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 № 2

по дисциплине «Технологии и инструментарий анализа больших данных»

Выполнил студент группы ИКБО-20-21 Сидоров С.Д.

Проверил ассистент кафедры ПМ ИИТ Тетерин Н.Н.

Москва 2024

Задание 1

Найти и выгрузить многомерные данные (с большим количеством признаков – столбцов) с использованием библиотеки pandas. В отчёте описать найденные данные.

Были выбраны данные о домах в Калифорнии, включая медианную стоимость, среднее количество комнат, количество жильцов и другие параметры.

Задание 2

Вывести информацию о данных при помощи методов .info(), .head(). Проверить данные на наличие пустых значений. В случае их наличия удалить данные строки или интерполировать пропущенные значения. При необходимости дополнительно предобработать данные для дальнейшей работы с ними.

Код к заданию 2 представлен в листинге 1

Листинг 1 – Код для получения данных о датасете

```
california_data = fetch_california_housing(as_frame=True)
data = california_data.data
target = california_data.target
data['MedHouseVal'] = target
print("Информация о данных:")
print(data.info())
print("\nПервые строки данных:")
print(data.head())
if data.isnull().sum().sum() > 0:
      print("Обнаружены пропущенные значения. Выполняется их удаление...")
      data = data.dropna()
else:
      print("Пропущенные значения не обнаружены.")
```

Результат выполнения представлен на рисунке 1.

```
Информация о данных:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   MedInc       20640 non-null  float64
 1   HouseAge     20640 non-null  float64
 2   AveRooms     20640 non-null  float64
 3   AveBedrms    20640 non-null  float64
 4   Population   20640 non-null  float64
 5   AveOccup     20640 non-null  float64
 6   Latitude     20640 non-null  float64
 7   Longitude    20640 non-null  float64
 8   MedHouseVal  20640 non-null  float64
dtypes: float64(9)
memory usage: 1.4 MB
None

Первые строки данных:
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85

   Longitude  MedHouseVal
0    -122.23        4.526
1    -122.22        3.585
2    -122.24        3.521
3    -122.25        3.413
4    -122.25        3.422
Пропущенные значения не обнаружены.
```

Рисунок 1 – Данные о наборе данных

Задание 3

Построить столбчатую диаграмму (.bar) с использованием модуля graph_objs из библиотеки Plotly с определенными параметрами.

Код к заданию 3 представлен на листинге 2.

Листинг 2 – Код для создания столбчатой диаграммы

```
numerical_columns = data.select_dtypes(include=[float,
int]).columns.difference(['Population'])

bar_data = data[numerical_columns].copy()


mean_values = bar_data.mean()


plt.figure(figsize=(12, 8))

colors = plt.cm.get_cmap('tab20', len(mean_values))

plt.bar(mean_values.index, mean_values.values,
color=colors(range(len(mean_values))), edgecolor='black')

plt.title('Средние значения для различных характеристик домов (без
Population)', fontsize=20)

plt.xlabel('Характеристика', fontsize=16)

plt.ylabel('Среднее значение', fontsize=16)

plt.xticks(rotation=90, fontsize=12)

plt.grid(axis='y', linestyle='--', linewidth=0.7)

plt.show()
```

Столбчатая диаграмм представлена на рисунке 2.

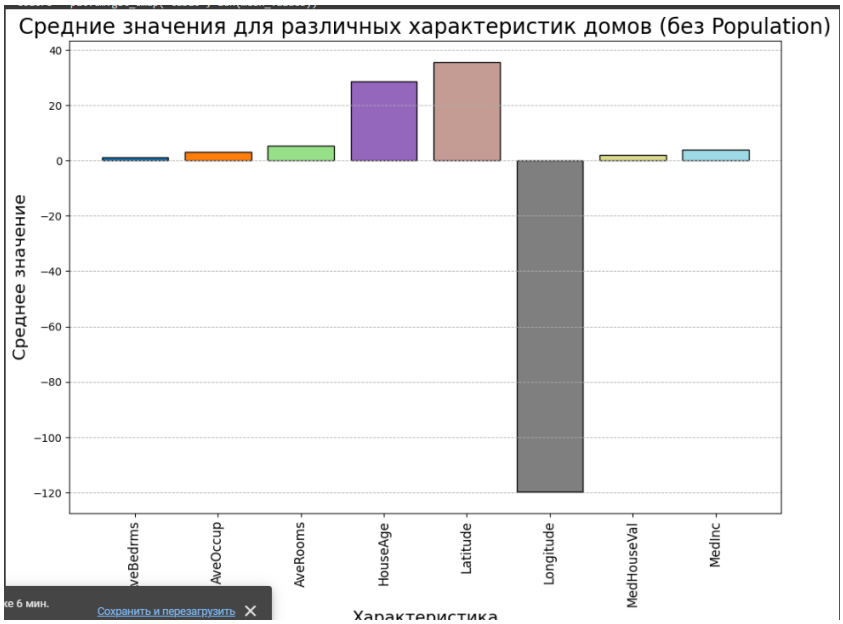

Рисунок 4 – Пример работы

Задание 4

Построить круговую диаграмму (go.Pie), использовав данные и стиль оформления из предыдущего графика. Сделать так, чтобы границы каждой доли были выделены чёрной линией с толщиной, равной 2 и категории круговой диаграммы были читаемы (к примеру, объединить часть объектов).

Листинг 3 – Код для создания круговой диаграммы

```
plt.figure(figsize=(8, 8))

plt.pie(mean_values[:5],                  labels=mean_values.index[:5],
autopct='%1.1f%%', colors=['lightcoral', 'lightskyblue', 'yellowgreen',
'gold', 'violet'],

        wedgeprops={'edgecolor': 'black', 'linewidth': 2})

plt.title('Распределение средних значений (топ 5 характеристик домов)',
fontsize=20)

plt.show()
```

Результат работы представлен на рисунке 3.

Рисунок 3 – Круговая диаграмма

Задание 5

Построить линейные графики, взять один из параметров и определить зависимость между другими несколькими (от 2 до 5) показателями с использованием библиотеки matplotlib.

Код к заданию 5 представлен на в листинге 4.

Листинг 4 – Код для создания линейной диаграммы

```
plt.figure(figsize=(14, 6))

plt.plot(data['AveRooms'][:100], label='Среднее количество комнат', marker='^', linestyle='-.', color='green', markerfacecolor='white',

         markeredgecolor='black', markeredgewidth=2)

plt.plot(data['AveOccup'][:100], label='Среднее количество жильцов', marker='d', linestyle=':', color='purple', markerfacecolor='white',

         markeredgecolor='black', markeredgewidth=2)

plt.plot(data['MedHouseVal'][:100], label='Медианная стоимость дома', marker='o', linestyle='-', color='orange', markerfacecolor='white',

         markeredgecolor='black', markeredgewidth=2)

plt.plot(data['MedInc'][:100], label='Медианный доход', marker='s', linestyle='--', color='blue', markerfacecolor='white',

         markeredgecolor='black', markeredgewidth=2)

plt.grid(True, linewidth=2, color='mistyrose')

plt.title('Линейный график различных показателей дома', fontsize=20)

plt.xlabel('Индекс', fontsize=16)

plt.ylabel('Значение', fontsize=16)

plt.xticks(rotation=315, fontsize=14)

plt.yticks(fontsize=14)

plt.legend()

plt.show()
```

Результат выполнения представлен на рисунке 4.

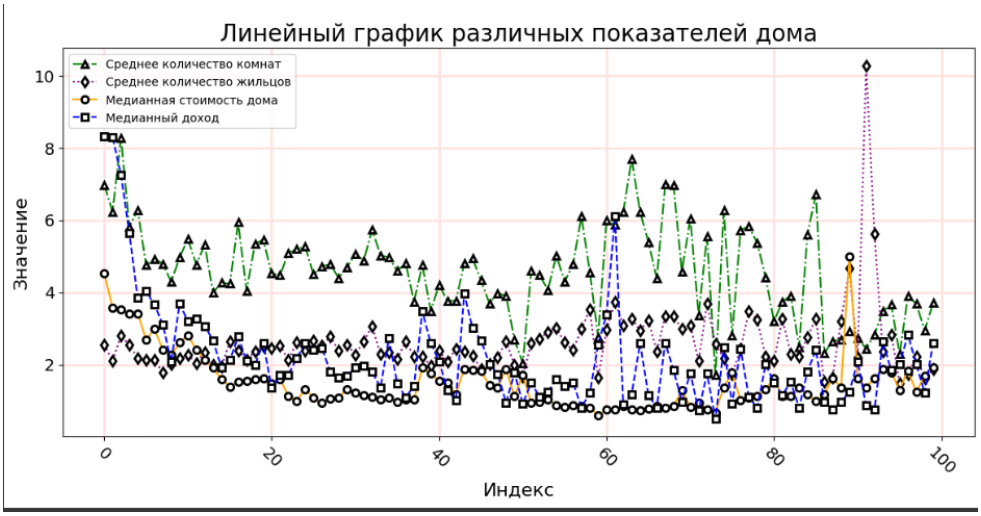

Рисунок 4 – Линейный график

Задание 6

Выполнить визуализацию многомерных данных, используя t-SNE. Необходимо использовать набор данных MNIST или fashion MNIST (можно использовать и другие готовые наборы данных, где можно наблюдать разделение объектов по кластерам). Рассмотреть результаты визуализации для разных значений перплексии.

Код к заданию 6 представлен в листинге 5

Листинг 5 – Визуализация многомерных данных

```
      sample_size = len(data_digits)

      data_subset = data_digits[:sample_size]

      target_subset = target_digits[:sample_size]

      tsne_params = [

          {'perplexity': 5, 'learning_rate': 100, 'n_iter': 1000},

          {'perplexity': 30, 'learning_rate': 200, 'n_iter': 2000},

          {'perplexity': 50, 'learning_rate': 300, 'n_iter': 3000},

          {'perplexity': 10, 'learning_rate': 150, 'n_iter': 1500},

          {'perplexity': 40, 'learning_rate': 250, 'n_iter': 2500}

      ]


      for params in tsne_params:

          start_time_tsne = time.time()

          tsne    =    TSNE(n_components=2,    perplexity=params['perplexity'],
learning_rate=params['learning_rate'],               n_iter=params['n_iter'],
random_state=42)

          tsne_result = tsne.fit_transform(data_subset)

          plt.figure(figsize=(8, 6))

          plt.scatter(tsne_result[:, 0], tsne_result[:, 1], c=target_subset,
cmap='viridis', s=5)

          plt.title(f"t-SNE     Digits:     perplexity={params['perplexity']},
learning_rate={params['learning_rate']}, n_iter={params['n_iter']}")

          plt.colorbar(label='Цифра (класс)')

          plt.show()

          print(f"Время        работы        t-SNE        с        параметрами
perplexity={params['perplexity']},   learning_rate={params['learning_rate']},
n_iter={params['n_iter']}: {time.time() - start_time_tsne:.4f} секунд")
```

Результат работы представлен на рисунках 5 - 9.

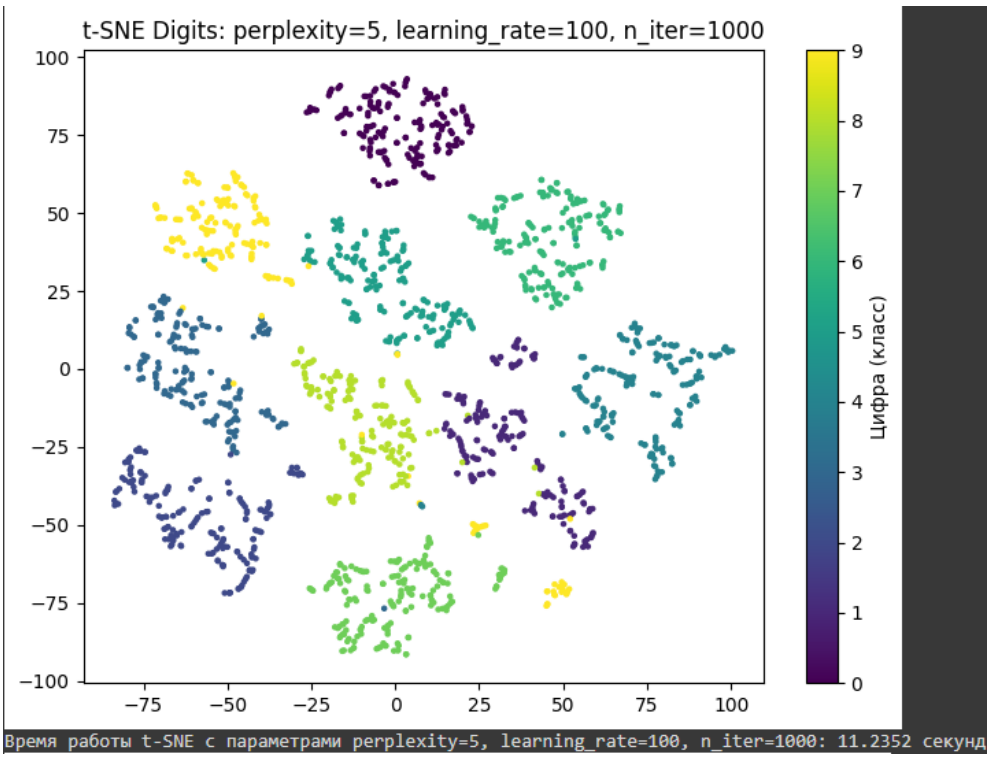

Рисунок 5 –Визуализация t-SNE

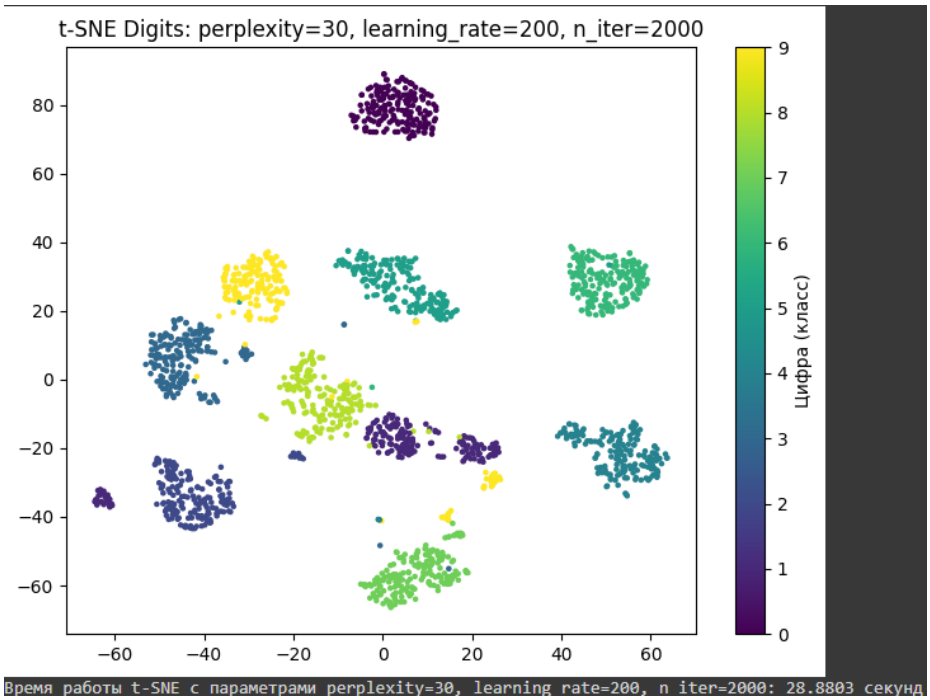

Рисунок 6 -Визуализация t-SNE

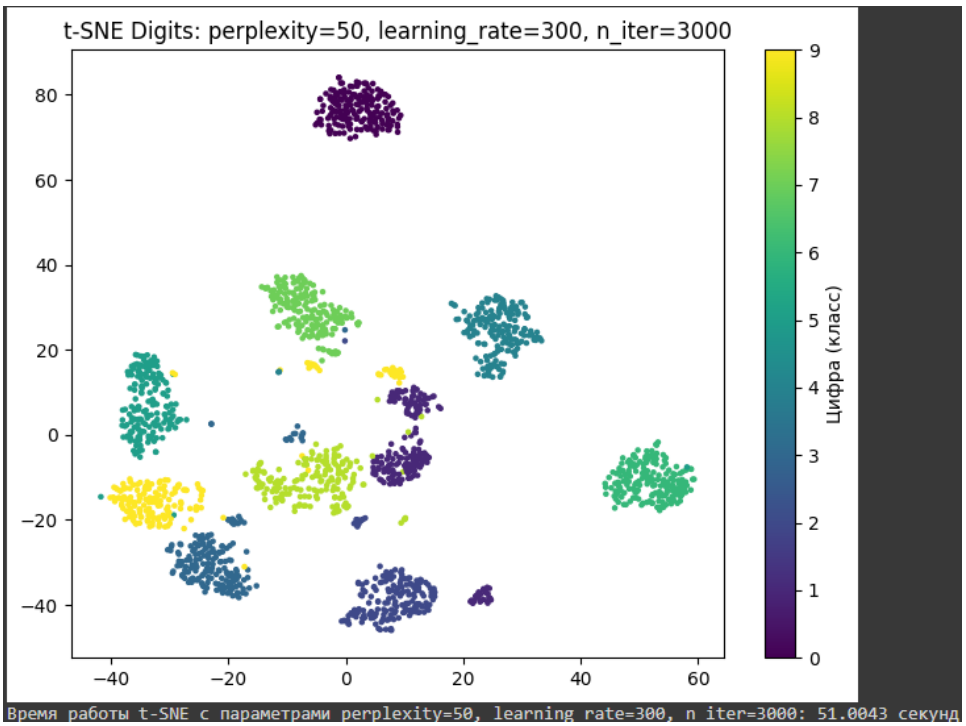

Рисунок 7 - Визуализация t-SNE

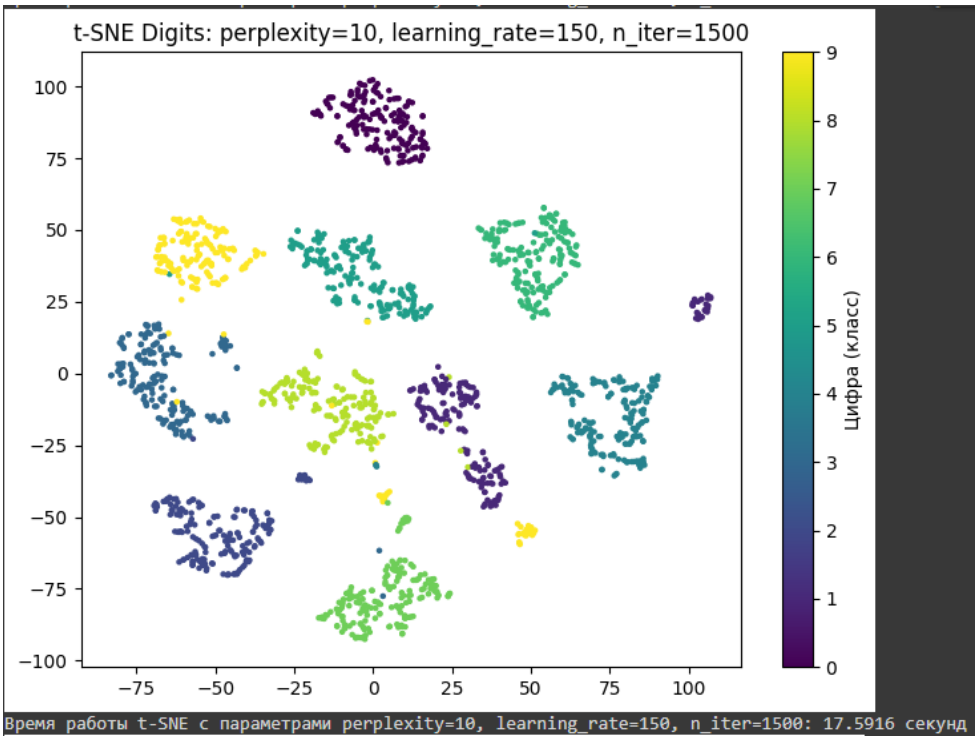

Рисунок 8 - Визуализация t-SNE

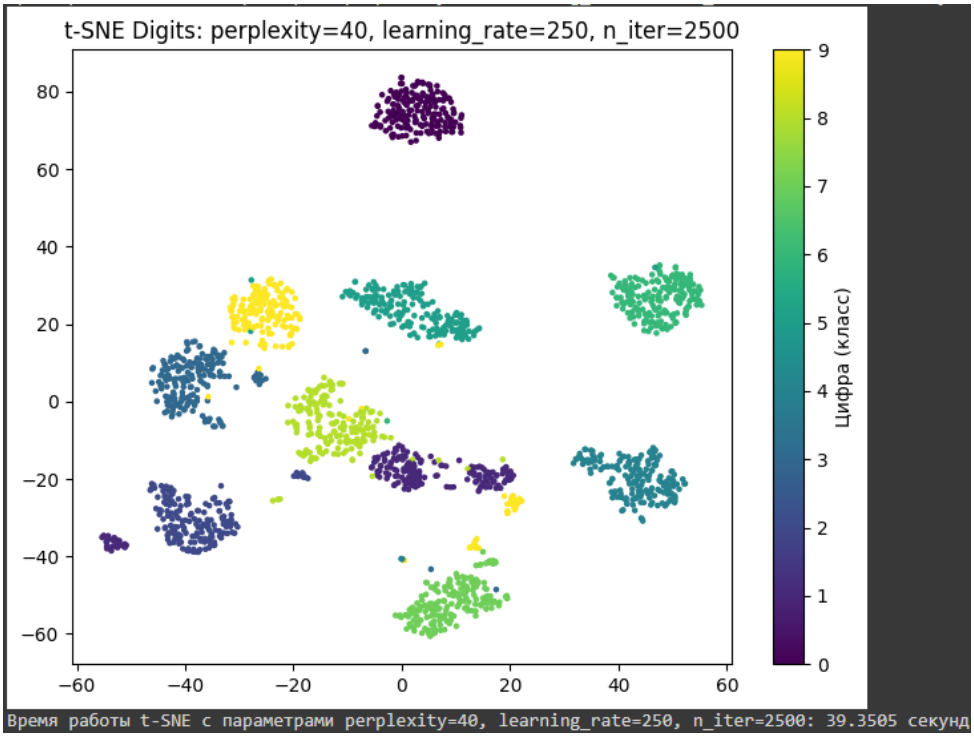

Рисунок 9 - Визуализация t-SNE

Задание 7

Выполнить визуализацию многомерных данных, используя UMAP с различными параметрами n_neighbors и min_dist. Рассчитать время работы алгоритма с помощью библиотеки time и сравнить его с временем работы t-SNE.

Код для выполнения задания представлен в листинге 6.

Листинг 6 – Код для создания визуализации с помощью UMAP

```
umap_params = [
    {'n_neighbors': 5, 'min_dist': 0.1, 'n_components': 2},
    {'n_neighbors': 15, 'min_dist': 0.3, 'n_components': 2},
    {'n_neighbors': 30, 'min_dist': 0.5, 'n_components': 2},
    {'n_neighbors': 10, 'min_dist': 0.2, 'n_components': 2},
    {'n_neighbors': 20, 'min_dist': 0.4, 'n_components': 2}
]


for params in umap_params:
    start_time_umap = time.time()
    reducer = umap.UMAP(n_neighbors=params['n_neighbors'], min_dist=params['min_dist'], n_components=params['n_components'], random_state=42)
    umap_result = reducer.fit_transform(data_subset)
    plt.figure(figsize=(8, 6))
    plt.scatter(umap_result[:, 0], umap_result[:, 1], c=target_subset, cmap='viridis', s=5)
    plt.title(f"UMAP Digits: n_neighbors={params['n_neighbors']}, min_dist={params['min_dist']}")
    plt.colorbar(label='Цифра (класс)')
    plt.show()
    print(f"Время работы UMAP с параметрами n_neighbors={params['n_neighbors']}, min_dist={params['min_dist']}: {time.time() - start_time_umap:.4f} секунд")
```

Результат выполнения представлен на рисунках 10 – 14.

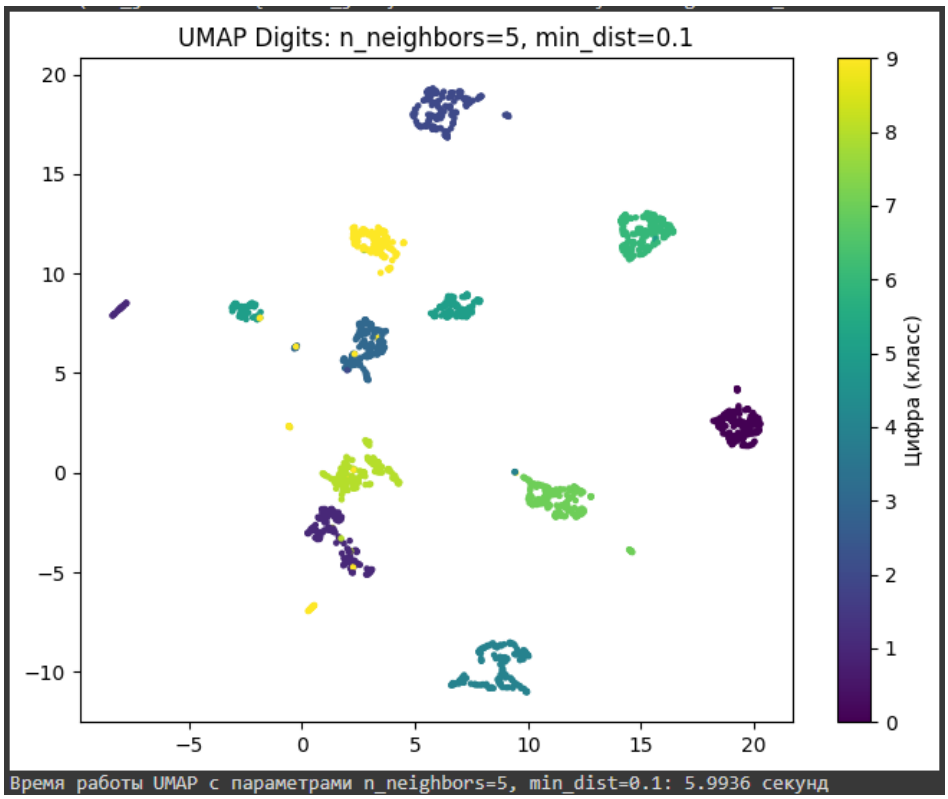

Рисунок 10 – Визуализация UMAP

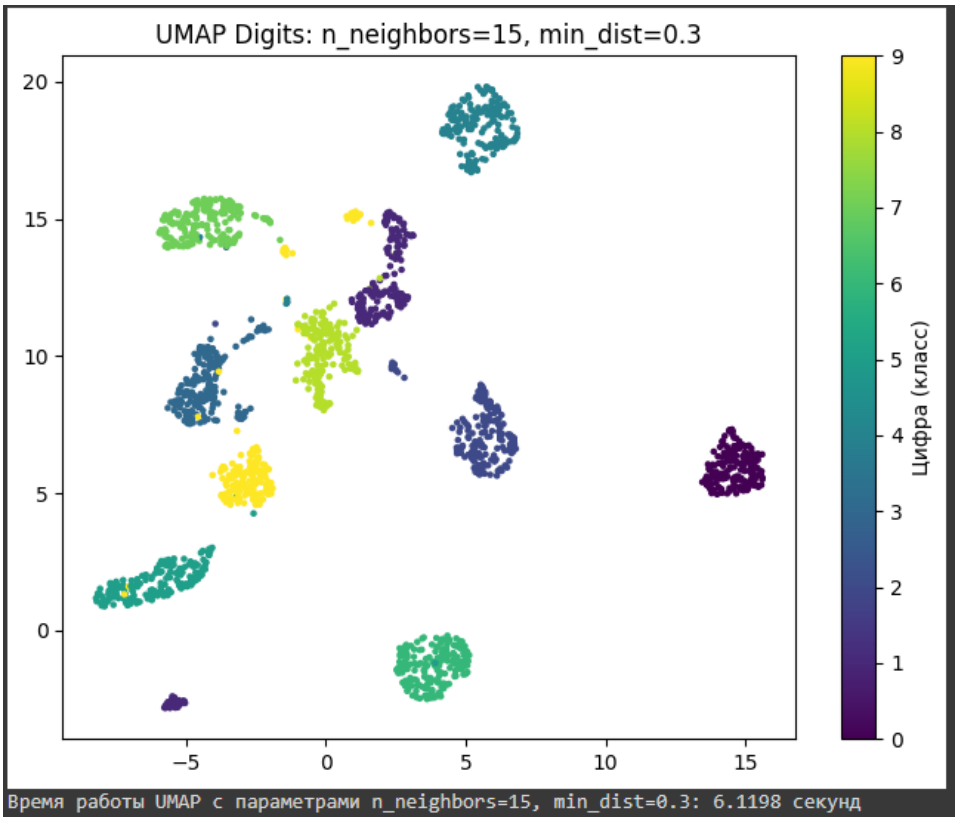

Рисунок 11 – Визуализация UMAP

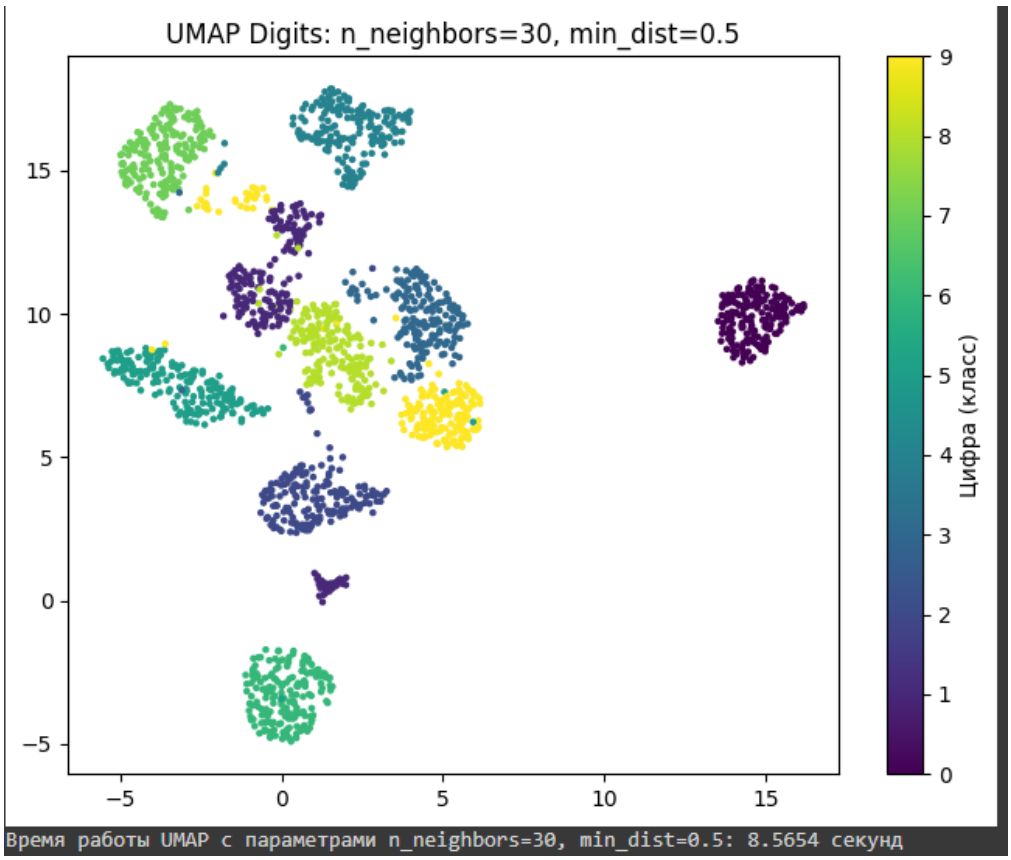

Рисунок 12 – Визуализация UMAP

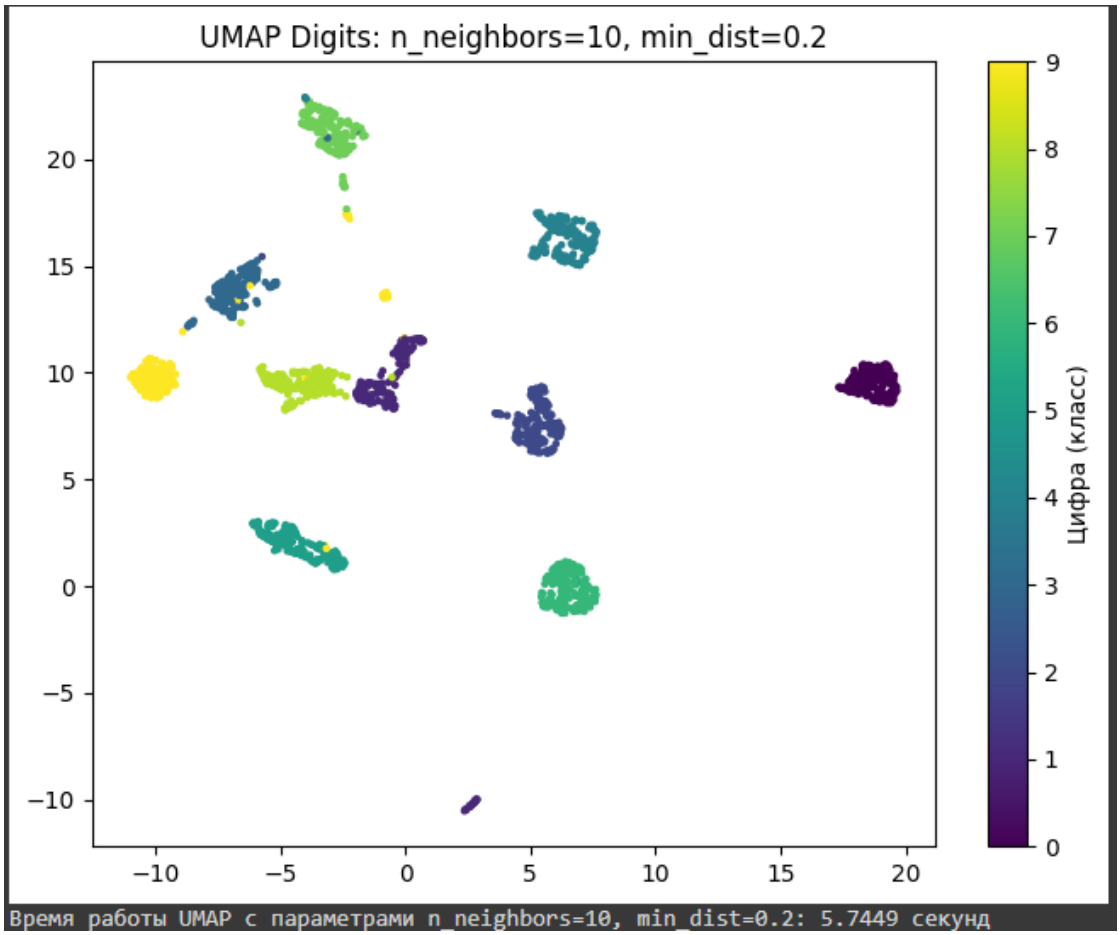

Рисунок 13 – Визуализация UMAP

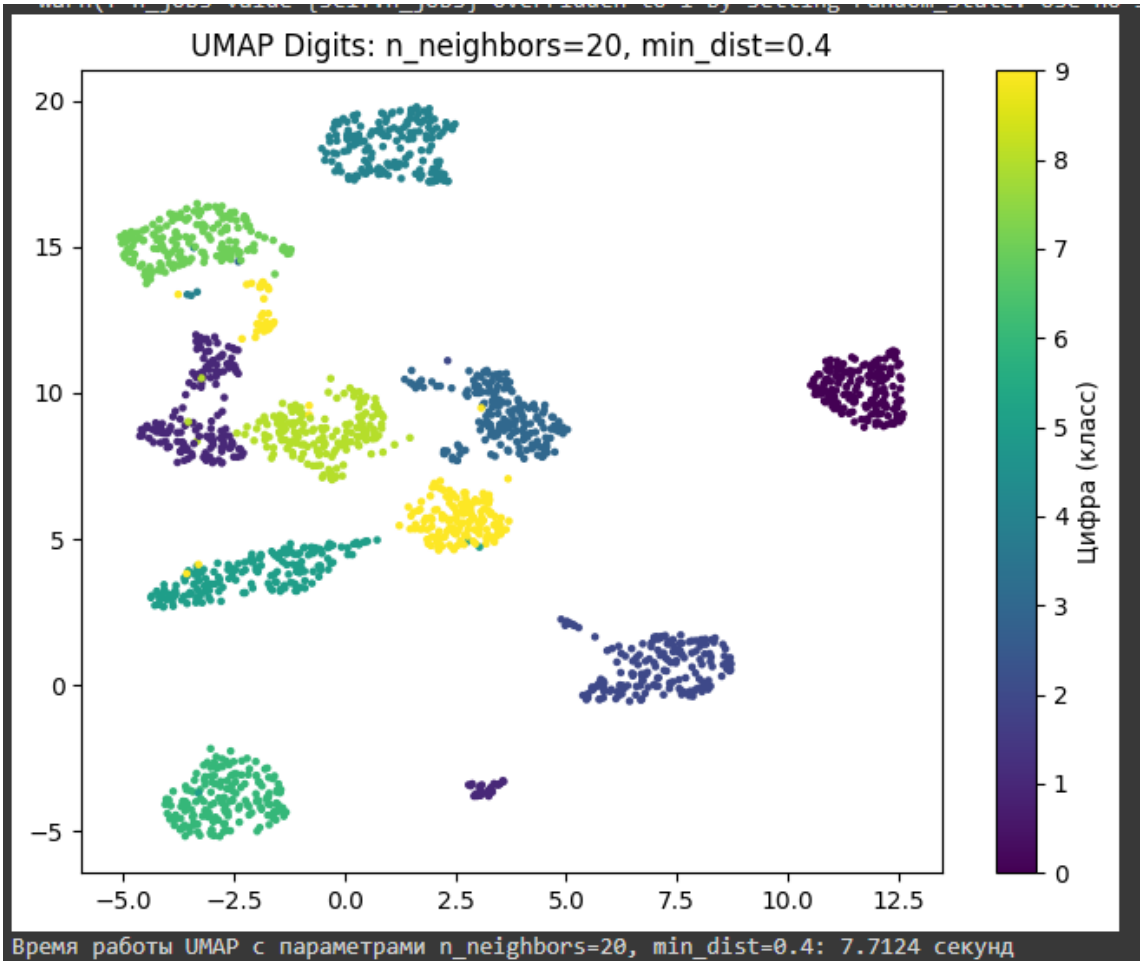

Рисунок 14 – Визуализация UMAP

С точки зрения времени, визуализация UMAP на используемых данных при достаточно схожих итоговых кластерах происходит в не сколько раз быстрее.