

Федеральное государственное автономное образовательное учреждение высшего  
образования

«Пермский государственный национальный исследовательский университет»

Институт Компьютерных Наук и Технологий

Лабораторная работа № 5

по дисциплине

«Введение в анализ данных»

Отчет

Студент Панькова Светлана

Группа ИТ-13-2023

Пермь 2025

## Задача №1

1. Импортируем нужные нам библиотеки

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2. Загрузите данные из файла «weather1.csv» о погоде в Перми. Загрузите только следующие столбцы: Местное время в Перми, T, P, U, Ff, N, H, VV

```
# 1. Загрузите данные из файла «weather1.csv» о погоде в
Перми.

data = pd.read_csv("weather1.csv", delimiter=";",
usecols=["Местное время в Перми", "T", "P", "U", "Ff",
"N", "H", "VV"])

# Преобразуем столбец "Местное время" в datetime для
корректности данных

# день идет перед месяцем dayfirst=True

data['Местное время в Перми'] =
pd.to_datetime(data['Местное время в Перми'],
dayfirst=True)

print('Вывод первых строк для просмотра данных:\n',
data.head())
```

Вывод первых строк для просмотра данных:

	Местное время в Перми	T	...	H	VV
0	2021-12-31 23:00:00	-9.8	...	600-1000	4.0
1	2021-12-31 20:00:00	-10.6	...	1000-1500	4.0
2	2021-12-31 17:00:00	-11.6	...	1000-1500	10.0
3	2021-12-31 14:00:00	-11.9	...	1000-1500	20.0
4	2021-12-31 11:00:00	-13.6	...	1000-1500	10.0

3. Постройте точечную диаграмму (диаграмму рассеяния) по признакам температуры и относительной влажности.

```
# 2. Постройте точечную диаграмму (диаграмму рассеяния)
# по признакам температуры и относительной влажности.
# T (температура воздуха в градусах Цельсия) и U
# (относительная влажность в %)

plt.figure(figsize=(10, 10)) # Устанавливаем размер
графика

plt.scatter(data['T'], data['U'], alpha=0.5) # Строим
точечную диаграмму, alpha=0.5 - прозрачность точек

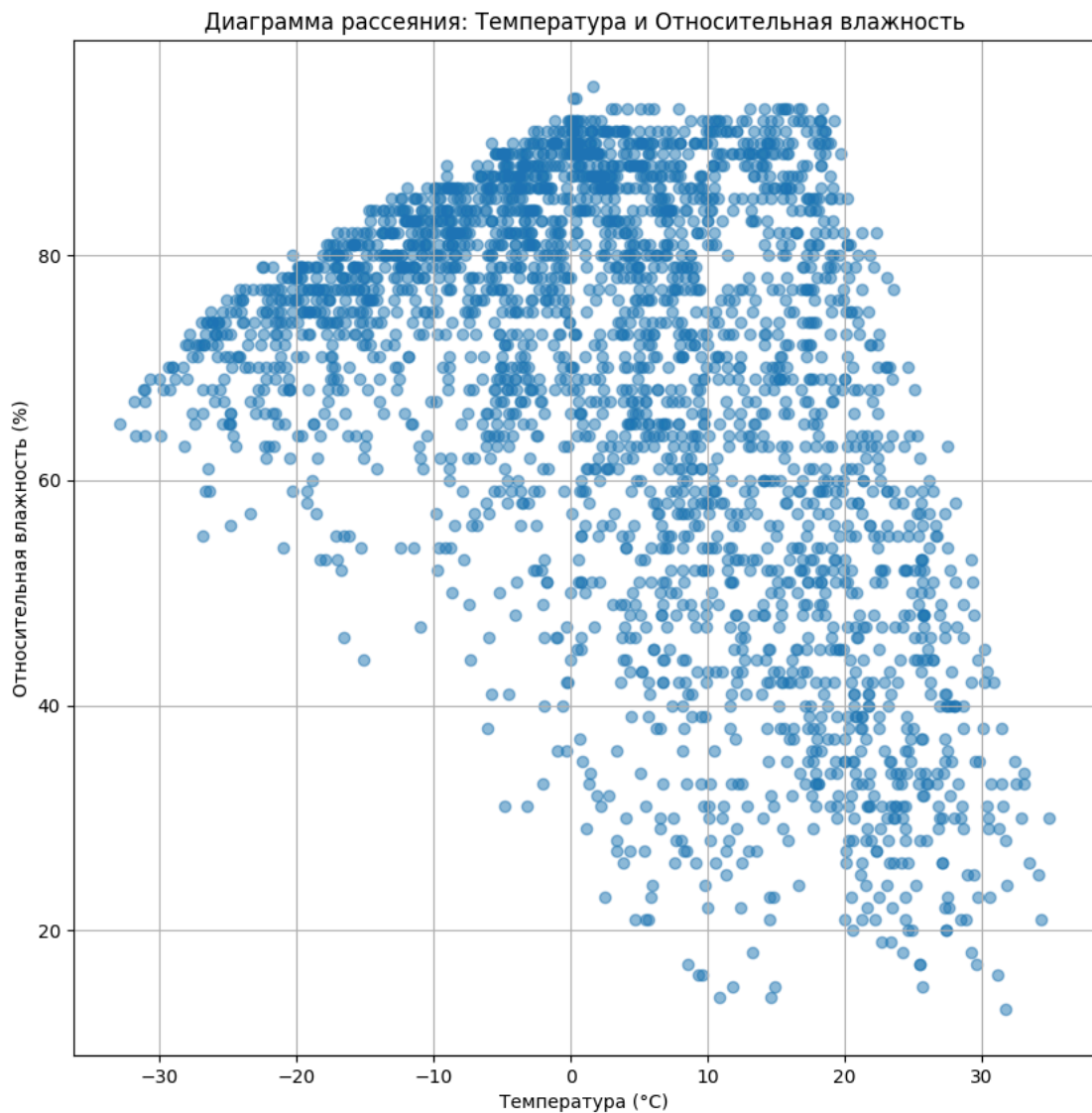
plt.title('Диаграмма рассеяния: Температура и
Относительная влажность')

plt.xlabel('Температура (°C)')

plt.ylabel('Относительная влажность (%)')

plt.grid() # Сетка для большей наглядности

plt.show()
```



4. На построенной в предыдущем пункте диаграмме выделите точки разными цветами в зависимости от облачности: синим – для которых облачность составляет 100%; красным – все остальные.

```
# 3. На построенной в предыдущем пункте диаграмме  
выделите точки разными цветами  
  
# в зависимости от облачности: синим – для которых  
облачность составляет 100%; красным – все остальные.  
  
plt.figure(figsize=(10, 10)) # Устанавливаем размер  
графика  
  
# Если облачность равна 100, точка будет синей, иначе –  
красной
```

```
# Строим точки для случаев, когда облачность равна 100%
# Значения по оси X берутся из столбца 'T', а по оси Y -
из столбца 'U'

plt.scatter(x=data[data["N"] == "100%."]["T"],
            y=data[data["N"] == "100%."]["U"],
            color="blue", alpha=0.5)

# Строим точки для случаев, когда облачность меньше 100%
plt.scatter(x=data[data["N"] != "100%."]["T"],
            y=data[data["N"] != "100%."]["U"],
            color="red", alpha=0.5)

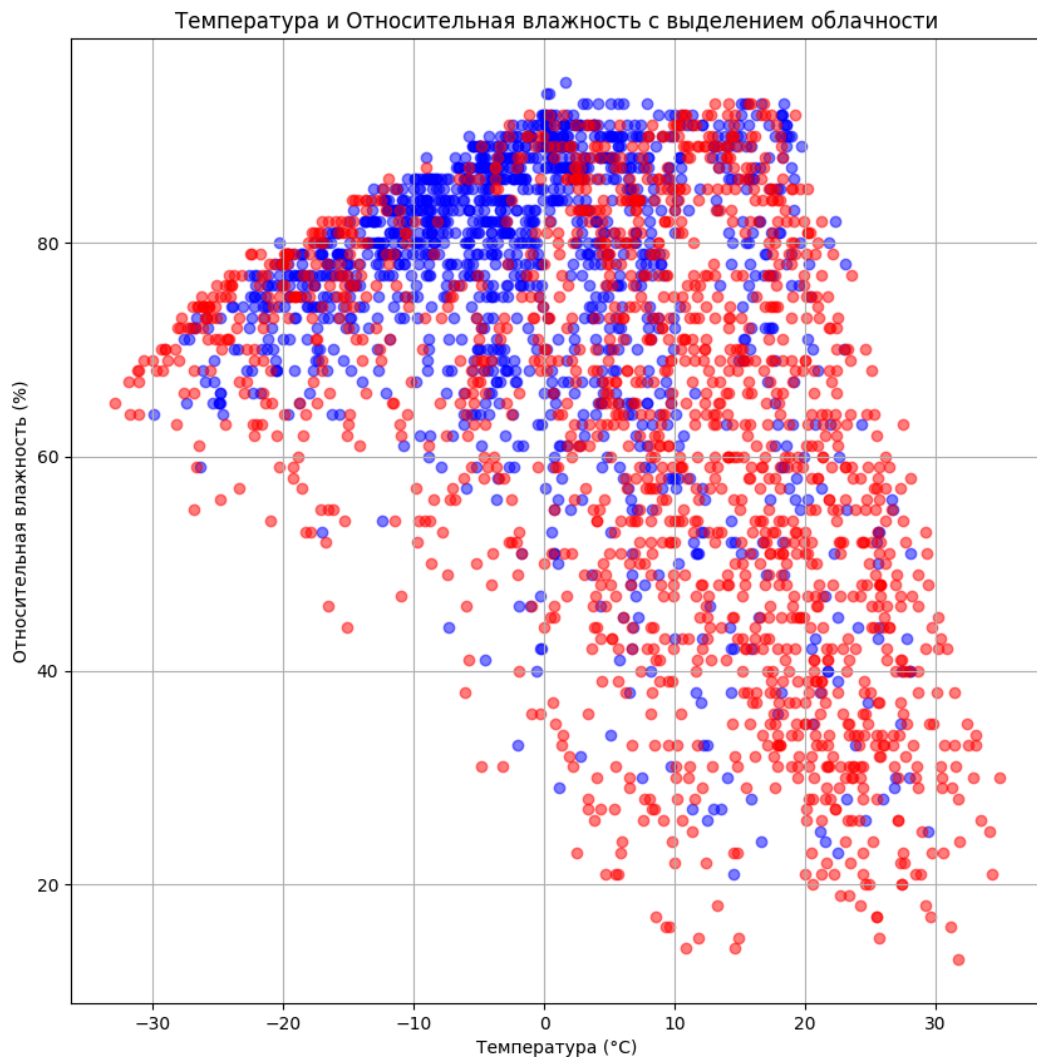
plt.title('Температура и Относительная влажность с
выделением облачности')

plt.xlabel('Температура (°C)')

plt.ylabel('Относительная влажность (%)')

plt.grid() # Сетка для большей наглядности

plt.show()
```



5. Постройте линейную диаграмму (график) изменения температуры в зависимости от местного времени.

```
# 4. Постройте линейную диаграмму (график) изменения
температуры в зависимости от местного времени.

plt.figure(figsize=(12, 12)) # Устанавливаем размер
графика

sns.lineplot(data=data, x='Местное время в Перми',
y='T') # Строим линейный график

plt.title('Изменение температуры в зависимости от
местного времени')

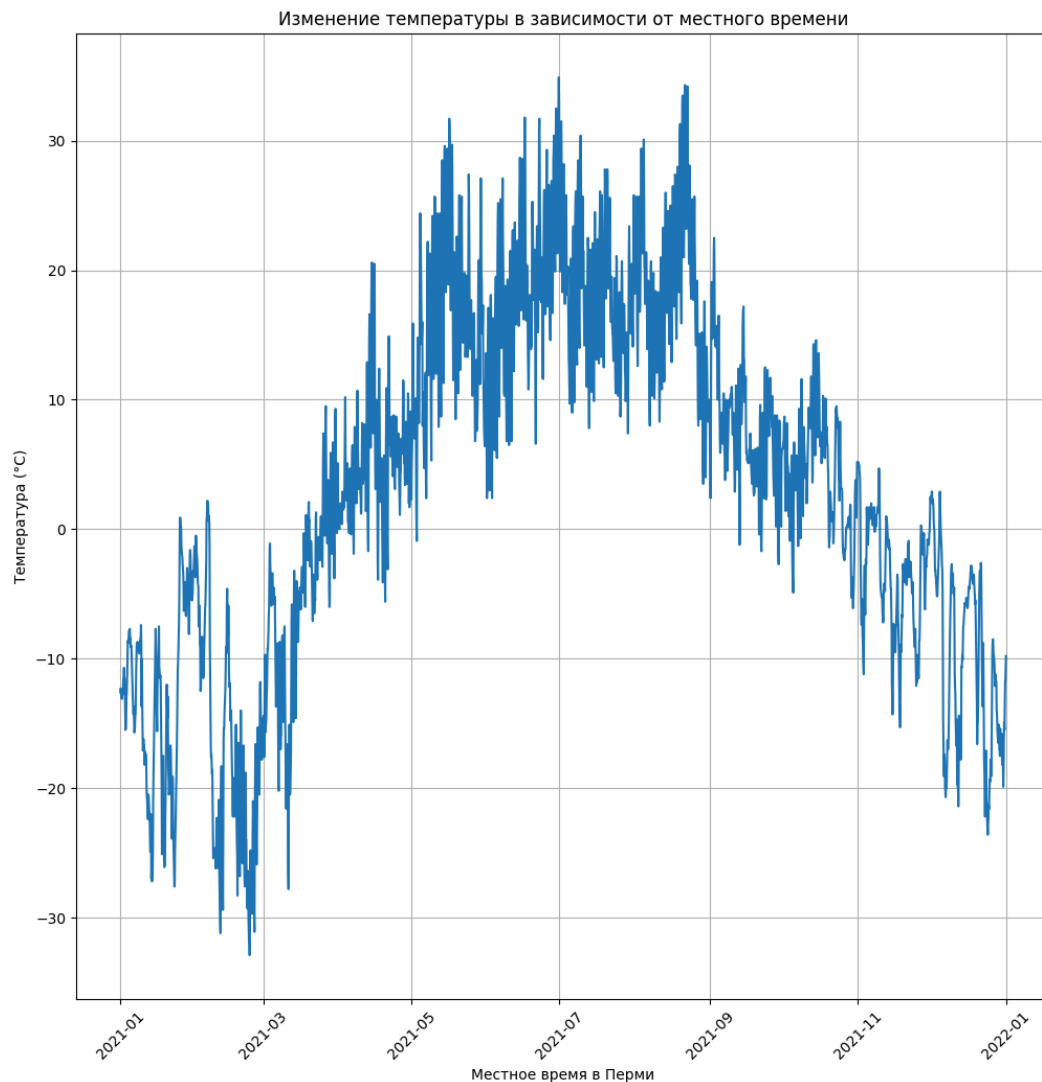
plt.xlabel('Местное время в Перми')

plt.ylabel('Температура (°C)')
```

```
plt.xticks(rotation=45) # Поворачиваем метки на оси X
для лучшей читаемости

plt.grid() # Сетка для большей наглядности

plt.show()
```



- Посчитайте по имеющимся данным среднемесячную температуру и постройте столбчатую диаграмму (вертикальную) зависимости средней температуры от месяца. Подсказка: создайте отдельный столбец с номером месяца (вычислив его из столбца «Местное время»), а затем сгруппируйте данные по этому столбцу.

```
# 5. Посчитайте по имеющимся данным среднемесячную
температуру и постройте столбчатую диаграмму
(вертикальную)
```

```
# зависимости средней температуры от месяца. Подсказка:
создайте отдельный столбец с номером месяца

# (вычислив его из столбца «Местное время»), а затем
сгруппируйте данные по этому столбцу.

data['Месяц'] = data['Местное время в Перми'].dt.month
# Извлекаем месяц из даты и создаем новый столбец


# Группируем данные по месяцам и вычисляем среднюю
температуру

# Сбрасываем индекс DataFrame после группировки и
агрегации

month_temp =
data.groupby('Месяц')['T'].mean().reset_index()

plt.figure(figsize=(10, 10)) # Устанавливаем размер
графика

sns.barplot(data=month_temp, x='Месяц', y='T') # Строим
столбчатую диаграмму для среднемесячной температуры

plt.title('Среднемесячная температура')

plt.xlabel('Месяц')

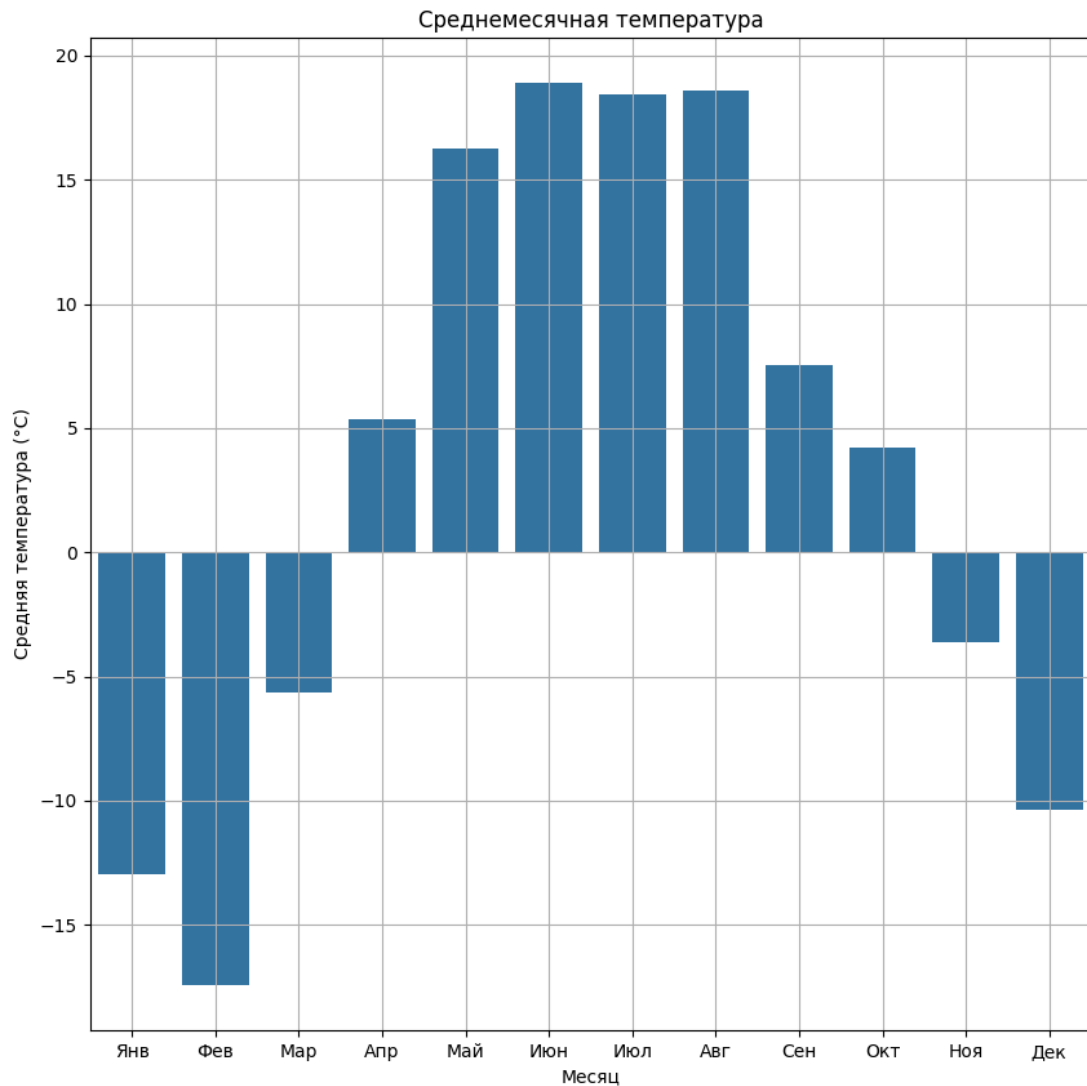
plt.ylabel('Средняя температура (°C)')

plt.xticks(ticks=range(12), labels=['Янв', 'Фев', 'Мар',
'Апр', 'Май', 'Июн', 'Июл', 'Авг', 'Сен', 'Окт', 'Ноя',
'Дек'])

plt.grid() # Сетка для большей наглядности

plt.show()
```





7. Постройте ленточную диаграмму (горизонтальную), отразив на ней количество имеющихся наблюдений для каждого варианта облачности.

```
# 6. Постройте ленточную диаграмму (горизонтальную),
отразив на ней количество имеющихся

# наблюдений для каждого варианта облачности. N
(облачность)

cloud_count = data['N'].value_counts().reset_index() #
Считаем количество наблюдений для каждой категории
облачности

cloud_count.columns = ['Облачность', 'Количество'] #
Переименовываем столбцы для удобства

print(cloud_count)
```

```
plt.figure(figsize=(10, 10)) # Устанавливаем размер
графика

sns.barplot(data=cloud_count, y='Облачность',
x='Количество', orient='h')

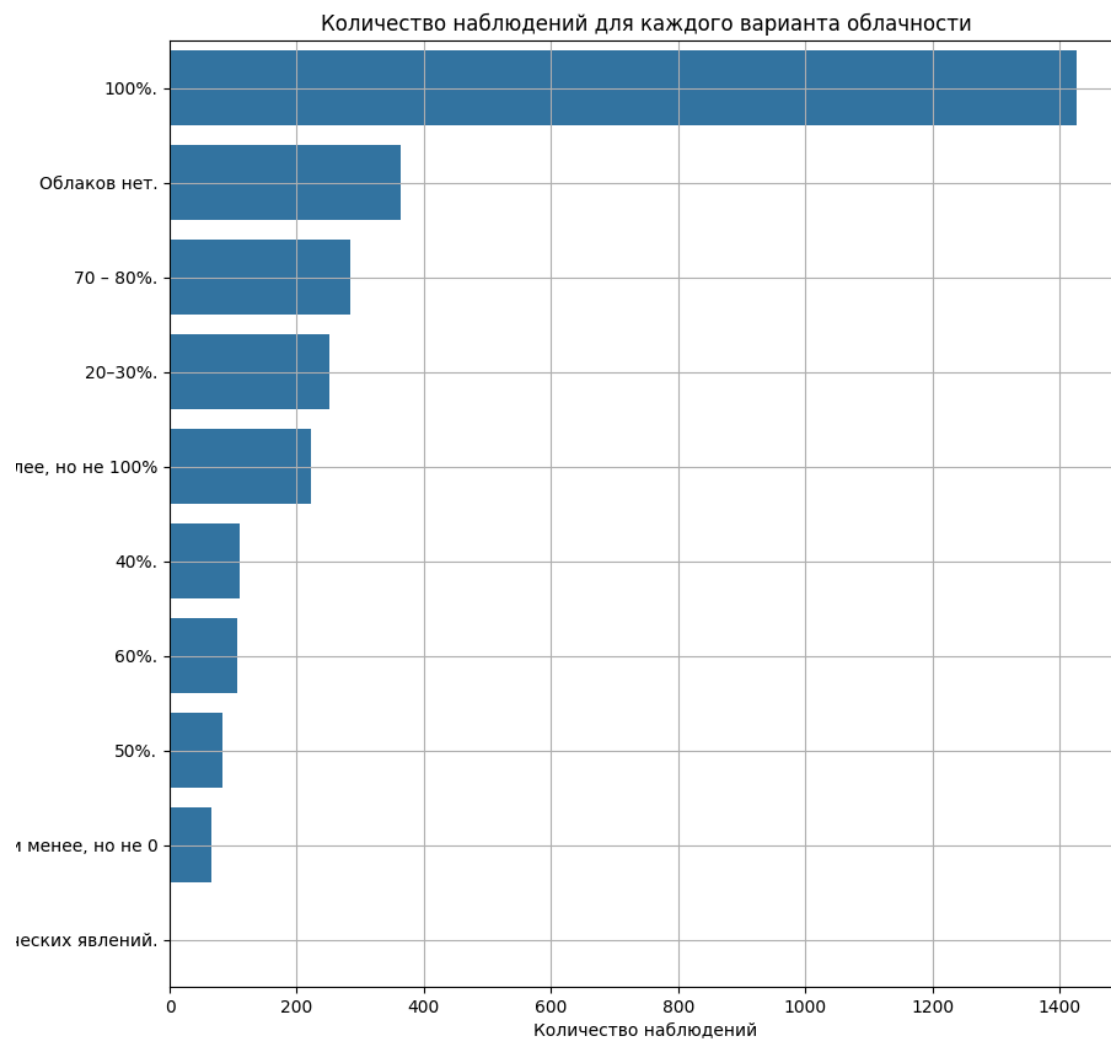
plt.title('Количество наблюдений для каждого варианта
облачности')

plt.xlabel('Количество наблюдений')

plt.ylabel('Облачность')

plt.grid() # Сетка для большей наглядности

plt.show()
```



	Облачность	Количество
0	100%.	1427
1	Облаков нет.	364
2	70 - 80%.	285
3	20-30%.	251
4	90 или более, но не 100%	222
5	40%.	111
6	60%.	106
7	50%.	83
8	10% или менее, но не 0	66
9	Небо не видно из-за тумана и/или других метеор...	1

8. Постройте гистограмму частот для температуры. На гистограмме должно быть 10 диапазонов температуры.

```
# 7. Постройте гистограмму частот для температуры. На
гистограмме должно быть 10 диапазонов температуры.

plt.figure(figsize=(10, 10)) # Устанавливаем размер
графика

sns.histplot(data['T'], bins=10) # Строим гистограмму
частот для температур

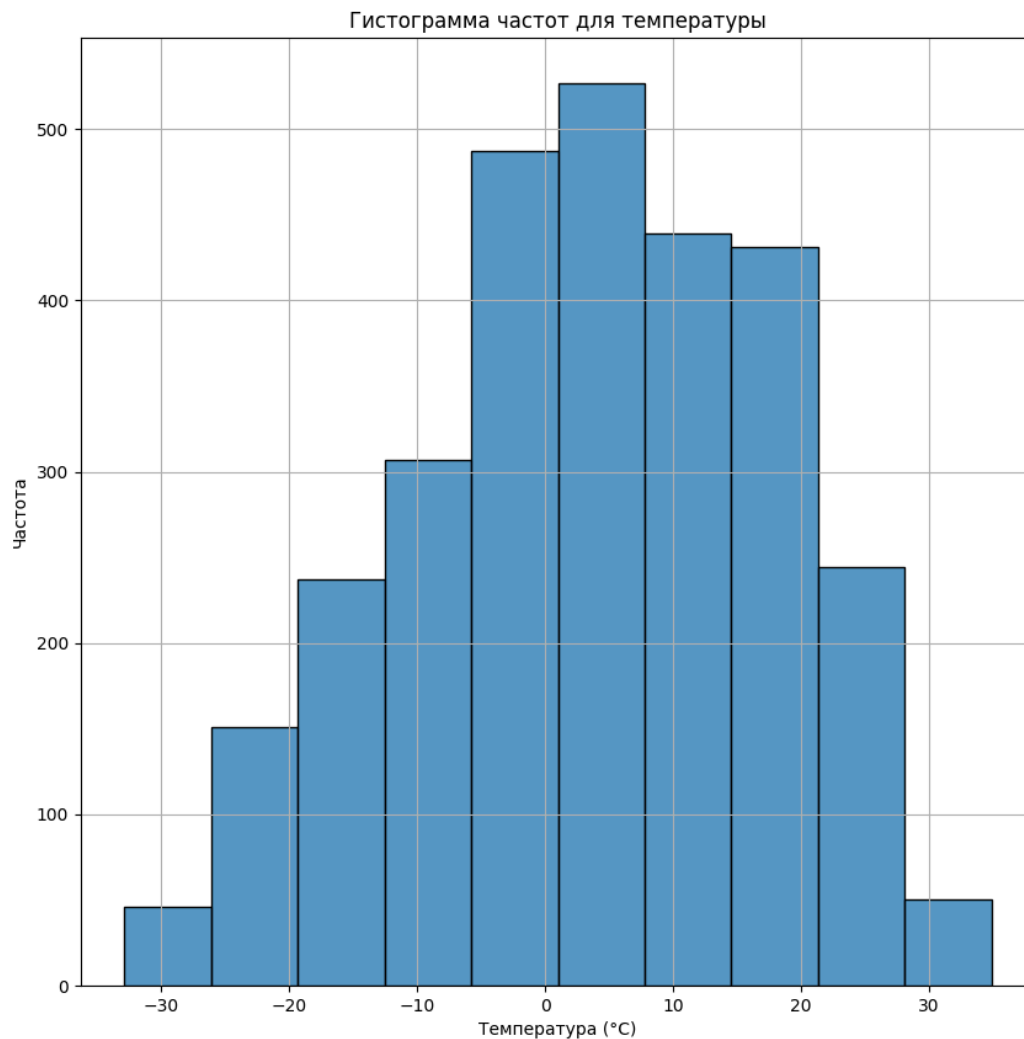
plt.title('Гистограмма частот для температуры')

plt.xlabel('Температура (°C)')

plt.ylabel('Частота')

plt.grid() # Сетка для большей наглядности

plt.show()
```



9. Разбейте данные на 3 группы по значению горизонтальной дальности видимости (одна группа – дальность видимости менее 5 км, вторая – от 5 до 15 км(включительно), третья – более 15 км). В одной области для каждой группы постройте boxplot (диаграмму «ящик с усами») для признака «атмосферное давление».

```
# 8. Разбейте данные на 3 группы по значению
горизонтальной дальности видимости

# (одна группа – дальность видимости менее 5 км, вторая
– от 5 до 15 км(включительно), третья – более 15 км).

# В одной области для каждой группы постройте boxplot
(диаграмму «ящик с усами») для признака «атмосферное
давление».
```

```
# Создаем группы видимости на основе расстояний
видимости (VV)

data['Группа видимости'] = pd.cut(data['VV'],
bins=[-float('inf'), 5, 15, float('inf')],
labels=['<5 км', '5-15
км', '>15 км'])

plt.figure(figsize=(12, 12)) # Устанавливаем размер
графика

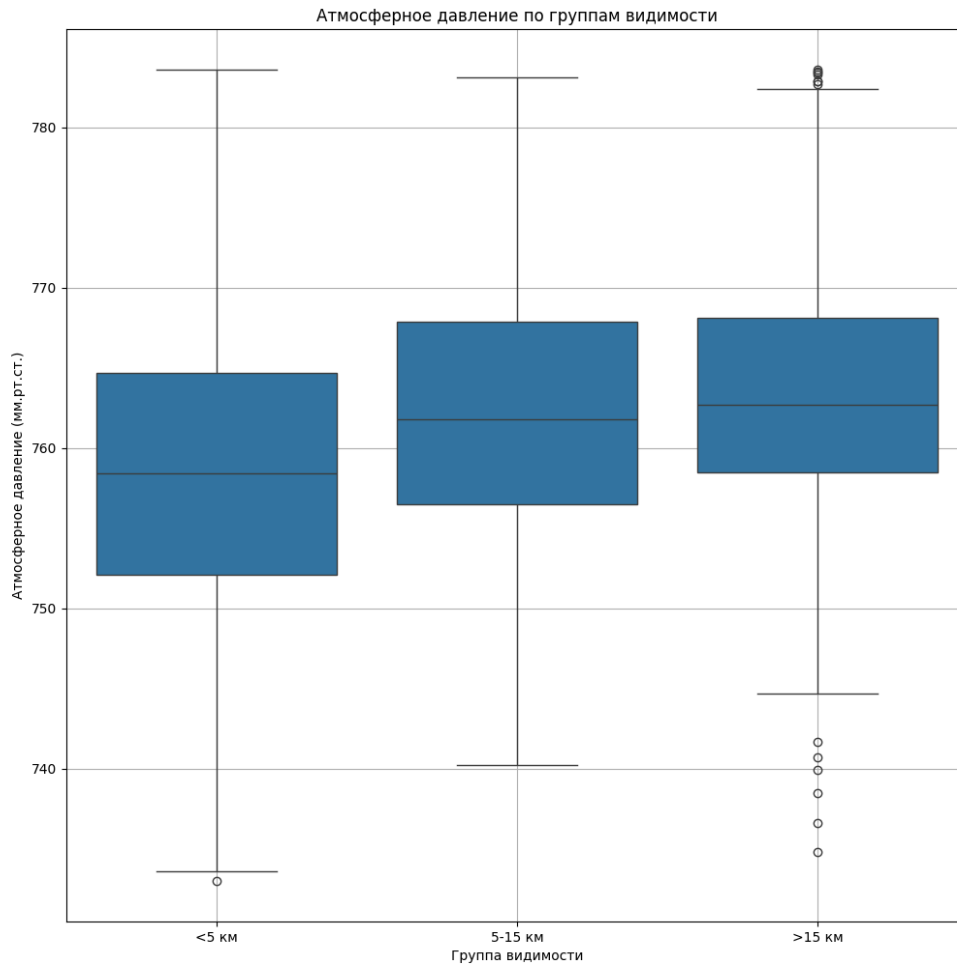
# Строим boxplot для атмосферного давления по группам
видимости

sns.boxplot(data=data, x='Группа видимости', y='P')

plt.title('Атмосферное давление по группам видимости')
plt.xlabel('Группа видимости')
plt.ylabel('Атмосферное давление (мм.рт.ст.)')

plt.grid() # Сетка для большей наглядности

plt.show()
```



10. Постройте круговую диаграмму для признака «высота основания облаков».

```
# 9. Постройте круговую диаграмму для признака «высота основания облаков».

# Считаем количество наблюдений для каждой высоты
основания облаков и создаем новый DataFrame

cloud_count = data['Н'].value_counts().reset_index()

cloud_count.columns = ['Высота основания облаков (м)',
'Количество']

plt.figure(figsize=(8, 8)) # Устанавливаем размер
графика для круговой диаграммы

# Строим круговую диаграмму, показывающую распределение
высоты основания облаков с процентами
```

```
plt.pie(cloud_count['Количество'],
labels=cloud_count['Высота основания облаков (м)'],
autopct='%1.1f%%')

plt.title('Распределение высоты основания облаков')

plt.axis('equal') # Устанавливаем равные масштабы по осям для круговой диаграммы

plt.grid() # Сетка для большей наглядности

plt.show()
```

