Packages to import

|  |  |  |  |
| --- | --- | --- | --- |
| In | [ | ]: | import numpy as np |
|  |  |  | import pandas as pd  import matplotlib.pyplot as plt import seaborn as sns  from collections import defaultdict  %matplotlib inline |
|  |  |  |  |
| In | [ | ]: | Train=pd.read\_csv(“Loan Prediction.csv”) |
|  |  |  |  |
| In | [ | ]: | Train.head() |
|  |  |  |  |
| In | [ | ]: |  |
|  |  |  |  |
| In | [ | ]: | Train.head() |
|  |  |  |  |
| In | [ | ]: | Train.info() |
|  |  |  |  |
| In | [ | ]: | Train.columns |
|  |  |  |  |
| In | [ | ]: | Train.shape |
|  |  |  |  |
| In | [ | ]: | Train[“Loan\_Status“].value\_counts() |
|  |  |  |  |
| In | [ | ]: | Train[“Loan\_Status”].value\_counts(normalize=True) |
|  |  |  |  |
| In | [ | ]: | Train[“Loan\_Status“].value\_counts().plot.bar() |
|  |  |  |  |
| In | [ | ]: | Train.isnull().sum() |
| In | [ | ]: | Train['Gender'].fillna(Train['Gender'].mode()[0], inplace=True) |
|  |  |  | Train['Married'].fillna(Train['Married'].mode()[0], inplace=True)  Train['Dependents'].fillna(Train['Dependents'].mode()[0], inplace=True) |
|  |  |  | Train['Self\_Employed'].fillna(Train['Self\_Employed'].mode()[0], inplace=True)  Train['Credit\_History'].fillna(Train['Credit\_History'].mode()[0], inplace=Tru e) |
|  |  |  |  |
| In | [ | ]: | Train['LoanAmount'].fillna(Train['LoanAmount'].median(), inplace=True) |
|  |  |  |  |
| In | [ | ]: | Train.isnull().sum() |
|  |  |  |  |
| In | [ | ]: | Train['Loan\_Amount\_Term'].fillna(Train['Loan\_Amount\_Term'].mode()[0], inplace= |
|  |  |  | **True)** |

|  |  |  |  |
| --- | --- | --- | --- |
| In | [ | ] : | Train.isnull().sum() |
|  |  |  |  |
| In | [ | ]: | Train['LoanAmount\_log'] = np.log(Train['LoanAmount']) Train['LoanAmount\_log'].hist(bins=20) |
|  |  |  | Train['Loan\_Status'] = Train['Loan\_Status'] |
|  |  |  |  |
| In | [ | ]: | Train.head() |
|  |  |  |  |
| In | [ | ]: | from sklearn.preprocessing import LabelEncoder label\_encoder = LabelEncoder() |
|  |  |  | Train['Gender'] label\_encoder.fit\_transform(Train['Gender'])  Train['Married'] label\_encoder.fit\_transform(Train['Married']) Train['Education'] label\_encoder.fit\_transform(Train['Education']) |
|  | | | Train['Self\_Employed'] label\_encoder.fit\_transform(Train['Self\_Employed'])  Train['Property\_Area'] label\_encoder.fit\_transform(Train['Property\_Area']) Train['Loan\_Status'] label\_encoder.fit\_transform(Train['Loan\_Status'])  print(Train.head()) |

|  |  |  |  |
| --- | --- | --- | --- |
| In | [ | ]: | index = ['Gender', 'Married', 'Dependents', 'Education', 'Self\_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan\_Amount\_Term', 'Credit\_History', 'Property\_Area','LoanAmount\_log',  'Loan\_Status',] |
|  |  |  |  |
| In | [ | ]: | Train = Train[index] |
|  |  |  |  |
| In | [ | ]: | X = Train.iloc[:, :-1].values y = Train.iloc[:, -1].values |
|  |  |  |  |
| In | [ | ]: | from sklearn.model\_selection import train\_test\_split |
|  |  |  | X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 1/3, ran |
|  |  |  | dom\_state = 0) |
|  |  |  |  |
| In | [ | ]: | y\_train |
|  |  |  |  |
| In | [ | ]: | from sklearn import tree |
|  |  |  | from sklearn.tree import DecisionTreeClassifier |
|  |  |  | DT = DecisionTreeClassifier(criterion=“entropy“) |
|  |  |  | tree\_1 = DT.fit(X\_train, y\_train) |
|  |  |  |  |
| In | [ | ]: | y\_pred = DT.predict(X\_test) |
|  |  |  | from sklearn.metrics import confusion\_matrix, accuracy\_score |
|  |  |  | cm = confusion\_matrix(y\_test, y\_pred) |
|  |  |  | print(cm) |
|  |  |  | accuracy\_score(y\_test, y\_pred) |