Model.py

```
import numpy as np

import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split


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

from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import cross_val_score


import pickle


import warnings

warnings.filterwarnings('ignore')


df=pd.read_csv("loan_prediction.csv")


numerical_features = df.select_dtypes(include = [np.number]).columns

categorical_features = df.select_dtypes(include = [np.object]).columns


df['Gender'] = df['Gender'].fillna(df['Gender'].mode()[0])

df['Married'] = df['Married'].fillna(df['Married'].mode()[0])

df['Dependents'] = df['Dependents'].str.replace('+','')
```

```
df['Dependents'] = df['Dependents'].fillna(df['Dependents'].mode()[0])

df['LoanAmount'] = df['LoanAmount'].fillna(df['LoanAmount'].mode()[0])

df['Self_Employed'] = df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])

df['Loan_Amount_Term'] = df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0])

df['Credit_History'] = df['Credit_History'].fillna(df['Credit_History'].mode()[0])


df['Gender'].replace({'Male':1,'Female':0},inplace=True)

df['Dependents'].replace({'0':0,'1':1,'2':2,'3':3},inplace=True)

df['Married'].replace({'Yes':1,'No':0},inplace=True)

df['Self_Employed'].replace({'Yes':1,'No':0},inplace=True)

df['Property_Area'].replace({'Urban':2,'Rural':0,'Semiurban':1},inplace=True)

df['Education'].replace({'Graduate':1,'Not Graduate':0},inplace=True)

df['Loan_Status'].replace({'Y':1,'N':0},inplace=True)


df['CoapplicantIncome']=df['CoapplicantIncome'].astype("int64")

df['LoanAmount']=df['LoanAmount'].astype("int64")

df['Loan_Amount_Term']=df['Loan_Amount_Term'].astype("int64")

df['Credit_History']=df['Credit_History'].astype("int64")


le = LabelEncoder()

df['Loan_ID'] = le.fit_transform(df.Loan_ID)


from imblearn.over_sampling import SMOTE

from imblearn.combine import SMOTETomek
```

```
smote = SMOTETomek(0.90)


y = df['Loan_Status']

x = df.drop(columns=['Loan_Status', 'Loan_ID'],axis=1)


x_bal,y_bal = smote.fit_resample(x,y)


scaler = StandardScaler()

x_bal = scaler.fit_transform(x_bal)

x_bal = pd.DataFrame(x_bal)

print(x.head())


x_train, x_test, y_train, y_test = train_test_split(x_bal, y_bal, test_size = 0.33, random_state = 42)


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_test, yPred))

y_pred=rf.predict(x_test)

y_pred1=rf.predict(x_train)

random_forest_test_accuracy = accuracy_score(y_test,y_pred)
```

```
random_forest_train_accuracy = accuracy_score(y_train,y_pred1)

print('Testing accuracy: ', random_forest_test_accuracy)

print('Training accuracy: ',random_forest_train_accuracy)


f1_score(yPred,y_test, average='weighted')

cv = cross_val_score(rf,x,y,cv=5)

print("cross_val_score:  ", np.mean(cv))


pickle.dump(rf, open('model.pkl','wb'))
```