**4th practical**

import pandas as pd

df=pd.read\_csv('BostonHousing.csv')

df

df.head()

print(df.columns)

X = df.drop('MEDV', axis=1)

Y = df['MEDV']

X.shape

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=0)

X\_train

X\_train.head()

X\_train.shape

X\_test.shape

from sklearn.impute import SimpleImputer

from sklearn.impute import SimpleImputer

X\_train = imputer.fit\_transform(X\_train)

X\_train = imputer.fit\_transform(X\_train)

X\_test = imputer.transform(X\_test)

from sklearn.linear\_model import LinearRegression

regressor=LinearRegression()

regressor.fit(X\_train,Y\_train)

regressor.coef\_

regressor.intercept\_

Y\_pred = regressor.predict(X\_test)

Y\_pred.shape

result=pd.DataFrame({'Actual':Y\_test,'Producted':Y\_pred})

result

residual\_error=(Y\_test-Y\_pred)

residual\_error

residual\_error

sum(residual\_error)/len(residual\_error)

from sklearn.metrics import mean\_absolute\_error

mean\_absolute\_error(Y\_pred,Y\_test)

from sklearn.metrics import mean\_absolute\_percentage\_error

mean\_absolute\_percentage\_error(Y\_test,Y\_pred)

regressor.score(X\_test,Y\_test)

#pip install scikit-Learn-U

#for the installation of scikit

from sklearn.metrics import r2\_score

r2\_score(Y\_test,Y\_pred)

new=[[0.7258,0,8.64,0,0.538,5.727,69.6,3.7965,4,307,22,391.95,11.28]]

new

regressor.predict(new)

**5th practical**

Social Network Ads Dataset – Kaggle

import pandas as pd

df = pd.read\_csv('Social\_Network\_Ads.csv')

df

#input data

x=df[['Age','EstimatedSalary']]

#output data

y=df['Purchased']

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

x\_scaled = scaler.fit\_transform(x)

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x\_scaled, y, test\_size=0.25, random\_state=0)

x\_train

y\_train

from sklearn.linear\_model import LogisticRegression

import seaborn as sns

sns.countplot(x=y)

y.value\_counts()

classifier = LogisticRegression()

classifier.fit(x\_train,y\_train)

y\_pred = classifier.predict(x\_test)

y\_train.shape

x\_train.shape

y\_pred

import matplotlib.pyplot as plt

plt.xlabel('Age')

plt.ylabel('Salary')

plt.scatter(x['Age'],x['EstimatedSalary'],c=y)

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

x\_scaled = scaler.fit\_transform(x)

pd.DataFrame(x\_scaled).describe()

plt.xlabel('Age')

plt.ylabel('Salary')

plt.scatter(x\_scaled[:,0],x\_scaled[:,1],c=y)

from sklearn.metrics import confusion\_matrix

confusion\_matrix(y\_test,y\_pred)

y\_test.value\_counts()

from sklearn.metrics import ConfusionMatrixDisplay

ConfusionMatrixDisplay.from\_estimator(classifier, x\_test, y\_test)

from sklearn.metrics import accuracy\_score

accuracy\_score(y\_test,y\_pred)

from sklearn.metrics import classification\_report

print(classification\_report(y\_test,y\_pred))

new1=[[26,34000]]

new2=[[57,138000]]

classifier.predict(scaler.transform(new1))

classifier.predict(scaler.transform(new2))

**8th practical**

import seaborn as sns

df= sns.load\_dataset('titanic')

df

df=df[['survived','class','sex','age','fare']]

df

sns.jointplot(x='age',y='fare',data=df)

sns.jointplot(x='age',y='fare',data=df,hue='survived')

sns.jointplot(x='age',y='fare',data=df,hue='class')

sns.pairplot(df,hue='sex')

sns.countplot(x=df['sex'])

sns.countplot(x=df['class'])

sns.barplot(x='sex',y='survived',data=df)

sns.histplot(df['fare'])

sns.kdeplot(df['fare,]

**9th practical**

import seaborn as sns

df = sns. load\_dataset('titanic')

df= df[['sex','age','survived']]

df

sns.boxplot(x='sex',y='age',data=df)

sns.boxplot(x='sex',y='age',hue='survived', data=df)

**10th practical**

sns.boxplot(x=df['petal\_width']);

df =sns.load\_dataset('iris')

df

df.columns

df.info()

df.dtypes

sns.pairplot(df,hue='species')

sns.pairplot(df)

sns.pairplot(df,hue='species',diag\_kind='hist')

sns.histplot(df['sepal\_length'],kde=True)

sns.histplot(df['sepal\_width'],kde=True)

sns.kdeplot(df['sepal\_width'])

sns.histplot(df['petal\_length'],kde=True)

sns.histplot(df['petal\_width'],kde=True)

sns.kdeplot(df['petal\_width'])

sns.boxplot(x=df['sepal\_length']);

sns.boxplot(x=df['sepal\_width']);

sns.boxplot(x=df['petal\_length']);

sns.boxplot(x=df['petal\_width']);

sns.boxplot(x=df['petal\_width']);