clustering_stability

June 13, 2021

[1]:
```python
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
```

[2]:
```python
df = get_data_train()
chosen_cols = get_columns(df, n_cols=25) + ['activity', 'subject']
```

[3]:
```python
X = df[chosen_cols].drop(['activity', 'subject'], axis=1)
```

[4]:
```python
from sklearn.metrics import adjusted_mutual_info_score

def cluster_stability(X, model, resamples_n:int, **kwargs):
    stability_scores = []
    # initialize instance of a model with passed key words args
    instance = model(**kwargs)
    # predict base labels
    labels = instance.fit_predict(X)

    for i in range(resamples_n):
        instance = model(**kwargs)
        # botstrap of X
        bootstraped = X.sample(frac=1, replace=True)

        # train on bootstrap
        instance.fit(bootstraped)

        # get predicted labels for permutated data
        predicted = instance.predict(X)

        # get mutual info score between base labels and just predicted
        stability_scores.append(adjusted_mutual_info_score(labels, predicted))

    return np.mean(stability_scores)
```

[11]:
```python
print(cluster_stability(X, KMeans, 10, n_clusters=4))
print(cluster_stability(X, KMeans, 10, n_clusters=6))
```

```
0.9858294606585304
0.9785783215554398
```

Wygląda na to, że klastrujemy stabilnie

[6]:
```python
scores = []
for i in range(2,11):
    scores.append(cluster_stability(X, KMeans, 10, n_clusters=i))
```

[10]:
```python
import plotly.express as px

fig = px.line(x=range(2,11), y=scores,
              labels={'x':'n_clusters', 'y':'Adjusted Mutual Info'},
              title='Clustering Stability for KMeans (mean of 10 resamplings)')
fig.show()
```

Najlepsza stabilność jest przy 2 lub 3, powyżej 6 spada mocniej.