Seth Kinsaul

Artificial Intelligence *COMP 5/6600, Fall2020*

HW05: K-means

Remember that only PDF submissions are accepted.

Requirement

Please submit your homework on Canvas before the deadline. If you need to use your delay coupon, please include your name, the student ID number, and how many days late it is (if handed in late) in the headline, and also make a comment on your submission on Canvas file submission system. If you have any questions, please contact our TA. (*), (**), or (***) indicates the difficulty of each question.

Policy

We apply the late policy explained in syllabus to all homework. Any grading questions must be raised with the TA in two weeks after the homework is returned. The homework must be completed individually. However, you are encouraged to discuss the general algorithms and ideas with classmates in order to help you answer the questions. You are also allowed to share examples that are not on the homework in order to demonstrate how to solve problems. If you work with one or more other people on the general discussion of the assignment questions, please record their names over every question they participated. However, the following behaviors will receive heavy penalties (lose all points and apply the honest policy explained in syllabus)

- explicitly tell somebody else the answers;
- explicitly copy answers or code fragments from anyone or anywhere;
- allow your answers to be copied;
- get code from Web.

1 (*) Ask Google for help: 2 points

List as least four applications of clustering algorithm.

1. Identifying fake news by clustering specific terms that have a high percentage of usage in the article. This helps determine whether the source is more likely to be fake news
2. Spam filter in email uses k means clustering algorithm to look at the different section of the email to determine it to be spam mail
3. Identifying fraudulent or criminal activity uses clustering to analyzing GPS logs to group similar behaviors. This helps determine which are real and which are fraudulent
4. Fantasy Football uses k means clustering algorithm to group together players with similar performance to help create a better team
5. Document analysis uses hierarchical clustering to identify themes of the documents, classify them and cluster them accordingly

2 (**) K-means: 8 points (programming homework)

Implement the K-means algorithm. Set the number of clusters as 3. The input data is a set of 2-D points shown in Figure 1 (You can download the coordinate file in "A.txt"). Please note that you may try a few times to obtain a reasonable clustering result.

You need to report

- the distortion function value

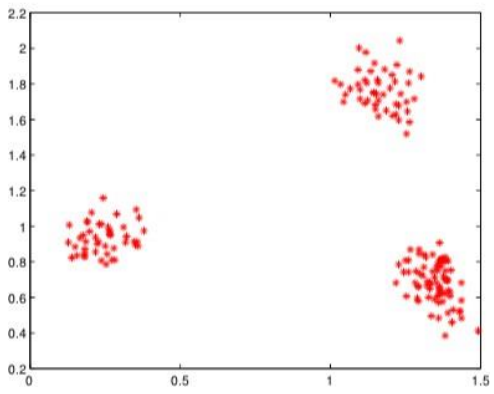
$$S = \sum_{i \in \text{ group 1}} \|x_i - c_1\|^2 + \sum_{i \in \text{ group 2}} \|x_i - c_2\|^2 + \sum_{i \in \text{ group 3}} \|x_i - c_3\|^2$$

  where c_1, c_2, and c_3 are the centers for three clusters you obtain;

- your clustering result shown in a figure (you should use different signs to mark 3 clusters; Photoshop or other similar softwares are forbidden in this problem).

You can use any programming language you feel comfortable. The code is not required to submit, but if TA had any question about your implementation or result, it is your obligation to provide runnable code which generates the result you submit. The fail to do that will cause losing points in this question. You should implement your own K-means algorithm. Directly calling existing functions for this algorithm from any programming languages is not permitted, but you are allowed to reuse any existing functions (even from your classmates) to print out your figure.

Figure 1: Input for K-means algorithm



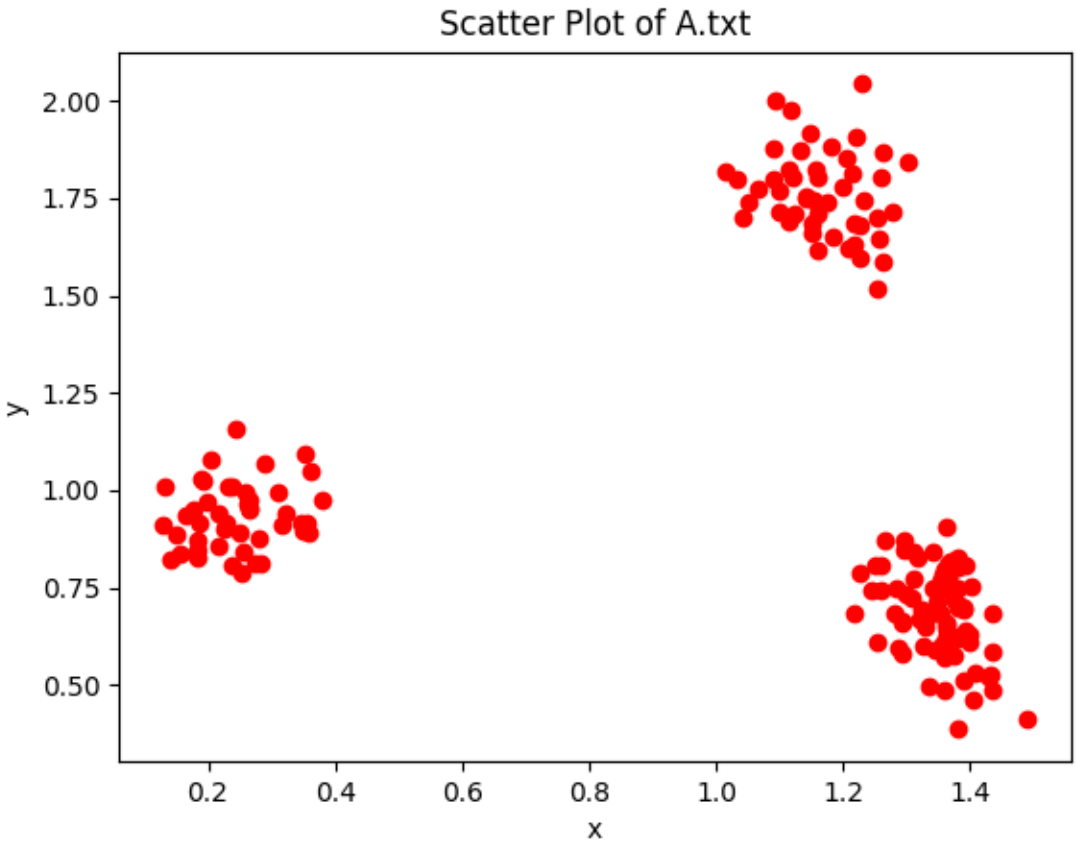
The distortion function value of A.txt is 18

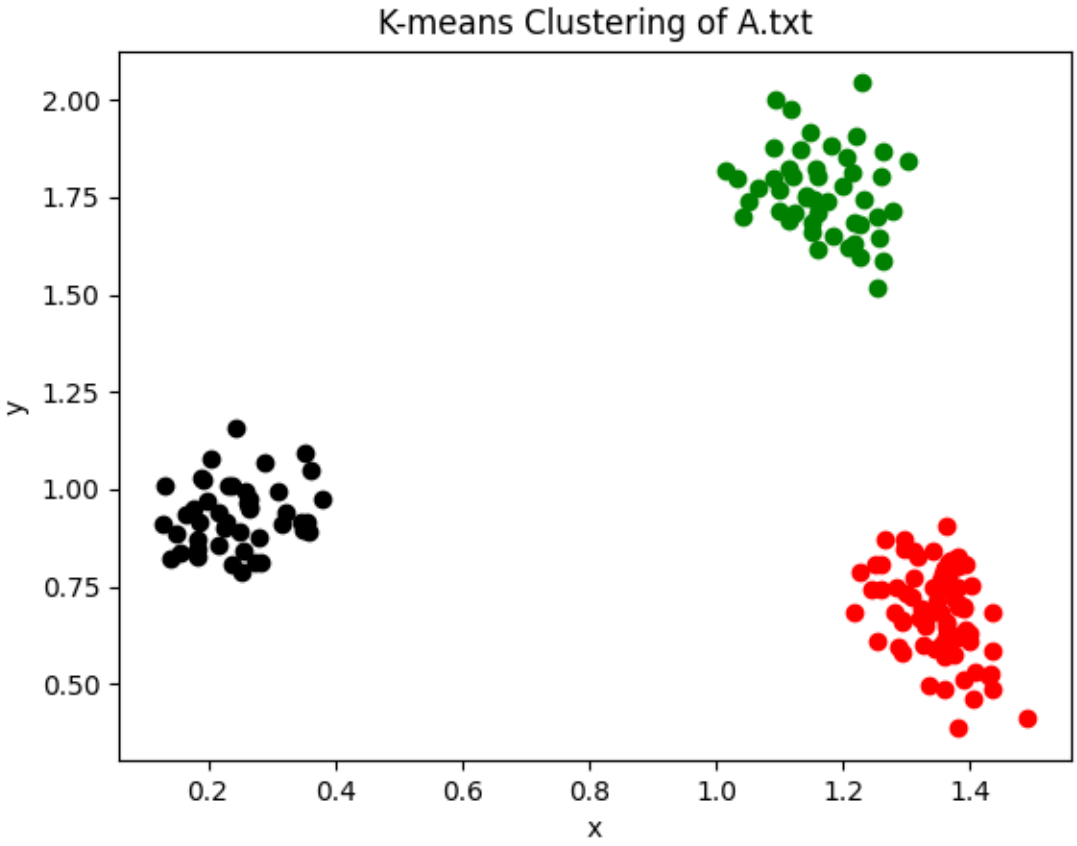
My code below:

```
"""
Seth Kinsaul (smk0036@auburn.edu)
10/15/2020
COMP 5600 Aritifical Intelligence
Homework 5
K means algorithm evaluates clusters by minimizing the distance of the data point from the average data point of the cluster (given the number of clusters)
"""
#Imports
import numpy as np
from matplotlib import pyplot as plt
class data:

    def __init__ (self, file_content, *colors, graph_name):
        self.file_content = file_content
        self.colors = colors
        self.graph_name = graph_name
        self.clusters = [] #list of clusters

    def init_clusters(self):
        for x in self.file_content:
            self.clusters.append([[(x[0],x[1])],0])

    def display_plot(self):
        plt.figure()
        plt.scatter(self.file_content[:,0], self.file_content[:,1], color = "red")
        plt.title("Scatter Plot of %s.txt" % self.graph_name)
        plt.xlabel("x")
        plt.ylabel("y")
        plt.show()

    def display_clusters(self):
        plt.figure()
        for i,x in enumerate(self.clusters):
            pts = x[0]
            x = []
            y = []
            for pt in pts:
                x.append(pt[0])
                y.append(pt[1])
            plt.scatter(x, y, color = '%s' % colors[i])
        plt.title("Hierarchical Clustering of %s.txt" % self.graph_name)
        plt.xlabel("x")
        plt.ylabel("y")
        plt.show()

    def calculateDistortion(self):
        S = 0
        for x in self.clusters:
            sum_x_values = 0
            sum_y_values = 0
            x_values = []
            y_values = []
            for pt in x[0]:
                x_val = pt[0]
                y_val = pt[1]
                x_values.append(x_val)
                y_values.append(y_val)
```

```
            sum_x_values += x_val
            sum_y_values += y_val
        avg_x_values = (sum_x_values/len(x_values))
        avg_y_values = (sum_y_values/len(y_values))
        sum_d = 0
        for i in range(len(x_values)):
            delta_x = (x_values[i] - avg_x_values)**2
            delta_y = (y_values[i] - avg_y_values)**2
            delta = np.sqrt(delta_x + delta_y)
            sum_d += delta
        S += sum_d
    return S

def k_means(self, k):
    self.init_clusters()
    while len(self.clusters) > k:
        best_d = 9999
        best_pt = self.clusters[0][0]
        best_pt2 = self.clusters[1][0]
        best_distance_x = 0
        best_distance_x2 = 1
        for x in range(len(self.clusters)):
            for y in range(len(self.clusters)):
                if y == x: continue
                else:
                    pt = self.clusters[x][0]
                    pt2 = self.clusters[y][0]

                    x_pt = []
                    y_pt = []

                    for i in pt:
                        x_pt.append(i[0])
                        y_pt.append(i[1])

                    x_pt2 = []
                    y_pt2 = []
                    for i in pt2:
                        x_pt2.append(i[0])
                        y_pt2.append(i[1])

                    sub_d_best = 9999
                    for i in range(len(pt)):
                        for j in range(len(pt2)):
                            delx = (x_pt[i]-x_pt2[j])**2
                            dely = (y_pt[i]-y_pt2[j])**2
                            sub_d = np.sqrt(delx+dely)
                            if sub_d <= sub_d_best:
                                sub_d_best = sub_d

                    d = sub_d_best

                    if d <= best_d:
                        best_d = d
                        best_pt = self.clusters[x]
                        best_pt2 = self.clusters[y]
                        best_distance_x = x
                        best_distance_x2 = y

        temp_pt_list = []
        temp_pt_list = self.clusters[best_distance_x][0] + self.clusters[best_distance_x2][0]
```

```
            self.clusters.remove(best_pt)
            self.clusters.remove(best_pt2)
            self.clusters.append([temp_pt_list,best_d])

if __name__  == '__main__':
    #load A.txt
    a = np.loadtxt('A.txt')
    #Colors for clusters
    colors = np.array(["red", "black", "green"])
    set_a = data(a, colors, graph_name = 'A')
    set_a.k_means(3)
    set_a.display_plot()
    set_a.display_clusters()
    S = set_a.calculateDistortion()
    print('Distortion Function Value = %d' % S) # In this instance S = 18
    plt.show()
```


Scatter Plot of A.txt
2.00
1.75
1.50
1.25
1.00
0.75
0.50
y
0.2
0.4
0.6
0.8
1.0
1.2
1.4
x


K-means Clustering of A.txt
2.00
1.75
1.50
1.25
1.00
0.75
0.50
y
0.2
0.4
0.6
0.8
1.0
1.2
1.4
x