

clustering_others

November 11, 2025

Author: Riccardo Guidotti

Python version: 3.x

```
[1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from collections import defaultdict
```

```
[2]: from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import silhouette_score
```

1 Data Preparation

```
[3]: from sklearn.datasets import load_breast_cancer

frame = load_breast_cancer(as_frame=True)
df = frame['data']
X = df.values
y = np.array(frame['target'])

df.head()
```

```
mean radius  mean texture  mean perimeter  mean area  mean smoothness \
0      17.99      10.38      122.80     1001.0      0.11840
1      20.57      17.77      132.90     1326.0      0.08474
2      19.69      21.25      130.00     1203.0      0.10960
3      11.42      20.38       77.58      386.1      0.14250
4      20.29      14.34      135.10     1297.0      0.10030

mean compactness  mean concavity  mean concave points  mean symmetry \
0      0.27760      0.3001      0.14710      0.2419
1      0.07864      0.0869      0.07017      0.1812
2      0.15990      0.1974      0.12790      0.2069
3      0.28390      0.2414      0.10520      0.2597
4      0.13280      0.1980      0.10430      0.1809
```

```

mean fractal dimension ... worst radius worst texture worst perimeter \
0           0.07871 ...      25.38      17.33      184.60
1           0.05667 ...      24.99      23.41      158.80
2           0.05999 ...      23.57      25.53      152.50
3           0.09744 ...      14.91      26.50      98.87
4           0.05883 ...      22.54      16.67      152.20

worst area worst smoothness worst compactness worst concavity \
0     2019.0          0.1622      0.6656      0.7119
1     1956.0          0.1238      0.1866      0.2416
2     1709.0          0.1444      0.4245      0.4504
3     567.7           0.2098      0.8663      0.6869
4     1575.0          0.1374      0.2050      0.4000

worst concave points worst symmetry worst fractal dimension
0           0.2654      0.4601      0.11890
1           0.1860      0.2750      0.08902
2           0.2430      0.3613      0.08758
3           0.2575      0.6638      0.17300
4           0.1625      0.2364      0.07678

```

[5 rows x 30 columns]

[4]: `scaler = MinMaxScaler()`

[5]: `X = scaler.fit_transform(X)`

2 X-Means

<https://github.com/annoviko/pyclustering/>

[2]: `#!pip install pyclustering`

[5]: `# standard installation might result in error due to numpy warnings (numpy > 1.
↳ 24.0)`

#after installing pyclustering, also install this warning fix below

`#!pip install https://github.com/KulikDM/pyclustering/archive/Warning-Fix.zip`

[4]: `from pyclustering.cluster import xmeans`

[7]: `xm = xmeans.xmeans(X)
xm.process()`

[7]: <pyclustering.cluster.xmeans.xmeans at 0x7ee062164710>

```
[8]: clusters = xm.get_clusters()

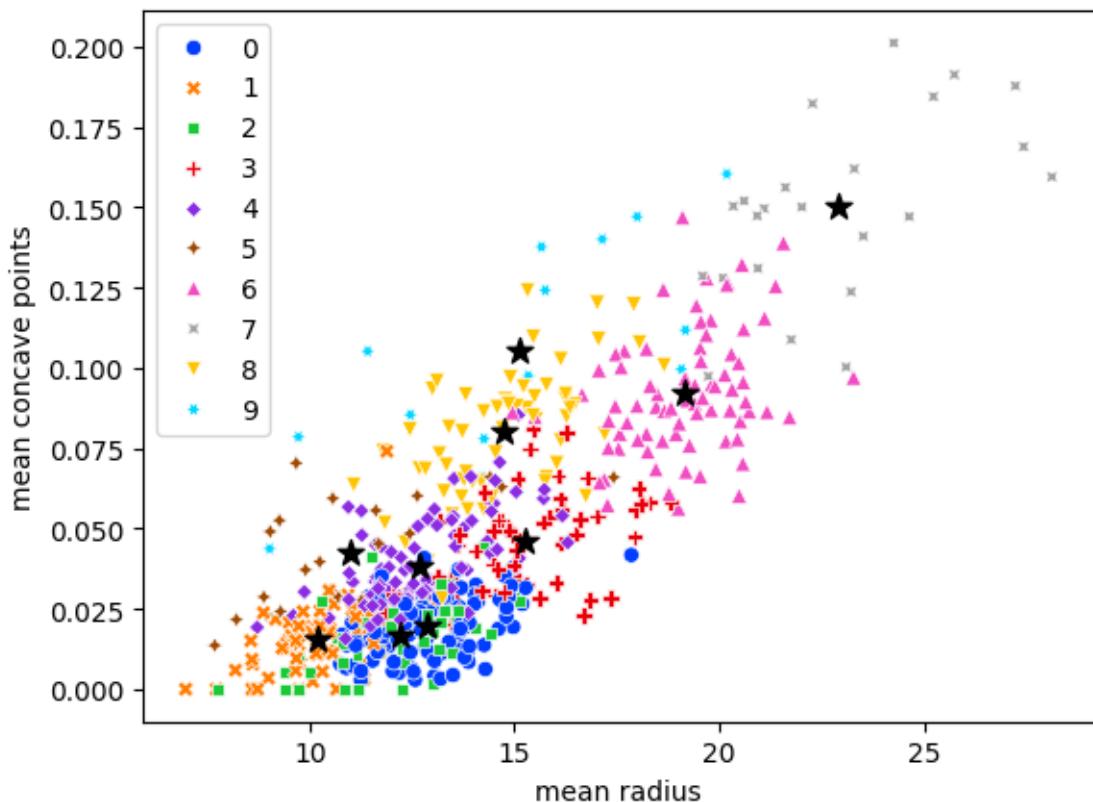
[9]: def clusters_to_labels(clusters):
    labels = np.empty(shape=(len(np.concatenate(clusters))), dtype=int)
    for i in range(len(clusters)):
        for idx in clusters[i]:
            labels[idx] = i
    return labels

[10]: labels = clusters_to_labels(clusters)

[11]: centers = np.array(xm.get_centers())
centers_unscaled = scaler.inverse_transform(centers)

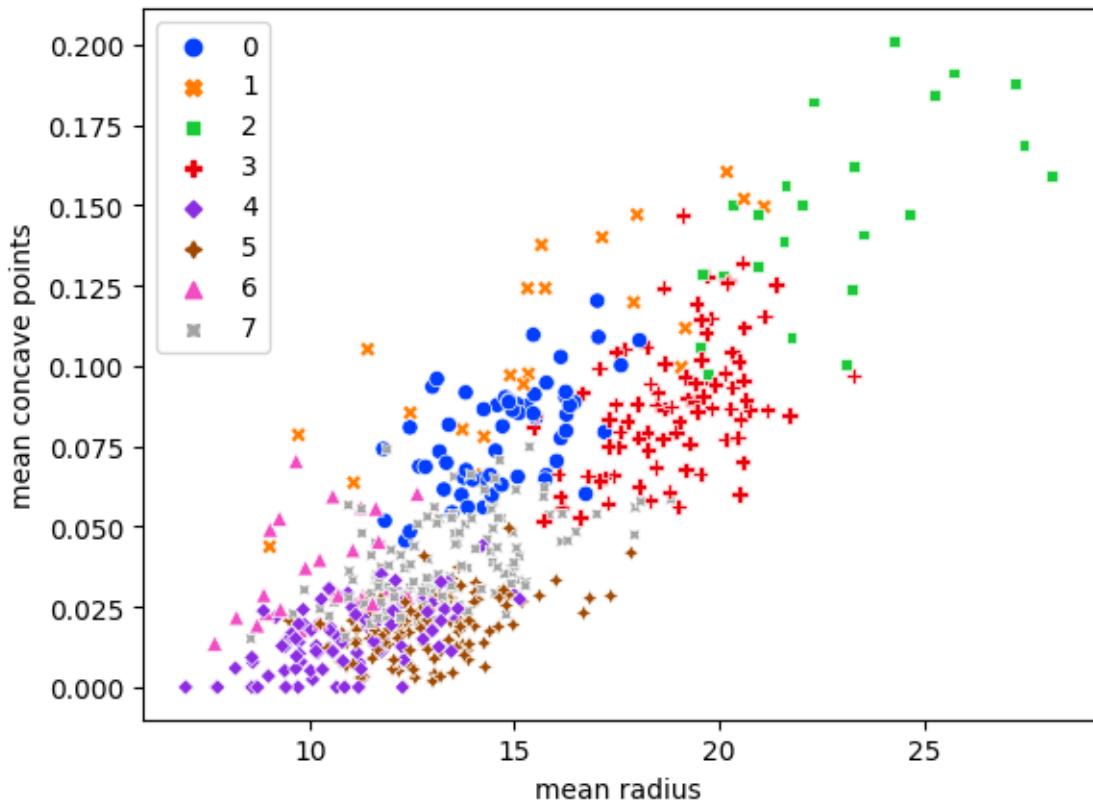
[12]: sns.scatterplot(data=df, x="mean radius", y="mean concave points", hue=labels,
    ↪palette="bright", style=labels)
plt.scatter(centers_unscaled[:, 0], centers_unscaled[:, 7], color="black",
    ↪marker="*", s=100)

[12]: <matplotlib.collections.PathCollection at 0x7ee03d6abda0>
```



3 BISECTING K-MEANS

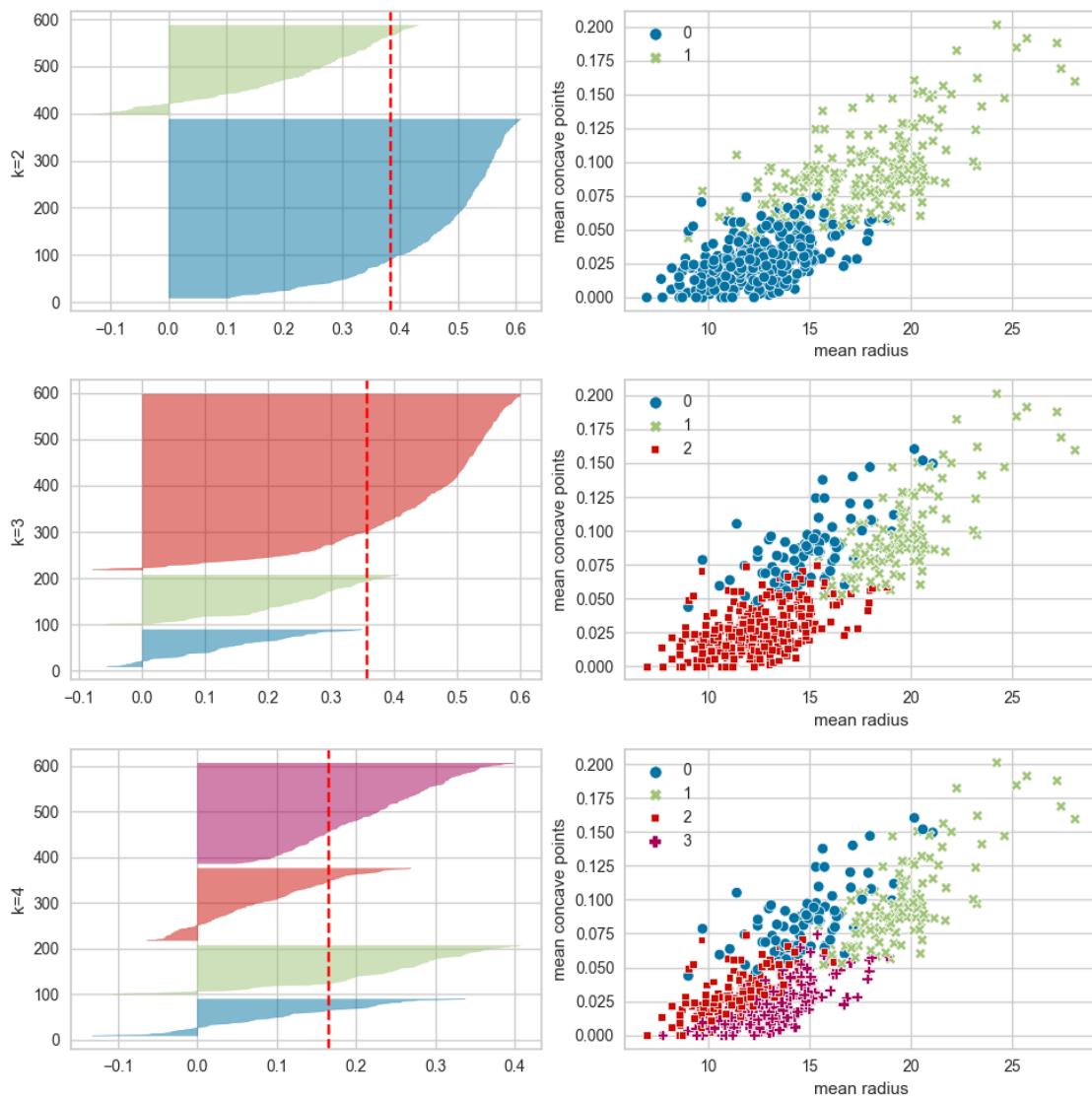
```
[ ]: from sklearn.cluster import BisectingKMeans  
  
[ ]: bkmeans = BisectingKMeans(n_clusters=8)  
bkmeans.fit(X)  
  
[ ]: BisectingKMeans()  
  
[ ]: sns.scatterplot(data=df, x="mean radius", y="mean concave points", hue=bkmeans.  
↳labels_,  
palette="bright", style=bkmeans.labels_)  
  
[ ]: <Axes: xlabel='mean radius', ylabel='mean concave points'>
```



Silhouette plot

```
[ ]: # !pip install yellowbrick  
  
[ ]: from yellowbrick.cluster import SilhouetteVisualizer
```

```
[ ]: colors=['C0', 'C1', 'C2', 'C3', 'C4', 'C5', 'C6', 'C7', 'C8', 'C9']
n_clust = 5
fig, axs = plt.subplots(n_clust-2, 2, figsize=(10,10))
for i in range(2, n_clust):
    bkmeans = BisectingKMeans(n_clusters=i)
    visualizer = SilhouetteVisualizer(bkmeans, colors=colors, ax=axs[i-2][0])
    axs[i-2][0].set_ylabel("k=" + str(i))
    visualizer.fit(X)
    sns.scatterplot(data=df, x="mean radius", y="mean concave points",
                    hue=bkmeans.labels_,
                    palette=sns.color_palette(colors[:i]), style=bkmeans.
                    labels_, ax=axs[i-2][1])
plt.tight_layout()
```



```
[ ]: silhouette_score(X, bkmeans.labels_)
```

```
[ ]: 0.16557736812746163
```

4 OPTICS

```
[ ]: from sklearn.cluster import OPTICS
```

```
[ ]: optics = OPTICS(min_samples=5, max_eps=np.inf)
optics.fit(X)
```

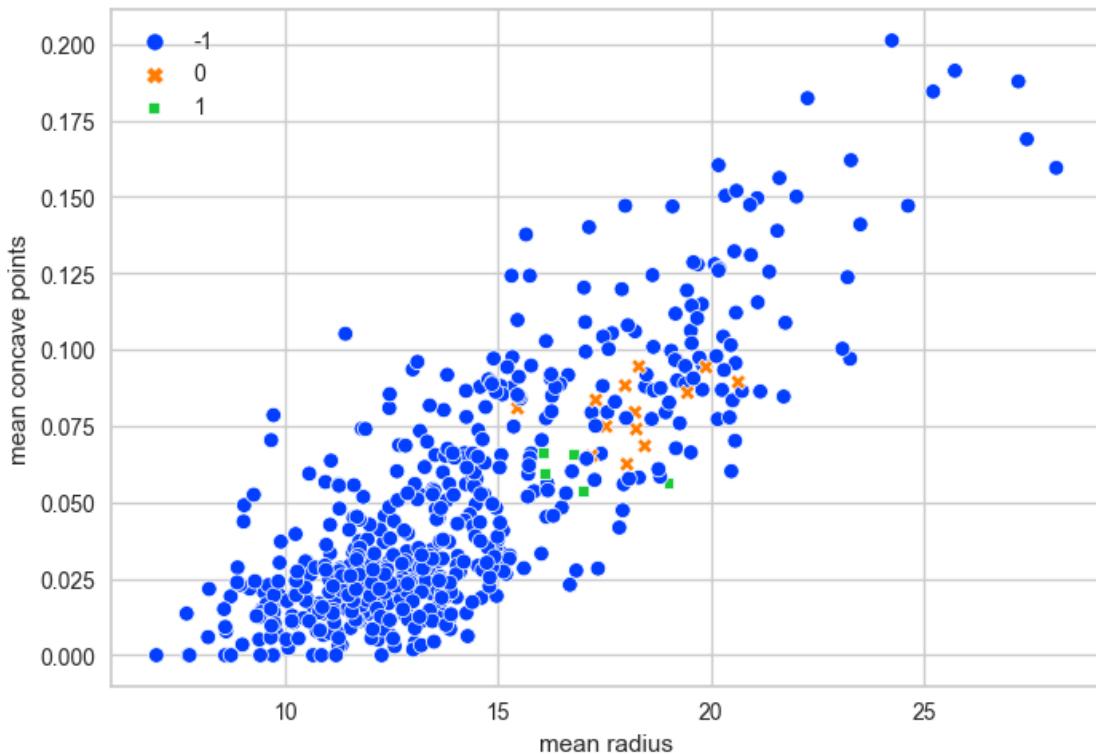
```
[ ]: OPTICS()
```

```
[ ]: silhouette_score(X[optics.labels_ != -1], optics.labels_[optics.labels_ != -1])
```

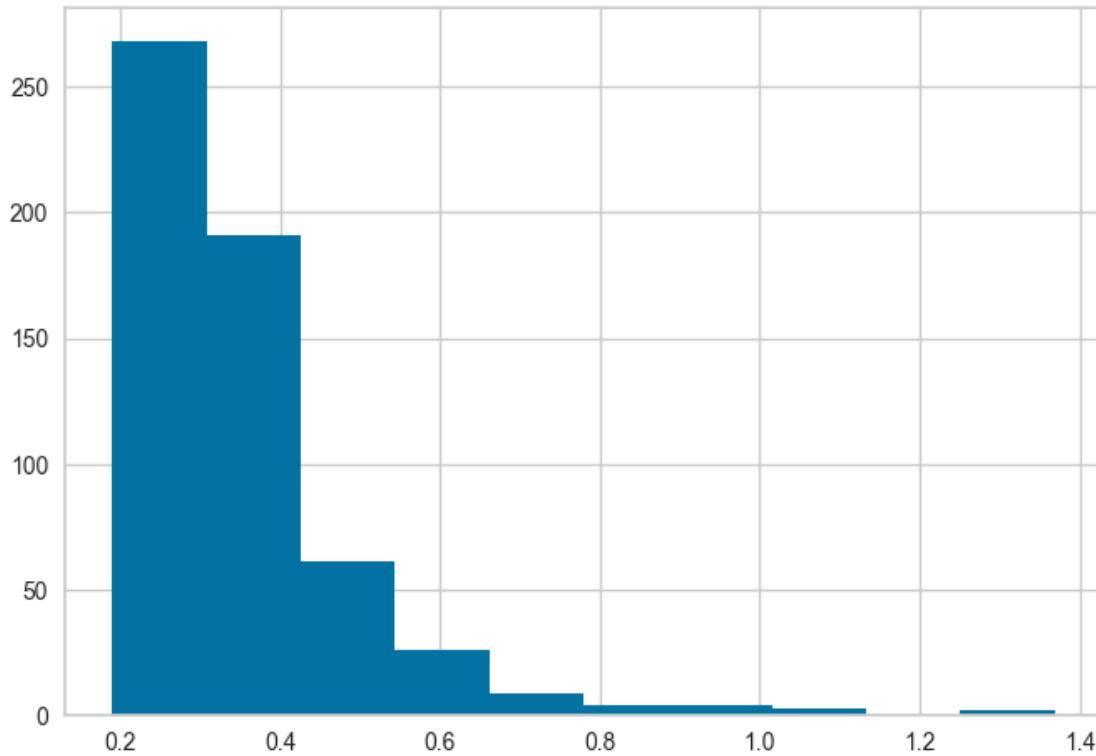
```
[ ]: 0.29091187971598037
```

```
[ ]: sns.scatterplot(data=df, x="mean radius", y="mean concave points", hue=optics.
                     labels_,
                     palette="bright", style=optics.labels_)
```

```
[ ]: <Axes: xlabel='mean radius', ylabel='mean concave points'>
```



```
[ ]: plt.hist(optics.reachability_[1:])
plt.show()
```



```
[ ]: optics = OPTICS(min_samples=5, max_eps=np.inf, cluster_method='dbSCAN', eps=0.3)
optics.fit(X)
```

```
[ ]: OPTICS(cluster_method='dbSCAN', eps=0.3)
```

```
[ ]: silhouette_score(X[optics.labels_ != -1], optics.labels_[optics.labels_ != -1])
```

```
[ ]: 0.07553499719530653
```

```
[ ]: np.unique(optics.labels_)
```

```
[ ]: array([-1,  0,  1,  2,  3])
```

```
[ ]: np.unique(optics.labels_, return_counts=True)
```

```
[ ]: (array([-1,  0,  1,  2,  3]), array([316,   13, 231,    4,     5]))
```

```
[ ]: # https://scikit-learn.org/stable/auto\_examples/cluster/plot\_optics.html
```