TITANIC CLASSIFICATION

Building a predictive model to determine the likelihood of survival for passengers on the Titanic involves several steps, including data preprocessing, feature engineering, model selection, training, and evaluation. Here's a basic outline of how you can accomplish this using Python:

**1.Import Libraries**:

Start by importing necessary libraries such as pandas, numpy, scikit-learn, matplotlib, and seaborn.

**2.Load the Dataset**:

Load the Titanic dataset into a pandas DataFrame.

**3.Exploratory Data Analysis (EDA)**:

Explore the dataset to understand its structure, check for missing values, and gain insights into the features.

**4.Data Preprocessing**: Handle missing values, encode categorical variables, and perform feature scaling if necessary.

**5. Feature Engineering**:

Create new features or modify existing ones that might improve the predictive power of the model.

**6.Split Data**:

Split the dataset into training and testing sets.

**7.Model Selection**:

Choose appropriate machine learning algorithms for classification. Common choices include logistic regression, decision trees, random forests, and gradient boosting algorithms.

**8.Model Training**:

Train the selected models on the training data.

**9.Model Evaluation**:

Evaluate the performance of the trained models using appropriate metrics such as accuracy, precision, recall, and F1-score. Also, visualize the results using confusion matrices or ROC curves.

**10.Hyperparameter Tuning**:

Fine-tune the hyperparameters of the chosen model to improve its performance.

**11**.**Final Model Selection**:

Select the best-performing model based on evaluation metrics.

**12**.**Prediction**:

Use the final model to make predictions on new data.

Here's a simplified example code to get you started:

import pandas as pd import numpy as np import seaborn as sns import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split from sklearn.preprocessing import standardscaler from sklearn.ensemble import RandomForestClassifier from sklearn.metrics import accuracy\_score,confusion\_matrix,classification\_report titanic\_df=pd.read\_cav(‘titanic.cav’) print(titanic\_df.head()) print(titanic\_df.info()) print(titanic\_df.describe()) titanic\_df.drop([‘passengerID’,’Name’,’Ticket’,’Cabin’],axis=1,inplace=true) titanic\_df[‘Age’].fillna(titanic\_df[‘Age’].median(),inplace=true) titanic\_df[‘Embarked’].fillna(titanic\_df[‘embarkrd’].mode()[0],inplace=true) titanic\_df=pd.get\_dummies(titanic\_df,colums=[‘sex’,’Embarked’],drop\_first=true) x=titanic\_df.drop(‘survived’,axis=1) y=titanic\_def[‘survived’]

x\_train,x\_test,y\_test,y\_train=train\_test\_split(x,y,test\_size=0.2,random\_state) model=RandomForestClassifier(n\_estimators=100,random\_state=42) model.fit(x\_train,y\_train) y\_pred=model.predict(x\_test)

print(“Accuracy:”,accuracy\_score(y\_test,y\_prede)) print(“confusion matrix:\n”,confusion\_matrix(y\_test,y\_pred)) print(“classification Report\n:”,classification\_report(y\_test,y\_pred))