МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)

Кафедра прикладной математики

ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 6

по дисциплине «Технологии и инструментарий анализа больших данных»

Выполнил студент группы ИКБО-20-21 Фомичев Р.А.

Проверил ассистент кафедры ПМ ИИТ Тетерин Н.Н.

Москва 2024

Задание 1

Найти данные для кластеризации. Данные в группе не должны повторяться. Если признаки в данных имеют очень сильно разные масштабы, то необходимо данные предварительно нормализовать. Код представлен на рисунке 1.

```
data = pd.read_csv("sample_data/obesity_data.csv")

orig_labels = data['NObeyesdad'].map({'Normal_Weight': 0, 'Overweight_Level_I': 1, 'Overweight_Level_II': 2,
 'Obesity_Type_I': 3, 'Insufficient_Weight': 4, 'Obesity_Type_II': 5, 'Obesity_Type_III': 6})

data['CAEC'] = data['CAEC'].map({ 'no': 0.0, 'Sometimes': 1.0,
                                 'Frequently': 2.0, 'Always': 3.0})
data['CALC'] = data['CALC'].map({ 'no': 0.0, 'Sometimes': 1.0,
                                 'Frequently': 2.0, 'Always': 3.0})
data['Gender'] = data['Gender'].map({"Male": 0.0, "Female": 1.0})
data['MTRANS'] = data['MTRANS'].map({'Walking': 0.0, 'Bike': 1.0,
                                     'Public_Transportation': 2.0,
                                     'Automobile': 3.0, 'Motorbike': 4.0})
data['family_history_with_overweight'] = data['family_history_with_overweight'].map(
    {"yes": 1.0, "no": 0.0})
data['FAVC'] = data['FAVC'].map({"yes": 1.0, "no": 0.0})
data['SMOKE'] = data['SMOKE'].map({"yes": 1.0, "no": 0.0})
data['SCC'] = data['SCC'].map({"yes": 1.0, "no": 0.0})
data = data.drop(columns=['NObeyesdad'])

scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)

tsne = TSNE(n_components=2, random_state=42)
tsne_results = tsne.fit_transform(scaled_data)
```

Рисунок 1 – Код программы

Задание 2

Провести кластеризацию данных с помощью алгоритма k-means. Использовать «правило локтя» и коэффициент силуэта для поиска оптимального количества кластеров. Код представлен на рисунке 2.

```
inertia = []
silhouette_scores = []

for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, random_state=1)
    kmeans.fit(scaled_data)
    inertia.append(kmeans.inertia_)
    silhouette_scores.append(silhouette_score(scaled_data, kmeans.labels_))

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(range(2, 11), inertia, marker='o')
plt.title('Правило локтя')
plt.xlabel('Количество кластеров')
plt.ylabel('Инерция')

plt.subplot(1, 2, 2)
plt.plot(range(2, 11), silhouette_scores, marker='o')
plt.title('Коэффициент силуэта')
plt.xlabel('Количество кластеров')
plt.ylabel('Силуэт')
plt.show()
```

Рисунок 2 – Код программы

Результат работы программы представлен на рисунке 3.

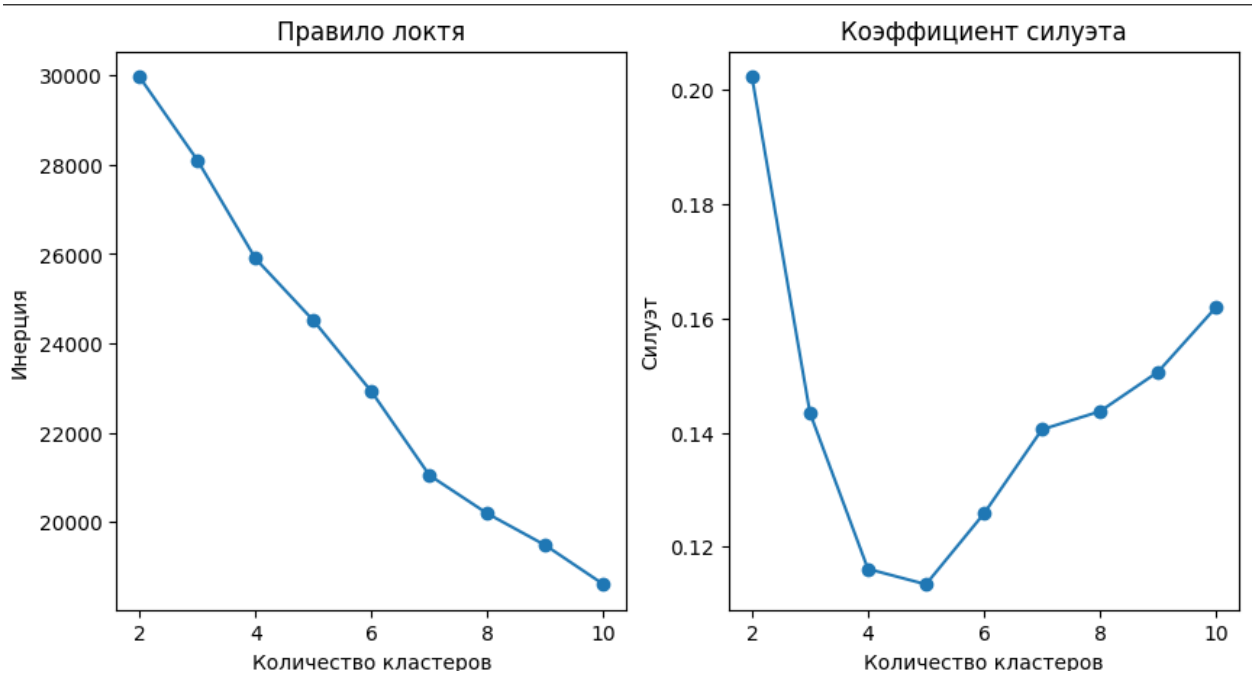

Рисунок 3 – Результат работы программы

Код представлен на рисунке 4.

```
kmeans = KMeans(n_clusters=7, random_state=1)
kmeans.fit(scaled_data)

labels_km = kmeans.predict(scaled_data)

plt.figure(figsize=(6, 4))
scatter = plt.scatter(tsne_results[:, 0], tsne_results[:, 1], c=labels_km, cmap='Accent', s=15)
plt.title('Кластеры K-means')
plt.xlabel('x')
plt.ylabel('y')
plt.colorbar(scatter, label='Шкала кластеров')
plt.show()
```

Рисунок 4 – Код программы

Результат работы представлен на рисунке 5.

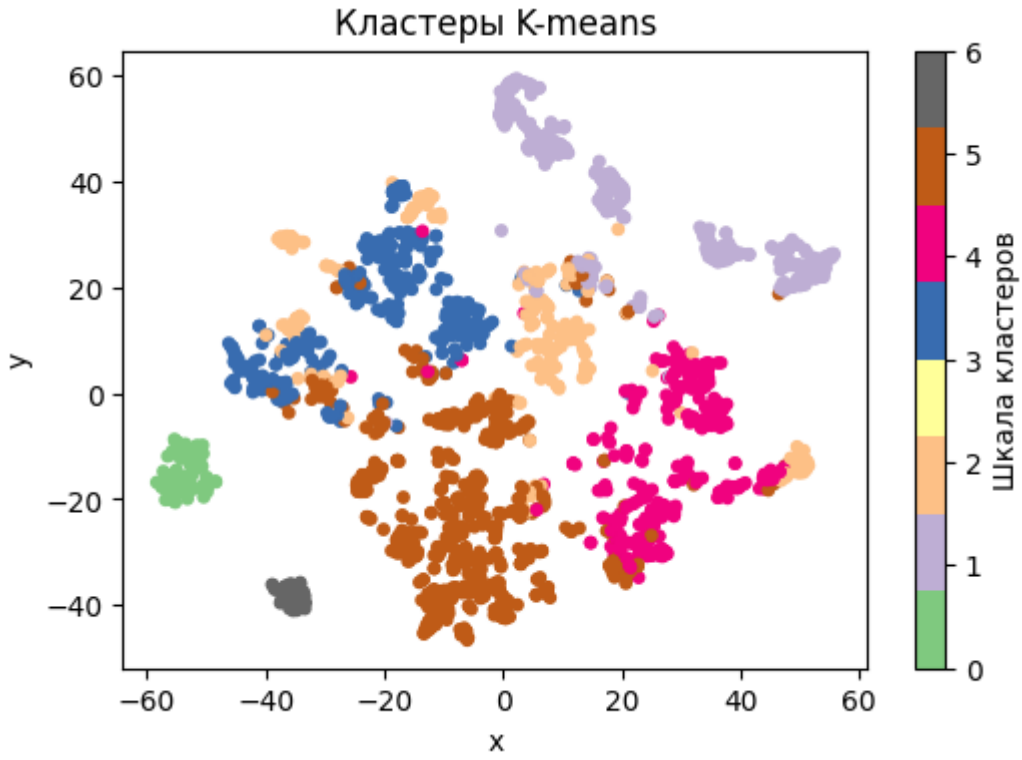

Рисунок 5 – Результат работы программы

Задание 3

Провести кластеризацию данных с помощью алгоритма иерархической кластеризации. Код представлен на рисунке 6.

```
plt.figure(figsize=(10, 5))
dendrogram = sch.dendrogram(sch.linkage(scaled_data, method='ward'))
plt.title('Дендрограмма')
plt.show()

hierarchical_clustering = AgglomerativeClustering(n_clusters=7)
labels_hc = hierarchical_clustering.fit_predict(scaled_data)

plt.figure(figsize=(6, 4))
scatter = plt.scatter(tsne_results[:, 0], tsne_results[:, 1], c=labels_hc, cmap='Accent', s=15)
plt.title('Кластеры иерархической кластеризации')
plt.xlabel('x')
plt.ylabel('y')
plt.colorbar(scatter, label='Шкала кластеров')
plt.show()
```

Рисунок 6 – Код программы

Результат работы программы представлен на рисунке 7.

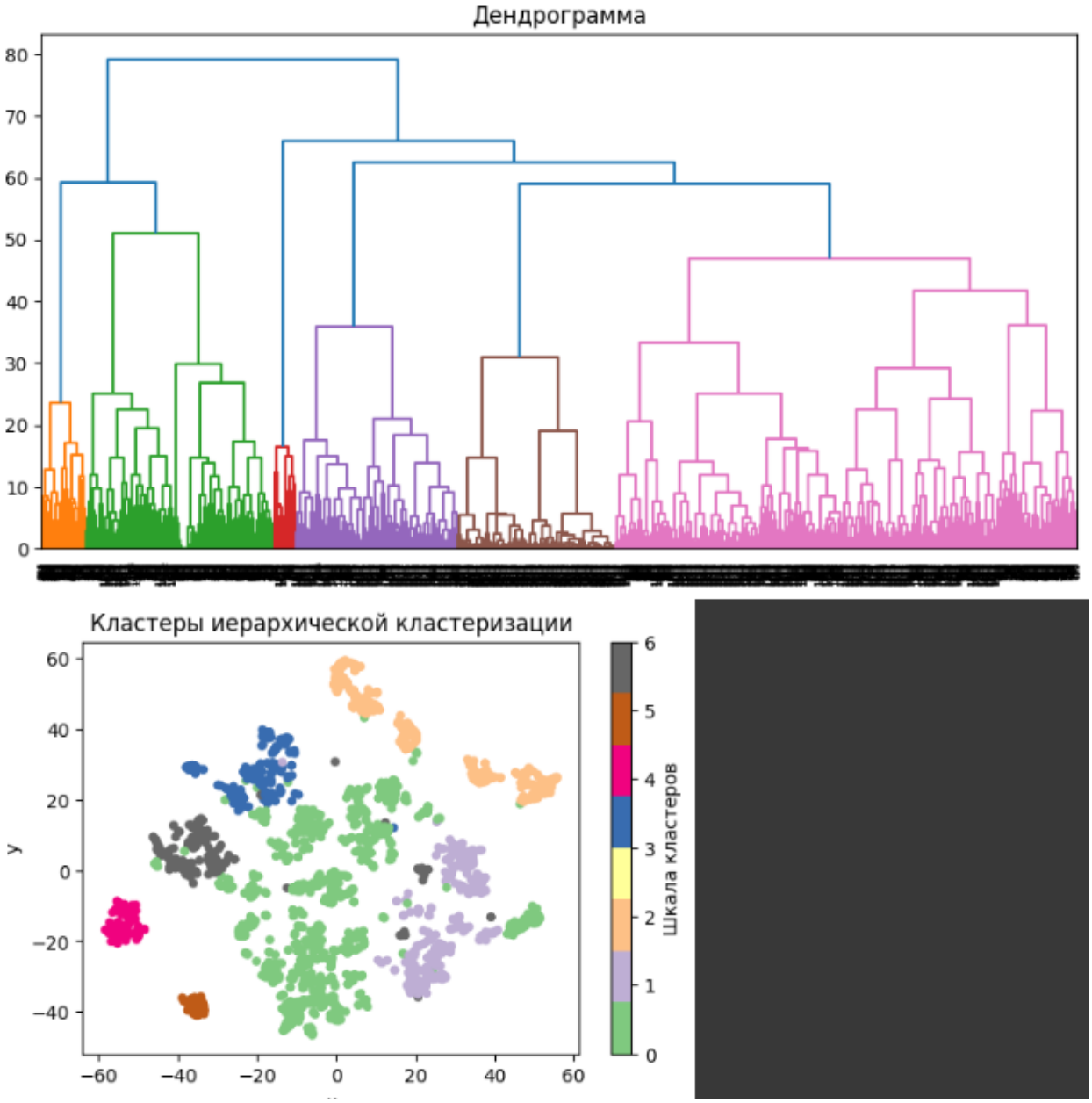

Рисунок 7 – Результат работы программы

Задание 4

Провести кластеризацию данных с помощью алгоритма DBSCAN. Код программы представлен на рисунке 8.

```
dbscan = DBSCAN(eps=2, min_samples=25)
dbscan.fit(scaled_data)
labels_db = dbscan.labels_

plt.figure(figsize=(6, 4))
scatter = plt.scatter(tsne_results[:, 0], tsne_results[:, 1], c=labels_db, cmap='Accent', s=15)
plt.title('Кластеры DBSCAN')
plt.xlabel('x')
plt.ylabel('y')
plt.colorbar(scatter, label='Шкала кластеров')
plt.show()
```

Рисунок 8 – Код программы

Результат работы программы представлен на рисунке 9.

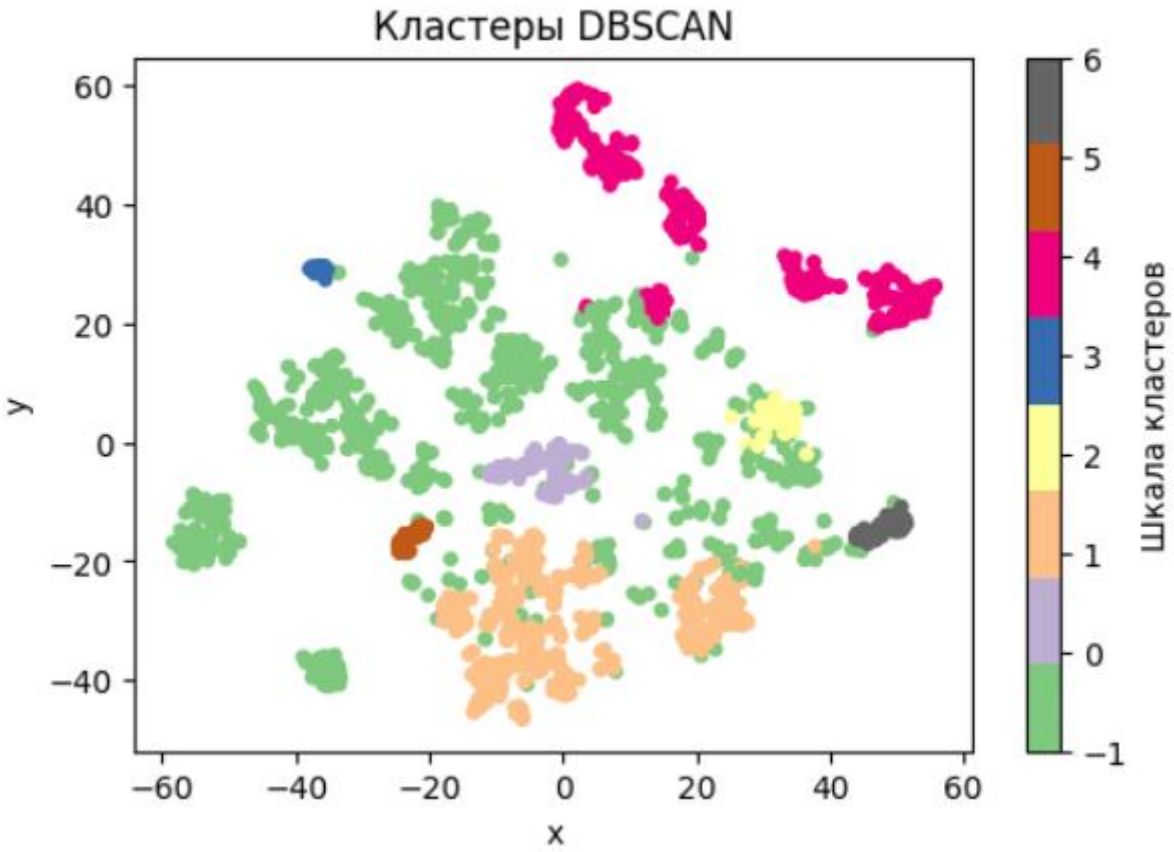

Рисунок 9 – Результат работы программы

Задание 5

Визуализировать кластеризованные данные с помощью t-SNE или UMAP, если необходимо. Если данные трехмерные, то можно использовать трехмерный точечный график. Код представлен на рисунке 10.

```
plt.figure(figsize=(6, 4))
scatter = plt.scatter(tsne_results[:, 0], tsne_results[:, 1], c=orig_labels, cmap='Accent', s=15)
plt.title('Исходная классификация')
plt.xlabel('x')
plt.ylabel('y')
plt.colorbar(scatter, label='Шкала кластеров')
plt.show()
```

Рисунок 10 – Код программы

Результат работы программы представлен на рисунке 11.

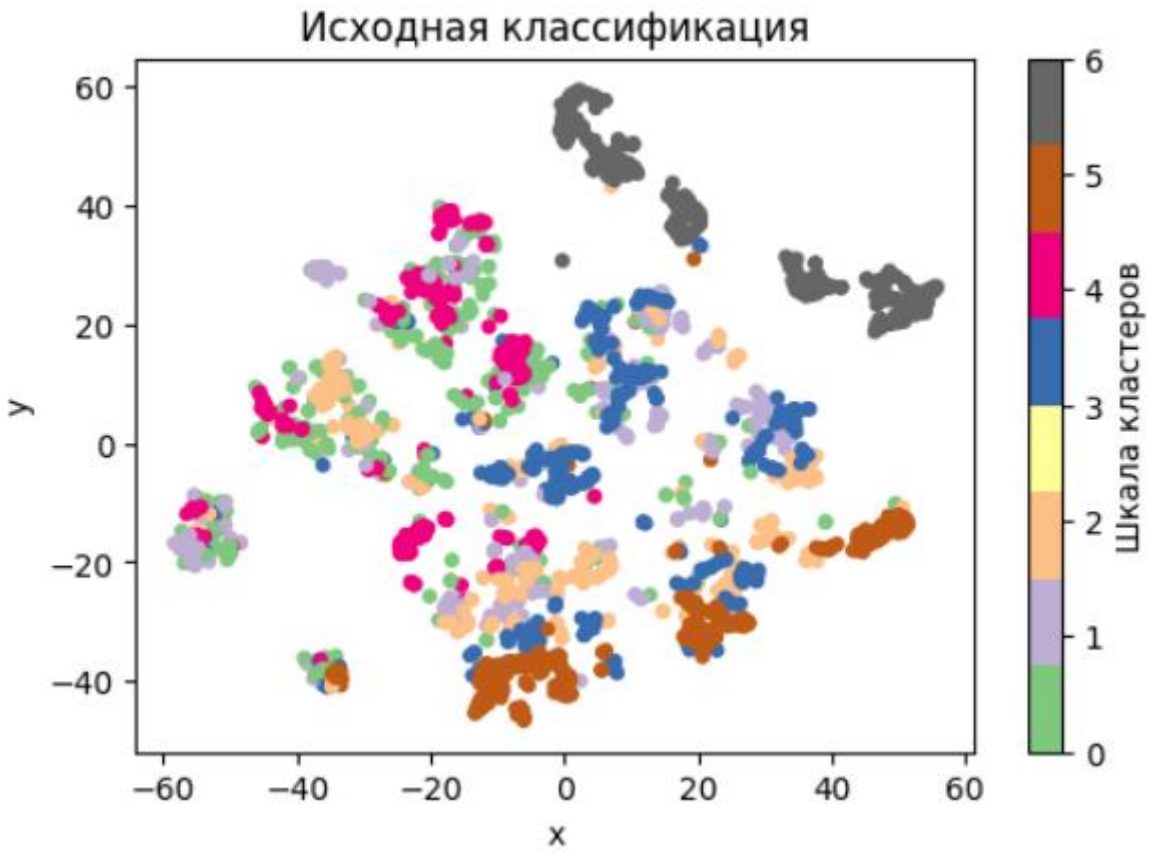

Рисунок 11 – Результат работы программы