Data Mining Homework 3

Fabio Fraschetti 1834942

November 2023

[Fabio Fraschetti]

1 Exercise

1.1

Algorithm 1 Pseudocode k-means

```
centroid ← calculate the mean of random_list
cost ← calculate the sum of squared differences between each element
dict ← 'centroid': centroid, 'npoints': length of list, 'cost': cost, 'listpoints': list
listdict ← [dict]
function SEPARATE(listdict)
    for idx, d in listdict do
        if d[cost] > costomax and d[npoints] > 1 then
            costmax ← d[cost]
            idxmax ← idx
    pardict ← listdict[idxmax]
    Cs ← elements in pardict[listpoints] that are less or equal to pardict[centroid]
    Cd ← elements in pardict[listpoints] that are greater than pardict[centroid]
    centroidCs ← calculate the mean of Cs
    centroidCd ← calculate the mean of Cd
    costCs ← calculate the sum of squared differences for Cs
    costCd ← calculate the sum of squared differences for Cd
    dictCs ← 'centroid': centroidCs, 'npoints': length of Cs, 'cost': costCs, 'listpoints': Cs
    dictCd ← 'centroid': centroidCd, 'npoints': length of Cd, 'cost': costCd, 'listpoints': Cd
    listdict[idxmax] ← dictCs
    listdict insert dictCd after idxmax
    if length of listdict ≥ k then
        return listdict
    else
        return SEPARATE(listdict)
```

In this algorithm I first calculate the centroid of the given points and the cost and store it in a dictionary, this dictionary is the parameter of the function that first of all find the idx of the dictionary with greater cost and than on this cluster find the left and right elements of the centroid recalculate the centroid of this left part and right part and the cost. Then store it in other 2 dictionaries and append this to a copy of the original list and the last if check if the k clusters are created otherwise continue but now the parameter list is the one divided into 2 parts and so on. The complexity of this algorithm is $O(n^2k)$ because I need to check most of the elements 2 times. I added an implementation of this code in the folder called HW3EX1.py

1.2

We consider only the positive P^+ points because we can apply the same reasoning to the negative one. From P^+ points we calculate the mean μ and the maximum possible cost of this set is when half of the points are in 0 and the other half is in 2μ. This cost is $4\mu^2\frac{P^+}{2}$. Now we add the centroids l to μ and find the cost decrement that is $\mu^2\frac{P^+}{2}$. Then we find the ratio of final cost and initial cost that is

$$\frac{\frac{4\mu^2 P^+}{2} - \frac{\mu^2}{2}}{\frac{4\mu^2 P^+}{2}} = \frac{3}{4}$$

for the negative part we obtain the same result so the ratio between final and initial cost is at maximum $\frac{3}{4}$ and is less if we take fewer dispositions. Now if we iterate this process $O(\frac{1}{\epsilon}$ and so augment the number of clusters, we obtain the ratio between final and initial cost as small as we like and so make the k-means cost at most $\epsilon \cdot \sum_{x \in P} x^2$.

2 Exercise

I have developed a python code and saved the results in Performance_report. I saved the most important combinations of parameters, for every combination I saved the time dataset generation, k-means and PCA + K-means and the accuracy. In the first 3 tests I changed only the parameter k and as we can see the accuracy is high (100) because the sigma is low but with k=50 and k=100 it's not convenient use K-means+PCA because it takes more time then only k-means instead with k=200 it's convenient use PCA+K-means. In the second three tests I changed the parameter sigma to $\frac{1}{\sqrt{(k)}}$ and as we can see the accuracy is a little bit lower (97) because we add noise to the dataset and here is convenient use PCA+K-means with k=100 and k=200. In the next 3 tests I have augmented the sigma to 0.5 and here the accuracy is very low (02). while the behaviour of times from k-means and PCA+K-means is almost the same as before, but now the time taken by this part is grater then the dataset generation for the first time. In the last 3 tests I have changed the parameter n=10000, n=100000 and d=5000 where in the first 2 it's not convenient do the PCA while with d=5000 the PCA+K-means has a lower time and a better accuracy respect to only K-means. In the last test I tried the greatest dataset that my computer can generate and here we can see that the clustering without PCA is the most expensive part, with PCA the time decrees of 4 times and the accuracy is almost the same .In general we can say that with an high sigma and high dimension we have less accuracy and it's convenient use PCA.

3 Exercise

In this exercise I select from the amazon products the Price,Number of reviews and Stars. On the left part there are the cluster obtained from the original dataset divided into 3 clusters because of the figure (2) that represent the elbow method and there is a curvature in 3. In the right part of figure 1 there is the cluster divided always in 3 clusters for the same reason of before(check Figure(3)) as we can see the clusters generated are different because of the features engineering that I applied that are: the Standardization that involves transforming

the data such that it has a mean of 0 and a standard deviation of 1 and the second is the min-max scaling that is applied to bring the values within a specific range that I choose as Price into 15, Stars into 5, Number of Reviews into 9. I decided this values to give to each parameter the right weight that I want. In Figure 4 we can see the same graphic but with the normalize data. From this figure we can analyze better the result of the clusters, we can see that the points are divided in the right way because the results of the number of elements in the cluster pass from:
Cluster 1: 397
Cluster 2: 7
Cluster 3: 5
to:
Cluster 1: 48
Cluster 2: 224
Cluster 3: 137
that are much more spread, so my feature engineering techniques are useful and give a better result on the elbow method giving an higher curvature in 3. The time taken for KMeans without feature engineering is 0.22 seconds while the time taken for feature engineering + KMeans is 0.22 seconds that are the same, but it can change depending on the KMeans function. For the clustering I used KMeans function from sklearn that is the most common algorithm. I think that my features engineering technique are useful because allow the k-means algorithm to better cluster the data having in some way less noise, because of a better spread of data. I decided to keep only Price, Stars , Number of Reviews for a better understanding of the cluster that I have created, since adding the descriptions where there are always the same words repeated isn't a good estimator. If I had inserted the descriptions too I could have add 2 other features engineering techniques such as PCA and removes of stopwords. The program run and stop at every plot to see all the next plots you need to close the current plot and at the end there are some results such as the times and the cluster divisions that I saved into two files: clustered_dataFE and clustered_data

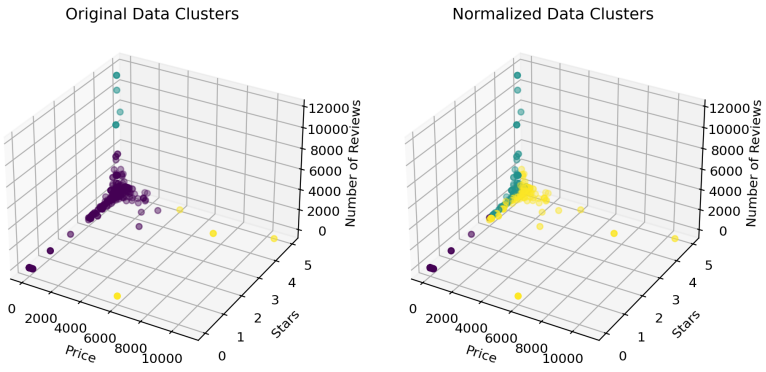

Figure (1)

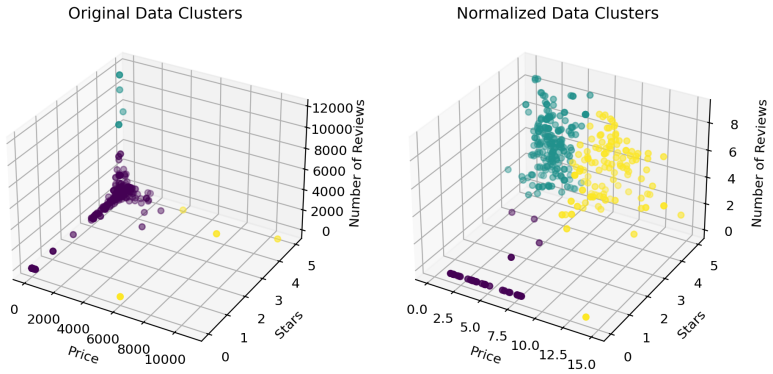

Figure (4)

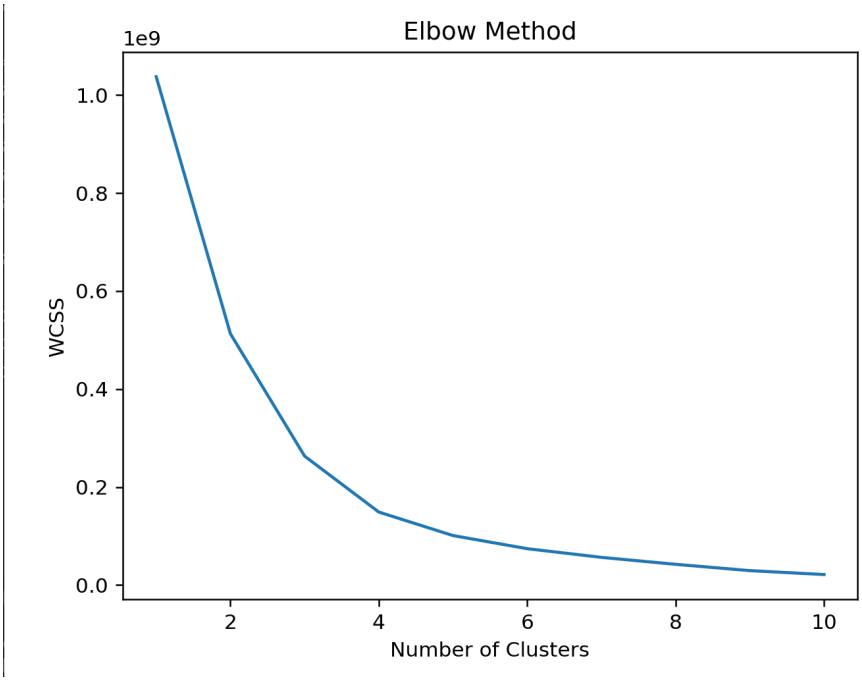

Figure (2)

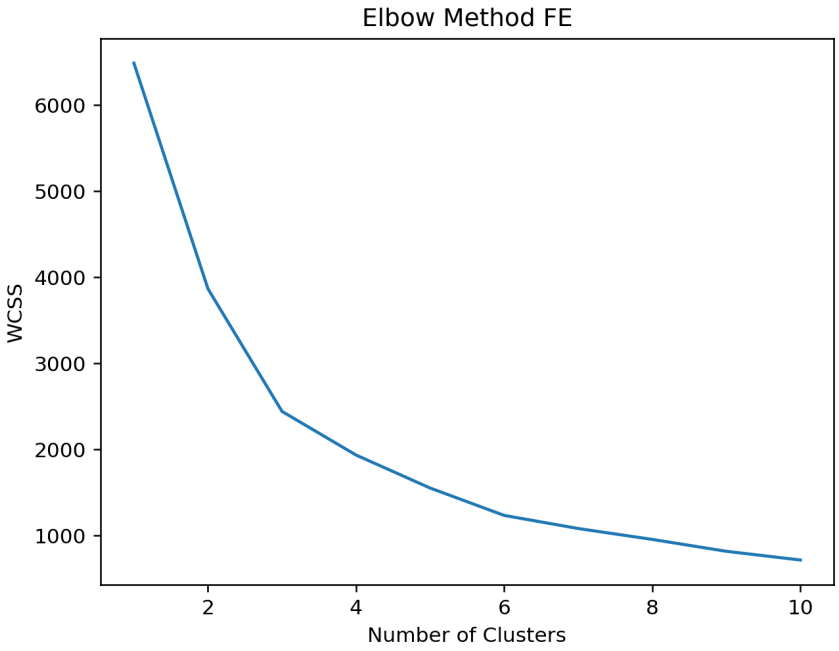
Elbow Method FE
WCSS
6000
5000
4000
3000
2000
1000
2
4
6
8
10
Number of Clusters

Figure (3)