Часть 3. Задача кластеризации

Датасет: https://www.kaggle.com/datasets/arjunbhasin2013/ccdata

Для представленного датасета необходимо решить задачу кластеризации методом k-средних. При решении задачи необходимо ответить на следующие вопросы:

1. Как выбор начальных центров кластеров влияет на результат кластеризации

2. Какую метрику выбрать для оценки качества кластеризации?

Решение

Влияние выбора начальных центров кластеров на результат кластеризации: В алгоритме k-средних начальное расположение центров кластеров может существенно повлиять на конечный результат. Это связано с тем, что алгоритм итеративно оптимизирует положение центров кластеров, но может сойтись к локальному минимуму, а не к глобальному. Существуют различные стратегии для выбора начальных центров, такие как случайный выбор, использование алгоритма k-means++ для более разнесённого начального распределения и т.д.

Метрики оценки качества кластеризации: Существует несколько метрик для оценки качества кластеризации. Наиболее популярными являются:

- **Индекс Дэвиса-Боулдина (DBI):** Эта метрика основана на сравнении внутрикластерных расстояний с межкластерными. Низкие значения DBI указывают на лучшую кластеризацию.
- **Silhouette Score:** Оценивает, насколько хорошо объекты кластеризованы, сравнивая внутрикластерное расстояние с ближайшим кластером. Значения близкие к 1 указывают на хорошую кластеризацию.
- **Инерция (Sum of Squared Distances):** Измеряет сумму квадратов расстояний между точками и центрами их кластеров. Меньшие значения инерции указывают на лучшую кластеризацию, но важно избегать чрезмерной кластеризации.

```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score, davies_bouldin_score
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns

# Загрузка и предварительная обработка данных
file_path = '/mnt/data/CC GENERAL.csv'
data = pd.read_csv(file_path)

# Проверка первых нескольких строк данных для понимания структуры
data.head()
```

Результат

```
  CUST_ID      BALANCE  BALANCE_FREQUENCY  PURCHASES  ONEOFF_PURCHASES  \
0  C10001    40.900749           0.818182      95.40              0.00
1  C10002  3202.467416           0.909091       0.00              0.00
2  C10003  2495.148862           1.000000     773.17            773.17
3  C10004  1666.670542           0.636364    1499.00           1499.00
4  C10005   817.714335           1.000000      16.00             16.00

   INSTALLMENTS_PURCHASES  CASH_ADVANCE  PURCHASES_FREQUENCY  \
0                    95.4      0.000000             0.166667
1                     0.0   6442.945483             0.000000
2                     0.0      0.000000             1.000000
3                     0.0    205.788017             0.083333
4                     0.0      0.000000             0.083333

   ONEOFF_PURCHASES_FREQUENCY  PURCHASES_INSTALLMENTS_FREQUENCY  \
0                    0.000000                          0.083333
1                    0.000000                          0.000000
2                    1.000000                          0.000000
3                    0.083333                          0.000000
4                    0.083333                          0.000000

   CASH_ADVANCE_FREQUENCY  CASH_ADVANCE_TRX  PURCHASES_TRX  CREDIT_LIMIT  \
0                0.000000                 0              2        1000.0
1                0.250000                 4              0        7000.0
2                0.000000                 0             12        7500.0
3                0.083333                 1              1        7500.0
4                0.000000                 0              1        1200.0

      PAYMENTS  MINIMUM_PAYMENTS  PRC_FULL_PAYMENT  TENURE
0   201.802084        139.509787          0.000000      12
1  4103.032597       1072.340217          0.222222      12
2   622.066742        627.284787          0.000000      12
3     0.000000               NaN          0.000000      12
4   678.334763        244.791237          0.000000      12
```

```
# Удаление нерелевантного столбца и обработка пропущенных значений
data_cleaned = data.drop(columns=['CUST_ID'])
data_cleaned.fillna(data_cleaned.mean(), inplace=True)

# Масштабирование данных
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data_cleaned)

# Функция для кластеризации и оценки
def cluster_and_evaluate(data, n_clusters, init_method):
    kmeans = KMeans(n_clusters=n_clusters, init=init_method, n_init=10,
random_state=42)
    labels = kmeans.fit_predict(data)

    # Оценка кластеризации
    silhouette = silhouette_score(data, labels)
    dbi = davies_bouldin_score(data, labels)

    return silhouette, dbi

# Тестирование разных методов инициализации и числа кластеров
init_methods = ['k-means++', 'random']
n_clusters_options = [3, 4, 5, 6]

# Сохранение результатов
```

```
results = []

for init_method in init_methods:
    for n_clusters in n_clusters_options:
        silhouette, dbi = cluster_and_evaluate(data_scaled, n_clusters,
init_method)
        results.append({
            'init_method': init_method,
            'n_clusters': n_clusters,
            'silhouette_score': silhouette,
            'davies_bouldin_index': dbi
        })

results_df = pd.DataFrame(results)
results_df.sort_values(by='silhouette_score', ascending=False)
```