2198-CSE-6363-001 Machine Learning

Project 3 Report

Note:

I have kept [my_path=r'C:\Users\Admin\Desktop\iris.data']

Please use appropriate path whilst running the ipython file o your machine. P.S as it is not object oriented, please execute all the kernels in sequence.

The given dataset is Iris (http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data). It is a program that performs statistical text classification. It is based on the Bow library.

Introduction:

Clustering is an unsupervised learning method that allows us to group set of objects based on similar characteristics. In general, it can help you find meaningful structure among your data, group similar data together and discover underlying patterns.

One of the most common clustering methods is K-means algorithm. The goal of this algorithm isto partition the data into set such that the total sum of squared distances from each point to the mean point of the cluster is minimized.

Data Implementation:

For the given project, I am going to use the Iris data set presented. This data consists of 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). It has four features from each sample: length and width of sepals and petals.

Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.

In other words, the K-means algorithm identifies *k* number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible.

The *'means'* in the K-means refers to averaging of the data; that is, finding the centroid.

Method:

K means works through the following iterative process:

1. Pick a value for k (the number of clusters to create)
2. Initialize k 'centroids' (starting points) in your data
3. Create your clusters. Assign each point to the nearest centroid.
4. Make your clusters better. Move each centroid to the center of its cluster.
5. Repeat steps 3–4 until your centroids converge.

Problems and strategies employed:

Here is a list of the main advantages and disadvantages of the algorithm implemented here.

Advantages:

- K-Means is simple and computationally efficient.
- It is very intuitive and their results are easy to visualize.

Disadvantages:

- K-Means is highly scale dependent and is not suitable for data of varying shapes and densities.
- Evaluating results is more subjective. It requires much more human evaluation than trusted metrics.

These aforementioned flaws are rectified by tweaking the N Cluster values.

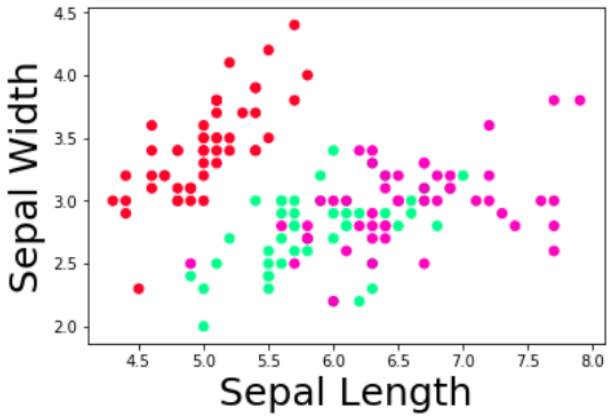
Conclusion:

The K-means algorithm is executed and the results are as follows,

Initial Plot:

```
In [40]: plt.scatter(X[:,0], X[:,1], c=y, cmap='gist_rainbow')
         plt.xlabel('Sepal Length', fontsize=25)
         plt.ylabel('Sepal Width', fontsize=25)
```

```
Out[40]: Text(0,0.5,'Sepal Width')
```



K-Means Label generations:

```
In [42]: # This is what KMeans thought
         km.labels_
```

```
Out[42]: array([1, 5, 5, 5, 1, 7, 5, 1, 5, 5, 7, 1, 5, 5, 7, 7, 7, 1, 7, 7, 1, 1,
                5, 1, 1, 5, 1, 1, 1, 5, 5, 1, 7, 7, 5, 5, 1, 5, 5, 1, 1, 5, 5, 1,
                7, 5, 7, 5, 7, 1, 4, 4, 4, 2, 4, 2, 4, 8, 4, 2, 8, 2, 2, 4, 2, 4,
                2, 2, 4, 2, 0, 2, 0, 4, 4, 4, 4, 4, 4, 8, 8, 8, 2, 0, 2, 4, 4, 4,
                2, 2, 2, 4, 2, 8, 2, 2, 2, 4, 8, 2, 6, 0, 3, 6, 6, 3, 2, 3, 6, 9,
                6, 0, 6, 0, 0, 6, 6, 9, 3, 0, 6, 0, 3, 0, 6, 3, 0, 0, 6, 3, 3, 9,
                6, 0, 0, 3, 6, 6, 0, 6, 6, 6, 0, 6, 6, 6, 0, 6, 6, 0])
```

Clustering:

```
In [41]: km = KMeans(n_clusters = 10, n_jobs = 4, random_state=21)
         km.fit(X)

Out[41]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
             n_clusters=10, n_init=10, n_jobs=4, precompute_distances='auto',
             random_state=21, tol=0.0001, verbose=0)
```

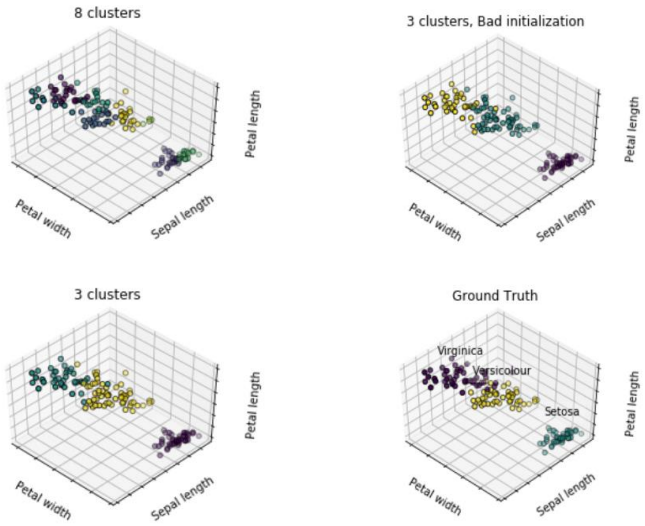
I have assigned 10 as the value for n clusters 4, **n** is the number of d-dimensional vectors (to be clustered) **k** the number of **clusters**. i the number of iterations needed until convergence. I have done so get the curve to fit all the vector parameters.

Cluster Centroid Values:

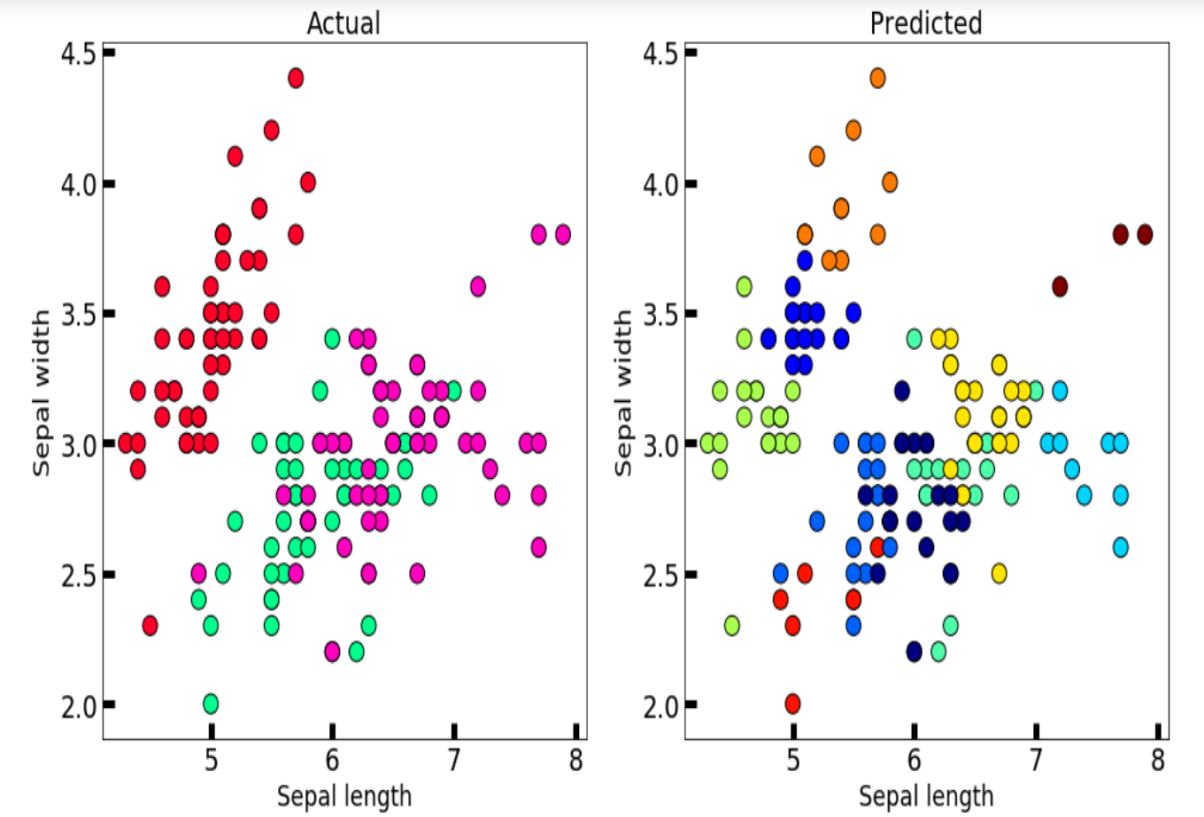
```
In [43]: centers = km.cluster_centers_
         print(centers)

[[6.02777778 2.73333333 5.02777778 1.79444444]
 [5.1        3.45       1.51111111 0.28333333]
 [5.62857143 2.72380952 4.13333333 1.2952381 ]
 [7.43333333 2.92222222 6.26666667 1.98888889]
 [6.43       2.94       4.59       1.435     ]
 [4.69       3.085      1.385      0.19      ]
 [6.56818182 3.08636364 5.53636364 2.16363636]
 [5.39166667 3.925      1.525      0.275     ]
 [5.24285714 2.37142857 3.44285714 1.02857143]
 [7.6        3.73333333 6.4        2.23333333]]
```

3D Representation:



Final Clustering:



K-means clustering is an extensively used technique for data cluster analysis.

However, its performance is usually not as competitive as those of the other sophisticated clustering techniques because slight variations in the data could lead to high variance.

Furthermore, clusters are assumed to be spherical and evenly sized, something which may reduce the accuracy of the K-means clustering Python results.