Name: Praveen Kumar K

Reg No: RA1911030010069

Exp 11: Implementation of learning algorithms for an application

# **Aim:**

1. Implementation of a Linear Regression algorithm to predict student's scores using the given dataset.
2. Implementation of Support Vector Classification algorithm to classify the cases of breast cancer

using the given dataset.

1. Implementation of K-means clustering algorithm to group the customers based on their

demographic detail using the given dataset.

# A: Linear Regression on Student's Score Code:

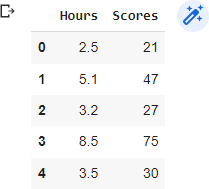
import pandas as pd import numpy as np

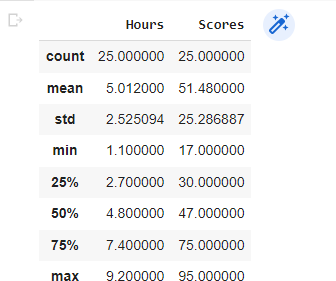
import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split from sklearn.linear\_model import LinearRegression from sklearn import metrics

%matplotlib inline

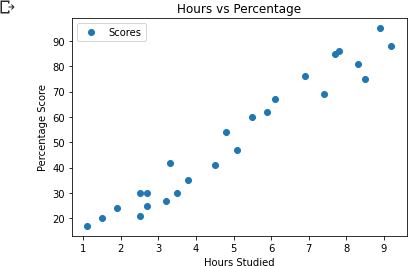
dataset = pd.read\_csv('student\_scores.csv') dataset.head()



dataset.describe()

dataset.plot(x='Hours', y='Scores', style='o') plt.title('Hours vs Percentage') plt.xlabel('Hours Studied') plt.ylabel('Percentage Score')

plt.show()



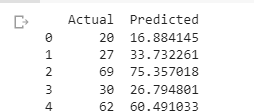
X = dataset.iloc[:, :-1].values y = dataset.iloc[:, 1].values

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

print('X train shape: ', X\_train.shape) print('Y train shape: ', Y\_train.shape) print('X test shape: ', X\_test.shape) print('Y test shape: ', Y\_test.shape)

regressor = LinearRegression() regressor.fit(X\_train, y\_train) print(regressor.intercept\_) print(regressor.coef\_)

df = pd.DataFrame({'Actual': y\_test, 'Predicted': y\_pred}) print(df)



print('Mean Absolute Error:',metrics.mean\_absolute\_error (y\_test, y\_pred)) print('Mean Squared Error:',metrics.mean\_squared\_error (y\_test, y\_pred)) print('Root Mean Squared Error:',np.sqrt(metrics.mean\_squared\_error (y\_test, y\_pred)))



B: Support Vector Classification algorithm to classify the cases of breast cancer

import pandas as pd import numpy as np

import matplotlib.pyplot as plt import seaborn as sns

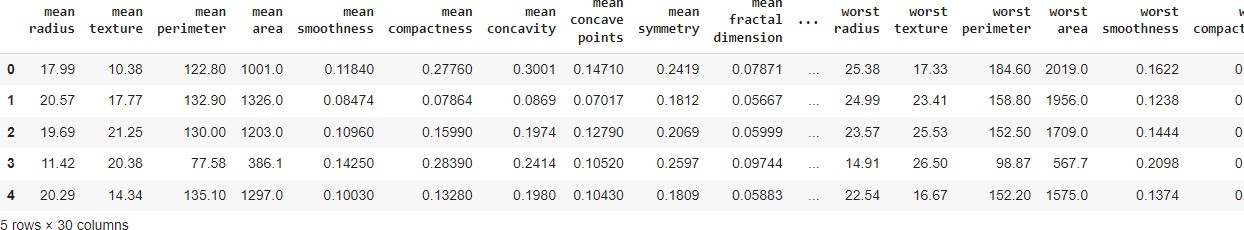
from sklearn.svm import SVC

%matplotlib inline

from sklearn.datasets import load\_breast\_cancer cancer = load\_breast\_cancer()

df\_cancer = pd.DataFrame(np.c\_[cancer['data'], cancer['target']], columns

= np.append(cancer['feature\_names'], ['target'])) df\_cancer.head()

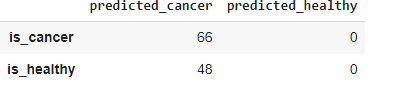


X = df\_cancer.drop(['target'], axis = 1) # We drop our "target" feature and use all the remaining features in our dataframe to train the model. X.head()

y = df\_cancer['target'] y.head()

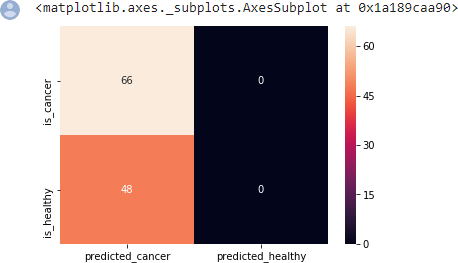
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 20)

svc\_model = SVC() svc\_model.fit(X\_train, y\_train) y\_predict = svc\_model.predict(X\_test)



cm = np.array(confusion\_matrix(y\_test, y\_predict, labels=[1,0])) confusion = pd.DataFrame(cm, index=['is\_cancer', 'is\_healthy'],

columns=['predicted\_cancer','predicted\_healthy']) sns.heatmap(confusion, annot=True)



# C: K-means clustering algorithm to group the customers based on their demographic detail using the given dataset.

Code:

import numpy as nm

import matplotlib.pyplot as mtp import pandas as pd

dataset = pd.read\_csv('Mall\_Customers\_data.csv')

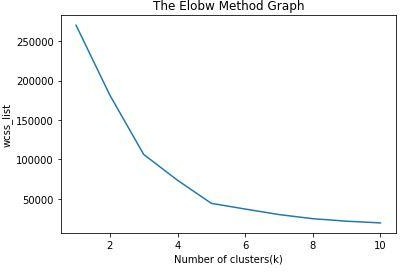
from sklearn.cluster import KMeans wcss\_list= []

#Using for loop for iterations from 1 to 10. for i in range(1, 11):

kmeans = KMeans(n\_clusters=i, init='k-means++', random\_state= 42) kmeans.fit(x)

wcss\_list.append(kmeans.inertia\_) mtp.plot(range(1, 11), wcss\_list) mtp.title('The Elobw Method Graph') mtp.xlabel('Number of clusters(k)') mtp.ylabel('wcss\_list')

mtp.show()



kmeans = KMeans(n\_clusters=5, init='k-means++', random\_state= 42) y\_predict= kmeans.fit\_predict(x)

mtp.scatter(x[y\_predict == 0, 0], x[y\_predict == 0, 1], s = 100, c = 'blue', label = 'Cluster 1') #for first cluster

mtp.scatter(x[y\_predict == 1, 0], x[y\_predict == 1, 1], s = 100, c = 'green', label = 'Cluster 2') #for second cluster

mtp.scatter(x[y\_predict== 2, 0], x[y\_predict == 2, 1], s = 100, c = 'red', label = 'Cluster 3') #for third cluster

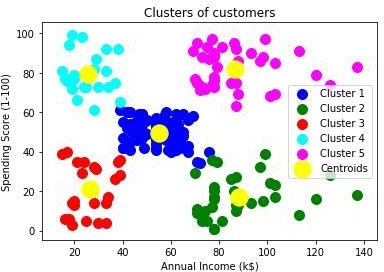
mtp.scatter(x[y\_predict == 3, 0], x[y\_predict == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4') #for fourth cluster

mtp.scatter(x[y\_predict == 4, 0], x[y\_predict == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5') #for fifth cluster

mtp.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], s = 300, c = 'yellow', label = 'Centroid')

mtp.title('Clusters of customers') mtp.xlabel('Annual Income (k$)') mtp.ylabel('Spending Score (1-100)') mtp.legend()

mtp.show()



**Result**:

Hence, we successfully implemented Linear Regression, SVM and K-means, verified the output, and documented the result.