```
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
```

1. PCA

```
file_path = 'winequality-white.csv'
df = pd.read_csv(file_path, sep=';')

df.head()
```

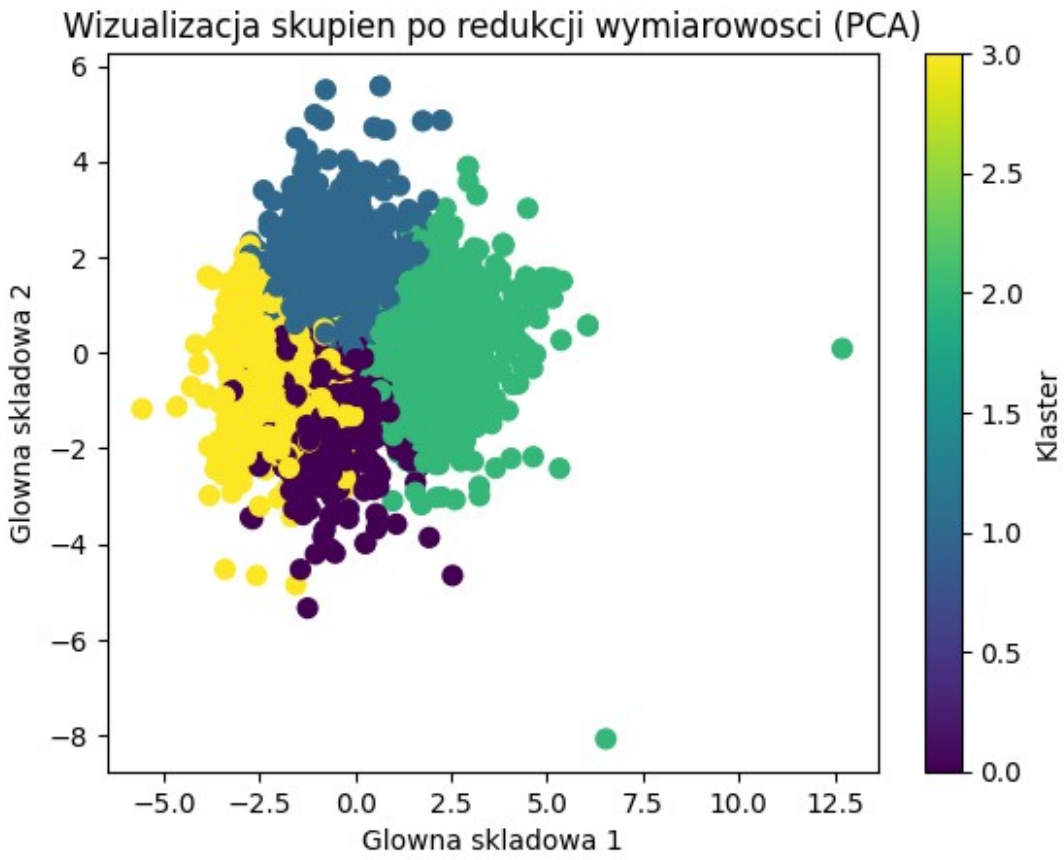

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 4898,\n  \"fields\":
[\n    {\n      \"column\": \"fixed acidity\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
0.8438682276875188,\n        \"min\": 3.8,\n        \"max\": 14.2,\n
\"num_unique_values\": 68,\n        \"samples\": [\n          10.3,\n
5.8,\n          6.2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"volatile acidity\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.10079454842486428,\n        \"min\":
0.08,\n        \"max\": 1.1,\n        \"num_unique_values\": 125,\n
\"samples\": [\n          0.14,\n          0.595,\n          0.13\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"citric acid\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.12101980420298301,\n        \"min\": 0.0,\n        \"max\": 1.66,\n
\"num_unique_values\": 87,\n        \"samples\": [\n          0.64,\n
0.36,\n          0.24\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"residual sugar\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 5.072057784014864,\n        \"min\":
0.6,\n        \"max\": 65.8,\n        \"num_unique_values\": 310,\n
\"samples\": [\n          15.5,\n          19.25,\n          3.3\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"chlorides\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.02184796809372882,\n        \"min\": 0.009,\n        \"max\":
0.346,\n        \"num_unique_values\": 160,\n        \"samples\": [\n
0.167,\n          0.133,\n          0.015\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"free sulfur dioxide\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
17.007137325232566,\n        \"min\": 2.0,\n        \"max\": 289.0,\n
\"num_unique_values\": 132,\n        \"samples\": [\n          24.0,\n
122.5,\n          7.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"total sulfur dioxide\",\n      \"properties\": {\n        \"dtype\":
```

\"number\",\n   \"std\": 42.49806455414294,\n   \"min\":
9.0,\n   \"max\": 440.0,\n   \"num_unique_values\": 251,\n
\"samples\": [\n   260.0,\n   63.0,\n   70.0\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"density\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0.0029909069169369354,\n   \"min\": 0.98711,\n   \"max\":
1.03898,\n   \"num_unique_values\": 890,\n   \"samples\": [\
n   0.99362,\n   0.99388,\n   0.9929\n   ],\
n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"pH\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0.1510005996150667,\n
\"min\": 2.72,\n   \"max\": 3.82,\n   \"num_unique_values\":
103,\n   \"samples\": [\n   3.34,\n   3.41,\n
3.49\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"sulphates\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0.11412583394883138,\n   \"min\":
0.22,\n   \"max\": 1.08,\n   \"num_unique_values\": 79,\n
\"samples\": [\n   0.41,\n   0.45,\n   0.46\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"alcohol\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
1.2306205677573183,\n   \"min\": 8.0,\n   \"max\": 14.2,\n
\"num_unique_values\": 103,\n   \"samples\": [\n   12.6,\n
11.3666666666667,\n   10.0333333333333\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"quality\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\": 0,\n
\"min\": 3,\n   \"max\": 9,\n   \"num_unique_values\": 7,\n
\"samples\": [\n   6,\n   5,\n   3\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4898 entries, 0 to 4897
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         4898 non-null   float64
 1   volatile acidity      4898 non-null   float64
 2   citric acid           4898 non-null   float64
 3   residual sugar        4898 non-null   float64
 4   chlorides             4898 non-null   float64
 5   free sulfur dioxide   4898 non-null   float64
 6   total sulfur dioxide  4898 non-null   float64
 7   density               4898 non-null   float64
 8   pH                    4898 non-null   float64
```

```
 9   sulphates             4898 non-null   float64
 10  alcohol               4898 non-null   float64
 11  quality               4898 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 459.3 KB

features = df.drop(columns=['quality'])

scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)

pca = PCA(n_components=2)

features_pca = pca.fit_transform(features_scaled)

kmeans_pca = KMeans(n_clusters=4, random_state=42)
labels_pca = kmeans_pca.fit_predict(features_pca)

plt.scatter(features_pca[:, 0], features_pca[:, 1], c=labels_pca,
cmap='viridis', s=50)
plt.xlabel('Glowna skladowa 1')
plt.ylabel('Glowna skladowa 2')
plt.title('Wizualizacja skupien po redukcji wymiarowosci (PCA)')
plt.colorbar(label='Klaster')
plt.show()
```




```
#PORÓWNANIE WYNIKÓW PRZED I PO REDUKCJI WIELOWYMIAROWOŚCI - KORZYSTAMY
ZE WSPÓŁCZYNNIKA SILHOUETTE
from sklearn.metrics import silhouette_score

silhouette_pca = silhouette_score(features_pca, labels_pca)
print(f"Silhouette Score (PCA-transformed data):
{silhouette_pca:.2f}")
```

```
Silhouette Score (PCA-transformed data): 0.38
```

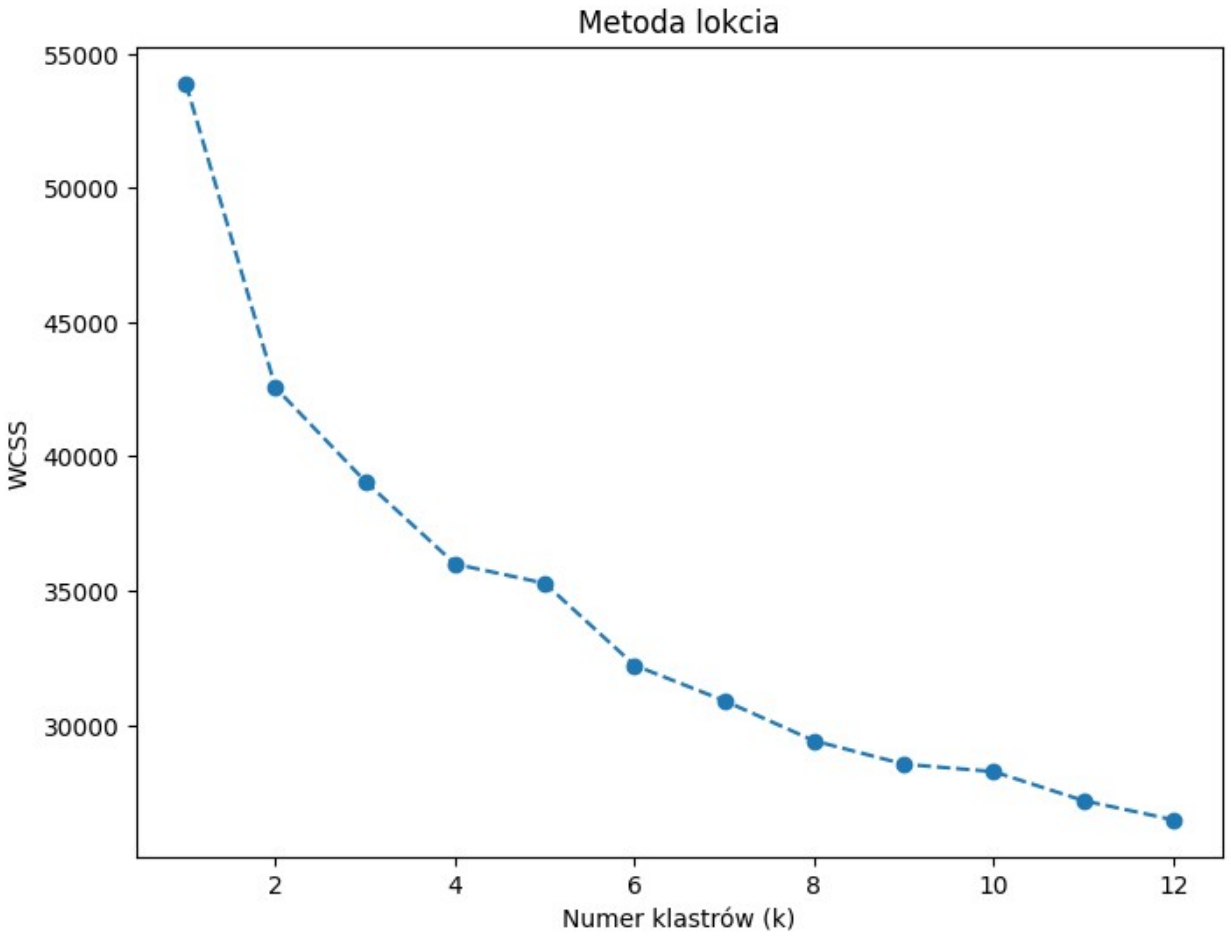
```
kmeans_original = KMeans(n_clusters=3, random_state=42)
labels_original = kmeans_original.fit_predict(features_scaled)
silhouette_original = silhouette_score(features_scaled,
labels_original)

print(f"Silhouette Score (original data): {silhouette_original:.2f}")
```

```
Silhouette Score (original data): 0.14
```

2. METODY NIEHIERARCHICZNE

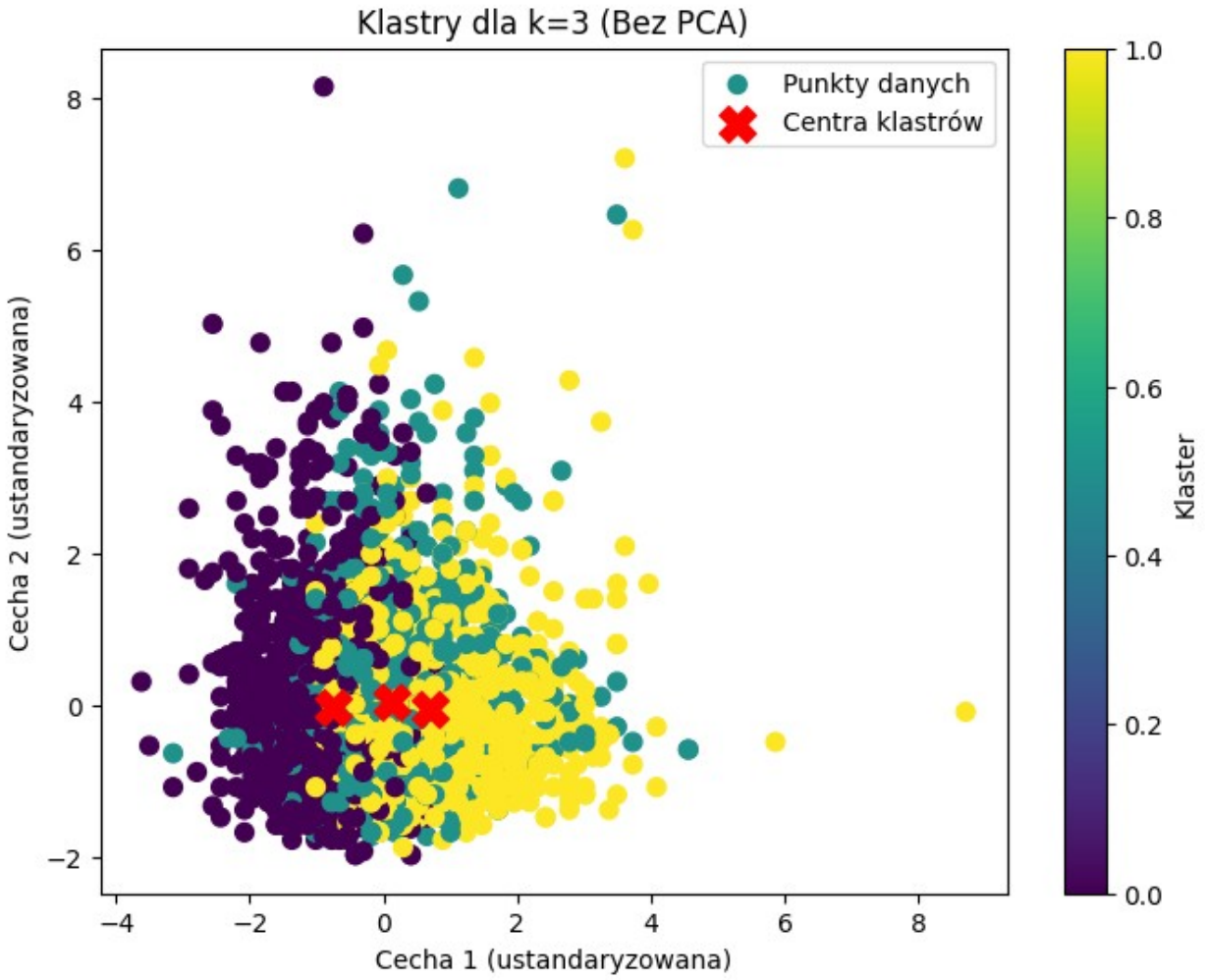
```
wcss = []
k_values = range(1, 13)
for k in k_values:
```

```
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(features_scaled)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(8, 6))
plt.plot(k_values, wcss, marker='o', linestyle='--')
plt.title('Metoda lokcia')
plt.xlabel('Numer klastrów (k)')
plt.ylabel('WCSS')
plt.show()
```



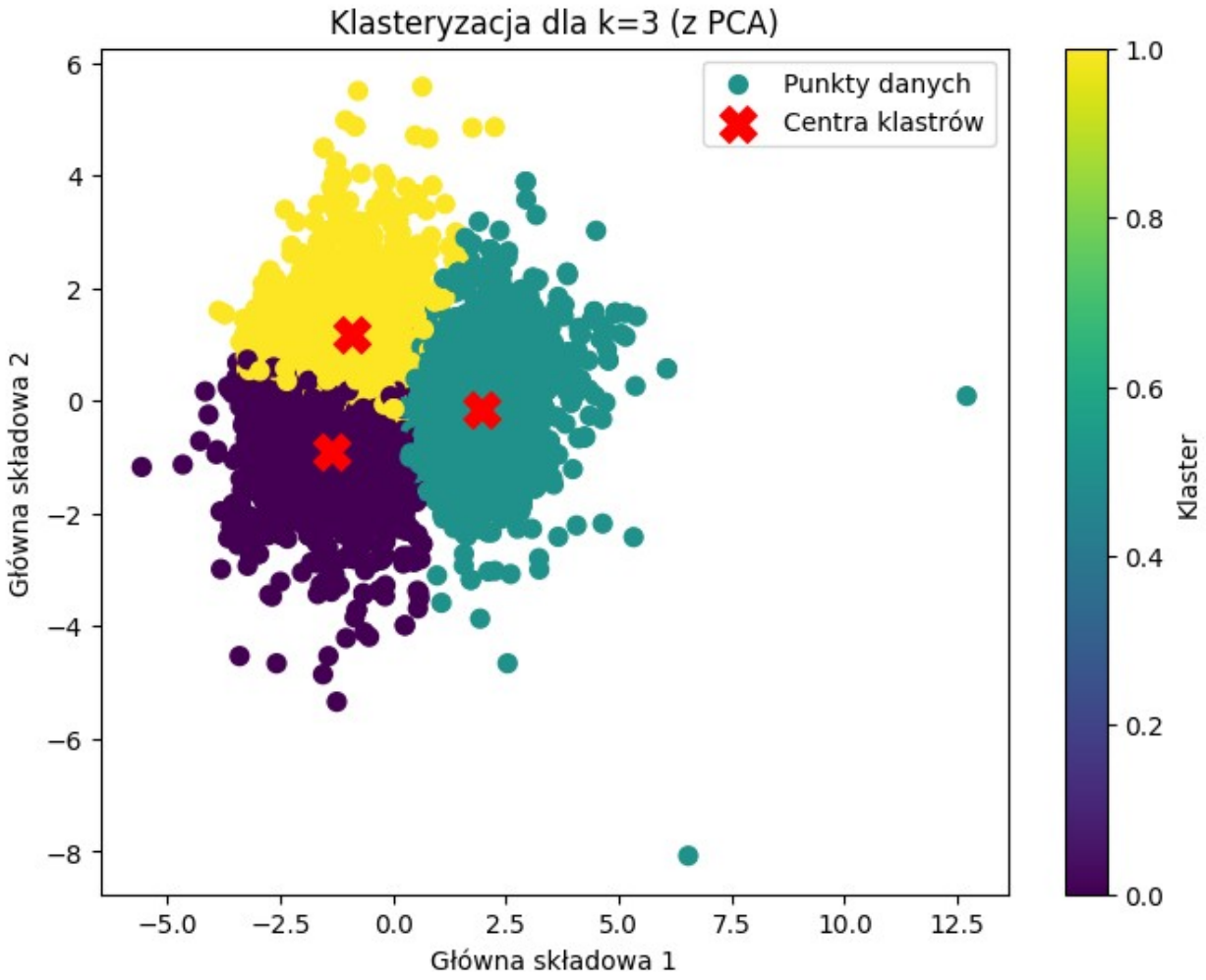

```
#wybieramy k=3

optimal_k = 3

kmeans_optimal = KMeans(n_clusters=optimal_k, random_state=42)
labels_optimal = kmeans_optimal.fit_predict(features_scaled)

plt.figure(figsize=(8, 6))
plt.scatter(features_scaled[:, 0], features_scaled[:, 1],
c=labels_optimal, cmap='viridis', s=50, label='Punkty danych')
```

```
plt.scatter(kmeans_optimal.cluster_centers_[:, 0],
kmeans_optimal.cluster_centers_[:, 1],
            c='red', marker='X', s=200, label='Centra klastrów')
plt.xlabel('Cecha 1 (ustandaryzowana)')
plt.ylabel('Cecha 2 (ustandaryzowana)')
plt.title(f'Klastry dla k={optimal_k} (Bez PCA)')
plt.colorbar(label='Klaster')
plt.legend()
plt.show()
```




```
kmeans_optimal = KMeans(n_clusters=optimal_k, random_state=42)
labels_optimal = kmeans_optimal.fit_predict(features_scaled)

pca = PCA(n_components=2)
features_pca = pca.fit_transform(features_scaled)

cluster_centers_pca = pca.transform(kmeans_optimal.cluster_centers_)
```

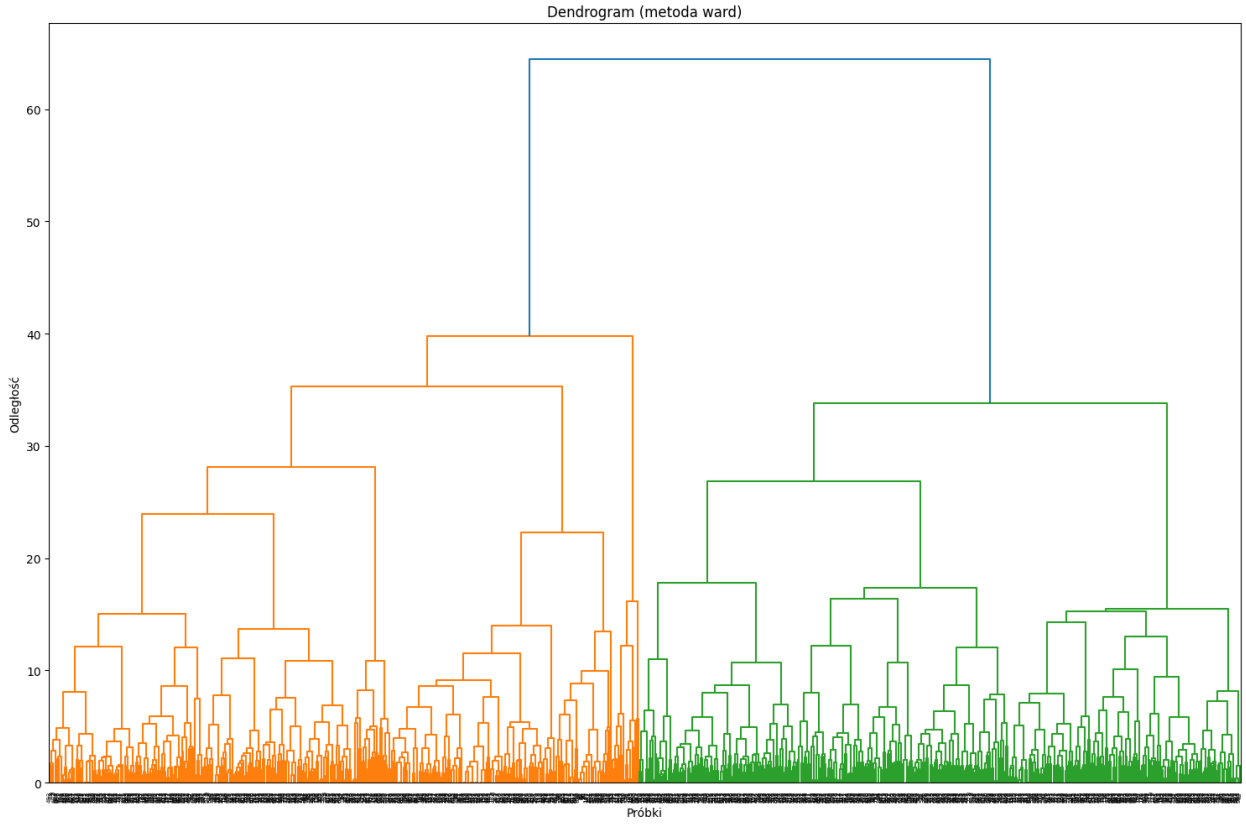
```
plt.figure(figsize=(8, 6))
plt.scatter(features_pca[:, 0], features_pca[:, 1], c=labels_optimal,
cmap='viridis', s=50, label='Punkty danych')
plt.scatter(cluster_centers_pca[:, 0], cluster_centers_pca[:, 1],
            c='red', marker='X', s=200, label='Centra klastrów')
plt.xlabel('Główna składowa 1')
plt.ylabel('Główna składowa 2')
plt.title(f'Klasteryzacja dla k={optimal_k} (z PCA)')
plt.colorbar(label='Klaster')
plt.legend()
plt.show()
```

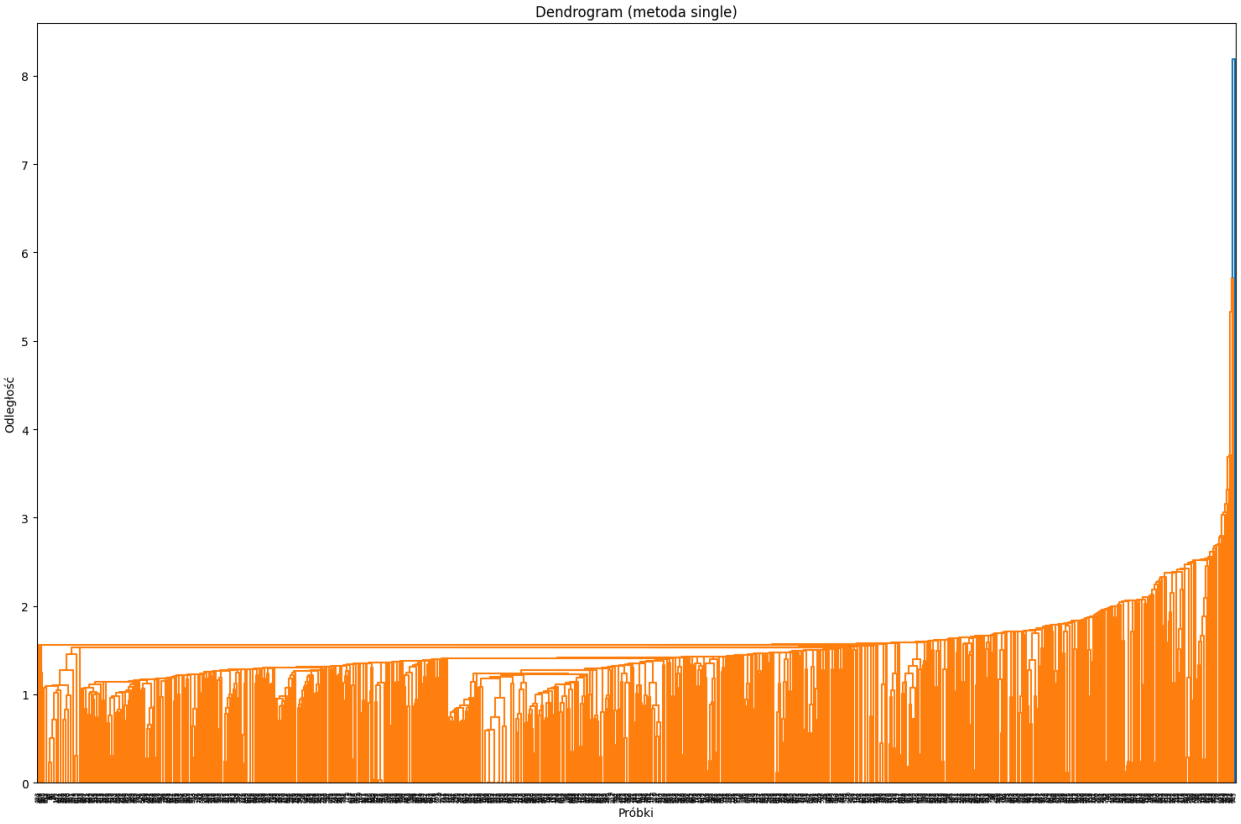


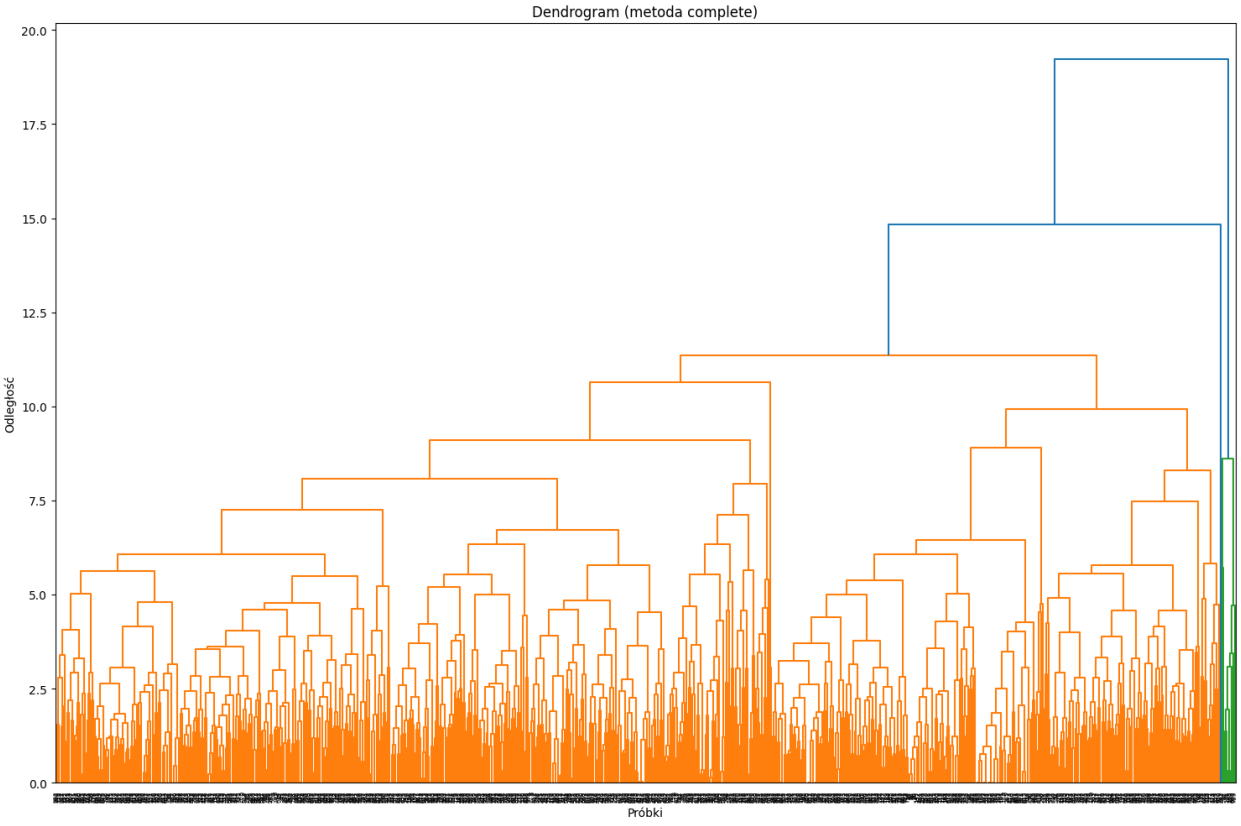

```
import scipy.cluster.hierarchy as sch
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster

cechy = df.drop(columns=['quality'])
scaler = StandardScaler()
dane_standaryzowane = scaler.fit_transform(cechy)
```

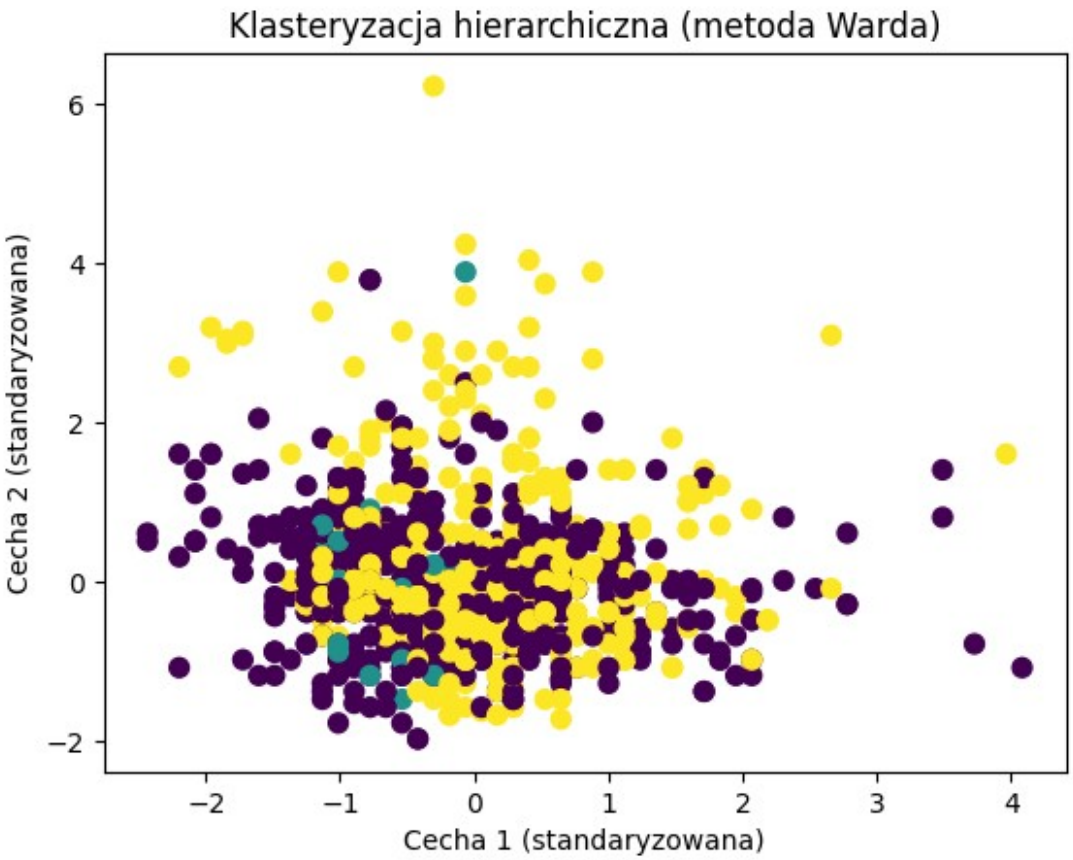
```
dane_standaryzowane = dane_standaryzowane[:1000]

metody_laczenia = ['ward', 'single', 'complete']


for i, metoda in enumerate(metody_laczenia):
    plt.figure(figsize=(15, 10))
    linked = linkage(dane_standaryzowane, method=metoda)
    dendrogram(linked, orientation='top', distance_sort='descending',
show_leaf_counts=False)
    plt.title(f'Dendrogram (metoda {metoda})')
    plt.xlabel('Próbki')
    plt.ylabel('Odległość')
    plt.tight_layout()
    plt.show()



linked = linkage(dane_standaryzowane, method='ward')
klastry_hierarchiczne = fcluster(linked, t=3, criterion='maxclust')

plt.scatter(dane_standaryzowane[:, 0], dane_standaryzowane[:, 1],
c=klastry_hierarchiczne, cmap='viridis', s=50)
plt.title('Klasteryzacja hierarchiczna (metoda Warda)')
plt.xlabel('Cecha 1 (standaryzowana)')
plt.ylabel('Cecha 2 (standaryzowana)')
plt.show()


kmeans = KMeans(n_clusters=3, random_state=42)
klastry_kmeans = kmeans.fit_predict(dane_standaryzowane)


plt.scatter(dane_standaryzowane[:, 0], dane_standaryzowane[:, 1],
c=klastry_kmeans, cmap='viridis', s=50)
plt.title('Klasteryzacja K-Means')
plt.xlabel('Cecha 1 (standaryzowana)')
plt.ylabel('Cecha 2 (standaryzowana)')
plt.show()


pca = PCA(n_components=2)
dane_zredukowane = pca.fit_transform(dane_standaryzowane)


plt.scatter(dane_zredukowane[:, 0], dane_zredukowane[:, 1],
c=klastry_hierarchiczne, cmap='viridis', s=50)
plt.title('Klasteryzacja hierarchiczna na danych zredukowanych PCA')
plt.xlabel('Główna składowa 1')
plt.ylabel('Główna składowa 2')
plt.show()
```

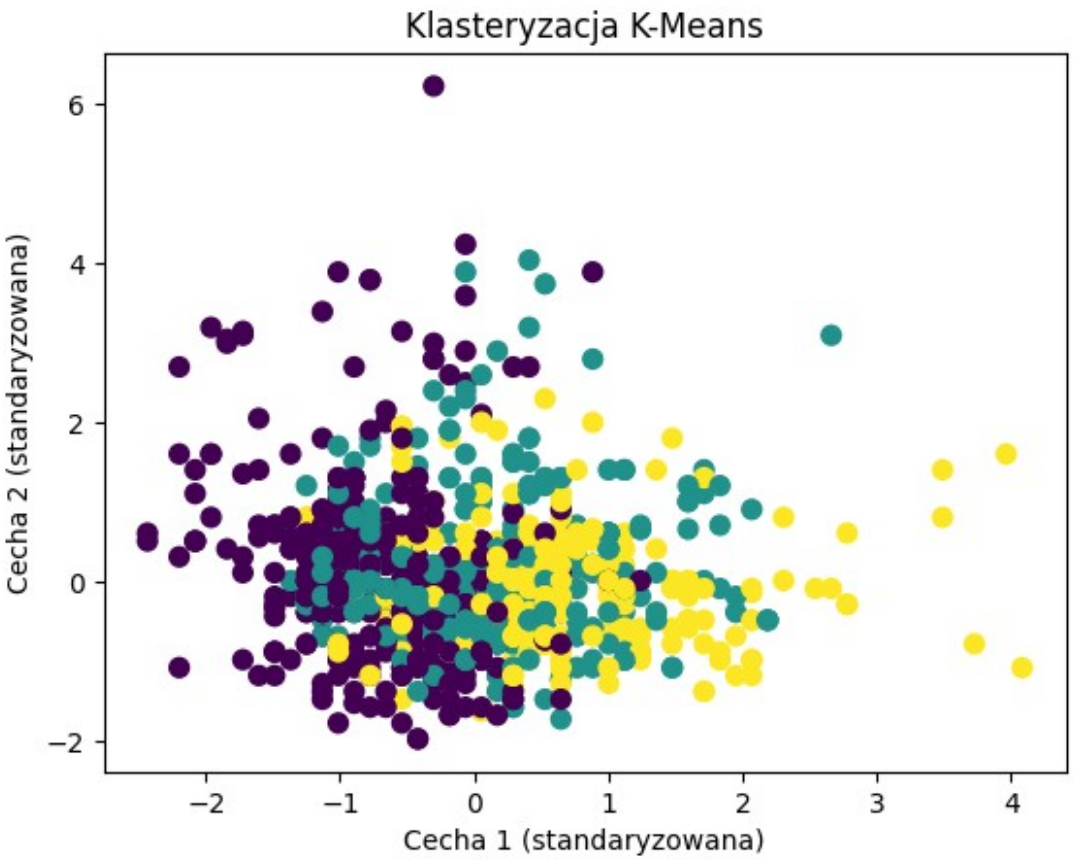
```
plt.scatter(dane_zredukowane[:, 0], dane_zredukowane[:, 1],
c=klastry_kmeans, cmap='viridis', s=50)
plt.title('Klasteryzacja K-Means na danych zredukowanych PCA')
plt.xlabel('Główna składowa 1')
plt.ylabel('Główna składowa 2')
plt.show()
```

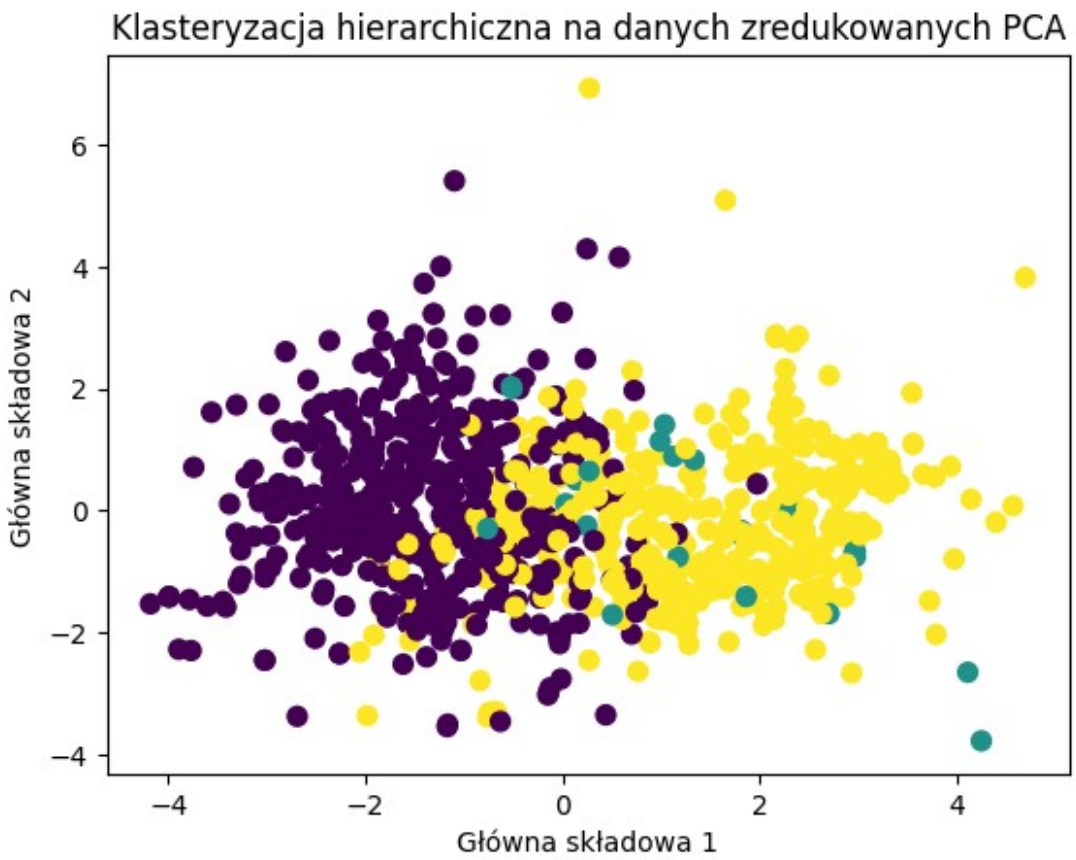


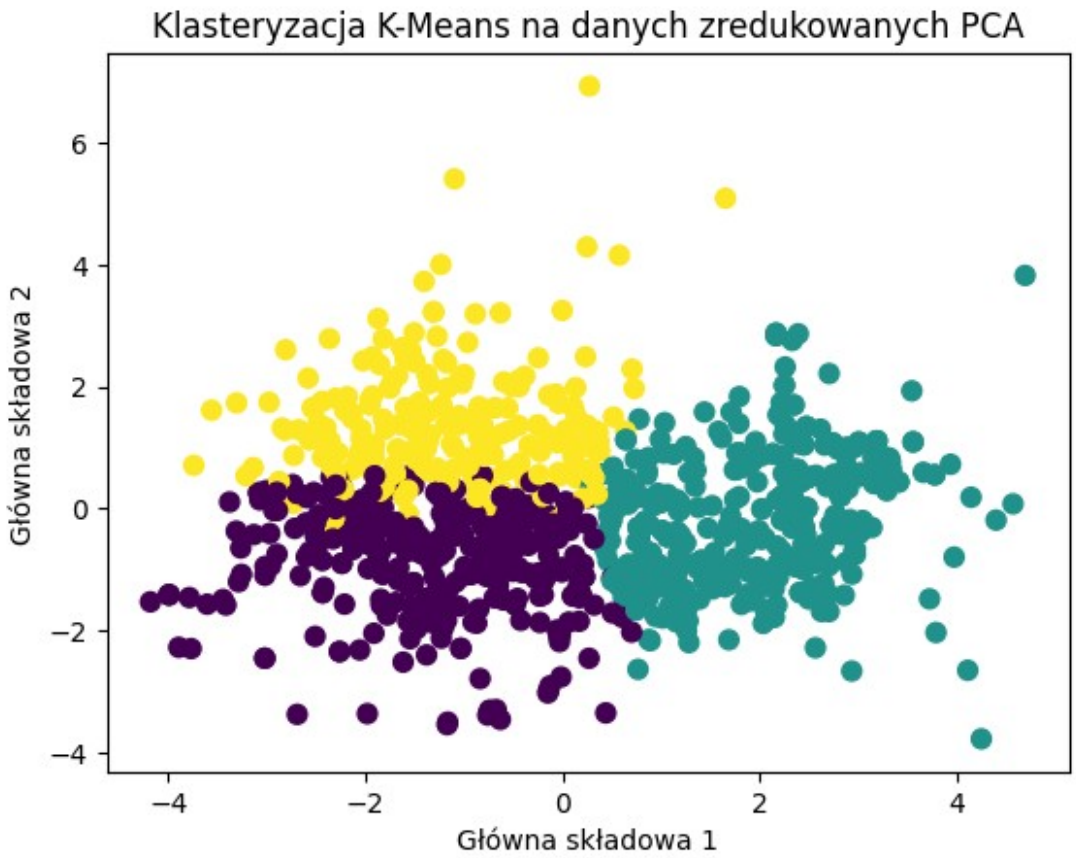

Dendrogram (metoda single)
Odległość
0
1
2
3
4
5
6
7
8
Próbki


Dendrogram (metoda complete)
Odległość
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Próbki


Klasteryzacja hierarchiczna (metoda Warda)
Cecha 2 (standaryzowana)
Cecha 1 (standaryzowana)
6
4
2
0
−2
−2
−1
0
1
2
3
4


Klasteryzacja K-Means
Cecha 2 (standaryzowana)
Cecha 1 (standaryzowana)
6
4
2
0
−2
−2
−1
0
1
2
3
4


Klasteryzacja hierarchiczna na danych zredukowanych PCA
Główna składowa 2
Główna składowa 1
6
4
2
0
−2
−4
−4
−2
0
2
4


Klasteryzacja K-Means na danych zredukowanych PCA
Główna składowa 2
Główna składowa 1
6
4
2
0
−2
−4
−4
−2
0
2
4