THE UNIVERSITY of ADELAIDE

Mining Big Data

Assignment on Frequent items, Clustering and Pagerank

Manivannan Meenakshi sundaram

Uni Adeladie

a1842066

Due Date : May 13, 2022

Assignment 3

Exercise 1 : Frequent Item-Sets

1. Frequent items with threshold 5

Frequent items with a threshold 5 are represented in the following set A

A={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20}

Explanation : Items from 1 to 20 appears at least 5 times when they get divided by a number b from the basket which satisfies the condition that they need to maintain the remainder 0 when they get divided by the number b.

2. Confidence of the following association rules.

   a. {5,7} -> 2

   Explanation : LCM of {5 ,7} is 35, and the multiples of 35 are {35,70} which are covered with the total baskets of 100.

   - Item 5 and 7 appears on only 2 baskets (i.e 35,70)
   - Item 5,7 and 2 appears only in 1 basket (i.e 70)

   Hence, Confidence of {5,7}-> is **1/2** => 0.5

   b. {2,3,4} -> 5

   Explanation : LCM of {2,3,4} is 12 and the multiples of 12 are {12,24,36,48,60,72,84,96} which comes under the constrain og 100 total baskets.

   - Item 2,3,4 appears on 8 baskets (i.e 12,24,36,48,60,72,84,96)
   - Item 2,3,4,5 appears on 1 basket (i.e 60)

   Hence, Confidence of {2,3,4}->5 is **1/8** => 0.125

Exercise 2 : Page rank Algorithm

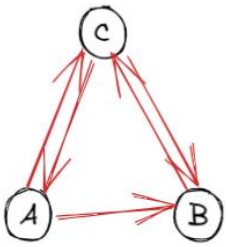
1. Implementation of Page rank algorithm with top 10 nodes.

Overview :

PageRank algorithm works in way that it gives rank to each pages and helps to rank accordingly in websites as they provide better display results. It helps for all the internet surfers who visits websites and visit other websites via the links present in the website. The random probability of the surfer lands on the page by clicking the link is defined by the PageRank algorithm. The websites with link and having a higher probability ranks in the top position along with other webpages.

Approach:

To solve this Web-Google pagerank, we will look in to the toy examples. Explaining parts of web-google is larger, hence making as a part of explanation.



This above diagram represents 3 nodes/web pages namely A,B.C and the arrow markings are the links which are traversed between the pages. They are linked together as shown in the diagram.

Initially, Page ranks for all pages are same and they have the same value. On a iterative approach the values get changed based on probability of the ranks each pages obtain. Hence they get changed and ranked accordingly in search results in Google.

With A,B,C

PageRank(A) + PageRank(B)+ PageRank(C)= 1/3 .

Based on 3 nodes, we initialize the rank for pages as 1/3. If there are n nodes 1/n will be initialized.

Factor which are considered to changes the ranks of the pages are inbound and outbound links, (From nodes and To nodes, in our case)

With this, pagerank of A will be

Pagerank (A)= 0 * pageRank(B)/ Outboundlink(B) + 1 * pagerank(c)/ outboundlink(C)

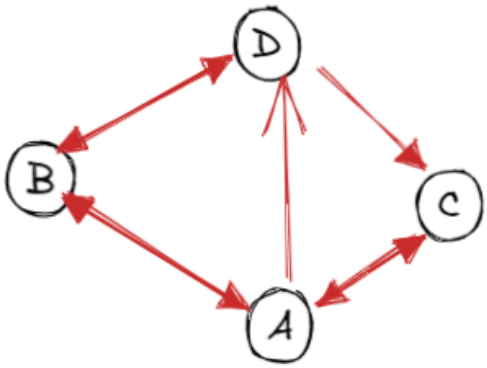
The co-efficient 0 and 1 are denotes the outbound link to the nodes. It has 0 because there is no outbound link to node A.

With the same calculation followed for other page ranks and iterating the procedure for times, we get the PageRank of all elements as 0.23,0.33 and 0.44.

If we sort the pagerank, we get the order as C -> B -> A , Page C ranks first than the other pages.

Given Nodes : [A,B,C,D]
Edges ={
(A, B)
(A,C)
(A,D)
(B,A)
(B,D)
(D,B)
(D,C)
(C,A)
}

Graph Representation :



Converting the above graph in to a matrix array,

MatrixArray ([1 , 1/3 ,1/3 , 1/2]

[1/2 , 0 , 0 ,1/3]

[1 , 0 , 0 , 0]

[0 , ½ , ½ , 0])

Matrix is arranged in such a way that it has n number of nodes as rows and columns respectively, and each element takes 1/n values based on the nodes.

Initially, all the nodes will have the same probability hence 1/n for each pages. Then to calculate the matrix we have to distribute a value v0.

Muliplyting the matrix m with v0 , we get,

VoM = [0.37, 0.20, 0.20, 0.20] => iteration 1.

With the result from iteration 1, we get page A has larger page rank. Iterating them to a certain amount of value. We have to check this until there is no change in pagerank between the nodes.

After 25 iterations, we get

VoM= [0.33, 0,22, 0.22,0.22]

In this iteration, We obtain a value which has highest rank for page A. This will remain same even for 50 iterations.

This is called Markov chain process. With this we can say that graph are strictly connected with each other.

Taxation:

Dealing with dead ends and spider traps are improves efficiency in ranking pages. Spider traps are nodes and all the edges are linked to its own pages.Having a self loop in pages causes spider traps and reduces the efficiency.

Dead ends are definded as pages with no out links, as a result they wont have any page ranks. With these negatives the probability of user going to other pages is 0. To avoid these kind of issues, we add a small value which will be added to the page where there is no outlinks.

It is represented as Beta(B)

$$v' = \beta M v + (1 - \beta) e/n$$

- Beta (B) is the constant which can be in the range of 0.8 to 0.9
- n be the number of pages.
- Beta(B) is called as taxation and as a damping factor as we are using networkx.

Once we deal with dead end and spider trap, we can see the pagrank of pages increases based on the availability of dead ends and traps.

Reading File :

Reading the given web-google file

With this approach, we get these page nodes as a top result.

Nodes , Page rank

[(597621, 0.0009241304287175943),

(41909, 0.0009214639855385305),

(163075, 0.0009049071491006198),

(537039, 0.0008992631802222364),

(384666, 0.0007872247076457703),

(504140, 0.0007655435306714571),

(486980, 0.0007262612106941519),

(605856, 0.0007190037632145288),

(32163, 0.0007129961306074605),

(558791, 0.0007103219427793649)]

Run time : --- Total run time: 576.5915057659149 seconds ---

Exercise 3 : Clustering

1. Hierarchical clustering on the one-dimensional set

One Dimensional set = { 1,4,9,16,25,36,49,64,81}

Explanation :

Centroids are calculated by taking mean(average) of the item set.

Round 1 :

Centroid = 1+4/2 = 2.5

Items	1	4	9	16	25	36	49	64	81
Centroid		2.5							
Distance		3	5	7	9	11	13	15	17

Centroid obtained is 2.5 and the closes distance to centroid is 3 , so the items 1 and 4 gets patched up

Items		(1,4)	9	16	25	36	49	64	81
Centroid		2.5							
Distance			6.5	7	9	11	13	15	17

Round 2 :

Centroid = 1+4+9/3 = 4.66

Items		(1,4)	9	16	25	36	49	64	81
Centroid			4.66						
Distance			6.5	7	9	11	13	15	17

Centroid obtained is 4.66 and the closes distance to centroid is 6.5 , so the items (1,4)and 9 gets patched up

Items			(1,4,9)	16	25	36	49	64	81
Centroid			4.66						
Distance				11.34	9	11	13	15	17

Round 3 :

Centroid = 1+4+9+16/4 = 7.5

Items			(1,4,9)	16	25	36	49	64	81
Centroid			4.66						
Distance				11.34	9	11	13	15	17

Centroid obtained is 7.5 and the closes distance to centroid is 9 , so the items 16 and 25 gets patched up

Items			(1,4,9)		(16,25)	36	49	64	81
Centroid			4.66		20.5				
Distance					15.9	15.5	13	15	17

Round 4 :

Centroid = 1+4+9+16+25/5 = 11

Items			(1,4,9)		(16,25)	36	49	64	81
Centroid			4.66		20.5				
Distance					15.9	15.5	13	15	17

Centroid obtained is 11 and the closes distance to centroid is 13 , so the items 36 and 49 gets patched up

Items			(1,4,9)		(16,25)		(36,49)	64	81
Centroid			4.66		20.5		42.5		
Distance					15.9		22	21.5	17

Round 5 :

Centroid = 1+4+9+16+25+36/6 = 15.1

Items			(1,4,9)		(16,25)		(36,49)	64	81
Centroid			4.66		20.5		42.5		
Distance					15.9		22	21.5	17

Centroid obtained is 15.1 and the closes distance to centroid is 15.9 , so the items (1,4,9) and (16,25) gets patched up

Items					(1,4,9,16,25)		(36,49)	64	81
Centroid					11		42.5		
Distance							31.5	21.5	17

Round 6 :

Centroid = 1+4+9+16+25+36+49/7 = 20

Items					(1,4,9,16,25)		(36,49)	64	81
Centroid					11		42.5		
Distance							31.5	21.5	17

Centroid obtained is 20 and the closes distance to centroid is 21.5 , so the items (36,49) and 64 gets patched up

Items					(1,4,9,16,25)			(36,49,64)	81
Centroid					11			49.6	
Distance								38.6	31.4

Round 7 :

Centroid = 1+4+9+16+25+36+49+64/8 = 25.5

Items					(1,4,9,16,25)			(36,49,64)	81
Centroid					11			49.6	
Distance								38.6	31.4

Centroid obtained is 25.5 and the closes distance to centroid is 31.4 , so the items (36,49,64) and 81 gets patched up

Items					(1,4,9,16,25)				(36,49,64,81)
Centroid					11				57.5
Distance									46.5

Round 8 :

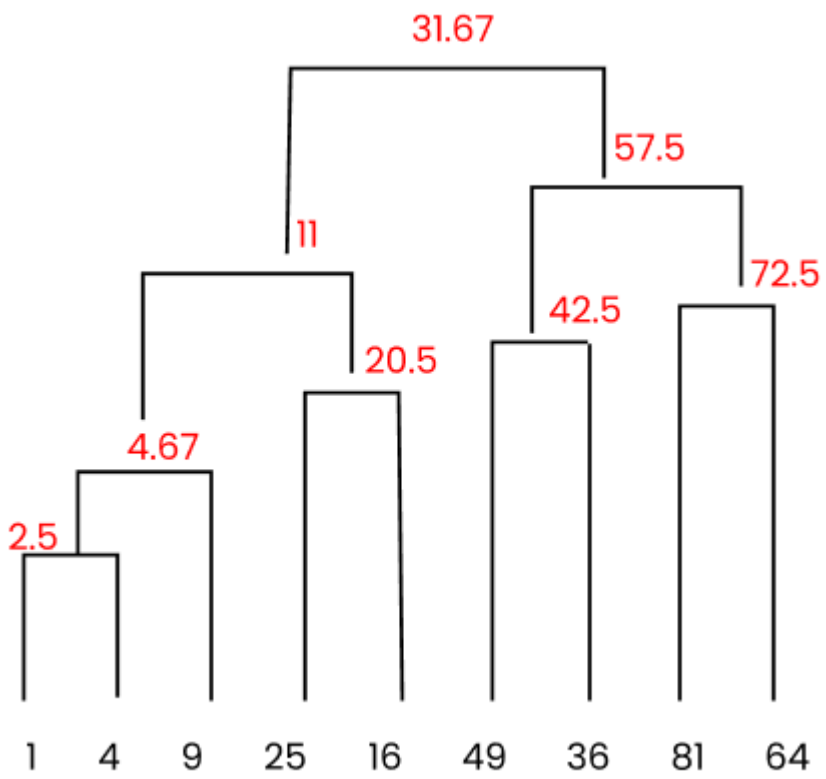
Centroid = 1+4+9+16+25+36+49+64+81/9 = 24.5

Items					(1,4,9,16,25)				(36,49,64,81)
Centroid					11				57.5
Distance									46.5

Centroid obtained is 24.5 and the closes distance to centroid is 46.5 , so the items (1,4,9,16,25) and (36,49,64,81) gets patched up

Items									(1,4,9,16,25,36,49,64,81)
Centroid									24.5
Distance									

Dendrograms :



2. K-means Clustering

    a. Explanation :

K-Means clustering is a unsupervised learning method that is used to predict a set of items which has similar characters. This algorithms helps to analyze a set of pattern and predicts items which have a similar pattern.

K- Means algorithm is a iterative process of certain steps which include the values of centroid and distance between the items.
K in K-Means is referred for clusters, which can be defined randomly or any methods like elbow. Random allocation doesn't have any efficiency and makes the modeling weak. Elbow method allows to maintain a correct ratio of value that helps to increase the efficiency and accuracy of the algorithm

In this task, we are using IRIS data set from scikit learn which has a data of 150 Rows with four features namely sepal length, sepal width, petal length and petal width.

Step 1 : Importing required libraries.

To start modeling the algorithm, importing the required libraries is necessary. We are using pandas and numpy to cluster and load the dataset. We also use mathplot library to plot graph based on the requirements.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
```

✓ 0.7s

Step 2 : Loading data set

In this task, we are using IRIS dataset, loading the dataset from Sklearn and it to a variable called data.

```
iris = load_iris()
data = pd.DataFrame(iris.data, columns=iris.feature_names)
data.head()
```

✓ 0.6s

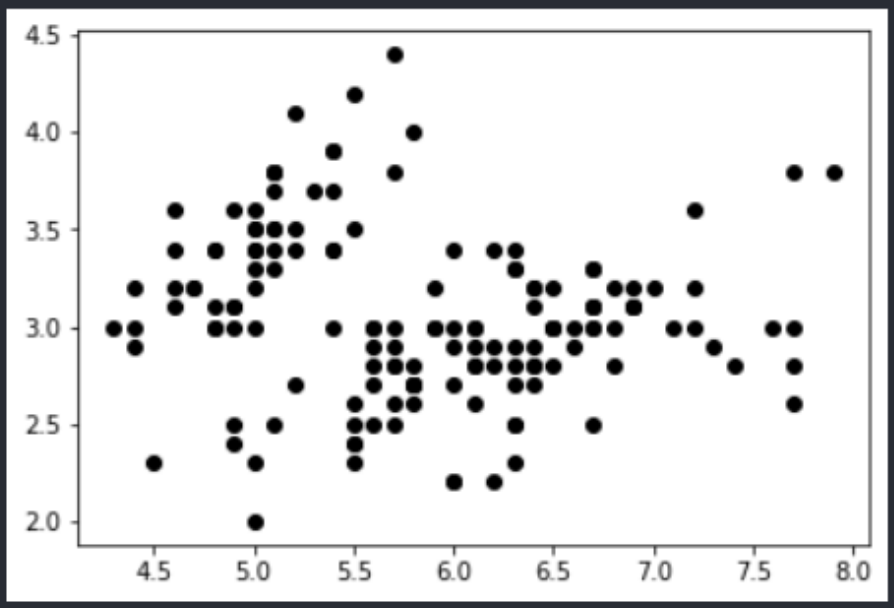
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Step 3 : Plotting the Points of the features in 2d for sepal

Scatter is a function that is used to plot the points available in the features set of sepals.

```
plt.scatter(X[:,0],X[:,1],c='black')
plt.show()
```

✓ 1.2s



Step 4 : The Algorithm

Kmeans algorithm works based on the following steps

- Select the required amount of K clusters
- Initialize the K centers (centroids)
- Assign each point to its nearest centroid, Calculate the Euclidian distance on every step with each centroid and the data points. Then assign the point to the centroid where the distance is minimum
- Compute the new centroid based on the distance of the updated centroid in that cluster (Maximization step)
- Repeat steps 3 and 4 until the positions of the centroid doesn't change.

Step 5 : Giving the value of K

After analyzing the data points, the best optimal value of K that can possibly be obtained is 3 . The value K= 3 is obtained using the elbow method which will be explained later in this section

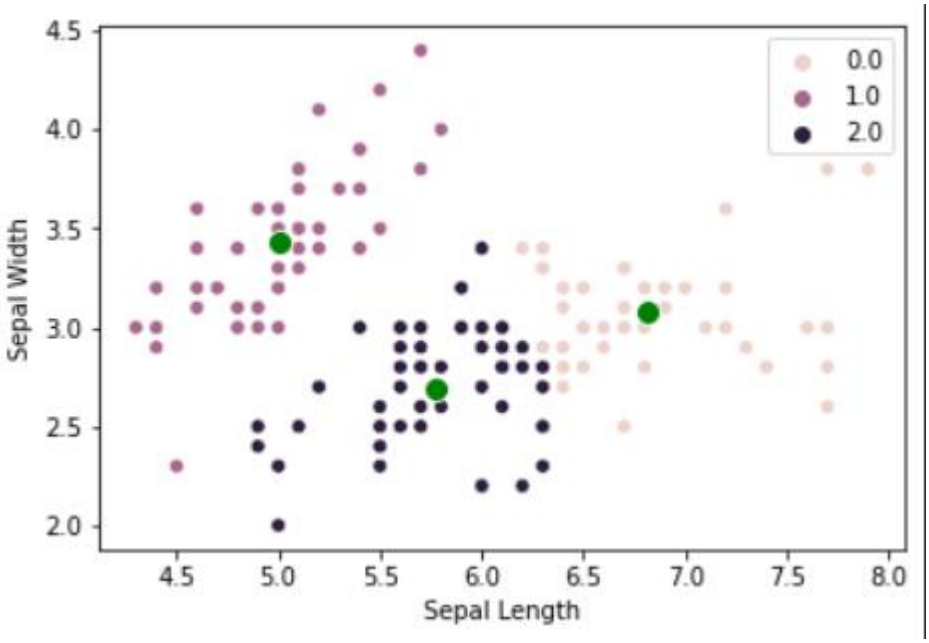
```
k = 3
centroids, cluster = kmeans(X,k)
```
✓ 0.7s

Step 6 : Plot the graph

Once the values are provided, the algorithm is modeled and a graph is plotted with the desired clusters and its centroids.

```
sns.scatterplot(X[:,0], X[:, 1], hue=cluster)
sns.scatterplot(centroids[:,0], centroids[:, 1], s=100, color='g')
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.show()
```
✓ 0.3s

Graph :



This K-means is calculated only with 2 features namely sepal length and sepal width, so the graph is on 2d plane. It can also be calculated with the features of petal with the same algorithm and code written. It gives different results and different clusters.

Analysis on K-Means algorithm :

- K-Means is relatively simpler than any machine learning algorithms
- It has a ability to scale larger datasets and items
- Using K-Means algorithm for datasets there is a convergence point for sure
- Clusters can be of any shape and size

Though K-Means has to be truly dependent on the initial value of K , which has to be chosen manually or any methods like elbow. There is a lot of chance that there will be outlier in this algorithm.

b. How to choose the value of K in K-Means ?

Explantion :

Value of K can be chosen randomly based on the assumptions, but it may or may not be correct. Choosing the wrong value will directly affect the performance.

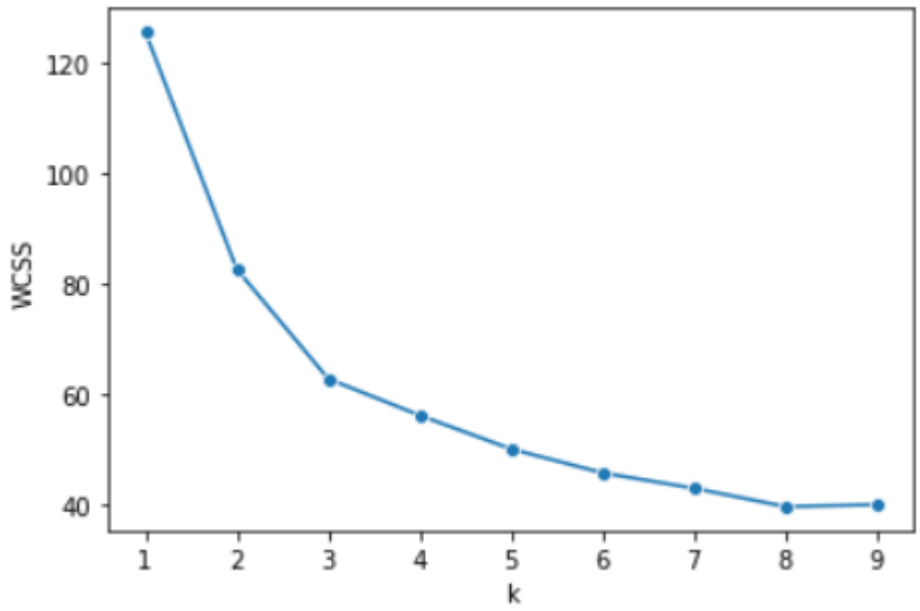
So, in general there are two ways in predicting the correct value for K

- Elbow method
- Silhouette method

Elbow is the most commonly used naïve approach for finding the value for K. It works by the basis of picking arrange of values and calculates the sum between the points and squaring them.

WCSS is the within cluster sum of square , where it is the sum of squared distance between the two points and the centroid of the cluster.

When the value of K increases the WCSS decreases. The wcss is always high when the K is equal to 1. With all the points, we plot a graph and It looks like this.

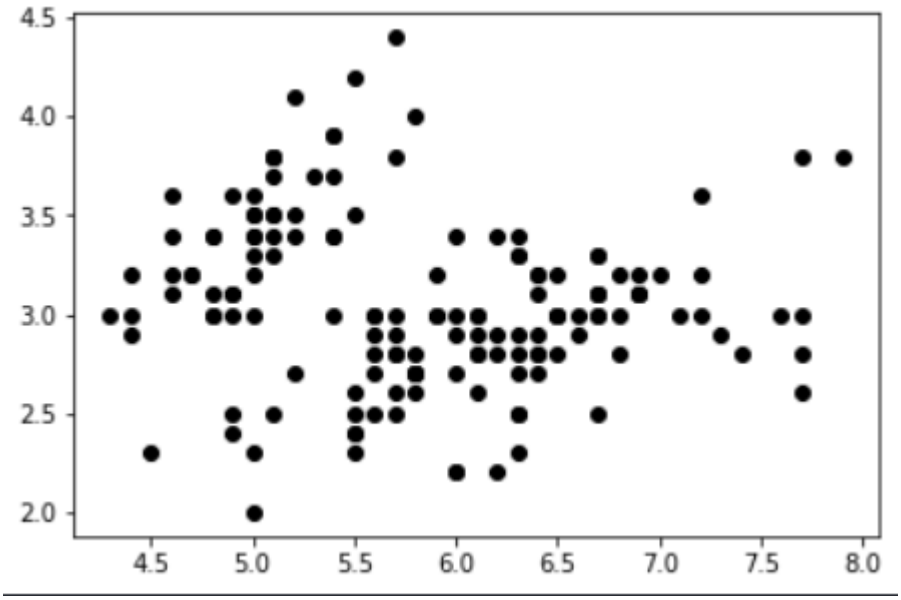


When we examine the graph carefully, we can see that graph gets decreased abruptly at some points. That point is literally considered to be a value of K. In other words K clusters.

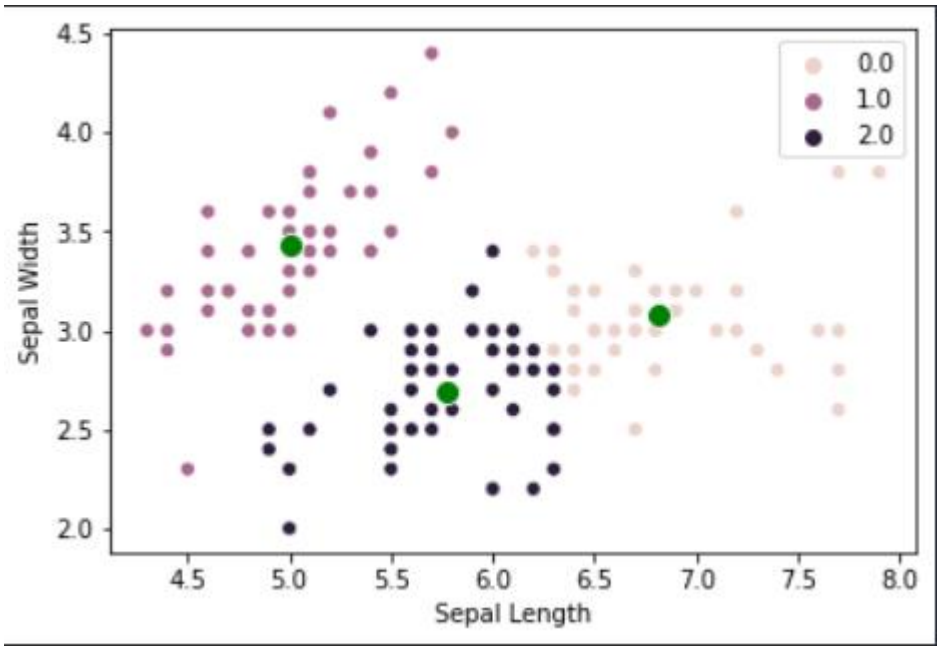
In our case **the point at which the graph decreases abruptly is 3**, hence the case.

K-Means for IRIS data set :

Before :



After :



Appendix :

All files can be access from the [github]