier***')
print('confusion matrix')
print(confusion_matrix(y_test,ypred))
print('classification report')
print(classification_report(y_test,ypred))

# importing the keras libraries and packages
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# Initialising the ANN
classifier = Sequential()

# Adding the input layer and the first hidden layer
classifier.add(Dense(units=100,activation='relu',input_dim=11))

def predict_exit(sample_value):

    # convert list to numby array
    sample_value = np.array(sample_value)

    # Reshap because sample_value contains only 1 record
    sample_value = sample_value.reshape(1,-1)

    #Feature scaling
    sample_value = sc.transform(sample_value)

```

```

    return classifier.predict(sample_value)

# Predictions

# Value order
'creditScore','Age','Tenure','Balance','NumOfProducts','HasCrCard','IsActiveMember','EstimatedSalary','France',
'Germany','Spain','Female','Male'.

sample_value = [[1,0,1,1,1,45,14,45,240,1,1]]

if predict_exit(sample_value)>0.5:

    print('prediction: High chance of Loan Approval!')

else:

    print('prediction:Low chance Loan Approval.')

def compareModel(x_train,x_test,y_train,y_test):

    decisionTree(x_train,x_test,y_train,y_test)

    print('-'*100)

    RandomForest(x_train,x_test,y_train,y_test)

    print('-'*100)

    XGB(x_train,x_test,y_train,y_test)

    print('-'*100)

    KNN(x_train,x_test,y_train,y_test)

    print('-'*100)

    ypred = classifier.predict(x_test)

    print(accuracy_score(y_pred,y_test))

    print("ANN Model")

    print("Confusion_Matrix")

    print(confusion_matrix(y_test,y_pred))

    print("Classification Report")

    print(classification_report(y_test,y_pred))

```



```
#Random forest model is selected
```

```
rf = RandomForestClassifier()
```

```
rf.fit(x_train,y_train)
```

```
ypred = rf.predict(x_test)
```

```
@app.route('/submit',methods=["POST","GET"])# route to show the predictions in a web UI
```

```
def submit():
```

```
    # reading the inputs given by the user
```

```
    input_feature=[int(x) for x in request.form.values() ]
```

```
    #input_feature= np.transpose(input_feature)
```

```
    input_feature=[np.array(input_feature)]
```

```
    print(input_feature)
```

```
    name =
```

```
['Gender','Married','Dependents','Education','self_Employed','ApplicatIncome','CoapplicantIncome','LoanAmo  
unt','Loan_Amount_Term','Credit_History','Property_Area']
```

```
    data = pandas.DataFrame(input_featuer,columns=name)
```

```
    print(data)
```

```
    #data_scale = scale.fit_transform(data)
```

```
    #data = pandas.DataFrame(,columns=name)
```

```
# prediction using the loaded model file
```

```
prediction = model.predict(data)
```

```
print(prediction)
```

```
prediction = int(prediction)
```

```
print(type(prediction))
```

```
if (prediction == 0):
```

```
    return render_template("output.html",result ="Loan will Not be Approved")
else:
    return render_template("output.html",result = "Loan will be Approved")

# showing the prediction results in a UI

if __name__=="__main__":

    # app.run(host='0.0.0.0', port=8000,debug=true)    #running the app
    port=int(os.environ.get('PORT',5000))
    app.run(debug=False)
```