|  |  |  |
| --- | --- | --- |
|  | |  |
| МИНОБРНАУКИ РОССИИ | |  |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА - Российский технологический университет"**  **РТУ МИРЭА** | |  |
|  | Институт информационных технологий (ИТ) | |
|  | Кафедра прикладной математики | |

|  |  |
| --- | --- |
| **Отчет по практической работе №2.** | |
| **по дисциплине** | |
| **«Технологии и инструментарий анализа больших данных»** | |
|  | |
| Выполнил студент группы ИКБО- 22-20 | Гологузов К.А. |
| Принял | Парамонов А.А. |

Москва 2023

Решение

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

Для текущей работы был выбран датасет, связанный с красными винами, содержащий в себе 12 признаков-столбцов, например содержание остаточного сахара в вине, показатель кислотности pH, содержание диоксида серы и т.д.

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.

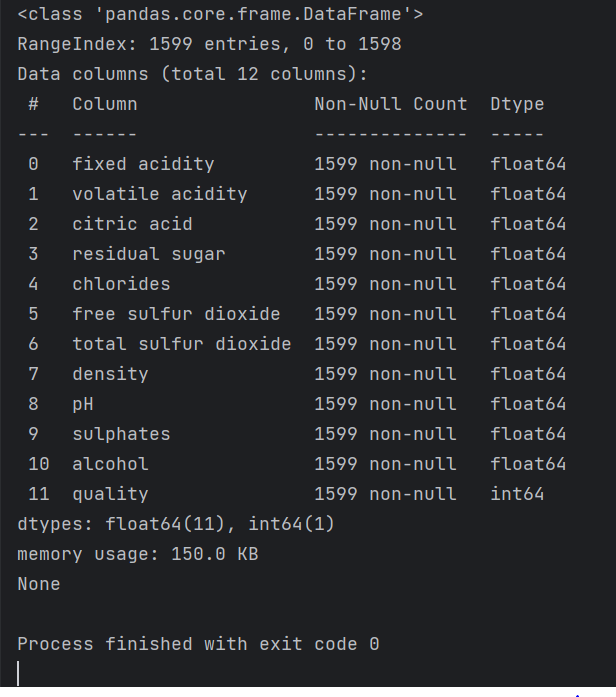


Рис. 2.1 - Результат .info()

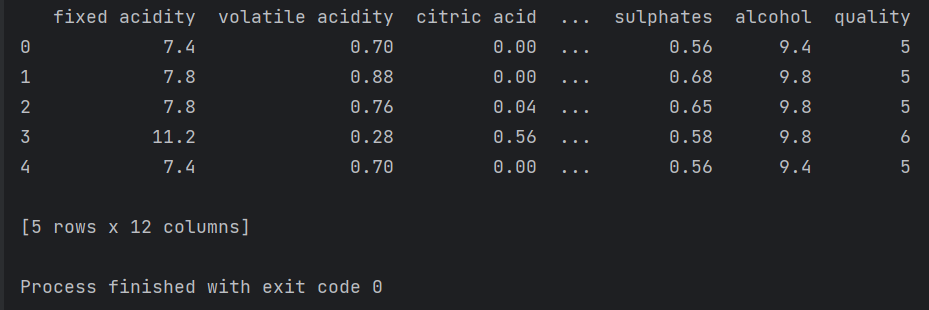


Рис. 2.2 - Результат .head()

Пустые значения отсутствуют. Обрабатывать данные для дальнейшей работы не нужно.

3. Построить столбчатую диаграмму (.bar) с использованием модуля graph\_objs из библиотеки Plotly со следующими параметрами:

3.1. По оси Х указать дату или название, по оси У указать количественный показатель.

3.2. Сделать так, чтобы столбец принимал цвет в зависимости от значения показателя (marker=dict(color=признак, coloraxis="coloraxis")).

3.3. Сделать так, чтобы границы каждого столбца были выделены чёрной линией с толщиной равной 2.

3.4. Отобразить заголовок диаграммы, разместив его по центру сверху, с 20 размером текста.

3.5. Добавить подписи для осей X и Y с размером текста, равным 16. Для оси абсцисс развернуть метки так, чтобы они читались под углом, равным 315.

3.6. Размер текста меток осей сделать равным 14.

3.7. Расположить график во всю ширину рабочей области и присвоить высоту, равную 700 пикселей.

3.8. Добавить сетку на график, сделать её цвет 'ivory' и толщину равную 2. (Можно сделать это при настройке осей с помощью gridwidth=2, gridcolor='ivory')

3.9. Убрать лишние отступы по краям.

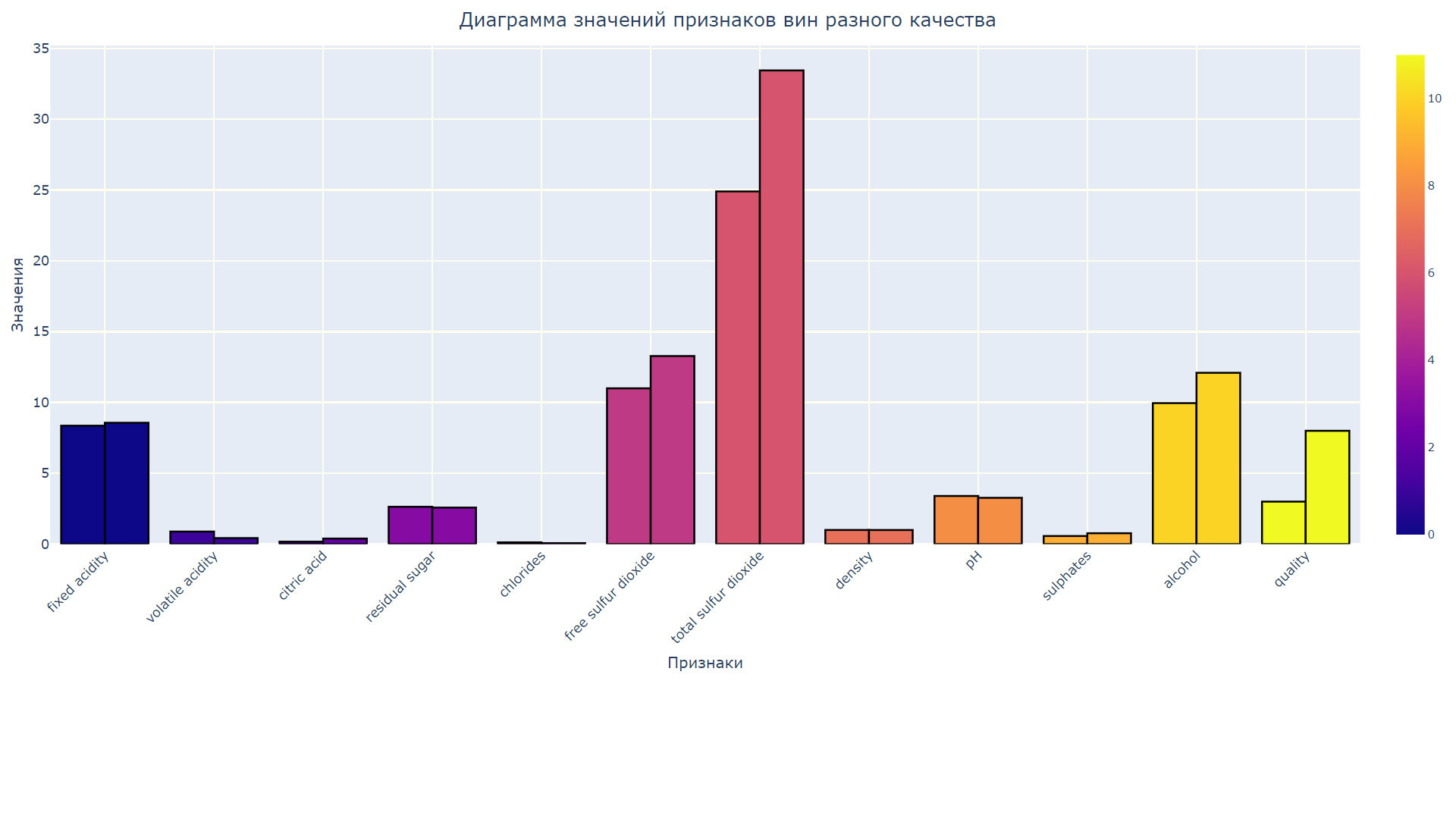


Рис. 3.1 – Результат

Приложение 1 – Код программы

import pandas as pd  
import plotly.graph\_objs as go  
  
  
df = pd.read\_csv("winequality-red.csv")  
columns = df.columns.values.tolist()  
first\_bar = df.loc[df["quality"] == 3]  
second\_bar = df.loc[df["quality"] == 8]  
  
fig = go.Figure(data=[go.Bar(x=columns, y=first\_bar.mean(), marker=dict(color=list(range(len(columns))), coloraxis="coloraxis", line=dict(color="black",width=2)), showlegend=False),  
 go.Bar(x=columns, y=second\_bar.mean(), marker=dict(color=list(range(len(columns))), coloraxis="coloraxis", line=dict(color="black",width=2)), showlegend=False)])  
  
fig.update\_layout(  
 title="Диаграмма значений признаков вин разного качества", title\_font\_size=20, title\_x=0.5,  
 xaxis\_title="Признаки", xaxis\_title\_font\_size=16, xaxis\_tickangle=315, xaxis\_tickfont\_size=14,  
 yaxis\_title="Значения", yaxis\_title\_font\_size=16, yaxis\_tickfont\_size=14,  
 height=700,  
 margin=dict(l=0, r=0, t=40)

Продолжение – Листинг 1

)  
fig.update\_xaxes(gridwidth=2, gridcolor="ivory")  
fig.update\_yaxes(gridwidth=2, gridcolor="ivory")  
fig.show()

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)

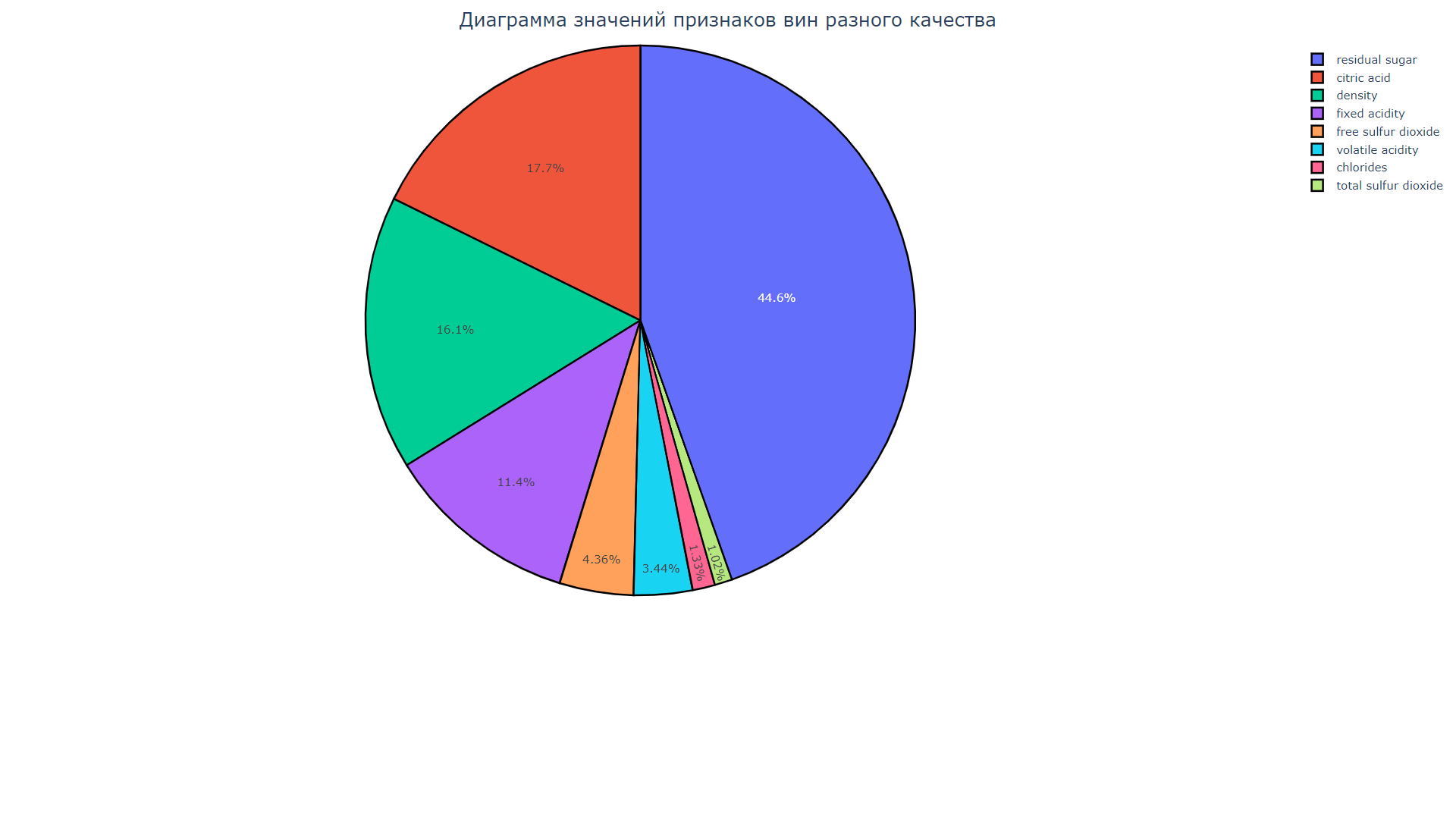


Рис. 4.1 – Результат

Приложение 2 – Код программы

import pandas as pd  
import plotly.graph\_objs as go  
  
  
df = pd.read\_csv("winequality-red.csv")  
columns = df.columns.values.tolist()  
first\_bar = df.loc[df["quality"] == 8]  
first\_bar = first\_bar.drop(columns=["quality", "chlorides", "citric acid", "volatile acidity"], inplace=False)  
values = first\_bar.mean()  
  
fig = go.Figure(data=[go.Pie(labels=columns, values=values)])  
fig.update\_traces(  
 marker=dict(colors=list(range(len(columns))), line=dict(color='#000000', width=2))  
)  
fig.update\_layout(  
 title="Диаграмма значений признаков вин разного качества", title\_font\_size=20, title\_x=0.5,

Продолжение – Листинг 2

height=700,  
 margin=dict(l=0, r=0, t=40)  
)  
print(first\_bar.head())  
fig.show()

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

5.1. Сделать график с линиями и маркерами, цвет линии 'crimson', цвет точек 'white', цвет границ точек 'black', толщина границ точек равна 2.

5.2. Добавить сетку на график, сделать её цвет 'mistyrose' и толщину равную 2. (Можно сделать это при настройке осей с помощью linewidth=2, color='mistyrose').

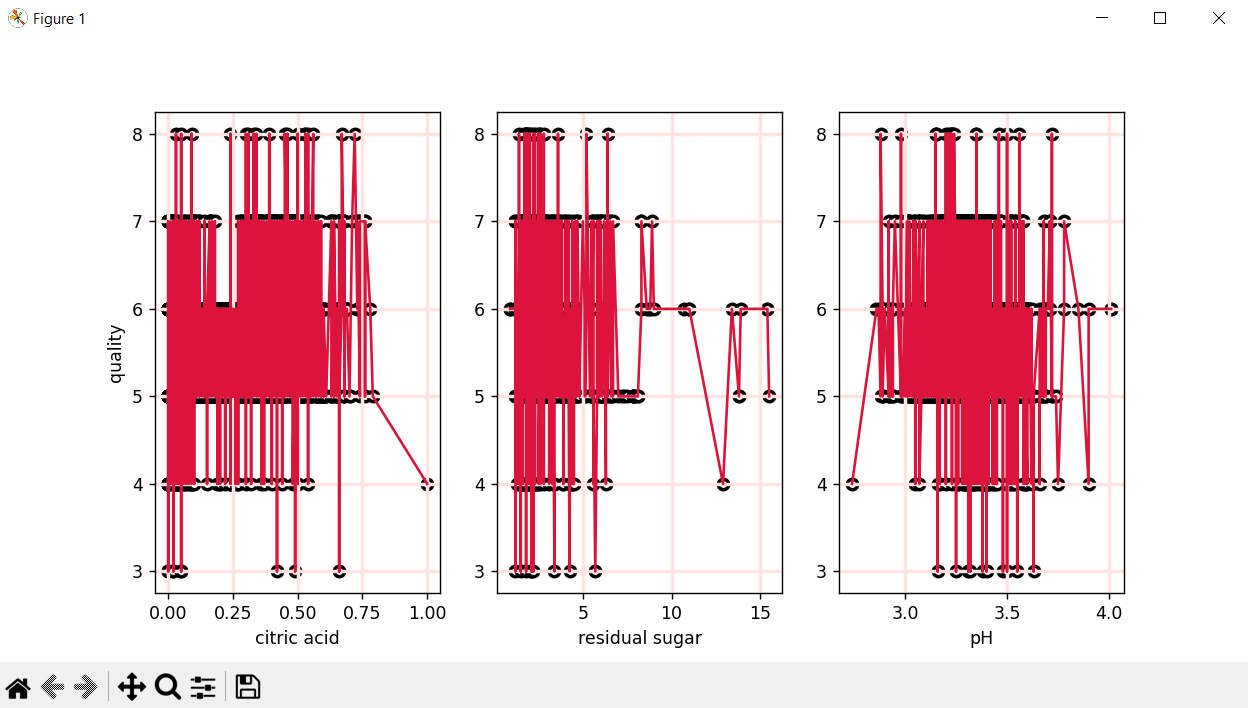


Рис. 5.1 – Результат

Приложение 3 – Код программы

import matplotlib.pyplot as plt  
import pandas as pd  
  
df = pd.read\_csv("winequality-red.csv")  
columns = df.columns.values.tolist()

Продолжение – Листинг 3

print(columns)  
first\_bar = df.loc[df["quality"] == 3]  
second\_bar = df.loc[df["quality"] == 8]  
  
df = df.sort\_values("citric acid")  
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize = (10,5))  
ax1.plot(df["citric acid"], df["quality"], c="crimson")  
ax1.scatter(df["citric acid"].sort\_values(), df["quality"], c="white", edgecolors='black', linewidth=2)  
ax1.grid(linewidth=2, color='mistyrose')  
ax1.set\_xlabel("citric acid")  
ax1.set\_ylabel("quality")  
  
df = df.sort\_values("residual sugar")  
ax2.plot(df["residual sugar"], df["quality"], c="crimson")  
ax2.scatter(df["residual sugar"].sort\_values(), df["quality"], c="white", edgecolors='black', linewidth=2)  
ax2.grid(linewidth=2, color='mistyrose')  
ax2.set\_xlabel("residual sugar")  
  
df = df.sort\_values("pH")  
ax3.plot(df["pH"], df["quality"], c="crimson")  
ax3.scatter(df["pH"].sort\_values(), df["quality"], c="white", edgecolors='black', linewidth=2)  
ax3.grid(linewidth=2, color='mistyrose')  
ax3.set\_xlabel("pH")  
  
  
plt.show()

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.

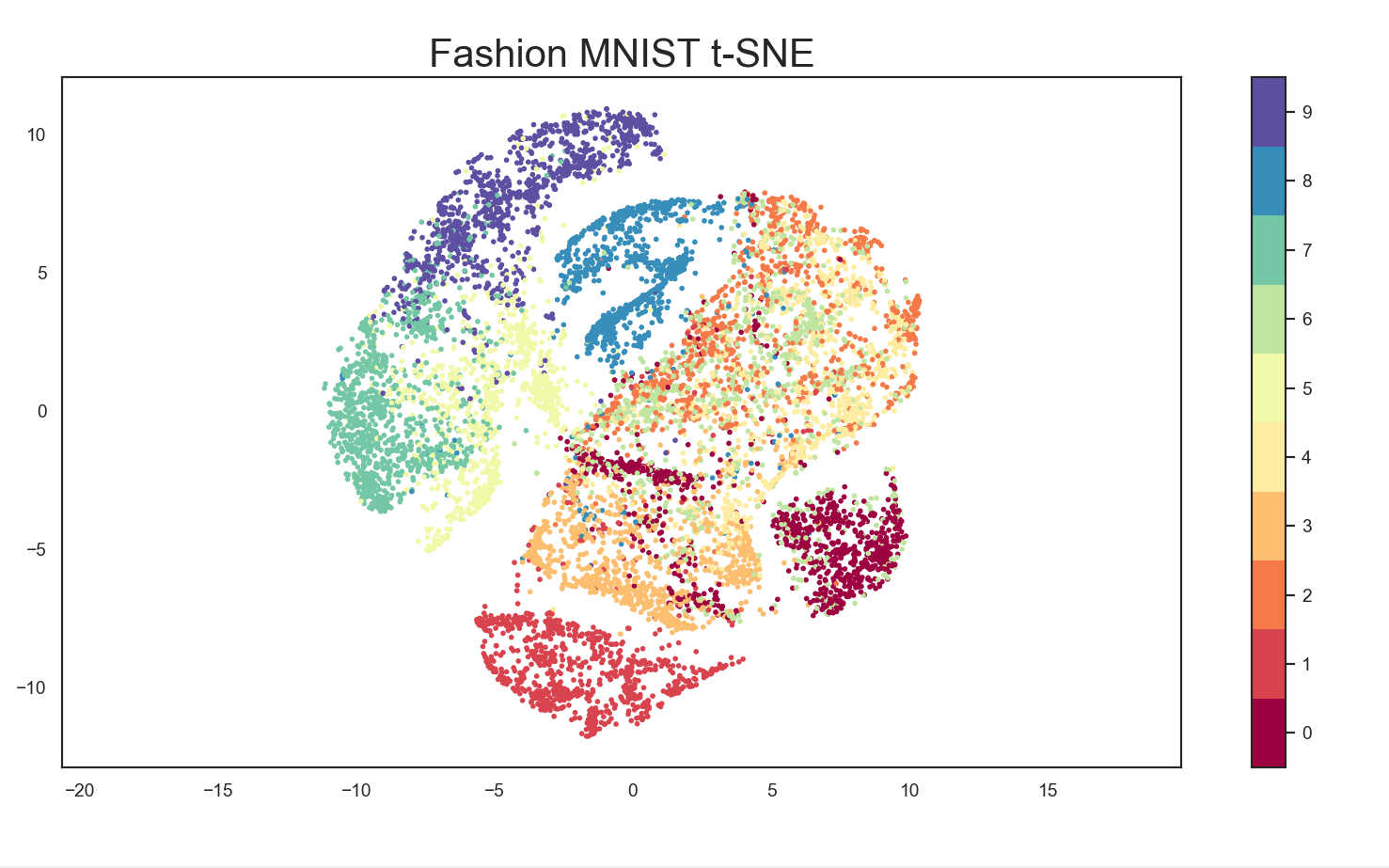


Рис. 6.1 – Результат работы программы, perplexity = 40

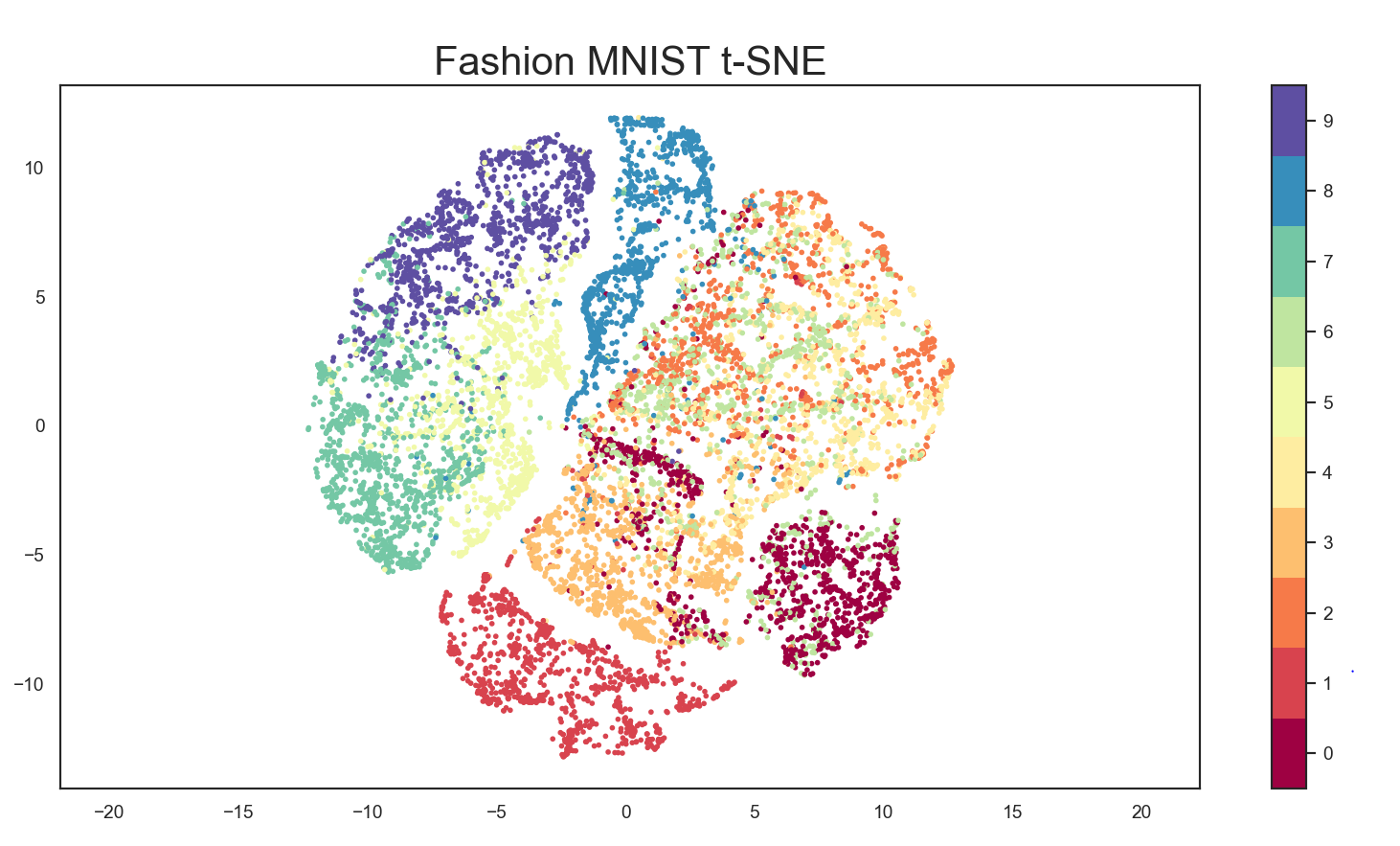


Рис. 6.2 – Результат работы программы, perplexity = 10

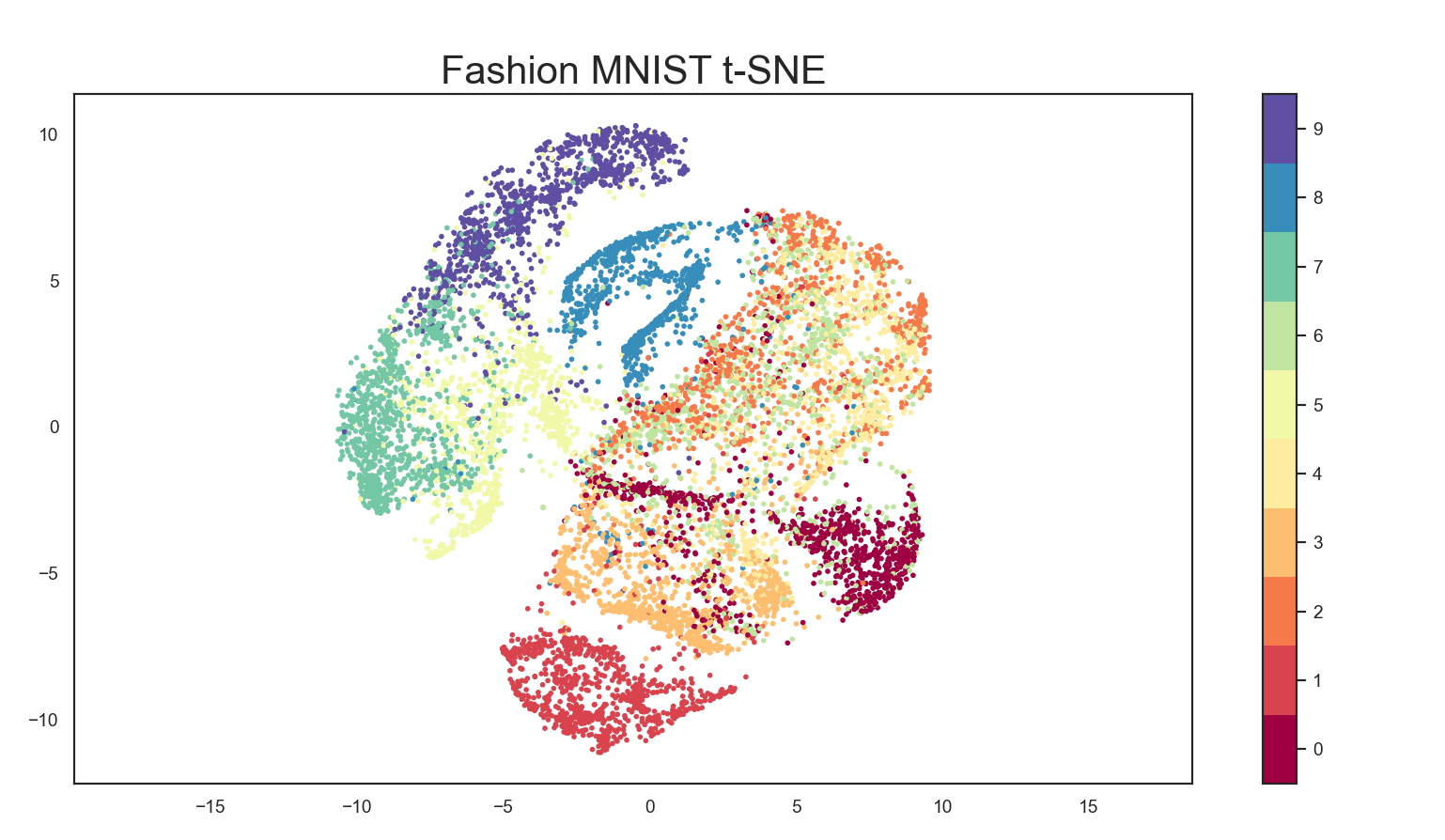


Рис. 6.3 – Результат работы программы, perplexity = 70

Среднее время работы алгоритма составляет примерно 14 секунд.

Приложение 4 – Код программы

import pandas as pd  
import numpy as np  
from time import time  
  
  
from matplotlib import offsetbox  
import matplotlib.pyplot as plt  
import matplotlib.patheffects as PathEffects  
import seaborn as sns  
import plotly.graph\_objects as go  
  
sns.set(style="white", context="notebook",rc={'figure.figsize':(14,10)})  
  
from sklearn.preprocessing import StandardScaler  
from sklearn.manifold import TSNE  
import warnings  
warnings.filterwarnings("ignore")  
  
train = pd.read\_csv("fashion-mnist\_train.csv")  
  
y = train.loc[:, "label"].values  
x = train.loc[:,"pixel1":].values  
  
print(x.shape)  
print(y)  
  
def plot\_digits(data):  
 fig, axes = plt.subplots(4, 10, figsize=(10, 4),  
 subplot\_kw={'xticks':[], 'yticks':[]},  
 gridspec\_kw=dict(hspace=0.1, wspace=0.1))

Продолжение – листинг 4

for i, ax in enumerate(axes.flat):  
 ax.imshow(data[i].reshape(28, 28),  
 cmap='binary', interpolation='nearest',  
 clim=(0, 16))  
 plt.show()  
  
  
standardized\_data = StandardScaler().fit\_transform(x)  
print(standardized\_data.shape)  
  
x\_subset = x[0:10000]  
y\_subset = y[0:10000]  
  
print(np.unique(y\_subset))  
tsne = TSNE(random\_state = 42, n\_components=2,verbose=0, perplexity=70, n\_iter=300).fit\_transform(x\_subset)  
  
plt.scatter(tsne[:, 0], tsne[:, 1], s= 5, c=y\_subset, cmap='Spectral')  
plt.gca().set\_aspect('equal', 'datalim')  
plt.colorbar(boundaries=np.arange(11)-0.5).set\_ticks(np.arange(10))  
plt.title('Fashion MNIST t-SNE', fontsize=24)  
plt.show()

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.

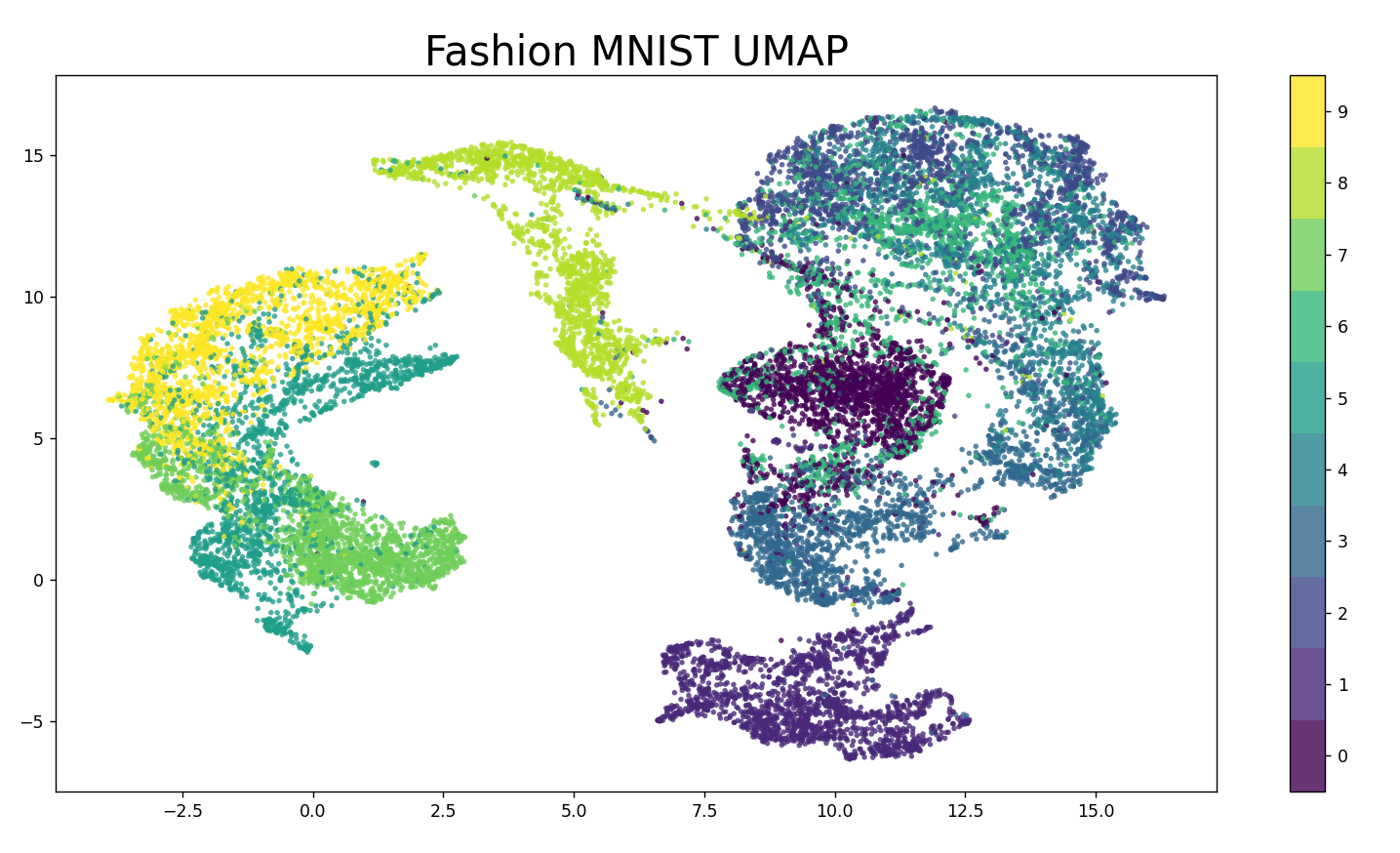


Рис. 7.1 – n\_neighbors = 5, min\_dist = 0.3

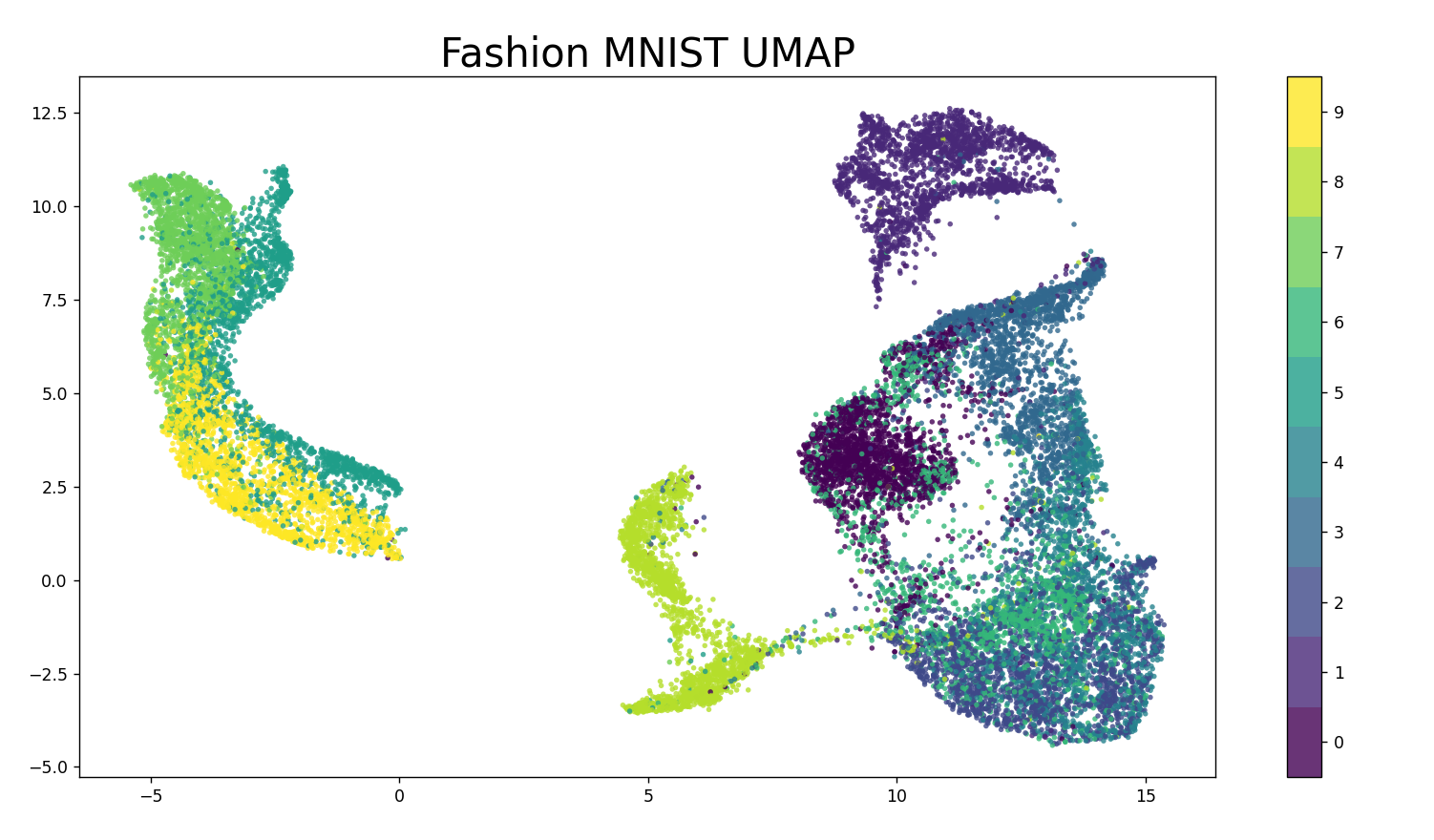


Рис. 7.2 – n\_neighbors = 50, min\_dist = 0.3

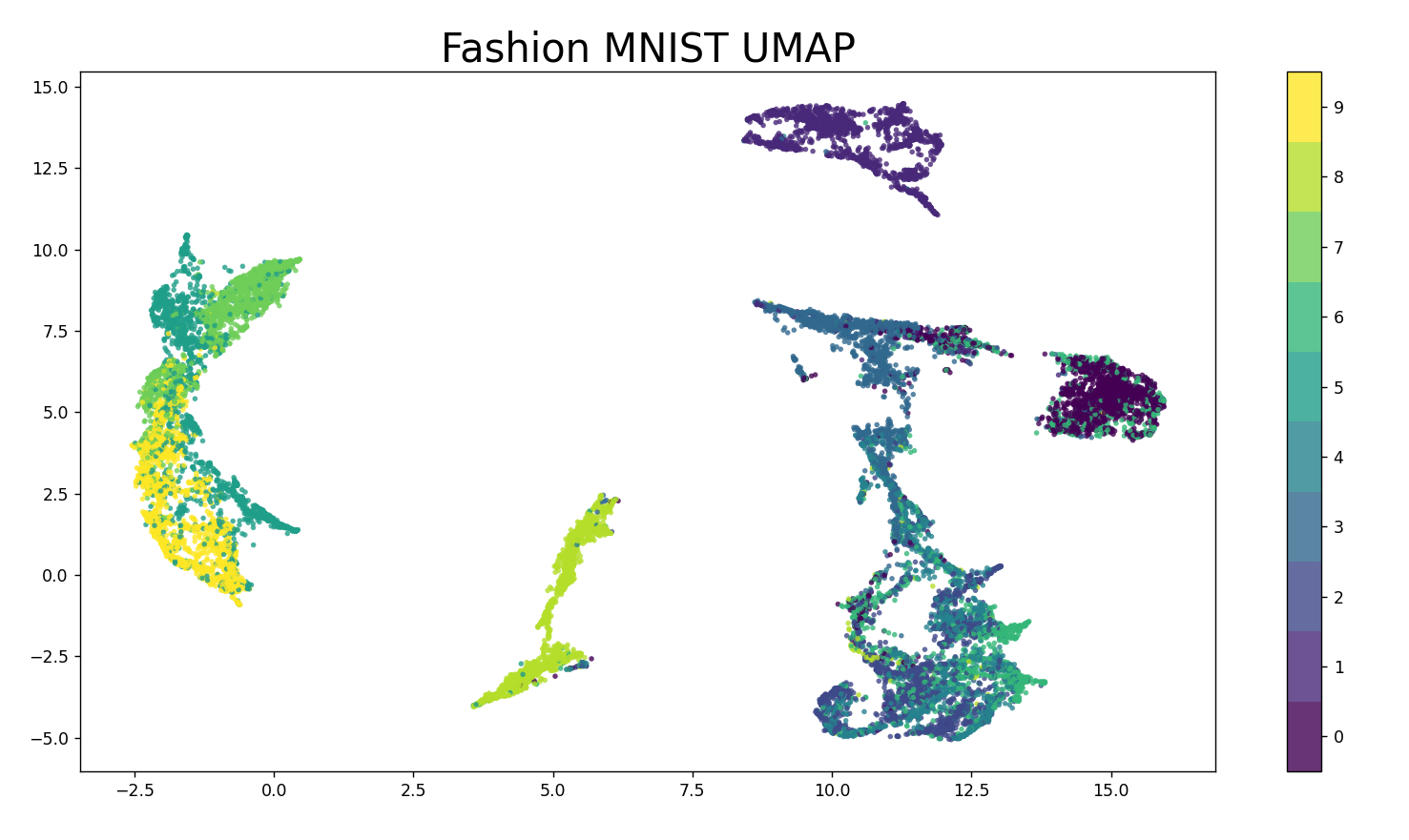


Рис. 7.3 – n\_neighbors = 10, min\_dist = 0.03

Среднее время работы алгоритма составило 21 секунду, следовательно, можно сделать вывод, что для текущих данных алгоритм t-sne подходит лучше.

Приложение 5 – Код программы

import pandas as pd  
import umap  
import numpy as np  
from time import time  
import matplotlib.pyplot as plt  
  
  
start\_time = time()  
digits = pd.read\_csv("fashion-mnist\_train.csv")  
  
embedding = umap.UMAP(n\_neighbors=10,  
 min\_dist=0.01,  
 metric='correlation').fit\_transform(digits.iloc[:20000, 1:])  
  
  
plt.figure(figsize=(12,12))  
plt.scatter(embedding[:20000, 0], embedding[:20000, 1],  
 c=digits.iloc[:20000, 0],  
 edgecolor='none',  
 alpha=0.80,  
 s=10)  
plt.colorbar(boundaries=np.arange(11)-0.5).set\_ticks(np.arange(10))  
plt.title('Fashion MNIST UMAP', fontsize=24)  
# plt.axis('off')  
end\_time = time()  
exec\_time = end\_time - start\_time  
print("Time: ", exec\_time)  
plt.show()