Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Рубежный контроль №2
по дисциплине
«Методы машинного обучения»
Вариант №3

Выполнил:
студент группы ИУ5-23М
Умряев Д. Т.

Москва — 2020 г.

1. Задание

Для заданного набора данных решите задачу кластеризации с использованием методов 1) K-Means, 2) DBSCAN и 3) Birch. Оцените качество модели на основе подходящих метрик качества (не менее двух метрик, если это возможно). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? [1].

2. Решение

2.1. Импорт библиотек

Импортируем библиотеки с помощью команды import.

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
from sklearn.preprocessing import LabelEncoder
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 KMeans
from sklearn.cluster import DBSCAN
from sklearn.cluster import Birch
```

[2]:
```
warnings.simplefilter(action='ignore', category=FutureWarning)
```

Настроим отображение графиков [2, 3]:

[3]:
```
# Enable inline plots
%matplotlib inline

# Set plot style
sns.set(style="ticks")

# Set plots formats to save high resolution PNG
from IPython.display import set_matplotlib_formats
set_matplotlib_formats("retina")
```

Зададим ширину текстового представления данных, чтобы в дальнейшем текст в отчёте влезал на A4 [4]:

[4]:
```
pd.set_option("display.width", 70)
```

2.2. Загрузка данных

В качестве набора данных будем использовать датасет `iris` [5]:

[5]:
```
data = pd.read_csv('iris.csv')
```

2.3. Основные характеристики датасета

[6]:
```
data.head()
```

[6]:
```
   SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0            5.1           3.5            1.4           0.2
1            4.9           3.0            1.4           0.2
2            4.7           3.2            1.3           0.2
3            4.6           3.1            1.5           0.2
4            5.0           3.6            1.4           0.2

         Species
0  Iris-setosa
1  Iris-setosa
2  Iris-setosa
3  Iris-setosa
4  Iris-setosa
```

[7]:
```
data.dtypes
```

[7]:
```
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

[8]:
```
data.shape
```

[8]:
```
(150, 5)
```

[9]:
```
data['Species'].unique()
```

[9]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'],
  ↪dtype=object)
```

Выполним кодирование видов растений [6]:

[10]:
```
le = LabelEncoder()
data["Species"] = le.fit_transform(data["Species"])
```

Основные статистические характеристки набора данных:

[11]:
```
data.describe()
```

[11]:
```
       SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
count     150.000000    150.000000     150.000000    150.000000
mean        5.843333      3.054000       3.758667      1.198667
std         0.828066      0.433594       1.764420      0.763161
min         4.300000      2.000000       1.000000      0.100000
25%         5.100000      2.800000       1.600000      0.300000
50%         5.800000      3.000000       4.350000      1.300000
75%         6.400000      3.300000       5.100000      1.800000
max         7.900000      4.400000       6.900000      2.500000
```

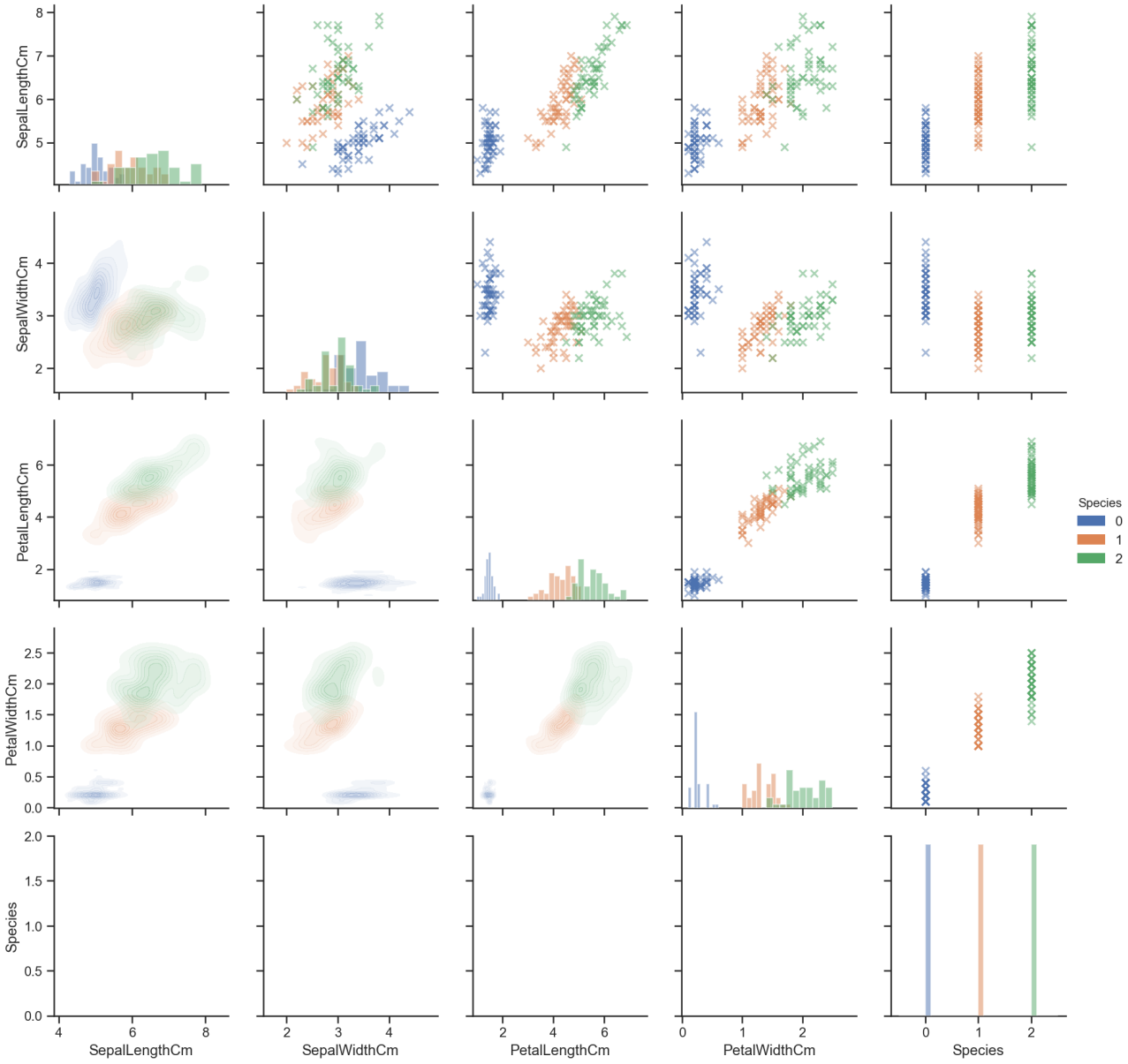
```
          Species
count  150.000000
mean     1.000000
std      0.819232
min      0.000000
25%      0.000000
50%      1.000000
75%      2.000000
max      2.000000
```

2.4. Визуальное исследование датасета

Парные диаграммы:

[12]:
```
g = sns.PairGrid(data, hue="Species")
g.map_diag(plt.hist, alpha=0.5)
g.map_upper(plt.scatter, alpha=0.5, marker='x')
g.map_lower(sns.kdeplot, shade=True, shade_lowest=False, alpha=0.3)
g.add_legend();
```

```
C:\Users\Deny\Anaconda3\lib\site-
packages\statsmodels\nonparametric\kernels.py:128: RuntimeWarning:
  ↪invalid value
encountered in true_divide
  return (1. / np.sqrt(2 * np.pi)) * np.exp(-(Xi - x)**2 / (h**2 * 2.))
C:\Users\Deny\Anaconda3\lib\site-packages\matplotlib\contour.py:1520:
UserWarning: Warning: converting a masked element to nan.
  self.zmax = float(z.max())
C:\Users\Deny\Anaconda3\lib\site-packages\matplotlib\contour.py:1521:
UserWarning: Warning: converting a masked element to nan.
  self.zmin = float(z.min())
C:\Users\Deny\Anaconda3\lib\site-packages\matplotlib\contour.py:1169:
RuntimeWarning: invalid value encountered in less
  under = np.nonzero(lev < self.zmin)[0]
C:\Users\Deny\Anaconda3\lib\site-packages\matplotlib\contour.py:1171:
RuntimeWarning: invalid value encountered in greater
  over = np.nonzero(lev > self.zmax)[0]
```

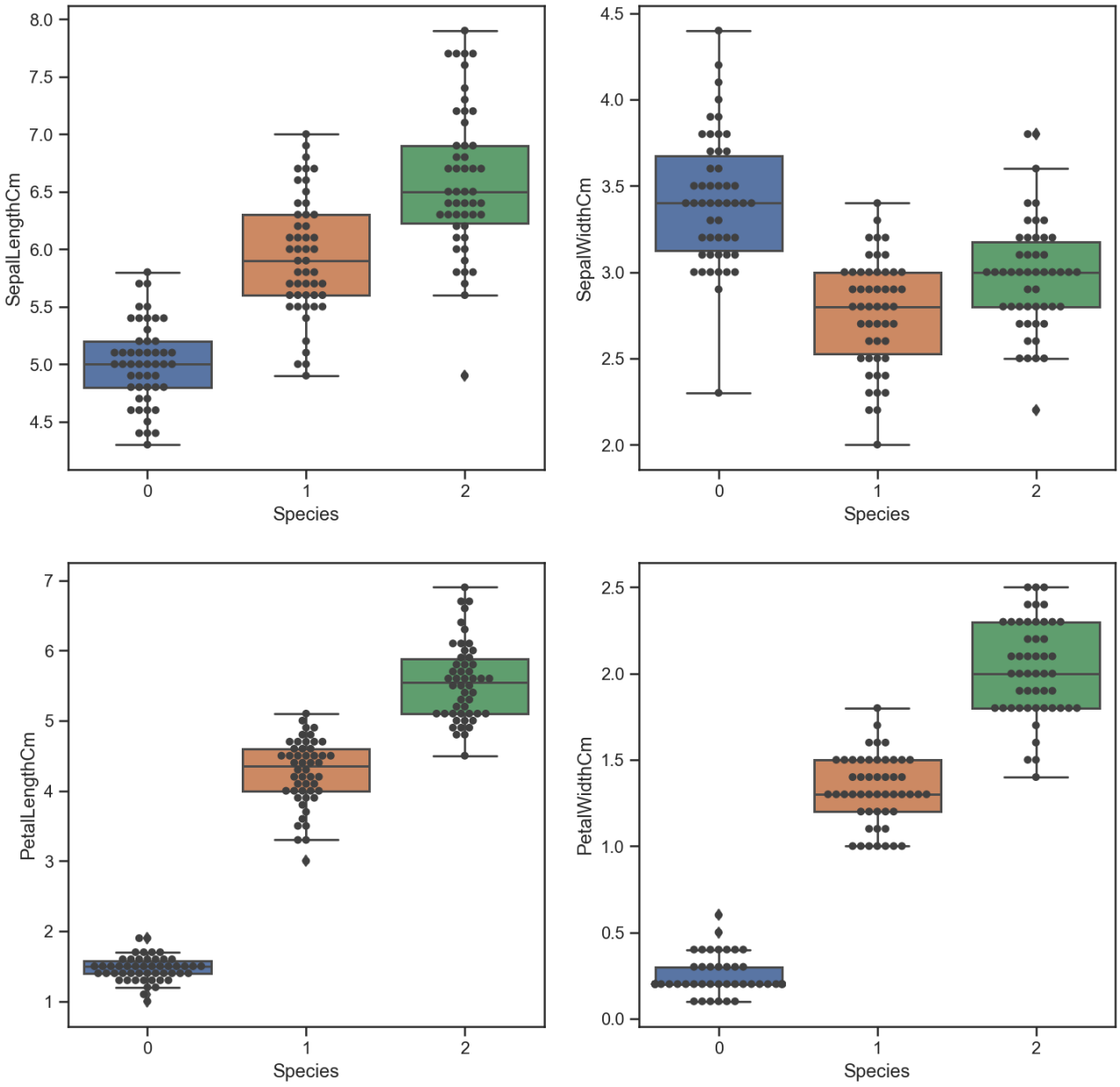


Ящики с усами:

[13]:
```
fig, axs = plt.subplots(ncols=2, nrows=2, figsize=(12, 12))

sns.boxplot(x="Species", y="SepalLengthCm", data=data, ax=axs[0, 0])
sns.swarmplot(x="Species", y="SepalLengthCm", data=data, color=".25",
              ax=axs[0, 0])

sns.boxplot(x="Species", y="SepalWidthCm", data=data, ax=axs[0, 1])
sns.swarmplot(x="Species", y="SepalWidthCm", data=data, color=".25",
              ax=axs[0, 1])

sns.boxplot(x="Species", y="PetalLengthCm", data=data, ax=axs[1, 0])
sns.swarmplot(x="Species", y="PetalLengthCm", data=data, color=".25",
              ax=axs[1, 0])
```

```
sns.boxplot(x="Species", y="PetalWidthCm", data=data, ax=axs[1, 1])
sns.swarmplot(x="Species", y="PetalWidthCm", data=data, color=".25",
              ax=axs[1, 1])
```

[13]: <matplotlib.axes._subplots.AxesSubplot at 0x195f6f02488>



2.5. Метрики качества кластеризации

1) Adjusted Rand index

Метрика применяется в том случае, если известны истинные метки классов. Для вычисления метрики используется функция adjusted_rand_score. Отчасти метрика напоминает accuracy, так как сравнивает полученные метки классов с известными истинными классами.

Метрика возвращает результат в диапазоне [-1;+1]. Значение близкое к +1 говорит об очень хорошем качестве кластеризации. Значение близкое к 0 соответствует случайным разбиениям. Отрицательные значения говорят о плохом качестве кластеризации.

2) Adjusted Mutual Information

Для вычисления метрики используется функция adjusted_mutual_info_score.
Значение близкое к +1 говорит об очень хорошем качестве кластеризации. Значение близкое к 0 соответствует случайным разбиениям.

3) Homogeneity, completeness, V-measure

Для вычисления метрик используется функция homogeneity_completeness_v_measure.

- Homogeneity - каждый кластер содержит только представителей единственного класса (под классом понимается истинное значение метки кластера). Значение в диапазоне [0;1], 1 говорит об очень хорошем качестве кластеризации.
- Completeness - все элементы одного класса помещены в один и тот же кластер. Значение в диапазоне [0;1], 1 говорит об очень хорошем качестве кластеризации.
- V-measure - среднее гармоническое от Homogeneity и Completeness.

4) Коэффициент силуэта

Для вычисления метрики используется функция silhouette_score
Данный метод не требует знания истинных значений меток кластеров.
Пусть:

- a - среднее расстояние между текущей точкой и другими точками этого же кластера.
- b - среднее расстояние между текущей точкой и другими точками следующего ближайшего кластера.

Тогда коэффициент силуэта для точки (объекта) определяется как:

$$s = \frac{b - a}{max(a, b)}$$

Силуэтом выборки называется средняя величина силуэта объектов данной выборки. Таким образом, силуэт показывает, насколько среднее расстояние до объектов своего кластера отличается от среднего расстояния до объектов других кластеров. Данная величина лежит в диапазоне [-1;1]. Значения, близкие к -1, соответствуют плохим (разрозненным) кластеризациям, значения, близкие к нулю, говорят о том, что кластеры пересекаются и накладываются друг на друга, значения, близкие к 1, соответствуют “плотным” четко выделенным кластерам. Таким образом, чем больше силуэт, тем более четко выделены кластеры, и они представляют собой компактные, плотно сгруппированные облака точек.

С помощью силуэта можно выбирать оптимальное число кластеров (если оно заранее неизвестно) — выбирается число кластеров, максимизирующее значение силуэта. В отличие от предыдущих метрик, силуэт зависит от формы кластеров, и достигает больших значений на более выпуклых кластерах, получаемых с помощью алгоритмов, основанных на восстановлении плотности распределения.

Будем использовать все вышеперечисленные метрики. Для этого напишем следующую функцию:

[14]:
```
def claster_metrics(method, cluster_dataset, cluster_true_y,
                    dataset_name):
    ari = []
    ami = []
    hl = []
    cl = []
    vl = []
    sl = []

    temp_cluster = method.fit_predict(cluster_dataset)
    ari.append(adjusted_rand_score(cluster_true_y, temp_cluster))
    ami.append(adjusted_mutual_info_score(cluster_true_y, temp_cluster))

    h, c, v = homogeneity_completeness_v_measure(cluster_true_y,
                                                 temp_cluster)
    hl.append(h)
    cl.append(c)
    vl.append(v)

    sl.append(silhouette_score(cluster_dataset, temp_cluster))

    result = pd.DataFrame({'Datasets':dataset_name,
                           'ARI':ari, 'AMI':ami,
                           'Homogeneity':hl,
                           'Completeness':cl,
                           'V-measure':vl, 'Silhouette':sl})
    return result
```

2.6. K-Means

[15]:
```
claster_metrics(KMeans(n_clusters=3),
                data[['SepalLengthCm', 'SepalWidthCm',
                      'PetalLengthCm', 'PetalWidthCm']],
                data['Species'], 'iris')
```

[15]:

	Datasets	ARI	AMI	Homogeneity	Completeness	V-measure	Silhouette
0	iris	0.730238	0.748372	0.751485	0.764986	0.758176	0.552592

2.7. DBSCAN

[16]:
```
claster_metrics(DBSCAN(eps=0.9),
                data[['SepalLengthCm', 'SepalWidthCm',
                      'PetalLengthCm', 'PetalWidthCm']],
                data['Species'], 'iris')
```

[16]:
```
  Datasets       ARI       AMI  Homogeneity  Completeness  \
0     iris  0.568116  0.576771      0.57938           1.0

   V-measure  Silhouette
0    0.73368    0.686393
```

2.8. Birch

[17]:
```
claster_metrics(Birch(),
                data[['SepalLengthCm', 'SepalWidthCm',
                      'PetalLengthCm', 'PetalWidthCm']],
                data['Species'], 'iris')
```

[17]:
```
  Datasets       ARI       AMI  Homogeneity  Completeness  \
0     iris  0.609625  0.670611     0.674706       0.73836

   V-measure  Silhouette
0   0.705099    0.501699
```

2.9. Выводы

В целом методы справились с задачей довольно хорошо. Лучше всех себя проявил метод `K-Means`. Методы `DBSCAN` и `Birch` справились с задачей чуть хуже.

Список литературы

[1] Гапанюк Ю. Е. Рубежный контроль №2 по дисциплине «Методы машинного обучения» [Электронный ресурс] // GitHub. — 2020. — Режим доступа: `https://github.com/ugapanyuk/ml_course_2020/wiki/MMO_RK_2` (дата обращения: 21.05.2020).

[2] Team The IPython Development. IPython 7.13.0 Documentation [Electronic resource] // Read the Docs. — 2020. — Access mode: `https://ipython.readthedocs.io/en/stable/` (online; accessed: 21.05.2020).

[3] Waskom M. seaborn 0.10.0 documentation [Electronic resource] // PyData. — 2020. — Access mode: `https://seaborn.pydata.org/` (online; accessed: 21.05.2020).

[4] pandas 1.0.1 documentation [Electronic resource] // PyData. — 2020. — Access mode: `http://pandas.pydata.org/pandas-docs/stable/` (online; accessed: 21.05.2020).

[5] Learning UCI Machine. Iris Species [Electronic resource] // Kaggle. — 2016. — Access mode: `https://www.kaggle.com/uciml/iris` (online; accessed: 21.05.2020).

[6] scikit-learn 0.22.2 documentation [Electronic resource]. — 2020. — Access mode: `https://scikit-learn.org/` (online; accessed: 21.05.2020).