**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

Факультет «Информатика и системы управления»

Кафедра «Теоретическая информатика и компьютерные технологии»

# Лабораторная работа № 4

## по курсу «Моделирование»

ОБРАБОТКА И АНАЛИЗ БОЛЬШИХ ДАННЫХ

Студент: Яровикова А. С.

Группа: ИУ9-81Б

Преподаватель: Домрачева А. Б.

Москва, 2024

# ЦЕЛЬ И ПОСТАНОВКА ЗАДАЧИ

## Цель

Целью данной работы является анализ больших данных и построение прогностической модели на основе предварительно обработанных данных. Необходимо провести очистку от выбросов, обработку пропусков, отбор целевого признака и признаков, подаваемых на вход модели, а также непосредственно выполнить обучение модели.

## Постановка задачи

Рассматривается датасет по недвижимости в России за 2021 год (https://www.kaggle.com/datasets/mrdaniilak/russia-real-estate-2021/data).

Необходимо:

- выделить регион Санкт-Петербург
- очистить данные от выбросов:
    - построить тепловую карту
    - построить гистограммы распределения и диаграммы рассеяния
    - отрезать выбросы по стандартному отклонению (z-индекс)
    - построить корреляционную матрицу
- в отчет обязательно включить описание данных:
    - использовать встроенные методы pandas
    - есть ли пропуски были ли обнаружены аномальные значения
    - как обрабатывались выбросы и как принималось решение, что это выброс
- разделить выборку на несколько в зависимости от стоимости и площади
- взять модель множественной линейной регрессии
- отобрать признаки, по которым будем обучать
- применить масштабирование данных с помощью StandardScaler. Попробовать как масштабировать таргет, так и нет (если масштабируем то можно, к примеру, применить логарифмирование)
- разделить выборку на обучающую и тестовую (метод из sklearn)

- обучить модель

- посчитать коэффициент-$R^2$ и среднюю абсолютную ошибку

## ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

### Обработка дубликатов и пропусков

Часто при анализе данных возникают проблема наличия дубликатов записей. Если не найти дубликаты, то анализ данных может привести к некорректным результатам исследования. Для эффективного выявления дублированных записей применяются различные методы.

Один из методов основан на использовании функции **duplicated()**, которая возвращают логические значения, указывающие на присутствие или отсутствие дубликатов в наборе данных. Этот метод может быть дополнен функцией **sum()** для подсчета общего количества обнаруженных дубликатов.

Второй метод основан на применении функции **value_counts()**, которая выдает список уникальных значений с их частотой появления в данных, упорядоченный по убыванию. Это позволяет выделить наиболее часто встречающиеся дубликаты и провести дальнейший анализ данных.

Выявление и обработка пропущенных значений также является необходимым шагом подготовки данных. Для получения списка всех уникальных значений в заданном столбце используется функция **unique()**. Удаление строк с пропущенными значениями осуществляется с помощью метода **dropna()**, который исключает строки с пропущенными значениями из исходного набора данных. После удаления строк возможно провести переназначение индексов строк данных в соответствии с новым порядком данных с помощью метода **reset_index(drop=True)**.

Особые значения **NaN** и **None** обозначают отсутствие значений в ячейке. **NaN** отвечает за отсутствующее в ячейке число. Его тип данных float, поэтому с NaN можно проводить математические операции. А **None** относится к нечисловому типу **NoneType**, и математические операции с ним неосуществимы.

Важно отметить, что пропущенные значения могут искажать результаты анализа данных и требуют специальной обработки, в том числе иногда удаления.

Метод **value_counts()** возвращает уникальные значения с их количеством в наборе данных. Метод **isnull()** возвращает булевский список, показывающий

присутствие или отсутствие пропущенных значений в столбце (True означает, что значение в колонке пропущено). Для замены пропусков на какое-то значение, применяется метод **fillna(value)**.

**Обработка выбросов**

Выбросы в данных представляют собой аномальные значения, которые существенно отклоняются от основной массы наблюдений и могут внести искажения в результаты анализа.

Одним из методов выявления и обработки выбросов является анализ, основанный на стандартном отклонении. Стандартное отклонение (обозначается как $\sigma$) является мерой разброса значений относительно их среднего значения $\mu$, и определяется формулой:

$$\sigma = \sqrt{\sum_{i=1}^{N} \frac{(x_i - \mu)^2}{N}},$$

где

- $x_i$ – отдельное наблюдение,
- $\mu$ – среднее значение,
- $N$ – количество наблюдений.

Выбросы могут быть идентифицированы как значения, находящиеся за пределами заданного диапазона от среднего значения, например, более чем на $k$ стандартных отклонений от среднего. Это можно выразить следующим образом:

$$\text{Выброс} = \{x_i : |x_i - \mu| > k\sigma\},$$

где $k$ - множитель, определяющий ширину диапазона для выявления выбросов.

Применение данного метода позволяет выделить и удалить аномальные значения, уменьшая их влияние на статистические показатели и результаты анализа. Это способствует повышению точности и достоверности интерпретируемости выводов, полученных в результате обработки данных.

**Модель множественной линейной регрессии**

Множественная линейная регрессия является статистическим методом, используемым для прогнозирования значения зависимой переменной на основе нескольких независимых переменных.

В контексте нашей задачи множественная регрессия позволяет моделировать зависимость стоимости недвижимости от разных факторов, таких как площадь помещения, количество этажей в доме, близость к центру города и т.п.

Модель множественной регрессии определяется уравнением:

$$Y = \alpha_0 + \alpha_1 X_1 + \cdots + \alpha_n X_n + \varepsilon\,,$$

где:

- $Y$ – зависимая переменная,
- $X_1, X_2, \ldots, X_n$ – независимые переменные,
- $\alpha_0, \ldots, \alpha_n$ – коэффициенты регрессии, параметры модели,
- $\varepsilon$ – случайная ошибка.

Идея состоит в том, чтобы оценить значения коэффициентов регрессии, которые минимизируют сумму квадратов остатков (расхождений между фактическими и прогнозируемыми значениями), что обеспечивает наилучшее приближение модели к данным.

**Оценки точности модели**

Для выбора наиболее подходящей модели и оценки ее производительности часто используются различные метрики. Например, коэффициент $R^2$ и средняя абсолютная ошибка (MAE).

**Коэффициент $R^2$** представляет собой меру соответствия модели данным и указывает на то, какую долю дисперсии зависимой переменной объясняет модель. Он вычисляется по формуле:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}\quad,$$

где

– $y_i$ – фактические значения зависимой переменной,

– $\widehat{y_i}$ – предсказанные значения зависимой переменной моделью,

– $\widehat{y_i}$ – среднее значение зависимой переменной, $n$ - количество наблюдений.

Коэффициент $R^2$ может принимать значения от 0 до 1, где 1 указывает на идеальное соответствие модели данным, а значение 0 означает, что модель не объясняет никакой доли вариации.

**Средняя абсолютная ошибка MAE** представляет собой среднее абсолютное значение разницы между фактическими и предсказанными значениями. Она вычисляется как среднее значение абсолютных значений остатков:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}|,$$

где

– $y_i$ – фактическое значение зависимой переменной,

– $\widehat{y_i}$ – предсказанное значение,

– $n$ – количество наблюдений.

MAE позволяет оценить среднюю величину ошибки модели и ее способность предсказывать значения зависимой переменной. Чем ниже значение MAE, тем лучше качество модели.

**Масштабирование данных**

Масштабирование данных – процесс преобразования значений признаков таким образом, чтобы они находились в определенном диапазоне или имели определенное распределение.

Одним из распространенных методов масштабирования является стандартизация, или масштабирование по стандартному методу.

Стандартизация основана на преобразовании значения признака по формуле:

$$x_{scale} = \frac{x - \mu}{\sigma},$$

где

- $x$ - исходное значение признака,
- $\mu$ - среднее значение признака,
- $\sigma$ - стандартное отклонение признака,
- $x_{scale}$ - масштабированное значение признака.

Процесс стандартизации делает значения признаков такими, что они имеют среднее значение 0 и стандартное отклонение 1. Это полезно для сравнения и анализа признаков, которые изначально имеют разные единицы измерения. Также это улучшает работу алгоритмов машинного обучения, которые зависят от масштаба признаков.

## ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ

В данном задаче были использованы выборки мужских и женских доходов из набора данных по покупке беговых дорожек для фитнеса. Всего было 104 и 76 записей в выборках для мужчин и женщин соответственно. Данные были поделены на обучающие и контрольные выборки – 78 и 26 значений для мужских доходов и 57 и 19 – для женских, соответственно.

Исходный код лабораторной работы представлен ниже.

Импортирование необходимых библиотек:

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
```

Загрузка датасета:

```
df = pd.read_csv('dataset/input_data.csv', sep = ';')
df.head()
```

Извлечение Необходимого региона – Санкт-Петербруга:

```
saint_petersburg_df = df[df.id_region == 78]
saint_petersburg_df
```

Добавление новых столбцов и удаление неактуальных для удобства анализа:

```
saint_petersburg_df['date'] = pd.to_datetime(saint_petersburg_df['date'])

saint_petersburg_df['year'] = saint_petersburg_df['date'].dt.year
saint_petersburg_df['month'] = saint_petersburg_df['date'].dt.month
saint_petersburg_df = saint_petersburg_df.drop(['date'], axis = 1)
saint_petersburg_df = saint_petersburg_df.drop(['id_region'], axis = 1)

saint_petersburg_df
```

Для построения тепловой карты используется библиотека *folium*:

```
import folium
from folium.plugins import HeatMap
mapa = folium.Map(location=[saint_petersburg_df.geo_lat.mean(),
saint_petersburg_df.geo_lon.mean()], zoom_start=6)
map_values = saint_petersburg_df[['geo_lat', 'geo_lon', 'price']]
data = map_values.values.tolist()
hm = HeatMap(data, min_opacity=0.05, max_opacity=0.9, radius=25).add_to(mapa)
```

```
mapa
```

Для построения гистограмм распределения и диаграмм рассеяния используется библиотека *seaborn*. Метод *subplots()* позволяет расположить несколько диаграмм на одном графике, что помогает эффективно проводить анализ и сравнение:

```
fig, axes = plt.subplots(4, 2, figsize = (25, 25))
sns.set()

sns.histplot(data = saint_petersburg_df.price, ax = axes[0, 0])
axes[0, 0].set_title('Распределение цены', loc = 'left')

sns.histplot(data = saint_petersburg_df.area, ax = axes[2, 0])
axes[2, 0].set_title('Распределение площадей квартир', loc = 'left')

sns.histplot(data = saint_petersburg_df.building_type, ax = axes[3, 0])
axes[3, 0].set_title('Распределение типов домов', loc = 'left')

sns.histplot(data = saint_petersburg_df.object_type, ax = axes[3, 1])
axes[3, 1].set_title('Распределение типов квартир', loc = 'left')

sns.histplot(data = saint_petersburg_df.level, ax = axes[0, 1])
axes[0, 1].set_title('Распределение количества этажей', loc = 'left')

sns.histplot(data = saint_petersburg_df.levels, ax = axes[1, 0])
axes[1, 0].set_title('Распределение этажей квартир в объявлениях', loc = 'left')

sns.histplot(data = saint_petersburg_df.rooms, ax = axes[1, 1])
axes[1, 1].set_title('Распределение количества комнат', loc = 'left')

sns.histplot(data = saint_petersburg_df.kitchen_area, ax = axes[2, 1])
axes[2, 1].set_title('Распределение площадей кухонь', loc = 'left')

plt.show()

fig, axes = plt.subplots(4, 2, figsize = (25, 25))
sns.set()
fig.suptitle('Санкт-Петербург')

sns.scatterplot(x='level', y='price', data=saint_petersburg_df, ax = axes[0, 0],
alpha = 0.01)

sns.scatterplot(x='levels', y='price', data=saint_petersburg_df, ax = axes[0, 1],
alpha = 0.01)

sns.scatterplot(x='rooms', y='price', data=saint_petersburg_df, ax = axes[1, 0],
alpha = 0.01)
```

```
sns.scatterplot(x='area', y='price', data=saint_petersburg_df, ax = axes[1, 1],
alpha = 0.01)

sns.scatterplot(x='kitchen_area', y='price', data=saint_petersburg_df, ax = axes[2,
0], alpha = 0.01)

sns.scatterplot(x='building_type', y='price', data=saint_petersburg_df, ax =
axes[2, 1], alpha = 0.01)

sns.scatterplot(x='object_type', y='price', data=saint_petersburg_df, ax = axes[3,
0], alpha = 0.01)
plt.show()
```

Построение корреляционной матрицы:

```
# корреляционная матрица
plt.figure(figsize=(15, 10))
sns.heatmap(saint_petersburg_df.corr(), center=0, cmap='mako', annot=True)
plt.title('Корреляционная матрица')
plt.show()
```

Информация о датасете:

```
saint_petersburg_df.info(verbose = True, show_counts = True)
```

Информация о пропусках:

```
saint_petersburg_df.isna().sum()
```

Удаление пропусков. Столбцы, в которых наблюдаются пропуски, удаляются, так как не имеют корреляции с стоимостью недвижимости:

```
saint_petersburg_df.replace([np.inf, -np.inf], np.nan, inplace=True)
saint_petersburg_df.isna().sum()

# Удаляем ненужные колонки с пропусками
saint_petersburg_df.drop(columns = ['postal_code', 'street_id', 'house_id',
'year'], inplace = True)
saint_petersburg_df
```

Информация о дубликатах:

```
saint_petersburg_df.duplicated().sum()
```

Удаление дубликатов:

```
saint_petersburg_df = saint_petersburg_df.drop_duplicates()
saint_petersburg_df.duplicated().sum()
```

Замена отрицательных значений для комнат и кухонь. Предположительно наличие отрицательных значений связано с ошибкой опечатки:

```
print(saint_petersburg_df.rooms.unique())
```

```
saint_petersburg_df['rooms'] = saint_petersburg_df["rooms"].apply(lambda x: -x if x
< 0 else x)

print(saint_petersburg_df.kitchen_area.unique())
saint_petersburg_df['kitchen_area'] =
saint_petersburg_df["kitchen_area"].apply(lambda x: -x if x < 0 else x)

# Если значения этажей в доме больше, чем этаж квартиры, то меняем их местами (так
как это скорее просто ошибка)
saint_petersburg_df.loc[saint_petersburg_df["level"] >
saint_petersburg_df["levels"], "level"] = saint_petersburg_df["levels"]
```

Отсечение выбросов по стандартному отклонению:

```
# Отрезаем по стандартному отклонению
def remove_outliers(df, columns):
    for column in columns:
        z_scores = (df[column] - df[column].mean()) / df[column].std()
        df = df[(z_scores.abs() < 1.5)]
    return df

numeric_columns = ['geo_lat', 'geo_lon', 'price', 'area', 'kitchen_area']
clean_data = remove_outliers(saint_petersburg_df, numeric_columns)
```

Масштабирование:

```
# Масштабирование
from sklearn.preprocessing import StandardScaler

X = data[['level', 'levels', 'rooms', 'area', 'kitchen_area', 'geo_lat', 'geo_lon',
'building_type', 'object_type']]
Y = data['price']

scaler = StandardScaler()
# Масштабирование данных
X = scaler.fit_transform(X)
```

Деление на обучающую и тестовую выборку и обучение модели:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2,
random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
print('Accuracy of Linear Regression on training data', model.score(X_train,
y_train))
print('Accuracy of Linear Regression on testing data', model.score(X_test, y_test))
y_pred = model.predict(X_test)
```

Метрики:

```
from sklearn.metrics import mean_absolute_error, r2_score
# коэффициент R^2
r2_score(y_test, y_pred)
# средняя абсолютная ошибка
mean_absolute_error (y_tests, y_pred)

saint_petersburg_df['price'].mean()
```

# РЕЗУЛЬТАТЫ

Данные загружены с помощью библиотеки *pandas*, после чего из них выделена выборка в соответствии с регионом 78 – городом Санкт-Петербург.

```
df = pd.read_csv('input_data.csv', sep = ';')
df
```

| | date | price | level | levels | rooms | area | kitchen_area | geo_lat | geo_lon | building_type | object_type | postal_code | street_id | id_region | house_id |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2021-01-01 | 2451300 | 15.0 | 31.0 | 1.0 | 30.3 | 0.0 | 56.780112 | 60.699355 | 0.0 | 2.0 | 620000.0 | NaN | 66.0 | 1632918.0 |
| **1** | 2021-01-01 | 1450000 | 5.0 | 5.0 | 1.0 | 33.0 | 6.0 | 44.608154 | 40.138381 | 0.0 | 0.0 | 385000.0 | NaN | 1.0 | NaN |
| **2** | 2021-01-01 | 10700000 | 4.0 | 13.0 | 3.0 | 85.0 | 12.0 | 55.540060 | 37.725112 | 3.0 | 0.0 | 142701.0 | 242543.0 | 50.0 | 681306.0 |
| **3** | 2021-01-01 | 3100000 | 3.0 | 5.0 | 3.0 | 82.0 | 9.0 | 44.608154 | 40.138381 | 0.0 | 0.0 | 385000.0 | NaN | 1.0 | NaN |
| **4** | 2021-01-01 | 2500000 | 2.0 | 3.0 | 1.0 | 30.0 | 9.0 | 44.738685 | 37.713668 | 3.0 | 2.0 | 353960.0 | 439378.0 | 23.0 | 1730985.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **26337** | 2021-01-03 | 1950000 | 5.0 | 5.0 | 1.0 | 32.0 | -100.0 | 56.517950 | 85.049590 | 4.0 | 0.0 | 634040.0 | NaN | 70.0 | NaN |
| **26338** | 2021-01-03 | 2779700 | 18.0 | 25.0 | 1.0 | 41.8 | -100.0 | 47.245329 | 39.701206 | 3.0 | 2.0 | 344038.0 | 203005.0 | 61.0 | 1816091.0 |
| **26339** | 2021-01-03 | 2779700 | 23.0 | 25.0 | 1.0 | 41.8 | -100.0 | 47.245329 | 39.701206 | 3.0 | 2.0 | 344038.0 | 203005.0 | 61.0 | 1816091.0 |
| **26340** | 2021-01-03 | 2779700 | 1.0 | 25.0 | 1.0 | 41.8 | -100.0 | 47.245329 | 39.701206 | 3.0 | 2.0 | 344038.0 | 203005.0 | 61.0 | 1816091.0 |
| **26341** | 2021-01-03 | 3500000 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

26342 rows × 15 columns

*Рисунок 1 - Данные из датасета*

```
saint_petersburg_df = df[df.id_region == 78]
saint_petersburg_df
```

| | date | price | level | levels | rooms | area | kitchen_area | geo_lat | geo_lon | building_type | object_type | postal_code | street_id | id_region | house_id |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **15** | 2021-01-01 | 8865000 | 10.0 | 13.0 | 2.0 | 66.7 | 14.6 | 59.851179 | 30.411657 | 4.0 | 2.0 | 192288.0 | 569976.0 | 78.0 | 1690045.0 |
| **63** | 2021-01-01 | 4200000 | 12.0 | 25.0 | -1.0 | 25.8 | -100.0 | 60.036889 | 30.226123 | 3.0 | 2.0 | 190000.0 | 291085.0 | 78.0 | 2484205.0 |
| **99** | 2021-01-01 | 6574995 | 10.0 | 11.0 | -1.0 | 31.0 | -100.0 | 59.936744 | 30.251800 | 3.0 | 0.0 | 199406.0 | 172995.0 | 78.0 | 1780424.0 |
| **108** | 2021-01-01 | 7068330 | 12.0 | 13.0 | 1.0 | 42.3 | 18.7 | 55.638429 | 37.652912 | 0.0 | 2.0 | NaN | NaN | 78.0 | NaN |
| **151** | 2021-01-01 | 8865000 | 11.0 | 13.0 | 2.0 | 66.7 | 14.6 | 59.851179 | 30.411657 | 4.0 | 2.0 | 192288.0 | 569976.0 | 78.0 | 1690045.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **39436** | 2021-01-05 | 5450000 | 5.0 | 12.0 | -1.0 | 25.8 | -100.0 | 59.926208 | 30.295114 | 3.0 | 2.0 | 190000.0 | 403955.0 | 78.0 | 2056222.0 |
| **39475** | 2021-01-05 | 5850174 | 1.0 | 5.0 | 2.0 | 49.1 | -100.0 | 59.798472 | 30.331733 | 4.0 | 2.0 | 196140.0 | 349079.0 | 78.0 | 1427199.0 |
| **39517** | 2021-01-05 | 6299000 | 3.0 | 8.0 | -1.0 | 31.0 | 0.0 | 59.984093 | 30.249716 | 0.0 | 0.0 | 197374.0 | 168232.0 | 78.0 | 1628642.0 |
| **39561** | 2021-01-05 | 8500000 | 7.0 | 16.0 | 2.0 | 80.0 | 15.5 | 59.968545 | 30.428010 | 4.0 | 0.0 | 195253.0 | 505784.0 | 78.0 | 2456992.0 |
| **39583** | 2021-01-05 | 7200000 | 6.0 | 17.0 | 3.0 | 62.2 | 8.5 | 59.860670 | 30.497967 | 0.0 | 0.0 | 193149.0 | 265751.0 | 78.0 | 766313.0 |

2393 rows × 15 columns

*Рисунок 2 - Данные из выборки по Санкт-Петербургу*

Целевым признаком было принято решение выбрать стоимость жилья – признак *price* в выборке.

Для анализа выбросов была построена тепловая карта по выборке. Она представлена на рисунках 3 и 4.

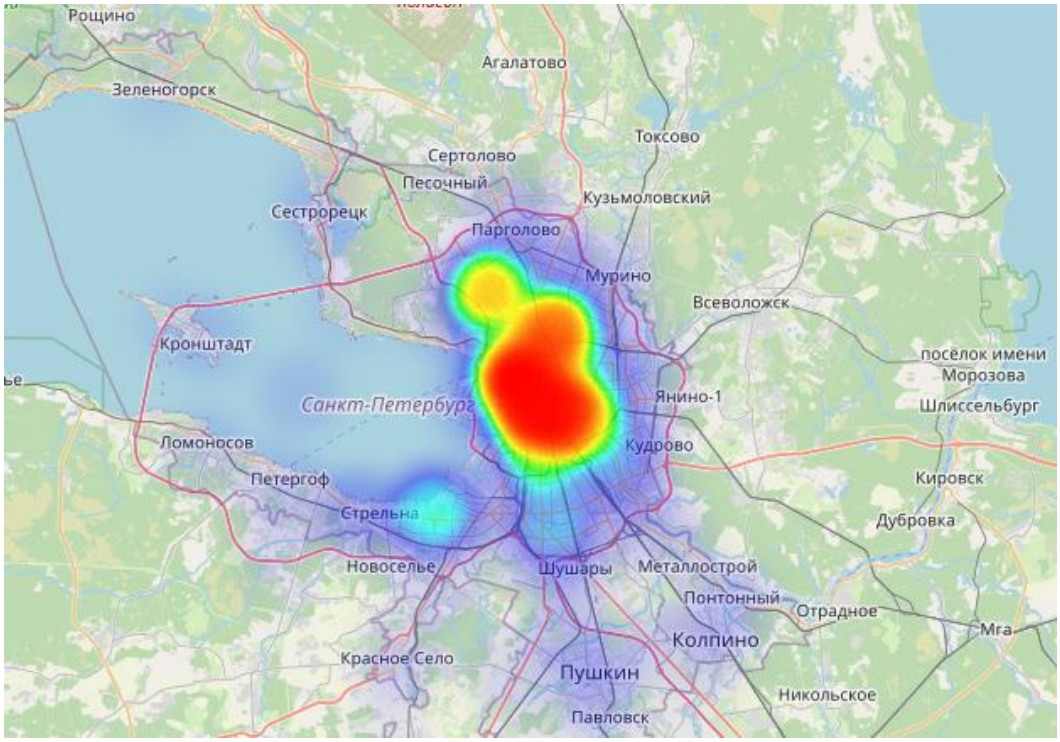


*Рисунок 3 - Тепловая карта до очистки*

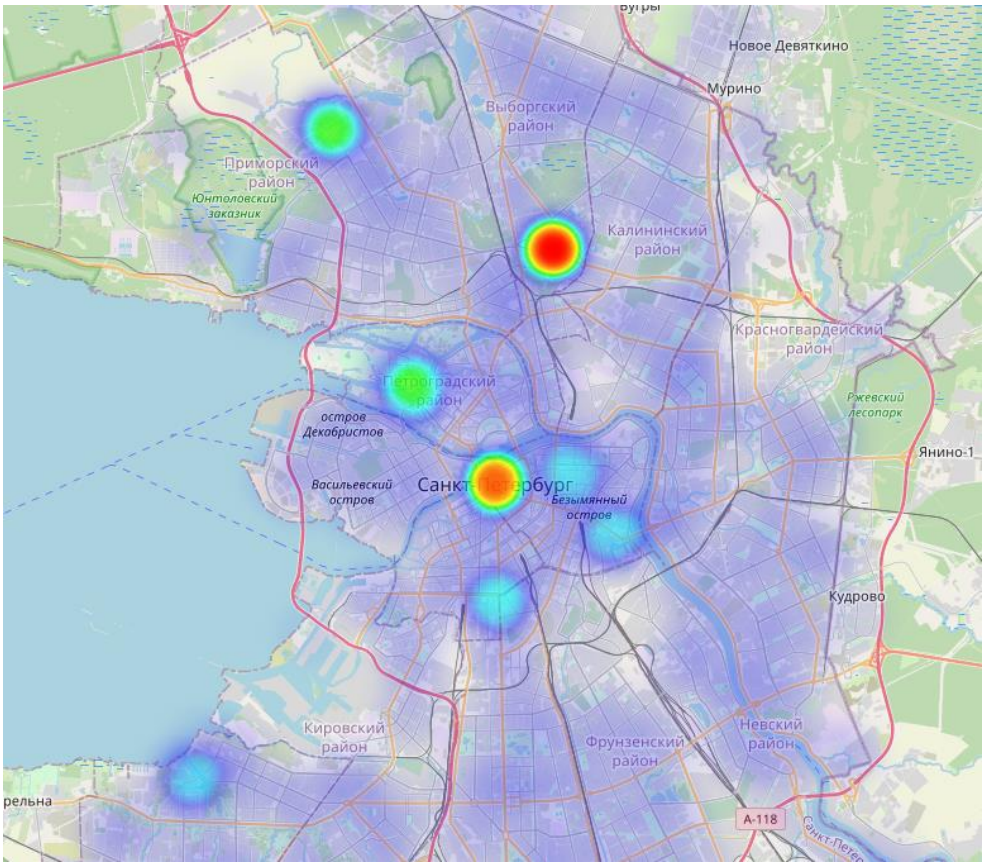


*Рисунок 4 - Тепловая карта до очистки. Вид ближе*

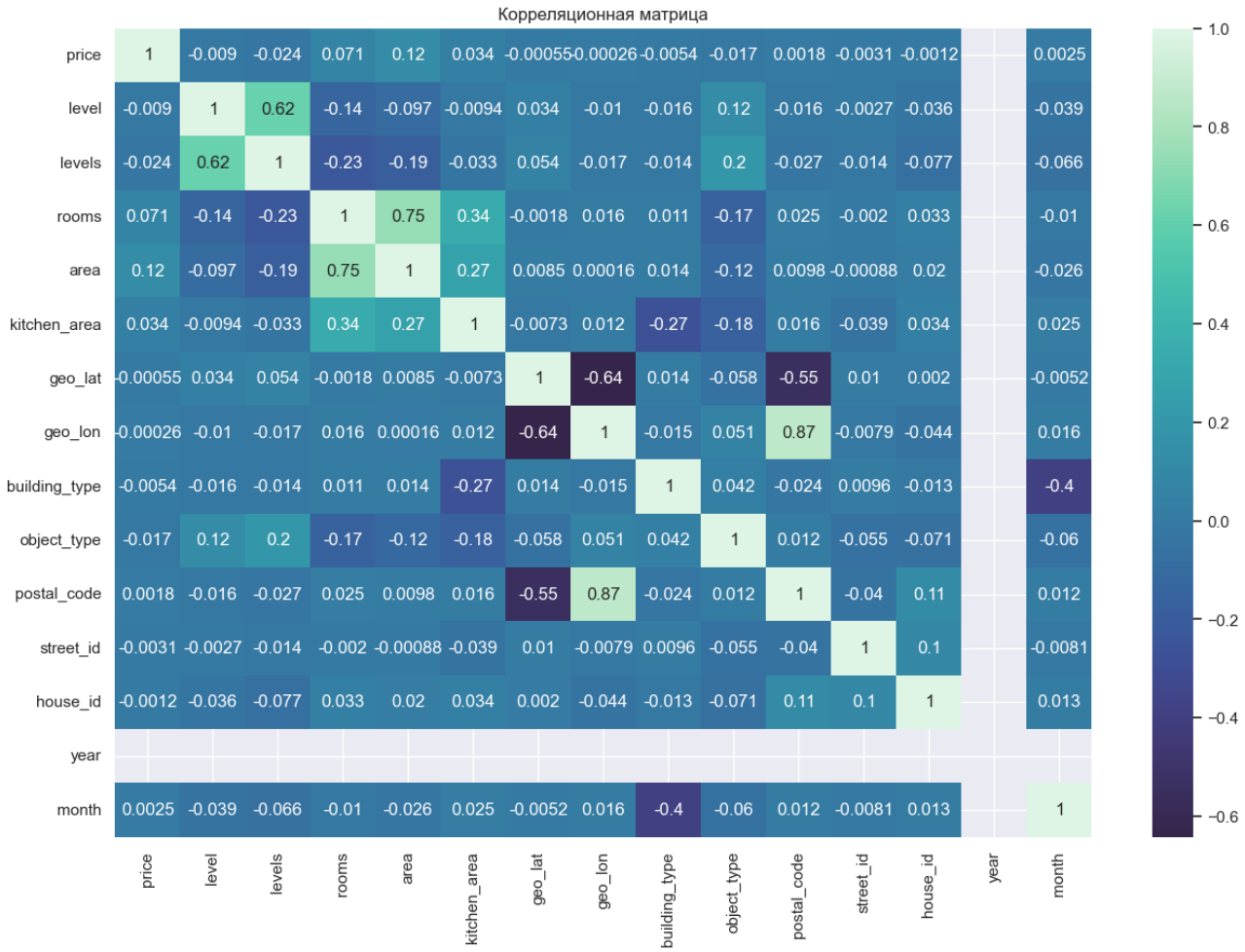


*Рисунок 5 - Корреляционная матрица до очистки*

Далее были построены гистограммы распределения и диаграммы рассеяния, на которых можно также наблюдать выбросы. Графики представлены на рисунках 6 и 7.

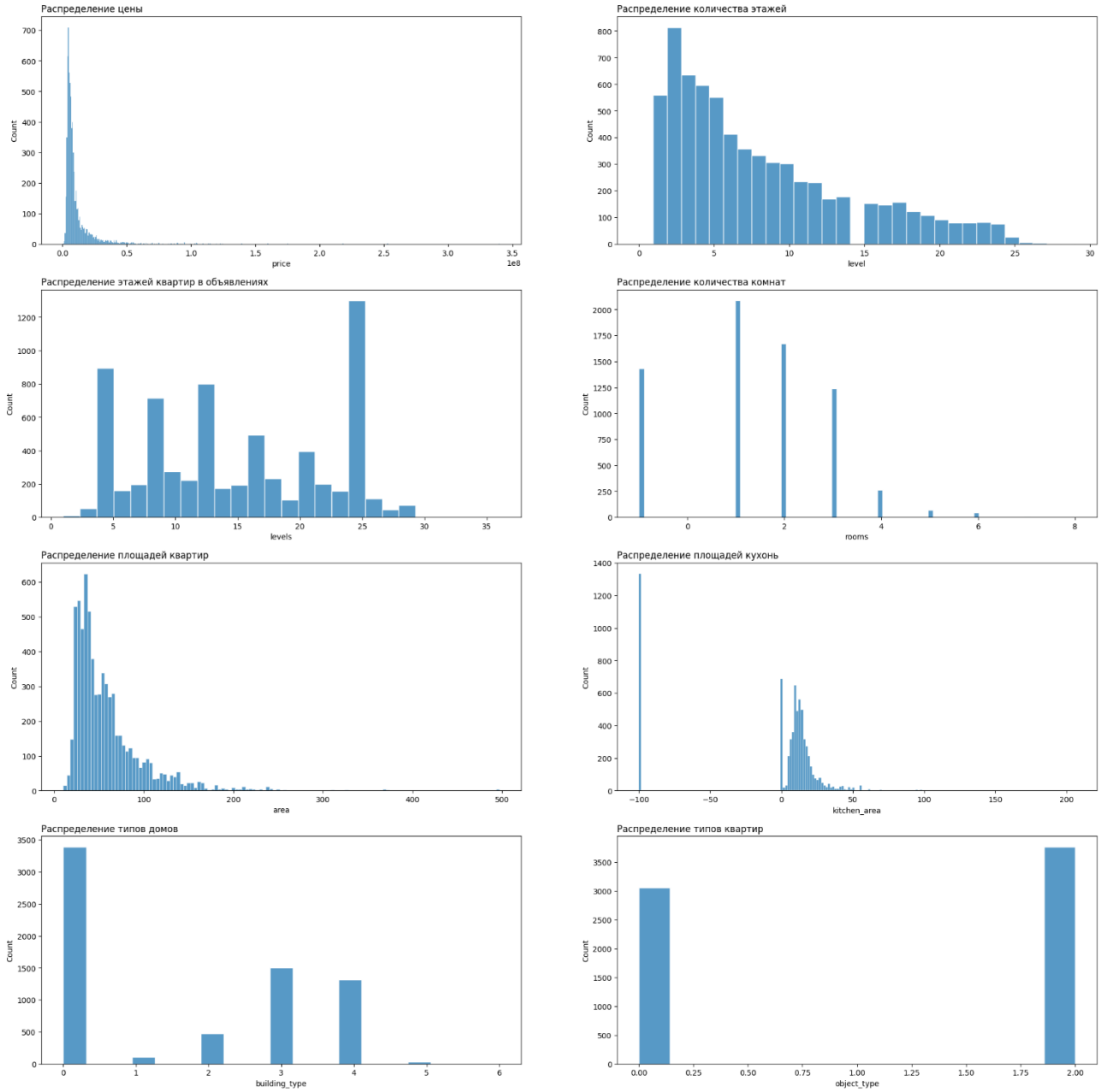


*Рисунок 6 - Гистограммы распределения до очистки*

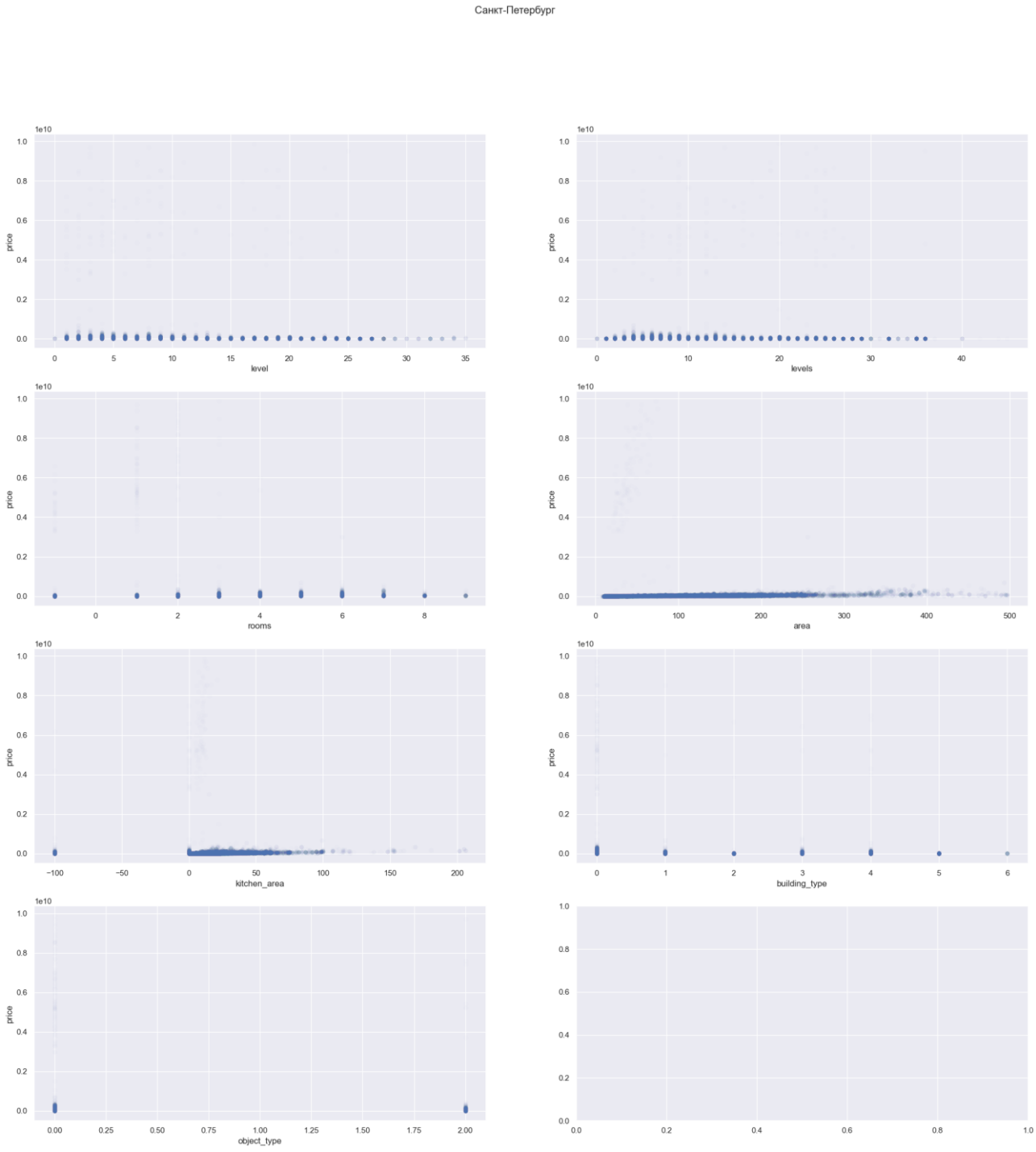


*Рисунок 7 - Диаграммы рассеяния до очистки данных*

На рисунке ниже представлена информация о данных по выборке по Санкт-Петербургу. Представлены следующие признаки:

- price – стоимость в рублях
- level – этаж квартиры
- levels – количество этажей в доме
- rooms – количество комнат в квартире (скольки-комнатная квартира)

– area – общая площадь квартиры
– kitchen_area – площадь кухни
– geo_lat – географическая долгота
– geo_lon – географическая широта
– building_type – тип (неизвестен, другой, панельный, монолит, кирпичный, блочный, деревянный). Категориальный признак представлен цифрами от 0 до 6
– object_type – тип квартиры (вторичный рынок, новостройка). Категориальный признак представлен цифрами от 1 до 2
– postal_code – почтовый индекс дома
– street_id – идентификатор улицы города
– house_id – идентификатор дома в городе
– year и month – год и месяц публикации объявления. Колонки созданы искусственно из начальной колонки date, поскольку день публикации практически не коррелирует со стоимостью жилья.

```
saint_petersburg_df.info(verbose = True, show_counts = True)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2393 entries, 15 to 39583
Data columns (total 15 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          2393 non-null   int64
 1   level          2393 non-null   float64
 2   levels         2393 non-null   float64
 3   rooms          2393 non-null   float64
 4   area           2393 non-null   float64
 5   kitchen_area   2393 non-null   float64
 6   geo_lat        2393 non-null   float64
 7   geo_lon        2393 non-null   float64
 8   building_type  2393 non-null   float64
 9   object_type    2393 non-null   float64
 10  postal_code    2228 non-null   float64
 11  street_id      2117 non-null   float64
 12  house_id       2328 non-null   float64
 13  year           2393 non-null   int32
 14  month          2393 non-null   int32
dtypes: float64(12), int32(2), int64(1)
memory usage: 345.0 KB
```

*Рисунок 8 - Информация о данных*

Данные были проверены на наличие пропусков. Пропуски были найдены в колонках postal_code, street_id, house_id, как видно на рисунке 9. Перечисленные признаки практически не имеют корреляции со стоимостью недвижимости, поэтому эти колонки были удалены.

```
saint_petersburg_df.replace([np.inf, -np.inf], np.nan, inplace=True)
saint_petersburg_df.isna().sum()
```

```
price             0
level             0
levels            0
rooms             0
area              0
kitchen_area      0
geo_lat           0
geo_lon           0
building_type     0
object_type       0
postal_code     120
street_id       203
house_id         50
year              0
month             0
dtype: int64
```

*Рисунок 9 - Пропущенные значения*

Далее была проверка на наличие дублированных записей. Найденные дубликаты были удалены.

```
saint_petersburg_df = saint_petersburg_df.drop_duplicates()
saint_petersburg_df.duplicated().sum()
```

```
0
```

*Рисунок 10 - Удаление дубликатов*

Далее были обработаны аномальные отрицательные значения в колонке kitchen_area и rooms. Поскольку отрицательные значения в данном случае рассматриваются как аномалия, они были заменены на противоположные по знаку положительные значения.

Затем выбросы в данных были отрезаны по стандартному отклонению (z-index): для широты, долготы, площади и количества комнат использовалось пороговое значение 1.5, для целевого признака стоимости – пороговое значение 0.15 в связи с достаточно большим разбросом значений.

По результатам обработки выбросов, пропусков и дубликатов, были заново созданы тепловая карта, диаграммы рассеяния и гистограммы распределения, на которых видно, что выбросы и аномалии устранены.

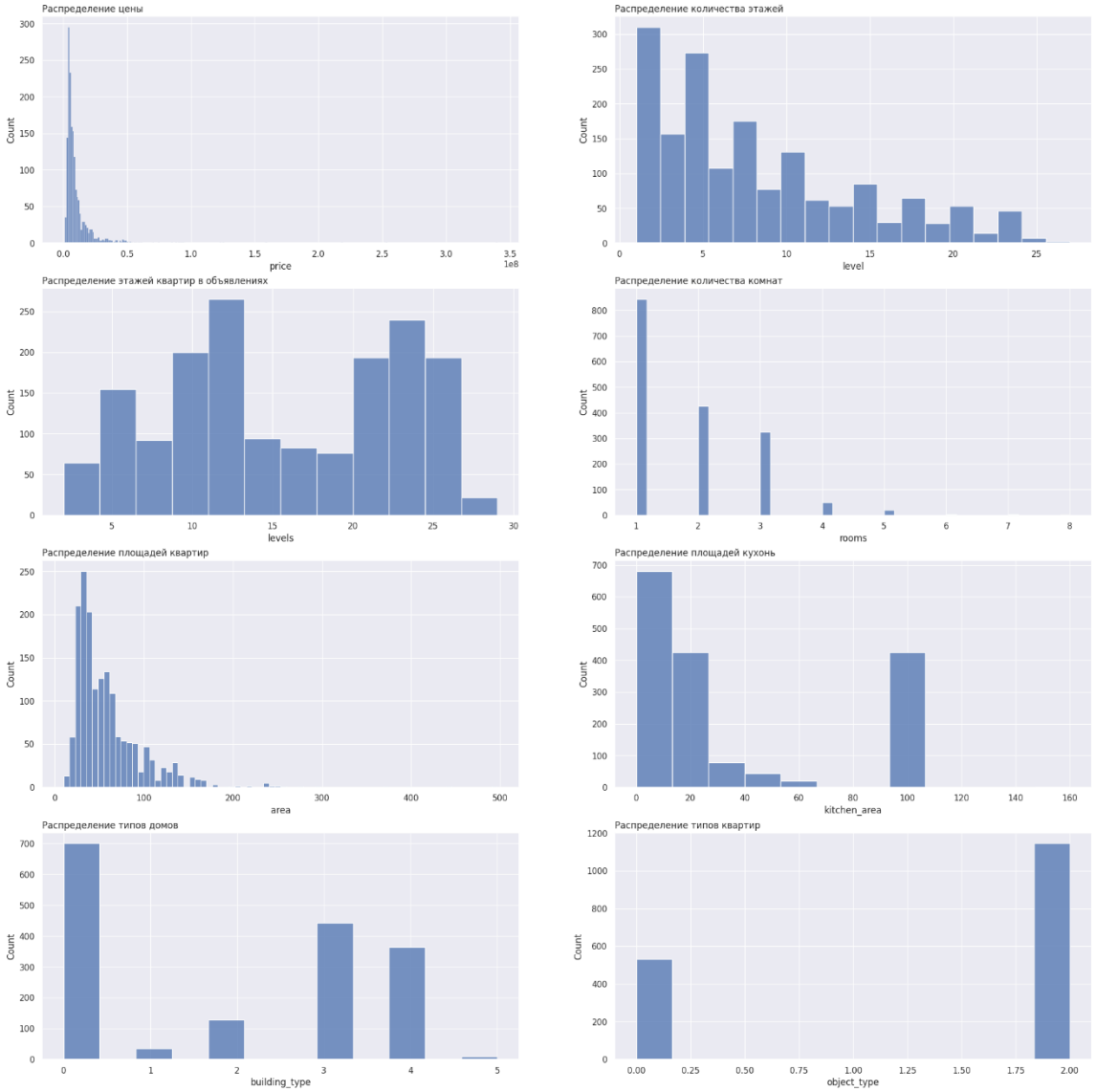


*Рисунок 11 - Гистограммы распределения после очистки*

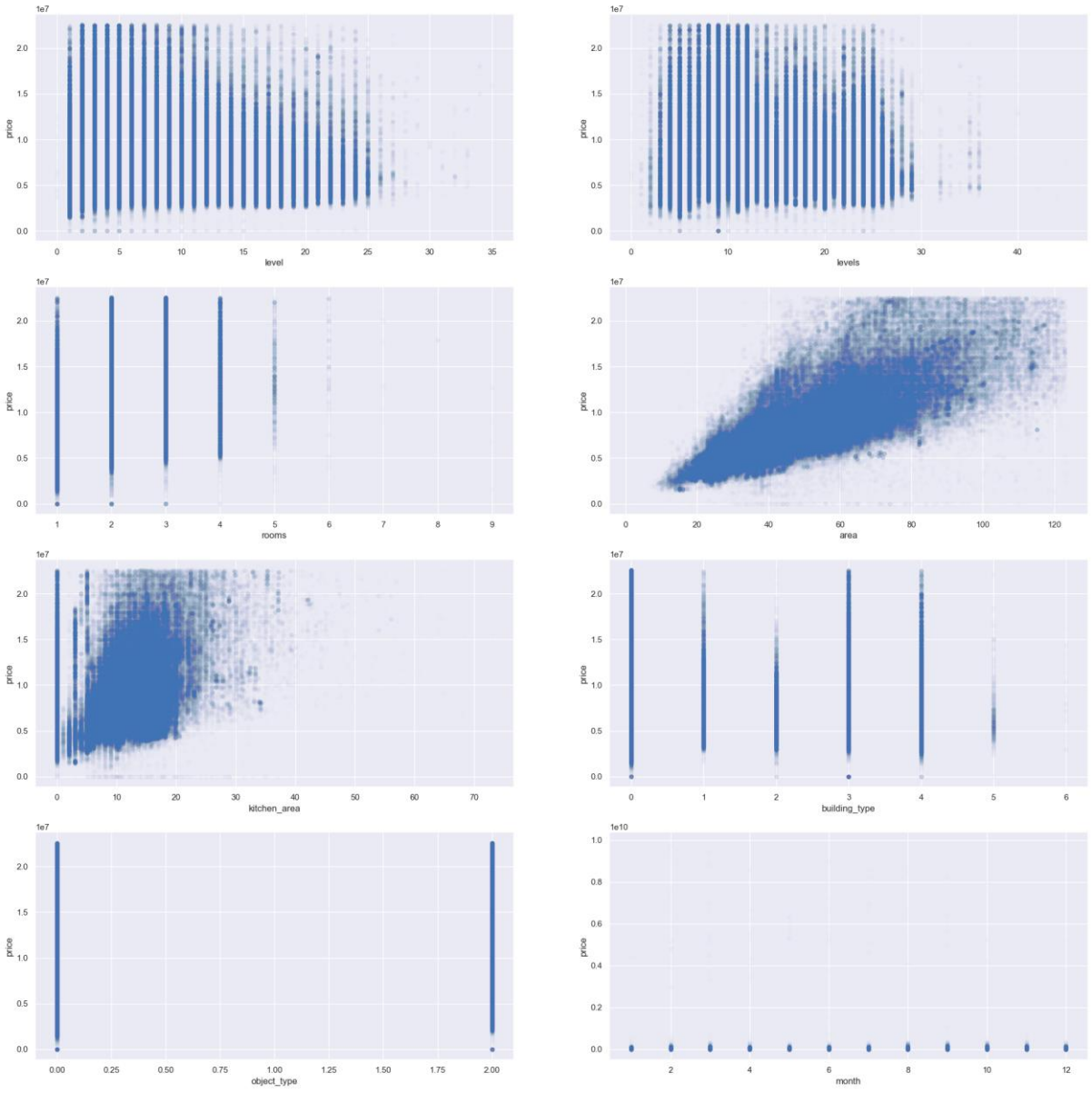


*Рисунок 12 - Гистограммы рассеяния после очистки. Увеличенный масштаб*

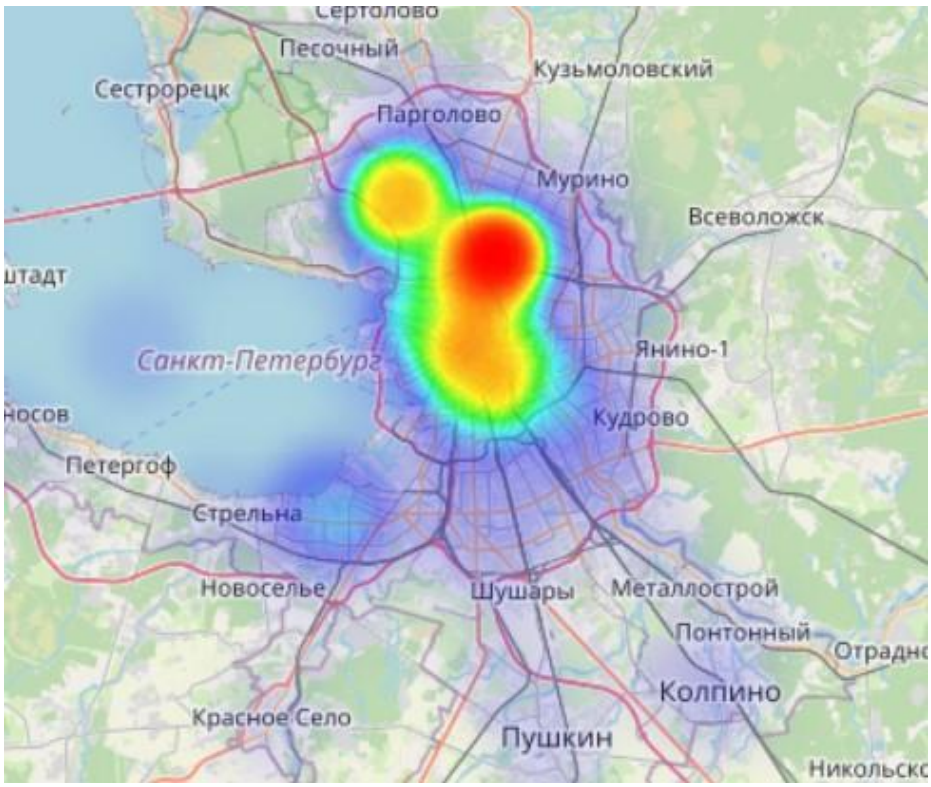


*Рисунок 13 - Тепловая карта после очистки*

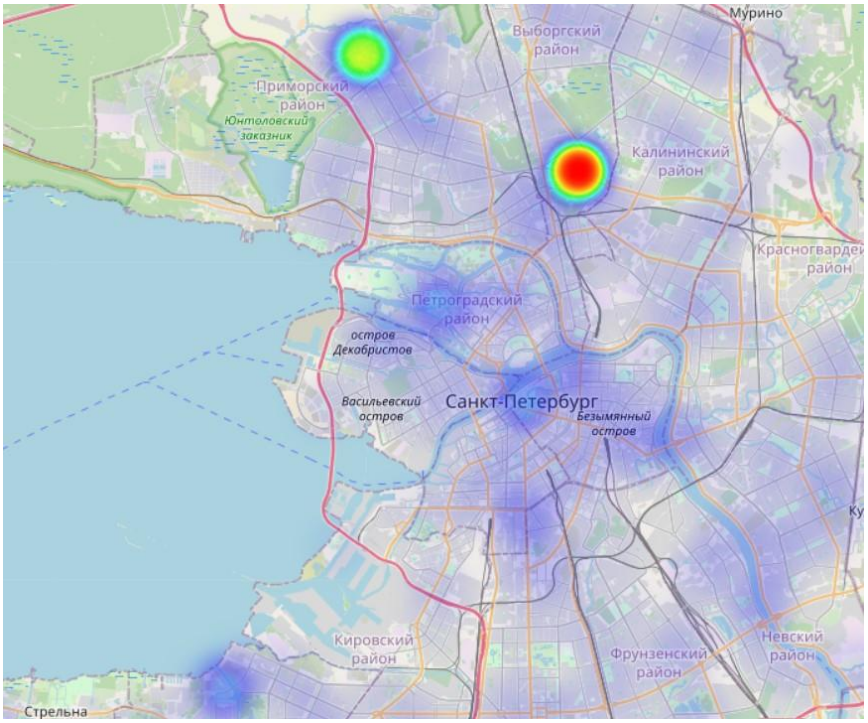


*Рисунок 14 - Тепловая карта после очистки. Вид ближе*

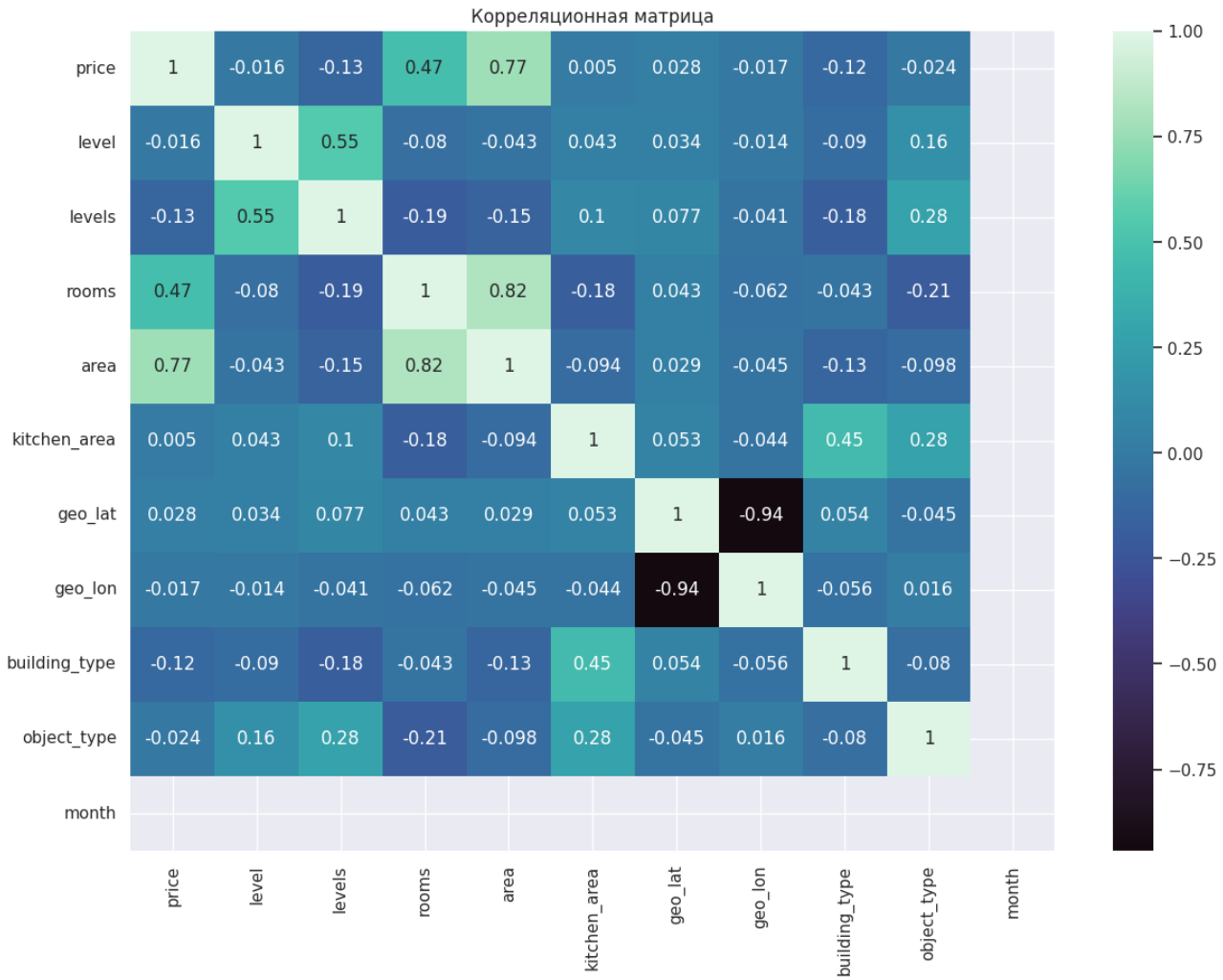


*Рисунок 15- Корреляционная матрица после очистки*

По обновленной корреляционной матрице можно сделать вывод на наибольшей корреляции между стоимостью и площадью и количеством комнат объекта недвижимости. Поэтому эти признаки были выбраны для обучения модели.

Перед обучением, данные были масштабированы по стандартному методу и разделены на тренировочную и валидационные выборку.

Точность модели на тренировочных данных составила приблизительно 0.65, в то время как на валидационной выборке – 0.74.

```
Accuracy of Linear Regression on training data 0.649042251172697
Accuracy of Linear Regression on testing data 0.739686089235194
```

*Рисунок 16 – Точность*

Оценки метрик получились следующими: $R^2 = 0.74;\ \ MAE = 2766111.$

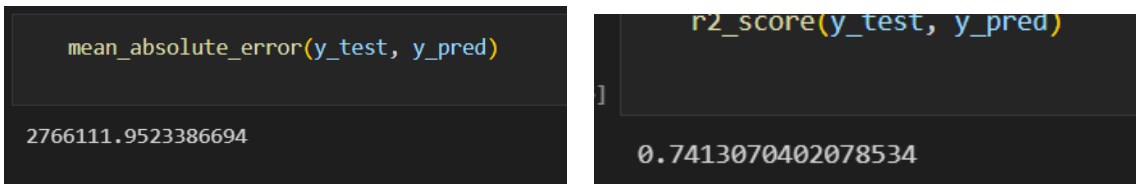


*Рисунок 17 – Метрики*

# ВЫВОДЫ

В ходе выполнения лабораторной работы был проведен анализ выборки о недвижимости в Санкт-Петербурге. Данные были обработаны и подготовлены для обучения построенной модели множественной линейной регрессии для прогнозирования стоимости жилья в данном регионе.

Результаты работы позволили сделать вывод о том, что модель делает достаточно точные предсказания.

Коэффициент детерминации $R^2$ равный 0.74 указывает на то, что 74% дисперсии целевой переменной (стоимости объектов недвижимости) объяснены с помощью используемых признаков модели. Это может признать хорошим результатом, поскольку значение близкое к 1 свидетельствует о хорошем соответствии модели выбранным данным.

Значение средней абсолютной ошибки MAE, равное 2766111 можно интерпретировать, опираясь на среднее значение предсказываемых стоимостей на объекты недвижимости. Поскольку средняя стоимость составляет 10800622, то среднюю абсолютную ошибку со значением порядка двух-трех миллионов можно считать приемлемой.

Таким образом, основываясь на предыдущих рассуждениях и значениях рассмотренных метрик, делаем вывод о том, что модель демонстрирует хорошую способность предсказывать.