Dhruvin Chawda
60004210159

Exp 6 ML

Aim : To implement K-Nearest Neighbour.

Theory : KNN is a simple ML algo based on Supervised Learning technique. It assumes similarity between new case and available cases and put new case into category that is most like available category. This means new data appears then it can be easily classified into a well suite category by using KNN algorithm. It can be used for regression as well as for classification but mostly used for classification. It is called a lazy learner algo as it does not learn from training set immediately instead it stores dataset and at time of classification it performs action on dataset. KNN working is as :

① Select number of K neighbours
② Calculate euclidean distance of K number of neighbours.
③ Take K nearest neighbours as per calculated euclidean distance.
④ Among these K neighbour count number of data points in each category.
⑤ Assign new data points to that category for which no. of neighbour is maximum.
⑥ For selection of K in KNN:

→ There is no way to determine best value for 'K', most preferred value for K is 5.
→ A very low value for K such as K=1 or K=2 can be noisy and lead to effects of outliers in model.
→ Large values for K are good but it may find some difficulties.

*Advantages:
→ It is simple to implement
→ It is robust to noisy training data
→ It can be more effective if training data is large.
* Disadvantages:
→ Always needs to determine value of K which is complex.
→ The computation cost is high because of calculating dist between data points for all training samples.

Problem:

dataset 1: Let new data point be $(x_n, y_n) = (5.5, 38)$

ID	height(x_i)	Age(y_i)	$\sqrt{(x_i - x_n)^2 + (y_i - y_n)^2}$	weight
1	5	45	$((5-5.5)^2 + (45-38)^2)^{1/2} = 7.0178$	77
2	5.11	26	$((5.11-5.5)^2 + (26-38)^2)^{1/2} = 12.006$	47
3	5.6	30	$((5.6-5.5)^2 + (30-38)^2)^{1/2} = 8$	55
4	5.9	34	$((5.9-5.5)^2 + (34-38)^2)^{1/2} = 4.019$	59
5	4.8	40	$((4.8-5.5)^2 + (40-38)^2)^{1/2} = 2.118$	72
6	5.8	36	$((5.8-5.5)^2 + (36-38)^2)^{1/2} = 2.022$	60
7	5.3	19	$((5.3-5.5)^2 + (19-38)^2)^{1/2} = 19.001$	40
8	5.8	28	$((5.8-5.5)^2 + (28-38)^2)^{1/2} = 10.004$	60
9	5.5	23	$((5.5-5.5)^2 + (23-38)^2)^{1/2} = 15$	45
10	5.6	32	$((5.6-5.5)^2 + (32-38)^2)^{1/2} = 6$	58

Now for data point (5.5, 38) if K=1 then weight will be 60 (as euclidean distance is less at that point as K=1 so only one nearest neighbour is checked) for K more than 1 average will be taken.

If K=3 then weight = (59+72+60)/3 = 63.6667

If K=5 then weight = (59+72+60+58+77)/5 = 65.2

Dataset 2 Let new data point (xn, yn) = (170, 57)

Height (xi)	weight (yi)	$\sqrt{(xi-xn)^2+(yi-yn)^2}$	class
167	51	$((167-170)^2+(51-57)^2)^{1/2} = 6.708$	Underweight
182	62	$((182-170)^2+(62-57)^2)^{1/2} = 13$	Normal
176	69	$((176-170)^2+(69-57)^2)^{1/2} = 13.416$	Normal
173	64	$((173-170)^2+(64-57)^2)^{1/2} = 7.615$	Normal
172	65	$((172-170)^2+(65-57)^2)^{1/2} = 8.246$	Normal
174	56	$((174-170)^2+(56-57)^2)^{1/2} = 4.123$	Underweight
169	58	$((169-170)^2+(58-57)^2)^{1/2} = 1.414$	Normal
173	57	$((173-170)^2+(57-57)^2)^{1/2} = 3$	Normal
170	55	$((170-170)^2+(55-57)^2)^{1/2} = 2$	Normal

for K=1, class for (170, 57) will be Normal as ~~nearest~~ one nearest neighbour is (169, 58) having class Normal

for K=3, 3 nearest neighbours class for (170, 57) are Normal (1.414), Normal (2), Normal (3). so class when K=3 is Normal.

for K=5, 5 nearest neighbour classes are Normal (1.414), Normal (2), Normal (3), ~~Normal~~ Underweight (4.123), Underweight (6.708), as most of the class is Normal. Therefore for K=5, class is Normal

Implemenation :

```
import numpy as np

from collections import Counter

def knn(x,y,test,k,way):

  for k in range(1,k+1):

    if k%2!=0:

      for i in test:

            dist = [np.sqrt(np.sum((i - x)**2)) for x in x]

            dist_index = np.argsort(dist)[:k]

            y_nearest = [y[i] for i in dist_index]

            if way == 'num':

              sum=0

              for i in y_nearest:

                  sum+=i

              predicted=sum/k

              print(f"so for {test} at k value as {k} we get {predicted}")

            elif way == 'cat':

              count_y = Counter(y_nearest)

              predicted,_ = count_y.most_common(1)[0]

              print(f"so for {test} at k value as {k} we get {predicted}")

ds1 = np.array([[5, 45], [5.11, 26], [5.6, 30], [5.9, 34], [4.8, 40], [5.8, 36], [5.3, 19],
[5.8, 28], [5.5, 23], [5.6, 32]])

class1 = np.array([77, 47, 55, 59, 72, 60, 40, 60, 45, 58])

y1 = np.array([5.5, 38])

ds2 = np.array([[167, 51], [182, 62], [176,
69],[173,64],[172,65],[174,56],[169,58],[173,57],[170,55]])

class2 = np.array([0,1,1,1,1,0,1,1,1])

y2=np.array([170,57])

print("for 1st dataset")

knn(ds1, class1, np.array([y1]), 5,'num')

print("for 2nd dataset")

knn(ds2, class2, np.array([y2]), 5,'cat')

print("for other dataset :")

url = "https://raw.githubusercontent.com/Dhruvin3103/ML/main/CSV/suv_data.csv"

ds = pd.read_csv(url)

x = ds.iloc[:, 2:4]

y = ds.iloc[:, -1]
```

```
test = x.iloc[-1,:]

if x.isnull().values.any():

    x = x.fillna(x.mean())

knn(x, [y], np.array([test]), k=5)
```

Output :

Q1)

```
for 1st dataset
so for [[ 5.5 38. ]] at k value as 1 we get 60.0
so for [[ 5.5 38. ]] at k value as 3 we get 63.666666666666664
so for [[ 5.5 38. ]] at k value as 5 we get 65.2
```

Q2)

```
for 2nd dataset
so for [[170  57]] at k value as 1 we get 1
so for [[170  57]] at k value as 3 we get 1
so for [[170  57]] at k value as 5 we get 1
```

Q3)

```
for other dataset :
For test data [  0.37   0.52   2.   158.     3.  ], at k=1, predicted salary is: low
For test data [  0.37   0.52   2.   158.     3.  ], at k=3, predicted salary is: low
For test data [  0.37   0.52   2.   158.     3.  ], at k=5, predicted salary is: low
```