6-clustering

March 11, 2021

#

Ciencia de Datos

Víctor Muñiz Sánchez

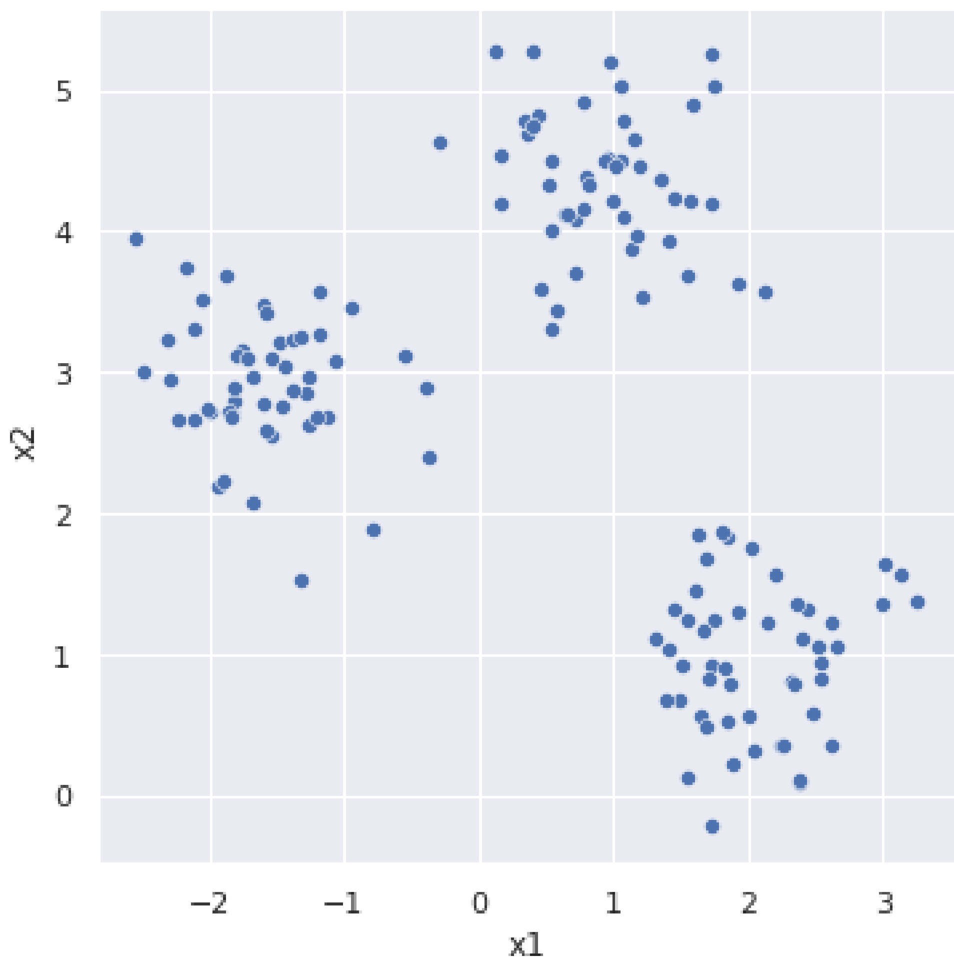
Maestría en Cómputo Estadístico

Enero a junio 2021

0.1 $k-$ means y métodos relacionados

[1]:
```
import pandas as pd
from sklearn.datasets import make_blobs
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
%matplotlib inline

# toy dataset
X, y = make_blobs(
   n_samples=150, n_features=2, centers=3, cluster_std=0.5, shuffle=True,␣
 ↪random_state=0
)

data_toy = pd.DataFrame(X)
data_toy.columns = ['x1','x2']
data_toy = pd.DataFrame(data_toy).assign(cl = y)
sns.relplot(x='x1', y='x2', data = data_toy)
```

[1]:
```
<seaborn.axisgrid.FacetGrid at 0x7fb872a75a00>
```



0.2 $k-$means

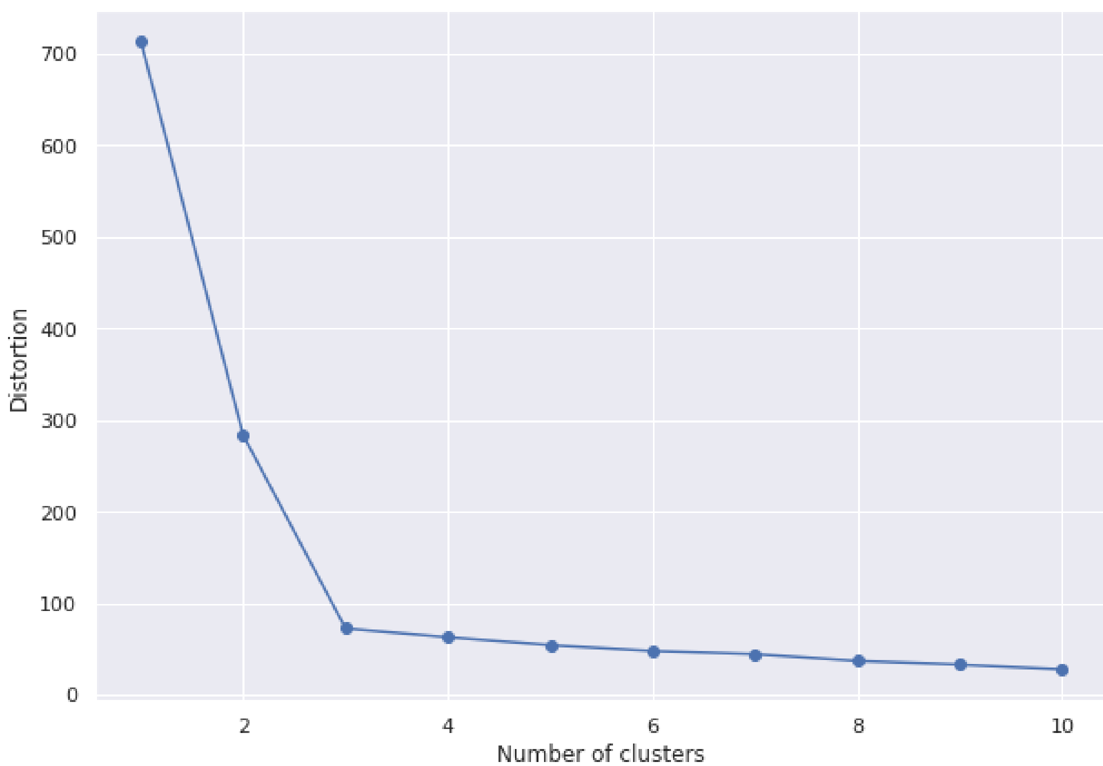
[2]:
```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=3, init='random', n_init=10, max_iter=300,␣
↪tol=1e-04, random_state=0)
y_km = kmeans.fit_predict(X)

data_toy_km = pd.DataFrame(data_toy).assign(cl_km = y_km)

custom_palette = ["red", "green", "blue"]
sns.relplot(x='x1', y='x2', data = data_toy_km, hue='cl_km', height=7, palette␣
↪= custom_palette,
           legend = 'brief')

# plot the centroids
```

```
plt.scatter(
    kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
    s=250, marker='*',
    c='black', edgecolor='black',
    label='centroids'
)
#plt.legend(scatterpoints=1)
plt.show()
```

0.2.1 Una pista del número de clusters. Elbow

0.2.2 clusters vs W(c), disim. dentro de clusters

[108]:
```
distortions = []
for i in range(1, 11):
    km = KMeans(
        n_clusters=i, init='random',
        n_init=10, max_iter=300,
        tol=1e-04, random_state=0
    )
    km.fit(X)
    distortions.append(km.inertia_)

# plot
plt.figure(figsize=(10, 7))
plt.plot(range(1, 11), distortions, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Distortion')
plt.show()
```



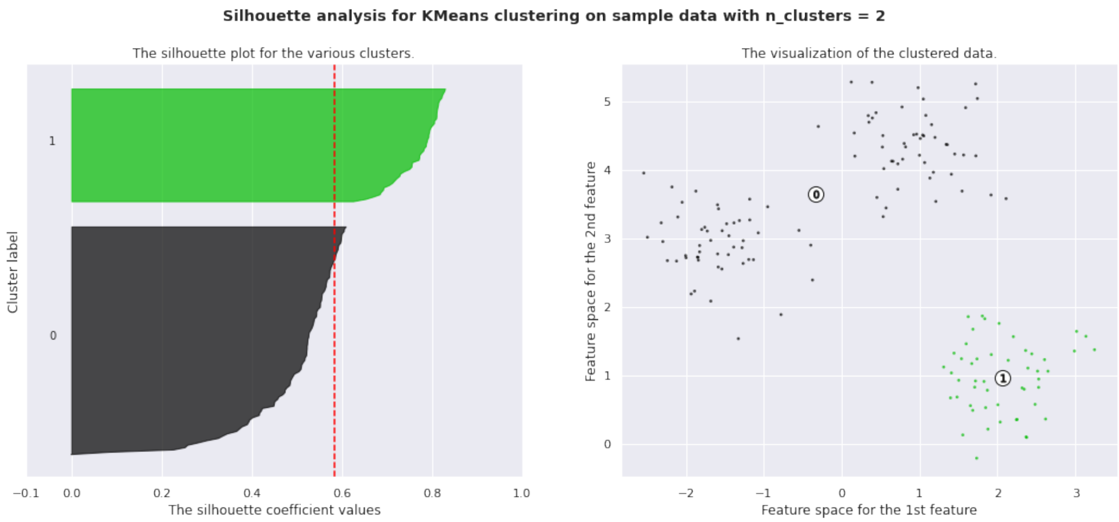
0.3 Gráfico de siluetas

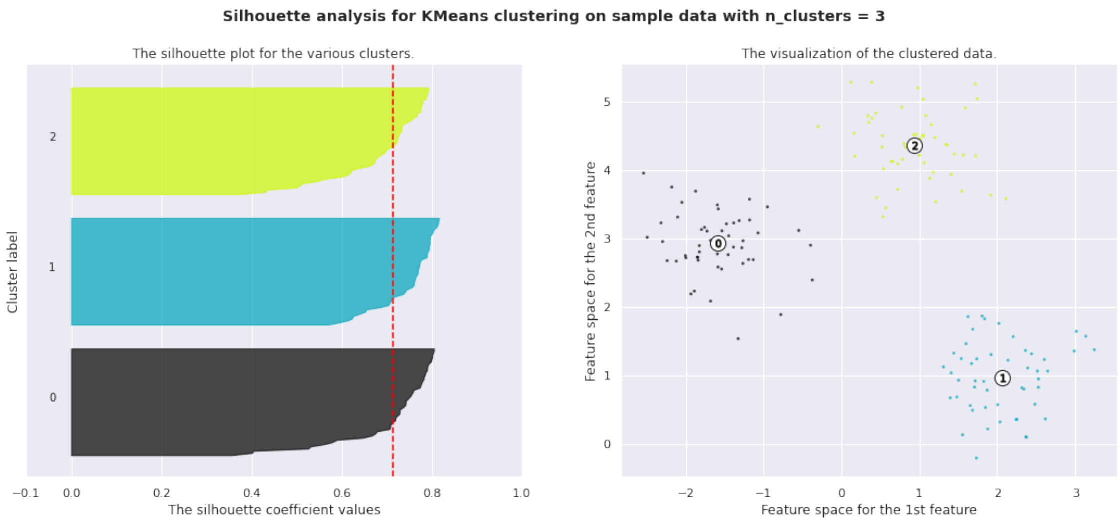
[6]:
```
from sklearn.metrics import silhouette_score
from sklearn.metrics import silhouette_samples, silhouette_score
import matplotlib.cm as cm
import numpy as np

range_n_clusters = [2, 3, 4]

for n_clusters in range_n_clusters:
    # Subplot (1 row, 2 columns)
    fig, (ax1, ax2) = plt.subplots(1, 2)
    fig.set_size_inches(18, 7)

    # 1st subplot es el grafico de silueta
    # Observa que, el coeficiente de silueta está en [-1,1], pero para␣
 ↪visualizar mejor los datos
    # lo ponemos en [-0.1, 1], ya que en este ejemplo todos caen en ese rango
    ax1.set_xlim([-0.1, 1])

    # ponemos un margen de (n_clusters+1)*10 entre cada silueta individual para
    # cada cluster
    ax1.set_ylim([0, len(X) + (n_clusters + 1) * 10])

    # instanciamos un objeto de KMeans, especificando n_clusters en el␣
 ↪constructor
    # fijamos la semilla (random_state) para poder reproducir el resultado
    clusterer = KMeans(n_clusters=n_clusters, random_state=10)
    cluster_labels = clusterer.fit_predict(X)

    # el silhouette_score es el valor promedio para cada muestra ($\bar{s}_K$␣
 ↪en la notación de clase)
    silhouette_avg = silhouette_score(X, cluster_labels)
    print("For n_clusters =", n_clusters,
          "The average silhouette_score is :", silhouette_avg)

    # valor de silueta para cada observacion
    sample_silhouette_values = silhouette_samples(X, cluster_labels)

    y_lower = 10
    for i in range(n_clusters):
        # agregar el score silhouette scores para las observaciones que caen en␣
 ↪el cluster i
        # y se ordenan
        ith_cluster_silhouette_values = \
            sample_silhouette_values[cluster_labels == i]
```

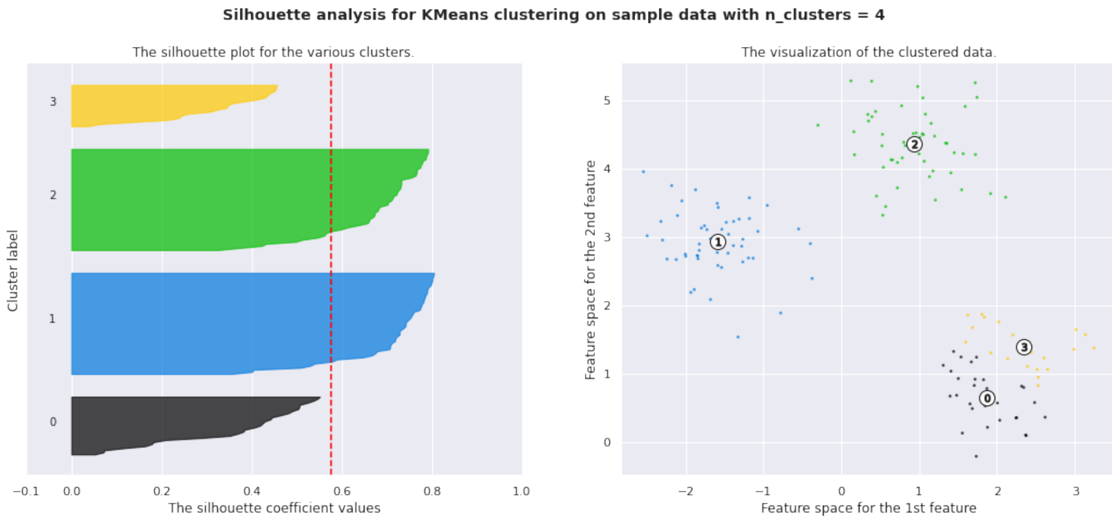
```python
        ith_cluster_silhouette_values.sort()

        size_cluster_i = ith_cluster_silhouette_values.shape[0]
        y_upper = y_lower + size_cluster_i

        color = cm.nipy_spectral(float(i) / n_clusters)
        ax1.fill_betweenx(np.arange(y_lower, y_upper),
                          0, ith_cluster_silhouette_values,
                          facecolor=color, edgecolor=color, alpha=0.7)

        # etiquetas de los silhouette plots
        ax1.text(-0.05, y_lower + 0.5 * size_cluster_i, str(i))

        y_lower = y_upper + 10  # 10 for the 0 samples

    ax1.set_title("The silhouette plot for the various clusters.")
    ax1.set_xlabel("The silhouette coefficient values")
    ax1.set_ylabel("Cluster label")

    # pone el score average silhouette como una linea puteada en rojo
    ax1.axvline(x=silhouette_avg, color="red", linestyle="--")

    ax1.set_yticks([])
    ax1.set_xticks([-0.1, 0, 0.2, 0.4, 0.6, 0.8, 1])

    # El segundo grafico muestra los clusters formados
    colors = cm.nipy_spectral(cluster_labels.astype(float) / n_clusters)
    ax2.scatter(X[:, 0], X[:, 1], marker='.', s=30, lw=0, alpha=0.7,
                c=colors, edgecolor='k')

    centers = clusterer.cluster_centers_
    # grafica centroides
    ax2.scatter(centers[:, 0], centers[:, 1], marker='o',
                c="white", alpha=1, s=200, edgecolor='k')

    for i, c in enumerate(centers):
        ax2.scatter(c[0], c[1], marker='$%d$' % i, alpha=1,
                    s=50, edgecolor='k')

    ax2.set_title("The visualization of the clustered data.")
    ax2.set_xlabel("Feature space for the 1st feature")
    ax2.set_ylabel("Feature space for the 2nd feature")

    plt.suptitle(("Silhouette analysis for KMeans clustering on sample data "
                  "with n_clusters = %d" % n_clusters),
                 fontsize=14, fontweight='bold')
```

```
plt.show()
```

```
For n_clusters = 2 The average silhouette_score is : 0.5848706144251782
For n_clusters = 3 The average silhouette_score is : 0.7143417887288687
For n_clusters = 4 The average silhouette_score is : 0.5768508858868746
```





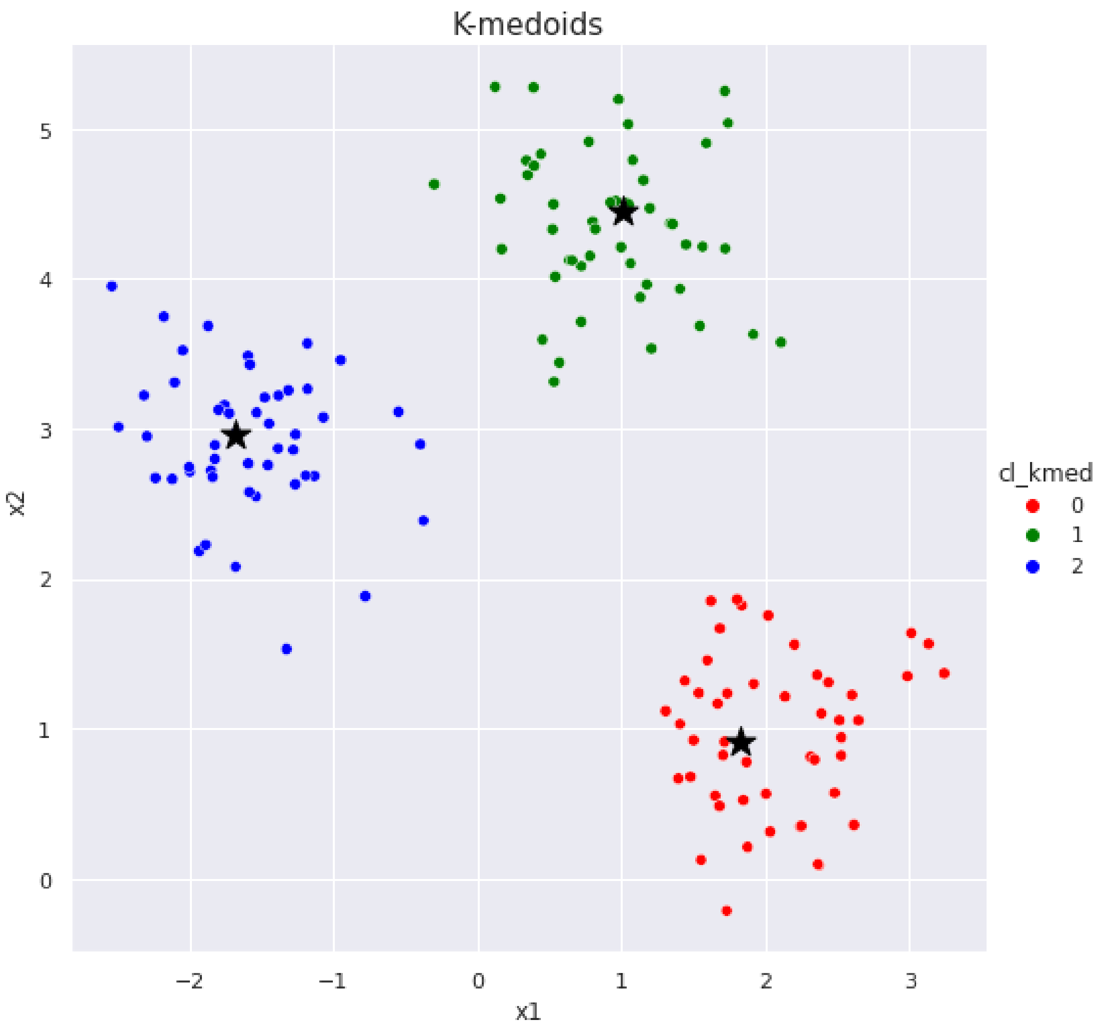


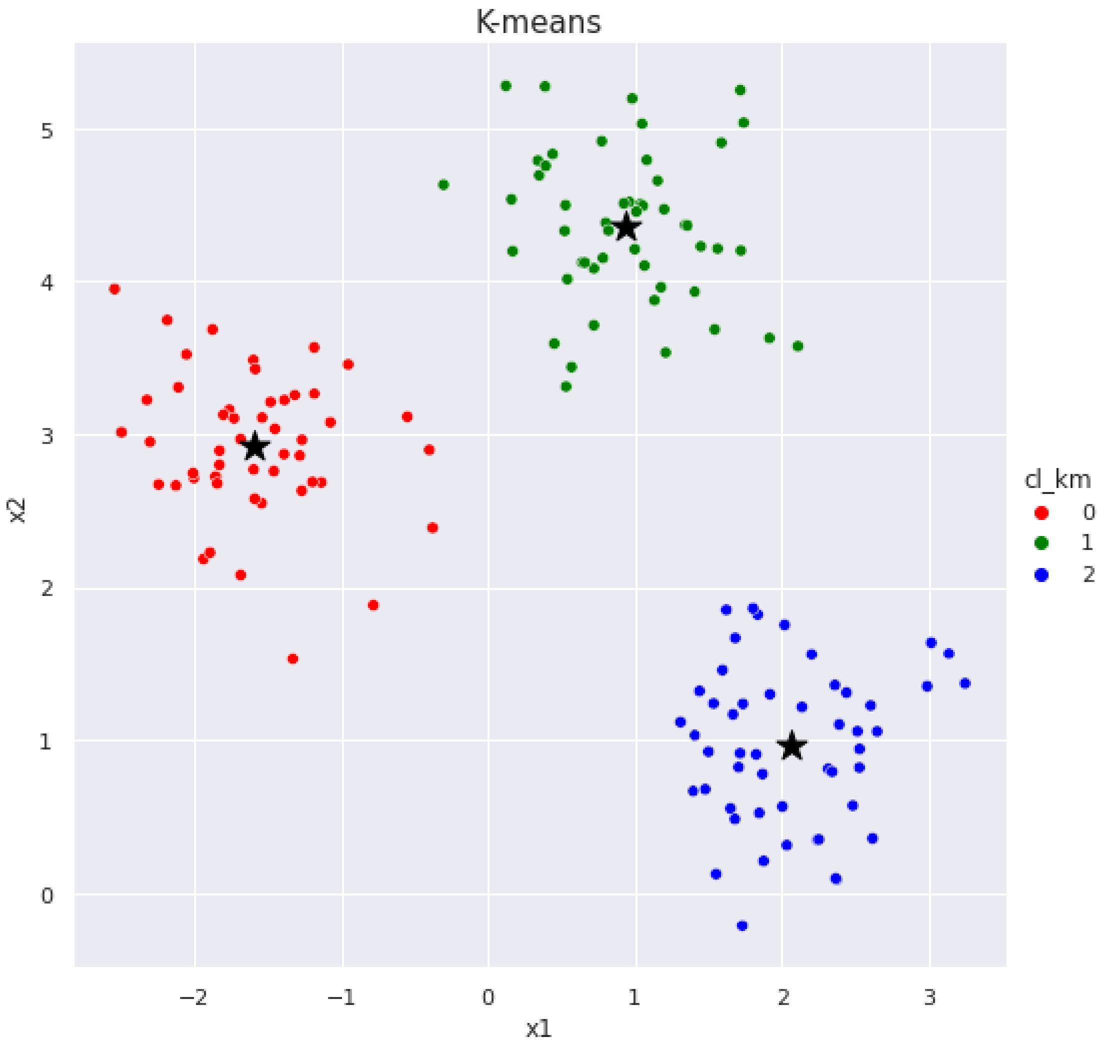
0.4 $k-$medoids

[8]:
```
from sklearn_extra.cluster import KMedoids

kmedoids = KMedoids(n_clusters=3, random_state=0)
kmedoids.fit(X)

data_toy_kmed = pd.DataFrame(data_toy).assign(cl_kmed = kmedoids.labels_)

custom_palette = ["red", "green", "blue"]
sns.relplot(x='x1', y='x2', data = data_toy_kmed, hue='cl_kmed', height=7,␣
 ↪palette = custom_palette,
           legend = 'brief')
plt.title("K-medoids", fontsize=15)
# plot the centroids
plt.scatter(
    kmedoids.cluster_centers_[:, 0], kmedoids.cluster_centers_[:, 1],
    s=250, marker='*',
    c='black', edgecolor='black',
    label='centroids'
)

sns.relplot(x='x1', y='x2', data = data_toy_km, hue='cl_km', height=7, palette␣
 ↪= custom_palette,
           legend = 'brief')
plt.title("K-means", fontsize=15)
# plot the centroids
plt.scatter(
    kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
```

```
    s=250, marker='*',
    c='black', edgecolor='black',
    label='centroids'
)

#plt.legend(scatterpoints=1)
#plt.show()
```

[8]: <matplotlib.collections.PathCollection at 0x7fb8654204c0>





0.5 Fuzzy $k-$means

[11]:
```
from sklearn_extensions.fuzzy_kmeans import KMedians, FuzzyKMeans
# m es el parámetro de fuzyness
fuzzy_kmeans = FuzzyKMeans(k=3, m=2)
fuzzy_kmeans.fit(X)
#dir(fuzzy_kmeans)
res = pd.DataFrame(np.round(fuzzy_kmeans.fuzzy_labels_,3)).
 ↪assign(label=fuzzy_kmeans.labels_)
print(res.to_string())
```

```
          0      1      2  label
0     0.017  0.954  0.028      1
1     0.184  0.111  0.704      2
2     0.002  0.001  0.997      2
```

3	0.024	0.011	0.965	2
4	0.026	0.940	0.034	1
5	0.048	0.043	0.909	2
6	0.064	0.060	0.877	2
7	0.033	0.928	0.039	1
8	0.958	0.016	0.026	0
9	0.075	0.066	0.859	2
10	0.053	0.844	0.103	1
11	0.639	0.148	0.212	0
12	0.985	0.006	0.010	0
13	0.016	0.010	0.974	2
14	0.093	0.032	0.874	2
15	0.973	0.010	0.017	0
16	0.982	0.005	0.013	0
17	0.004	0.992	0.005	1
18	0.995	0.002	0.003	0
19	0.020	0.958	0.022	1
20	0.020	0.011	0.969	2
21	0.014	0.963	0.023	1
22	0.031	0.028	0.941	2
23	0.038	0.025	0.937	2
24	0.958	0.015	0.027	0
25	0.004	0.989	0.006	1
26	0.014	0.964	0.022	1
27	0.076	0.086	0.837	2
28	0.822	0.051	0.127	0
29	0.026	0.947	0.028	1
30	0.946	0.014	0.040	0
31	0.990	0.003	0.006	0
32	0.939	0.021	0.040	0
33	0.989	0.003	0.008	0
34	0.058	0.024	0.918	2
35	0.045	0.882	0.073	1
36	0.039	0.892	0.069	1
37	0.012	0.971	0.017	1
38	0.018	0.010	0.972	2
39	0.013	0.010	0.977	2
40	0.749	0.132	0.119	0
41	0.900	0.044	0.056	0
42	0.065	0.027	0.908	2
43	0.028	0.942	0.030	1
44	0.004	0.989	0.007	1
45	0.014	0.967	0.019	1
46	0.934	0.017	0.049	0
47	0.099	0.205	0.696	2
48	0.951	0.017	0.032	0
49	0.026	0.026	0.949	2
50	0.052	0.854	0.094	1

51	0.054	0.024	0.923	2
52	0.019	0.013	0.967	2
53	0.012	0.973	0.014	1
54	0.007	0.983	0.010	1
55	0.993	0.002	0.005	0
56	0.067	0.043	0.890	2
57	0.051	0.849	0.100	1
58	0.871	0.029	0.100	0
59	0.016	0.014	0.970	2
60	0.933	0.022	0.045	0
61	0.706	0.072	0.222	0
62	0.999	0.000	0.001	0
63	0.994	0.002	0.004	0
64	0.010	0.008	0.982	2
65	0.651	0.099	0.250	0
66	0.004	0.003	0.994	2
67	0.069	0.813	0.119	1
68	0.240	0.058	0.702	2
69	0.141	0.053	0.806	2
70	0.067	0.092	0.841	2
71	0.039	0.913	0.049	1
72	0.011	0.973	0.016	1
73	0.101	0.045	0.854	2
74	0.025	0.942	0.032	1
75	0.001	0.001	0.998	2
76	0.035	0.037	0.928	2
77	0.916	0.036	0.048	0
78	0.973	0.008	0.019	0
79	0.003	0.002	0.995	2
80	0.007	0.984	0.009	1
81	0.019	0.961	0.021	1
82	0.136	0.067	0.796	2
83	0.009	0.007	0.985	2
84	0.022	0.939	0.039	1
85	0.052	0.853	0.095	1
86	0.011	0.973	0.016	1
87	0.989	0.003	0.007	0
88	0.928	0.021	0.051	0
89	0.032	0.929	0.038	1
90	0.018	0.960	0.022	1
91	0.003	0.002	0.994	2
92	0.044	0.907	0.049	1
93	0.033	0.018	0.949	2
94	0.009	0.981	0.010	1
95	0.089	0.160	0.752	2
96	0.974	0.009	0.017	0
97	0.950	0.016	0.035	0
98	0.034	0.932	0.035	1

99	0.039	0.890	0.071	1
100	0.013	0.969	0.017	1
101	0.010	0.974	0.016	1
102	0.881	0.033	0.085	0
103	0.008	0.982	0.010	1
104	0.021	0.955	0.024	1
105	0.004	0.003	0.994	2
106	0.994	0.002	0.004	0
107	0.047	0.056	0.897	2
108	0.003	0.002	0.996	2
109	0.055	0.023	0.922	2
110	0.960	0.011	0.029	0
111	0.008	0.005	0.986	2
112	0.003	0.993	0.004	1
113	0.974	0.008	0.018	0
114	0.054	0.034	0.912	2
115	0.918	0.029	0.053	0
116	0.002	0.001	0.997	2
117	0.038	0.042	0.920	2
118	0.976	0.008	0.016	0
119	0.993	0.002	0.005	0
120	0.042	0.021	0.936	2
121	0.016	0.959	0.025	1
122	0.024	0.011	0.965	2
123	0.106	0.036	0.858	2
124	0.057	0.837	0.105	1
125	0.009	0.978	0.013	1
126	0.973	0.010	0.017	0
127	0.020	0.958	0.022	1
128	0.945	0.019	0.036	0
129	0.989	0.004	0.007	0
130	0.989	0.004	0.008	0
131	0.928	0.020	0.052	0
132	0.031	0.934	0.035	1
133	0.978	0.008	0.014	0
134	0.956	0.012	0.032	0
135	0.711	0.146	0.143	0
136	0.147	0.088	0.765	2
137	0.984	0.006	0.010	0
138	0.034	0.919	0.047	1
139	0.982	0.007	0.011	0
140	0.017	0.015	0.968	2
141	0.024	0.022	0.954	2
142	0.061	0.879	0.060	1
143	0.005	0.987	0.007	1
144	0.826	0.038	0.136	0
145	0.945	0.020	0.034	0
146	0.907	0.025	0.068	0

```
147  0.925  0.032  0.044      0
148  0.018  0.960  0.022      1
149  0.047  0.906  0.046      1
```

[10]:
```python
data_toy_fkm = pd.DataFrame(data_toy).assign(cl_fkm = fuzzy_kmeans.labels_)

custom_palette = ["red", "green", "blue"]
sns.relplot(x='x1', y='x2', data = data_toy_fkm, hue='cl_fkm', height=7,␣
 ↪palette = custom_palette,
            legend = 'brief')

# plot the centroids
plt.scatter(
    fuzzy_kmeans.cluster_centers_[:, 0], fuzzy_kmeans.cluster_centers_[:, 1],
    s=250, marker='*',
    c='black', edgecolor='black',
    label='centroids'
)
#plt.legend(scatterpoints=1)
plt.show()
```

5
4
3
2
1
0
x2
−2
−1
0
1
2
3
x1
cl_fkm
0
1
2