Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Лабораторна робота №2

Програмування ітелектуальних інформаційних систем

Виконав
студент групи ІП-11:
Панченко С. В.

Перевірила:
Баришич Л. М

Київ 2023

ЗМІСТ

1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Класифікація даних.

2 ЗАВДАННЯ

Метрики і спосіб виконання описані тут:
https://www.kaggle.com/code/prashant111/naive-bayes-classifier-in-python

Лабу можна виконати в онлайн-редакторах типу Google Collab.

1. Dataset1: /kaggle/input/adult-dataset/adult.csv'

Bayesian Classification + Support Vector Machine

Зробити предікшн двома вищезгаданими алгоритмами. Порівняти наступні метрики:

Recall, f1-score, Confusion matrix, accuracy score. Порівняти з нуль-гіпотезою і перевірити на оверфітинг. Пояснити результати.

2. Dataset2: https://www.kaggle.com/code/stieranka/k-nearest-neighbors

K nearest neighbours.

Те саме що і в 1 завданні, але порівнюємо між собою метрики. Euclidean, Manhattan, Minkowski. Кластери потрібно візуалізувати. Метрики аналогічно п.1

3. Dataset3: https://www.kaggle.com/code/nuhashafnan/cluster-analysis-kmeans-kmediod-agnes-birch-dbscan

Agnes,Birch,DBSCAN

Інші методи можна ігнорувати. Зняти метрики (Silhouette Coefficient, ARI, NMI. Можна з п.1-2), пояснити.

4. Dataset4: https://www.kaggle.com/code/datark1/customers-clustering-k-means-dbscan-and-ap

Affinity propagation.

Порівняти з k-means. Метрики - Silhouette Coefficient, ARI, NMI

У звіті до кожної задачі:

1 Візуалізувати кластери

2 Вивести метрики. Для кластерів - Silhouette Coefficient, ARI, NMI

3 Порівняння з нулем і перевірка на оверфіт.

4 Висновок.

SVM і AP можна виконати на будь-якому датасеті.

3 ВИКОНАННЯ

3.1Завдання третє

Для початку імпортуємо модулі. Завантажимо датафрейм та виведемо його вміст.

```
In [24]: import numpy as np
         import pandas as pd
         import matplotlib.pyplot as plt
         import seaborn as sns
         import os
         np.random.seed(10)

         from sklearn import cluster, datasets, mixture
         X1,Y1 = datasets.make_moons(n_samples=2000, noise=.09,random_state=10)
```

Рисунок 3.1.1 - Сутності

Тренуємо моделі та прогнузуємо результати.

```
In [25]: from sklearn.cluster import Birch
         from sklearn.cluster import DBSCAN
         from sklearn.cluster import AgglomerativeClustering

         birchmodel=Birch(n_clusters=2,threshold=0.5,branching_factor=100)
         y_birch=birchmodel.fit_predict(X1)

         agnesmodel = AgglomerativeClustering(n_clusters=2)
         y_agnes=birchmodel.fit_predict(X1)
```

Рисунок 3.1.2 - Тренування моделей та прогнозування результатів

Зобразимо результати кластеризації.

```
In [26]: plt.scatter(X1[:, 0], X1[:, 1], s=100, c=y_birch)
         plt.show()
```

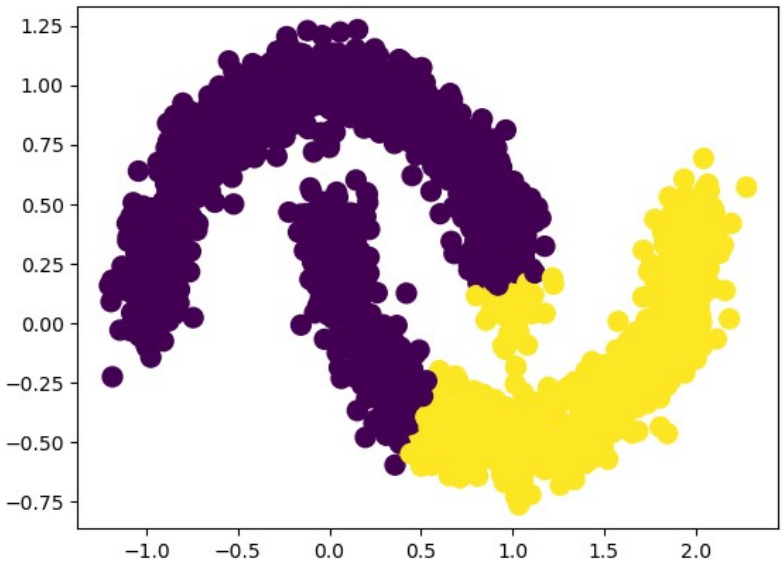

Рисунок 3.1.3 - кластеризація Birch

In [27]:
```
plt.scatter(X1[:, 0], X1[:, 1], s=100, c=y_agnes)
plt.show()
```

Рисунок 3.1.4 - кластеризація AGNES

Натренуємо модель DBSCAN.

In [28]:
```
def MyDBSCAN(D, eps, MinPts):
    labels = [0]*len(D)
    C = 0
    for P in range(0, len(D)):
        if not (labels[P] == 0):
           continue
        NeighborPts = regionQuery(D, P, eps)
        if len(NeighborPts) < MinPts:
            labels[P] = -1
        else:
           C += 1
           growCluster(D, labels, P, NeighborPts, C, eps, MinPts)

    return labels


def growCluster(D, labels, P, NeighborPts, C, eps, MinPts):
    labels[P] = C
    i = 0
    while i < len(NeighborPts):
        Pn = NeighborPts[i]
        if labels[Pn] == -1:
           labels[Pn] = C
        elif labels[Pn] == 0:
            labels[Pn] = C
            PnNeighborPts = regionQuery(D, Pn, eps)
            if len(PnNeighborPts) >= MinPts:
                NeighborPts = NeighborPts + PnNeighborPts
        i += 1


def regionQuery(D, P, eps):

    neighbors = []

    for Pn in range(0, len(D)):

        if np.linalg.norm(D[P] - D[Pn]) < eps:
           neighbors.append(Pn)

    return neighbors
```

Рисунок 3.1.5 - Тренування DBSCAN

Прогнозування DBSCAN.

```
In [32]: dbscanmodel = DBSCAN(eps=0.2, min_samples=70)
y_dbscan=dbscanmodel.fit_predict(X1)
plt.scatter(X1[:, 0], X1[:, 1], s=100, c=y_dbscan)
plt.show()
```

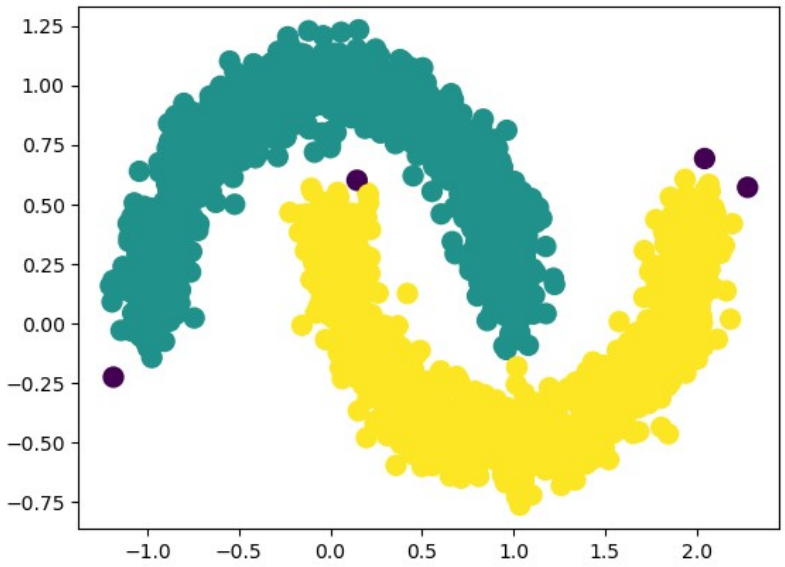

Рисунок 3.1.6 - кластеризація DBSCAN

Розрахуємо метрики ARI, NMI, Silhouette Coefficient для кожного методу.

```
In [30]: from sklearn.metrics.cluster import adjusted_rand_score
from sklearn.metrics.cluster import normalized_mutual_info_score
from sklearn.metrics.cluster import silhouette_score

for (name, pred) in [('Birch', y_birch), ('AGNES', y_agnes), ('DBSCAN', y_dbscan)]:
    print(f'ARI {name}: {adjusted_rand_score(Y1, pred)}')
    print(f'NMI {name}: {normalized_mutual_info_score(Y1, pred)}')
    print(f'Silhouette Coefficient {name}: {silhouette_score(X1, pred)}')
```

```
ARI Birch: 0.3767076067566142
NMI Birch: 0.341366173543779
Silhouette Coefficient Birch: 0.45835031870569487
ARI AGNES: 0.3767076067566142
NMI AGNES: 0.341366173543779
Silhouette Coefficient AGNES: 0.45835031870569487
ARI DBSCAN: 0.9920149895714532
NMI DBSCAN: 0.9787649300611727
Silhouette Coefficient DBSCAN: 0.3010813290557993
```

Рисунок 3.1.7 - Розраховані метрики ARI, NMI, Silhouette Coefficient

3.2 Пояснення результатів

Для Birch і AGNES обидві моделі мають подібні показники ARI і NMI, що вказує на помірну подібність і взаємну інформацію між їх кластеризацією.

Для DBSCAN він має значно вищі показники ARI та NMI, що свідчить про кращу відповідність і більшу згоду з справжніми кластеризаціями порівняно з Birch та AGNES.

Однак коефіцієнт силуету для DBSCAN нижчий, ніж у Birch і AGNES, що свідчить про те, що кластери в DBSCAN можуть бути не так добре розділені та можуть містити деякі точки, що перекриваються.

Підсумовуючи, оцінки ARI та NMI дають змогу зрозуміти подібність і взаємну інформацію між результатами кластеризації, тоді як коефіцієнт силуету оцінює якість і поділ кластерів. Різні моделі можуть перевершувати в різних аспектах, а вибір алгоритму кластеризації залежить від конкретних цілей і характеристик даних.

ДОДАТОК А ТЕКСТИ ПРОГРАМНОГО КОДУ

Тексти програмного коду

(Найменування програми (документа))

Жорсткий диск

(Вид носія даних)

(Обсяг програми (документа), арк.)

Студента групи ІП-113 курсу

Панченка С. В

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
np.random.seed(10)
from sklearn import cluster, datasets, mixture
X1,Y1 = datasets.make_moons(n_samples=2000, noise=.09,random_state=10)
from sklearn.cluster import Birch
from sklearn.cluster import DBSCAN
from sklearn.cluster import AgglomerativeClustering
birchmodel=Birch(n_clusters=2,threshold=0.5,branching_factor=100)
y_birch=birchmodel.fit_predict(X1)
agnesmodel = AgglomerativeClustering(n_clusters=2)
y_agnes=birchmodel.fit_predict(X1)
plt.scatter(X1[:, 0], X1[:, 1], s=100, c=y_birch)
plt.show()
plt.scatter(X1[:, 0], X1[:, 1], s=100, c=y_agnes)
plt.show()
def MyDBSCAN(D, eps, MinPts):
labels = [0]*len(D)
C = 0
for P in range(0, len(D)):
if not (labels[P] == 0):
continue
NeighborPts = regionQuery(D, P, eps)
if len(NeighborPts) < MinPts:
labels[P] = -1
else:
C += 1
growCluster(D, labels, P, NeighborPts, C, eps, MinPts)
return labels
def growCluster(D, labels, P, NeighborPts, C, eps, MinPts):
labels[P] = C
i = 0
while i < len(NeighborPts):
Pn = NeighborPts[i]
if labels[Pn] == -1:
labels[Pn] = C
elif labels[Pn] == 0:
labels[Pn] = C
```

```
      PnNeighborPts = regionQuery(D, Pn, eps)
      if len(PnNeighborPts) >= MinPts:
      NeighborPts = NeighborPts + PnNeighborPts
      i += 1
      def regionQuery(D, P, eps):
      neighbors = []
      for Pn in range(0, len(D)):
      if np.linalg.norm(D[P] - D[Pn]) < eps:
      neighbors.append(Pn)
      return neighbors
      dbscanmodel = DBSCAN(eps=0.2, min_samples=70)
      y_dbscan=dbscanmodel.fit_predict(X1)
      plt.scatter(X1[:, 0], X1[:, 1], s=100, c=y_dbscan)
      plt.show()
      from sklearn.metrics.cluster import adjusted_rand_score
      from sklearn.metrics.cluster import normalized_mutual_info_score
      from sklearn.metrics.cluster import silhouette_score
      for (name, pred) in [('Birch', y_birch), ('AGNES', y_agnes), ('DBSCAN',
y_dbscan)]:
      print(f'ARI {name}: {adjusted_rand_score(Y1, pred)}')
      print(f'NMI {name}: {normalized_mutual_info_score(Y1, pred)}')
      print(f'Silhouette Coefficient {name}: {silhouette_score(X1, pred)}')
```