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

Рубежный контроль №2
по дисциплине
«Технологии машинного обучения»

Выполнил:
студент группы ИУ5-63Б
Кривцов Н. А.

Москва — 2020 г.

1. Условие задания

Требуется провести кластерный анализ набора данных, с использованем алгоритмов **K-means++** и **Birch**. Для сравнения результатов работы обоих алгоритмов необходимо использовать следующие метрики:

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

2. Разведочный анализ и предобработка данных

[0]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import Birch, KMeans
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19:
FutureWarning: pandas.util.testing is deprecated. Use the functions in
  ↪the
public API at pandas.testing instead.
  import pandas.util.testing as tm
```

[0]:
```
df = pd.read_csv("/content/drive/My Drive/Colab Notebooks/
  ↪hotel_bookings.csv")
```

[0]:
```
# Типы признаков
df.dtypes
```

[0]:
```
hotel                           object
is_canceled                      int64
lead_time                        int64
arrival_date_year                int64
arrival_date_month              object
arrival_date_week_number         int64
arrival_date_day_of_month        int64
stays_in_weekend_nights          int64
stays_in_week_nights             int64
adults                           int64
children                       float64
babies                           int64
meal                            object
country                         object
market_segment                  object
distribution_channel            object
is_repeated_guest                int64
previous_cancellations           int64
```

```
previous_bookings_not_canceled      int64
reserved_room_type                 object
assigned_room_type                 object
booking_changes                     int64
deposit_type                       object
agent                             float64
company                           float64
days_in_waiting_list                int64
customer_type                      object
adr                               float64
required_car_parking_spaces         int64
total_of_special_requests           int64
reservation_status                 object
reservation_status_date            object
dtype: object
```

[0]:
```
# Размерность датасета
df.shape
```

[0]:
```
(119390, 32)
```

[0]:
```
df.isnull().sum()
```

[0]:
```
hotel                               0
is_canceled                         0
lead_time                           0
arrival_date_year                   0
arrival_date_month                  0
arrival_date_week_number            0
arrival_date_day_of_month           0
stays_in_weekend_nights             0
stays_in_week_nights                0
adults                              0
children                            4
babies                              0
meal                                0
country                           488
market_segment                      0
distribution_channel                0
is_repeated_guest                   0
previous_cancellations              0
previous_bookings_not_canceled      0
reserved_room_type                  0
assigned_room_type                  0
booking_changes                     0
deposit_type                        0
agent                           16340
company                        112593
days_in_waiting_list                0
customer_type                       0
adr                                 0
```

```
required_car_parking_spaces            0
total_of_special_requests              0
reservation_status                     0
reservation_status_date                0
dtype: int64
```

[0]:
```
df1 = df.drop(labels=["agent", "company"], axis=1)
df1 = df1.dropna()
df1.isnull().sum()
```

[0]:
```
hotel                           0
is_canceled                     0
lead_time                       0
arrival_date_year               0
arrival_date_month              0
arrival_date_week_number        0
arrival_date_day_of_month       0
stays_in_weekend_nights         0
stays_in_week_nights            0
adults                          0
children                        0
babies                          0
meal                            0
country                         0
market_segment                  0
distribution_channel            0
is_repeated_guest               0
previous_cancellations          0
previous_bookings_not_canceled  0
reserved_room_type              0
assigned_room_type              0
booking_changes                 0
deposit_type                    0
days_in_waiting_list            0
customer_type                   0
adr                             0
required_car_parking_spaces     0
total_of_special_requests       0
reservation_status              0
reservation_status_date         0
dtype: int64
```

Закодируем признак `arrival_date_month` упорядоченными целыми числами от 0 до 11 с помощью класса `sklearn.preprocessing.OrdinalEncoder`

[0]:
```
from sklearn.preprocessing import OrdinalEncoder
months = [
          "January",
          "February",
          "March",
          "April",
```

```
            "May",
            "June",
            "July",
            "August",
            "September",
            "October",
            "November",
            "December",
]

oe = OrdinalEncoder(categories=[months])
oe.fit(df1[["arrival_date_month"]])
encoded_months = oe.transform(df1[["arrival_date_month"]])
df2 = df1.copy()
df2["arrival_date_month"] = encoded_months
df1["arrival_date_month"].value_counts()
```

[0]:
```
August       13852
July         12628
May          11779
October      11095
April        11045
June         10927
September    10467
March         9739
February      8012
November      6752
December      6728
January       5874
Name: arrival_date_month, dtype: int64
```

[0]:
```
df2["arrival_date_month"].value_counts()
```

[0]:
```
7.0     13852
6.0     12628
4.0     11779
9.0     11095
3.0     11045
5.0     10927
8.0     10467
2.0      9739
1.0      8012
10.0     6752
11.0     6728
0.0      5874
Name: arrival_date_month, dtype: int64
```

[0]:
```
df2.dtypes
```

[0]:
```
hotel                           object
is_canceled                      int64
```

```
lead_time                         int64
arrival_date_year                 int64
arrival_date_month              float64
arrival_date_week_number          int64
arrival_date_day_of_month         int64
stays_in_weekend_nights           int64
stays_in_week_nights              int64
adults                            int64
children                        float64
babies                            int64
meal                             object
country                          object
market_segment                   object
distribution_channel             object
is_repeated_guest                 int64
previous_cancellations            int64
previous_bookings_not_canceled    int64
reserved_room_type               object
assigned_room_type               object
booking_changes                   int64
deposit_type                     object
days_in_waiting_list              int64
customer_type                    object
adr                             float64
required_car_parking_spaces       int64
total_of_special_requests         int64
reservation_status               object
reservation_status_date          object
dtype: object
```

[0]:
```
# Признаки, используемые для кластеризации
features_to_use = [
                    "arrival_date_month",
                    "arrival_date_week_number",
                    "stays_in_week_nights",
                    "stays_in_weekend_nights",
                    "adults",
                    "children",
                    "babies",
                    "is_repeated_guest",
                    "previous_cancellations",
                    "previous_bookings_not_canceled",
]
data = df2[features_to_use]
data.dtypes
```

[0]:
```
arrival_date_month              float64
arrival_date_week_number          int64
stays_in_week_nights              int64
stays_in_weekend_nights           int64
```

```
adults                              int64
children                          float64
babies                              int64
is_repeated_guest                   int64
previous_cancellations              int64
previous_bookings_not_canceled      int64
dtype: object
```

3. K-means++

[0]:
```
kmpp = KMeans(init="k-means++", n_clusters=2)
clustered_data = kmpp.fit_predict(data)
```

[0]:
```
clustered_data
```

[0]:
```
array([0, 0, 0, …, 0, 0, 0], dtype=int32)
```

[0]:
```
from sklearn.metrics import adjusted_rand_score,
 ↪adjusted_mutual_info_score, homogeneity_completeness_v_measure,
 ↪silhouette_score
```

[0]:
```
# Кластеризация на два кластера, в идеале соотвествующих признаку
 ↪"is_canceled"
print(adjusted_rand_score(df2["is_canceled"], clustered_data))
print(adjusted_mutual_info_score(df2["is_canceled"], clustered_data))
print(homogeneity_completeness_v_measure(df2["is_canceled"],
 ↪clustered_data))
# print(silhouette_score(data, clustered_data))
```

```
6.672116493180065e-05
1.747349294827852e-07
(6.553937761280808e-06, 6.246751223312576e-06, 6.396658613777868e-06)
```

4. Birch

[0]:
```
birch = Birch(n_clusters=2)
clustered_data = birch.fit_predict(data)
print(adjusted_rand_score(df2["is_canceled"], clustered_data))
print(adjusted_mutual_info_score(df2["is_canceled"], clustered_data))
print(homogeneity_completeness_v_measure(df2["is_canceled"],
 ↪clustered_data))
```

```
-1.173260477891877e-05
-6.218591663045837e-06
(9.887642271616516e-09, 9.442823200980936e-09, 9.66011480455824e-09)
```

В итоге оба алгоритма не справились с задачей кластеризации предложенного набора данных на два кластера по признаку ‘is_canceled’, т.к. их метрики чрезвычайно близки к нулевому значению, указывающему на случайное разбиение экземпляров данных на кластеры.