K-Nearest Neighbour Algorithm

Problem Statement: To predict the weight using KNN algorithm without the usage of any packages.

Formulas used: Euclidean distance formula-The distance two points (x1,y1) and (x2,y2) is given by the formula :

[(x2-x1)2 +(y2-y1)2] 1/2

Algorithm:

Step1: Start
Step 2: Load the train data
Step 3:Load the test data
Step 4:Assign k values
Step 5:Assign target variable
Step 6:Create the variable to store the predicted targeted values
Step 7:Repeat through the steps:
Find the difference matrix
Compute the distance using Euclidean distance formula
Sort the train data in ascending order w.r.t the distances
Compute average of the first k terms of train dataset Append to predicted targeted values.
Step 8: Display the predicted targeted values
Step 9: Stop

Code:

```
# -*- coding: utf-8 -*-
"""
 @script-author:Melisa Dalmeida
@script-description:To predict the value using knn algorithm
 @script-start date:08.01.20
 @script-last updated:13.01.20
```

"""

```
#setting train and test data
train=[[13,14,16],[12,17,14],[11,15,18]]
test=[13,19,17]
diff=[]
#Computing the difference matrix
for i in range(len(train)):
 im=[]
 for j in range(len(test)):
  im.append(test[j]-train[i][j])
 diff.append(im)
dist=[]
#Computing distance using euclidian formula
for i in range(len(train)):
 s=0
 for j in range(len(test)):
  s+=diff[i][j]**2
 dist.append(s)
dict1={} # creating a dictionary to link the train data and the distance
calculated
for i in range(len(dist)):
 dict1[dist[i]]=train[i]
#sorting based on distance
dict1=sorted(dict1.items())
dict1
#Using the k values estimating the predicted value
predict,s=[],0
for i in range(len(dict1)):
 s+=dict1[i][1][2]
 predict.append(s/(i+1))
predict
#Estimaing the error
```

```
error=[]
for i in range(len(predict)):
 error.append((test[2]-
predict[i])*100/test[2])
error
#based on the least error estimating the predicted value
print("Accurate value is ",predict[error.index(min(error))])
```

OUTPUT:

```
Accurate value is  16.0
```