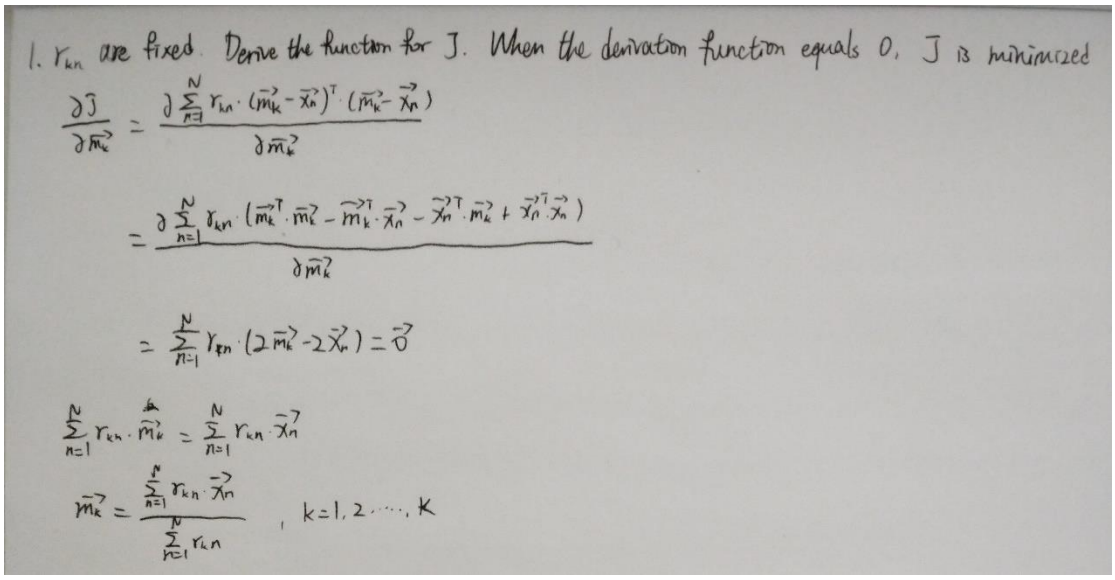
COEN 240 Machine Learning

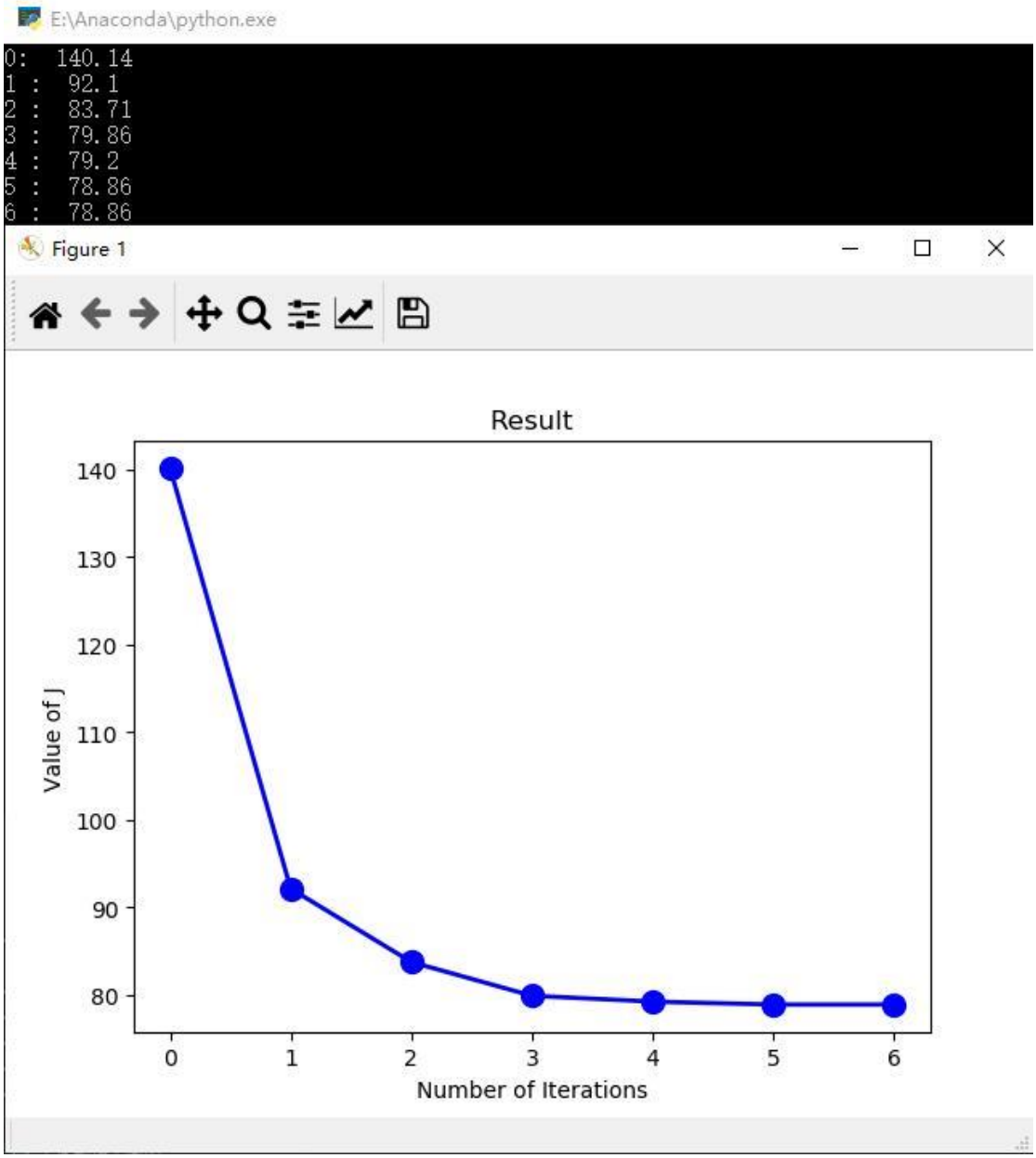
Homework #2

Name: Jinhao Wang ID: 4302178

Problem1:



Problem 2:



This plot shows the value of J (which is the sum of the squared distances of all data points to their assigned cluster centers) in K-means clustering against the number of iterations.

We can observer that, with the increase of iterations, the value of J will firstly decrease, and then tend to level off. The final value of J stabilizes at approximately 78.86.

The reason behind this is that after certain times of iterations, K-means clustering approach finds the optimized cluster centers, where the total distance of all points who belong to this center is minimized.

Problem 3:

3. a. math expression: $p(C_1|\vec{x}) = \frac{1}{1+\exp(-a)}$, $a = \vec{w}^T \cdot \vec{x} + w_0$

MAP criterion is used for the final classification

b. 2 parameters: $\vec{w}$ and w_0

Problem 4:

4. a. math expression: $p(C_k|\vec{x}) = \frac{\exp[a_k]}{\sum_{j=1}^{K} \exp[a_j]}$, $a_k = \vec{w_k}^T \cdot \vec{x} + w_{k0}$

MAP criterion is used for the final classification

b. We need to find $\vec{w_k}$, $k = 1, 2, \cdots, K$

Attachment:

Problem 2 Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
from math import sqrt
from random import uniform
import random

input = pd.read_excel('Iris.xls')
x = pd.DataFrame(input, columns = ["sepal length (cm)", "sepal width (cm)", "petal length (cm)", "petal width (cm)"])
y = pd.DataFrame(input, columns = ["outcome(Cluster Index)"])

def distance(pt1,pt2):
    i = 0
    sum = 0
    while i < 4:
        sum += (pt1[i]-pt2[i])**2
        i += 1
    return sqrt(sum)

def pt_assign(points,centres):
    assignments = []
    for pt in points:
        shortest = float('inf')
        centre_index = 0
        i = 0
        while i < len(centres):
            tmp = distance(pt,centres[i])
            if tmp < shortest:
                shortest = tmp
                centre_index = i
            i += 1
        assignments.append(centre_index)
    return assignments

def centres_update(points ,assignments,k):
    new_centres = []
    cluster1 = []
    cluster2 = []
```

```
    cluster3 = []
    for i in range(len(assignments)):
        if assignments[i] == 0:
            cluster1.append(points[i])
        elif assignments[i] == 1:
            cluster2.append(points[i])
        else:
            cluster3.append(points[i])
    i = 0
    centre1 = []
    centre2 = []
    centre3 = []
    while i < 4:
        sum1 = 0
        for pt in cluster1:
            sum1 += pt[i]
        centre1.append(float("%.2f"%(sum1/float(len(cluster1)))))
        sum2 = 0
        for pt in cluster2:
            sum2 += pt[i]
        centre2.append(float("%.2f"%(sum2/float(len(cluster2)))))
        sum3 = 0
        for pt in cluster3:
            sum3 += pt[i]
        centre3.append(float("%.2f"%(sum3/float(len(cluster3)))))
        i += 1
    new_centres.append(centre1)
    new_centres.append(centre2)
    new_centres.append(centre3)
    return new_centres

def jValue(assignments, points, centres):
    sum = 0
    for i in range(len(points)):
        sum += (distance(points[i],centres[assignments[i]]))**2
    return float("%.2f"%(sum))

data = np.array(x)
outcome = np.array(y)

centres = [data[random.randint(0,49)], data[random.randint(50,99)],
data[random.randint(100, 149)]]
assignments=pt_assign(data,centres)
value_j = jValue(assignments, data, centres)
```

```
itr = 1
tmp = 0
X = [0]
Y = [value_j]
print("0: ", value_j)
while value_j -tmp >= 0.00001:
    if tmp != 0:
        value_j = tmp
    new_centres=centres_update(data,assignments,3)
    assignments=pt_assign(data,new_centres)
    tmp = jValue(assignments, data, new_centres)
    X.append(itr)
    Y.append(tmp)
    print(itr, ": ", tmp)
    itr += 1

plt.plot(X, Y, color = 'blue', linestyle = 'solid', linewidth = 2, marker = 'o',
markerfacecolor = 'blue', markersize = 10)

plt.xlabel('Number of Iterations')
plt.ylabel('Value of J')
plt.xticks(range(len(X)))
plt.title('Result')
plt.show()
```