01-K Means Clustering with Python

May 16, 2023

1 K Means Clustering with Python

This notebook is just a code reference for the video lecture and reading.

1.1 Method Used

K Means Clustering is an unsupervised learning algorithm that tries to cluster data based on their similarity. Unsupervised learning means that there is no outcome to be predicted, and the algorithm just tries to find patterns in the data. In k means clustering, we have the specify the number of clusters we want the data to be grouped into. The algorithm randomly assigns each observation to a cluster, and finds the centroid of each cluster. Then, the algorithm iterates through two steps: Reassign data points to the cluster whose centroid is closest. Calculate new centroid of each cluster. These two steps are repeated till the within cluster variation cannot be reduced any further. The within cluster variation is calculated as the sum of the euclidean distance between the data points and their respective cluster centroids.

1.2 Import Libraries

```python
[1]: import seaborn as sns
     import matplotlib.pyplot as plt
     %matplotlib inline
```

1.3 Create some Data

```python
[2]: from sklearn.datasets import make_blobs
```
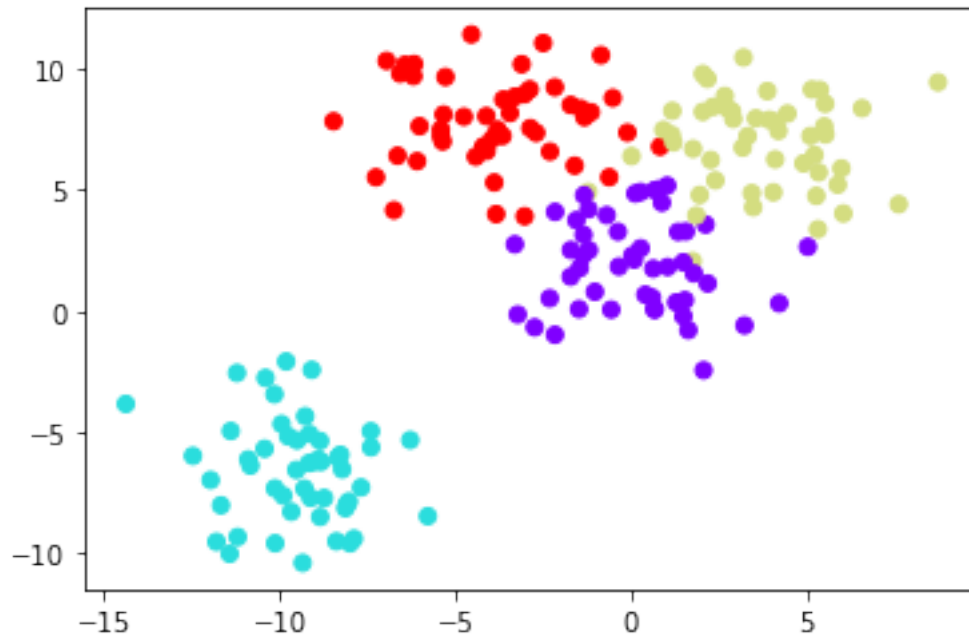
```python
[3]: # Create Data
     data = make_blobs(n_samples=200, n_features=2,
                       centers=4, cluster_std=1.8,random_state=101)
```

1.4 Visualize Data

```python
[4]: plt.scatter(data[0][:,0],data[0][:,1],c=data[1],cmap='rainbow')
```

```
[4]: <matplotlib.collections.PathCollection at 0x7f30bb3114d0>
```

1.5 Creating the Clusters

```
[5]: from sklearn.cluster import KMeans
```

```
[6]: kmeans = KMeans(n_clusters=4)
```

```
[7]: kmeans.fit(data[0])
```

```
[7]: KMeans(n_clusters=4)
```

```
[8]: kmeans.cluster_centers_
```

```
[8]: array([[-9.46941837, -6.56081545],
            [-0.0123077 ,  2.13407664],
            [-4.13591321,  7.95389851],
            [ 3.71749226,  7.01388735]])
```

```
[9]: kmeans.labels_
```

```
[9]: array([2, 3, 1, 3, 3, 0, 3, 1, 3, 1, 2, 1, 3, 3, 2, 1, 3, 1, 0, 2, 0, 1,
            1, 0, 2, 0, 0, 1, 3, 3, 2, 0, 3, 1, 1, 2, 0, 0, 0, 1, 0, 2, 2, 2,
            1, 3, 2, 1, 0, 1, 1, 2, 3, 1, 0, 2, 1, 1, 2, 3, 0, 3, 0, 2, 3, 1,
            0, 3, 3, 0, 3, 1, 0, 1, 0, 3, 3, 1, 2, 1, 1, 0, 3, 0, 1, 1, 1, 2,
            1, 0, 0, 0, 0, 1, 1, 0, 3, 2, 0, 3, 1, 0, 1, 1, 3, 1, 0, 3, 0, 0,
            3, 2, 2, 3, 0, 3, 2, 2, 3, 2, 1, 2, 1, 2, 1, 3, 2, 1, 0, 2, 2, 2,
```

```
         1, 0, 0, 2, 3, 2, 3, 1, 0, 3, 0, 2, 2, 3, 1, 0, 2, 2, 2, 2, 1, 3,
         1, 2, 3, 3, 3, 1, 3, 1, 1, 2, 0, 2, 1, 3, 2, 1, 3, 1, 2, 3, 1, 2,
         3, 3, 0, 3, 2, 0, 0, 2, 0, 0, 0, 0, 0, 1, 0, 3, 3, 2, 0, 1, 3, 3,
         0, 1], dtype=int32)
```

[10]:
```python
f, (ax1, ax2) = plt.subplots(1, 2, sharey=True,figsize=(10,6))
ax1.set_title('K Means')
ax1.scatter(data[0][:,0],data[0][:,1],c=kmeans.labels_,cmap='rainbow')
ax2.set_title("Original")
ax2.scatter(data[0][:,0],data[0][:,1],c=data[1],cmap='rainbow')
```

[10]: <matplotlib.collections.PathCollection at 0x7f30b9bf9a90>
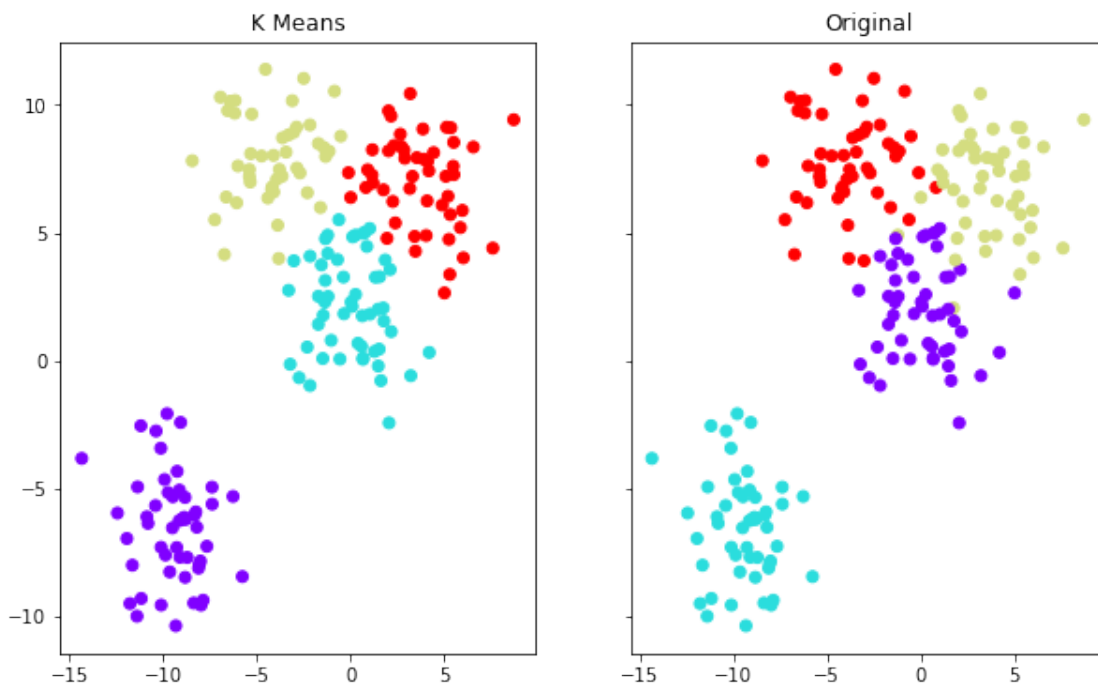
You should note, the colors are meaningless in reference between the two plots.