Spectral

June 15, 2021

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns

from matplotlib import pyplot as plt
from sklearn.cluster import KMeans, DBSCAN
from utils import get_data_train, get_columns
```

[7]:
```
import numpy as np

from sklearn.cluster import SpectralClustering
from sklearn import metrics
from sklearn.datasets import make_blobs
from sklearn.preprocessing import StandardScaler
from tqdm import tqdm
```

[3]:
```
df = get_data_train()
chosen_cols = get_columns(df, n_cols=25) + ['activity', 'subject']
```

[4]:
```
X = df[chosen_cols].drop(['activity', 'subject'], axis=1)
y = df['activity']
```

[10]:
```
epss = [4, 6, 8, 10, 12, 14, 16] # n_neighbors
min_samples = [2,3,4,5,6,7,8,9, 10] # n_clusters
n_epss = len( epss)
n_min_samples = len( min_samples)

homogenities = np.ndarray((n_epss, n_min_samples),)
completenesses = np.ndarray((n_epss, n_min_samples),)
v_measures = np.ndarray((n_epss, n_min_samples),)
adjusted_rands = np.ndarray((n_epss, n_min_samples),)
adjusted_mutual_infos = np.ndarray((n_epss, n_min_samples),)
silhuettes = np.ndarray((n_epss, n_min_samples),)
```

[11]:
```
for i in tqdm(range( n_epss)):
    for j in range( n_min_samples):
        sc = SpectralClustering( n_neighbors=epss[i],␣
↪n_clusters=min_samples[j]).fit(X)
```

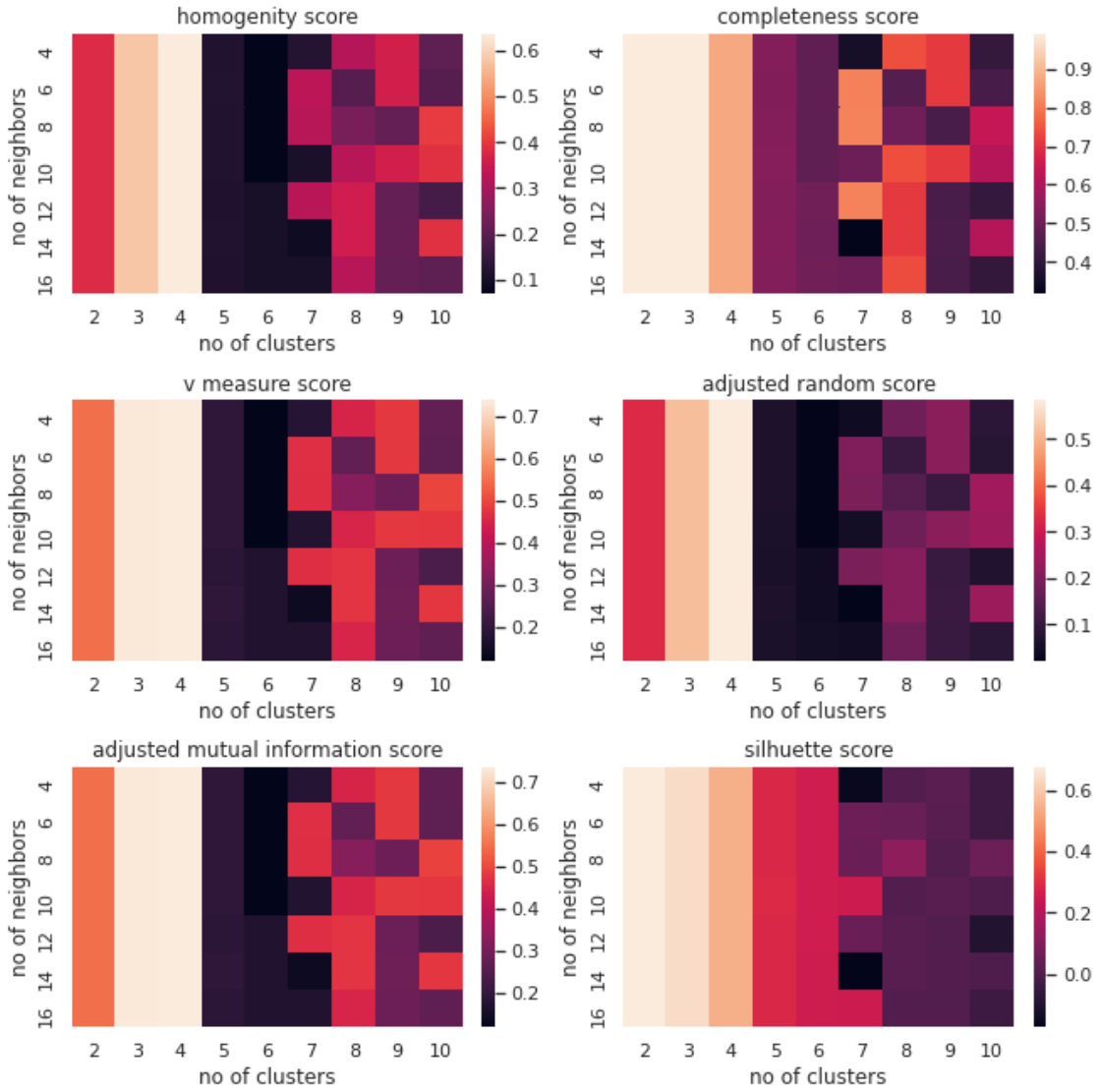
```
        labels = sc.labels_
        homogenities[i,j] = metrics.homogeneity_score(y, labels)
        completenesses[i,j] = metrics.completeness_score(y, labels)
        v_measures[i,j] = metrics.v_measure_score(y, labels)
        adjusted_rands[i,j] = metrics.adjusted_rand_score(y, labels)
        adjusted_mutual_infos[i,j] = metrics.adjusted_mutual_info_score(y,␣
 ↪labels)
        silhuettes[i,j] = metrics.silhouette_score(X, labels)
```

```
100%|      | 7/7 [13:15<00:00, 113.58s/it]
```

[9]:
```
import seaborn as sns; sns.set_theme()
import pandas as pd
```

[13]:
```
homogenities_df = pd.DataFrame( homogenities, columns=min_samples, index=epss )
completenesses_df = pd.DataFrame( completenesses, columns=min_samples,␣
 ↪index=epss )
v_measures_df = pd.DataFrame( v_measures, columns=min_samples, index=epss )
adjusted_rands_df = pd.DataFrame( adjusted_rands, columns=min_samples,␣
 ↪index=epss )
adjusted_mutual_infos_df = pd.DataFrame( adjusted_mutual_infos,␣
 ↪columns=min_samples, index=epss )
silhuettes_df = pd.DataFrame( silhuettes, columns=min_samples, index=epss )
```

[15]:
```
fig, axs = plt.subplots(3, 2,figsize=(9,9), constrained_layout=True)
#fig.tight_layout()
sns.heatmap( ax = axs[0,0], data = homogenities_df).set(title='homogenity␣
 ↪score', ylabel="no of neighbors", xlabel = "no of clusters")
sns.heatmap( ax = axs[0,1], data = completenesses_df).set(title='completeness␣
 ↪score', ylabel="no of neighbors", xlabel = "no of clusters")
sns.heatmap( ax = axs[1,0], data = v_measures_df).set(title='v measure score',␣
 ↪ylabel="no of neighbors", xlabel = "no of clusters")
sns.heatmap( ax = axs[1,1], data = adjusted_rands_df).set(title='adjusted␣
 ↪random score', ylabel="no of neighbors", xlabel = "no of clusters")
sns.heatmap( ax = axs[2,0], data = adjusted_mutual_infos_df).
 ↪set(title='adjusted mutual information score', ylabel="no of neighbors",␣
 ↪xlabel = "no of clusters")
sns.heatmap( ax = axs[2,1], data = silhuettes_df).set(title='silhuette score',␣
 ↪ylabel="no of neighbors", xlabel = "no of clusters")

plt.show()
```



[ ]:

[16]:
```
sc = SpectralClustering( n_neighbors=10, n_clusters=3).fit(X)
labels = sc.labels_
```

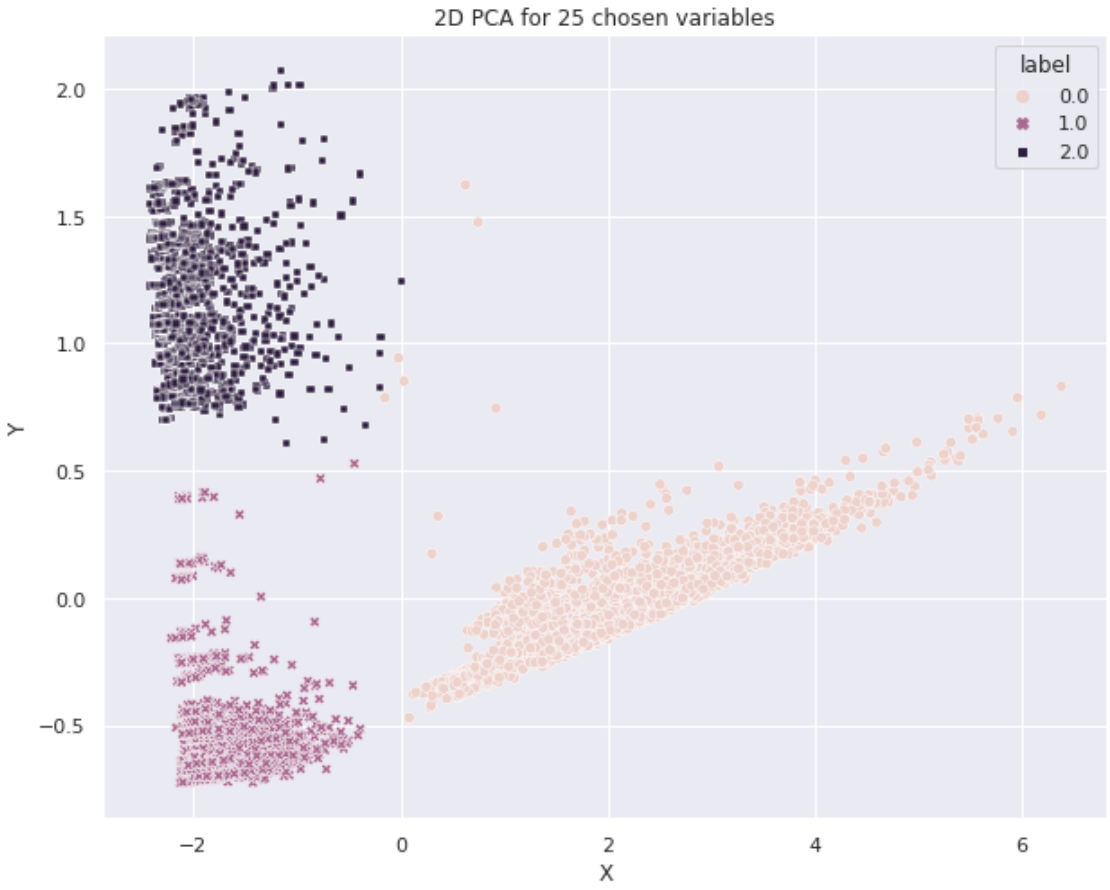
[ ]:

[17]:
```
import pandas as pd
import numpy as np
import seaborn as sns

from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
```

```
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE

from utils import get_data_train, get_columns
```

[18]:
```
def visualize(X, labels, title, vis_tool, ax=None, **kwargs):
    # vis tool = PCA or TSNE, **kwargs: for example random_state for TSNE
    vis_tool = vis_tool(n_components=2, **kwargs)
    res = vis_tool.fit_transform(X)
    res_labels = pd.DataFrame(np.column_stack((res,labels)),
                 columns=['X', 'Y', 'label'])
    if ax is None:
        ax = plt.figure(figsize=(10,8))
        no_return = False
    else:
        no_return = True
    sns.scatterplot(data=res_labels, x='X', y='Y', hue='label', style='label')
    plt.title(title)
    if no_return:
        return
    return ax
```

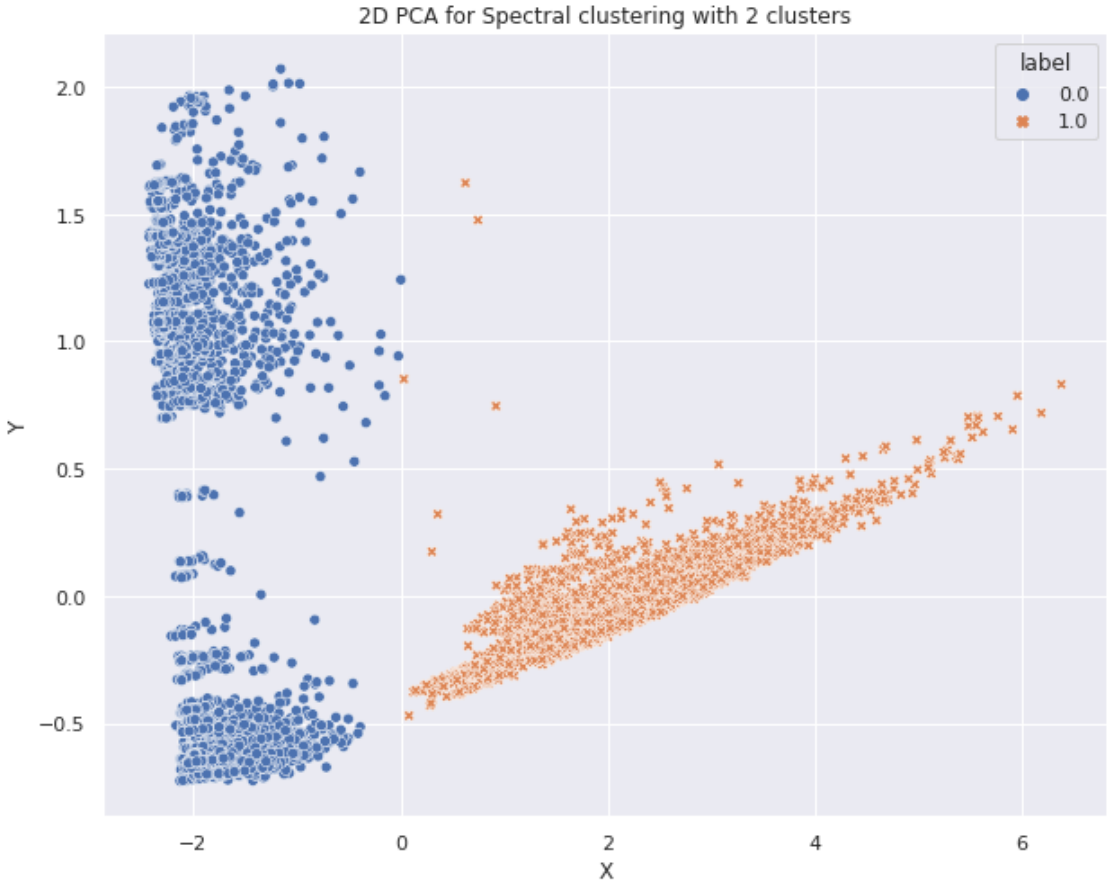
[19]:
```
ax = visualize(X, labels, '2D PCA for 25 chosen variables', PCA)
plt.show()
```

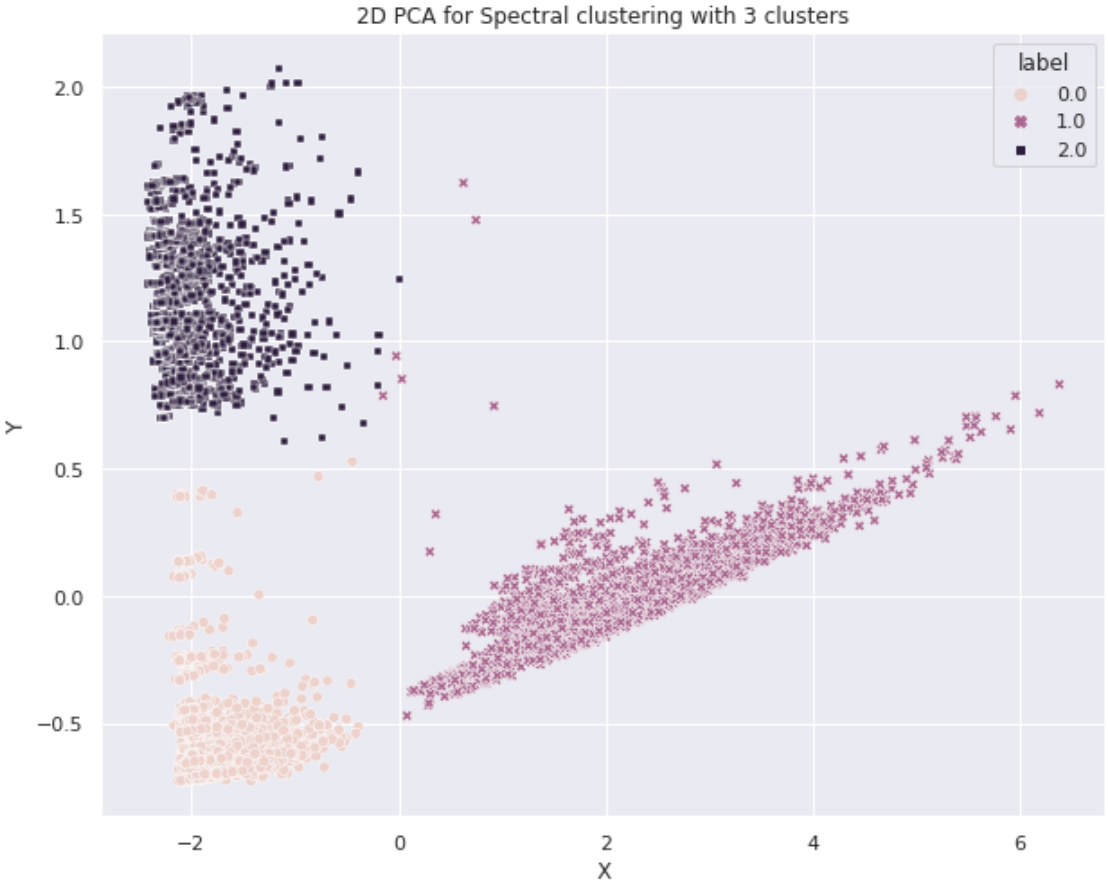


[ ]:

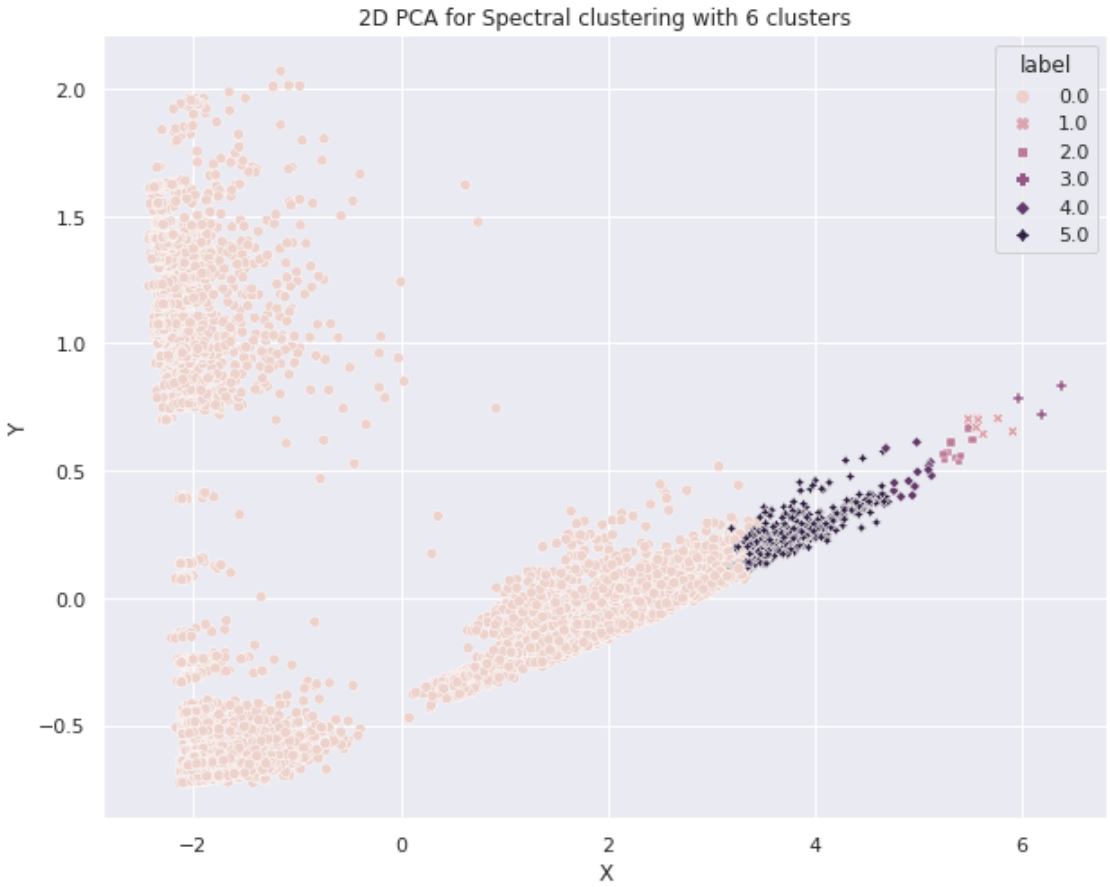
[ ]:

[22]:
```
n_clusters = [2,3,4,6]
for n in n_clusters:
    sc = SpectralClustering( n_neighbors=6, n_clusters=n).fit(X)
    clusters = sc.fit_predict(X)
    ax = visualize(X, clusters,
                   '2D PCA for Spectral clustering with ' + str(n) + '␣
 ↪clusters', PCA)
    plt.show()
```


2D PCA for Spectral clustering with 2 clusters
label
0.0
1.0
2.0
1.5
1.0
0.5
0.0
−0.5
Y
−2
0
2
4
6
X


2D PCA for Spectral clustering with 3 clusters
label
0.0
1.0
2.0
Y
X
2.0
1.5
1.0
0.5
0.0
−0.5
−2
0
2
4
6


2D PCA for Spectral clustering with 4 clusters
label
0.0
1.0
2.0
3.0
Y
X
2.0
1.5
1.0
0.5
0.0
−0.5
−2
0
2
4
6



[36]:
```
visualize(X, y, '2D PCA for 25 chosen cols colored by activity', PCA)
```
[36]:


2D PCA for 25 chosen cols colored by activity
label
STANDING
SITTING
LAYING
WALKING
WALKING_DOWNSTAIRS
WALKING_UPSTAIRS
Y
X
2.0
1.5
1.0
0.5
0.0
−0.5
−2
0
2
4
6

[34]:
```
visualize(X, SpectralClustering( n_neighbors=6, n_clusters=3).fit(X).labels_,
    '2D TSNE for 25 chosen cols colored by Spectral Clustering for ' + str(3) +␣
 ↪' clusters', TSNE, random_state=123, n_jobs=-1)
plt.show()
visualize(X, y,
    '2D TSNE for 25 chosen cols colored by activity', TSNE, random_state=123,␣
 ↪n_jobs=-1)
plt.show()
```


2D TSNE for 25 chosen cols colored by Spectral Clustering for 3 clusters
label
0.0
1.0
2.0
Y
X
80
60
40
20
0
−20
−40
−60
−80
−80
−60
−40
−20
0
20
40
60
80

[35]:
```
visualize(X, SpectralClustering( n_neighbors=6, n_clusters=4).fit(X).labels_,
    '2D TSNE for 25 chosen cols colored by Spectral Clustering for ' + str(4) +␣
 ↪' clusters', TSNE, random_state=123, n_jobs=-1)
plt.show()
```

[37]:
```
visualize(X, SpectralClustering( n_neighbors=6, n_clusters=2).fit(X).labels_,
    '2D TSNE for 25 chosen cols colored by Spectral Clustering for ' + str(2) +␣
 ↪' clusters', TSNE, random_state=123, n_jobs=-1)
plt.show()
```


2D TSNE for 25 chosen cols colored by Spectral Clustering for 2 clusters
label
0.0
1.0
80
60
40
20
0
−20
−40
−60
−80
Y
−80
−60
−40
−20
0
20
40
60
80
X