**Assignment 1**

#import the packges

import matplotlib.pyplot as plt

import pandas as pd

#Read Dataset

dataset=pd.read\_csv("hours.csv")

#index read

x=dataset.iloc[:,:-1].values #slice all column

y=dataset.iloc[:,1].values #last Column

#import packages of LR

from sklearn.linear\_model import LinearRegression

regressor=LinearRegression() #create object of LR

# Fit Function

regressor.fit(x,y)

#score Function

Accuracy=regressor.score(x,y)\*100

print('Accuracy')

print(Accuracy)

#Predict Function

y\_pred=regressor.predict([[10]])

print(y\_pred)

#input from user

hours=int(input("Enter the no of hours"))

# Coefficient

# intercept

eq=regressor.coef\_\*hours+regressor.intercept\_

print("Risk Score",eq[0])

plt.plot(x,y,'o')

plt.plot(x,regressor.predict(x));

plt.show()

**Assignment 2**

#import packages

import pydotplus

from sklearn.tree import export\_graphviz

from six import StringIO

from sklearn.tree import DecisionTreeClassifier

from sklearn.preprocessing import LabelEncoder

import pandas as pd

import numpy as np

# Read dataset

dataset = pd.read\_csv("tree1.csv")

x = dataset.iloc[:, :-1]

y = dataset.iloc[:, 5]

# Label encoder

le = LabelEncoder()

x = x.apply(le.fit\_transform)

print(x)

# 1 1 0 0

# import Decesion Tree Classifier

# Create decision tree classifer object

regressor = DecisionTreeClassifier()

# Train model

regressor.fit(x.iloc[:, 1:5], y)

x\_in = np.array([1, 1, 0, 0])

y\_pred = regressor.predict([x\_in])

print(y\_pred)

#from IPython.display import Image

dot\_data = StringIO()

export\_graphviz(regressor, out\_file=dot\_data, filled=True,

rounded=True, special\_characters=True)

# Draw Graph

graph = pydotplus.graph\_from\_dot\_data(dot\_data.getvalue())

# Show graph & Create png File

graph.write\_png("tree.png")

**Assignment 3**

# import the packages

from sklearn.neighbors import KNeighborsClassifier

import pandas as pd

import numpy as np

# Read dataset

dataset = pd.read\_csv("kdata.csv")

X = dataset.iloc[:, :-1].values

y = dataset.iloc[:, 2].values

# import KNeighborshood Classifier and create object of it

# Creating model

classifier = KNeighborsClassifier(n\_neighbors=3)

# Training model

classifier.fit(X, y)

# predict the class for the point(6,6)

X\_test = np.array([6, 6])

# Predictions for test data

y\_pred = classifier.predict([X\_test])

print(y\_pred)

# KNeighborsClassifier looks for the 5 nearest neighbors

# If set to uniform, all points in each neighbourhood have

# equal influence in predicting class i.e. predicted class is the class with highest number of points in the neighbourhood.

classifier = KNeighborsClassifier(n\_neighbors=3, weights='distance')

classifier.fit(X, y)

# predict the class for the point(6,6)

X\_test = np.array([6, 2])

y\_pred = classifier.predict([X\_test])

print(y\_pred)

**Assignment 4**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

x = [[0.1,0.6],[0.15,0.71],[0.08,0.9],[0.16,0.85],[0.2,0.3],[0.25,0.6],[0.24,0.1],[0.3,0.2]]

centers = np.array([[0.1,0.6],[0.3,0.2]])

print("initial centroids",centers)

from sklearn.cluster import KMeans

model = KMeans(n\_clusters = 2, init = centers , n\_init = 1)

model.fit(x)

print('Label',model.labels\_)

print('P6 belongs to cluster',model.labels\_[5])

print('no of population around cluster',np.count\_nonzero(model.labels\_ == 1))

print('New Centroids:\n',model.cluster\_centers\_)