Project 1 - Data Science with Python

Ali Nehrani

[a]*Ozyegin Universidad*

Resumen

Implementing 5 different clustering algorithms for clustering Iris and Boston data sets. For evaluating clustering performance use normalized mutual information score and the true labels of instances in the dataset.

Contents

1. Introduction

In this project I implemented at least 5 different clustering methods for classifying Iris and Boston Datasets.
In this report I summered the results by figures and tables. The complete codes are prepared in two python files: "Boston-data.py" for clustering Boston data and "Iris-data.py".

2. KMeans clustering

I applied KMeans clustering in both data sets with command sklearn.mixture.KMeans.
Normalized Mutual Info score for Boston dataset is: 0.38289
For Iris dataset, Normalized Mutual Info score is: 0.253333 and Metrics Accuracy Score is obtained as 0.794144
The complete form of KMeans in sklearn is as follows:
KMeans(n-clusters=8, init="k-means++", n-init=10, max-iter=300, tol=0.0001, precompute-distances="auto", verbose=0, random-state=None, copy-x=True, n-jobs=1, algorithm="auto")
where
n-clusters : number of clusters to form as well as the number of centroids to generate.

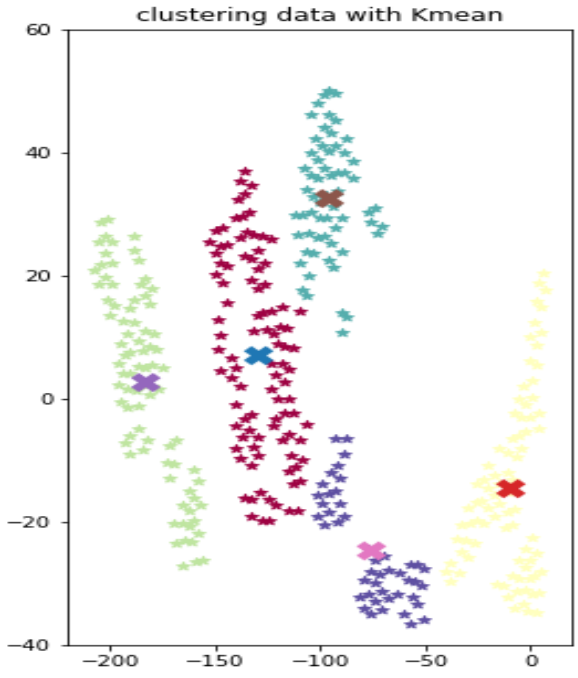

Figure 1: KMeans clustering on Boston data

cluster−centers− : Coordinates of cluster centers array, [n-clusters, n-features]
labels- : Labels of each point (Helps us in unsupervised clustering)
In the Figure 1 and Figure 2 The Results of clustering is presented for Boston and Iris dataset. Noting that in Figure 2 I compared feature selected data (left) with clustured data (right).

3. Gaussian Mixture clustering

Gaussian mixture model probability distribution is used in the class allows to estimate the parameters of a Gaussian mixture distribution. This is called by sklearn.mixture.GaussianMixture.
Important parameters in this method are number of components (**n-components**) I set to number of clusters.
Attributes are inclding; **weights-** (The weights of each mixture

Correo electrónico: (Ali Nehrani)

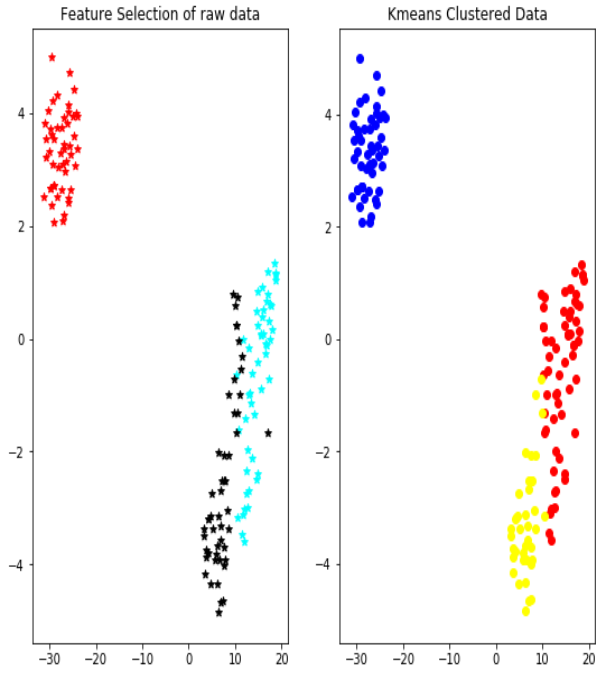

Figure 2: KMeans clustering on Iris data

components) and means- (The mean of each mixture component).
Normalized Mutual Info score for Boston dataset is: 0.379946
Normalized Mutual Info score for Iris dataset is: 0.233333
Metrics Accuracy Score for iris data set: 0.640453
In the Figure 3 and Figure 4 The Results of clustering is presented for Boston and Iris dataset (left- feature selected, right-clustered data).

4. Affinity Propagation Clustering

Affinity propagation (AP) is a clustering algorithm based on the concept of ”message passing” between data points. Unlike clustering algorithms such as k-means or k-medoids, affinity propagation does not require the number of clusters to be determined or estimated before running the algorithm. Similar to k-medoids, affinity propagation finds ”exemplars”, members of the input set that are representative of clusters [wikipedia].
The algorithm is called by sklearn.mixture.AffinityPropagation() with the following parameters: `(damping=0.5, max-iter=200, convergence-iter=15, copy=True, preference=None, affinity="euclidean", verbose=False)`
Attributes are inclding; **cluster-centers-indices-** :gives us the indices of cluster centers, **cluster-centers-** : gives the cluster centers (if affinity != precomputed). **labels-** : Labels of each point which can be used as prediction for the output.
Normalized Mutual Info score for Boston dataset is: 0.485508
Normalized Mutual Info score for Iris dataset is: 0.253333
Metrics Accuracy Score for Iris dataset is: 0.689496
In the Figure 5 and Figure 6 The Results of clustering is presented for Boston and Iris dataset (left- feature selected, right-clustered data).

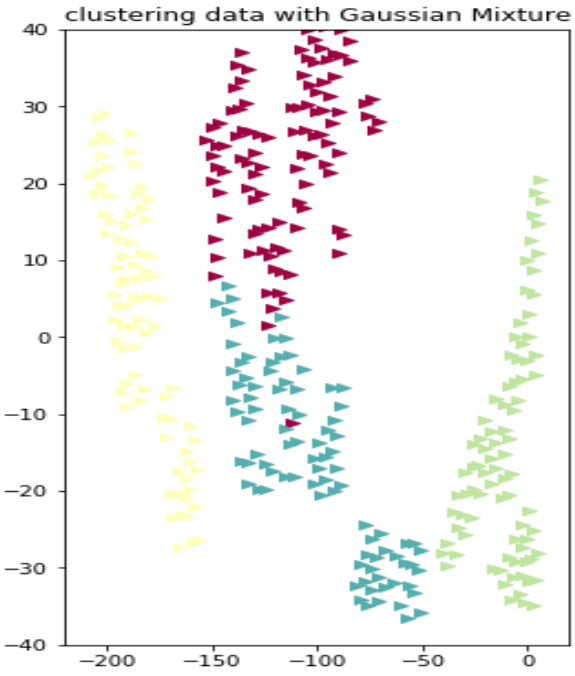

Figure 3: Gaussian Mixture clustering on Boston data

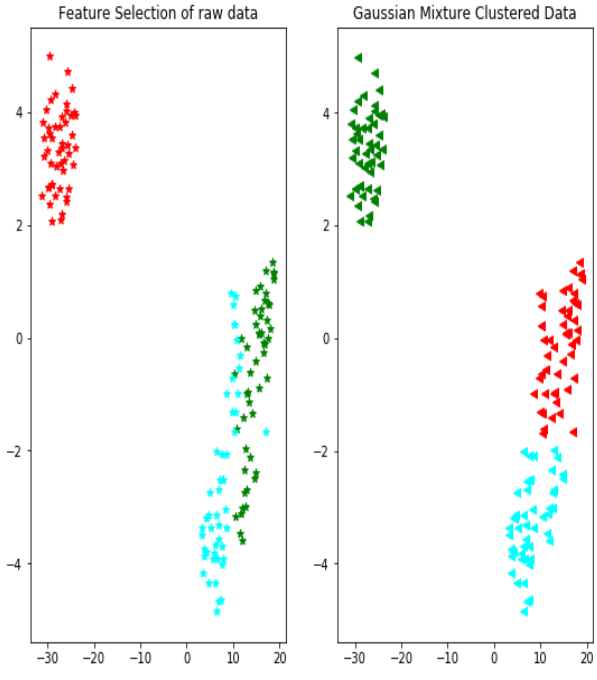

Figure 4: Gaussian Mixture clustering on Iris data

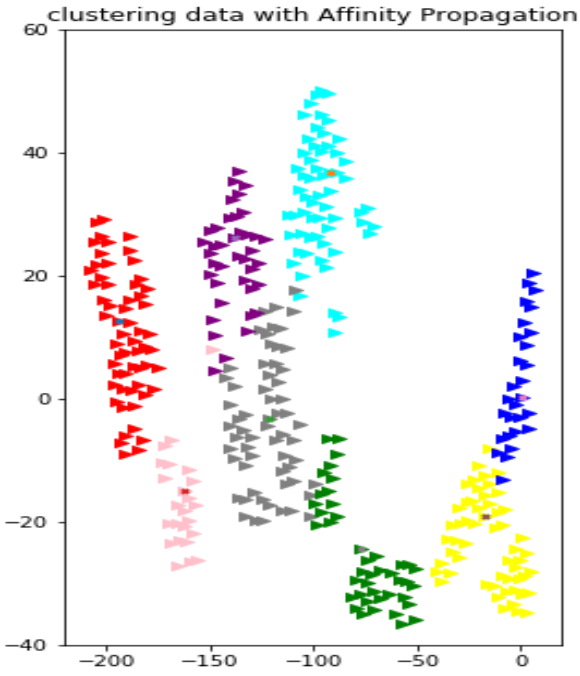

Figure 5: Affinity propagation clustering on Boston data

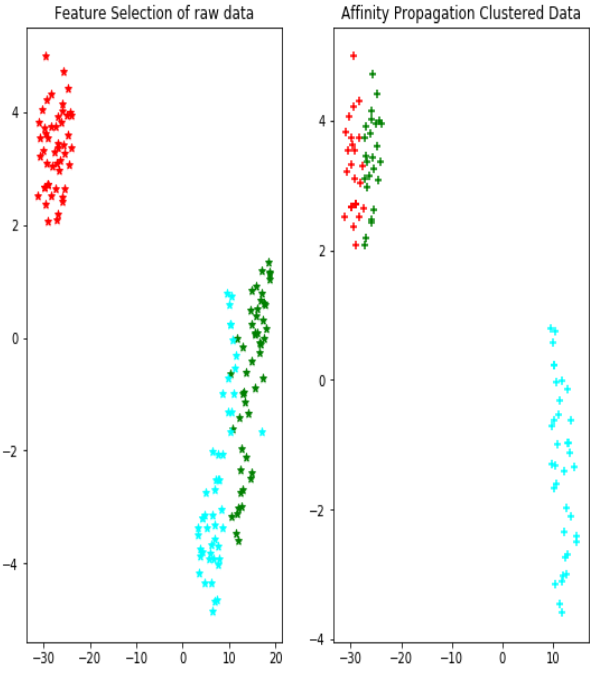

Figure 6: Affinity propagation clustering on Iris data

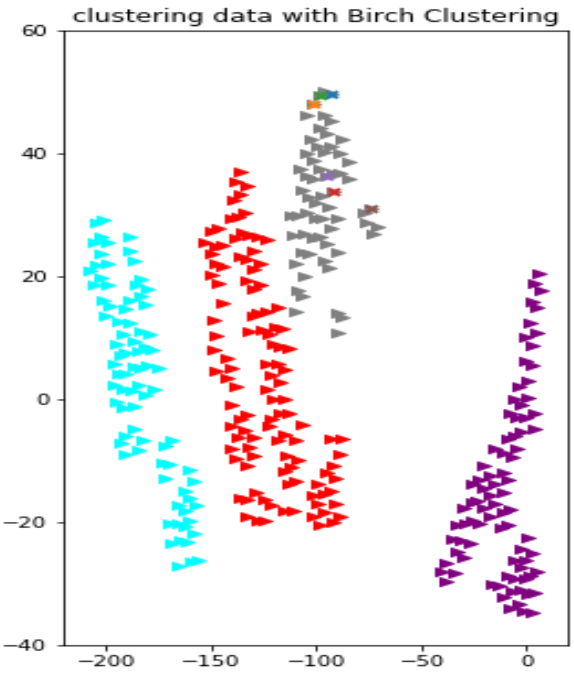

Figure 7: Birch clustering on Boston data

5. Birch clustering

It is a memory-efficient, online-learning algorithm provided as an alternative to Mini Batch KMeans. It constructs a tree data structure with the cluster centroids being read off the leaf. These can be either the final cluster centroids or can be provided as input to another clustering algorithm such as Agglomerative Clustering. The algorithm is called by sklearn.mixture.Birch() with the following parameters:
`(threshold=0.5, branching-factor=50, n-clusters=3, compute-labels=True, copy=True)`
The parameter ”n-clusters” determines number of clusters after the final clustering step, which treats the subclusters from the leaves as new samples.
Attributes are inclding; **subcluster-centers-** : Centroids of all subclusters read directly from the leaves and **subcluster-labels-** : Labels assigned to the centroids of the subclusters after they are clustered globally.
Normalized Mutual Info score for Boston dataset is: 0.379803
Normalized Mutual Info score for Iris dataset is: 0.24
Metrics Accuracy Score for Iris dataset is: 0.805754
In the Figure 7 and Figure 8 The Results of clustering is presented for Boston and Iris dataset (left- feature selected, right-clustered data).

6. Mean Shift Clustering

Mean shift clustering aims to discover “blobs” in a smooth density of samples. It is a centroid-based algorithm, which works by updating candidates for centroids to be the mean of the points within a given region. These candidates are then filtered in a post-processing stage to eliminate near-duplicates to

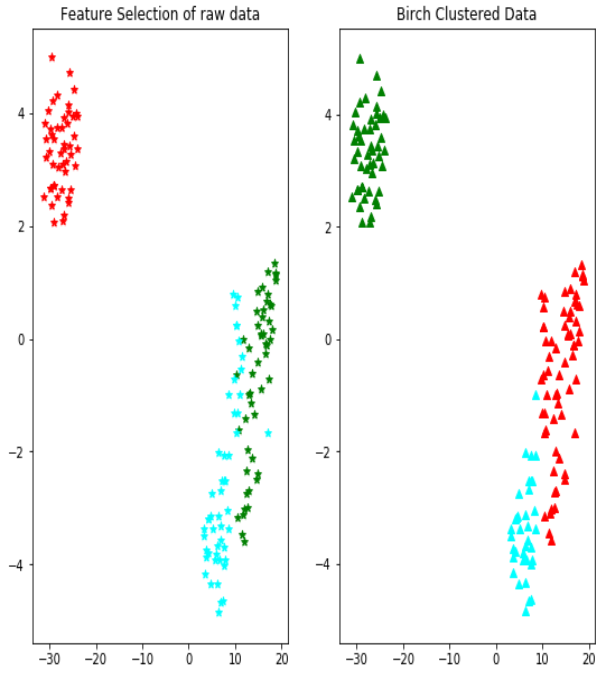

Figure 8: Birch clustering on Iris data

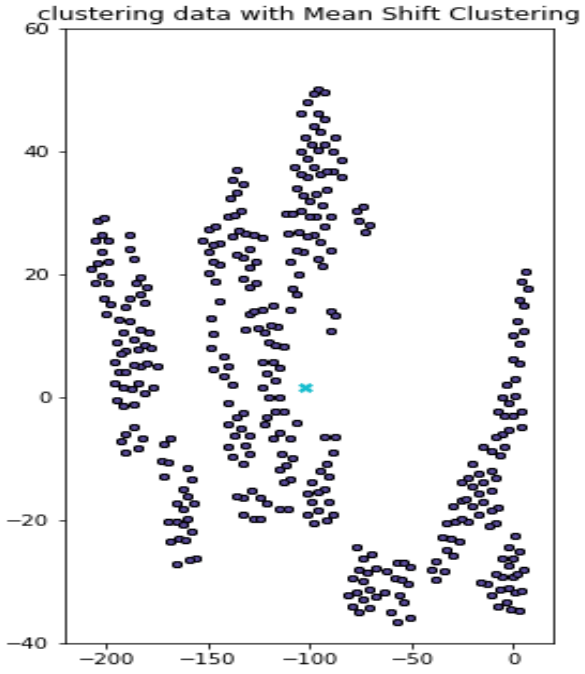

Figure 9: Mean Shift clustering on Boston data

form the final set of centroids.
The algorithm is called by sklearn.mixture.MeanShift() with the following parameters: `(bandwidth=None, seeds=None, bin-seeding=False, min-bin-freq=1, cluster-all=True, n-jobs=1)`
I used the default parameters.
Attributes are inclding; **cluster-centers-** : Coordinates of cluster centers in the form of [n-clusters, n-features]. **subcluster-labels-** : Labels of each point which is used as predicted target in Boston data case.
This clustering is only implemented for Boston data and results are presented in the Figure 9;
Normalized Mutual Info score for Boston dataset is: 0.242651

7. Agglomerative Clustering

It is a memory-efficient, online-learning algorithm provided as an alternative to Mini Batch KMeans. It constructs a tree data structure with the cluster centroids being read off the leaf. These can be either the final cluster centroids or can be provided as input to another clustering algorithm such as Agglomerative Clustering.

The algorithm is called by sklearn.mixture.AgglomerativeClustering() with the following parameters:
`(n-clusters=2, affinity="euclidean", memory=None, connectivity=None, compute-full-tree="auto", linkage="ward", pooling-func=<function mean>)`
The parameter **n-clusters** determines number of clusters that we want and the default is 2 (we set number of clusters to 7).
Attributes are inclding; **n-components-** : The estimated number of connected components in the graph and **labels-** : cluster labels for each point.
Normalized Mutual Info score for Boston dataset is: 0.379803
Normalized Mutual Info score for Iris dataset is: 0.000000
Metrics Accuracy Score for Iris dataset is: 0.76117
In the Figure 10 the Results of clustering is presented for Iris dataset.

8. Conclusion

I summerized the clustering results for both data sets in different tables. In Table-1 the Boston data case I computed NMI (Normalized Mutual Information) Score where the MAS (Metric Accuracy Score) was not computable in this case.

Table-1 Clustering results for Boston data set

No	Method Name	NMI Score
1	Kmeans clustering	0.382890
2	Gaussian Mixture Clustering	0.379032
3	Affinity Propagation Clustering	0.485508
4	Birch Clustering	0.379803
5	Mean Shift Clustering	0.242651
6	Agglomerative Clustering	0.379803

According to the NMI scores, Mean Shift method returned the minimum score although it is not good clustering method for this data set.
In Table 2, I represent the NMI and MAS results for Iris data set. We can see that NMI scores for different methods are close to eachother which is not good in comparing point of view. however MAS are higher but more clear for the clustering methods.

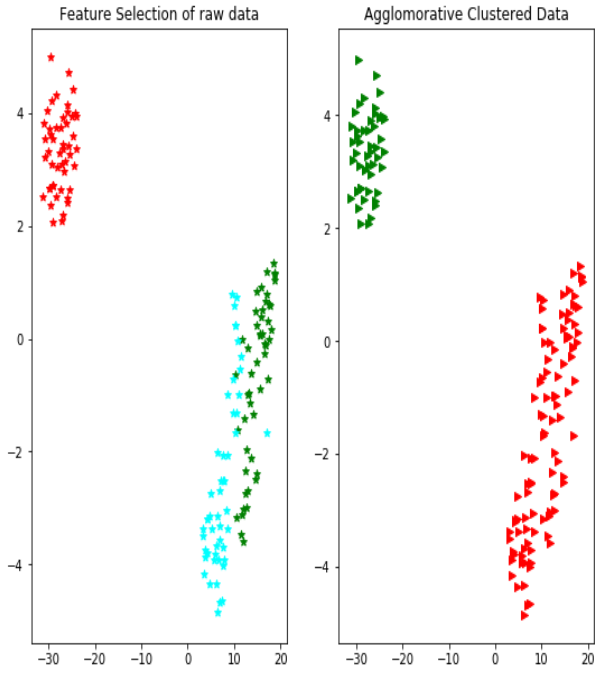

Figure 10: Agglomerative clustering on Iris data

Table 2- Results of NMIS and MAS For Iris data

No	Method Name	NMI Score	MAS
1	Kmeans clustering	0.253333	0.794144
2	Gaussian Mixture Clustering	0.233333	0.640453
3	Affinity Propagation Clustering	0.253333	0.689496
4	Birch Clustering	0.240000	0.805754
5	Agglomerative Clustering	0.00000	0.761170