|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра прикладной математики (ПМ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

по дисциплине «Технологии и инструментарий машинного обучения»

**Практическое занятие № 6**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент группы ИНБО-01-17 | *ИМБО-02-22, Ким Кирилл Сергеевич* | (подпись) | |
| Преподаватель | *Трушин Степан Михайлович, преподаватель* | (подпись) | |
| Отчет представлен | «\_\_\_»\_\_\_\_\_\_\_\_202\_\_г. | |  | |

Москва 2024 г.

**Цель занятия:**

Овладеть методами кластерного анализа с использованием различных алгоритмов кластеризации, таких как k-means, иерархическая кластеризация и DBSCAN. Научиться визуализировать результаты и оценивать качество кластеризации с помощью коэффициента силуэта и метода локтя.

**Постановка задачи:**

**Этап 1. Подготовка данных:**

1. Найти и загрузить набор данных для задачи классификации.

Листинг 1 – Загрузка данных

|  |
| --- |
| import pandas as pd  f = ('/content/wine-dataset-for-clustering/wine-clustering.csv')  data = pd.read\_csv(f)  data |

1. Провести необходимые шаги по предобработке данных:

* Провести очистку данных на повторяющие данные и пропущенные данные.

Листинг 2 – Очистка данных

|  |
| --- |
| data = data.drop\_duplicates()  data.fillna(data.mean(), inplace=True)  data = data.dropna() |

1. Провести нормализацию данных:

Листинг 3 – Нормализация данных

|  |
| --- |
| from sklearn.preprocessing import StandardScaler  scaler = StandardScaler()  normalized\_data = scaler.fit\_transform(data)  correlation\_matrix = data.corr()  print(correlation\_matrix)  Q1 = data.quantile(0.25)  Q3 = data.quantile(0.75)  IQR = Q3 - Q1  filtered\_data = data[~((data < (Q1 - 1.5 \* IQR)) | (data > (Q3 + 1.5 \* IQR))).any(axis=1)] |

**Этап 2. Реализация алгоритмов кластеризации:**

1. Алгоритм K-means.

Листинг 4 – Реализация алгоритма K-means

|  |
| --- |
| from sklearn.cluster import KMeans  kmeans = KMeans(n\_clusters=3)  kmeans.fit(data)  kmeans\_labels = kmeans.fit\_predict(data)  labels = kmeans.labels\_  centroids = kmeans.cluster\_centers\_ |

1. Иерархическая кластеризация

Листинг 5 – Код реализации иерархической кластеризации

|  |
| --- |
| from scipy.cluster.hierarchy import dendrogram, linkage  from sklearn.cluster import AgglomerativeClustering  import matplotlib.pyplot as plt  Z = linkage(data, method='average')  hierarchical = AgglomerativeClustering(n\_clusters=3)  hierarchical\_labels = hierarchical.fit\_predict(data)  plt.figure(figsize=(10, 7))  dendrogram(Z)  plt.show() |

1. Алгоритм DBSCAN

Листинг 6 – Код реализации DBSCAN

|  |
| --- |
| from sklearn.cluster import DBSCAN  dbscan = DBSCAN(eps=0.5, min\_samples=5)  dbscan\_labels = dbscan.fit\_predict(data)  dbscan.fit(data)  labels = dbscan.labels\_ |

**Этап 3. Определение количества кластеров**

1. Метод локтя

Листинг 7 – Код метода локтя

|  |
| --- |
| import matplotlib.pyplot as plt  from sklearn.cluster import KMeans  inertia = []  for k in range(1, 11):      kmeans = KMeans(n\_clusters=k)      kmeans.fit(data)      inertia.append(kmeans.inertia\_)  plt.figure(figsize=(8, 6))  plt.plot(range(1, 11), inertia, marker='o')  plt.title('Метод локтя')  plt.xlabel('Количество кластеров')  plt.ylabel('Инерция')  plt.show() |

1. Коэффициент силуэта

Листинг 8 – Код для расчёта коэффициента силуэта

|  |
| --- |
| from sklearn.metrics import silhouette\_score  import matplotlib.pyplot as plt  silhouette\_scores = []  for k in range(2, 11):      kmeans = KMeans(n\_clusters=k)      kmeans.fit(data)      labels = kmeans.labels\_      silhouette\_scores.append(silhouette\_score(data, labels))  plt.figure(figsize=(8, 6))  plt.plot(range(2, 11), silhouette\_scores, marker='o')  plt.title('Коэффициент силуэта')  plt.xlabel('Количество кластеров')  plt.ylabel('Средний коэффициент силуэта')  plt.show() |

**Этап 4. Визуализация иерархической кластеризации:**

1. Построить дендрограмму для иерархической кластеризации, чтобы проанализировать связи между объектами.

Листинг 9 – Код для построения дендрограммы

|  |
| --- |
| import scipy.cluster.hierarchy as sch  plt.figure(figsize=(10, 7))  dendrogram = sch.dendrogram(sch.linkage(data, method='ward'))  plt.title('Дендрограмма')  plt.xlabel('Объекты')  plt.ylabel('Расстояние')  plt.show() |

**Этап 5. Построение графиков:**

1. Построение графиков для K-means

Листинг 10 – Визуализация кластеров K-means

|  |
| --- |
| import matplotlib.pyplot as plt  from sklearn.decomposition import PCA  from sklearn.cluster import KMeans  pca = PCA(n\_components=2)  pca\_data = pca.fit\_transform(data)  kmeans = KMeans(n\_clusters=3)  kmeans.fit(data)  labels = kmeans.labels\_  plt.figure(figsize=(8, 6))  plt.scatter(pca\_data[:, 0], pca\_data[:, 1], c=labels, cmap='viridis', marker='o')  plt.title('Результаты K-means кластеризации')  plt.xlabel('Первая главная компонента')  plt.ylabel('Вторая главная компонента')  plt.show() |

1. Построение графиков для иерархической кластеризации

Листинг 11 – Визуализация результатов иерархической кластеризации

|  |
| --- |
| from scipy.cluster.hierarchy import fcluster  clusters = fcluster(Z, 3, criterion='maxclust')  plt.figure(figsize=(8, 6))  plt.scatter(pca\_data[:, 0], pca\_data[:, 1], c=clusters, cmap='plasma', marker='o')  plt.title('Результаты иерархической кластеризации')  plt.xlabel('Первая главная компонента')  plt.ylabel('Вторая главная компонента')  plt.show() |

1. Построение графиков для DBSCAN

Листинг 12 – Визуализация результатов DBSCAN

|  |
| --- |
| from sklearn.cluster import DBSCAN  dbscan = DBSCAN(eps=0.5, min\_samples=5)  dbscan.fit(data)  labels = dbscan.labels\_  plt.figure(figsize=(8, 6))  plt.scatter(pca\_data[:, 0], pca\_data[:, 1], c=labels, cmap='rainbow', marker='o')  plt.title('Результаты DBSCAN кластеризации')  plt.xlabel('Первая главная компонента')  plt.ylabel('Вторая главная компонента')  plt.show() |

1. Построение графика силуэта для K-means

Листинг 13 – Визуализация графика силуэта для K-means

|  |
| --- |
| from sklearn.metrics import silhouette\_samples, silhouette\_score  import numpy as np  kmeans = KMeans(n\_clusters=3)  labels = kmeans.fit\_predict(data)  silhouette\_vals = silhouette\_samples(data, labels)  plt.figure(figsize=(8, 6))  plt.barh(range(len(silhouette\_vals)), silhouette\_vals)  plt.title('График силуэта для кластеризации K-means')  plt.xlabel('Коэффициент силуэта')  plt.ylabel('Объекты')  plt.show() |

**Этап 6. Сравнение результатов:**

1. Сравнение по коэффициенту силуэта

Листинг 14 – Расчет среднего коэффициента силуэта для разных алгоритмов

|  |
| --- |
| from sklearn.metrics import silhouette\_score  from sklearn.cluster import KMeans, DBSCAN  # Для K-means  kmeans = KMeans(n\_clusters=3)  kmeans.fit(data)  kmeans\_labels = kmeans.labels\_  kmeans\_silhouette = silhouette\_score(data, kmeans\_labels)  # Для иерархической кластеризации  from scipy.cluster.hierarchy import fcluster  hierarchical\_labels = fcluster(Z, 3, criterion='maxclust')  hierarchical\_silhouette = silhouette\_score(data, hierarchical\_labels)  # Для DBSCAN  dbscan = DBSCAN(eps=0.5, min\_samples=5)  dbscan.fit(data)  dbscan\_labels = dbscan.labels\_  if len(np.unique(dbscan\_labels[dbscan\_labels != -1])) > 1:  # Исключаем шумовые точки      dbscan\_silhouette = silhouette\_score(data, dbscan\_labels)  else:      dbscan\_silhouette = None      print("DBSCAN: недостаточно уникальных меток для расчета силуэта.")  # Вывод средних коэффициентов силуэта для каждого алгоритма  print(f"K-means silhouette: {kmeans\_silhouette}")  print(f"Hierarchical silhouette: {hierarchical\_silhouette}")  print(f"DBSCAN silhouette: {dbscan\_silhouette}") |

1. Сравнение на основе устойчивости к параметрам

Листинг 14 – Тестирование устойчивости

|  |
| --- |
| dbscan\_1 = DBSCAN(eps=0.3, min\_samples=5)  dbscan\_1.fit(data)  dbscan\_labels\_1 = dbscan\_1.labels\_  # Проверка меток для первого DBSCAN  if len(np.unique(dbscan\_labels\_1[dbscan\_labels\_1 != -1])) > 1:  # Исключаем шумовые точки      dbscan\_silhouette\_1 = silhouette\_score(data, dbscan\_labels\_1)  else:      dbscan\_silhouette\_1 = None      print("DBSCAN с eps=0.3: недостаточно уникальных меток для расчета силуэта.")  dbscan\_2 = DBSCAN(eps=0.7, min\_samples=5)  dbscan\_2.fit(data)  dbscan\_labels\_2 = dbscan\_2.labels\_  # Проверка меток для второго DBSCAN  if len(np.unique(dbscan\_labels\_2[dbscan\_labels\_2 != -1])) > 1:  # Исключаем шумовые точки      dbscan\_silhouette\_2 = silhouette\_score(data, dbscan\_labels\_2)  else:      dbscan\_silhouette\_2 = None      print("DBSCAN с eps=0.7: недостаточно уникальных меток для расчета силуэта.")  print(f"Silhouette for DBSCAN with eps=0.3: {dbscan\_silhouette\_1}")  print(f"Silhouette for DBSCAN with eps=0.7: {dbscan\_silhouette\_2}") |

**Результат работы:**

Данную работу можете увидеть в блокноте Google Colab:

<https://colab.research.google.com/drive/1OAzH6KjABm8iyTUh96fOggj516ZTXRNc?usp=sharing>

**Вывод:**

**Список использованных источников и литературы:**

1. Ростовцев В.С. Искусственные нейронные сети,   
   Издательство "Лань", 2019. — 216 с. — URL: https://e.lanbook.com/book/122180
2. Араки М. Манга: Машинное обучение,   
   Издательство "ДМК Пресс", 2020. — 214 с. — URL: <https://e.lanbook.com/book/179473>
3. Платонов, А. В. Машинное обучение : учебное пособие для вузов / А. В. Платонов. — Москва : Издательство Юрайт, 2022. — 85 с. — (Высшее образование). — ISBN 978-5-534-15561-7. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/508804