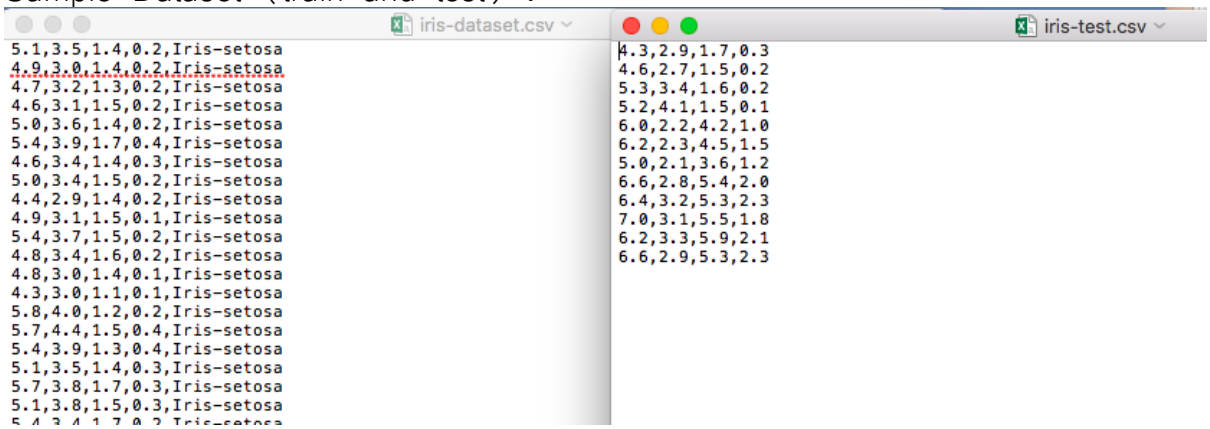
K-NEAREST NEIGHBOUR

DEVELOPED BY: VISHRUTH KHARE
ROLL NUMBER: 2K18/CO/393

IDE: PyCharm
Compiler: Terminal (Python 3.6)
DataSet: Iris Dataset

Sample Dataset (train and test):



iris-dataset.csv

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
```

iris-test.csv

```
4.3,2.9,1.7,0.3
4.6,2.7,1.5,0.2
5.3,3.4,1.6,0.2
5.2,4.1,1.5,0.1
6.0,2.2,4.2,1.0
6.2,2.3,4.5,1.5
5.0,2.1,3.6,1.2
6.6,2.8,5.4,2.0
6.4,3.2,5.3,2.3
7.0,3.1,5.5,1.8
6.2,3.3,5.9,2.1
6.6,2.9,5.3,2.3
```

Source Code:

```
from csv import reader
from sys import exit
from math import sqrt
from operator import itemgetter


def load_data_set(filename):
    try:
        with open(filename, newline='') as iris:
            return list(reader(iris, delimiter=','))
    except FileNotFoundError as e:
        raise e


def convert_to_float(data_set, mode):
    new_set = []
    try:
        if mode == 'training':
            for data in data_set:
                new_set.append([float(x) for x in data[:len(data)-1]] +
[data[len(data)-1]])

        elif mode == 'test':
            for data in data_set:
                new_set.append([float(x) for x in data])

        else:
            print('Invalid mode, program will exit.')
            exit()
```

```
        return new_set

    except ValueError as v:
        print(v)
        print('Invalid data set format, program will exit.')
        exit()


def get_classes(training_set):
    return list(set([c[-1] for c in training_set]))


def find_neighbors(distances, k):
    return distances[0:k]


def find_response(neighbors, classes):
    votes = [0] * len(classes)

    for instance in neighbors:
        for ctr, c in enumerate(classes):
            if instance[-2] == c:
                votes[ctr] += 1

    return max(enumerate(votes), key=itemgetter(1))


def knn(training_set, test_set, k):
    distances = []
    dist = 0
    limit = len(training_set[0]) - 1

    # generate response classes from training data
    classes = get_classes(training_set)

    try:
        for test_instance in test_set:
            for row in training_set:
                for x, y in zip(row[:limit], test_instance):
                    dist += (x-y) * (x-y)
                distances.append(row + [sqrt(dist)])
                dist = 0

            distances.sort(key=itemgetter(len(distances[0])-1))

            # find k nearest neighbors
            neighbors = find_neighbors(distances, k)

            # get the class with maximum votes
            index, value = find_response(neighbors, classes)

            # Display prediction
            print('DataSet: ' + str(test_instance) + ' is : ' +
classes[index])
            print('Closeness: ' + str(value) + '/' + str(k))

            # empty the distance list
            distances.clear()

    except Exception as e:
```

```
        print(e)


def main():
    try:
        # get value of k
        k = int(input(' k : '))

        # load the training and test data set
        training_file = input('Enter name of training data file : ')
        test_file = input('Enter name of test data file : ')
        training_set = convert_to_float(load_data_set(training_file),
'training')
        test_set = convert_to_float(load_data_set(test_file), 'test')

        if not training_set:
            print('Empty training set')

        elif not test_set:
            print('Empty test set')

        elif k > len(training_set):
            print('Expected number of neighbors is higher than number of
training data instances')

        else:
            knn(training_set, test_set, k)


if __name__ == '__main__':
    main()
```

OUTPUT:

```
Closeness: 11/11
DataSet: [4.6, 2.7, 1.5, 0.2] is : Iris-setosa
Closeness: 11/11
DataSet: [5.3, 3.4, 1.6, 0.2] is : Iris-setosa
Closeness: 11/11
DataSet: [5.2, 4.1, 1.5, 0.1] is : Iris-setosa
Closeness: 11/11
DataSet: [6.0, 2.2, 4.2, 1.0] is : Iris-versicolor
Closeness: 11/11
DataSet: [6.2, 2.3, 4.5, 1.5] is : Iris-versicolor
Closeness: 8/11
DataSet: [5.0, 2.1, 3.6, 1.2] is : Iris-versicolor
Closeness: 11/11
DataSet: [6.6, 2.8, 5.4, 2.0] is : Iris-virginica
Closeness: 11/11
DataSet: [6.4, 3.2, 5.3, 2.3] is : Iris-virginica
Closeness: 11/11
DataSet: [7.0, 3.1, 5.5, 1.8] is : Iris-virginica
Closeness: 10/11
DataSet: [6.2, 3.3, 5.9, 2.1] is : Iris-virginica
Closeness: 11/11
DataSet:[6.6, 2.9, 5.3, 2.3] is : Iris-virginica
Closeness: 11/11
(base) Vishruths-MacBook-Air-8:knn vishruthkhare$ _
```