09/22/22
Sehyun Cho
CAP4612

Homework 2

- **Describe the algorithms/approaches/tools used:**
  - **a. <u>What it is or What it does</u>**

    Algorithms used: Used KNN-pseudocode from the slides which,
    Step 1: Determine parameter K = number of nearest neighbors
    Step 2: Calculate the distance between the query-instance and all the training examples
    Step 3: Sort the distance and determine nearest neighbors based on the k-th minimum distance.
    Step 4: Gather the category Y of the nearest neighbors.
    Step 5: Use simple majority of the category of nearest neighbors as the prediction value of the query instance.

    Tools used: used DataFrame by pandas library to stores data, KNeighborsClassifier and train_test_split library for fitting data.
  - **b. <u>How it does & Application</u>**

    1. Hardcode the table 1 from the instruction and sorting as classes.

Figure 1 – Hardcode distance table

```
# Import the dataset as Multidimensional Array (x1 ~ x19)
dataset = ([0,1.5,1.4,1.6,1.7,1.3,1.6,1.5,1.4,2.3,2.9,3.2,3.3,3.4,4.2,4.1,5.9,6.1,6.0],
           [1.5,0,1.6,1.4,1.4,1.4,1.3,1.4,1.3,2.4,2.8,3.3,3.4,3.2,4.1,4.1,6.2,6.3,6.1],
           [1.4,1.6,0,1.3,1.5,1.4,1.4,1.6,1.4,2.5,2.9,3.2,3.2,3.5,4.1,4.1,6.2,6.2,6.2],
           [1.6,1.4,1.3,0,1.5,1.5,1.4,1.3,1.5,2.3,3.0,3.1,3.2,3.4,4.1,4.1,5.8,5.8,5.8],
           [1.7,1.4,1.5,1.5,0,1.4,1.5,1.7,1.2,2.6,2.9,3.3,3.3,3.7,4.1,4.1,6.1,6.1,6.1],
           [1.3,1.4,1.4,1.5,1.4,0,1.8,1.6,1.4,2.7,3.1,3.4,3.4,3.5,4.1,4.1,6.0,6.0,6.0],
           [1.6,1.3,1.4,1.4,1.5,1.8,0,1.4,1.3,2.8,2.9,3.3,3.2,3.6,4.1,4.1,6.1,6.1,6.1],
           [1.5,1.4,1.6,1.3,1.7,1.6,1.4,0,1.5,2.7,3.1,3.4,3.3,3.3,4.1,4.1,5.9,5.9,5.9],
           [1.4,1.3,1.4,1.5,1.2,1.4,1.3,1.5,0,3.1,3.0,3.5,3.5,3.5,4.1,4.1,5.8,5.8,5.8],
           [2.3,2.4,2.5,2.3,2.6,2.7,2.8,2.7,3.1,0,1.5,3.3,3.6,3.6,4.1,4.1,6.0,6.0,6.0],
           [2.9,2.8,2.9,3.0,2.9,3.1,2.9,3.1,3.0,1.5,0,3.3,3.6,3.6,4.1,4.1,6.0,6.0,6.0],
           [3.2,3.3,3.2,3.1,3.3,3.4,3.3,3.4,3.5,3.3,1.6,0,1.4,1.5,1.7,1.6,2.3,3.1,3.0],
           [3.3,3.4,3.2,3.2,3.3,3.4,3.2,3.3,3.5,3.6,1.4,1.7,0,1.8,1.6,1.5,2.3,2.7,2.9],
           [3.4,3.2,3.5,3.4,3.7,3.5,3.6,3.3,3.5,3.6,1.5,1.8,0.5,0,0.3,0.4,2.5,2.6,2.7],
           [4.2,4.1,4.1,4.1,4.1,4.1,4.1,4.1,4.1,4.1,1.7,1.6,0.3,0.5,0,0.5,2.3,2.3,2.4],
           [4.1,4.1,4.1,4.1,4.1,4.1,4.1,4.1,4.1,4.1,1.6,1.5,0.4,0.5,0.4,0,2.4,2.5,2.5],
           [5.9,6.2,6.2,5.8,6.1,6.0,6.1,5.9,5.8,6.0,2.3,2.3,2.5,2.3,2.4,2.5,0,2.6,2.8],
           [6.1,6.3,6.2,5.8,6.1,6.0,6.1,5.9,5.8,6.0,3.1,2.7,2.6,2.3,2.5,2.6,3.0,0,0.4],
           [6.0,6.1,6.2,5.8,6.1,6.0,6.1,5.9,5.8,6.0,3.0,2.9,2.7,2.4,2.5,2.8,3.1,0.4,0])
```

2. Assign dataset indexes as instruction shows.

Figure 2 – Sort dataset as instruction shows

```
# Set each index of dataset as x1 ~ x19
x1 = dataset[0]
x2 = dataset[1]
x3 = dataset[2]
x4 = dataset[3]
x5 = dataset[4]
x6 = dataset[5]
x7 = dataset[6]
x8 = dataset[7]
x9 = dataset[8]
x10 = dataset[9]
x11 = dataset[10]
x12 = dataset[11]
x13 = dataset[12]
x14 = dataset[13]
x15 = dataset[14]
x16 = dataset[15]
x17 = dataset[16]
x18 = dataset[17]
x19 = dataset[18]
```

```
# Original Sample
class1 = [x1,x2,x5,x6,x9,x10,x13,x14,x17,x18]
class2 = [x3,x4,x7,x8,x11,x12,x15,x16,x19]

# Training Sample
training_class1 = [x1,x5,x9,x13,x17]
training_class2 = [x3,x7,x11,x15,x19]

# Test sample
test_class1 = [x2,x6,x10,x14,x18]
test_class2 = [x4,x8,x12,x16]
```

3. Put test samples into the data frame with label 1 =’class1’, label 2 =’class2’.

Figure 3 – Acquire data frame, test samples with label

	x2	x6	x10	x14	x18	x4	x8	x12	x16	label
0	1.5	1.3	2.3	3.4	6.1	1.6	1.5	3.2	4.1	1
1	1.4	1.4	2.6	3.7	6.1	1.5	1.7	3.3	4.1	1
2	1.3	1.4	3.1	3.5	5.8	1.5	1.5	3.5	4.1	1
3	3.4	3.4	3.6	0.5	2.6	3.2	3.3	3.6	0.4	1
4	6.2	6.0	6.0	2.5	3.0	5.8	5.9	2.3	2.4	1
5	1.6	1.4	2.5	3.5	6.2	1.3	1.6	3.2	4.1	2
6	1.3	1.8	2.8	3.6	6.1	1.4	1.4	3.3	4.1	2
7	2.8	3.1	1.5	1.5	3.1	3.0	3.1	1.6	1.6	2
8	4.1	4.1	4.1	0.3	2.5	4.1	4.1	4.1	0.4	2
9	6.1	6.0	6.0	2.7	0.4	5.8	5.9	3.0	2.5	2

4. Split dataset ‘y’ as label, ‘x’ as the other datasets.

Figure 4 – Split dataset y as label

```
# label 1 = class1, label2 = class
y = df_all['label']
y
```

✓ 0.1s

```
0    1
1    1
2    1
3    1
4    1
5    2
6    2
7    2
8    2
9    2
Name: label, dtype: int64
```

5. Sort the target sample, and shows each target represent distance from which datapoint.
   Figure 5 – Sort the target sample

	x2
x9	1.3
x7	1.3
x5	1.4
x1	1.5
x3	1.6
x11	2.8
x13	3.4
x15	4.1
x19	6.1
x17	6.2

6. Split the data to use classifier library.
7. Append accuracy list by K-th values change.

- **Describe results:**

  Figure 6 – Accuracy list by K value changes

```
[0.0,
 0.3333333333333333,
 0.0,
 0.3333333333333333,
 0.3333333333333333,
 0.3333333333333333,
 0.3333333333333333]
```

1. **<u>Describe the figure and table.</u>**
   Stored accuracies into the list whenever K changes, from 1 to 8.
2. **<u>Your observation about the figure and table.</u>**
   From my observation, the accuracy is not high as I expected, and K values did not execute until it reaches 10. On the other hand, only with those accuracies, I would explain when K is 2,4,5,6,7 is same as the best choices.
3. **<u>Conclusion.</u>**
   In conclusion, I believe there was some problem during the process of fitting and splitting datasets, that led could not test all K values which supposed to.