

Рубежный контроль №2

Работа Беловой Е.О., группа ИУ5-62Б, вариант 2 (задача №2, набор данных №2)

Задача

Кластеризуйте данные с помощью двух алгоритмов кластеризации. Алгоритмы для студентов группы ИУ5-62Б: MeanShift и иерархическая кластеризация. Сравните качество кластеризации с помощью следующих метрик качества кластеризации (если это возможно для Вашего набора данных):

1. Adjusted Rand index
2. Adjusted Mutual Information
3. Homogeneity, completeness, V-measure
4. Коэффициент силуэта

Сделайте выводы о том, какой алгоритм осуществляет более качественную кластеризацию на Вашем наборе данных. Набор данных №3: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_iris.html#sklearn.datasets.load_iris

Загрузка данных

```
In [1]: import numpy as np
import pandas as pd
from typing import Dict, Tuple
from scipy import stats
from IPython.display import Image
from sklearn import cluster, datasets, mixture
from sklearn.neighbors import kneighbors_graph
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import adjusted_rand_score
from sklearn.metrics import adjusted_mutual_info_score
from sklearn.metrics import homogeneity_completeness_v_measure
from sklearn.metrics import silhouette_score
from sklearn.cluster import MeanShift, AgglomerativeClustering
from itertools import cycle, islice
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
from sklearn.datasets import load_iris
```

```
In [2]: iris = load_iris()
for x in iris:
    print(x)
```

```
data
target
target_names
DESCR
feature_names
```

```
filename
```

```
In [3]: # Признаки
print(iris.feature_names)

['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

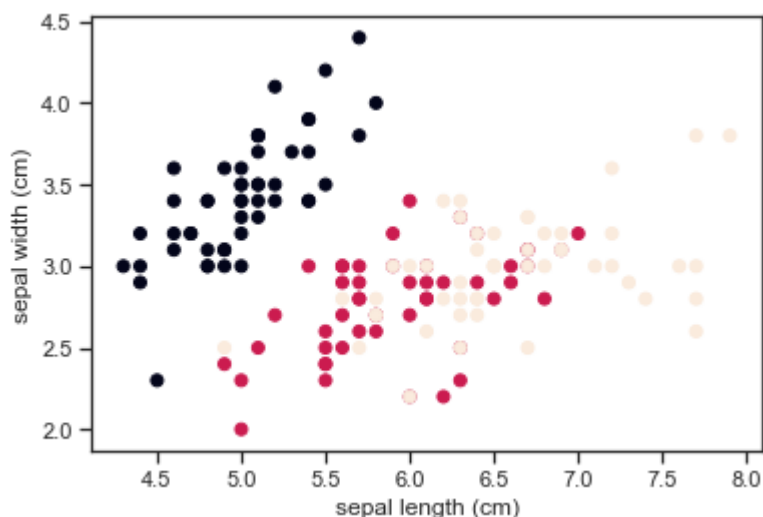
[illegible]

```
In [5]: # Имена меток
print(iris.target_names)

['setosa' 'versicolor' 'virginica']
```

```
In [6]: #Разделение набора данных
x_axis = iris.data[:, 0]
y_axis = iris.data[:, 1]
```

```
In [7]: # Построение
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])
plt.scatter(x_axis, y_axis, c=iris.target)
plt.show()
```



```
In [8]: data = pd.DataFrame(data= np.c_[iris.data[:, 0], iris.data[:, 1]],
                             columns= ['total phenols', 'color intensity'])
```

```
In [9]: data.head()
```

```
Out[9]:
```

| | | |
|---|-----|-----|
| 0 | 5.1 | 3.5 |
| 1 | 4.9 | 3.0 |
| 2 | 4.7 | 3.2 |
| 3 | 4.6 | 3.1 |
| 4 | 5.0 | 3.6 |

```
In [10]: data.head()
```

```
Out[10]:
```

| | total_phenols | color_intensity |
|---|---------------|-----------------|
| 0 | 5.1 | 3.5 |
| 1 | 4.9 | 3.0 |
| 2 | 4.7 | 3.2 |
| 3 | 4.6 | 3.1 |
| 4 | 5.0 | 3.6 |

```
In [11]: data.shape
```

```
Out[11]: (150, 2)
```

```
In [12]: def do_clustering(cluster_dataset, method):
        """
        Выполнение кластеризации для данных примера
        """
        temp_cluster = method.fit_predict(cluster_dataset)
        return temp_cluster
```

```
In [13]: import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

def cluster_metrics(method, data, true_y):
    """
    Вычисление метрик кластеризации
    """
    result_Method = do_clustering(data, method)

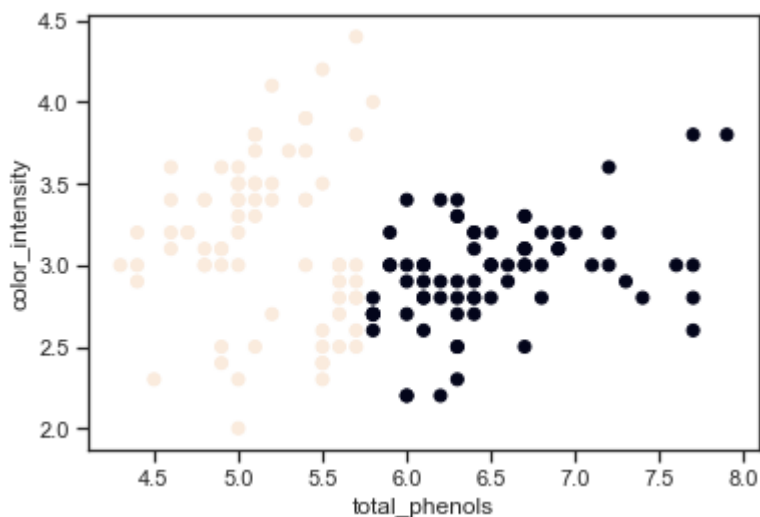
    list = []
    list.append(adjusted_rand_score(true_y, result_Method))
    list.append(adjusted_mutual_info_score(true_y, result_Method))
    h, c, v = homogeneity_completeness_v_measure(true_y, result_Method)
    list.append(h)
    list.append(c)
    list.append(v)
    list.append(silhouette_score(data, result_Method))

    names = ['ARI', 'AMI', 'Homogeneity', 'Completeness', 'V-measure', 'Silhouette']
    for i in range(0, 6):
        print('{}: {}'.format(names[i], list[i]))
```

MeanShift

```
In [14]: result_MeanShift = do_clustering(data, MeanShift())
```

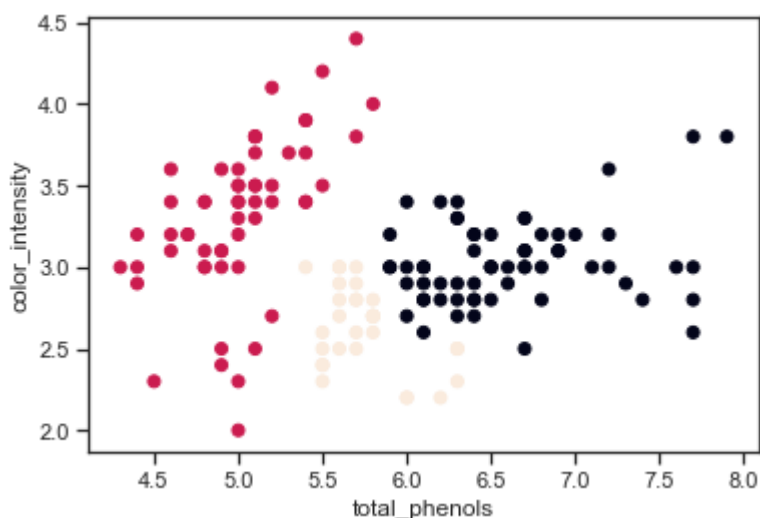
```
In [15]: plt.xlabel('total_phenols')
plt.ylabel('color_intensity')
plt.scatter(data['total_phenols'], data['color_intensity'], c=result_MeanShift)
plt.show()
```



Иерархическая кластеризация

```
In [16]: result_AgglomerativeClustering = do_clustering(data, AgglomerativeClustering(n_clusters=3))
```

```
In [17]: plt.xlabel('total_phenols')
plt.ylabel('color_intensity')
plt.scatter(data['total_phenols'], data['color_intensity'], c=result_AgglomerativeClustering)
plt.show()
```



Сравнение качества кластеризации

```
In [18]: cluster_metrics(MeanShift(), data, iris.target)
```

```
ARI: 0.3944401908806803;  
AMI: 0.43177435829008837;  
Homogeneity: 0.355574438925241;  
Completeness: 0.5636444355672562;  
V-measure: 0.43606057162569084;  
Silhouette: 0.4644681851183547;
```

```
In [19]: cluster_metrics(AgglomerativeClustering(n_clusters=3), data, iris.target  
)
```

```
ARI: 0.5112126489117526;  
AMI: 0.5240179186847518;  
Homogeneity: 0.5190720845536648;  
Completeness: 0.5414839345877656;  
V-measure: 0.5300412040588491;  
Silhouette: 0.3653346819163389;
```

Иерархическая кластеризация оказалась более качественной по сравнению с MeanShift.