

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное образовательное учреждение высшего  
образования  
«Омский государственный технический университет»

Факультет информационных технологий и компьютерных систем  
Кафедра «Прикладная математика и фундаментальная информатика»

**Домашнее задание**


по дисциплине Практикум по программированию

Студента Шелепова Дениса Дмитриевича  
фамилия, имя, отчество полностью

Курс 2 Группа ФИТ-231

Направление 02.03.02. Фундаментальная информатика и  
информационные технологии  
код, наименование

Руководитель старший преподаватель  
должность, ученая степень, звание  
Саматов А. П.  
фамилия, инициалы, дата, подпись

Выполнил 01.12.2024   
дата, <sup>подпись</sup> студента(ки)

Итоговый рейтинг	
------------------	--

Омск 2024

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	3
1 Поиск, загрузка и подготовка данных .....	4
2 Разведывательный анализ данных .....	8
3 Предварительная обработка данных.....	13
ЗАКЛЮЧЕНИЕ .....	16
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	17

## ВВЕДЕНИЕ

Анализ данных играет ключевую роль в современном обществе, активно влияя на принятие решений в бизнесе, науке и многих других сферах. Прежде всего он позволяет извлечь ценную информацию из массивов данных, которые в противном случае оставались бы без внимания. Эти данные, будь то числовые показатели продаж, медицинские исследования или результаты социологических опросов, предоставляют необходимую основу для глубинного понимания процессов и тенденций.

Используя статистические методы и современные технологии, организации могут выявлять закономерности, прогнозировать будущее и находить новые возможности для роста. Например, компании могут оптимизировать свою деятельность, анализируя предпочтения клиентов и разрабатывая персонализированные предложения, что существенно увеличивает шансы на успех.

Еще анализ данных помогает в повышении эффективности и снижении рисков. В условиях постоянно меняющейся рыночной среды способность быстро адаптироваться к новым условиям становится конкурентным преимуществом. Таким образом, данные становятся не просто цифрами, а мощным инструментом, открывающим перспективы для инноваций и стратегического развития.

Анализировать данные помогают следующие библиотеки, написанные для работы на языке программирования *Python*: *Pandas*, *Numpy*, *Scipy*, *Seaborn* и *Matplotlib*. Первая из них, *Pandas*, позволяет хранить и обрабатывать датафреймы. *Numpy* и *Scipy* предоставляют широкий спектр математических процедур над различными объектами, например поиск моды, медианы, среднего значения. *Seaborn* и *Matplotlib* содержат набор инструментов, позволяющих строить разнообразные графики и диаграммы.

## 1 Поиск, загрузка и подготовка данных

Для выполнения задания на платформе *Kaggle* был выбран датасет «*World Cities Dataset*», который отражает информацию о всех городах мира. В нем содержатся такие поля, как название города, его географические координаты (широта и долгота), страна, в которой он находится, её сокращенное название, население города и его рейтинг. Чтобы описать датасет, был создан файл *describe.md*, представленный на рисунке 1.

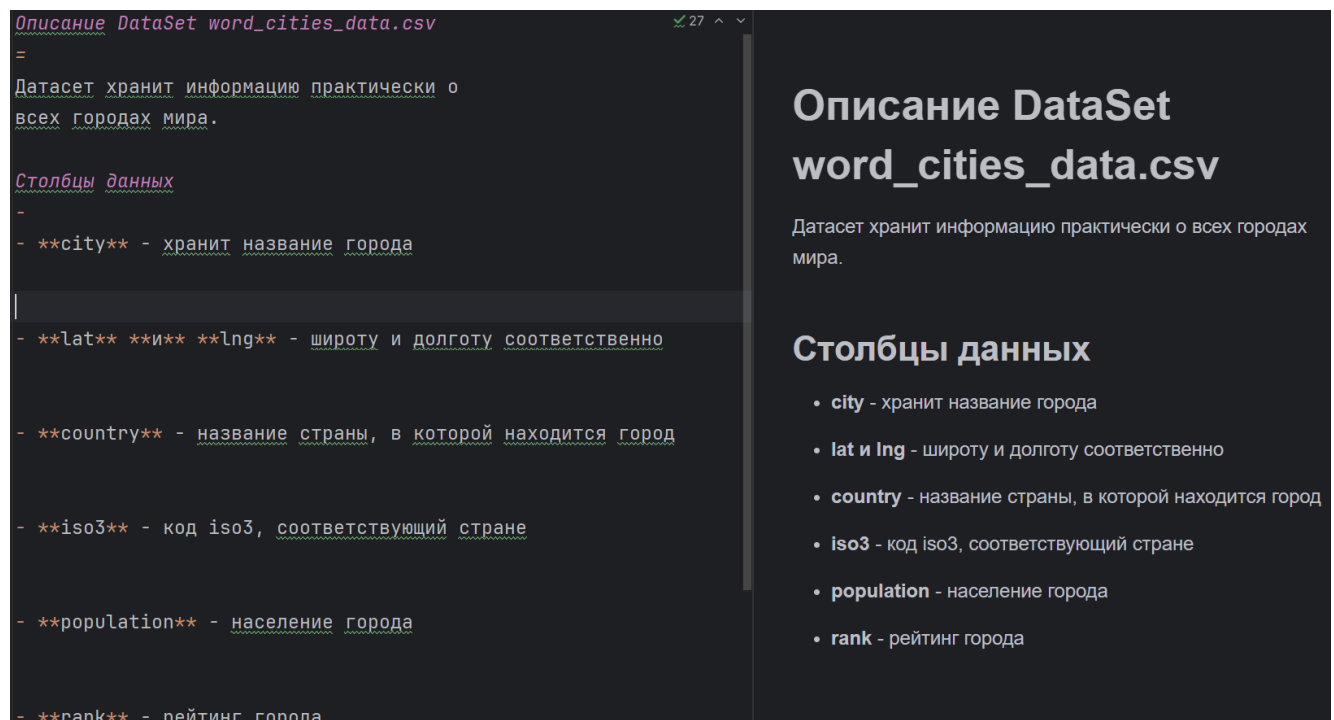


Рисунок 1 – Файл *describe.md*

Датасет представляет собой *csv*-файл, показанный на рисунке 2.

```
id,city,lat,lng,country,iso3,population,rank
0,Tokyo,35.6897,139.6922,Japan,JPN,37732000.0,1.0
1,Jakarta,-6.175,106.8275,Indonesia,IDN,33756000.0,2.0
2,Delhi,28.61,77.23,India,IND,32226000.0,3.0
3,Guangzhou,23.13,113.26,China,CHN,26940000.0,4.0
4,Mumbai,19.0761,72.8775,India,IND,24973000.0,5.0
5,Manila,14.5958,120.9772,Philippines,PHL,24922000.0,6.0
6,Shanghai,31.1667,121.4667,China,CHN,24073000.0,7.0
7,São Paulo,-23.55,-46.6333,Brazil,BRA,23086000.0,8.0
8,Seoul,37.56,126.99,South Korea,KOR,23016000.0,9.0
```

Рисунок 2 – Исходный датасет

Датасет был загружен в программу с помощью функции `read_csv` библиотеки *Pandas* (рисунок 3).

```
df = pd.read_csv(filepath_or_buffer: 'data/world_cities_data.csv', delimiter=',')
```

Рисунок 3 – Загрузка данных датасета в программу

На рисунках 4-5 представлены первые пять и последние пять строчек датасета.

	id	city	lat	lng	country	iso3	population	rank
0	0	Tokyo	35.6897	139.6922	Japan	JPN	37732000.0	1.0
1	1	Jakarta	-6.1750	106.8275	Indonesia	IDN	33756000.0	2.0
2	2	Delhi	28.6100	77.2300	India	IND	32226000.0	3.0
3	3	Guangzhou	23.1300	113.2600	China	CHN	26940000.0	4.0
4	4	Mumbai	19.0761	72.8775	India	IND	24973000.0	5.0

Рисунок 4 – Первые 5 строчек датасета

	id	city	lat	...	iso3	population	rank
44686	44686	Numto	63.6667	...	RUS	10.0	44355.0
44687	44687	Nord	81.7166	...	GRL	10.0	44355.0
44688	44688	Timmiarmiut	62.5333	...	GRL	10.0	44355.0
44689	44689	San Rafael	-16.7795	...	BOL	NaN	NaN
44690	44690	Nordvik	74.0165	...	RUS	0.0	44382.5

Рисунок 5 – Последние 5 строчек датасета

На рисунке 6 находится статистическая информация о датасете, а на рисунке 7 – информация о типах данных в наборе.

	id	lat	lng	population	rank
count	44691.000000	44691.000000	44691.000000	4.438400e+04	44384.000000
mean	22345.000000	25.933692	14.526105	1.143739e+05	22192.500000
std	12901.324777	23.225258	71.153080	7.148542e+05	12812.701256
min	0.000000	-54.933300	-179.600000	0.000000e+00	1.000000
25%	11172.500000	12.900000	-48.109750	1.218700e+04	11096.750000
50%	22345.000000	32.340000	13.500000	2.099900e+04	22187.000000
75%	33517.500000	43.333300	77.316550	4.800775e+04	33288.000000
max	44690.000000	81.716600	179.370300	3.773200e+07	44382.500000

Рисунок 6 - Статистическая информация о датасете

```

RangeIndex: 44691 entries, 0 to 44690
Data columns (total 8 columns):
#   Column      Non-Null Count  Dtype
---  -
0   id           44691 non-null  int64
1   city         44691 non-null  object
2   lat          44691 non-null  float64
3   lng          44691 non-null  float64
4   country      44691 non-null  object
5   iso3         44691 non-null  object
6   population   44384 non-null  float64
7   rank         44384 non-null  float64
dtypes: float64(4), int64(1), object(3)
memory usage: 2.7+ MB

```

Рисунок 7 – Информация о типах данных в датасете

Были удалены дубликаты строк (рисунок 8) и поле *ID* (рисунок 9). Также поле «*rank*» переименовано в «*rating*» (рисунок 10).

```
df = df.drop_duplicates()
```

Рисунок 8 – Удаление дубликатов строк

```
df = df.drop(columns=['id'])
```

Рисунок 9 – Удаление поля *ID*

```
df.rename(columns={'rank': 'rating'}, inplace=True)
```

Рисунок 10 – Переименование поля «*rank*»

Таким образом, датасет был загружен в программу и подготовлен для дальнейшей работы с ним.

## 2 Разведывательный анализ данных

Был проведён анализ данных датасета. Инструменты библиотеки *Matplotlib* и *SeaBorn* позволили визуализировать данные. Была построена гистограмма распределения городов по широтам планеты (рисунок 11).

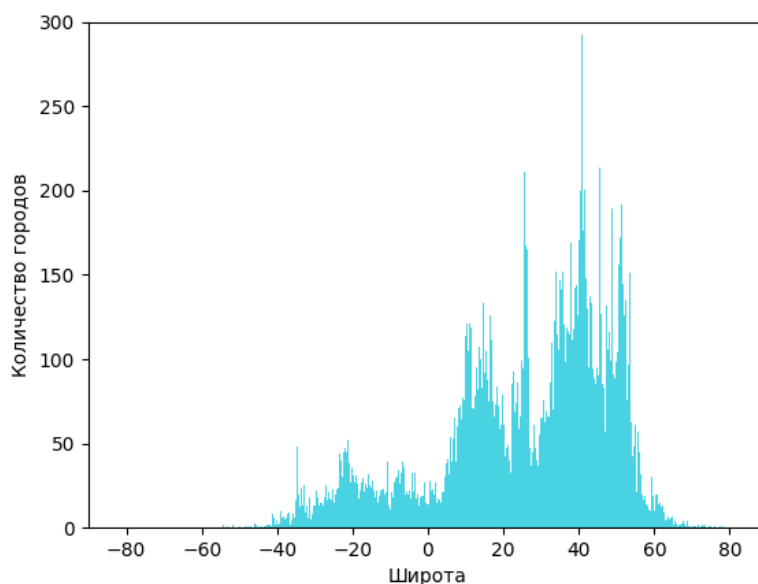


Рисунок 11 – Распределение городов по широтам

По гистограмме видно, что больше всего городов располагаются примерно на широте 42°.

Была построена диаграмма «ящик с усами» для долготы городов (рисунок 12).

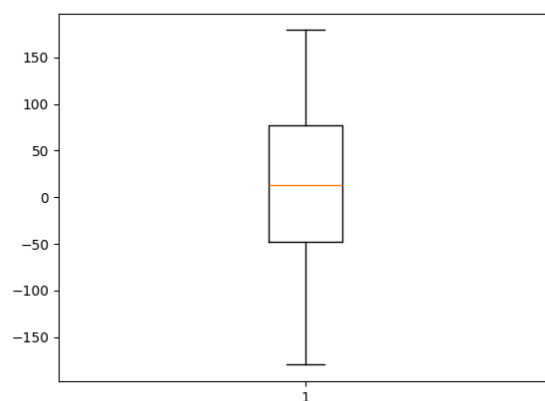


Рисунок 12 – Диаграмма «ящик с усами»



Эта диаграмма показывает, что медиана значений долготы приблизительно равна  $10^\circ$ . Её межквартильный размах идёт от  $-50^\circ$  до  $75^\circ$ .

Далее была построена круговая диаграмма, в которой представлены страны с наибольшим количеством городов. Поскольку стран очень много, для удобства на диаграмме показано только 15 наиболее крупных стран, оставшаяся доля представлена в части «*Other*» (рисунок 13).

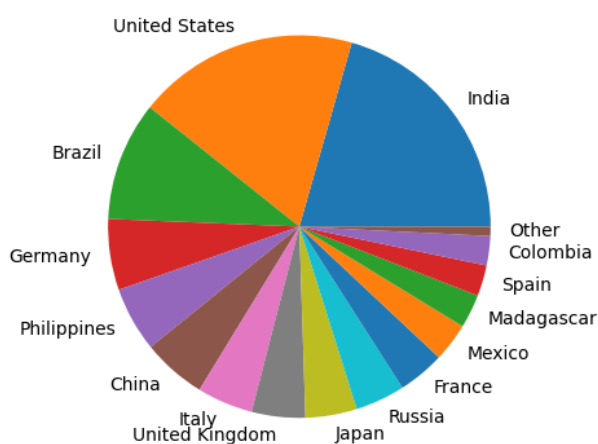


Рисунок 13 – Круговая диаграмма

По диаграмме видно, что на первом месте по наибольшему количеству городов находится Индия, на втором месте – США, на третьем – Бразилия.

Была построена тепловая карта со значениями взаимной корреляции между всеми парами признаков набора данных (рисунок 14).

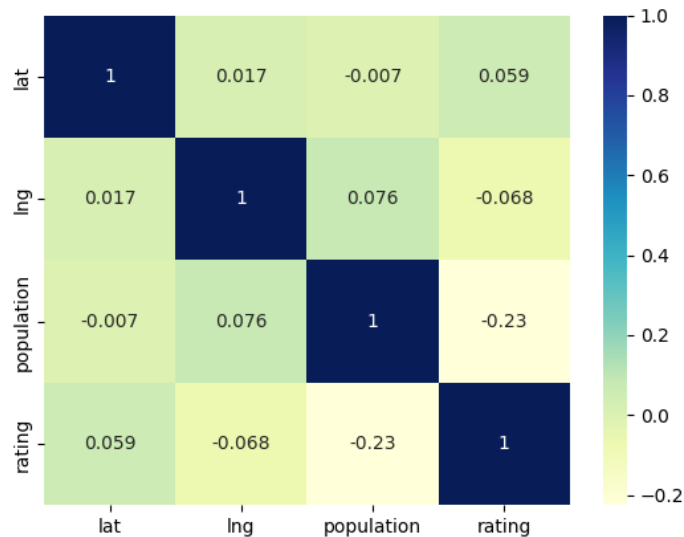


Рисунок 14 – Тепловая карта со значениями взаимной корреляции

Данная тепловая карта показывает отрицательную зависимость рейтинга города от его населения. Остальные пары признаков практически независимы друг от друга.

Также была построена диаграмма, показывающая население некоторых городов (рисунок 15).

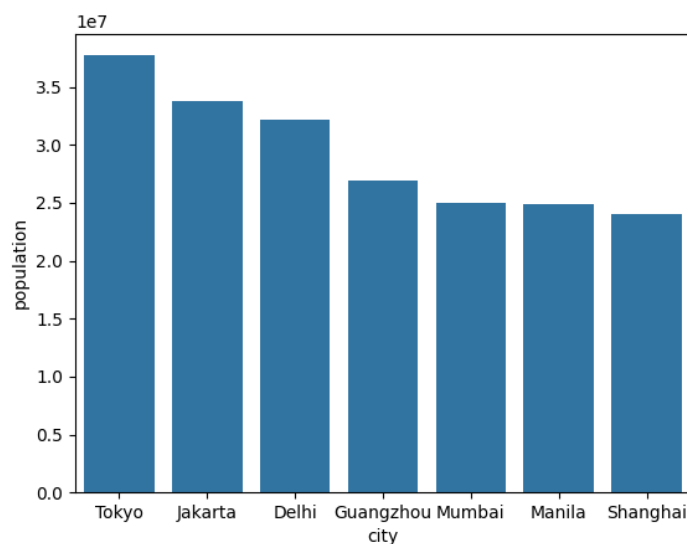


Рисунок 15 – Диаграмма населения городов

Среди выбранных городов самым густонаселённым является Токио.

Также была проанализирована на нормальность выборка из 200 первых значений населения городов (рисунок 16).

**Распределение не является нормальным**

Рисунок 16 – Результат проверки выборки на нормальность

Выборка не оказалась нормальной.

На рисунках 17-22 представлены реализации построения диаграмм и проверки выборки на нормальность.

```
lats = df['lat']
plt.xlim([-90, 90])
plt.ylim([0, 300])
plt.title('')
plt.xlabel('Широта')
plt.ylabel('Количество городов')
plt.hist(lats, bins=1000, color=get_random_color())
plt.savefig('data/latitude_hist.png')
plt.clf()
```

Рисунок 17 – Реализация построения гистограммы

```
longs = df['lng']
plt.boxplot(x=longs)
plt.savefig('data/longitude_boxplot.png')
plt.clf()
```

Рисунок 18 – Реализация построения диаграммы «ящик с усами»

```

uniq, counts = np.unique(df['country'], return_counts=True)
uniq_tuple = [(uniq[i], counts[i]) for i in range(len(uniq))]
uniq_tuple.sort(key=lambda x: -x[1])
cut_number = 15
extra_number = len(uniq_tuple) - cut_number
uniq_tuple = uniq_tuple[:cut_number]
counts = [x[1] for x in uniq_tuple]
counts.append(extra_number)
uniq = [x[0] for x in uniq_tuple]
uniq.append('Other')
plt.pie(counts, labels=[x for x in uniq])
plt.savefig('data/cities_number_in_countries.png')
plt.clf()

```

Рисунок 19 – Реализация построения круговой диаграммы

```

dataplot = sb.heatmap(df.corr(numeric_only=True), cmap="YlGnBu", annot=True)
plt.savefig('data/correlation.png')
plt.clf()

```

Рисунок 20 – Реализация построения тепловой карты

```

cut_df = df[:7]
sb.barplot(x="city", y="population", data=cut_df)
plt.savefig('data/countplot.png')
plt.clf()

```

Рисунок 21 – Реализация построения диаграммы населения городов

```

df_part = df[:200]
ntest = sp.stats.normaltest(df_part['population'])
if ntest[1] < 0.05:
    print('Распределение не является нормальным')
else:
    print('Распределение нормальное')

```

Рисунок 22 – Проверка выборки на нормальность

Таким образом, с помощью различных визуализаций был проанализирован датасет.

### 3 Предварительная обработка данных

Данные в датасете оказались неполными – нашлись пропуски в значениях. На рисунке 23 показан поиск полей, в которых присутствуют пустые значения, а на рисунке 24 – собственно поля с пропусками.

```
keys = df.columns.values
cols_with_empties = []
for k in keys:
    if any(df[k].isnull()):
        print(f"Столбец {k} имеет пустые ячейки")
        cols_with_empties.append(k)
```

Рисунок 23 – Поиск столбцов с пропусками

```
Столбец population имеет пустые ячейки
Столбец rating имеет пустые ячейки
```

Рисунок 24 – Поля с пропусками

Пропуски были заполнены по следующему правилу: если значением признака является целое число, то пропуск заполняется значением медианы по столбцу, если – действительное число, то в пустое значение становится равным средним значением по столбцу, иначе – пропуск заполняется значением моды. Реализация этого правила представлена на рисунке 25.

```
for k in cols_with_empties:
    filling = 0
    if df[k].dtype == int:
        filling = sp.ndimage.median(df[k])
    elif df[k].dtype == float:
        filling = np.mean(df[k])
    else:
        filling = sp.stats.mode(df[k])
    df[k] = df[k].fillna(filling)
```

Рисунок 25 – Заполнение пропусков значениями

Результат заполнения пустых значений показан на рисунке 26.

Было:							
	city	lat	lng	country	iso3	population	rating
44686	Numto	63.6667	71.3333	Russia	RUS	10.0	44355.0
44687	Nord	81.7166	-17.8000	Greenland	GRL	10.0	44355.0
44688	Timmiarmiut	62.5333	-42.2167	Greenland	GRL	10.0	44355.0
44689	San Rafael	-16.7795	-60.6799	Bolivia	BOL	NaN	NaN
44690	Nordvik	74.0165	111.5100	Russia	RUS	0.0	44382.5
Стало:							
	city	lat	lng	country	iso3	population	rating
44686	Numto	63.6667	71.3333	Russia	RUS	10.0	44355.0
44687	Nord	81.7166	-17.8000	Greenland	GRL	10.0	44355.0
44688	Timmiarmiut	62.5333	-42.2167	Greenland	GRL	10.0	44355.0
44689	San Rafael	-16.7795	-60.6799	Bolivia	BOL	114373.0	22192.0
44690	Nordvik	74.0165	111.5100	Russia	RUS	0.0	44382.5

Рисунок 26 – Результат заполнения пропусков значениями

Как видно, значения предпоследней строчки вместо *NaN* стали равны медиане по столбцу.

Предобработанные данные подверглись *one-hot* кодированию, и затем с помощью метода *to\_csv* были сохранены в файл *cities\_stats.csv* (рисунок 27).

```
df_encoded = pd.get_dummies(df)
df_encoded.to_csv(path_or_buf='data/cities_stats.csv', sep='\t')
```

Рисунок 27 – Кодирование и сохранение данных

Как выглядит файл с закодированными данными, представлено на рисунке 28.

```
t lng population rating city_A Coruña city_A Yun Pa city_Aabenraa city_Aachen city_Aadorf city_Aalborg,
city_Aalen city_Aaley city_Aalsmeer city_Aalst city_Aalten city_Aarau city_Aarhus city_Aarschot
ity_Aarsâl city_Aartselaar city_Aasiaat city_Aba city_Abadan city_Abadiânia city_Abadla city_Abadou
ity_Abaetetuba city_Abaeté city_Abaiara city_Abaji city_Abakaliki city_Abakan city_Abalessa
ity_Abancay city_Abangaritos city_Abano Terme city_Abarkūh city_Abarán city_Abaré city_Abashiri
ity_Abasingammedda city_Abasolo city_Abay city_Abaza city_Abai city_Abaíra city_Abbeville
ity_Abbiategrasso city_Abbigeri city_Abbots Langley city_Abbotsford city_Abbottabad city_Abcoude city_Abdul
akim city_Abdulino city_Abdullahnagar city_Abdurahmoni Jomí city_Abejorral city_Abelardo Luz
ity_Abengourou city_Abensberg city_Abeokuta city_Aberaman city_Aberbargoed city_Aberdare
```

Рисунок 28 – Файл с закодированными данными

Таким образом были обработаны данные датасета: устранены пропуски в значениях. Данные были закодированы и сохранены в *csv*-файл.

## ЗАКЛЮЧЕНИЕ

В ходе выполнения задания был:

- найден и загружен в программу датасет,
- написан *md*-файл, описывающий структуру датасета,
- произведен разведывательный анализ данных, в процессе которого были

построены:

- 1) гистограмма распределения городов по широтам,
  - 2) диаграмма «ящик с усами» по долготе городов,
  - 3) круговая диаграмма, показывающая страны с наибольшим числом городов,
  - 4) тепловая карта со значениями взаимной корреляции по всем парам признаков,
  - 5) диаграмма населения городов,
- проверена на нормальность выборка из 200 значений населения городов,
  - произведено *one-hot* кодирование всех категориальных признаков,
  - данные сохранены в файл *cities\_stats.csv*.

В результате выполнения задания получены начальные навыки анализа данных датасета, работы с датафреймами и изучены виды диаграмм – «ящик с усами» и тепловая карта значений взаимной корреляции.



## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Skillbox – образовательная платформа. Библиотека Matplotlib для построения графиков, 2023. URL: <https://skillbox.ru/media/code/biblioteka-matplotlib-dlya-postroeniya-grafikov/> (дата обращения 26.11.2024)
2. Skillbox – образовательная платформа. Библиотека Numpy: все, что нужно знать новичку, 2024. URL: <https://skillbox.ru/media/code/biblioteka-numpy-vsye-chto-nuzhno-znat-novichku/> (дата обращения 24.11.2024)
3. Википедия. Свободная энциклопедия. Ящик с усами, 2024. URL: [https://ru.wikipedia.org/wiki/Ящик\\_с\\_усами](https://ru.wikipedia.org/wiki/Ящик_с_усами) (дата обращения 26.11.2024)
4. Романовский Р. К. Элементы теории вероятностей и случайных процессов: учеб. пособие. Омск: Изд-во ОмГТУ, 2015. – 116с.
5. Сайт Pandas. Pandas Documentation, 2024. URL: <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html> (дата обращения 24.11.2024)
6. Сайт PythonRu. Руководство по Scipy: что это и как её использовать. URL: <https://pythonru.com/biblioteki/scipy-python> (дата обращения 24.11.2024)
7. Сайт Seaborn. seaborn-heatmap, 2024. URL: <https://seaborn.pydata.org/generated/seaborn.heatmap.html> (дата обращения 26.11.2024)