**8.** **Write a program to implement k-Nearest Neighbour algorithm to classify the**

**iris data set. Print both correct and wrong predictions. Java/Python ML library**

**classes can be used for this problem.**

THEORY:**K-Nearest Neighbour Algorithm**

* K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
* K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
* K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
* K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
* K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
* It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
* KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

**How does KNN works?**

o Step-1: Select the number K of the neighbors

o Step-2: Calculate the Euclidean distance of K number of neighbors

o Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

o Step-4: Among these k neighbors, count the number of the data points in each category.

o Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

o Step-6: Our model is ready.

**PROCEDURE/PROGRAM:**

from sklearn.datasets import load\_iris

from sklearn.neighbors import KNeighborsClassifier

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

import numpy as np

dataset=load\_iris()

X\_train,X\_test,y\_train,y\_test=train\_test\_split(dataset["data"],dataset["target"],random\_state=0)

kn=KNeighborsClassifier(n\_neighbors=1)

kn.fit(X\_train,y\_train)

KNeighborsClassifier(algorithm='auto', leaf\_size=30, metric='minkowski',

metric\_params=None, n\_jobs=1, n\_neighbors=1, p=2,

weights='uniform')

for i in range(len(X\_test)):

    x=X\_test[i]

    x\_new=np.array([x])

    prediction=kn.predict(x\_new)

    print("TARGET=",y\_test[i],dataset["target\_names"][y\_test[i]],"PREDICTED=",prediction,dataset["target\_names"][prediction])

print(kn.score(X\_test,y\_test))

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 2 virginica PREDICTED= [2] ['virginica']

TARGET= 1 versicolor PREDICTED= [1] ['versicolor']

TARGET= 0 setosa PREDICTED= [0] ['setosa']

TARGET= 1 versicolor PREDICTED= [2] ['virginica']

0.9736842105263158