k-means

January 5, 2023

1 K-means

-
  -
  - -
-
- 2
  - ( )
  -

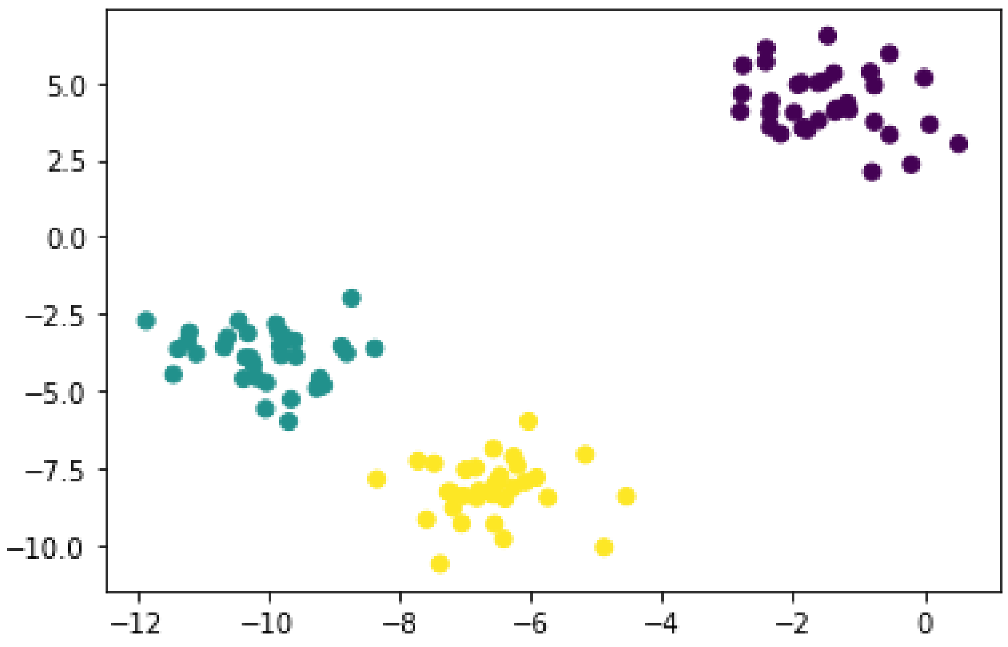
[87]:
```
from sklearn.datasets import make_blobs, make_moons
from sklearn.cluster import KMeans, AgglomerativeClustering, DBSCAN
from sklearn.metrics import pairwise_distances
from sklearn.metrics.cluster import adjusted_rand_score
from sklearn.metrics import silhouette_score
import numpy as np

import matplotlib.pyplot as plt
```

[11]:
```
# 3
X, y = make_blobs(random_state=1)
```

[12]:
```
plt.scatter(X[:, 0], X[:, 1], c=y)
```

[12]:
```
<matplotlib.collections.PathCollection at 0x7fcec07ef820>
```

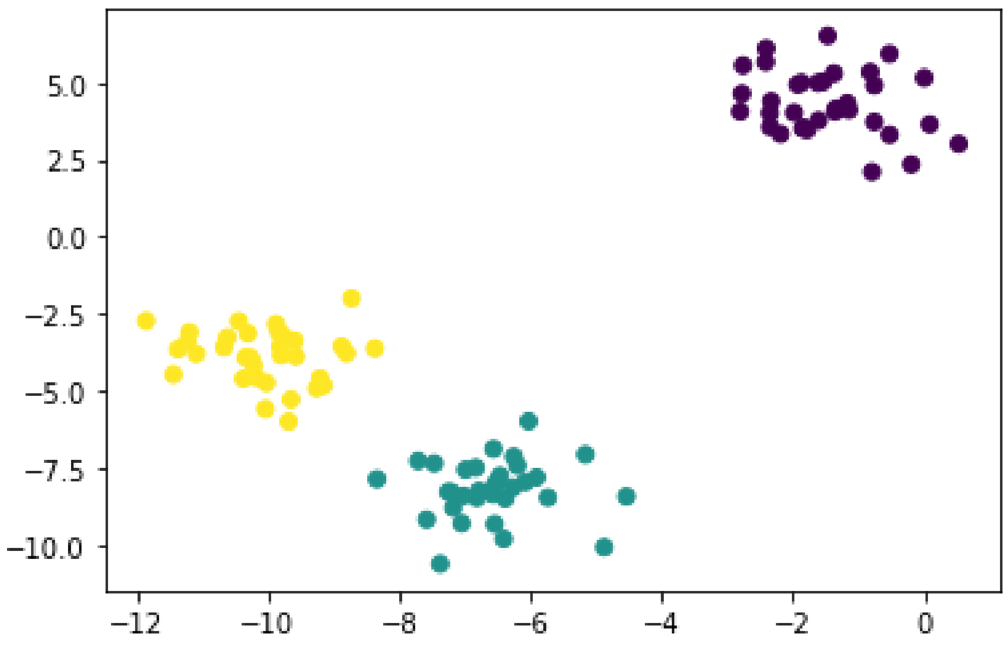


```
[13]: kmeans = KMeans(n_clusters=3)
      kmeans.fit(X)
      kmeans.labels_
```

```
[13]: array([0, 2, 2, 2, 1, 1, 1, 2, 0, 0, 2, 2, 1, 0, 1, 1, 1, 0, 2, 2, 1, 2,
             1, 0, 2, 1, 1, 0, 0, 1, 0, 0, 1, 0, 2, 1, 2, 2, 2, 1, 1, 2, 0, 2,
             2, 1, 0, 0, 0, 0, 2, 1, 1, 1, 0, 1, 2, 2, 0, 0, 2, 1, 1, 2, 2, 1,
             0, 1, 0, 2, 2, 2, 1, 0, 0, 2, 1, 1, 0, 2, 0, 2, 2, 1, 0, 0, 0, 0,
             2, 0, 1, 0, 0, 2, 2, 1, 1, 0, 1, 0], dtype=int32)
```

```
[14]: plt.scatter(X[:, 0], X[:, 1], c=kmeans.labels_)
```

```
[14]: <matplotlib.collections.PathCollection at 0x7fce307a1370>
```



[15]:
```
y
```

[15]:
```
array([0, 1, 1, 1, 2, 2, 2, 1, 0, 0, 1, 1, 2, 0, 2, 2, 2, 0, 1, 1, 2, 1,
       2, 0, 1, 2, 2, 0, 0, 2, 0, 0, 2, 0, 1, 2, 1, 1, 1, 2, 2, 1, 0, 1,
       1, 2, 0, 0, 0, 0, 1, 2, 2, 2, 0, 2, 1, 1, 0, 0, 1, 2, 2, 1, 1, 2,
       0, 2, 0, 1, 1, 1, 2, 0, 0, 1, 2, 2, 0, 1, 0, 1, 1, 2, 0, 0, 0, 0,
       1, 0, 2, 0, 0, 1, 1, 2, 2, 0, 2, 0])
```

[16]:
```
kmeans.labels_ == y
```

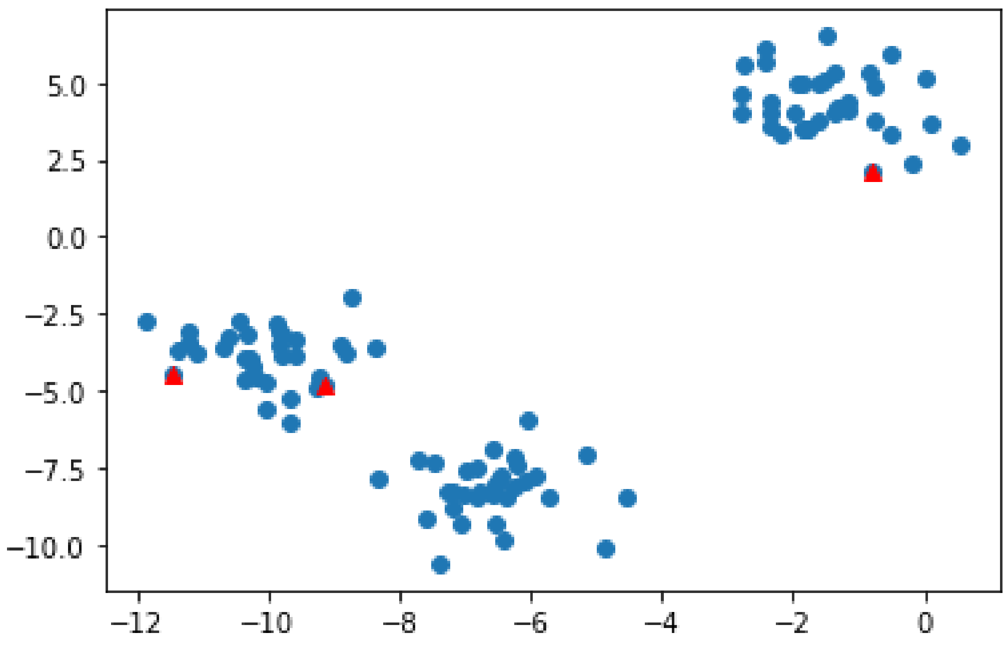
[16]:
```
array([ True, False, False, False, False, False, False, False,  True,
        True, False, False, False,  True, False, False, False,  True,
       False, False, False, False, False,  True, False, False, False,
        True,  True, False,  True,  True, False,  True, False, False,
       False, False, False, False, False, False,  True, False, False,
       False,  True,  True,  True,  True, False, False, False, False,
        True, False, False, False,  True,  True, False, False, False,
       False, False, False,  True, False,  True, False, False, False,
       False,  True,  True, False, False, False,  True, False,  True,
       False, False, False,  True,  True,  True,  True, False,  True,
       False,  True,  True, False, False, False, False,  True, False,
        True])
```

[ ]:

2

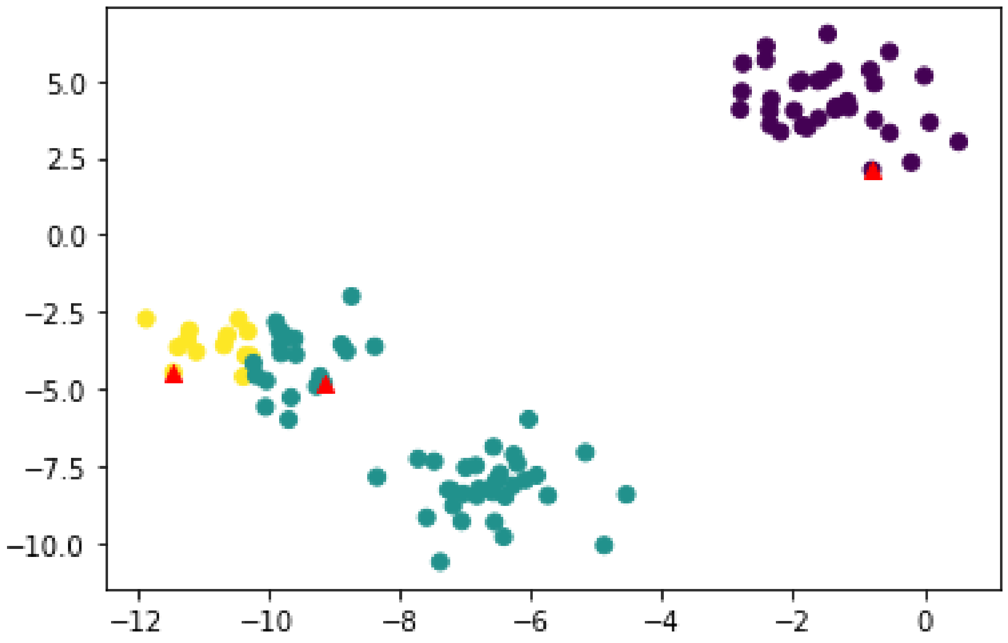
[17]:
```
#
X, y = make_blobs(random_state=1)
center = X[:3, :]
#
plt.scatter(X[:, 0], X[:, 1])
plt.scatter(center[:, 0], center[:, 1], marker='^', color = "red")
```

[17]:
```
<matplotlib.collections.PathCollection at 0x7fce70b9dee0>
```



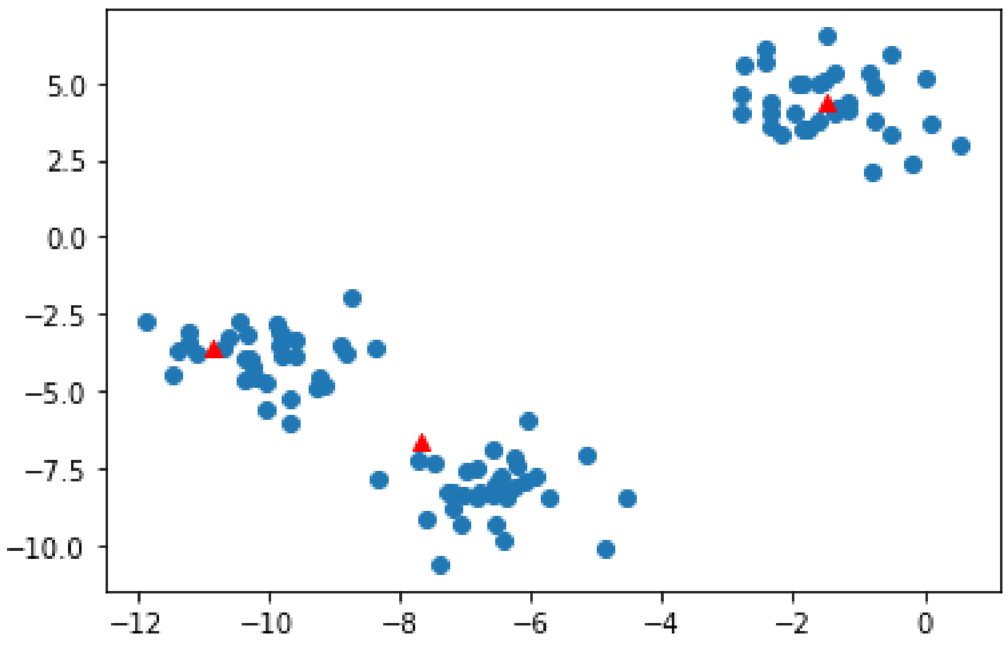
[18]:
```
#
labels = np.argmin(pairwise_distances(center, X), axis=0)
plt.scatter(X[:, 0], X[:, 1], c = labels)
plt.scatter(center[:, 0], center[:, 1], marker='^', color = "red")
```

[18]:
```
<matplotlib.collections.PathCollection at 0x7fce88d8e130>
```



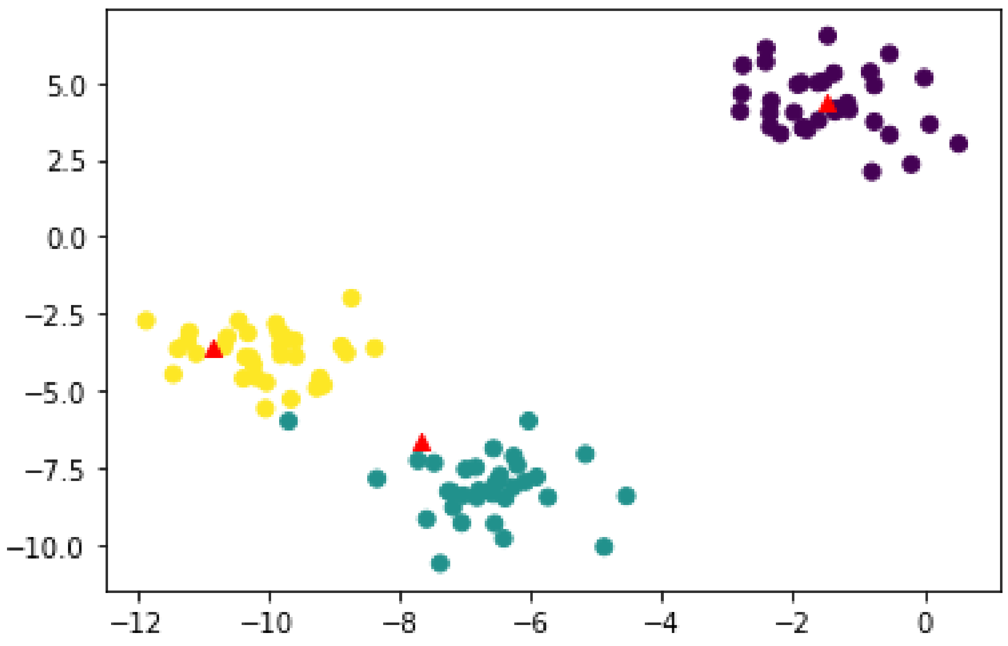
```
[19]: #
      center2 = np.array([X[labels==0].mean(0), X[labels==1].mean(0), X[labels==2].
       ↪mean(0)])
      plt.scatter(X[:, 0], X[:, 1])
      plt.scatter(center2[:, 0], center2[:, 1], marker='^', color = "red")
```

```
[19]: <matplotlib.collections.PathCollection at 0x7fce50540ac0>
```



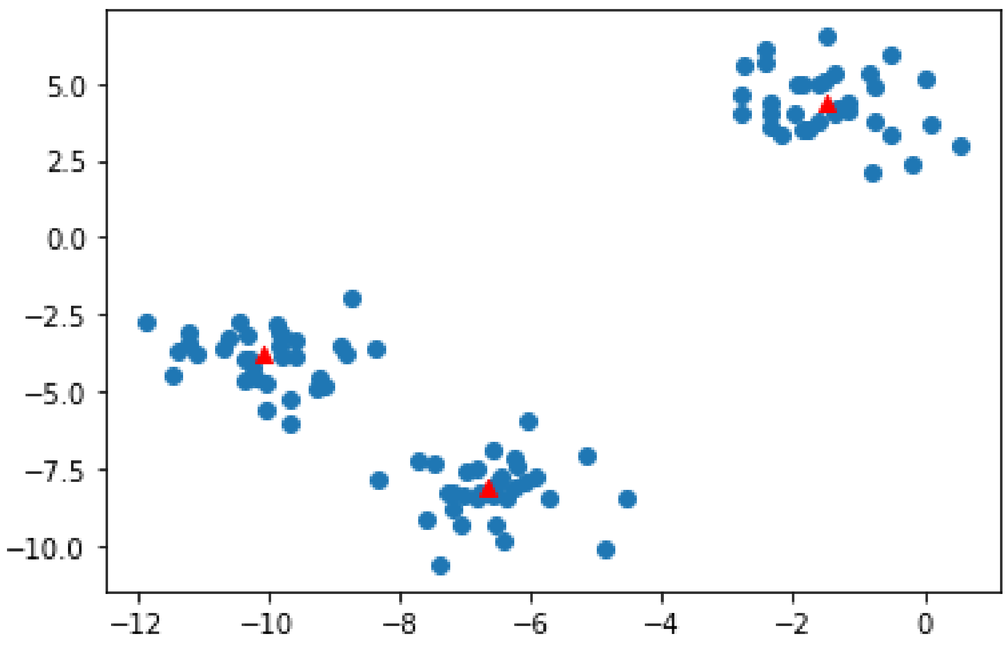
```
[20]: #
      labels = np.argmin(pairwise_distances(center2, X), axis=0)
      plt.scatter(X[:, 0], X[:, 1], c = labels)
      plt.scatter(center2[:, 0], center2[:, 1], marker='^', color = "red")
```

```
[20]: <matplotlib.collections.PathCollection at 0x7fce88db9ac0>
```



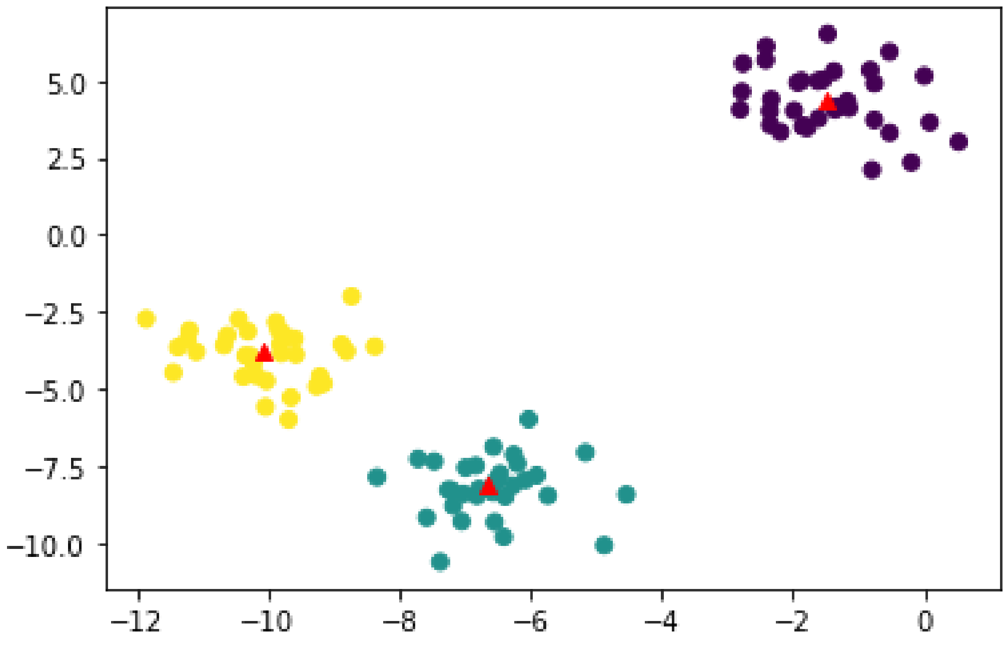
[21]:
```
#
center3 = np.array([X[labels==0].mean(0), X[labels==1].mean(0), X[labels==2].
 ↪mean(0)])
plt.scatter(X[:, 0], X[:, 1])
plt.scatter(center3[:, 0], center3[:, 1], marker='^', color = "red")
```

[21]:
```
<matplotlib.collections.PathCollection at 0x7fce88efb0d0>
```



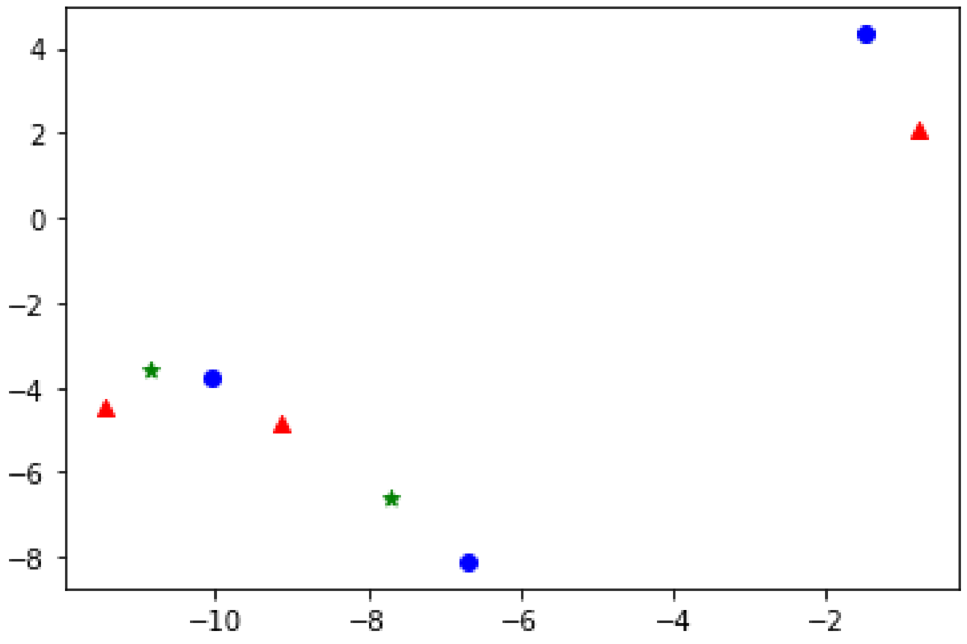
[22]:
```
#
labels = np.argmin(pairwise_distances(center3, X), axis=0)
plt.scatter(X[:, 0], X[:, 1], c = labels)
plt.scatter(center3[:, 0], center3[:, 1], marker='^', color = "red")
```

[22]:
```
<matplotlib.collections.PathCollection at 0x7fcea090f730>
```



[23]:
```
#
plt.scatter(center[:, 0], center[:, 1], marker='^', color = "red")
plt.scatter(center2[:, 0], center2[:, 1], marker='*', color = "green")
plt.scatter(center3[:, 0], center3[:, 1], marker='o', color = "blue")
```

[23]:
```
<matplotlib.collections.PathCollection at 0x7fcec08eab20>
```



2.1

-
    - ARI:
    - NMI:
-
    -
    - 0-1

2.2 kmeans

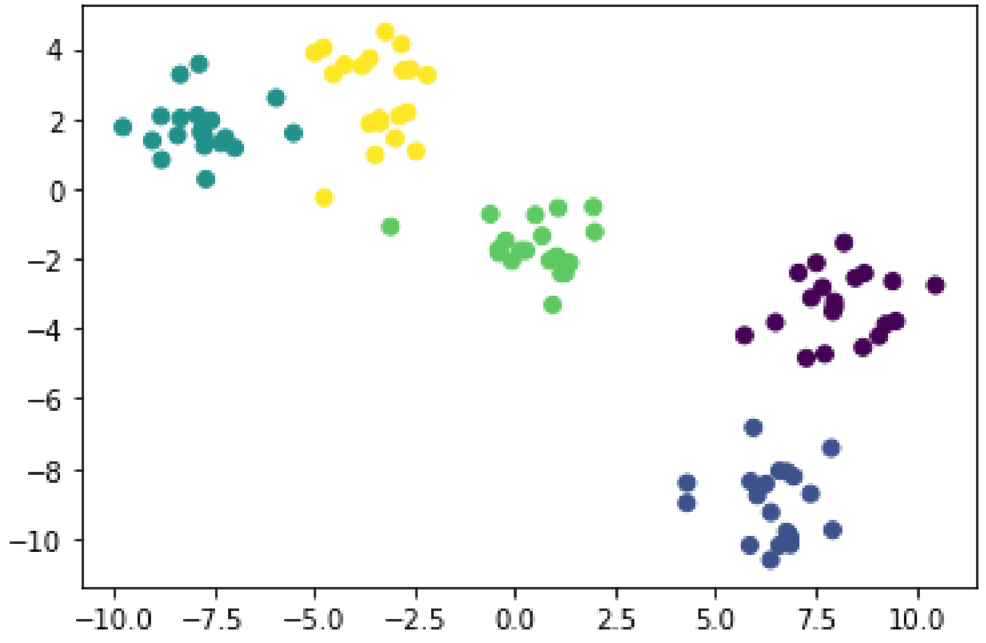
-
    -
-

2.2.1

-
-
    -
    -
    - ( )

[45]:
```
# 5
X, y = make_blobs(random_state=6, centers=5)
plt.scatter(X[:,0], X[:,1], c=y)
```

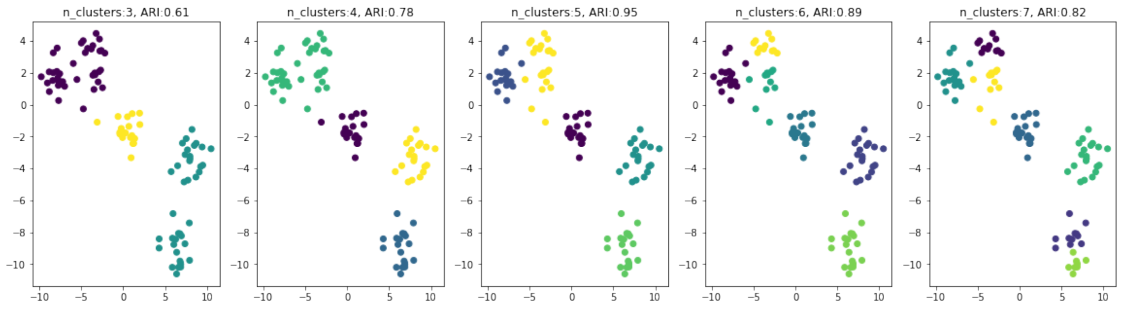
[45]:
```
<matplotlib.collections.PathCollection at 0x7fce90d48160>
```



```
[48]: kmeans = KMeans(n_clusters=3)
      kmeans.fit(X)
      silhouette_score(X, kmeans.labels_)
```

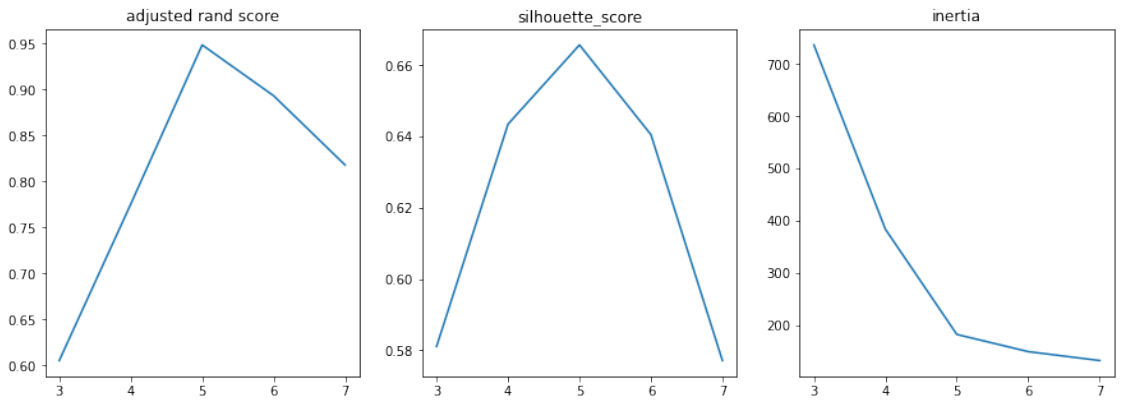
```
[48]: 0.5810848635319777
```

```
[57]: # 3-7          ARI
      #    : ARI
      fig, axes = plt.subplots(1, 5, figsize=(20, 5))
      aris, sils, inrs = [], [], []
      for i, nc in enumerate(range(3, 8)):
          kmeans = KMeans(n_clusters=nc)
          kmeans.fit(X)
          aris.append(adjusted_rand_score(y, kmeans.labels_))
          sils.append(silhouette_score(X, kmeans.labels_))
          inrs.append(kmeans.inertia_)
          axes[i].scatter(X[:,0], X[:,1], c=kmeans.labels_)
          axes[i].set_title("n_clusters:{}, ARI:{:.2f}".format(nc, aris[-1]))
```



[59]:
```
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
axes[0].plot(range(3, 8), aris)
axes[0].set_title("adjusted rand score")
axes[1].plot(range(3, 8), sils)
axes[1].set_title("silhouette_score")
axes[2].plot(range(3, 8), inrs)
axes[2].set_title("inertia")
```

[59]:
```
Text(0.5, 1.0, 'inertia')
```

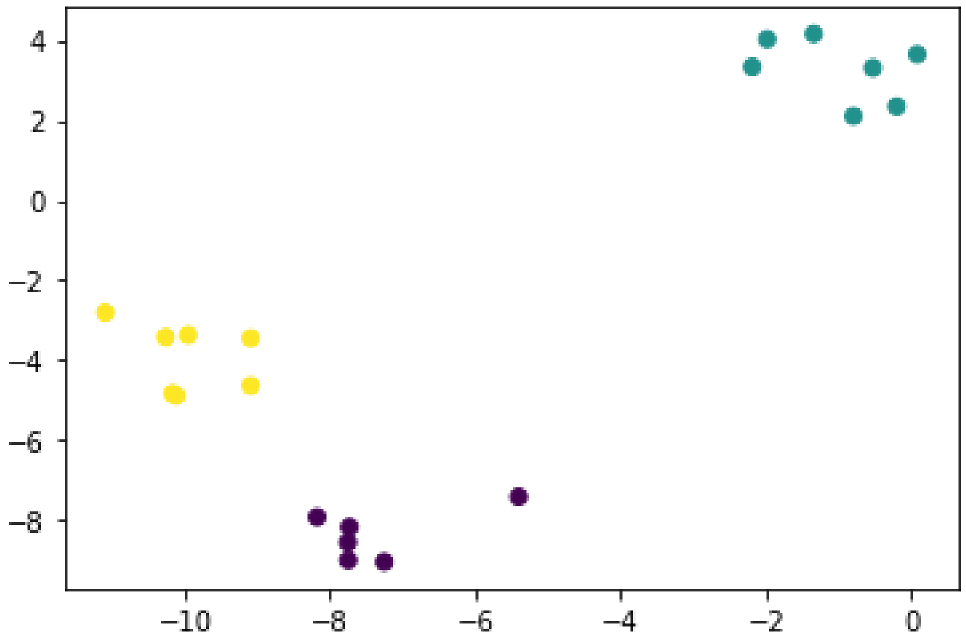


2.3

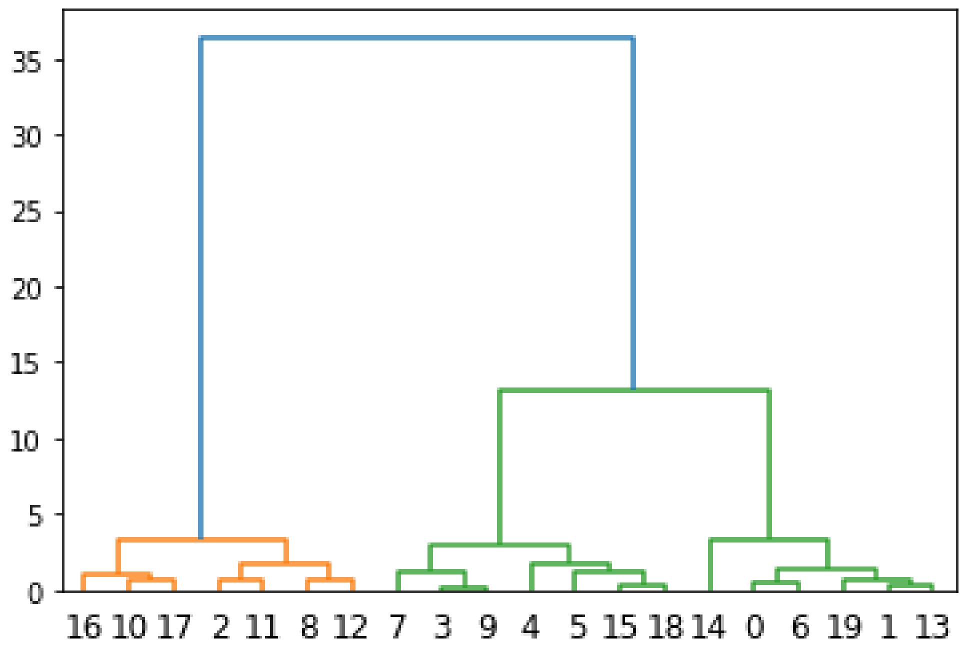
- 1 1
- 2
-

[78]:
```
X, y = make_blobs(random_state=1, n_samples=20)
agg = AgglomerativeClustering(n_clusters=3)
agg.fit_predict(X)
plt.scatter(X[:,0], X[:,1], c=agg.labels_)
```

```
[78]: <matplotlib.collections.PathCollection at 0x7fce8912f5b0>
```



```
[82]: #
      from scipy.cluster.hierarchy import dendrogram, ward
      linkage_array = ward(X)
      dendrogram(linkage_array)
      None
```

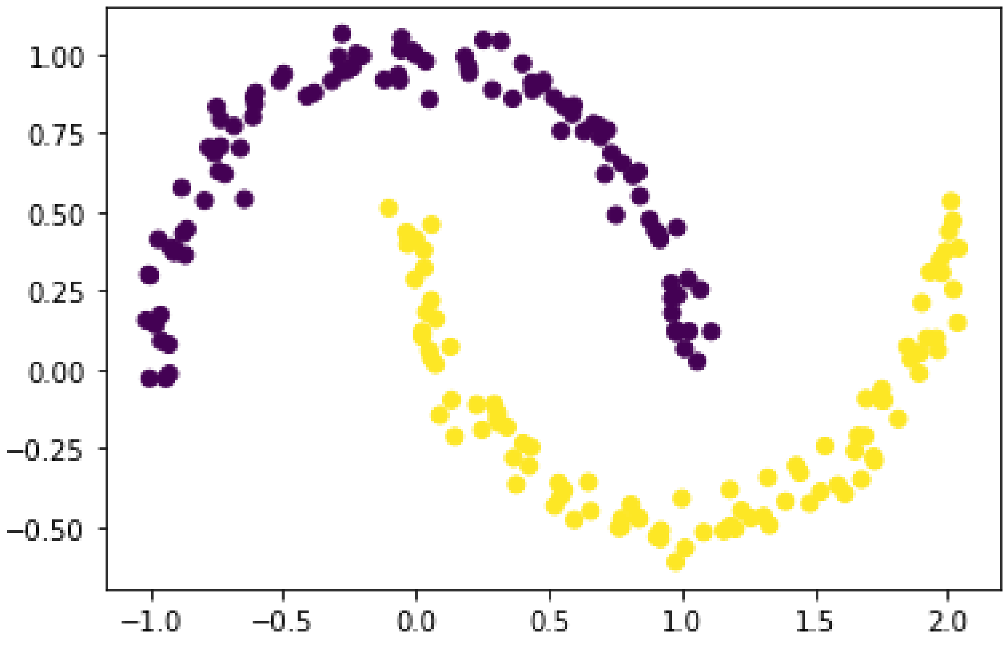


2.4 K-means

-
-

```
[83]: X, y = make_moons(n_samples=200, noise=0.05, random_state=1)
      plt.scatter(X[:,0], X[:,1], c=y)
```

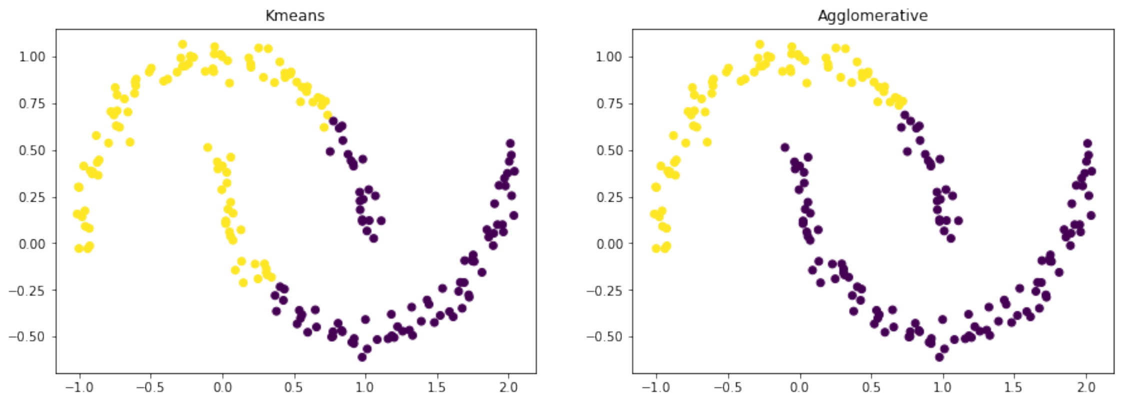
```
[83]: <matplotlib.collections.PathCollection at 0x7fce30a7afd0>
```



[86]:
```python
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
agg = AgglomerativeClustering(n_clusters=2)
agg.fit_predict(X)

fig, axes = plt.subplots(1, 2, figsize=(15, 5))
axes[0].scatter(X[:,0], X[:,1], c=kmeans.labels_)
axes[0].set_title("Kmeans")
axes[1].scatter(X[:,0], X[:,1], c=agg.labels_)
axes[1].set_title("Agglomerative")
```

[86]:
```
Text(0.5, 1.0, 'Agglomerative')
```

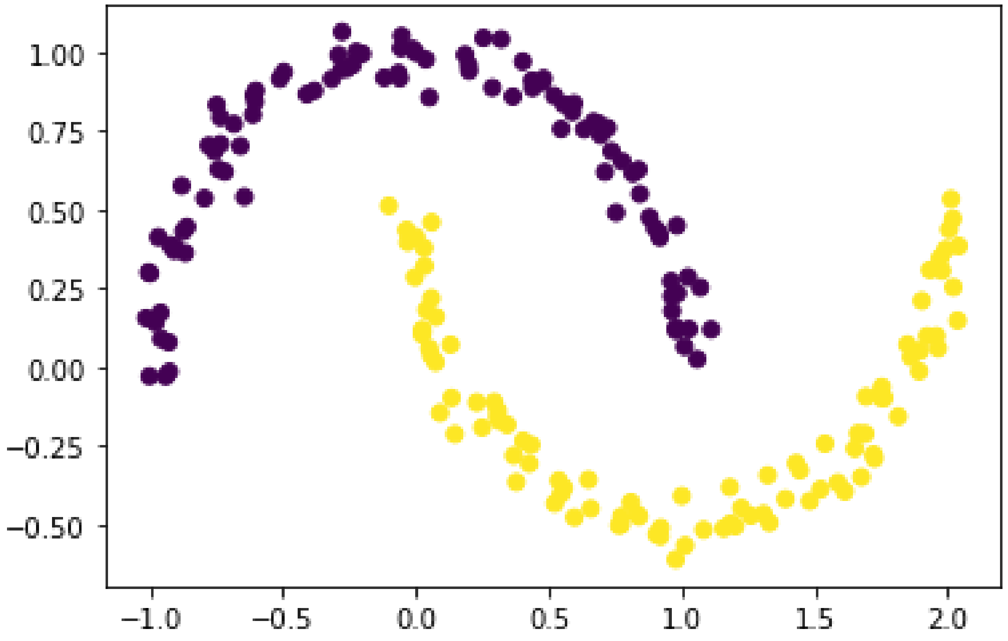


2.4.1 DBSCAN

-
-
-
-
-
    - min_samples
    - eps

[94]:
```
dbscan = DBSCAN(min_samples=5, eps=0.2)
dbscan.fit_predict(X)
plt.scatter(X[:,0], X[:,1], c=dbscan.labels_)
```

[94]:
```
<matplotlib.collections.PathCollection at 0x7fce9148d970>
```



[ ]:
```

```