

Анализ рынка заведений общественного питания Москвы

Содержание:

- Выгрузка и знакомство с данными
 - Предобработка и анализ данных по колонкам
 - Вывод после предварительного анализа общей выборки по колонкам
- Анализ взаимосвязей и особенностей заведений общественного питания Москвы:
 - Анализ количества заведений по категориям
 - Анализ среднего значения количества посадочных мест по категориям
 - Анализ соотношения сетевых и несетевых заведений
 - Топ 15 заведений по количеству точек
 - Анализ категорий заведений по Административным округам
 - Анализ рейтингов заведений
 - Посчитаем расстояния от центра до каждого объекта
 - Отобразим все заведения на карте Москвы
 - Найдем топ 15 улиц по количеству заведений
 - Посмотрим на самые непопулярные улицы
 - Проведем анализ среднего чека по округам и в зависимости от удаленности из центра
 - Анализ среднего чека по категориям заведений
 - Анализ доли круглосуточных заведений по округам и по категориям
 - Анализ взаимосвязи количества посадочных мест заведений с наличием торговых центров на расстоянии 400 метров
 - Вывод по анализу заведений общественного питания
- Анализ кофеен
 - Посмотрим на распределение кофеен по округам
 - Разберем топ 15 самых популярных улиц для расположения кофеен
 - Посмотрим на количество самых популярных кофеен
 - Проведем анализ доли круглосуточных кофеен
 - Посмотрим на распределение рейтингов кофеен
 - Проведем анализ распределения среднего чека и средней стоимости чашки капучино
 - Анализируем зависимость удаленности из центра и стоимость чашки капучино
 - Проведем анализ распределения количества посадочных мест кофеен
 - Проведем анализ влияния наличия торговых центров в радиусе 400 метров на количество посадочных мест в кофейнях
 - Вывод по анализу кофеен
- Рекомендации по выбору места для открытия заведений общественного питания
- Рекомендации по выбору помещения для концептуальной идеи кофейни "Central Perk"

```
In [1]: # импортируем библиотеки
import missingno as msno
import pandas as pd
import numpy as np
import math
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from plotly import graph_objects as go
import folium
import json
from folium import Map, Choropleth
```

[к содержанию](#)

Выгрузка и знакомство с данными

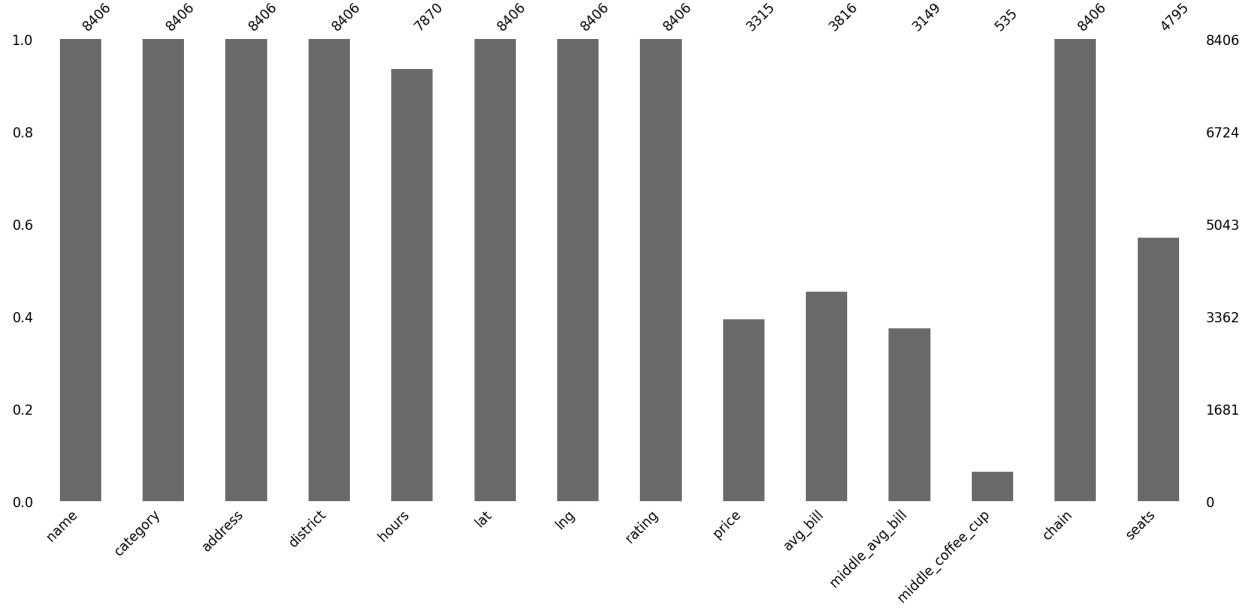
In [3]: `# посмотрим на первые строки
display(data.head())`

	name	category	address	district	hours	lat	lng	rating	price	ε
0	WoWфли	кафе	Москва, улица Дыбенко, 7/1	Северный административный округ	ежедневно, 10:00–22:00	55.878494	37.478860	5.0	NaN	
1	Четыре комнаты	ресторан	Москва, улица Дыбенко, 36, корп. 1	Северный административный округ	ежедневно, 10:00–22:00	55.875801	37.484479	4.5	выше среднего	Ср. счёт
2	Хазри	кафе	Москва, Клязьминская улица, 15	Северный административный округ	пн-чт 11:00–02:00; пт,сб 11:00–05:00; вс 11:00...	55.889146	37.525901	4.6	средние	Ср. счёт:от
3	Dormouse Coffee Shop	кофейня	Москва, улица Маршала Федоренко, 12	Северный административный округ	ежедневно, 09:00–22:00	55.881608	37.488860	5.0	NaN	Цена капучин
4	Илья Марко	пиццерия	Москва, Правобережная улица, 1Б	Северный административный округ	ежедневно, 10:00–22:00	55.881166	37.449357	5.0	средние	Ср. счёт:40

In [4]: `# отобразим основную информацию
display(data.info())`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8406 entries, 0 to 8405
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
 --- 
 0   name             8406 non-null   object  
 1   category         8406 non-null   object  
 2   address          8406 non-null   object  
 3   district         8406 non-null   object  
 4   hours            7870 non-null   object  
 5   lat              8406 non-null   float64 
 6   lng              8406 non-null   float64 
 7   rating           8406 non-null   float64 
 8   price            3315 non-null   object  
 9   avg_bill         3816 non-null   object  
 10  middle_avg_bill 3149 non-null   float64 
 11  middle_coffee_cup 535 non-null   float64 
 12  chain            8406 non-null   int64  
 13  seats            4795 non-null   float64 
dtypes: float64(6), int64(1), object(7)
memory usage: 919.5+ KB
None
```

In [5]: `# построим столбчатую диаграмму количества пропусков
msno.bar(data)
plt.show()`



In [6]: # посчитаем количество полных дубликатов

```
print(f'Количество полных дубликотов в данных равно {data.duplicated().sum()}')
print('*****')
print('Количество дубликотов по имени и координатах в данных равно:', data[['name', 'lat', 'lng']]
```

Количество полных дубликотов в данных равно 0

Количество дубликотов по имени и координатах в данных равно: 0

- В данных присутствует большое количество пропусков в колонках, содержащих финансовую информацию и количество посадочных мест. Пропуски распределены относительно равномерно по данным. Некоторые колонки необходимо привести к правильному типу данных. Дубликаты отсутствуют

к содержанию

Предобработка и анализ данных

разберем колонки с названиями заведений

In [7]: # # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
display(data['name'].unique()[:50])
посчитаем количество уникальных заведений
print(f'количество уникальных заведений в городе Москва равно:', len(data['name'].unique()))

```
array(['WoWфли', 'Четыре комнаты', 'Хазри', 'Dormouse Coffee Shop',
       'Иль Марко', 'Sergio Pizza', 'Огни города', 'Мг. Уголёк',
       'Donna Maria', 'Готика', 'Great Room Bar', 'Шашлык Шефф',
       'Заправка', 'Буханка', 'У Сильвы', 'Дом обеда', 'База Стритфуд',
       'Чайхана Беш-Бармак', 'Час-Пик', 'Пекарня', 'Чебуреки Манты',
       '7/12', 'Крымские чебуреки', 'Drive Café', 'В парке вкуснее',
       'Пикочино', 'Шаурму X@чу', 'Mafe', 'Кушай Город', 'Кафедра',
       'Алталия', 'Додо Пицца', 'Изба', 'Домино'с Пицца', 'Виладж пицца',
       'Пекарня № 1 Грузинская кухня', 'Халяль закусочная', 'Ижора',
       'Шаурмагия', 'Кафе', 'Крошка Картошка', 'Варня кафе',
       'Суши & пицца', 'Кафетерий', '9 Bar Coffee', 'CofeFest',
       'БлинБери', '2U-Ty-10', 'Шаурма', 'Арамье'], dtype=object)
```

количество уникальных заведений в городе Москва равно: 5614

In [8]: # отобрази топ по количеству заведений
plt.style.use('seaborn-whitegrid')
data.groupby(by='name')['name'].count().sort_values(ascending=False).head(20).plot(kind='bar',
figsize=(15, 5),
alpha=0.6,
color='g')
plt.title('количество заведений в Москве')
plt.ylabel('количество')

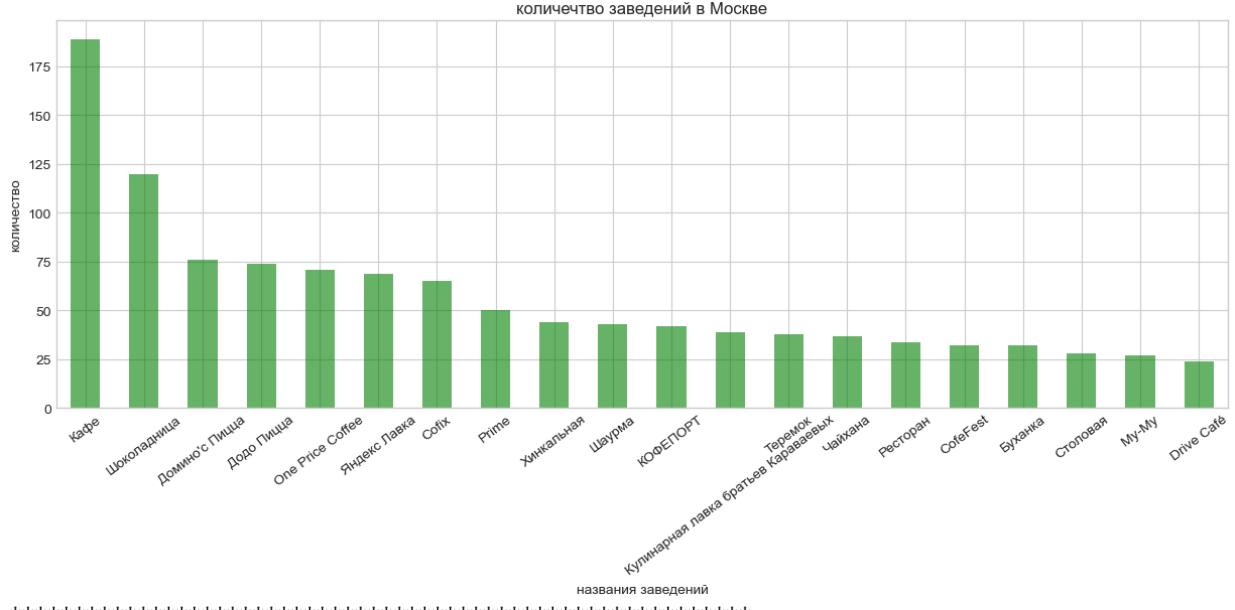
```

plt.xlabel('название заведений')
plt.xticks(rotation=37)
plt.show()
print('*****')
display(
    data.groupby(by='name')[['name']].count().sort_values(
        ascending=False).head(20))

```

C:\Users\coder\AppData\Local\Temp\ipykernel_4180\3802653872.py:2: MatplotlibDeprecationWarning:
The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond
to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0_8-<style>'.
Alternatively, directly use the seaborn API instead.

```
plt.style.use('seaborn-whitegrid')
```



```

*****
name
Кафе                               189
Шоколадница                         120
Домино'с Пицца                      76
Додо Пицца                           74
One Price Coffee                     71
Яндекс Лавка                          69
Cofix                                65
Prime                                 50
Хинкальная                          44
Шаурма                                43
КОФЕПОРТ                            42
Кулинарная лавка братьев Караваевых 39
Теремок                                38
Чайхана                                37
Ресторан                                34
CofeFest                                32
Буханка                                32
Столовая                                28
My-My                                    27
Drive Café                             24
Name: name, dtype: int64

```

- В топе 'Шоколадница', 'Домино'с Пицца' и 'Додо Пицца' с 120, 76 и 74 точками соответственно, что может говорить о наличии многих сетевых заведений в городе. Названия - 'Кафе', 'Ресторан' и 'Шаурма' отображают количество несетевых заведений и показывают наличие не малого их количества в сравнении с крупными сетями

Проработаем столбец category с категориями заведений

```

In [9]: # # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
data[['category']].unique()

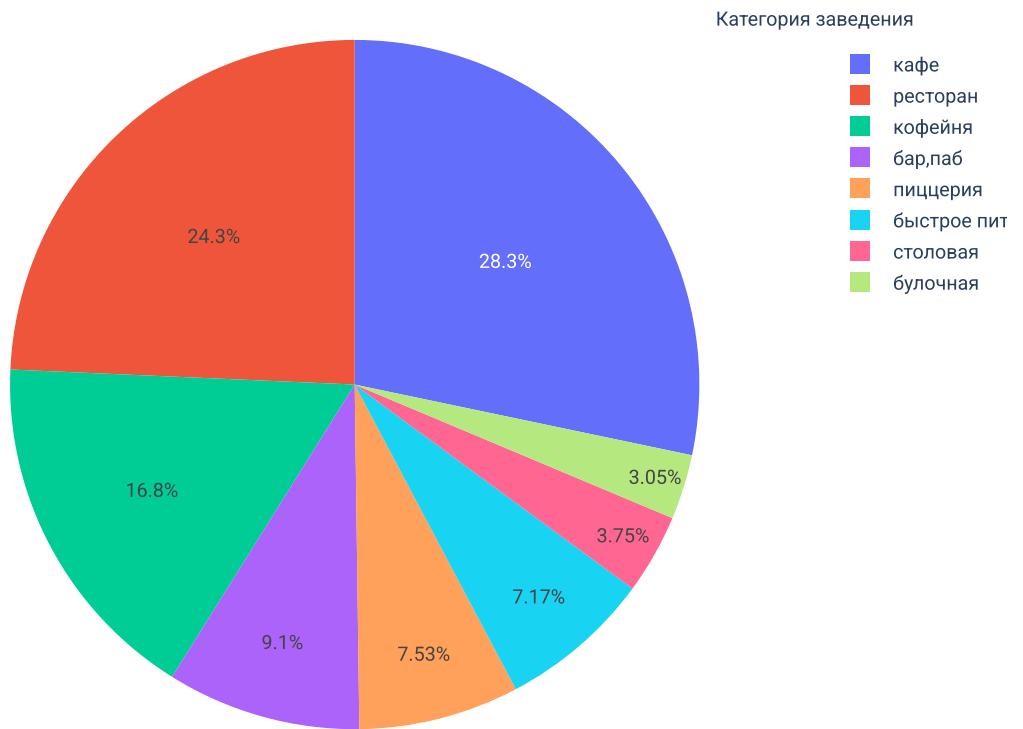
Out[9]: array(['кафе', 'ресторан', 'кофейня', 'пиццерия', 'бар,паб',
       'быстрое питание', 'булочная', 'столовая'], dtype=object)

```

```
In [10]: data['category'] = data['category'].astype('category')
```

```
In [11]: # готовим данные для графика
category = pd.DataFrame(data['category'].value_counts()).reset_index()
# строим диаграмму с сегментами
fig = go.Figure(data=[go.Pie(labels=category['index'], # указываем значения, которые появятся на
                                values=category['category'], # указываем данные, которые отобразятся
                                )]) # добавляем аргумент, который выделит сегмент-лидер на графике
fig.update_layout(title='Число заведений в зависимости от их категории', # указываем заголовок
                  width=800, # указываем размеры графика
                  height=600,
                  annotations=[dict(x=1.12, # вручную настраиваем аннотацию легенды
                                    y=1.05,
                                    text='Категория заведения',
                                    showarrow=False)])
fig.show() # выводим график
```

Число заведений в зависимости от их категории



- лидируют заведения в категориях кафе, ресторан и кофейня и имеют кратное превосходство в количестве точек над заведениями в остальных категориях. Столовые и булочные менее популярны и имеют 315 и 256 заведений соответственно

Разберем адреса в колонке address и выделим улицу из названия

```
In [12]: # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
data['address'].unique()[:50]
```

```
Out[12]: array(['Москва, улица Дыбенко, 7/1', 'Москва, улица Дыбенко, 36, корп. 1',
 'Москва, Клязьминская улица, 15',
 'Москва, улица Маршала Федоренко, 12',
 'Москва, Правобережная улица, 1Б', 'Москва, Ижорская улица, вл8Б',
 'Москва, Клязьминская улица, 9, стр. 3',
 'Москва, Дмитровское шоссе, 107, корп. 4',
 'Москва, Ангарская улица, 39', 'Москва, Левобережная улица, 12',
 'Москва, улица Маршала Федоренко, 10с1',
 'Москва, МКАД, 80-й километр, 1',
 'Москва, Базовская улица, 15, корп. 1',
 'Москва, Ангарская улица, 42с1',
 'Москва, улица Бусиновская Горка, 2',
 'Москва, Базовская улица, 15, корп. 8',
 'Москва, Ленинградское шоссе, 71Б, стр. 2',
 'Москва, Коровинское шоссе, 30А', 'Москва, Ижорский проезд, 5',
 'Москва, Прибрежный проезд, 7',
 'Москва, Коровинское шоссе, 35, стр. 17',
 'Москва, Лобненская улица, 13к2', 'Москва, улица Дыбенко, 9Ас1',
 'Москва, парк Левобережный', 'Москва, Дмитровское шоссе, 107к2',
 'Москва, улица Дыбенко, 7, стр. 1',
 'Москва, МКАД, 78-й километр, 14к1',
 'Москва, Дмитровское шоссе, 157, стр. 15',
 'Москва, Базовская улица, 15, корп. 6',
 'Москва, Коровинское шоссе, 23, корп. 1',
 'Москва, Лобненская улица, 4А',
 'Москва, Клязьминская улица, 11, корп. 4',
 'Москва, Базовская улица, 15, корп. 15',
 'Москва, Коровинское шоссе, 33А',
 'Москва, Коровинское шоссе, 46, стр. 5',
 'Москва, Ижорский проезд, 5А', 'Москва, Базовская улица, 15А',
 'Москва, Ижорская улица, 18, стр. 1',
 'Москва, Коровинское шоссе, 29, корп. 1',
 'Москва, Дмитровское шоссе, 169, корп. 6',
 'Москва, Ангарская улица, 24А',
 'Москва, Коровинское шоссе, 41, стр. 1',
 'Москва, улица Маршала Федоренко, 6с1',
 'Москва, улица Маршала Федоренко, 7', 'Москва, Ижорская улица, 8А',
 'Москва, улица Дыбенко, 44', 'Москва, Ангарская улица, 30/25',
 'Москва, Дмитровское шоссе, 151, корп. 5',
 'Москва, Угличская улица, 13, стр. 8',
 'Москва, Абрамцевская улица, 30, стр. 1'], dtype=object)
```

```
In [13]: len(data['address'].unique())
```

```
Out[13]: 5753
```

```
In [14]: # напишем функцию
def od(s):
    """This function take string with address and return name of street"""
    try:
        st = s.split(',')[1]
        st = ''.join(st.split(' улица'))
        st = ''.join(st.split('улица '))
    except:
        'err'
    print(od('Москва, Дмитровское шоссе, 151, корп. 5'))
```

```
Дмитровское шоссе
```

```
In [15]: # применем
data['street'] = data['address'].apply(od)
```

```
In [16]: print(f"Количество уникальных названий улиц равно: {len(data['street'].unique())}")
```

```
Количество уникальных названий улиц равно: 1448
```

```
In [17]: # посмотрим на неявные дубликаты
data['street'].unique()[:50]
```

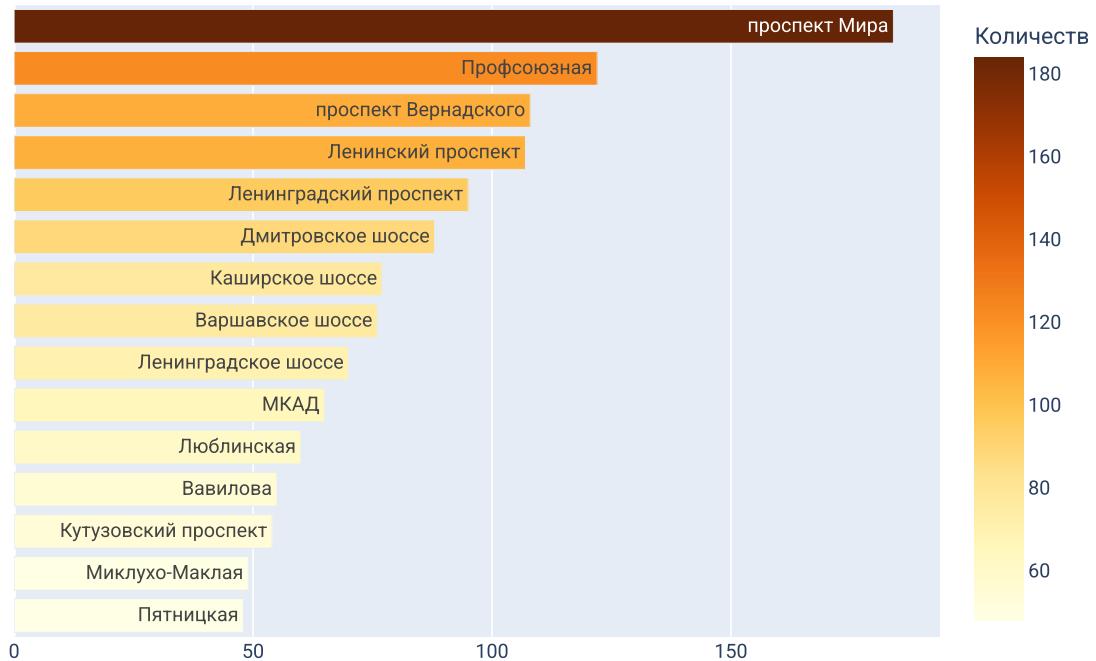
```
Out[17]: array(['Дыбенко', 'Клязьминская', 'Маршала Федоренко', 'Правобережная',
   'Ижорская', 'Дмитровское шоссе', 'Ангарская', 'Левобережная',
   'МКАД', 'Базовская', 'Бусиновская Горка', 'Ленинградское шоссе',
   'Коровинское шоссе', 'Ижорский проезд', 'Прибрежный проезд',
   'Лобненская', 'парк Левобережный', 'Угличская', 'Абрамцевская',
   'ландшафтный заказник Лианозовский', 'Череповецкая',
   'Алтуфьевское шоссе', 'Лианозовский парк культуры и отдыха',
   'Дубнинская', 'Илимская', 'Шенкурский проезд', 'Новгородская',
   'парк Алтуфьево', 'Пришвина', 'бульвар Академика Ландау',
   'Лескова', 'Плещеева', 'Бибиревская', 'Лианозовский проезд',
   'Челобитьевское шоссе', 'Костромская', 'Карельский бульвар',
   'Инженерная', 'парк Ангарские пруды', 'Софья Ковалевской',
   'Корнейчука', 'Проектируемый проезд № 5265', 'Римского-Корсакова',
   '800-летия Москвы', 'Юрловский проезд', 'парк Ангарские Пруды',
   'Вагоноремонтная', 'Мурановская', 'Конёнкова', 'Широкая'],
  dtype=object)
```

```
In [18]: # посчитаем топ
top_street = data.groupby('street')[['street']].count()
top_street.columns = ['point_count']
# отсортируем и оставим пять лидеров
top_street = top_street.reset_index().sort_values(by='point_count', ascending=False).head(15)
top_street.columns = ['Улица', 'Количество заведений']
display(top_street)
```

Улица Количество заведений		
1440	проспект Мира	184
1038	Профсоюзная	122
1437	проспект Вернадского	108
696	Ленинский проспект	107
693	Ленинградский проспект	95
479	Дмитровское шоссе	88
589	Каширское шоссе	77
358	Варшавское шоссе	76
694	Ленинградское шоссе	70
731	МКАД	65
725	Люблинская	60
353	Вавилова	55
683	Кутузовский проспект	54
806	Миклухо-Маклая	49
1048	Пятницкая	48

```
# строим столбчатую диаграмму
fig = px.bar(top_street.sort_values(by='Количество заведений', ascending=True),
              x='Количество заведений', # указываем столбец с данными для оси X
              y='Улица', # указываем столбец с данными для оси Y
              text='Улица', color='Количество заведений', color_continuous_scale='ylorbr' # добавляем цвет
            )
# оформляем график
fig.update_layout(title='ТОП-15 улиц по количеству заведений',
                  xaxis_title='Количество заведений',
                  yaxis_title='Улица'
                 )
fig.update_yaxes(title='y', visible=False, showticklabels=False)
fig.show() # выводим график
```

ТОП-15 улиц по количеству заведений



- В топе по количеству заведений лидируют шоссе и проспекты, количество уникальных названий улиц равно: 1448

Разберем районы в столбце district

```
In [20]: # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
data.district.unique()
```

```
Out[20]: array(['Северный административный округ',
 'Северо-Восточный административный округ',
 'Северо-Западный административный округ',
 'Западный административный округ',
 'Центральный административный округ',
 'Восточный административный округ',
 'Юго-Восточный административный округ',
 'Южный административный округ',
 'Юго-Западный административный округ'], dtype=object)
```

```
In [21]: # приведем к верному типу
data.district = data.district.astype('category')
```

```
In [22]: print('Общее количество заведений в Москве равно:', len(data))
```

Общее количество заведений в Москве равно: 8406

```
In [23]: # готовим данные для графика
district = pd.DataFrame(data['district'].value_counts()).reset_index()
# строим диаграмму с сегментами
fig = go.Figure(data=[go.Pie(labels=district['index'], # указываем значения, которые появятся на
                                values=district['district'], # указываем данные, которые отобразятся
                                )]) # добавляем аргумент, который выделит сегмент-лидер на графике
fig.update_layout(title='Число заведений в зависимости от принадлежности к административному окр',
                  width=800, # указываем размеры графика
                  height=600,
                  annotations=[dict(x=1.12, # вручную настраиваем аннотацию легенды
                                    y=1.05,
                                    text='Округ города',
                                    showarrow=False)])
fig.show() # выводим график
print('*****')
display(district)
```

Число заведений в зависимости от принадлежности к административному округу



	index	district
0	Центральный административный округ	2242
1	Северный административный округ	900
2	Южный административный округ	892
3	Северо-Восточный административный округ	891
4	Западный административный округ	851
5	Восточный административный округ	798
6	Юго-Восточный административный округ	714
7	Юго-Западный административный округ	709
8	Северо-Западный административный округ	409

- Количество заведений в Центральном округе кратно превосходит остальные

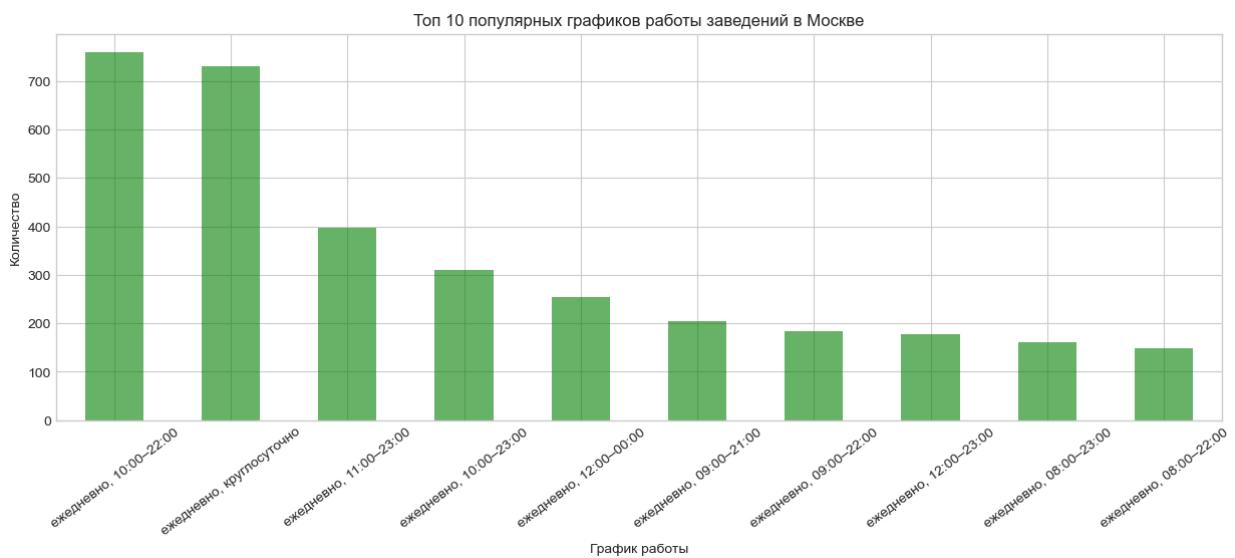
Разберем столбец с часами работы hours

```
In [24]: # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
display(data.hours.unique()[:20])
```

```
array(['ежедневно, 10:00-22:00',
       'пн-чт 11:00-02:00; пт,сб 11:00-05:00; вс 11:00-02:00',
       'ежедневно, 09:00-22:00', 'ежедневно, 10:00-23:00',
       'пн 15:00-04:00; вт-вс 15:00-05:00',
       'пн-чт 10:00-22:00; пт,сб 10:00-23:00; вс 10:00-22:00',
       'ежедневно, 12:00-00:00', 'ежедневно, круглосуточно',
       'ежедневно, 10:00-21:00', 'вт-сб 09:00-18:00',
       'ежедневно, 08:00-22:00', 'ежедневно, 13:00-00:00',
       'пн-пт 08:30-18:30; сб 10:00-20:00', 'ежедневно, 09:00-21:00',
       'пн-пт 09:00-21:00',
       'пн-чт 11:00-22:00; пт,сб 11:00-23:00; вс 11:00-22:00',
       'пн-пт 08:00-22:00; сб,вс 10:00-22:00', 'ежедневно, 10:00-19:00',
       'пн-пт 09:00-16:00', 'ежедневно, 08:00-21:00'], dtype=object)
```

In [25]: # отобразим топ популярных графиков работы

```
data.value_counts('hours').head(10).plot(kind='bar', figsize=(15, 5), alpha=0.6, color='g')
plt.title('Топ 10 популярных графиков работы заведений в Москве')
plt.ylabel('Количество')
plt.xlabel('График работы')
plt.xticks(rotation=37)
plt.show()
print('*****')
display(data.value_counts('hours').head(10))
```



```
*****
hours
ежедневно, 10:00-22:00      759
ежедневно, круглосуточно    730
ежедневно, 11:00-23:00      396
ежедневно, 10:00-23:00      310
ежедневно, 12:00-00:00      254
ежедневно, 09:00-21:00      204
ежедневно, 09:00-22:00      184
ежедневно, 12:00-23:00      178
ежедневно, 08:00-23:00      160
ежедневно, 08:00-22:00      148
dtype: int64
```

In [26]: # Выделим и посчитаем количество заведений с графиком 24/7, создадим отдельный столбец с булевым

```
# посмотрим на результат фильтрации
display(data.loc[((data['hours'].str.contains("ежедневно, круглосуточно"))|(data['hours'].str.contains("24/7")))&(~data['hours'].isna())].head())
print('*****')
print('Количество заведений с графиком работы 24/7 равно:', len(data.loc[((data['hours'].str.contains("ежедневно, круглосуточно"))|(data['hours'].str.contains("24/7")))&(~data['hours'].isna())]))
```

		name	category	address	district	hours	lat	lng	rating	price	avg.
10	Great Room Bar	бар,паб	Москва, Левобережная улица, 12	Северный административный округ	ежедневно, круглосуточно	55.877832	37.469171	4.5	средние	Ц бок пива:2! 35	
17	Чайхана Беш-Бармак	ресторан	Москва, Ленинградское шоссе, 71Б, стр. 2	Северный административный округ	ежедневно, круглосуточно	55.876908	37.449876	4.4	средние	Сред счёт:3! 50	
19	Пекарня	булочная	Москва, Ижорский проезд, 5	Северный административный округ	ежедневно, круглосуточно	55.887969	37.515688	4.4	NaN	Н	
24	Drive Café	кафе	Москва, улица Дыбенко, 9Ас1	Северный административный округ	ежедневно, круглосуточно	55.879992	37.481571	4.0	NaN	Н	
49	2U-Tu-IO	пиццерия	Москва, Ижорская улица, 8А	Северный административный округ	ежедневно, круглосуточно	55.886160	37.508784	2.7	NaN	Сред счёт:	

Количество заведений с графиком работы 24/7 равно: 730

```
In [27]: # все верно взглянем на строки с пропусками в этом столбце
display(data.loc[(data['hours'].isna())].tail())
display(data.loc[(data['hours'].isna())].head())
```

		name	category	address	district	hours	lat	lng	rating	price	avg_bill	m
8236	1y	кафе	Москва, Нагатинская набережная, 40/1к1	Южный административный округ	NaN	55.685528	37.673546	3.4	NaN	NaN		
8375	Улица Гурьянова 55	кафе	Москва, улица Гурьянова, 55	Юго-Восточный административный округ	NaN	55.679981	37.717034	4.5	NaN	NaN		
8378	Восточно-грузинская кухня	быстрое питание	Москва, Зеленодольская улица, 32, корп. 3	Юго-Восточный административный округ	NaN	55.710540	37.767864	4.3	NaN	NaN		
8381	Аэлита	кафе	Москва, Ферганская улица, 8, корп. 2, стр. 1	Юго-Восточный административный округ	NaN	55.708871	37.803831	3.8	NaN	NaN		
8395	Истира Запрафка	кафе	Москва, Юго-Восточный административный округ, ...	Юго-Восточный административный округ	NaN	55.724686	37.710558	2.9	NaN	NaN		

		name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_av
38	Ижора	булочная	Москва, Ижорский проезд, 5А	Северный административный округ	NaN	55.888366	37.514856	4.4	NaN	NaN		
40	Кафе	кафе	Москва, Ижорская улица, 18, стр. 1	Северный административный округ	NaN	55.895115	37.524902	3.7	NaN	NaN		
44	Кафетерий	кафе	Москва, Ангарская улица, 24А	Северный административный округ	NaN	55.876289	37.519315	3.8	NaN	NaN		
56	Рыба из тандыра	быстрое питание	Москва, Коровинское шоссе, 46, стр. 5	Северный административный округ	NaN	55.888010	37.515960	1.5	NaN	NaN		
108	Кафе	бар,паб	МКАД, 82-й километр, вл18	Северо-Восточный административный округ	NaN	55.908930	37.558777	4.2	NaN	NaN		

ничего необычного, просто отсутствует финансовая информация, однако для общей статистики много полезного.

Возможно оказывается дополненеи сета из разных источников

```
In [28]: # добавим столбец новый 'is_24/7'
# заполним положительные значения
data['is_24/7'] = data.loc[(data['hours'].str.contains("ежедневно, круглосуточно"))&(~data['hours'].str.contains("24/7"))]
```

```

    data.loc[(data['hours'].str.contains("ежедневно, круглосуточно")) & (~data['hour

# заполним отрицательные значения
data.loc[~(data['hours'].isna()) & (data['is_24/7'].isna()), 'is_24/7'] = False

# выполним проверку
len(data[data['is_24/7']==False])

```

Out[28]: 7140

- Основная часть заведений работает с ежедневно 10:00 до 22:00 либо 24/7, что может быть связано с графиком работы торговых центров в первом случае. В топ 10 популярных графиков все заведения работают без выходных.

Значение координат обычно добавляются автоматически системой, поэтому подробно смотреть эти столбцы нет смысла, взглянем лишь на их описание во избежание серьезных ошибок

```
In [29]: print('широта', data.lat.describe());
print("*****")
print('долгота', data.lng.describe());
```

```

широта count    8406.000000
mean      55.750109
std       0.069658
min      55.573942
25%      55.705155
50%      55.753425
75%      55.795041
max      55.928943
Name: lat, dtype: float64
*****
долгота count    8406.000000
mean      37.608570
std       0.098597
min      37.355651
25%      37.538583
50%      37.605246
75%      37.664792
max      37.874466
Name: lng, dtype: float64
```

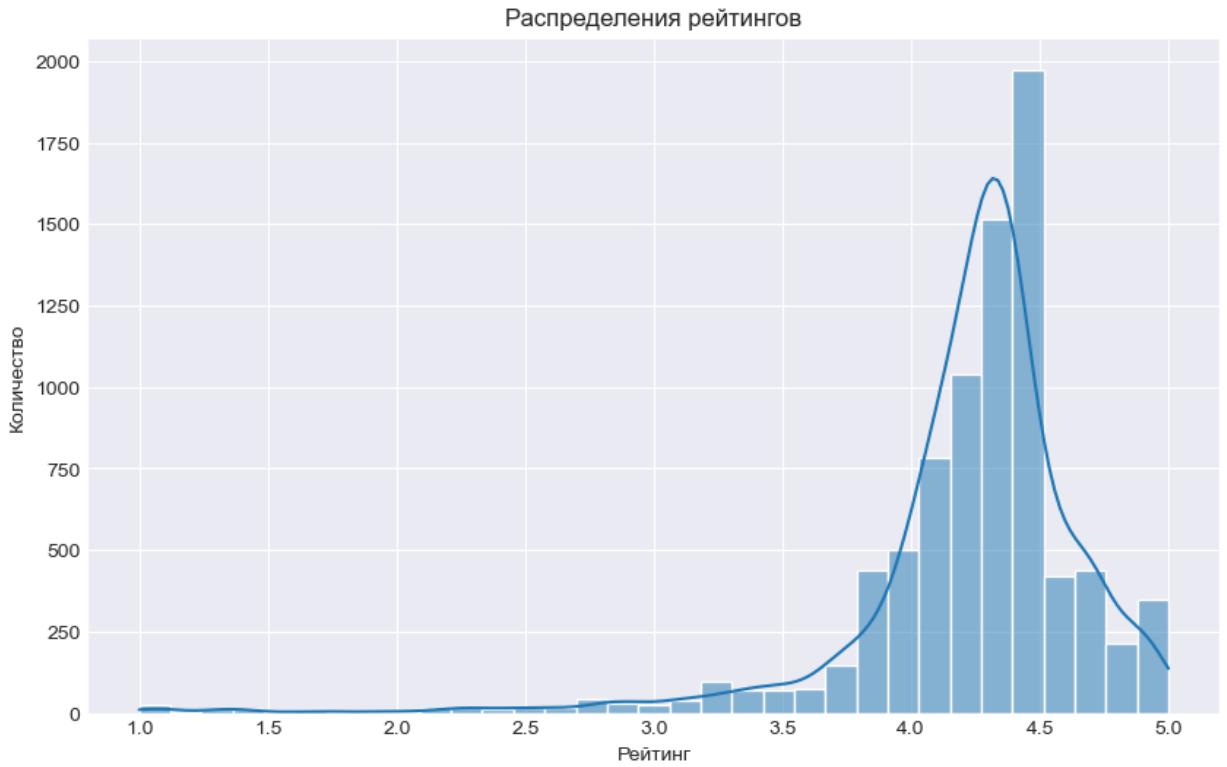
все в пределах нормы, идем дальше

Посмотрим на рейтинги в столбце rating

```
In [30]: # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
display(data.rating.unique())

array([5. , 4.5, 4.6, 4.4, 4.7, 4.8, 4.3, 4.9, 4.2, 4.1, 4. , 3.8, 3.9,
       3.7, 3.6, 2.8, 2.7, 3.1, 1.5, 2. , 1.4, 3.3, 3.5, 3.2, 2.9, 3. ,
       3.4, 2.3, 2.2, 2.5, 2.6, 1.7, 1. , 1.1, 2.4, 1.3, 1.2, 2.1, 1.8,
       1.9, 1.6])
```

```
In [31]: # построим гистограмму для отображения распределения среднего рейтинга
sns.set_style('darkgrid')
plt.figure(figsize=(10,6))
ax = sns.histplot(x='rating', data=data, kde=True, bins=33)
plt.grid(True)
ax.set_title('Распределение рейтингов')
ax.set_ylabel('Количество')
ax.set_xlabel('Рейтинг')
plt.show();
print('*****')
print(data.rating.describe())
```



- Средний медианный рейтинг равен 4.3 балла, рейтинги считаются по 5 бальной шкале. Основная часть заведений имеют рейтинг между 4.1 и 4.4 баллами.

Разберем строку с категориями цен price

```
In [32]: # # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
display(data['price'].unique())
```

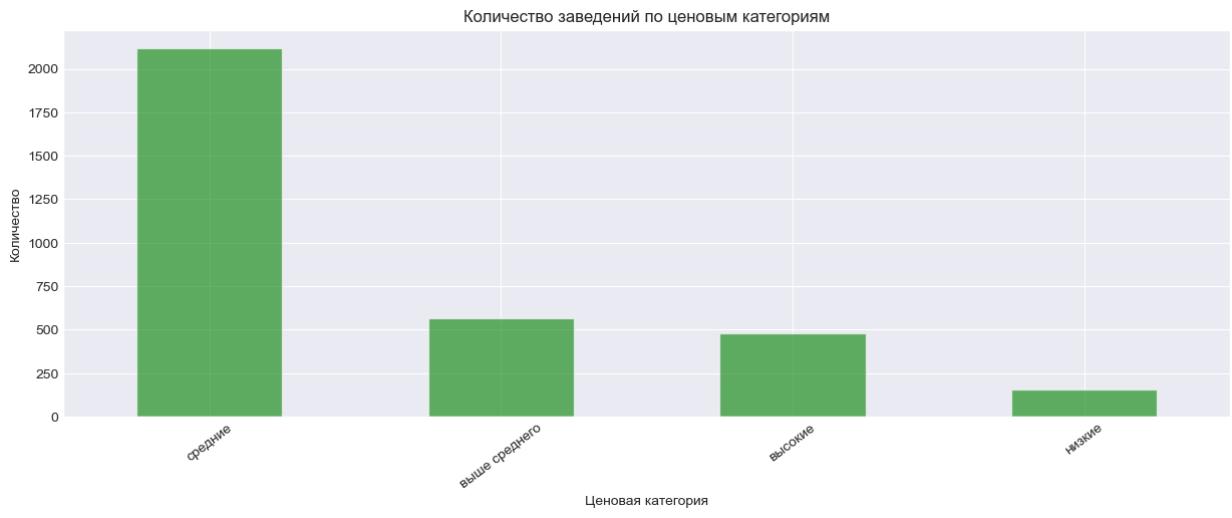
```
# приведем к категориальному типу
data['price']= data['price'].astype('category')
```

```
array([nan, 'выше среднего', 'средние', 'высокие', 'низкие'], dtype=object)
```

```
In [33]: print('Количество пропусков в колонке price равно', data['price'].isna().sum() )
```

```
Количество пропусков в колонке price равно 5091
```

```
In [34]: # отобразим Количество заведений с по ценовым категориям для
data.value_counts('price').plot(kind='bar',figsize=(15,5),alpha=0.6,color='g')
plt.title('Количество заведений по ценовым категориям')
plt.ylabel('Количество')
plt.xlabel('Ценовая категория')
plt.xticks(rotation=37)
plt.show();
print('*****')
display(data.value_counts('price'))
```



```
*****
price
средние      2117
выше среднего 564
высокие      478
низкие        156
dtype: int64
```

- Основная часть заведений с доступной информацией по принадлежности к ценовой группе относится к средней категории, что логично и направлено на самую большую целевую группу потребителей. Заведения низкой ценовой категории имеют самое низкое значение, доля дорогих и очень дорогих заведений имеет среднее значение, но кратно уступает лидеру. Количество пропусков в столбце равно 5091

Разберем столбец со значением среднего счета avg_bill

```
In [35]: # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
display(data.avg_bill.unique()[:40])
```

```
array(['Средний счёт:1500-1600 ', 'Средний счёт:от 1000 ',
       'Цена чашки капучино:155-185 ', 'Средний счёт:400-600 ',
       'Средний счёт:199 ', 'Средний счёт:200-300 ',
       'Средний счёт:от 500 ', 'Средний счёт:1000-1200 ',
       'Цена бокала пива:250-350 ', 'Средний счёт:330 ',
       'Средний счёт:1500 ', 'Средний счёт:300-500 ',
       'Средний счёт:140-350 ', 'Средний счёт:350-500 ',
       'Средний счёт:300-1500 ', 'Средний счёт:от 240 ',
       'Средний счёт:200-250 ', 'Средний счёт:328 ',
       'Средний счёт:300 ', 'Средний счёт:от 345 ',
       'Средний счёт:60-400 ', 'Средний счёт:900 ',
       'Средний счёт:500-800 ', 'Средний счёт:500-1000 ',
       'Средний счёт:600-700 ', 'Цена бокала пива:120-350 ',
       'Средний счёт:1000-1500 ', 'Средний счёт:1500-2000 ',
       'Цена чашки капучино:150-190 ', 'Средний счёт:2000-2500 ',
       'Средний счёт:600 ', 'Средний счёт:450 ',
       'Цена чашки капучино:120-170 ', 'Средний счёт:100-500 ',
       'Средний счёт:от 850 ', 'Цена чашки капучино:100-200 ',
       'Средний счёт:250-600 ', 'Средний счёт:2100 ',
       'Средний счёт:349 '], dtype=object)
```

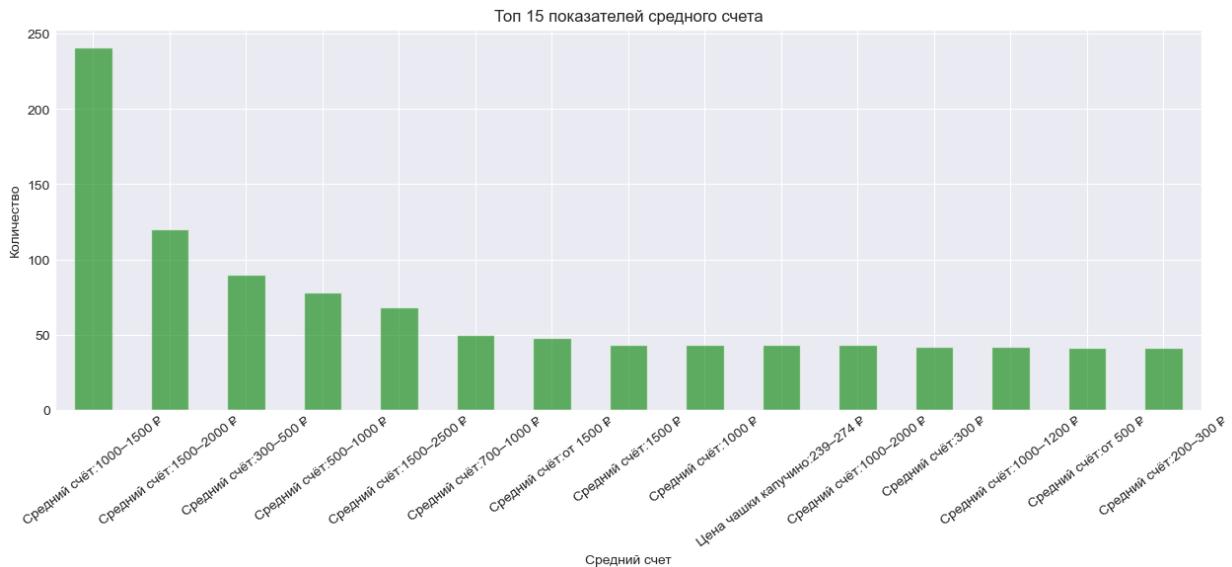
явных ошибок и дубликатов не видно

```
In [36]: print('Количество пропусков в колонке avg_bill равно', data['avg_bill'].isna().sum() )
```

Количество пропусков в колонке avg_bill равно 4590

```
In [37]: # отобразим топ 15 самых популярных значений среднего счета
```

```
data.avg_bill.value_counts().head(15).plot(kind='bar', figsize=(15,5), alpha=0.6, color='g')
plt.title('Топ 15 показателей среднего счета')
plt.ylabel('Количество')
plt.xlabel('Средний счет')
plt.xticks(rotation=37)
plt.show();
print('*****')
display(data.avg_bill.value_counts().head(15))
```



```
*****
Средний счёт:1000–1500          241
Средний счёт:1500–2000          120
Средний счёт:300–500            90
Средний счёт:500–1000           78
Средний счёт:1500–2500          68
Средний счёт:700–1000           50
Средний счёт:от 1500            48
Средний счёт:1500               43
Средний счёт:1000               43
Средний счёт:1000               43
Цена чашки капучино:239–274    43
Средний счёт:1000–2000          43
Средний счёт:300                42
Средний счёт:1000–1200          42
Средний счёт:от 500              41
Средний счёт:200–300             41
Name: avg_bill, dtype: int64
```

```
In [38]: # посмотрим на значения не выделенные в отдельный столбец
data.loc[(~data['avg_bill'].isna())&(data['middle_coffee_cup'].isna())&(data['middle_avg_bill'].isna())]
```

```
Out[38]: 10      Цена бокала пива:250–350
67      Цена бокала пива:120–350
97      Цена бокала пива:90–230
241     Цена бокала пива:160–499
417     Цена бокала пива:199–300
418     Цена бокала пива:от 140
421     Цена бокала пива:от 149
424     Цена бокала пива:150–450
814     Цена бокала пива:130–150
1030    Цена бокала пива:220–350
Name: avg_bill, dtype: object
```

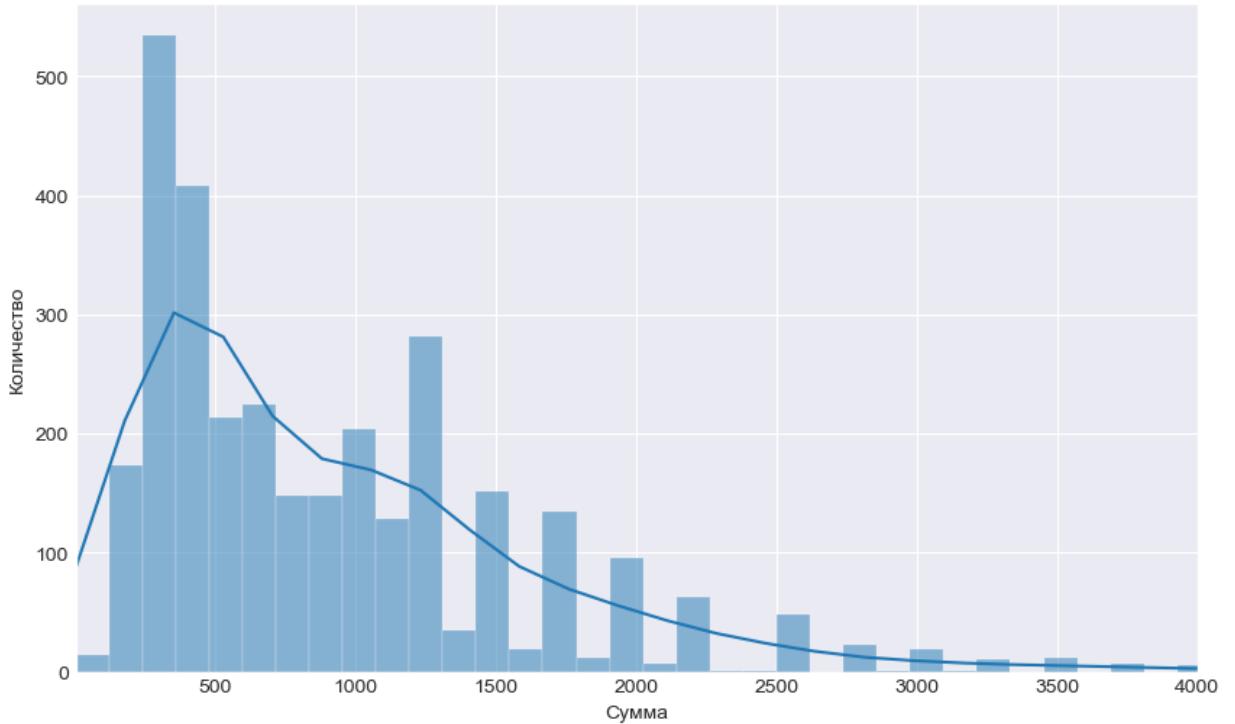
- Самый частый счет в заведениях находится в пределах 1000-2000. Количество пропусков в столбце 4590.
- Цена бокала пива не выделена из значений

Разберем столбец со значением среднего счета middle_avg_bill

```
In [39]: print('Количество пропусков в колонке middle_avg_bill равно', data['middle_avg_bill'].isna().sum)
Количество пропусков в колонке middle_avg_bill равно 5257
```

```
In [40]: # построим гистограмму для отображения распределения среднего чека
sns.set_style('darkgrid')
plt.figure(figsize=(10,6))
ax = sns.histplot(x='middle_avg_bill', data=data, kde=True)
plt.xlim(1,4000)
plt.grid(True)
ax.set_title('Распределения сумм среднего чека')
ax.set_ylabel('Количество')
ax.set_xlabel('Сумма')
plt.show();
print('*****')
print(data.middle_avg_bill.describe())
```

Распределения сумм среднего чека



```
*****
```

```
count      3149.000000
mean       958.053668
std        1009.732845
min        0.000000
25%       375.000000
50%       750.000000
75%      1250.000000
max      35000.000000
Name: middle_avg_bill, dtype: float64
```

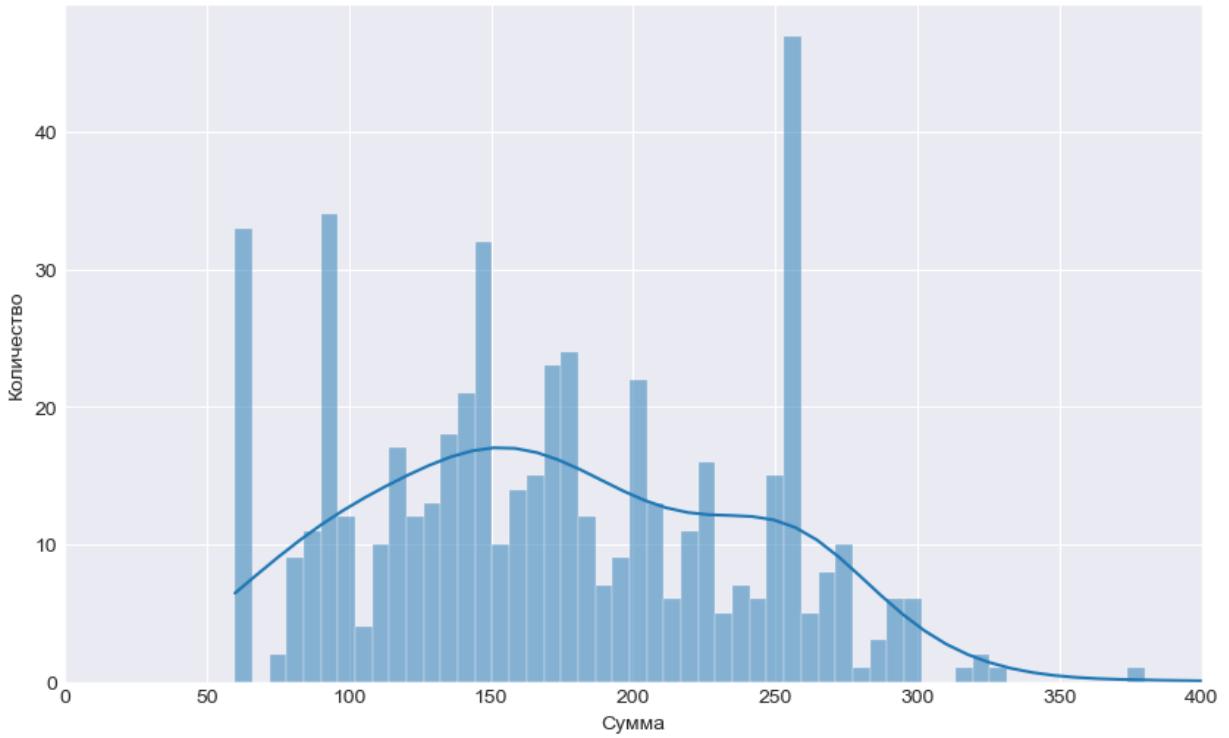
- Среднее медианное значение среднего чека по всем заведениям расположилось в районе 750 рублей, самые часто встречающиеся суммы среднего чека находятся между 375 и 1250 рублями. На гистограмме видны ступени в виде пиков, что может говорить об особенностях подсчета в разных заведениях, например крупных сетевых. Количество пропусков равно 5257

Разберем столбец со значением средней стоимости чашки капучино `middle_coffee_cup`

```
In [41]: print('Количество пропусков в колонке middle_coffee_cup равно', data['middle_coffee_cup'].isna())
Количество пропусков в колонке middle_coffee_cup равно 7871

In [42]: # построим гистограмму для отображения распределения средней стоимости чашки капучино
plt.figure(figsize=(10,6))
ax = sns.histplot(x='middle_coffee_cup', data=data, kde=True, bins=250)
sns.set_style('darkgrid')
plt.xlim(0,400)
plt.grid(True)
ax.set_title('Распределения сумм средней стоимости чашки капучино')
ax.set_ylabel('Количество')
ax.set_xlabel('Сумма')
plt.show();
print('*****')
print(data.middle_coffee_cup.describe())
```

Распределения сумм средней стоимости чашки капучино



```
*****
count      535.000000
mean       174.721495
std        88.951103
min        60.000000
25%       124.500000
50%       169.000000
75%       225.000000
max       1568.000000
Name: middle_coffee_cup, dtype: float64
```

- Средняя медианная стоимость чашки капучино равняется 169 рублям, основная часть чашек капучино продается в границах между 124 и 225 рублями. На гистограмме видны пики, что может говорить о стандартной стоимости чашки в крупных сетевых заведениях

Разберем количество посадочных мест в колонке seats

```
In [43]: # посмотрим на уникальные значения
display(data.seats.unique()[:40])
```

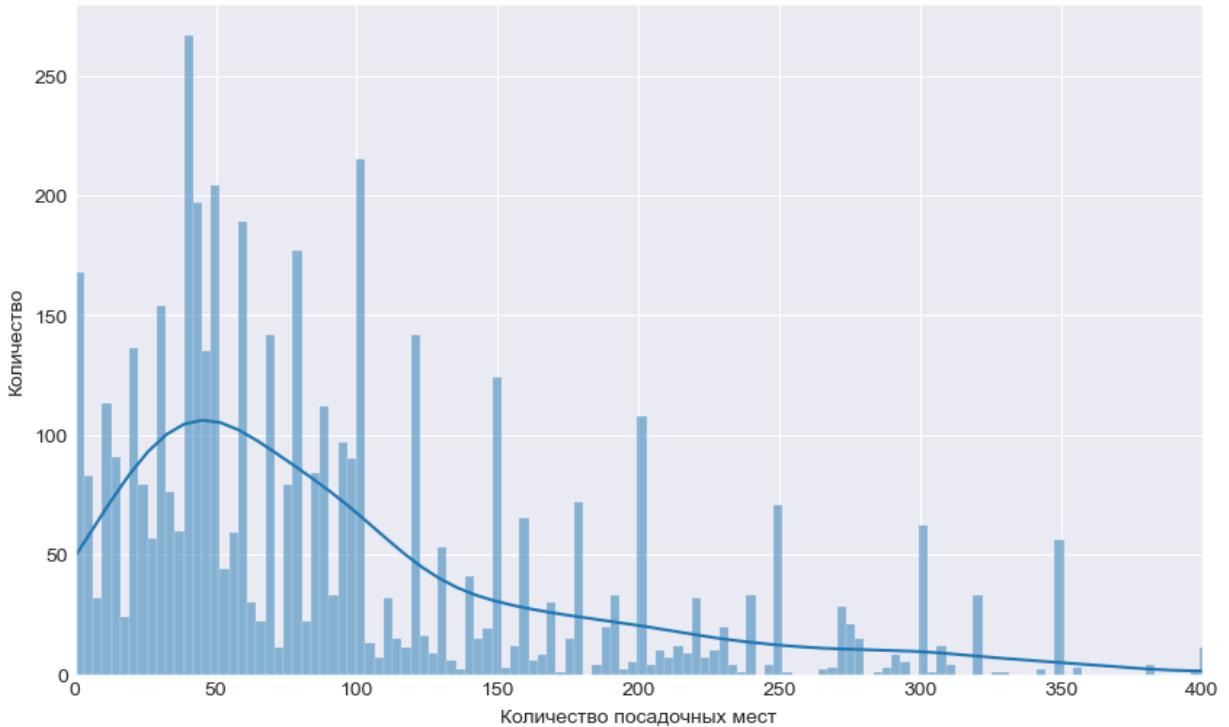
```
array([ nan,    4.,   45.,  148.,   79.,   65.,  102.,  180.,   96.,   25.,
       46.,   40.,  247.,   21.,    8.,   35.,  240.,   85.,   12.,   60.,  120.,
      16.,   80.,   50.,   44.,   43.,   90.,    0.,  198.,   95.,  350.,  124.,
     10.,   70.,   48.,    6.,   98.,   30.,   52.,   20.])
```

```
In [44]: print('Количество пропусков в колонке seats равно', data['seats'].isna().sum() )
```

```
Количество пропусков в колонке seats равно 3611
```

```
In [45]: # построим гистограмму для отображения распределения количества посадочных мест среди заведений
plt.figure(figsize=(10,6))
ax = sns.histplot(x='seats', data=data, kde=True, bins=400)
sns.set_style('darkgrid')
plt.xlim(0,400)
plt.grid(True)
ax.set_title('Распределения количества посадочных мест среди заведений')
ax.set_ylabel('Количество')
ax.set_xlabel('Количество посадочных мест')
plt.show();
print('*****')
print(data.seats.describe())
```

Распределения количества посадочных мест среди заведений



```
*****
```

```
count      4795.000000
mean       108.421689
std        122.833396
min        0.000000
25%       40.000000
50%       75.000000
75%       140.000000
max      1288.000000
Name: seats, dtype: float64
```

- Среднее медианное значение посадочных мест среди заведений общественного питания равно 75, основная часть их часть имеет значения между 40 и 140, на гистограмме можно увидеть пики кратные 50 посадочным местам, что говорит об их ступенчатом увеличении. Количество пропусков в колонке seats равно 3611

Разберем столбец, отображающий является ли заведение сетевым

```
In [46]: # отобразим уникальные значения для проверки на ошибки и неявные дубликаты
display(data['chain'].unique())
array([0, 1], dtype=int64)
```

```
In [47]: # выделим не единичные заведения маркированные как несетевые
b = data.loc[data['chain']==0].groupby(by='name')['name'].count().sort_values()
print('Количество не единичных заведений маркерованных несетевыми', len(b.loc[b>1]))
Количество не единичных заведений маркерованных несетевыми 16
```

```
In [48]: # выделим единичные заведения маркированные как сетевые
k = data.loc[data['chain']==1].groupby(by='name')['name'].count().sort_values()
print('Количество единичных заведений маркерованных сетевыми', len(k.loc[k==1]))
Количество единичных заведений маркерованных сетевыми 64
```

```
In [49]: # выделем и посмотрим на эти заведения
k = k.loc[k==1]

is_not_chain = data.query("name in @k.index")
display(is_not_chain.head())

display(is_not_chain.groupby(by='category')['name'].count());
```

		name	category	address	district	hours	lat	lng	rating	price	avg_bill
133	Крепери	бар,паб	Москва, улица Лескова, 14	Северо-Восточный административный округ	ежедневно, 10:00–22:00	55.897490	37.604722	4.0	NaN		
206	Vintage	кафе	Москва, Дмитровское шоссе, 163Ак1	Северо-Восточный административный округ	ежедневно, 10:00–22:00	55.908956	37.540036	4.1	NaN		
382	Halal food	кафе	Москва, Петрозаводская улица, 34	Северный административный округ	ежедневно, круглосуточно	55.867428	37.489429	4.4	средние	Средн е, счёт: 4	
613	Waурма	кафе	Москва, улица Героев Панфиловцев, 1/2	Северо-Западный административный округ	NaN	55.852060	37.438371	4.1	NaN		
745	В своей тарелке	столовая	Москва, Локомотивный проезд, 21	Северный административный округ	NaN	55.845414	37.573537	3.9	NaN		

```
category
бар,паб      5
булочная    1
быстрое питание  5
кафе        30
кофейня     6
пиццерия    5
ресторан    9
столовая    3
Name: name, dtype: int64
```

In [50]: # выделим и посмотрим на эти заведения
b = b.loc[b>1]

```
is_chain = data.query("name in @b.index")
display(is_chain.head())

display(is_chain.groupby(by='category')['name'].count());
```

		name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle
40	Кафе	кафе	Москва, Ижорская улица, 18, стр. 1	Северный административный округ	NaN	55.895115	37.524902	3.7	NaN	NaN		
47	Кафе	кафе	Москва, улица Маршала Федоренко, 7	Северный административный округ	ежедневно, 11:00–00:00	55.880306	37.489760	2.8	NaN	NaN		
50	Шаурма	быстрое питание	Москва, улица Дыбенко, 44	Северный административный округ	ежедневно, круглосуточно	55.878339	37.483154	3.9	NaN	NaN		
108	Кафе	бар,паб	Москва, МКАД, 82-й километр, вл18	Северо-Восточный административный округ	NaN	55.908930	37.558777	4.2	NaN	NaN		
123	Кафе	кафе	Москва, Шенкурский проезд, 14	Северо-Восточный административный округ	NaN	55.897794	37.591395	4.3	NaN	NaN		

```
category
бар,паб      4
булочная    3
быстрое питание  45
кафе        210
кофейня     15
пиццерия    4
ресторан    44
столовая    40
Name: name, dtype: int64
```

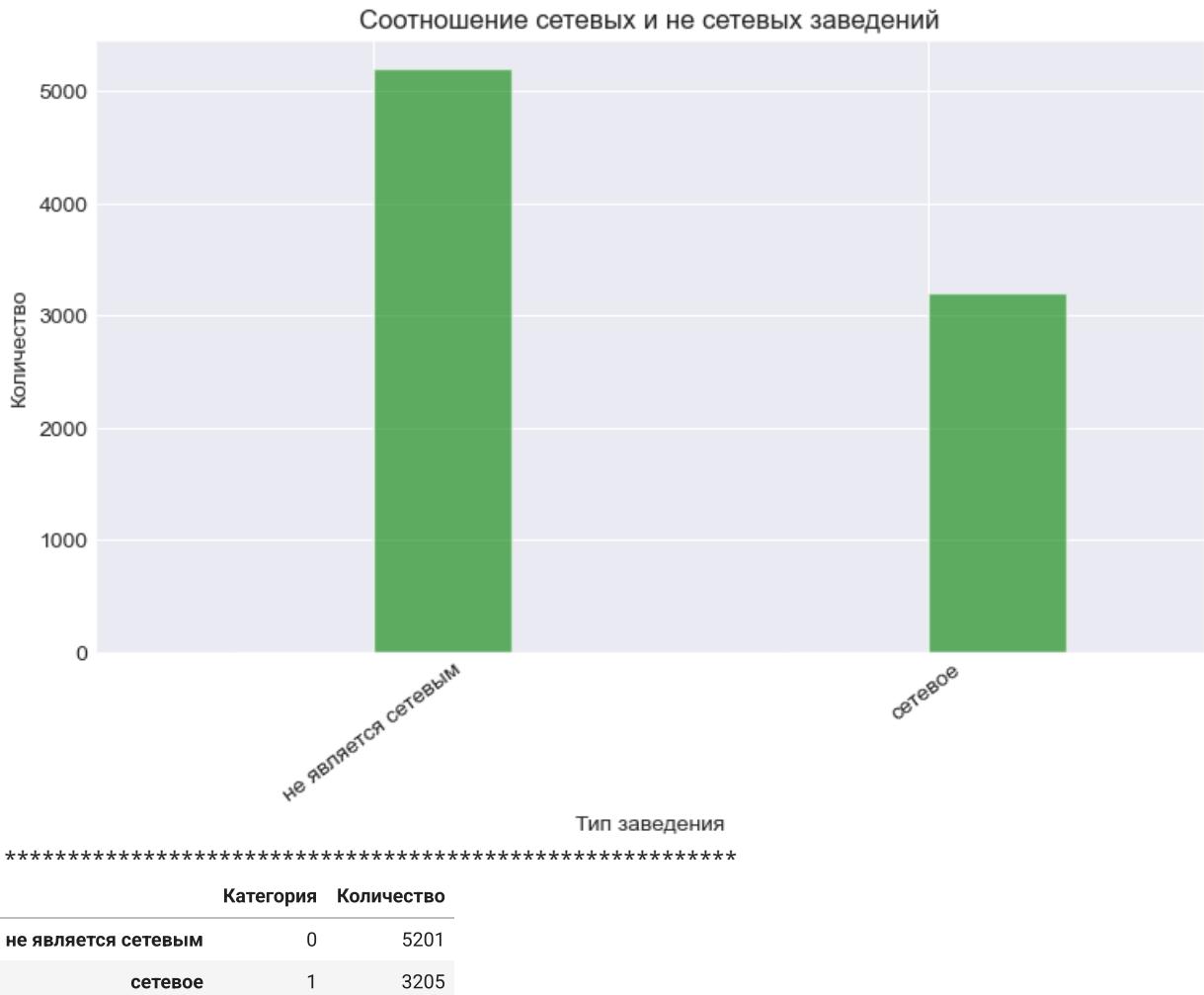
In [51]: # отобразим соотношение сетевых и не сетевых заведений

```
chain = data.chain.value_counts().reset_index()
```

```

chain.index = ['не является сетевым', 'сетевое']
chain.columns = ['Категория', 'Количество']
chain.plot(kind='bar', figsize=(9,5), alpha=0.6, color='g')
plt.title('Соотношение сетевых и не сетевых заведений')
plt.ylabel('Количество')
plt.xlabel('Тип заведения')
plt.legend('')
plt.xticks(rotation=37)
plt.show();
print('*****')
display(chain)

```



- Число заведений которые не является сетевыми больше чем сетевых и равняется 5201, тогда как сетевых 3205. В данных возможно наличие ошибок в принадлежности заведений к типу

[к содержанию](#)

ВЫВОД ПОСЛЕ ПРЕДВАРИТЕЛЬНОГО АНАЛИЗА ПО КОЛОНКАМ ОБЩЕЙ ВЫБОРКИ ДАННЫХ:

- В данных присутствует большое количество пропусков в колонках, содержащих финансовую информацию и количество посадочных мест. Пропуски распределены относительно равномерно по данным. Дубликаты отсутствуют
- В топе 'Шоколадница', 'Доминос Пицца' и 'Додо Пицца' с 120, 76 и 74 точками соответственно, что может говорить о наличии многих сетевых заведений в городе. Названия - 'Кафе', 'Ресторан' и 'Шаурма' отображают количество несвязанных заведений и показывают наличие не малого их количества в сравнении с крупными сетями

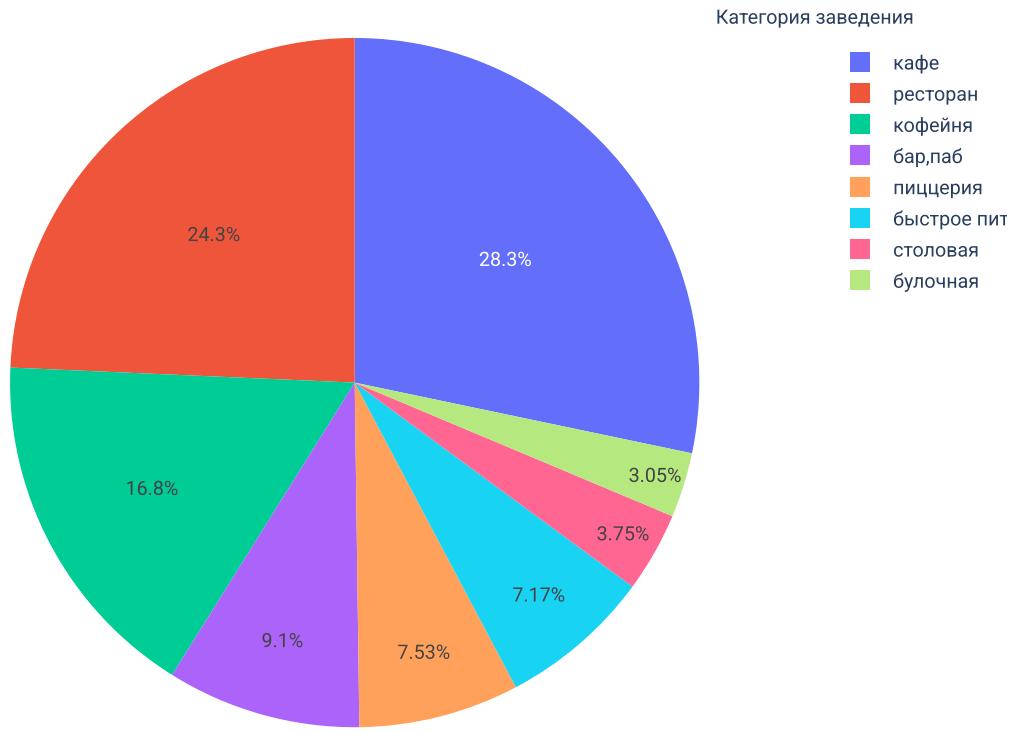
- лидируют заведения в категориях кафе, ресторан и кофейня и имеют кратное превосходство в количестве точек над заведениями в остальных категориях. Столовые и буточные менее популярны и имеют 315 и 256 заведений соответственно
- В топе по количеству заведений лидируют шоссе и проспекты, количество уникальных названий улиц равно: 1448
- Количество заведений в Центральном округе кратно превосходит остальные
- Основная часть заведений работает ежедневно с 10:00 до 22:00 либо 24/7, что может быть связано с графиком работы торговых центров в первом случае. В топ 10 популярных графиков все заведения работают без выходных.
- Средний медианный рейтинг равен 4.3 балла, рейтинги считаются по 5 бальной шкале. Основная часть заведений имеют рейтинг между 4.1 и 4.4 баллами.
- Основная часть заведений с доступной информацией по принадлежности к ценовой группе относится к средней категории, что логично и направлено на самую большую целевую группу потребителей. Заведения низкой ценовой категории имеют самое низкое значение, доля дорогих и очень дорогих заведений имеет среднее значение, но кратно уступает лидеру
- Самый частый счет в заведениях находится в пределах 1000-2000. Цена бокала пива не выделена из значений
- Среднее медианное значение среднего чека по всем заведениям расположилось в районе 750 рублей, самые часто встречающиеся суммы среднего чека находятся между 375 и 1250 рублями
- Средняя медианная стоимость чашки капучино равняется 169 рублям, основная часть чашек капучино продается в границах между 124 и 225 рублями. На гистограмме видны пики, что может говорить о стандартной стоимости чашки в крупных сетевых заведениях
- Среднее медианное значение посадочных мест среди заведений общественного питания равен 75, основная часть их часть имеет значения между 40 и 140
- Число заведений которые не является сетевыми больше чем сетевых и равняется 5201, тогда как сетевых 3205. В данных возможно наличие ошибок в принадлежности заведений к типу

АНАЛИЗ ВЗАИМОСВЯЗЕЙ И ОСОБЕННОСТЕЙ ЗАВЕДЕНИЙ ОБЩЕСТВЕННОГО ПИТАНИЯ МОСКВЫ

Проведем анализ количества заведений по категориям

```
In [52]: # готовим данные для графика
category = pd.DataFrame(data['category'].value_counts()).reset_index()
# строим диаграмму с сегментами
fig = go.Figure(data=go.Pie(labels=category['index'], # указываем значения, которые появятся на
                                values=category['category'],# указываем данные, которые отобразятся
                                ))
fig.update_layout(title='Число заведений в зависимости от их категории', # указываем заголовок
                  width=800, # указываем размеры графика
                  height=600,
                  annotations=[dict(x=1.12, # вручную настраиваем аннотацию легенды
                                    y=1.05,
                                    text='Категория заведения',
                                    showarrow=False)])
fig.show() # выводим график
```

Число заведений в зависимости от их категории



- Лидируют кафе, рестораны и кофейни с долями 28%, 24% и 17% соответственно от общего количества заведений. Столовые и булочные занимают 4% и 3%.

```
In [53]: # подготовим выборку для топа заведений по категориям
top_cat_name_all = data.groupby(by=['category', 'name']).count().sort_values(by='lat', ascending=False)

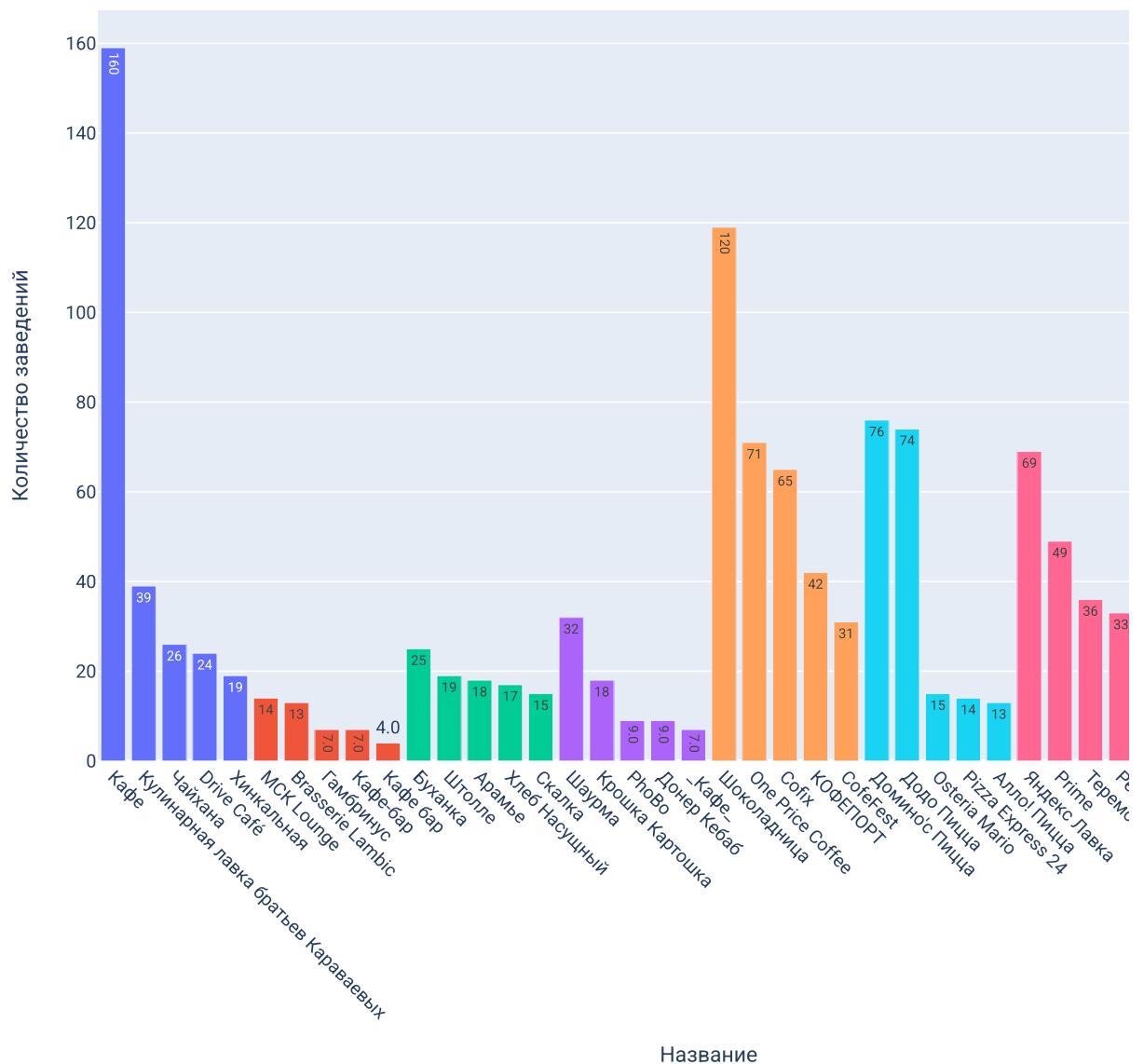
# подпишем колонки и выделим топ 5
top_cat_name_all.columns = ['Категория', 'Название', 'Количество заведений']
top_cat_name_good = top_cat_name_all.loc[top_cat_name_all['Категория'] == 'кафе'].head(5).reset_index()

In [54]: # повторим для остальных категорий
for cat in ['бар,лаб', 'булочная', 'быстрое питание', 'кофейня', 'пиццерия', 'ресторан', 'столовая']:
    top_cat_name = top_cat_name_all.loc[top_cat_name_all['Категория'] == cat].head(5).reset_index()
    top_cat_name_good = pd.concat([top_cat_name_good, top_cat_name])

In [55]: # добавим правки для графика
top_cat_name_good.loc[(top_cat_name_good['Категория'] == 'столовая') & (top_cat_name_good['Название'] == 'столовая'), 'Количество заведений'] = 4
top_cat_name_good.loc[(top_cat_name_good['Категория'] == 'быстрое питание') & (top_cat_name_good['Название'] == 'быстрое питание'), 'Количество заведений'] = 3

In [56]: # строим график
fig = px.bar(top_cat_name_good, x='Название', y='Количество заведений',
             color='Категория', title='Топ 5 заведений по количеству точек в разных категориях',
             fig.update_xaxes(tickangle=45)
             fig.update_layout(width=1050, height=800)
             fig.show()
```

Топ 5 заведений по количеству точек в разных категориях



- Кафе с самым большим количеством точек "Кулинарная лавка братьев Караваевых", у кофеен лидирует "Шоколадница", в барах - "MCK Lounge", среди ресторанов лидируют "Яндекс Лавка" и "Prime". Кафе и столовые с одноименными названиями говорят о наличии большого количества мелких заведений без названия

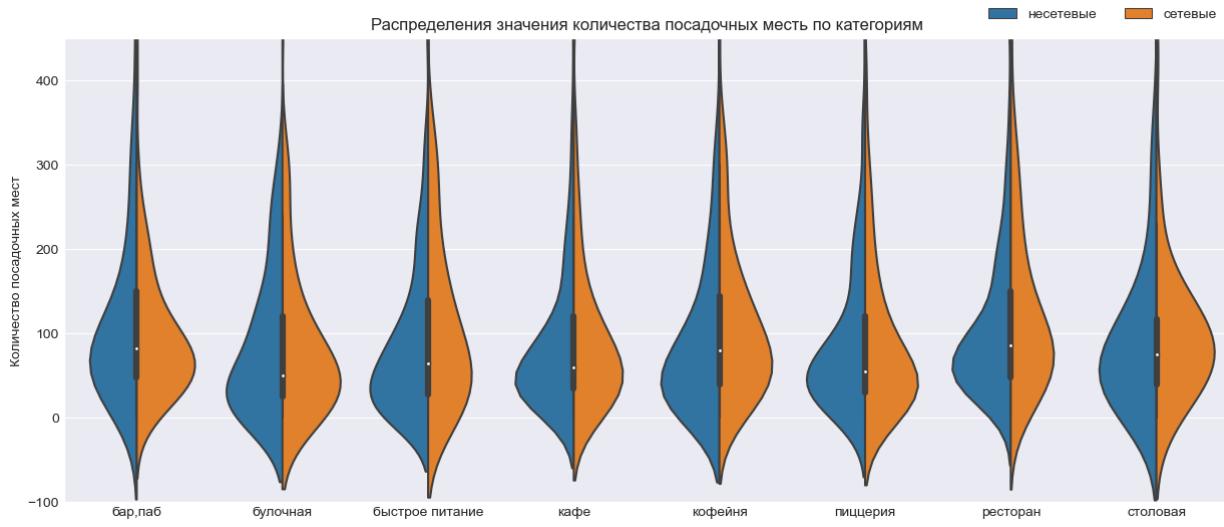
Проведем анализ среднего значения количества посадочных мест по категориям

```
In [57]: # сделаем функцию для создания нового столбца
def ne(s):
    try:
        if s == 0:
            return 'несетевые'
        else:
            return 'сетевые'
    except:
        err

# применим функцию
data['new_chain'] = data['chain'].apply(ne)
```

```
In [58]: sns.set_style('darkgrid')
plt.figure(figsize=(15,6))
ax = sns.violinplot(y='seats', x ='category', data=data,hue='new_chain',split=True)
sns.move_legend(ax, "lower right", bbox_to_anchor=(1, 1.01), ncol=3, title=None, frameon=False)
plt.title('Распределения рейтингов по категориям')
plt.title('Распределения значения количества посадочных мест по категориям')
plt.xlabel('')
plt.ylim(-100,450)
plt.ylabel('Количество посадочных мест')
plt.show();

display('СЕТЕВЫЕ',data.loc[data['new_chain']=='сетевые'].groupby(by='category')['seats'].median())
print('*****')
display('НЕСЕТЕВЫЕ',data.loc[data['new_chain']=='несетевые'].groupby(by='category')['seats'].med
```



```
'СЕТЕВЫЕ'
category
ресторан      98.0
кофейня       90.0
бар,паб        80.0
столовая      80.0
быстрое питание 75.0
кафе          70.0
пиццерия      52.0
булочная       50.0
Name: seats, dtype: float64
*****
'НЕСЕТЕВЫЕ'
category
бар,паб        85.0
ресторан      80.0
столовая      66.0
кофейня       60.0
пиццерия      60.0
быстрое питание 56.0
кафе          55.0
булочная       50.0
Name: seats, dtype: float64
```

```
In [59]: # выделим данные
top_cat_name_seats = data.pivot_table(index=['category','name'],values='seats',aggfunc=['count'],

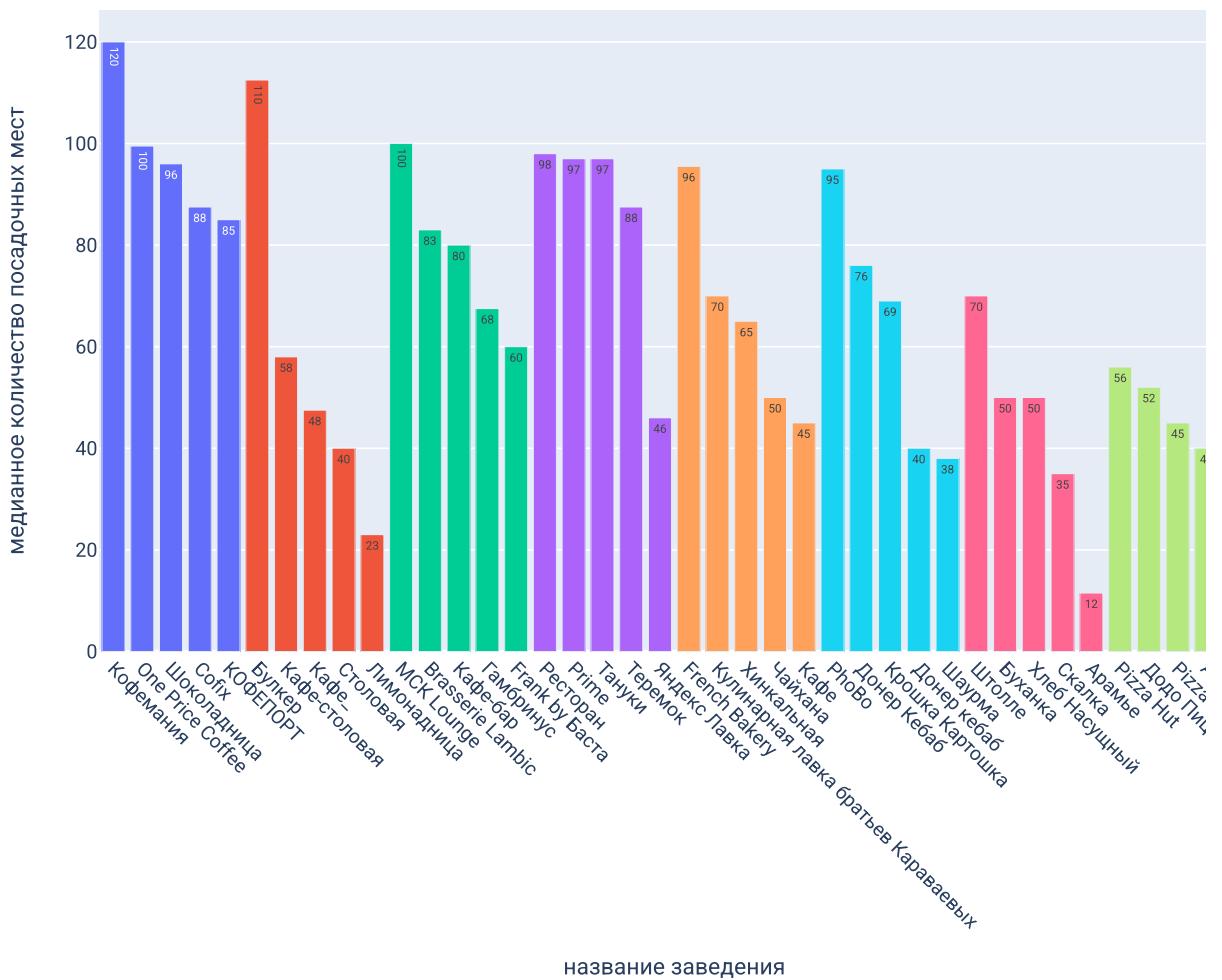
# подпишем колонки, отсортируем и уберем дубликаты
top_cat_name_seats.columns = ['категория', 'название заведения', 'количество заведений', 'медианное'
top_cat_name_seats = top_cat_name_seats.sort_values(by=['количество заведений', 'категория'],ascending=False)
top_cat_name_seats = top_cat_name_seats.drop_duplicates(subset=['категория', 'название заведения'])
```

```
In [60]: # соберем топ 5 по количеству точек для каждой категории
top_cat_name_seats_good = pd.DataFrame()
for cat in ['кафе', 'бар,паб', 'булочная', 'быстрое питание', 'кофейня', 'пиццерия', 'ресторан',
            top_cat_name_seats_ = top_cat_name_seats.loc[top_cat_name_seats['категория'] == cat].sort_v
            top_cat_name_seats_good = pd.concat([top_cat_name_seats_good,top_cat_name_seats_])
```

```
In [61]: # поправим одноименные названия для графика
top_cat_name_seats_good.loc[(top_cat_name_seats_good['категория'] == 'столовая')\n    &(top_cat_name_seats_good['название заведения'] == 'Кафе'), 'название'\n\n    top_cat_name_seats_good = top_cat_name_seats_good.sort_values(by='медианное количество посадочных\n\n    мест', ascending=False)
```

```
In [62]: # строим график
fig = px.bar(top_cat_name_seats_good, x='название заведения', y='медианное количество посадочных\n\n    мест', color='категория', title='Медианные значения количества посадочных мест в топ 5-ти\n\n    заведений', text_auto='.2s')
fig.update_xaxes(tickangle=45)
fig.update_layout(width=950, height=700)
fig.show()
```

Медианные значения количества посадочных мест в топ 5-ти заведений по количеству



- Лидируют по количеству посадочных мест заведения в категориях: рестораны бары и кофейни и столовые, что можно увязать с особенностью ведения бизнеса, где клиенты получают полный набор услуг непосредственно в заведении, в отличие от булочных и пиццерий где довольно популярна работа через доставку и на вынос. В топе по количеству заведений для категории кофеен находится 'Шоколадница' со средним показателем в 120 мест, у ресторанов лидируют 'Prime' и 'Тануки' с 97 местами. В сетевых заведениях у ресторанов, кофеен, и столовых наблюдается превышение посадочных мест над несетевыми заведениями, что может говорить об особенностях ведения бизнеса и привлечения клиентов

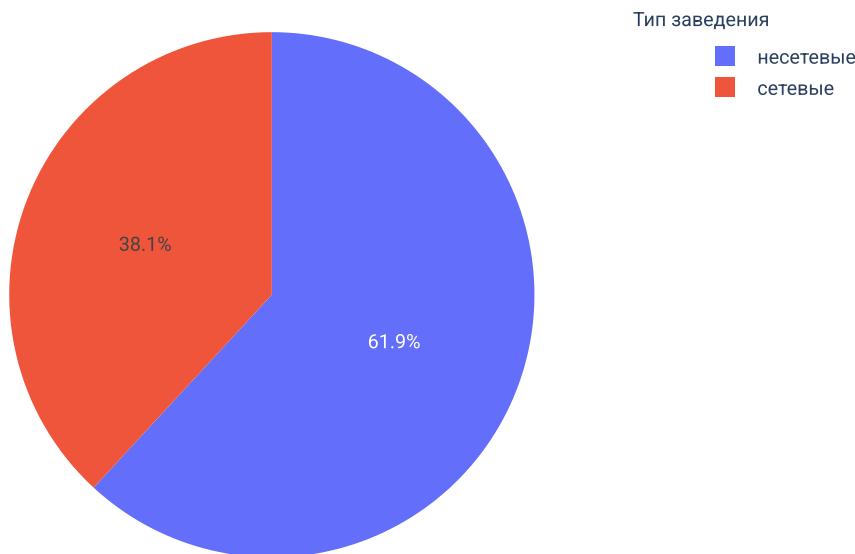
Проведем анализ соотношения сетевых и несетевых заведений

```
In [63]: # подготуим данные
category = pd.DataFrame(data['chain'].value_counts()).reset_index()

# переименуем колонки
category.columns = ['Тип заведения', 'Число заведений']
category.loc[category['Тип заведения']==0, 'Тип заведения'] = 'несетевые'
category.loc[category['Тип заведения']==1, 'Тип заведения'] = 'сетевые'

In [64]: # строим диаграмму с сегментами
fig = go.Figure(data=[go.Pie(labels=category['Тип заведения'], # указываем значения, которые показываем
                               values=category['Число заведений'], # указываем данные, которые отображаем
                               )])
fig.update_layout(title='Число заведений в зависимости от их типа', # указываем заголовок графика
                  width=700, # указываем размеры графика
                  height=500,
                  annotations=[dict(x=1.12,
                                     y=1.05,
                                     text='Тип заведения',
                                     showarrow=False)])
fig.show() # выводим график
```

Число заведений в зависимости от их типа



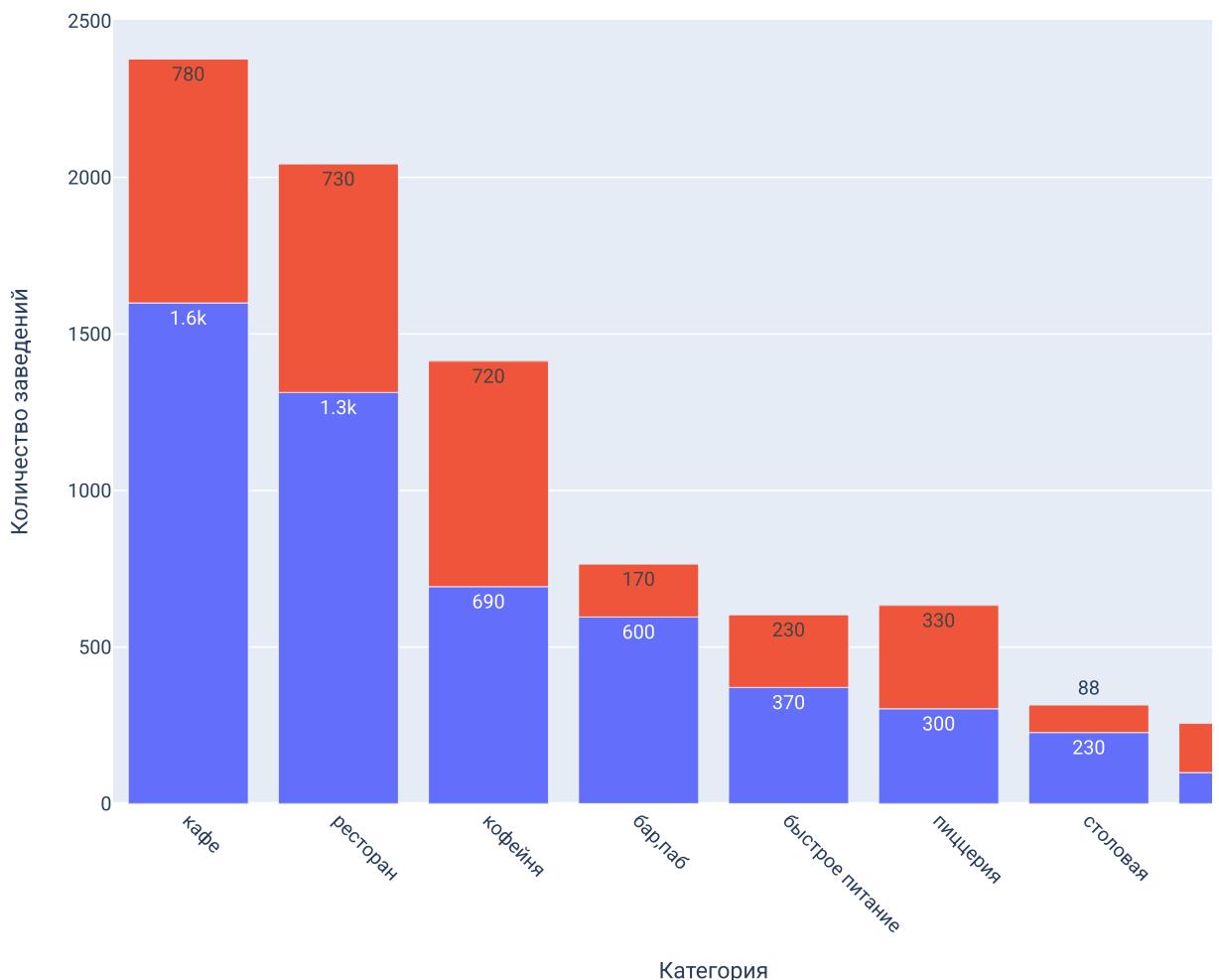
```
In [65]: # подготовим выборку для топа заведений по категориям
top_cat_chain = data.groupby(by=['new_chain', 'category']).count().sort_values(by='lat', ascending=False)

# подпишем колонки и выделим топ 5
top_cat_chain.columns = ['Тип', 'Категория', 'Количество заведений']
top_cat_chain_good = top_cat_chain.loc[top_cat_chain['Категория'] == 'кафе'].head(5).reset_index()
```

```
In [66]: # повторим для остальных категорий
for cat in ['бар,паб', 'булочная', 'быстрое питание', 'кофейня', 'пиццерия', 'ресторан', 'столовая']:
    top_cat = top_cat_chain.loc[top_cat_chain['Категория'] == cat].head(5).reset_index(drop=True)
    top_cat_chain_good = pd.concat([top_cat_chain_good, top_cat])
```

```
In [67]: # строим график
fig = px.bar(top_cat_chain_good.sort_values(by='Количество заведений', ascending=False), x='Категория',
             color='Тип', title='Соотношение в топ 5 заведениях по количеству точек сетевых и несетевых')
fig.update_xaxes(tickangle=45)
fig.update_layout(width=950, height=700)
fig.show();
```

Соотношение в топ 5 заведениях по количеству точек сетевых и несетевых заведений



- Большая часть в 61,9% заведений не являются сетевыми, тогда как доля сетевых составляет 38,1%. Около половины всех кофеен и пиццерий сетевые, тогда как среди ресторанов и кафе эта доля порядка 30%, самая высокая доля сетевых заведений в категории булочных. В данных возможно наличие ошибок в принадлежности заведений к типу

Проанализируем топ 15 заведений по количеству точек

```
In [68]: # посчитаем топы
top_name = data.groupby(by='name')['name'].count().sort_values(ascending=False).head(15)

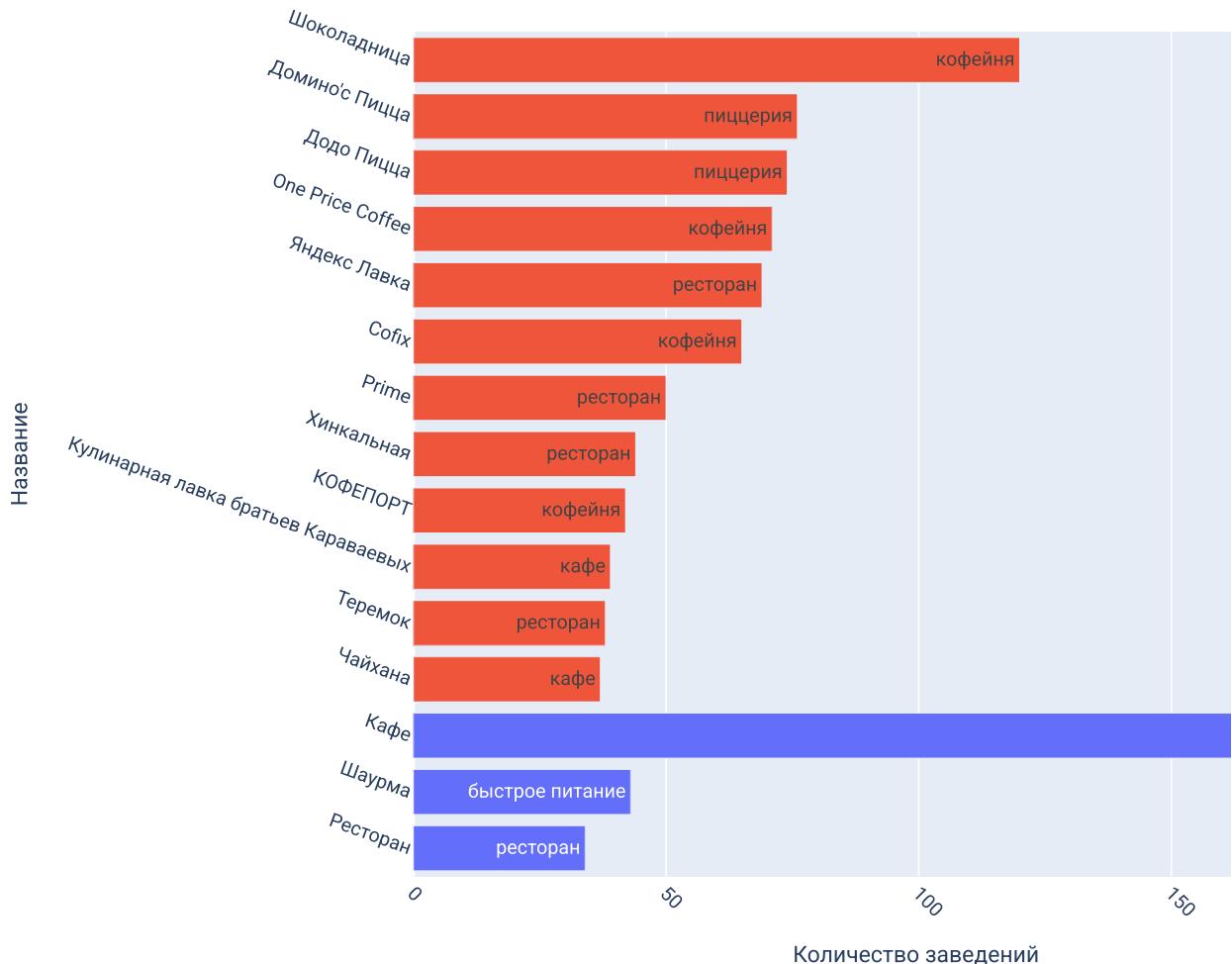
# выделим данные для дополнительной информации
top_name1 = data.query("name in @top_name.index")
top_name1 = top_name1.drop_duplicates(subset=['name'], keep='last')
top_name1 = top_name1.reset_index(drop=True)

# сформируем итоговую таблицу
top_name = pd.DataFrame(top_name)
top_name.columns = ['Количество']
top_name.reset_index(drop=False, inplace=True)
top_name.columns = ['name', 'Количество заведений']

# добавим информацию о категориях и типе заведения
top_name_all = pd.merge(top_name,top_name1, on=["name", "name"])
top_name['Категория'] = top_name_all['category']
top_name['Тип'] = top_name_all['new_chain']
top_name.columns = ['Название', 'Количество заведений', 'Категория', 'Тип']
```

```
In [69]: # построим график для наглядного отображения результатов
fig = px.bar(top_name.sort_values(by='Количество заведений', ascending=True), x='Количество заведений',
             color='Тип', title='Топ 15 заведений по количеству точек в разных категориях', text='Название')
fig.update_xaxes(tickangle=45)
fig.update_yaxes(tickangle=20)
fig.update_layout(width=1000, height=700)
fig.show();
```

Топ 15 заведений по количеству точек в разных категориях



- Подавляющее большинство заведений в топ 15 по количеству относятся к сетевому типу, лидирующую позицию среди них занимает "Шоколадница". Так же можно отметить что большое количество заведений относятся к категориям кофеен и пиццерий. Среди несетевых заведений сгруппировались мелкие заведения с одноименными названиями, что говорит о неплохой заполненности ими сегмента рынка

Проведем анализ категорий заведений по Административным округам

```
In [70]: # сделаем функцию правки значений district для графика
def cleaner(s):
    s = ''.join(s.split(' административный округ'))
    return s
cleaner('Северный административный округ')
```

Out[70]: 'Северный'

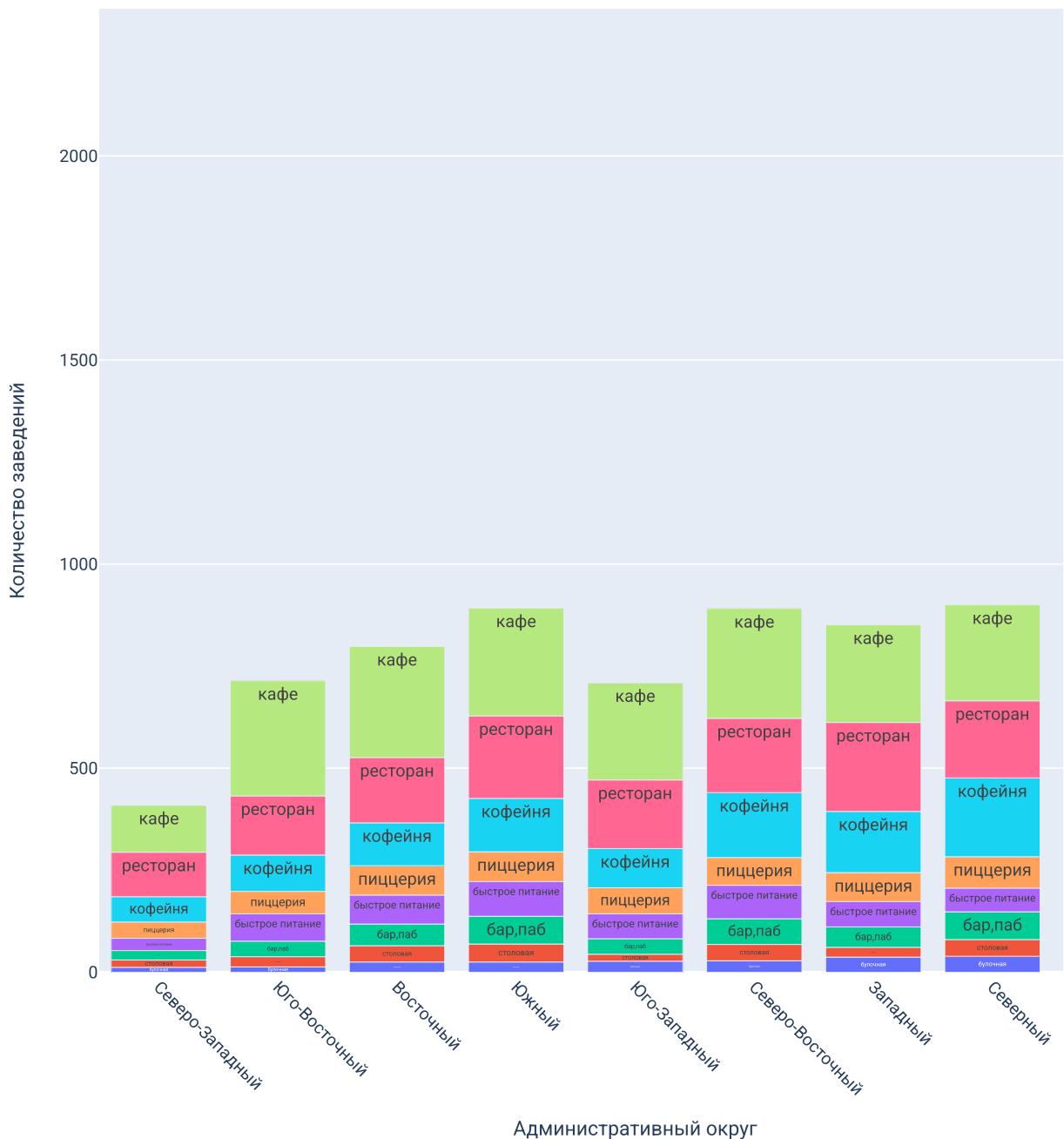
```
In [71]: # подготовим данные
data['district_good'] = data['district'].apply(cleaner)
```

```
district = data.groupby(by=['district_good', 'category'])['name'].count().reset_index()
district.columns = ['Административный округ', 'Категория', 'Количество заведений']
```

In [72]: # построим график для наглядного отображения результатов

```
fig = px.bar(district.sort_values(by='Количество заведений', ascending=True), x='Административный округ',
              color='Категория', title='Распределение заведений по Административным округам с учетом категорий',
              fig.update_xaxes(tickangle=45)
fig.update_layout(width=1000, height=900)
fig.show();
```

Распределение заведений по Административным округам с учетом категорий

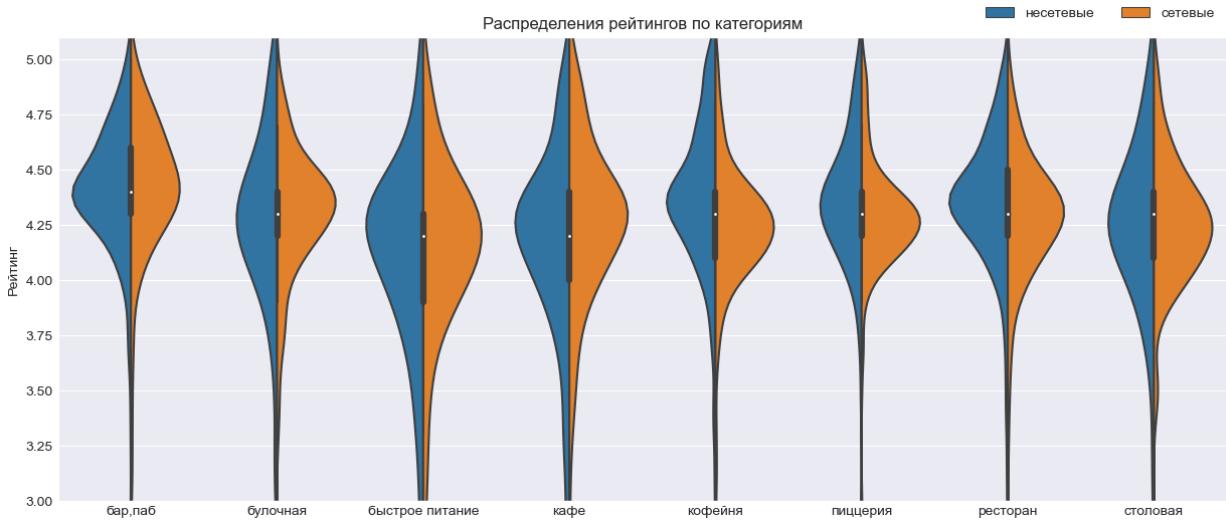


- Подавляющее большинство заведений в топ 15 по количеству относятся к сетевому типу, лидирующую позицию среди них занимает "Шоколадница". Так же можно отметить что большое количество заведений относятся к категориям кофеен и пиццерий. Среди несетевых заведений сгруппировались мелкие заведения с одноименными названиями, что говорит о неплохой заполненности ими сегмента рынка

Проведем анализ рейтингов заведений

```
In [73]: sns.set_style('darkgrid')
plt.figure(figsize=(15,6))
ax = sns.violinplot(y='rating', x ='category', data=data,hue='new_chain',split=True)
sns.move_legend(ax, "lower right", bbox_to_anchor=(1, 1.01), ncol=3, title=None, frameon=False)
plt.title('Распределения рейтингов по категориям')
plt.xlabel('')
plt.ylim(3,5.1)
plt.ylabel('Рейтинг')
plt.show();

display('СЕТЕВЫЕ',data.loc[data['new_chain']=='сетевые'].groupby(by='category')['rating'].median)
print('*****')
display('НЕСЕТЕВЫЕ',data.loc[data['new_chain']=='несетевые'].groupby(by='category')['rating'].me
```



```
'СЕТЕВЫЕ'
category
бар,паб      4.40
булочная    4.30
кафе        4.30
пиццерия   4.30
ресторан    4.30
столовая    4.25
быстрое питание 4.20
кофейня     4.20
Name: rating, dtype: float64
*****
'НЕСЕТЕВЫЕ'
category
бар,паб      4.4
кофейня     4.4
ресторан    4.4
булочная    4.3
пиццерия   4.3
столовая    4.3
быстрое питание 4.2
кафе        4.2
Name: rating, dtype: float64
```

```
In [74]: # выделим данные
top_cat_name_rating = data.pivot_table(index=['category','name'],values='rating',aggfunc=['count'])

# подпишем колонки, отсортируем и уберем дубликаты
top_cat_name_rating.columns = ['категория', 'название заведения', 'количество заведений', 'медианное значение']
top_cat_name_rating = top_cat_name_rating.sort_values(by=['количество заведений','категория'], ascending=False)
top_cat_name_rating = top_cat_name_rating.drop_duplicates(subset=['категория', 'название заведен
```

```
In [75]: # соберем топ 5 по количеству точек для каждой категории
top_cat_name_rating_good = pd.DataFrame()
for cat in ['кафе','бар,паб', 'булочная', 'быстрое питание', 'кофейня', 'пиццерия', 'ресторан',
            top_cat_name_rating_ = top_cat_name_rating.loc[top_cat_name_rating['категория'] == cat].sort_index()
            top_cat_name_rating_good = pd.concat([top_cat_name_rating_good,top_cat_name_rating_])

top_cat_name_rating_good
```

```
In [76]: # поправим одноименные названия для графика
top_cat_name_rating_good.loc[(top_cat_name_rating_good['категория'] == 'столовая')\
```

```

&(top_cat_name_rating_good['название заведения'] == 'Кафе'), 'название заведения']

top_cat_name_rating_good.loc[(top_cat_name_rating_good['категория'] == 'быстрое питание')\n
                             &(top_cat_name_rating_good['название заведения'] == 'Кафе'), 'название заведения']

top_cat_name_rating_good = top_cat_name_rating_good.sort_values(by='медианное количество рейтинга')

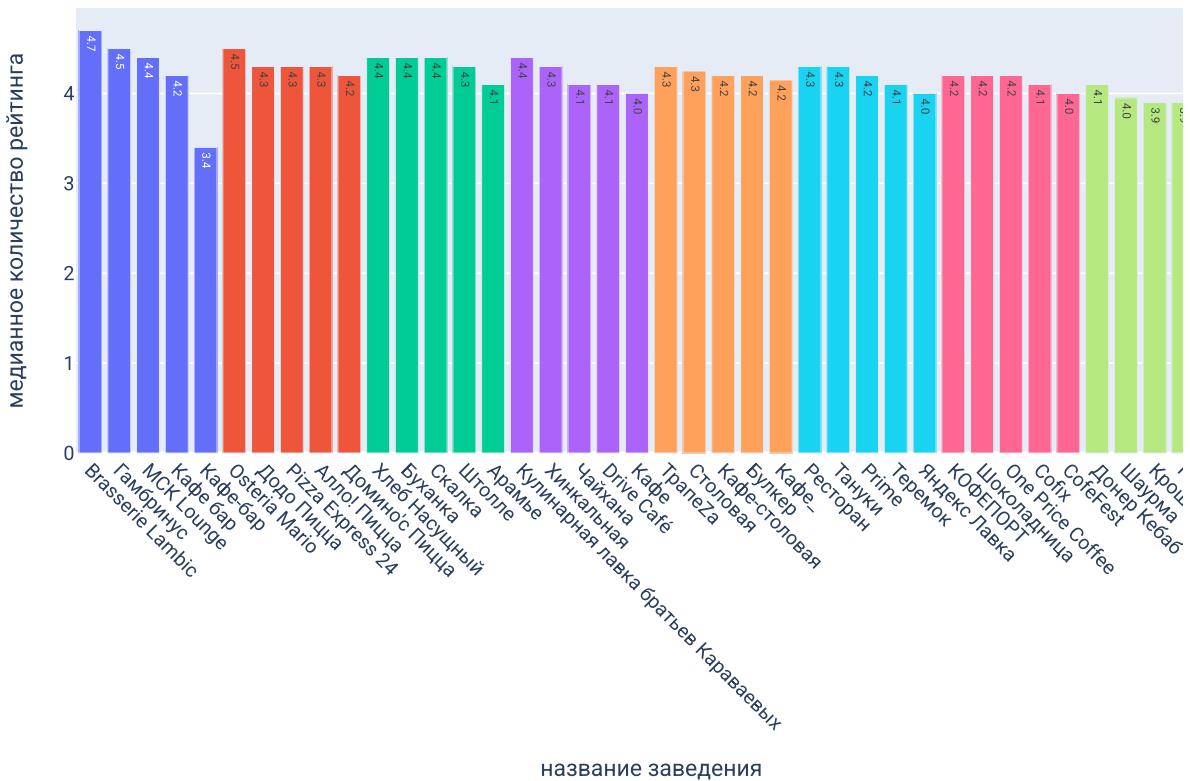
```

```

In [77]: # строим график
fig = px.bar(top_cat_name_rating_good, x='название заведения', y='медианное количество рейтинга'
             color='категория', title='Медианные значения рейтинга в топ 5-ти заведений по количеству точек',
             text_auto='.2s')
fig.update_xaxes(tickangle=45)
fig.update_layout(width=950, height=580)
fig.show()

```

Медианные значения рейтинга в топ 5-ти заведений по количеству точек



- По анализу распределений рейтингов мы видим что среднее его значение расположилось между 4 и 4.4 по 5 бальной шкале, причем сетевые и не сетевые бары вверху рейтинга. Так же можно заметить что распределения в категориях кофейни и пиццерии имеют различия: сетевые заведения здесь в среднем получают более стабильные оценки, однако более высокие мы имеем в несетевых что может говорить о более высоком качестве обслуживания в несетевых заведениях.

```

In [78]: # посчитаем средний рейтинг для каждого округа
rating_df = data.groupby('district', as_index=False)[['rating']].agg('median')
display(rating_df)

```

	district	rating
0	Восточный административный округ	4.3
1	Западный административный округ	4.3
2	Северный административный округ	4.3
3	Северо-Восточный административный округ	4.2
4	Северо-Западный административный округ	4.3
5	Центральный административный округ	4.4
6	Юго-Восточный административный округ	4.2
7	Юго-Западный административный округ	4.3
8	Южный административный округ	4.3

In [79]: # отобразим результат на карте москвы

```
import json
from folium import Map, Choropleth

try:
    state_geo = 'F:\\обучение\\projects\\moscow_restaurants\\admin_level_geomap.geojson'
except:
    state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'
state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'

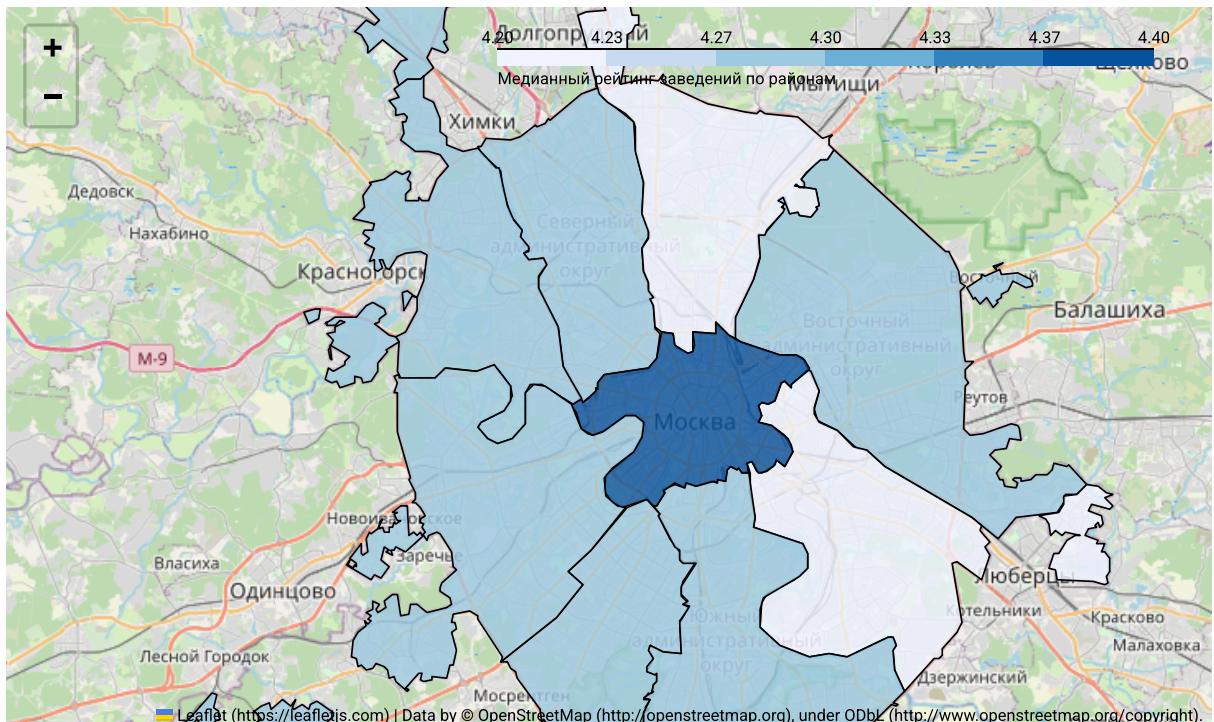
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаём карту Москвы
m = folium.Map(location=[moscow_lat, moscow_lng])
# выводим карту

# создаём хороплет с помощью конструктора Choropleth и добавляем его на карту
Choropleth(
    geo_data=state_geo,
    data=rating_df,
    columns=['district', 'rating'],
    key_on='feature.name',
    fill_color='Blues',
    fill_opacity=0.8,
    legend_name='Медианный рейтинг заведений по районам',
).add_to(m)

# выводим карту
m
```

Out[79]:



- Самый высокий медианный рейтинг равен 4.4 балла в центральном административном округе, что может говорить о более высоком уровне и качестве обслуживания заведений в этом округе. Самый низкий рейтинг в северо-восточном и юго-восточном округах. В целом разброс рейтингов по округам составляет около 4 процентов

Посчитаем расстояния от центра до каждого объекта

In [80]:

```
import math
# напишем функцию

point = [55.751244, 37.618423]

def distance(origin):
    lat1, lon1 = origin['lat'], origin['lng']
    point = [55.751244, 37.618423]
    lat2, lon2 = point
    radius = 6371 # km

    dlat = math.radians(lat2-lat1)
    dlon = math.radians(lon2-lon1)
    a = math.sin(dlat/2) * math.sin(dlat/2) + math.cos(math.radians(lat1)) \
        * math.cos(math.radians(lat2)) * math.sin(dlon/2) * math.sin(dlon/2)
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1-a))
    d = radius * c

    return round(d,3)
```

In [81]:

```
# применим функцию и создадим два новых столбца с расстояниями для дальнейшего анализа
data['distance'] = data.apply(distance, axis=1)
data['distance_km'] = round(data['distance'], 0)
```

Отобразим все заведения на катре москвы

In [82]:

```
# подготовим иконки для разных категорий
icon_cafe = 'https://img.icons8.com/external-victoruler-solid-victoruler/64/null/external-cafe-b
icon_coffee_house = 'https://img.icons8.com/ios-filled/50/null/cafe-building.png'
icone_pizza = 'https://img.icons8.com/external-kiranshastry-solid-kiranshastry/64/null/external-
icone_bar = 'https://img.icons8.com/pastel-glyph/64/null/cocktail--v2.png'
icote_fast_food = 'https://img.icons8.com/external-nawicon-glyph-nawicon/64/null/external-fast-f
icote_backery = 'https://img.icons8.com/ios-filled/50/null/bakery.png'
icone_dinning_room = 'https://img.icons8.com/external-goofy-solid-kerismaker/96/null/external-Di
icone_restaurant = 'https://img.icons8.com/sf-black-filled/64/null/restaurant-building.png'

# соберем для нанесения на карту
icons = {'кафе':icon_cafe, 'кофейня':icon_coffee_house, 'пиццерия':icone_pizza, 'бар, паб':icone_ba
'быстрое питание':icote_fast_food, 'булочная':icote_backery, 'столовая':icone_dinning_rac
```

In [83]:

```
# импортируем собственные иконки
from folium.features import CustomIcon
from folium.plugins import MarkerCluster
# импортируем карту и маркер
from folium import Map, Marker
# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаём карту Москвы
m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
# создаём пустой кластер, добавляем его на карту
marker_cluster = MarkerCluster().add_to(m)

# строим карту
for key, value in icons.items():

    def create_clusters(row):
        # сохраняем URL-адрес изображения со значком торгового центра с icons8,
        # это путь к файлу на сервере icons8
        icon_url = value

        # создаём объект с собственной иконкой размером 30x30
```

```

icon = CustomIcon(icon_url, icon_size=(35, 35))

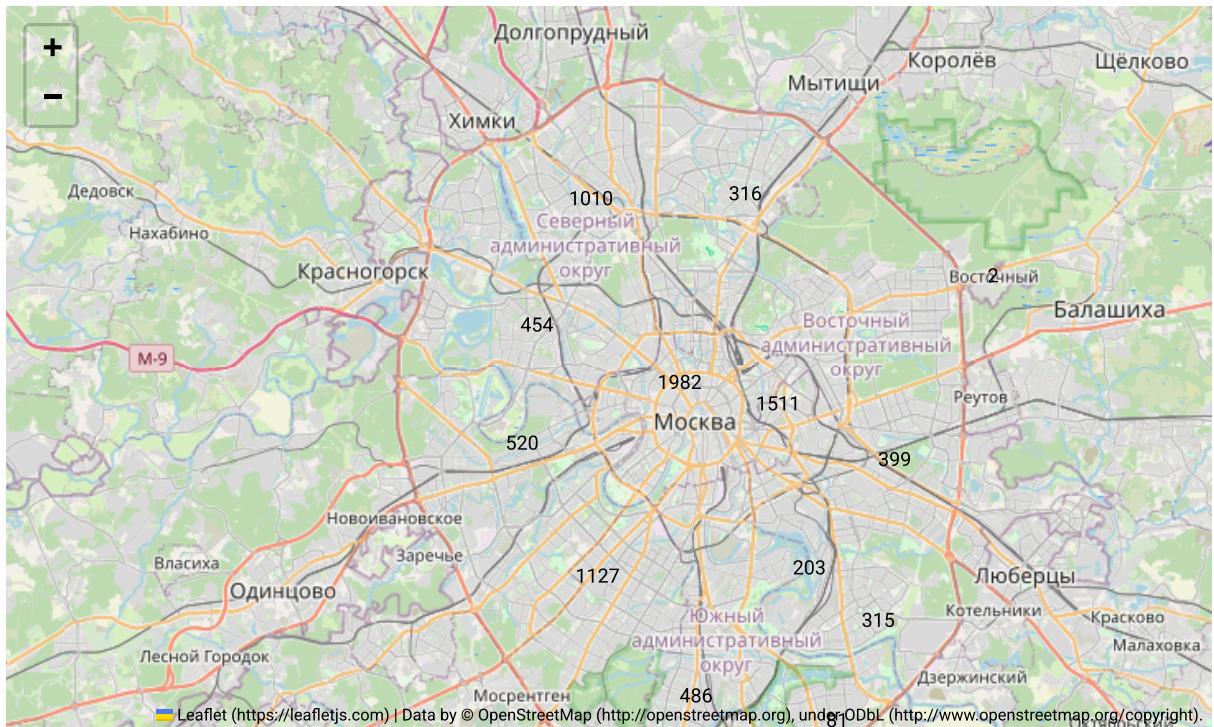
# создаём маркер с иконкой icon и добавляем его в кластер
Marker(
    [row['lat'], row['lng']],
    popup=f'{row['name']} {row['category']} рейтинг: {row['rating']} режим работы: {row['work_time']}\nрасстояние до центра: {row['distance']} км",
    icon=icon,
).add_to(marker_cluster)

# применяем функцию для создания кластеров к каждой строке датафрейма
data.loc[data['category']==key].apply(create_clusters, axis=1)

# выводим карту
m

```

Out[83]:



Найдем топ 15 улиц по количеству заведений

```

In [84]: # выделим данные
top_street_all = data.groupby(by=['street']).count().sort_values(by='lat', ascending=False)[['lat']]
top_street_all_m = data.query("street in @top_street_all.index")

In [85]: # выделим топ
# подготовим выборку для топа заведений по категориям
top_street_all = top_street_all_m.groupby(by=['street', 'category']).count().sort_values(by='lat', ascending=False)

# подпишем колонки
top_street_all.columns = ['Улица', 'Категория', 'Количество заведений']
top_cat_street_good = top_street_all.loc[top_street_all['Категория'] == 'кафе'].reset_index(drop=True)

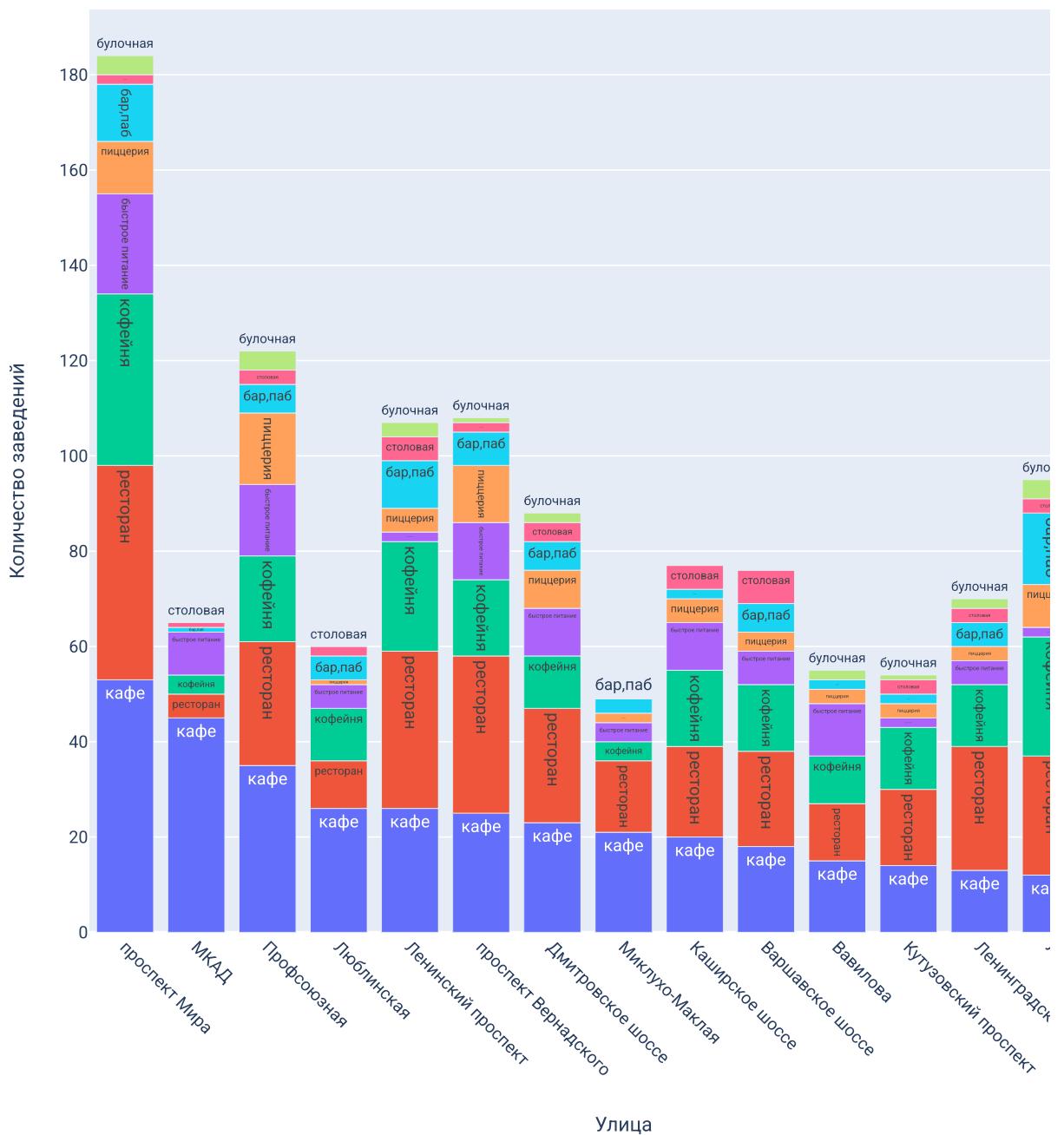
In [86]: # повторим для остальных категорий
for cat in ['бар,паб', 'ресторан', 'булочная', 'быстрое питание', 'кофейня', 'пиццерия', 'столовая']:
    top_cat_street_ = top_street_all.loc[top_street_all['Категория'] == cat].reset_index(drop=True)
    top_cat_street_good = pd.concat([top_cat_street_good, top_cat_street_])

In [87]: # построим график для наглядного отображения результатов

fig = px.bar( top_cat_street_good.sort_values(by='Количество заведений', ascending=False), x='Улица',
              color='Категория', title='Распределение заведений по топ 15 улицам с учетом категорий')
fig.update_xaxes(tickangle=45)
fig.update_layout(width=1000, height=900)
fig.show();

```

Распределение заведений по топ 15 улицам с учетом категорий



- Самое большое количество заведений на проспекте Мира, далее идет Профсоюзная улица и следом еще два проспекта: Ленинский и Вернадского. Пятницкая и Миклухо-Маклая расположились на последнем месте из топ 15 улиц. Большое количество заведений обусловлено оживленным трафиком, протяженностью и инфраструктурой, облегчающей прохождение этого трафика по ним

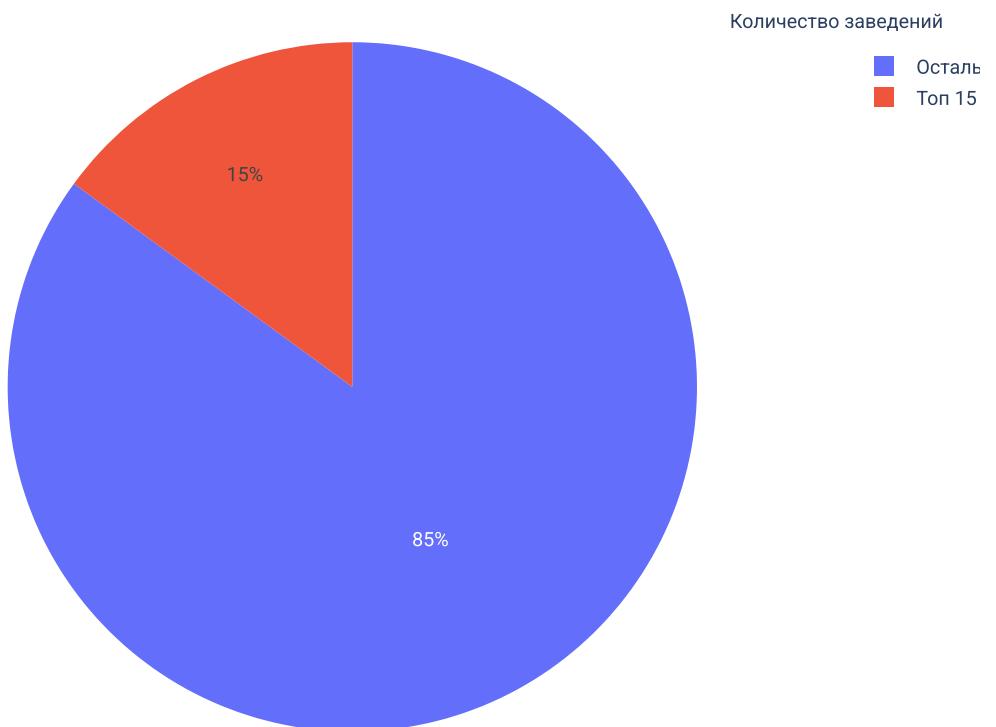
```
In [88]: # подготовим данные для отображения долей
share_top_street = pd.DataFrame()
share_top_street.index=['Топ 15 улиц', 'Остальные']
#share_top_street.columns = ['Количество заведений']
share_top_street['Количество заведений'] = [sum(top_cat_street_good['Количество заведений']),len(top_cat_street_good)-sum(top_cat_street_good['Количество заведений'])]
share_top_street
```

Количество заведений	
Топ 15 улиц	1258
Остальные	7148

```
In [89]: # построим график с долей заведений, которые приходятся на топ 15 улиц

# строим диаграмму с сегментами
fig = go.Figure(data=[go.Pie(labels=share_top_street.index, # указываем значения, которые появляются на сегментах
                               values=share_top_street['Количество заведений'], # указываем данные,
                               )]) # добавляем аргумент, который выделит сегмент-лидер на графике
fig.update_layout(title='Доля заведений на топ 15 улицах от общего количества', # указываем заголовок
                  width=800, # указываем размеры графика
                  height=600,
                  annotations=[dict(x=1.12, # вручную настраиваем аннотацию легенды
                                    y=1.05,
                                    text='Количество заведений',
                                    showarrow=False)])
fig.show() # выводим график
```

Доля заведений на топ 15 улицах от общего количества



- На 15 самых популярных улицах расположилось 15 процентов всех заведений города, что является довольно большим показателем

```
In [90]: # посмотрим на особенности их размещения на карте
from folium.features import CustomIcon
from folium.plugins import MarkerCluster
# импортируем карту и маркер
from folium import Map, Marker
# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаем карту Москвы
m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
# создаем пустой кластер, добавляем его на карту
marker_cluster = MarkerCluster().add_to(m)

# строим карту
for key, value in icons.items():

    def create_clusters(row):
        # сохраняем URL-адрес изображения со значком торгового центра с icons8,
        # это путь к файлу на сервере icons8
        icon_url = value
```

```

# создаём объект с собственной иконкой размером 30x30
icon = CustomIcon(icon_url, icon_size=(35, 35))

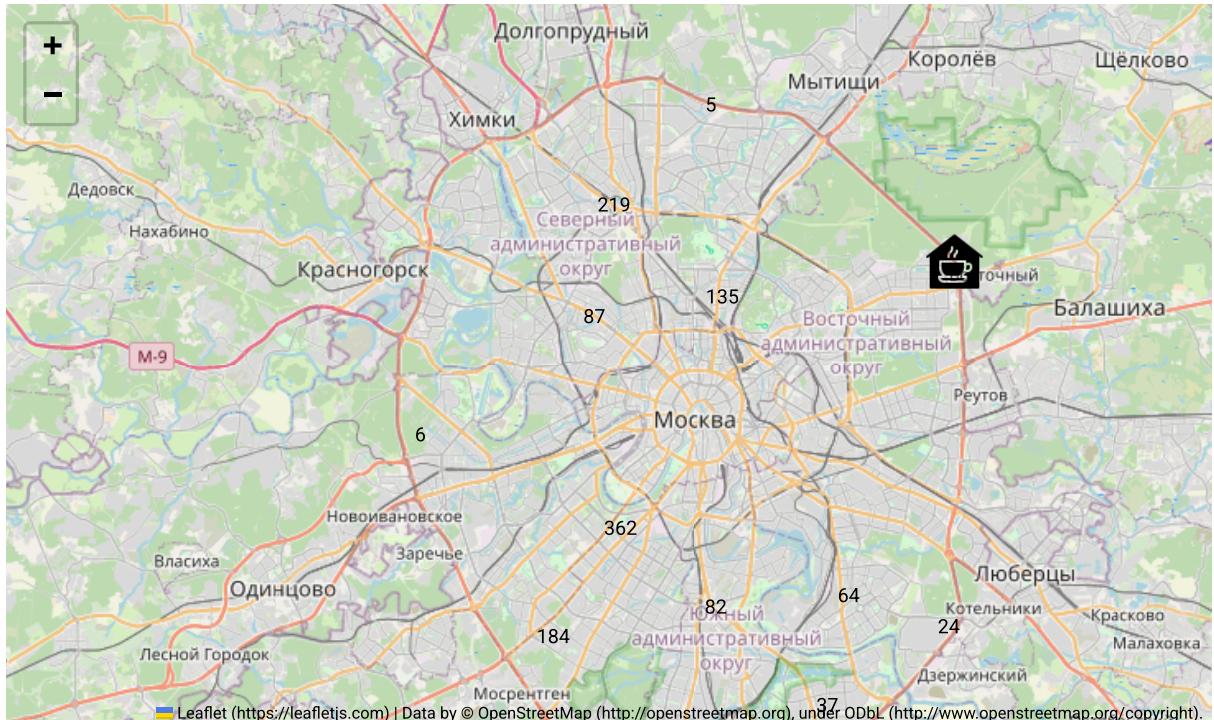
# создаём маркер с иконкой icon и добавляем его в кластер
Marker(
    [row['lat'], row['lng']],
    popup=f'{row['name']} {row['category']} рейтинг: {row['rating']} режим работы: {row['work_time']}\nрасстояние до центра: {row['distance']} км",
    icon=icon,
).add_to(marker_cluster)

# применяем функцию для создания кластеров к каждой строке датафрейма
top_street_all_m.loc[top_street_all_m['category']==key].apply(create_clusters, axis=1)

```

```
# выводим карту
m
```

Out[90]:



- На карте видно, что улицы из топ 15 по количеству заведений практически не направлены, а северо-восточном и восточном направлениях. Также можно обратить внимание на особенности размещения заведений на этих улицах, центрами их притяжения здесь становятся объекты которые, генерируют трафик: это парки, станции метро, торговые центры, муниципальные учреждения и прочее.

Посмотрим на самые непопулярные улицы

```
In [91]: print(f'Все заведения размещены на {len(data.groupby(by=['street']).count())} улицах')
```

Все заведения размещены на 1448 улицах

```
In [92]: # выделим данные
tail_street_all = data.groupby(by=['street']).count().sort_values(by='lat', ascending=False)[['lat']]
tail_street_all = tail_street_all.loc[tail_street_all['lat']==1]
print(f'Количество улиц с 1 заведением равно: {len(tail_street_all)} из общих 1448')
```

Количество улиц с 1 заведением равно: 458 из общих 1448

```
In [93]: # посмотрим на названия этих улиц
tail_street_all.loc[tail_street_all['lat']==1]['street'][:25]
```

```

Out[93]: 990          Тучковская
991          сад Эрмитаж
992          Турчанинов переулок
993          сквер имени М.И. Калинина
994          Якиманский переулок
995          Шкулёва
996          Черёмушкинский проезд
997          Ставропольский проезд
998          ул. Профсоюзная
999          Таганский парк культуры и отдыха
1000         проспект Лихачёва
1001         Шкулёва 4
1002         проезд Ольминского
1003         Сумской проезд
1004         ул. Ярославская
1005         Стрелецкая
1006         проезд Одоевского
1007         Светлогорский проезд
1008         Фадеева
1009         Яковоапостольский переулок
1010         Сытинский тупик
1011         Стромынский переулок
1012         Угличская
1013         Чечулина
1014         Чистова
Name: street, dtype: object

```

- Количество улиц с 1 заведением равно: 458 из общих 1448, что составляет 5% от заведений и практически 30% от улиц. Судя по названиям, непопулярные улицы это переулки, тупики и проезды с низким трафиком

Проведем анализ среднего чека по округам и в зависимости от удаленности из центра

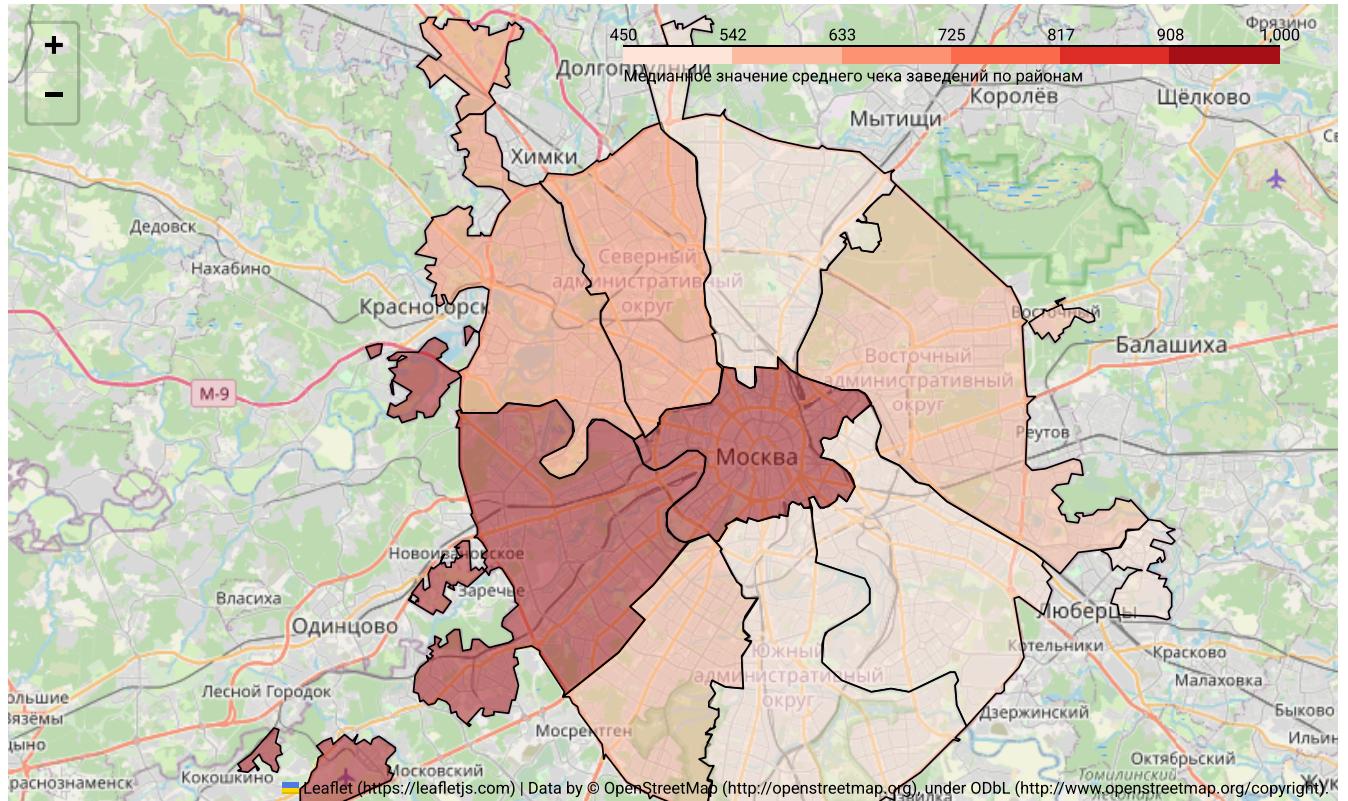
Анализ по округам

```

In [94]: # подготовим данные
median_bill_distr = data.groupby(by='district')['middle_avg_bill'].median().reset_index()
median_bill_distr.columns = ['Округ', 'Средний медианный счет']
display(median_bill_distr.sort_values(by='Средний медианный счет', ascending=False))

```

	Округ	Средний медианный счет
1	Западный административный округ	1000.0
5	Центральный административный округ	1000.0
4	Северо-Западный административный округ	700.0
2	Северный административный округ	650.0
7	Юго-Западный административный округ	600.0
0	Восточный административный округ	575.0
3	Северо-Восточный административный округ	500.0
8	Южный административный округ	500.0
6	Юго-Восточный административный округ	450.0



- Среднее медианное значение среднего счета заведений по районам расположилось в пределах 450 и 1000 рублей. Центральный и Западный округа имеют самое высокое значение в 1000 рублей, что связано с близостью к центру в первом случае и наличием оживленных проспектов, ведущих в западное направление во втором

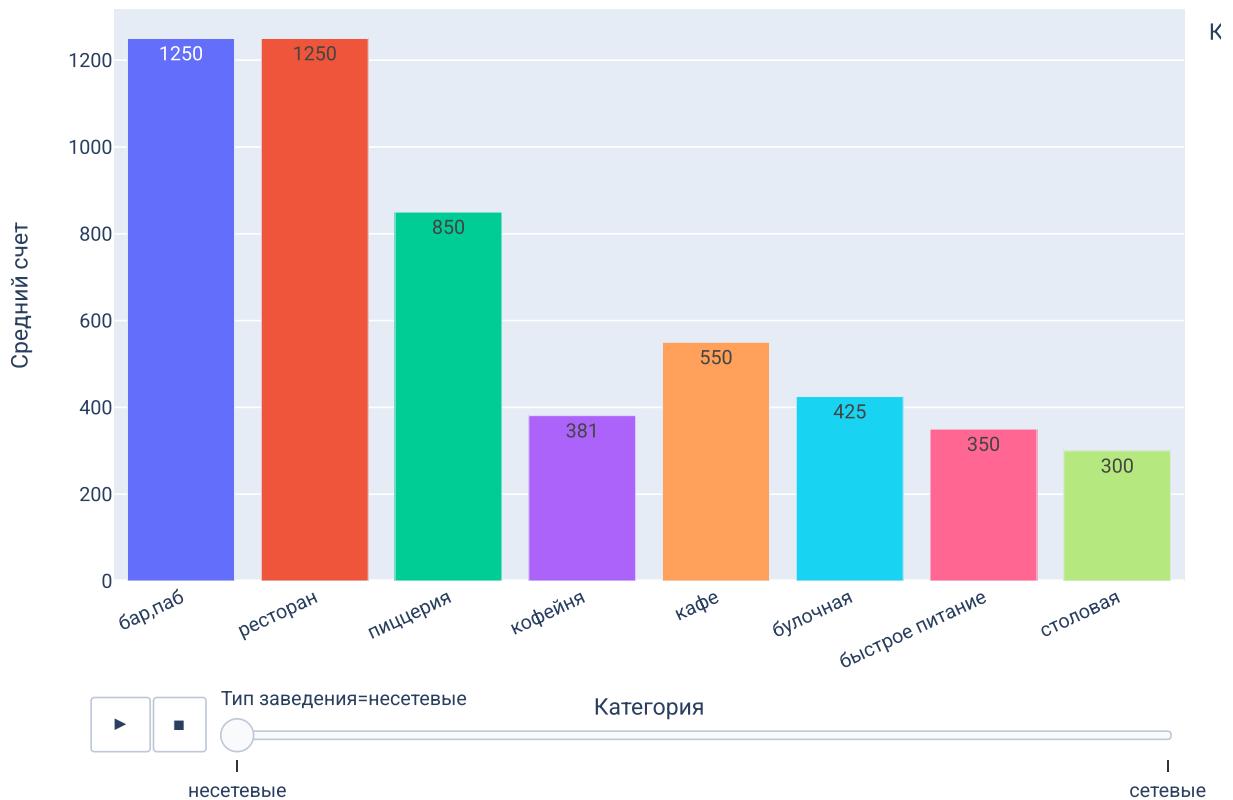
Анализ среднего чека по категориям заведений

```
In [96]: # подготовим данные
category_bill = data.groupby(by=['new_chain', 'category'])['middle_avg_bill'].median().sort_values()
category_bill.columns = ['Тип заведения', 'Категория', 'Средний счет']
category_bill
```

	Тип заведения	Категория	Средний счет
0	несетевые	бар,паб	1250.0
1	несетевые	ресторан	1250.0
2	сетевые	бар,паб	1060.0
3	сетевые	ресторан	1000.0
4	несетевые	пиццерия	850.0
5	сетевые	кофейня	575.0
6	несетевые	кафе	550.0
7	сетевые	булочная	550.0
8	сетевые	кафе	550.0
9	сетевые	пиццерия	500.0
10	несетевые	булочная	425.0
11	сетевые	быстрое питание	425.0
12	несетевые	кофейня	381.0
13	несетевые	быстрое питание	350.0
14	несетевые	столовая	300.0
15	сетевые	столовая	300.0

```
In [97]: ##### построим график
fig = px.bar( category_bill, x='Категория', y='Средний счет', \
    color='Категория', animation_frame='Тип заведения', title='Распределение среднего счета по категориям')
fig.update_xaxes(tickangle=-25)
fig.update_layout(width=900, height=600)
fig.show();
```

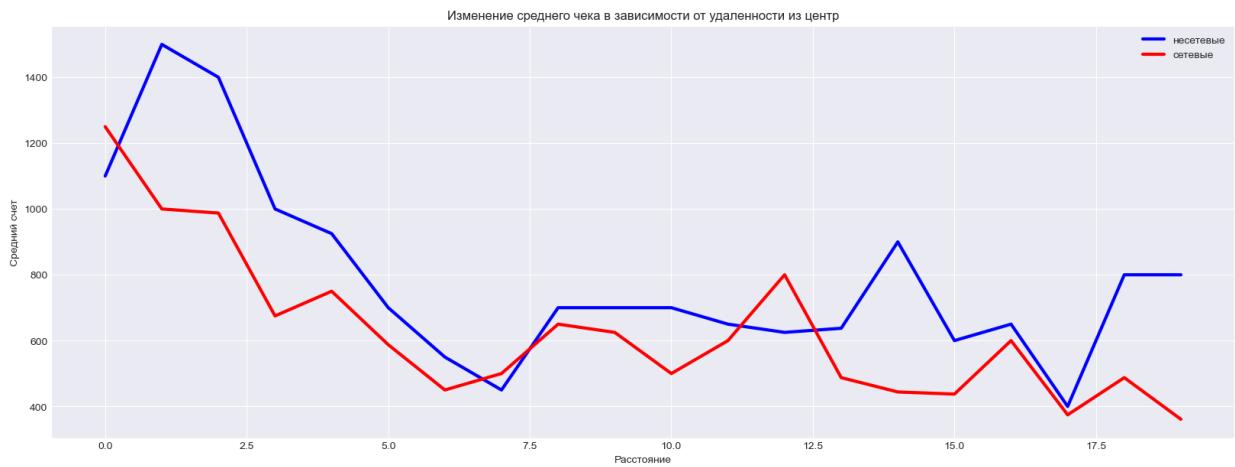
Распределение среднего счета заведений по категориям с учетом его типа



- Самый высокий средний счет имеют бары и рестораны, причем сетевые заведения данной категории в среднем имеют на 250 рублей меньший счет. Несетевые кофейни и булочные имеют меньший счет в 381 и 425 рублей соответственно напротив 575 и 550 у сетевых. Столовые и кафе имеют одинаковый средний счет в сетевых и несетевых заведениях. Самыми недорогими заведениями общественного питания являются столовые со средним счетом в 300 рублей

Анализ зависимости удаленности из центра со значением среднего счета

```
In [98]: # Построим график изменения среднего чека в зависимости от удаленности из центра
ax = data.loc[data['new_chain']=='несетевые'].groupby(by='distance_km')['middle_avg_bill'].median()
data.loc[data['new_chain']=='сетевые'].groupby(by='distance_km')['middle_avg_bill'].median().plot()
plt.legend(labels=['несетевые', 'сетевые'])
plt.title('Изменение среднего чека в зависимости от удаленности из центра')
plt.xlabel('Расстояние')
plt.ylabel('Средний счет')
plt.show()
```

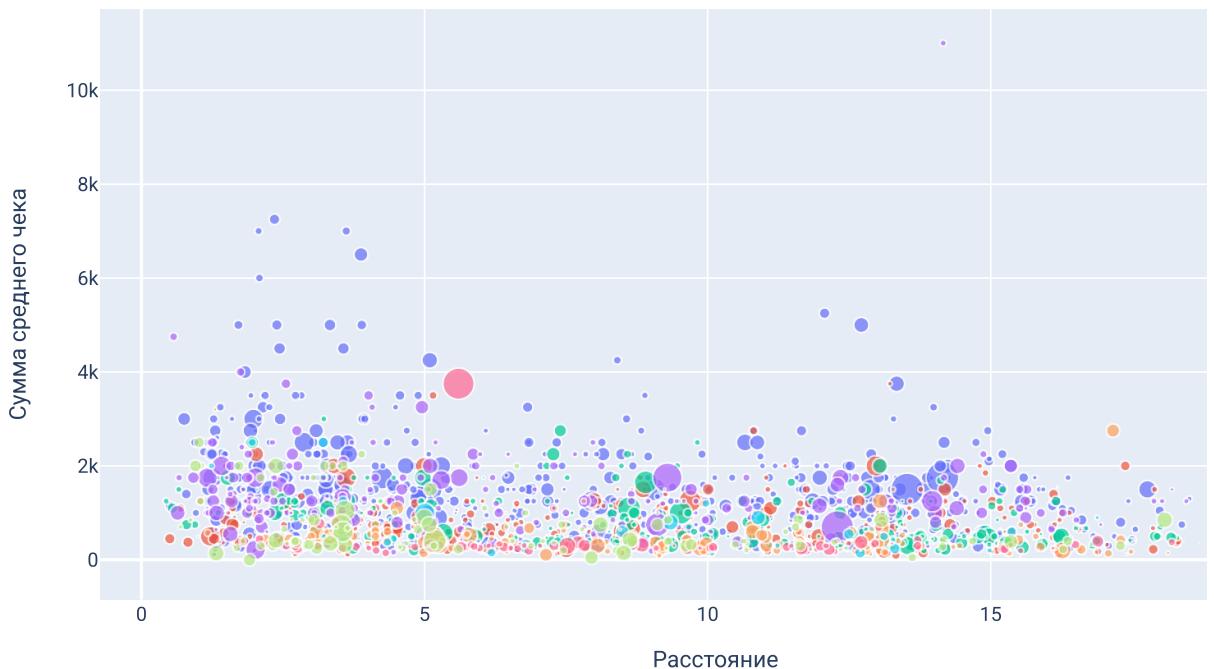


- На графике видно снижение среднего счета при удалении из центра на расстояние порядка 6-7 километров, далее значение среднего счета стабилизируется, причем поведение сетевых и несетевых заведений подобно. Так же мы можем увидеть превышение среднего чека несетевых заведений над сетевыми в пределах 6 км.

```
In [99]: # построим график для более подробно анализа зависимости счета от рассеяния по категориям
fig = px.scatter(data.loc[(data['middle_avg_bill']!=35000)&(~data['seats'].isna())], x='distance',
                  color='category', size='seats')

fig.update_layout(title='Зависимость среднего чека от удаленности из центра по категориям',
                  xaxis_title='Расстояние', showlegend=True)
fig.update_yaxes(title='Сумма среднего чека')
fig.show() # выводим график
```

Зависимость среднего чека от удаленности из центра по категориям

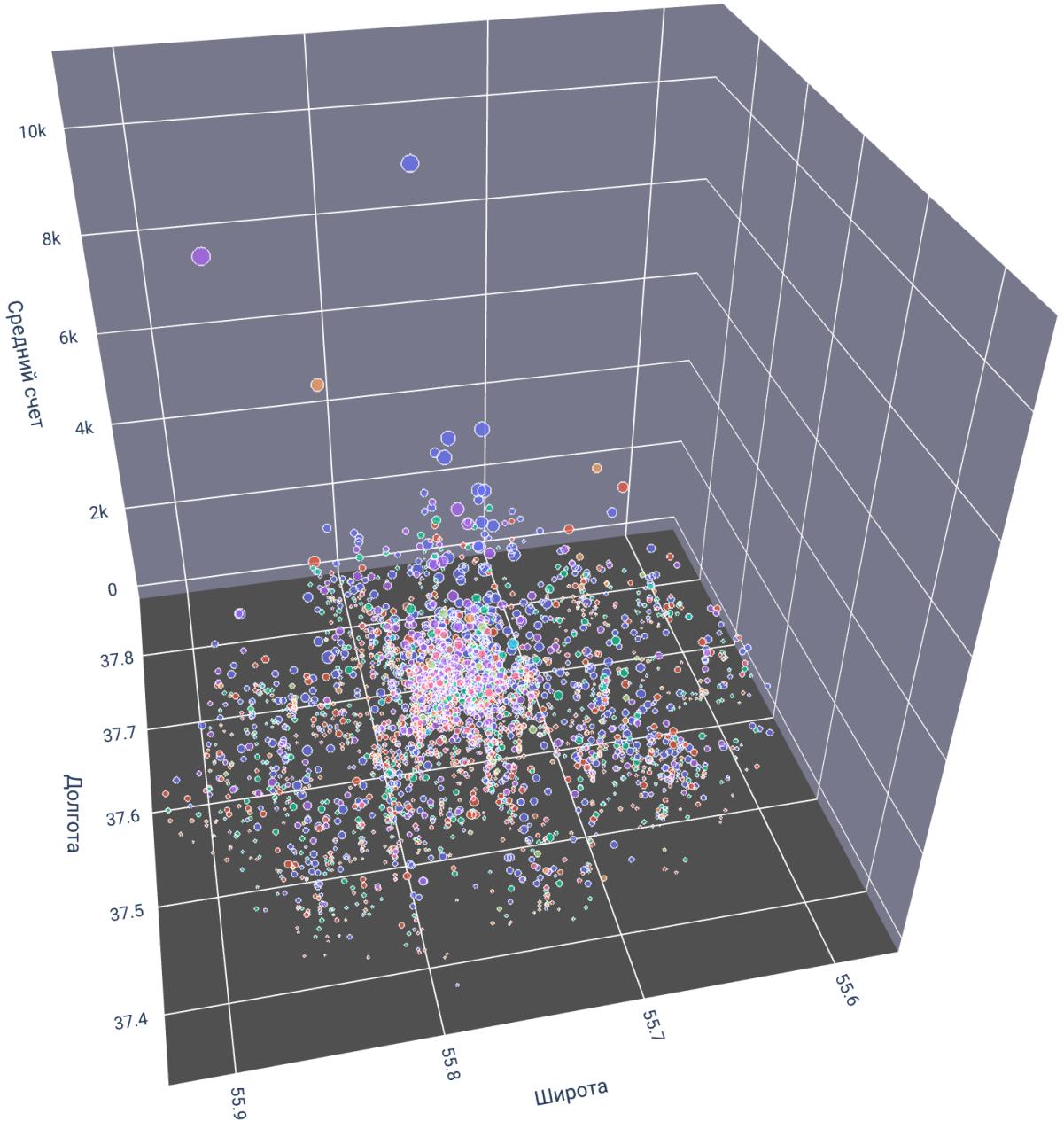


- При глубоком анализе мы увидели что зависимость в уменьшении счета при увеличении расстояния от центра не во всех категориях подобна, так она более ярко выражена в ресторанах, барах и кофейнях, тогда как средний счет в заведениях быстрого питания и сотовых практически не меняется

```
In [100...]: # перенесем график в 3 трехмерное пространство для более детального изучения изменений счета от
fig = px.scatter_3d(data.loc[(data['middle_avg_bill']!=35000)&(~data['middle_avg_bill'].isna())])
```

```
    color='category',color_continuous_scale = 'ylorrd',hover_name='name' , size='middle_
fig.update_layout(scene = dict(
                    xaxis_title='Долгота',
                    yaxis_title='Широта',
                    zaxis_title='Средний счет',
                    xaxis = dict(
                        backgroundcolor="rgb(120, 120, 140)",
                        gridcolor="white",
                        showbackground=True,
                        zerolinecolor="white",),
                    yaxis = dict(
                        backgroundcolor="rgb(120, 120,140)",
                        gridcolor="white",
                        showbackground=True,
                        zerolinecolor="white"),
                    zaxis = dict(
                        backgroundcolor="rgb(80, 80,80)",
                        gridcolor="white",
                        showbackground=True,
                        zerolinecolor="white",)),
                    width=1000,
                    height=900,
                    margin=dict(r=20, b=10, l=10, t=10),showlegend=True
                )
fig.update_layout(
    title={
        'text': "Зависимость среднего чека от удаленности из центра по категориям",
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'})
fig.show()
```

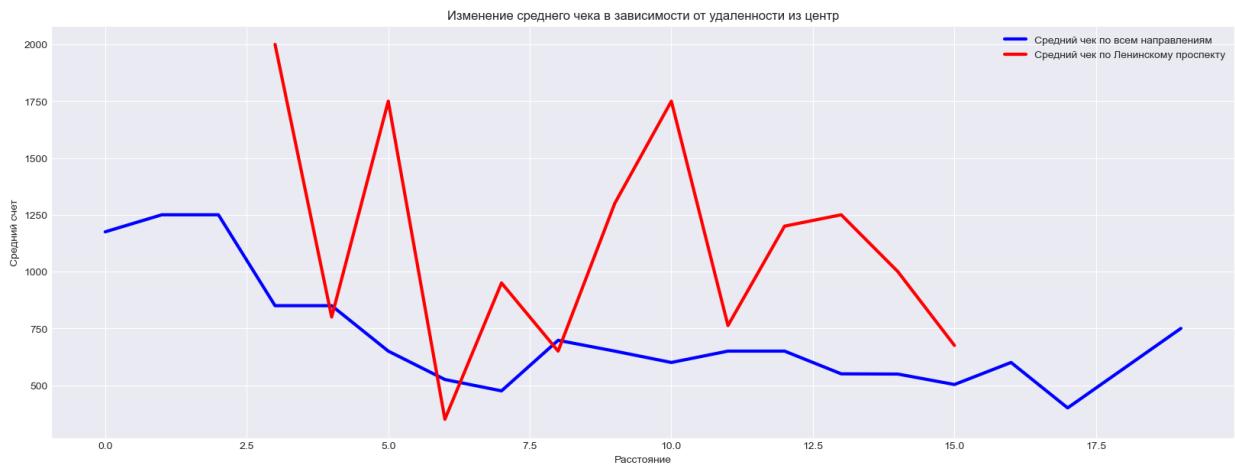
Зависимость среднего чека от удаленности из центра по категориям



- Проведя анализ 3д модели распределения заведений в зависимости от координат и среднего чека мы обнаружили конусообразное распределение с возвышением среднего счета в центре и тем самым подтвердили наши выводы о снижении среднего чека при удалении из центра в ресторанах, барах и кофейнях и практически полном его отсутствии в заведениях быстрого питания и сотовых. Также можно отметить неравномерность снижения среднего чека от центра к окраинам, здесь мы видим как заведения расходятся в форме лучей, что повторяет направление основных шоссе и проспектов, по ним снижение идет более плавно и даже далеко от центра встречаются высокие показатели среднего чека

In [101]:

```
# Проверим нашу теорию, построим график
ax = data.groupby(by='distance_km')['middle_avg_bill'].median().plot(figsize=(20,7), grid=True, color='black')
data.loc[(data['street']=='Ленинский проспект')].groupby(by='distance_km')['middle_avg_bill'].median().plot(ax=ax, color='red')
plt.legend(labels=['Средний чек по всем направлениям', 'Средний чек по Ленинскому проспекту'])
plt.title('Изменение среднего чека в зависимости от удаленности из центра')
plt.xlabel('Расстояние')
plt.ylabel('Средний счет')
plt.show()
```



Анализ доли круглосуточных заведений по округам и по категориям

Анализ по округам

```
In [102...]: # подготовим данные
share_is24_distr = data.groupby(by='district')['is_24/7'].mean().reset_index()
share_is24_distr.columns = ['Округ', 'Доля круглосуточных заведений']

display(share_is24_distr.sort_values(by='Доля круглосуточных заведений', ascending=False))
```

	Округ	Доля круглосуточных заведений
6	Юго-Восточный административный округ	0.147854
0	Восточный административный округ	0.134163
4	Северо-Западный административный округ	0.116531
7	Юго-Западный административный округ	0.112308
1	Западный административный округ	0.091720
8	Южный административный округ	0.090689
3	Северо-Восточный административный округ	0.090361
2	Северный административный округ	0.083924
5	Центральный административный округ	0.059249

```
In [103...]: # отобразим результат на карте москвы
import json
from folium import Map, Choropleth

try:
    state_geo = 'F:\\обучение\\projects\\moscow_restaurants\\admin_level_geomap.geojson'
except:
    state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'

# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

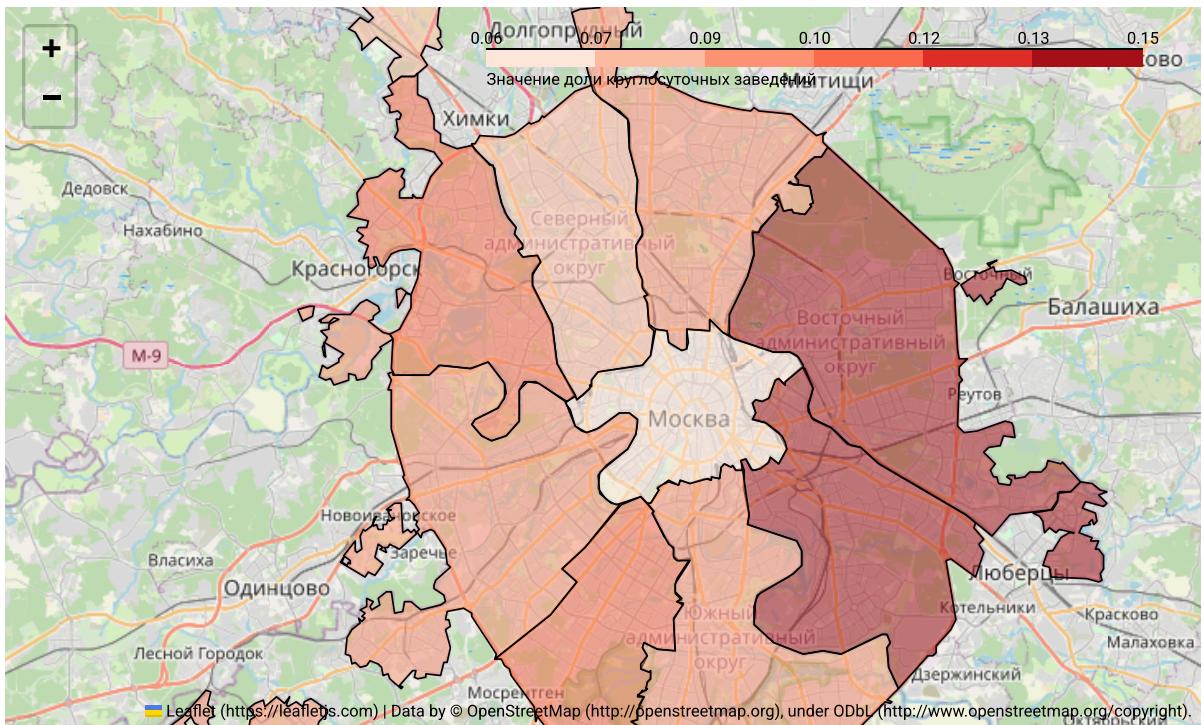
# создаём карту Москвы
m = folium.Map(location=[moscow_lat, moscow_lng], zoom_start=9.9)
# выводим карту

# создаём хороплет с помощью конструктора Choropleth и добавляем его на карту
Choropleth(
    geo_data=state_geo,
    data=share_is24_distr,
    columns=['Округ', 'Доля круглосуточных заведений'],
    key_on='feature.name',
    fill_color='Reds',
    fill_opacity=0.5,
    legend_name='Значение доли круглосуточных заведений',
```

```
) .add_to(m)

# выводим карту
m
```

Out[103]:



- Доли круглосуточных заведений неравномерно распределены по округам между 6 и 15 процентами от общего их числа. Так в Центральном округе доля равна 6%, тогда как в Юго-Восточном порядка 15%

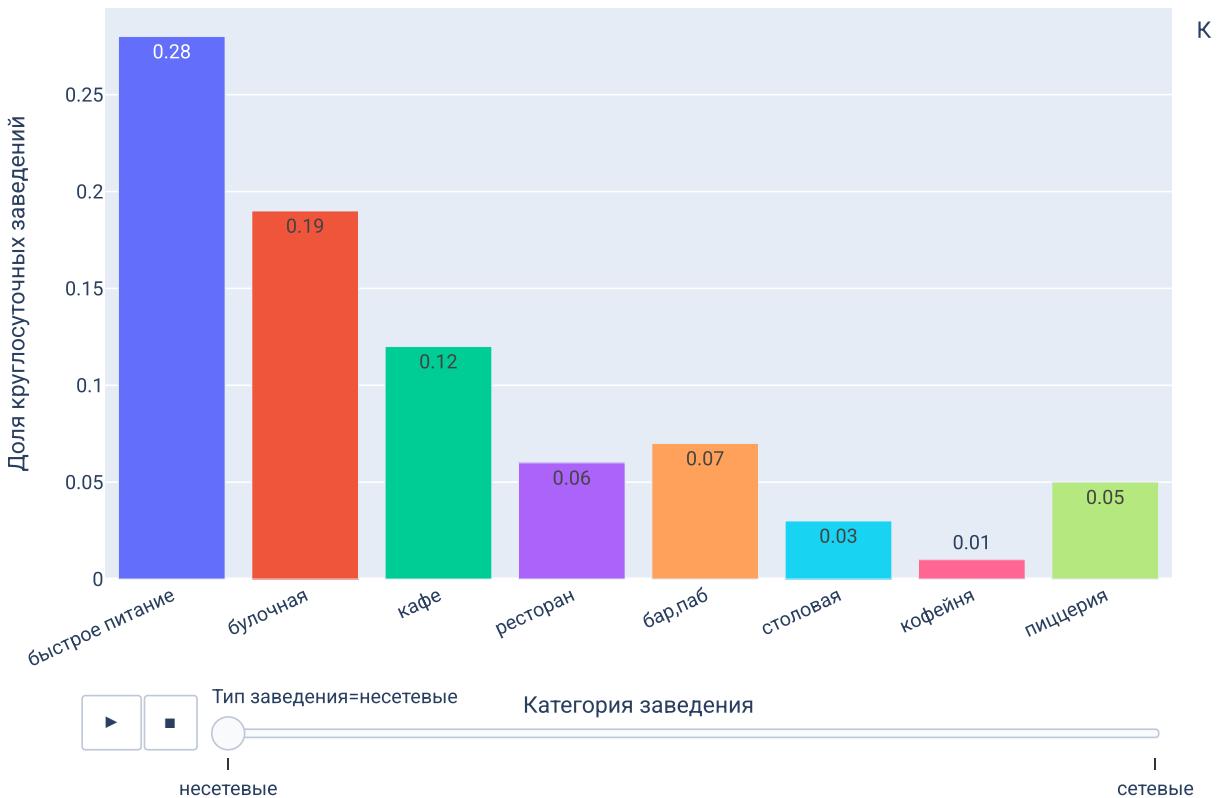
Анализ долей круглосуточных заведений по категориям

In [104...]

```
# выделим данные и построим график
share_by_cat = data.groupby(by=['new_chain','category'])['is_24/7'].mean().sort_values(ascending=False)
share_by_cat.columns = ['Тип заведения','Категория заведения','Доля круглосуточных заведений']
share_by_cat['Доля круглосуточных заведений'] = round(share_by_cat['Доля круглосуточных заведений'],2)

fig = px.bar(share_by_cat, x='Категория заведения', y='Доля круглосуточных заведений',\
             color='Категория заведения', animation_frame='Тип заведения', title='Распределение')
fig.update_xaxes(tickangle=-25)
fig.update_layout(width=900, height=600)
fig.show();
print('*****')
display(round(share_by_cat.describe(),2))
```

Распределение долей заведений заведений умеющих рядом торговый центр с учетом к



Доля круглосуточных заведений

count	16.00
mean	0.10
std	0.08
min	0.01
25%	0.05
50%	0.07
75%	0.13
max	0.28

- Доля круглосуточных заведений быстрого питания более 25%, тогда как пиццерий около 5% независимо от типа заведения. Доля круглосуточных сетевых булочных порядка 4% тогда как несетевых 19% и напротив несетевых кофеен 1% против 7% сетевых

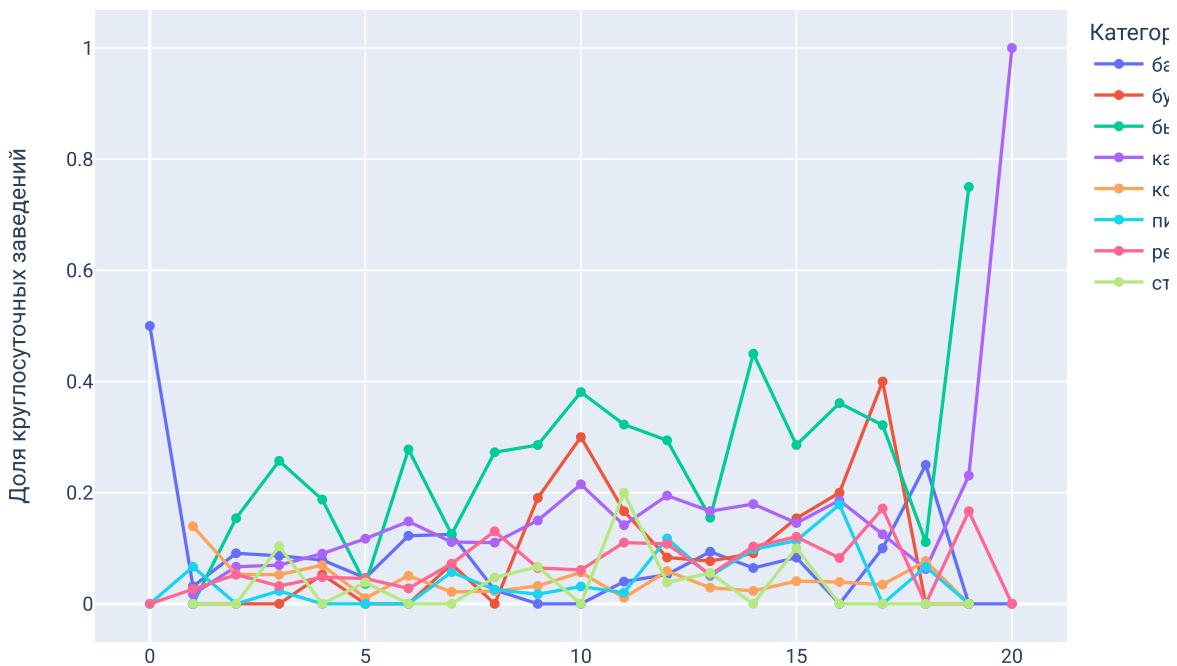
In [105...]

```
# Выделим данные
share_by_cat_d = data.pivot_table(index=['category','distance_km'],values='is_24/7',aggfunc=[len])
share_by_cat_d.columns = ['Категория','Расстояние от центра','Доля круглосуточных заведений','Ко
```

In [106...]

```
fig = px.line(share_by_cat_d, x='Расстояние от центра', y='Доля круглосуточных заведений', \
              color='Категория', markers=True,hover_name = 'Количество заведений')
fig.update_layout(showlegend=True,title='Изменения доли круглосуточных заведений в зависимости о
fig.show()
```

Изменения доли круглосуточных заведений в зависимости от расстояния до центра



- Заведения быстрого питания и кафе имеют тенденцию к увеличению доли круглосуточных при удалении от центра

Анализ взаимосвязи количества посадочных мест заведений с наличием торговых центров на расстоянии 400 метров

```
In [107...]: # напишем функцию для подсчета количества объектов в указанном радиусе по координатам(считает до
import math

def distance_to_object_counter(origin,points_goal,name_column='number_obejts',control_distance=0
    """this function takes four parameters
    origin - datafram with information about first point with columns 'lat' and 'lng'
    points_goal - datafram with informatin about second point with columns 'lat' and 'lng'
    name_column - name column with counting result which will be created in first dataset
    distance - distance between two points in killometers
    result is column wiht number of second colomns' objects in taken radius
    """
    try:
        origin[name_column] = origin['lat']*0
        origin[name_column] = origin[name_column].astype('int')
        def distance_to_counter(origin):
            lat1, lon1 = origin['lat'],origin['lng']
            lat2, lon2 = p
            radius = 6371 # km

            dlat = math.radians(lat2-lat1)
            dlon = math.radians(lon2-lon1)
            a = math.sin(dlat/2) * math.sin(dlat/2) + math.cos(math.radians(lat1)) \
                * math.cos(math.radians(lat2)) * math.sin(dlon/2) * math.sin(dlon/2)
            c = 2 * math.atan2(math.sqrt(a), math.sqrt(1-a))
            d = radius * c

            if d<=control_distance:
                return 1
            else:
                return 0

        for i in range(0,len(points_goal)):
            p = points_goal['lat'][i],points_goal['lng'][i]
            points_goal[name_column][i] = distance_to_counter(origin,p)
    except:
        print("ошибка")
    finally:
        return points_goal[name_column]
```

```

        tmp = origin.apply(distance_to_counter, axis=1)
        origin[name_column] = origin[name_column]+tmp
    except:
        return "ERROR: \
this function takes four parametrs \
origin data - datafram with information about first point with columns 'lat' and 'lng' \
points_goal - datafram with informain about second point with columns 'lat' and 'lng' \
name_column - name column wiht counting result wchich will be created in first dataset\
distance - distance beetween two points in kilopments\
result is column with number of second columns' objects in taken radius"

```

In [108]: # применим функцию(считает долго, из-за высокой асимптотики)
`distance_to_object_counter(data,malls,'malls_around_0.40km',0.40)`

In [109]: # проверим
`display(data.head())`

	name	category	address	district	hours	lat	lng	rating	price	z
0	WoWфли	кафе	Москва, улица Дыбенко, 7/1	Северный административный округ	ежедневно, 10:00–22:00	55.878494	37.478860	5.0	NaN	
1	Четыре комнаты	ресторан	Москва, улица Дыбенко, 36, корп. 1	Северный административный округ	ежедневно, 10:00–22:00	55.875801	37.484479	4.5	выше среднего	Ср счёт
2	Хазри	кафе	Москва, Клязьминская улица, 15	Северный административный округ	пн-чт 11:00–02:00; пт,сб 11:00–05:00; вс 11:00...	55.889146	37.525901	4.6	средние	Ср счёт
3	Dormouse Coffee Shop	кофейня	Москва, улица Маршала Федоренко, 12	Северный административный округ	ежедневно, 09:00–22:00	55.881608	37.488860	5.0	NaN	Цена капучин
4	Илья Марко	пиццерия	Москва, Правобережная улица, 1Б	Северный административный округ	ежедневно, 10:00–22:00	55.881166	37.449357	5.0	средние	Ср счёт:40

5 rows × 21 columns

Анализ распределения долей заведений имеющих торговый центр в указанной радиусе в по категориям заведения

In [110]: # добавим колонку для отражения факта наличия торговоо центра на заданном расстоянии
`data['near_is_mall'] = data['malls_around_0.40km']!=0`
`data['near_is_mall_str'] = data['near_is_mall'].map({True:'есть', False:'нет'})`

In [111]: share = data.groupby(by=['new_chain', 'category'])[['near_is_mall']].mean().sort_values(by='near_is_mall') = round(share['near_is_mall'],2)
`share.columns = ['Тип заведения', 'Категория заведения', 'Доля заведений']`
`share`

Out[111]: Тип заведения Категория заведения Доля заведений

0	несетевые	быстрое питание	0.68
1	сетевые	быстрое питание	0.67
2	сетевые	кофейня	0.65
3	сетевые	ресторан	0.62
4	сетевые	пиццерия	0.61
5	сетевые	булочная	0.59
6	сетевые	кафе	0.56
7	сетевые	бар,лаб	0.55
8	несетевые	булочная	0.55
9	несетевые	пиццерия	0.50
10	несетевые	ресторан	0.48
11	несетевые	кафе	0.48
12	несетевые	кофейня	0.47
13	несетевые	бар,лаб	0.46
14	сетевые	столовая	0.43
15	несетевые	столовая	0.40

```
In [112... fig = px.bar( share, x='Категория заведения', y='Доля заведений', \
    color='Категория заведения', animation_frame='Тип заведения', title='Распределение д...
fig.update_xaxes(tickangle=-25)
fig.update_layout(width=900, height=600)
fig.show();
```

Распределение долей заведений умеющих рядом торговый центр с учетом категорий



- Доля заведений быстрого питания равна порядка 68% как для сетевых, так и для несетевых заведений. Для остальных категорий мы видим прирост доли заведений с наличием торгового центра у сетевых заведений, причем кофейни рестораны и пиццерии имеют самый значимый - от 10% до 18%. Исходя из

этого можно сделать вывод, что сетевые кофейни, рестораны и пиццерии учитывают факт наличия торговых центров при выборе места расположения своих заведений, а посетители торговых центра навещают эти заведения.

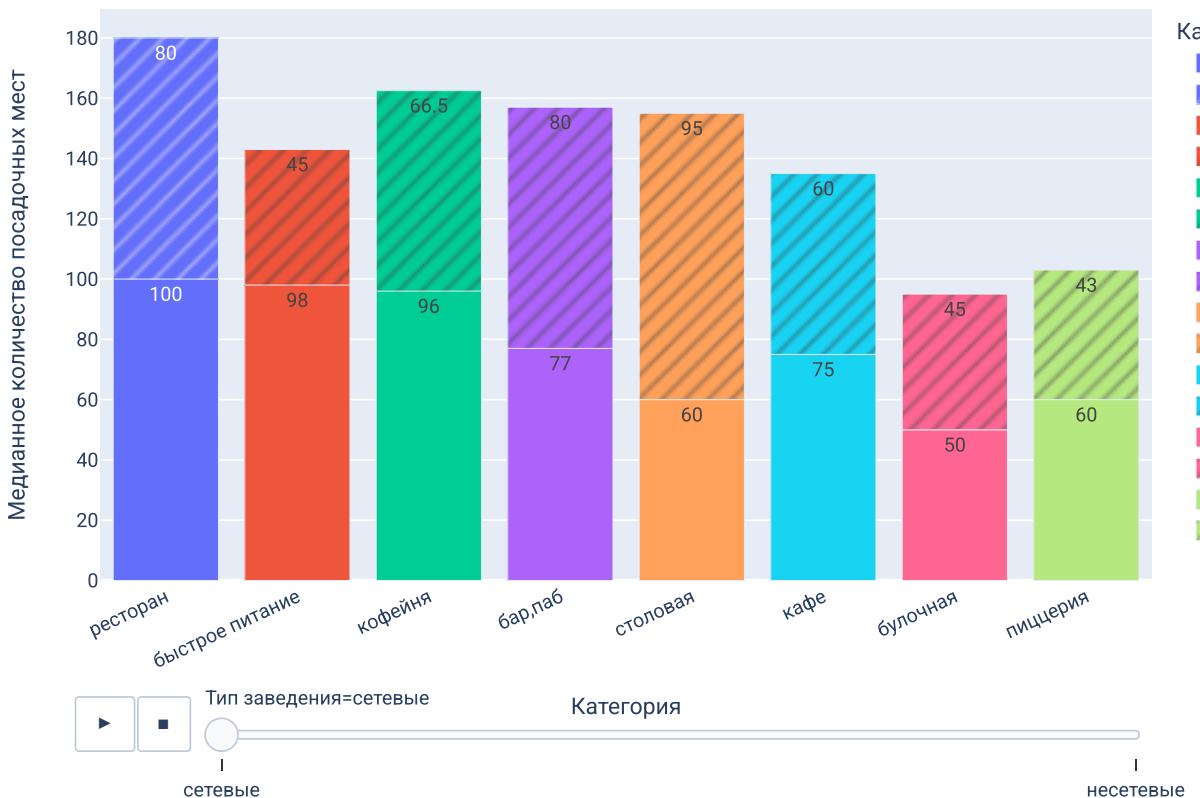
Анализ изменения количества посадочных мест в группах с наличием торгового центра и без, по категориям

```
In [113...]: # подготоим данные
median_seats = data.groupby(by=['new_chain', 'near_is_mall_str', 'category'])[['seats']].median()
median_seats.columns = ['Тип заведения', 'Наличие торгового центра', 'Категория', 'Медианное количество посадочных мест']
median_seats.head()
```

	Тип заведения	Наличие торгового центра	Категория	Медианное количество посадочных мест
0	несетевые	есть	бар,лаб	95.0
1	несетевые	есть	булочная	67.0
2	несетевые	есть	быстрое питание	55.5
3	несетевые	есть	кафе	60.0
4	несетевые	есть	кофейня	60.0

```
In [114...]: # построим график
fig = px.bar(median_seats.sort_values(by='Медианное количество посадочных мест', ascending=False),
              color='Категория', pattern_shape='Наличие торгового центра', animation_frame='Тип заведения',
              fig.update_xaxes(tickangle=-25)
fig.update_layout(width=1000, height=600)
fig.show();
```

Изменение количества посадочных мест в группах с наличием торгового центра и без |



- Проведя анализ целевых групп, можно заметить что в сетевых заведениях которые имеют proximity торговый центр в целом количество посадочных мест больше, самый большой прирост мы видим в категориях быстрого питания, кофеен и ресторанов, тогда как в столовых оно наоборот снижается что может говорить о слабой конкурентоспособности сетевых столовых среди аудитории торговых центров, что сети и учитывают. Количество посадочных мест в сетевых барах практически не меняется. Что

касается несетевых заведений то тут похожая тенденция, однако заведения быстрого питания и кофейни практически не меняют количество посадочных мест от наличия торговых центров по близости, это может свидетельствовать о том что аудитория торговых центров больше предпочитает сетьевые заведения этих категорий. В целом можно сказать что торговые центры генерируют значительный трафик для заведений общественного питания

[к содержанию](#)

Вывод по анализу заведений общественного питания:

- Кафе с самым большим количеством точек "Кулинарная лавка братьев Караваевых", у кофеен лидирует "Шоколадница", в барах - "МСК Lounge", среди ресторанов лидируют "Яндекс Лавка" и "Prime". Кафе и столовые с одноименными названиями говорят о наличии большого количества мелких заведений без названия
- Лидируют по количеству посадочных мест заведения в категориях: рестораны бары и кофейни и столовые, что можно увязать с особенностью ведения бизнеса, где клиенты получают полный набор услуг непосредственно в заведении, в отличие от булочных и пиццерий где довольно популярна работа через доставку и на вынос. В топе по количеству заведений для категории кофеен находится 'Шоколадница' со средним показателем в 120 мест, у ресторанов лидируют 'Prime' и 'Тануки' с 97 местами. В сетевых заведениях у ресторанов, кофеен, и столовых наблюдается превышение посадочных мест над несетевыми заведениями, что может говорить об особенностях ведения бизнеса и привлечения клиентов
- Большая часть в 61,9% заведений не являются сетевыми, тогда как доля сетевых составляет 38,1%. Около половины всех кофеен и пиццерий сетевые, тогда как среди ресторанов и кафе эта доля порядка 30%, самая высокая доля сетевых заведений в категории булочных. В данных возможно наличие ошибок в принадлежности заведений к типу
- Подавляющее большинство заведений в топ 15 по количеству относятся к сетевому типу, лидирующую позицию среди них занимает "Шоколадница". Так же можно отметить что большое количество заведений относятся к категориям кофеен и пиццерий. Среди несетевых заведений сгруппировались мелкие заведения с одноименными названиями, что говорит о неплохой заполненности ими сегмента рынка
- Большинство заведений находится в центральном округе, меньше всего в Северо-Западном. Во всех округах доминируют кафе, рестораны и кофейни. Превосходство количества кофеен, ресторанов и баров центрального округа над остальными более существенное чем в других категориях
- По анализу распределений рейтингов мы видим что среднее его значение расположилось между 4 и 4.4 по 5 бальной шкале, причем сетевые и не сетевые бары вверху рейтинга. Так же можно заметить что распределения в категориях кофейни и пиццерии имеют различия: сетевые заведения здесь в среднем получают более стабильные оценки, однако более высокие мы имеем в несетевых что может говорить о более высоком качестве обслуживания в несетевых заведениях.
- Самый высокий медианный рейтинг равен 4.4 балла в центральном административном округе, что может говорить о более высоком уровне и качестве обслуживания заведений в этом округе. Самый низкий рейтинг в северо-восточном и юго-восточном округах. В целом разброс рейтингов по округам составляет около 4 процентов
- Самое большое количество заведений на проспекте Мира, далее идет Профсоюзная улица и следом еще два проспекта: Ленинский и Вернадского. Пятницкая и Миклухо-Маклая расположились на последнем месте из топ 15 улиц. Большое количество заведений обусловлено оживленным трафиком, протяженностью и инфраструктурной, облегчающей прохождение этого трафика по ним
- На 15 самых популярных улицах расположилось 15 процентов всех заведений города, что является довольно большим показателем
- На карте видно, что улицы из топ 15 по количеству заведений практически не направлены в северо-восточном и восточном направлениях. Также можно обратить внимание на особенности размещения заведений на этих улицах, центрами их притяжения здесь становятся объекты, которые генерируют трафик: это парки, станции метро, торговые центры, муниципальные учреждения и прочее.
- Количество улиц с 1 заведением равно: 458 из общих 1448 что составляет 5% от заведений и практически 30% от улиц. Судя по названиям, непопулярные улицы это переулки, тупики и проезды с низким трафиком

- Среднее медианное значение среднего счета заведений по районам расположилось в пределах 450 и 1000 рублей. Центральный и Западный округа имеют самое высокое значение в 1000 рублей, что связано с близостью к центру в первом случае и наличием оживленных проспектов, ведущих в западном направлении во втором
- Самый высокий средний счет имеют бары и рестораны, причем сетевые заведения данной категории в среднем имеют на 250 рублей меньший счет. Несетевые кофейни и булочные имеют меньший счет в 381 и 425 рублей соответственно напротив 575 и 550 у сетевых. Столовые и кафе имеют одинаковый средний счет в сетевых и несетевых заведениях. Самыми недорогими заведениями общественного питания являются столовые со средним счетом в 300 рублей
- На линейном графике мы увидели снижение среднего счета при удалении из центра на расстояние порядка 6-7 километров, далее значение среднего счета стабилизируется, причем поведение сетевых и несетевых заведений подобно. Так же есть превышение среднего чека несетевых заведений над сетевыми в пределах 6 км.
- При глубоком анализе мы увидели что зависимость в уменьшении счета при увеличении расстояния от центра не во всех категориях подобна, так она более ярко выражена в ресторанах, барах и кофейнях, тогда как средний счет в заведениях быстрого питания и столовых практически не меняется
- Проведя анализ 3д модели распределения заведений в зависимости от координат и среднего чека мы обнаружили конусообразное распределение с возвышением среднего счета в центре и тем самым подтвердили наши выводы о снижении среднего чека при удалении из центра в ресторанах, барах и кофейнях и практически полном его отсутствии в заведениях быстрого питания и столовых. Также можно отметить неравномерность снижения среднего чека от центра к окраинам, здесь мы видим как заведения расходятся в форме лучей что повторяет направление основных шоссе и проспектов, по ним снижение идет более плавно и даже далеко от центра встречаются высокие показатели среднего чека
- Доли круглосуточных заведений неравномерно распределены по округам между 6 и 15 процентами от общего их числа. Так в Центральном округе доля равна 6%, тогда как в Юго-Восточном порядка 15%
- Доля круглосуточных заведений быстрого питания более 25%, тогда как пиццерий около 5% независимо от типа заведения. Доля круглосуточных сетевых булочных порядка 4% тогда как несетевых 19% и напротив несетевых кофеен 1% против 7% сетевых
- Заведения быстрого питания и кафе имеют тенденцию к увеличению доли круглосуточных при удалении от центра
- Доля заведений быстрого питания равна порядка 68% как для сетевых, так и для несетевых заведений. Для остальных категорий мы видим прирост доли заведений с наличием торгового центра у сетевых заведений, причем кофейни рестораны и пиццерии имеют самый значимый - от 10% до 18%. Исходя из этого можно сделать вывод, что сетевые кофейни, рестораны и пиццерии учитывают факт наличия торговых центров при выборе места расположения своих заведений, а посетители торговых центров навещают эти заведения.
- Проведя анализ целевых групп, можно заметить что в сетевых заведениях которые имеют поблизости торговый центр в целом количество посадочных мест больше, самый большой прирост мы видим в категориях быстрого питания, кофеен и ресторанов, тогда как в столовых оно наоборот снижается что может говорить о слабой конкурентоспособности сетевых столовых среди аудитории торговых центров, что сети и учитывают. Количество посадочных мест в сетевых барах практически не меняется. Что касается несетевых заведений то тут похожая тенденция, однако заведения быстрого питания и кофейни практически не меняют количество посадочных мест от наличия торговых центров по близости, это может свидетельствовать о том что аудитория торговых центров больше предпочитает сетевые заведения в этих категорий. В целом можно сказать что торгового центра генерируют значительный трафик для заведений общественного питания

Анализ кофеен

In [115...]

```
# подготовим данные для отображения долей
share_fofee_house = pd.DataFrame()
share_fofee_house.index=['Кофейни', 'Остальные']

share_fofee_house['Количество заведений'] = [len(data.loc[data['category']=='кофейня']), len(data)
share_fofee_house
```

Out[115]:

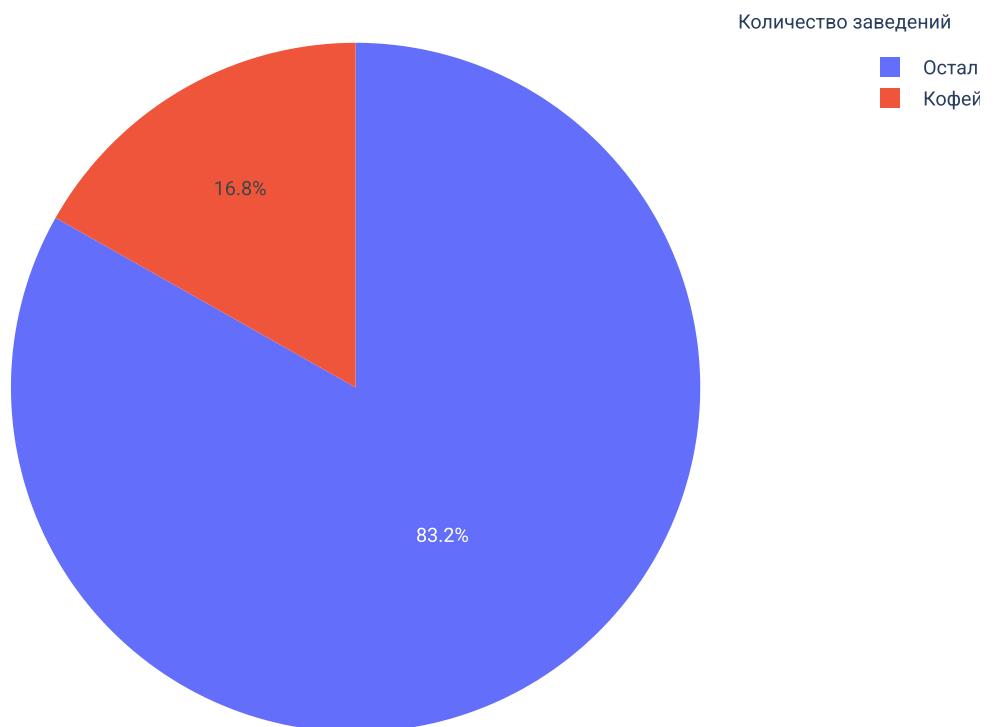
Количество заведений	
Кофейни	1413
Остальные	6993

In [116...]

```
# построим график с долей кофеен от общего числа заведений

# строим диаграмму с сегментами
fig = go.Figure(data=[go.Pie(labels=share_fofee_house.index, # указываем значения, которые появятся на сегментах
                               values=share_fofee_house['Количество заведений'], # указываем данные для сегментов
                               )]) # добавляем аргумент, который выделит сегмент-лидер на графике
fig.update_layout(title='Доля кофеен общего количества заведений', # указываем заголовок графика
                  width=800, # указываем размеры графика
                  height=600,
                  annotations=[dict(x=1.12, # вручную настраиваем аннотацию легенды
                                    y=1.05,
                                    text='Количество заведений',
                                    showarrow=False)])
fig.show() # выводим график
```

Доля кофеен общего количества заведений



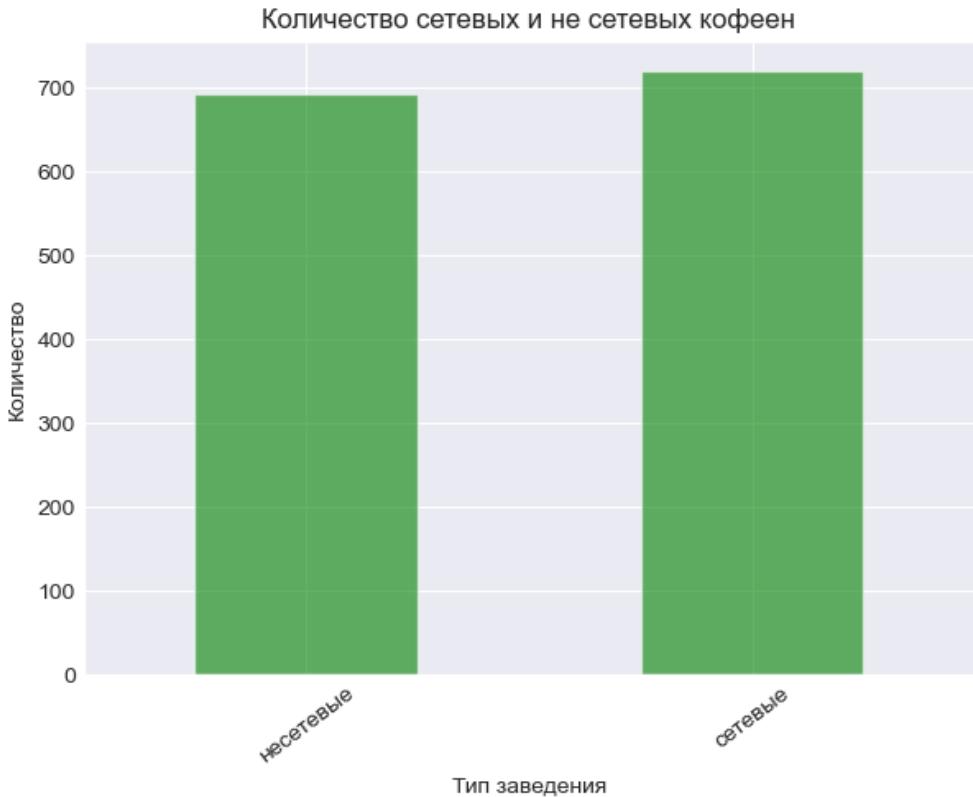
- Кофейни составляют 16,8% от общего числа заведений

In [117...]

```
# выделим их в отдельный сет
cofee_house = data.loc[data['category']=='кофейня'].copy()
```

In [118...]

```
# отобразим распределение сетевых и не сетевых кофеен
cofee_house.groupby(by='new_chain').count()['name'].plot(kind='bar', figsize=(7,5), alpha=0.6, color='blue')
plt.title('Количество сетевых и не сетевых кофеен')
plt.ylabel('Количество')
plt.xlabel('Тип заведения')
plt.xticks(rotation=37)
plt.show();
print('*****')
display(cofee_house.groupby(by='new_chain').count()['name'])
```



```
*****
new_chain
несетевые    693
сетевые      720
Name: name, dtype: int64
```

- Количество сетевых и несетевых кофеен распределились примерно поровну: 720 и 693 соответственно

```
In [119]: # нанесем на карту
# импортируем собственные иконки
from folium.features import CustomIcon
from folium.plugins import MarkerCluster
# импортируем карту и маркер
from folium import Map, Marker
# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаем карту Москвы
m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
# создаем пустой кластер, добавляем его на карту
marker_cluster = MarkerCluster().add_to(m)

# строим карту
for key, value in icons.items():

    def create_clusters(row):
        # сохраняем URL-адрес изображения со значком торгового центра с icons8,
        # это путь к файлу на сервере icons8
        icon_url = value

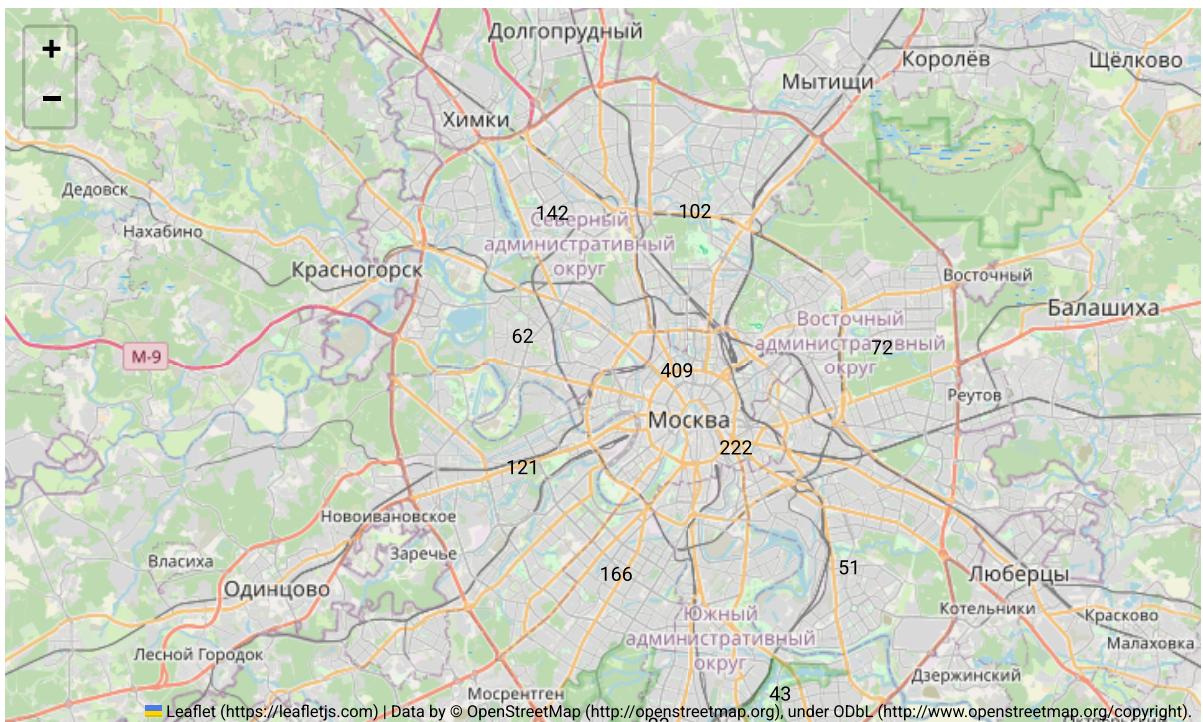
        # создаем объект с собственной иконкой размером 30x30
        icon = CustomIcon(icon_url, icon_size=(35, 35))

        # создаем маркер с иконкой icon и добавляем его в кластер
        Marker(
            [row['lat'], row['lng']],
            popup=f'{row['name']} {row['category']} рейтинг: {row['rating']} режим работы: {row['open_time']} - {row['close_time']},\nрасстояние до центра: {row['distance']} км",
            icon=icon,
        ).add_to(marker_cluster)

    # применяем функцию для создания кластеров к каждой строке датафрейма
    cofee_house.loc[cofee_house['category']==key].apply(create_clusters, axis=1)
```

```
# выводим карту
m
```

Out[119]:



Посмотрим на распределение кофеен по округам

In [120...]

```
# подготовим данные
cofee_house_count_distr = coffee_house.groupby(by='district')[['is_24/7']].count().reset_index()
cofee_house_count_distr.columns = ['Округ', 'Количество кофеен']

display(coffee_house_count_distr.sort_values(by='Количество кофеен', ascending=False))
```

	Округ	Количество кофеен
5	Центральный административный округ	426
2	Северный административный округ	191
3	Северо-Восточный административный округ	157
1	Западный административный округ	148
8	Южный административный округ	130
0	Восточный административный округ	102
7	Юго-Западный административный округ	94
6	Юго-Восточный административный округ	88
4	Северо-Западный административный округ	62

In [121...]

```
# Отобразим результат на карте Москвы
import json
from folium import Map, Choropleth

try:
    state_geo = 'F:\\обучение\\projects\\moscow_restaurants\\admin_level_geomap.geojson'
except:
    state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'

# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаём карту Москвы
m = folium.Map(location=[moscow_lat, moscow_lng], zoom_start=9.9)
# выводим карту

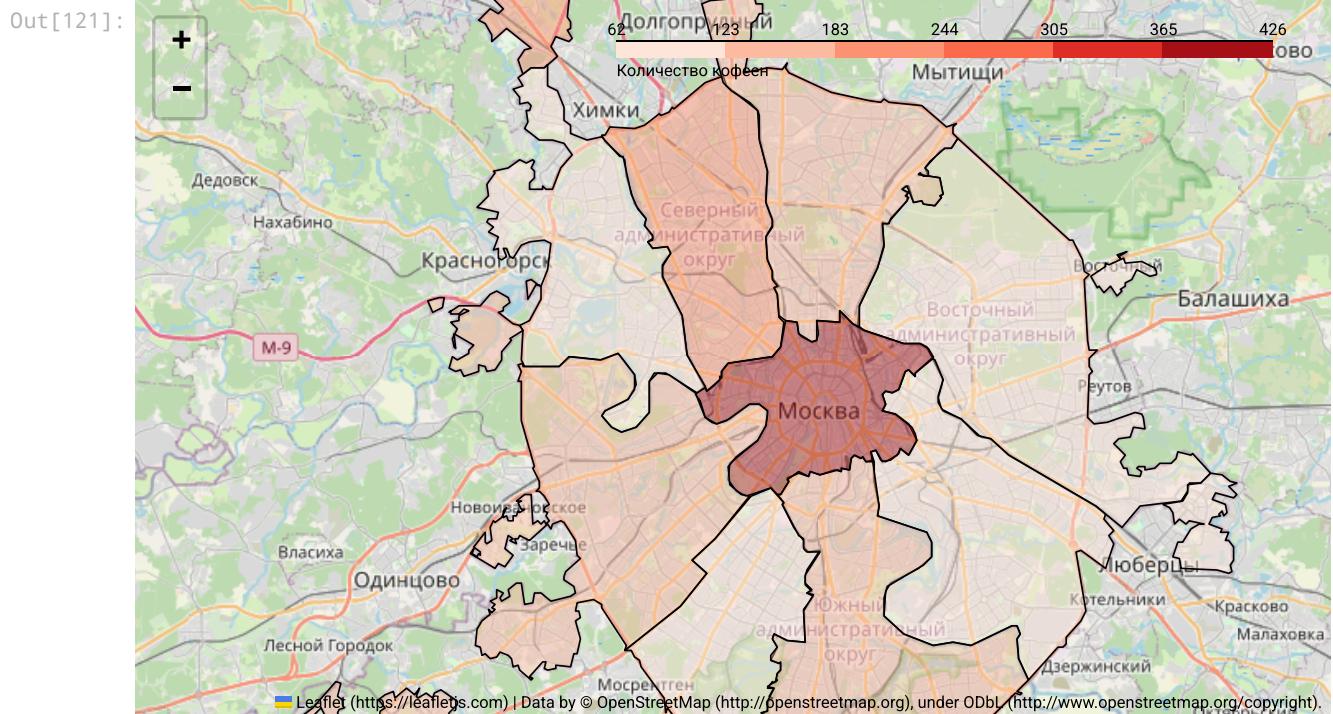
# создаём хороплет с помощью конструктора Choropleth и добавляем его на карту
```

```

Choropleth(
    geo_data=state_geo,
    data=cofee_house_count_distrit,
    columns=['Округ', 'Количество кофеен'],
    key_on='feature.name',
    fill_color='Reds',
    fill_opacity=0.45,
    legend_name='Количество кофеен',
).add_to(m)

# выводим карту
m

```



- Основная часть кофеен, 426, расположилась в центральном округе, Северо-Западный административный округ имеет самое низкое количество 62

Разберем топ 15 самых популярных улиц для расположения кофеен

In [122...]

```

# выделим данные
top_coffee_street = coffee_house.groupby(by=['street'])['lat'].count().sort_values(ascending=False)
#top_coffee_street.columns = ['Улица', 'Количество заведений']
top_coffee_street_s = coffee_house.query('street in @top_coffee_street.index')
top_coffee_street = top_coffee_street_s.groupby(by=['street', 'new_chain'])['lat'].count().sort_values()
top_coffee_street.columns = ['Улица', 'Тип заведения', 'Количество заведений']
display(top_coffee_street.head())
print('*****')
print(f"На 15 популярных улицах размещены {len(top_coffee_street_s)} клфейн из общих {len(coffee_h

```

	Улица	Тип заведения	Количество заведений
0	проспект Мира	сетевые	21
1	проспект Мира	несетевые	15
2	Ленинский проспект	несетевые	15
3	Ленинградский проспект	несетевые	14
4	Профсоюзная	сетевые	12

На 15 популярных улицах размещены 240 клфейн из общих 1413 что составляет порядка 17%

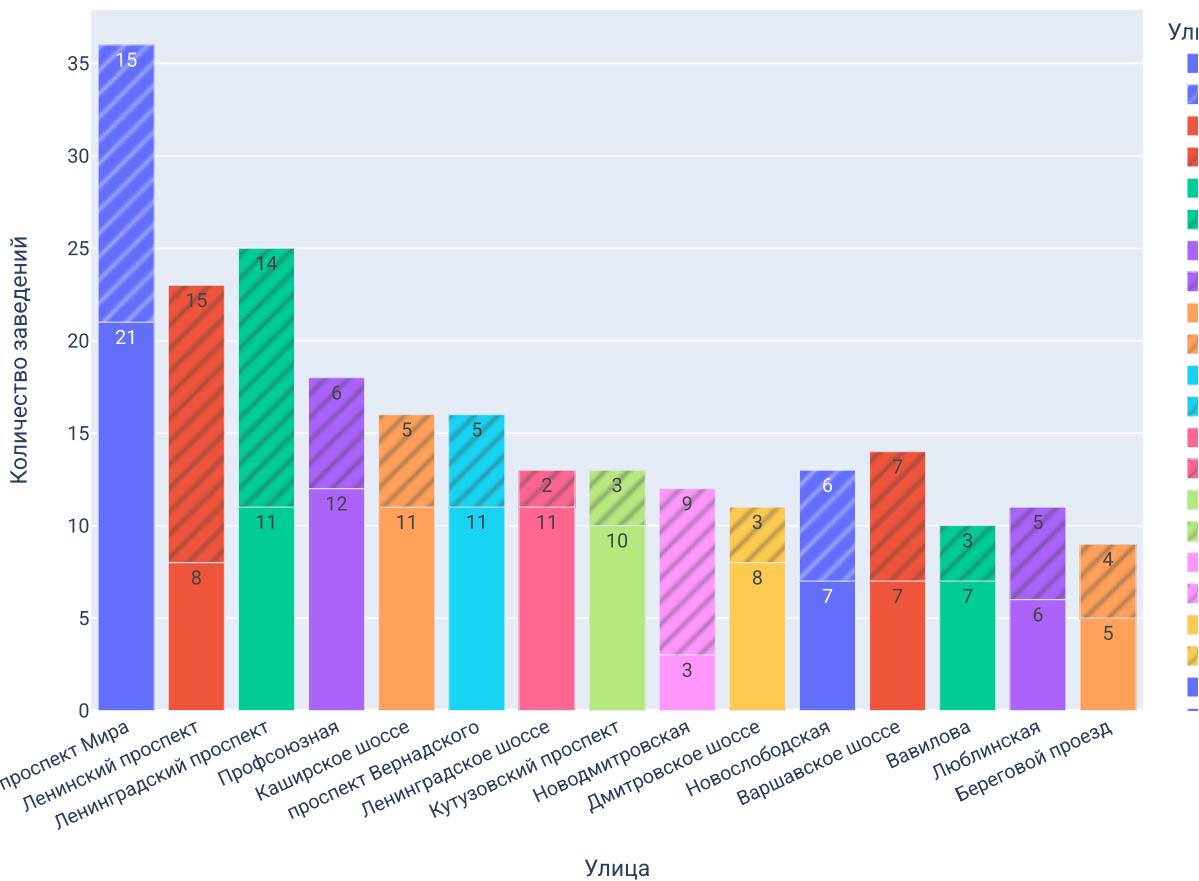
In [123...]

```

# построим график
fig = px.bar( top_coffee_street.sort_values(by='Количество заведений', ascending=False), x='Улица',
               color='Улица', text='Количество заведений', title='', pattern_shape='Тип заведения')

```

```
fig.update_xaxes(tickangle=-25)
fig.update_layout(width=1000, height=600)
fig.show();
```



- На 15 популярных улицах размещены 240 кофеен из общего числа 1413, что составляет порядка 17%. В топе по количеству заведений лидируют проспекты и шоссе, на первом месте проспект Мира, а на последнем Береговой проезд. Количество сетевых и несетевых заведений на этих улицах распределены неравномерно, так на Новодмитровской только 3 сетевых заведения из 12, тогда как на Ленинградском шоссе наоборот 11 сетевых и только 2 - нет.

```
In [124...]
# отобразим заведения с топ 15 улиц на карте
# импортируем собственные иконки
from folium.features import CustomIcon
from folium.plugins import MarkerCluster
# импортируем карту и маркер
from folium import Map, Marker
# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаём карту Москвы
m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
# создаём пустой кластер, добавляем его на карту
marker_cluster = MarkerCluster().add_to(m)

# строим карту
for key, value in icons.items():

    def create_clusters(row):
        # сохраняем URL-адрес изображения со значком торгового центра с icons8,
        # это путь к файлу на сервере icons8
        icon_url = value

        # создаём объект с собственной иконкой размером
        icon = CustomIcon(icon_url, icon_size=(20, 20))

        # создаём маркер с иконкой icon и добавляем его в кластер
        Marker(
```

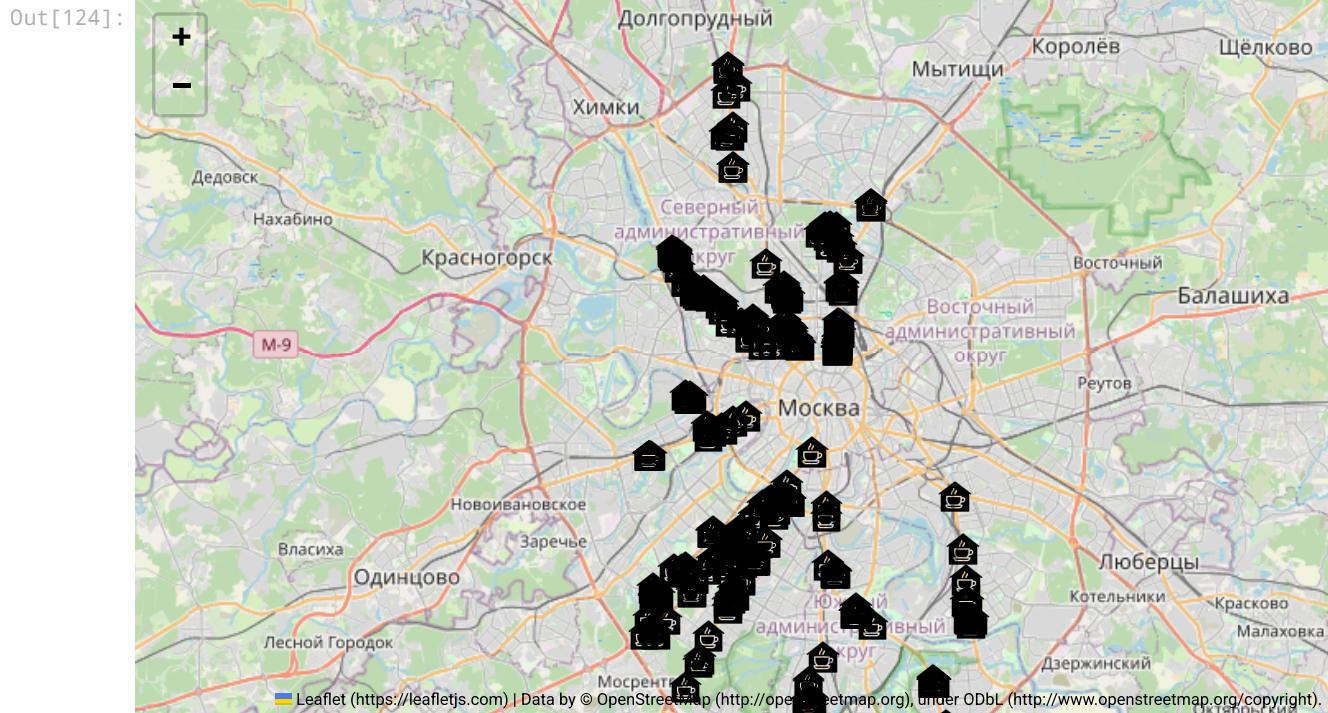
```

        [row['lat'], row['lng']],
        popup=f'{row['name']} {row['new_chain']} рейтинг: {row['rating']} режим работы: {row['work_time']}
        расстояние до центра: {row['distance']} км",
        icon=icon,
    ).add_to(m)

# применяем функцию для создания кластеров к каждой строке датафрейма
top_coffee_street_s.loc[top_coffee_street_s['category']==key].apply(create_clusters, axis=1)

# выводим карту
m

```



- На карте можно заметить особенности расположения заведений на топ 15 улицах. Во-первых: все улицы крупные с развитой инфраструктурой что обеспечивает хорошую транспортную и пешую доступность к заведениям, во вторых: можно отметить что улицы идущие в восточном и северо-восточном направлении не попали в наш топ, в третьих: на самих улицах кофейни в основном сосредотачиваются в точках которые генерируют трафик - это торговые центры, деловые центры, крупные муниципальные учреждения, станции метро и прочее.

Посмотрим на количество самых популярных кофеен

In [125...]

```

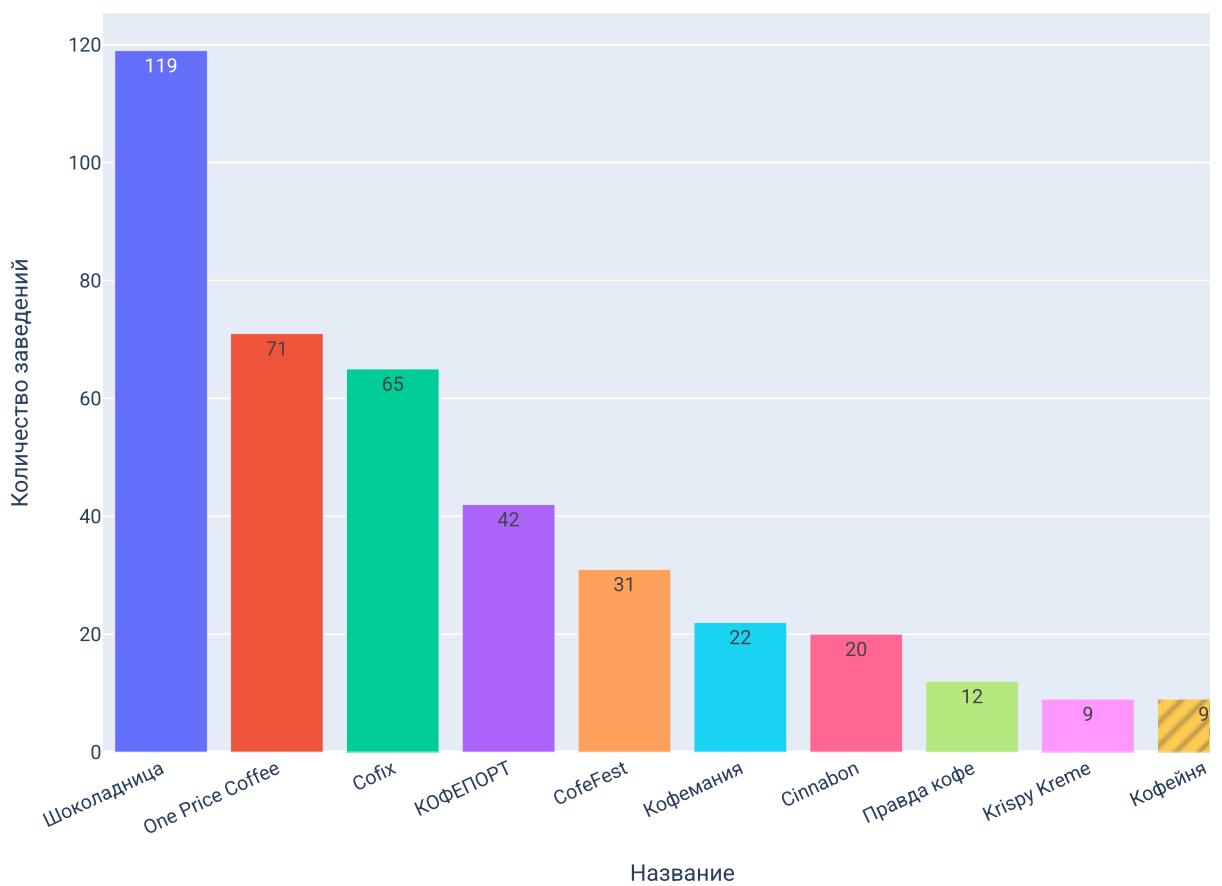
# выделим данные
top_coffee_name = coffee_house.groupby(by=['name'])['lat'].count().sort_values(ascending=False).head(15)
#top_coffee_street.columns = ['Улица', 'Количество заведений']
top_coffee_name_s = coffee_house.query('name in @top_coffee_name.index')
top_coffee_name = top_coffee_name_s.groupby(by=['name', 'new_chain'])['lat'].count().sort_values(ascending=False)
top_coffee_name.columns = ['Название', 'Тип заведения', 'Количество заведений']
display(top_coffee_name)
print('*****')
print(f"Топ 10 популярных кофеен составляет {len(top_coffee_name_s)} из общих {len(coffee_house)}")

```

	Название	Тип заведения	Количество заведений
0	Шоколадница	сетевые	119
1	One Price Coffee	сетевые	71
2	Cofix	сетевые	65
3	КОФЕПОРТ	сетевые	42
4	CofeFest	сетевые	31
5	Кофемания	сетевые	22
6	Cinnabon	сетевые	20
7	Правда кофе	сетевые	12
8	Krispy Kreme	сетевые	9
9	Кофейня	несетевые	9

Топ 10 популярных кофеен составляет 400 из общих 1413

```
In [126...]: # построим график
fig = px.bar( top_coffee_name.sort_values(by='Количество заведений', ascending=False), x='Название',
               color='Название', text='Количество заведений', title='', pattern_shape='Тип заведения'
fig.update_xaxes(tickangle=-25)
fig.update_layout(width=1000, height=600)
fig.show();
```



- В топе по количеству находятся крупные сетевые заведения, Шоколадница имеет самое большое количество заведений.

Проведем анализ доли круглосуточных кафеен

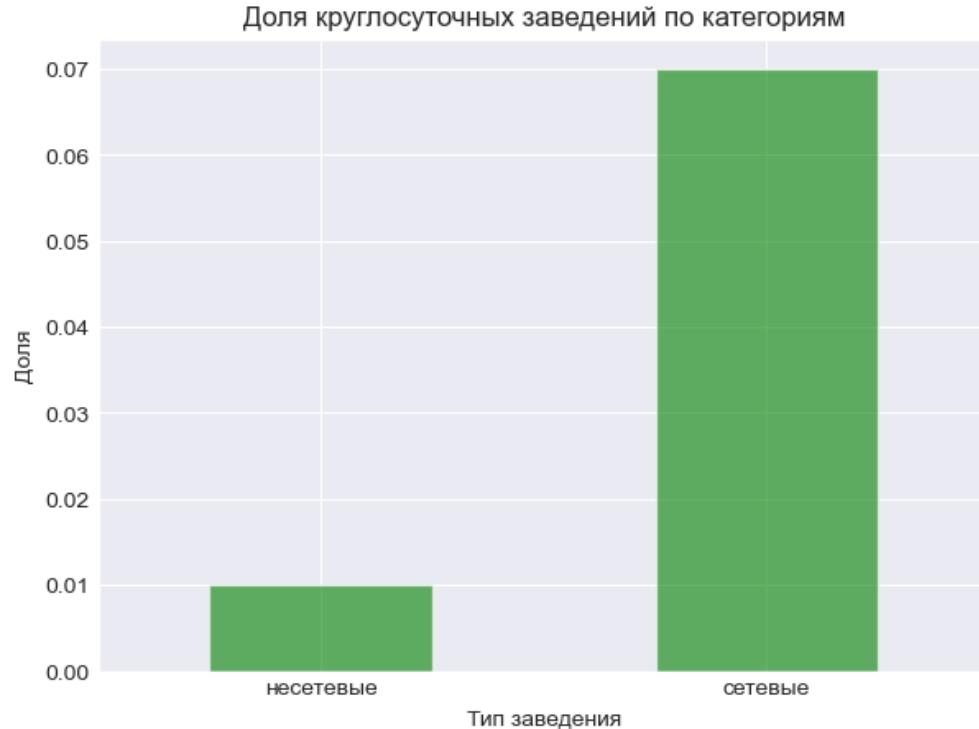
```
In [127...]: # выделим данные и построим график
chain_share = round(coffee_house.groupby(by='new_chain')['is_24/7'].mean(), 2)
```

```

chain_share.columns = ['Тип заведения', 'Доля круглосуточных']
chain_share.plot(kind='bar', figsize=(7,5), alpha=0.6, color='g')
plt.title('Доля круглосуточных заведений по категориям')
plt.ylabel('Доля')

plt.xlabel('Тип заведения')
plt.xticks(rotation=0)
plt.show();
print('*****')
display(chain_share)

```



```

new_chain
несетевые    0.01
сетевые      0.07
Name: is_24/7, dtype: float64

```

- Доля круглосуточных сетевых кофеен 7% против 1% у несетевых

In [128...]

```

# подготовим данные
cofee_house_is24_distr = coffee_house.groupby(by='district')['is_24/7'].mean().reset_index()
cofee_house_is24_distr.columns = ['Округ', 'Доля круглосуточных заведений']
cofee_house_is24_distr['Доля круглосуточных заведений'] = round(cofee_house_is24_distr['Доля круглосуточных заведений'])
display(cofee_house_is24_distr.sort_values(by='Доля круглосуточных заведений', ascending=False))
print(f"Доля круглосуточных кофеен от общего их числа равна {round(coffee_house['is_24/7'].mean())}")

```

Округ Доля круглосуточных заведений		
7	Юго-Западный административный округ	0.07
1	Западный административный округ	0.06
5	Центральный административный округ	0.06
0	Восточный административный округ	0.05
2	Северный административный округ	0.03
4	Северо-Западный административный округ	0.03
3	Северо-Восточный административный округ	0.02
6	Юго-Восточный административный округ	0.01
8	Южный административный округ	0.01

Доля круглосуточных кофеен от общего их числа равна 0.04

In [129...]

```

# отобразим результат на карте москвы
import json
from folium import Map, Choropleth

```

```

try:
    state_geo = 'F:\\обучение\\projects\\moscow_restaurants\\admin_level_geomap.geojson'
except:
    state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'

# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

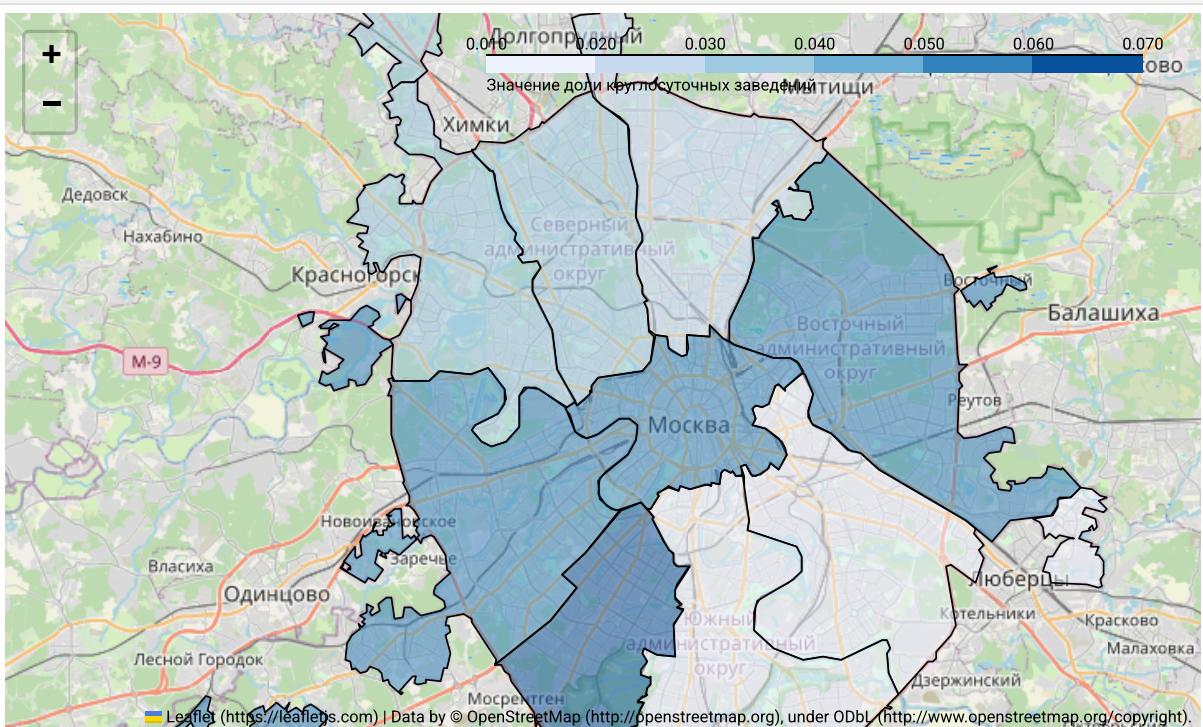
# создаём карту Москвы
m = folium.Map(location=[moscow_lat, moscow_lng], zoom_start=9.9)
# выводим карту

# создаём хороплет с помощью конструктора Choropleth и добавляем его на карту
Choropleth(
    geo_data=state_geo,
    data=cofee_house_is24_distrit,
    columns=['Округ', 'Доля круглосуточных заведений'],
    key_on='feature.name',
    fill_color='Blues',
    fill_opacity=0.5,
    legend_name='Значение доли круглосуточных заведений',
).add_to(m)

# выводим карту
m

```

Out[129]:



- Доля круглосуточных кофеен довольно мала и составляет порядка 4% от всех заведений. Распределение круглосуточных кофеен по округам неравномерно и располагается между 1% и 7%, где 7% в Юго-Западном административном округе, 1% в Южном и Юго-Восточном

Посмотрим на распределение рейтингов кофеен

In [130...]

```

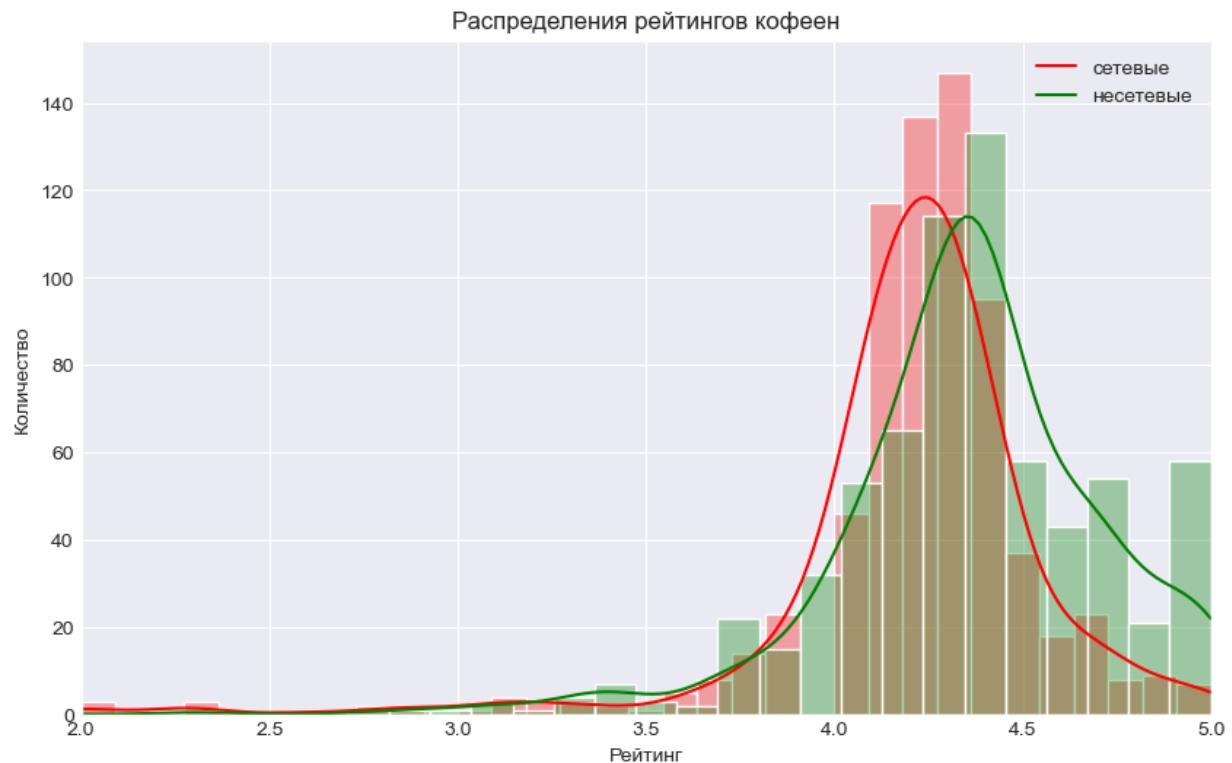
# построим гистограмму для отображения распределения рейтингов сетевых и несетевых кофеен
sns.set_style('darkgrid')
plt.figure(figsize=(10,6))
ax = sns.histplot(x='rating', data=cofee_house.loc[cofee_house['new_chain']=='сетевые'], kde=True)
sns.histplot(ax=ax,x='rating', data=cofee_house.loc[cofee_house['new_chain']=='несетевые'], kde=True)
#sns.move_legend(ax, "lower right", bbox_to_anchor=(1, 1.05), ncol=3, title=None, frameon=False)
plt.xlim(2,5)
plt.legend(['сетевые', 'несетевые'])
plt.grid(True)
ax.set_title('Распределения рейтингов кофеен')
ax.set_ylabel('Количество')

```

```

ax.set_xlabel('Рейтинг')
plt.show();
print('*****')
print('Описательная статистика рейтингов кофеен')
print(round(cofee_house.rating.describe(),2))

```



```

*****
Описательная статистика рейтингов кофеен
count    1413.00
mean      4.28
std       0.37
min       1.40
25%      4.10
50%      4.30
75%      4.40
max      5.00
Name: rating, dtype: float64

```

- Рейтинги сетевых кофеен сдвинуты в меньшую сторону, хотя имеют более стабильные показатели, тогда как несетевые кофейни чаще получают более высокие оценки**

In [131...]: # посчитаем средний рейтинг по округам

```

rating_coffee = coffee_house.groupby('district', as_index=False)[['rating']].agg('median')
display(rating_coffee)

```

	district	rating
0	Восточный административный округ	4.3
1	Западный административный округ	4.2
2	Северный административный округ	4.3
3	Северо-Восточный административный округ	4.3
4	Северо-Западный административный округ	4.3
5	Центральный административный округ	4.3
6	Юго-Восточный административный округ	4.3
7	Юго-Западный административный округ	4.3
8	Южный административный округ	4.3

In [132...]: # отобразим результат на карте москвы

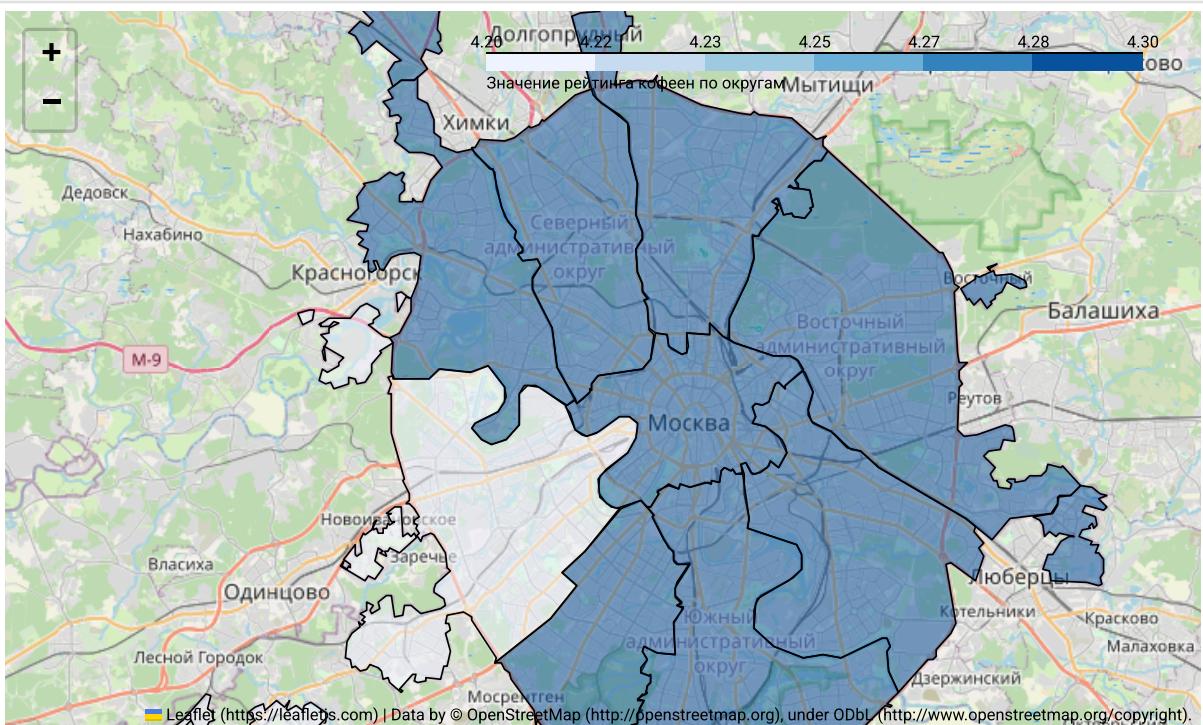
```

import json
from folium import Map, Choropleth

```

```
try:  
    state_geo = 'F:\\обучение\\projects\\moscow_restaurants\\admin_level_geomap.geojson'  
except:  
    state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'  
  
# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы  
moscow_lat, moscow_lng = 55.751244, 37.618423  
  
# создаем карту Москвы  
m = folium.Map(location=[moscow_lat, moscow_lng], zoom_start=9.9)  
# выводим карту  
  
# создаем хороплет с помощью конструктора Choropleth и добавляем его на карту  
Choropleth(  
    geo_data=state_geo,  
    data=rating_coffee,  
    columns=['district', 'rating'],  
    key_on='feature.name',  
    fill_color='Blues',  
    fill_opacity=0.5,  
    legend_name='Значение рейтинга кофеен по округам',  
) .add_to(m)  
  
# выводим карту  
m
```

Out[132]:



- Рейтинг кофеен практически не меняется по округам

Проведем анализ распределения среднего чека и средней стоимости чашки капучино

In [133...]

```
# построим гистограмму для отображения распределения стоимости чашки капучино сетевых и несетевых кофеен  
sns.set_style('darkgrid')  
plt.figure(figsize=(10,6))  
ax = sns.histplot(x='middle_coffee_cup', data=coffee_house.loc[coffee_house['new_chain']=='сетевые'])  
sns.histplot(ax=ax,x='middle_coffee_cup', data=coffee_house.loc[coffee_house['new_chain']=='несетевые'])  
#sns.move_legend(ax, "lower right", bbox_to_anchor=(1, 1.05), ncol=3, title=None, frameon=False)  
plt.xlim(0,400)  
plt.legend(['сетевые', 'несетевые'])  
plt.grid(True)  
ax.set_title('Распределения стоимости чашки капучино')  
ax.set_ylabel('Количество')  
ax.set_xlabel('Стоимость')  
plt.show();
```

```

print('*****')
print('Описательная статистика стоимости чашки капучино для всех кофеен')
display(round(cofee_house.middle_coffee_cup.describe(),2))

```



```
*****
```

Описательная статистика стоимости чашки капучино для всех кофеен

count	521.00
mean	175.06
std	89.75
min	60.00
25%	124.00
50%	170.00
75%	225.00
max	1568.00
Name:	middle_coffee_cup, dtype: float64

- Стоимость чашки капучино в несетевых заведениях имеет нормальную форму распределения со средним значением около 160 рублей, тогда как в сетевых мы четко видим сегментацию по ценовым категориям с пиками которые соответствуют стандартной стоимости капучино, где в верхней категории мы имеем пик на уровне 250 рублей в нижней два пика в 60 и 90 рублей.

In [134...]

```

# посчитаем стоимость чашки капучино по округам, подготовим данные
cofee_house_cup_distr = coffee_house.groupby(by='district')['middle_coffee_cup'].median().reset_index()
cofee_house_cup_distr.columns = ['Округ', 'Стоимость чашки капучино']
display(cofee_house_cup_distr.sort_values(by='Стоимость чашки капучино', ascending=False))
print(f"Средняя стоимость чашки капучино {coffee_house['middle_coffee_cup'].median()}")

```

	Округ	Стоимость чашки капучино
7	Юго-Западный административный округ	198.0
5	Центральный административный округ	190.0
1	Западный административный округ	189.0
4	Северо-Западный административный округ	165.0
3	Северо-Восточный административный округ	162.5
2	Северный административный округ	159.0
8	Южный административный округ	150.0
6	Юго-Восточный административный округ	147.5
0	Восточный административный округ	135.0

Средняя стоимость чашки капучино 170.0

```
In [135...]
# отобразим результат на карте москвы
import json
from folium import Map, Choropleth

try:
    state_geo = 'F:\\обучение\\projects\\moscow_restaurants\\admin_level_geomap.geojson'
except:
    state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'

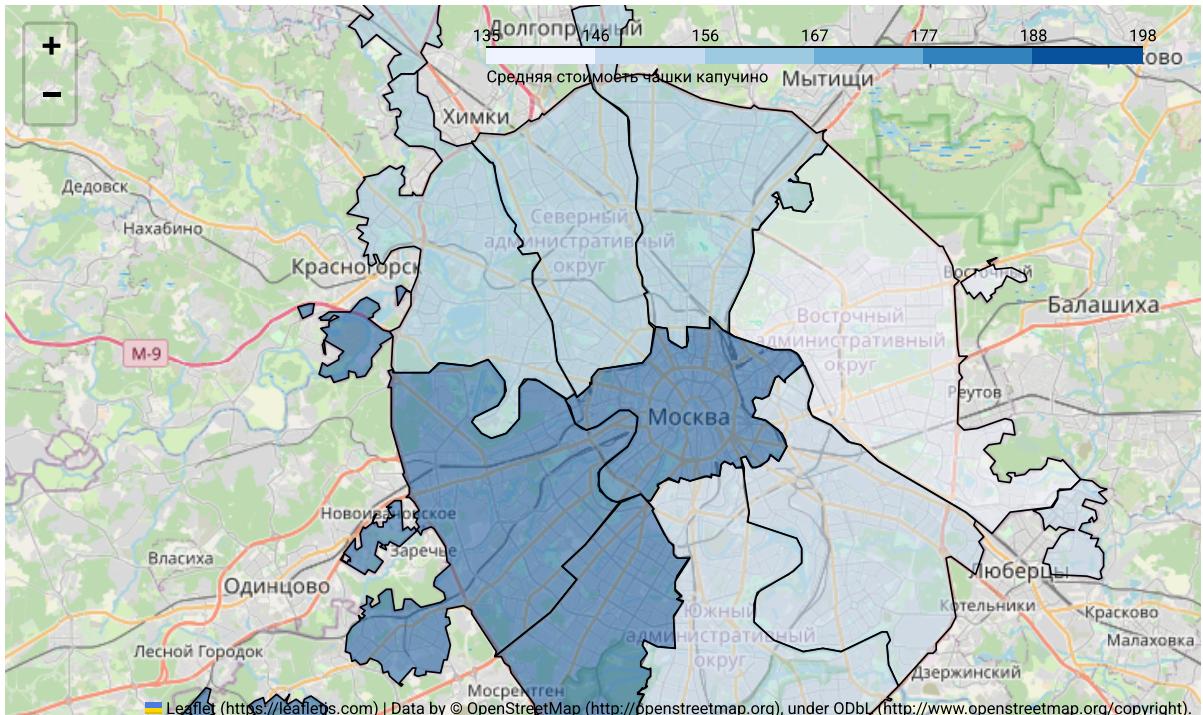
# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаём карту Москвы
m = folium.Map(location=[moscow_lat, moscow_lng], zoom_start=9.9)
# выводим карту

# создаём хороплет с помощью конструктора Choropleth и добавляем его на карту
Choropleth(
    geo_data=state_geo,
    data=coffee_house_cup_distr,
    columns=['Округ', 'Стоимость чашки капучино'],
    key_on='feature.name',
    fill_color='Blues',
    fill_opacity=0.5,
    legend_name='Средняя стоимость чашки капучино',
).add_to(m)

# выводим карту
m
```

Out[135]:



- Средняя стоимость чашки капучино неравномерно распределена по округам Москвы, так Центральный и Юго-западный округа имеют более высокую среднюю стоимость в районе 190 рублей, тогда как средняя стоимость в Восточном округе равна 135 рублям

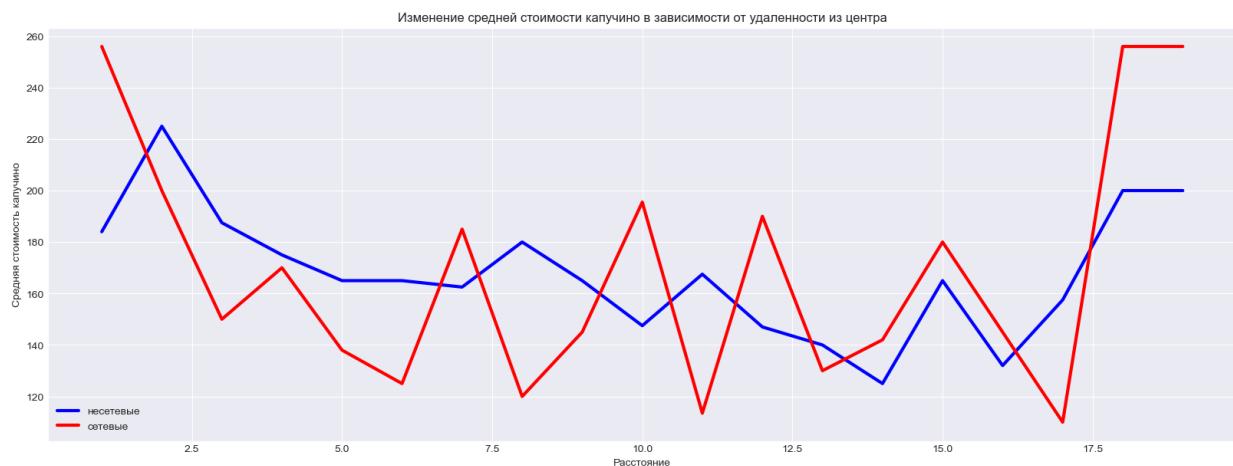
Анализируем зависимость удаленности из центра и стоимость чашки капучино

```
In [141...]
# Построим график изменения средней стоимости капучино в зависимости от удаленности из центра
ax = coffee_house.loc[coffee_house['new_chain']=='несетевые'].groupby(by='distance_km')['middle_cost'].mean()
coffee_house.loc[coffee_house['new_chain']=='сетевые'].groupby(by='distance_km')['middle_coffee_cost'].mean().plot()
plt.legend(labels=['несетевые', 'сетевые'])
plt.title('Изменение средней стоимости капучино в зависимости от удаленности из центра')
```

```

plt.xlabel('Расстояние')
plt.ylabel('Средняя стоимость капучино')
plt.show()

```



- Стоимость чашки капучино резко снижается с 220 до 160 рублей первые 6 километров из центра, далее стабилизируется на уровне своего медианного значения, причем изменение цены в несетевых заведениях происходит более плавно, тогда как в сетевых мы видим ступени, которые отражают отсутствие или наличие заведений с дорогим кофе в указанном радиусе.

Проведем анализ распределения количества посадочных мест кофеен

```

## построим гистограмму для отображения распределения стоимости чашки капучино сетевых и несетевых
sns.set_style('darkgrid')
plt.figure(figsize=(10,6))
ax = sns.histplot(x='seats', data=cofee_house.loc[cofee_house['new_chain']=='сетевые'], kde=True)
sns.histplot(ax=ax,x='seats', data=cofee_house.loc[cofee_house['new_chain']=='несетевые'], kde=True)
#sns.move_legend(ax, "lower right", bbox_to_anchor=(1, 1.05), ncol=3, title=None, frameon=False)
plt.xlim(0,500)
plt.legend(['сетевые', 'несетевые'])
plt.grid(True)
ax.set_title('Распределения количества посадочных мест в кафеях')
ax.set_ylabel('Количество')
ax.set_xlabel('Количество посадочных мест в кафеях')
plt.show();
print('*****')
print('Описательная статистика количества посадочных мест для всех кофеен')
display(round(coffee_house.seats.describe(),2))

```



```
*****
Описательная статистика количества посадочных мест для всех кофеен
count    751.00
mean     111.20
std      127.84
min      0.00
25%     40.00
50%     80.00
75%     144.00
max     1288.00
Name: seats, dtype: float64
```

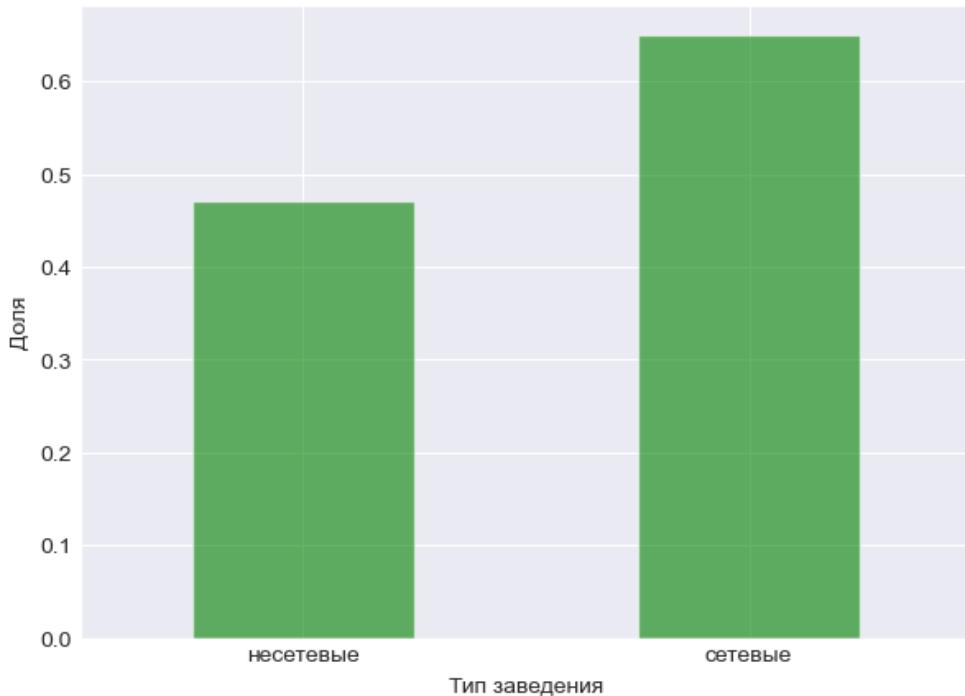
- Среднее количество посадочных мест в кафе равно 80. Сетевые заведения имеют небольшой сдвиг в большую сторону в распределении количества посадочных мест

Проведем анализ влияния наличия торговых центров в радиусе 400 метров на количество посадочных мест в кафе

```
In [143...]: # выделим данные и построим график
near_is_mall_share = round(cofee_house.groupby(by='new_chain')['near_is_mall'].mean(),2)
near_is_mall_share.columns = ['Доля с наличием торгового центра']
near_is_mall_share.plot(kind='bar', figsize=(7,5), alpha=0.6, color='g')
plt.title('Доля заведений с наличием торгового центра по категориям')
plt.ylabel('Доля')

plt.xlabel('Тип заведения')
plt.xticks(rotation=0)
plt.show();
print('*****')
display(near_is_mall_share)
```

Доля заведений с наличием торгового центра по категориям



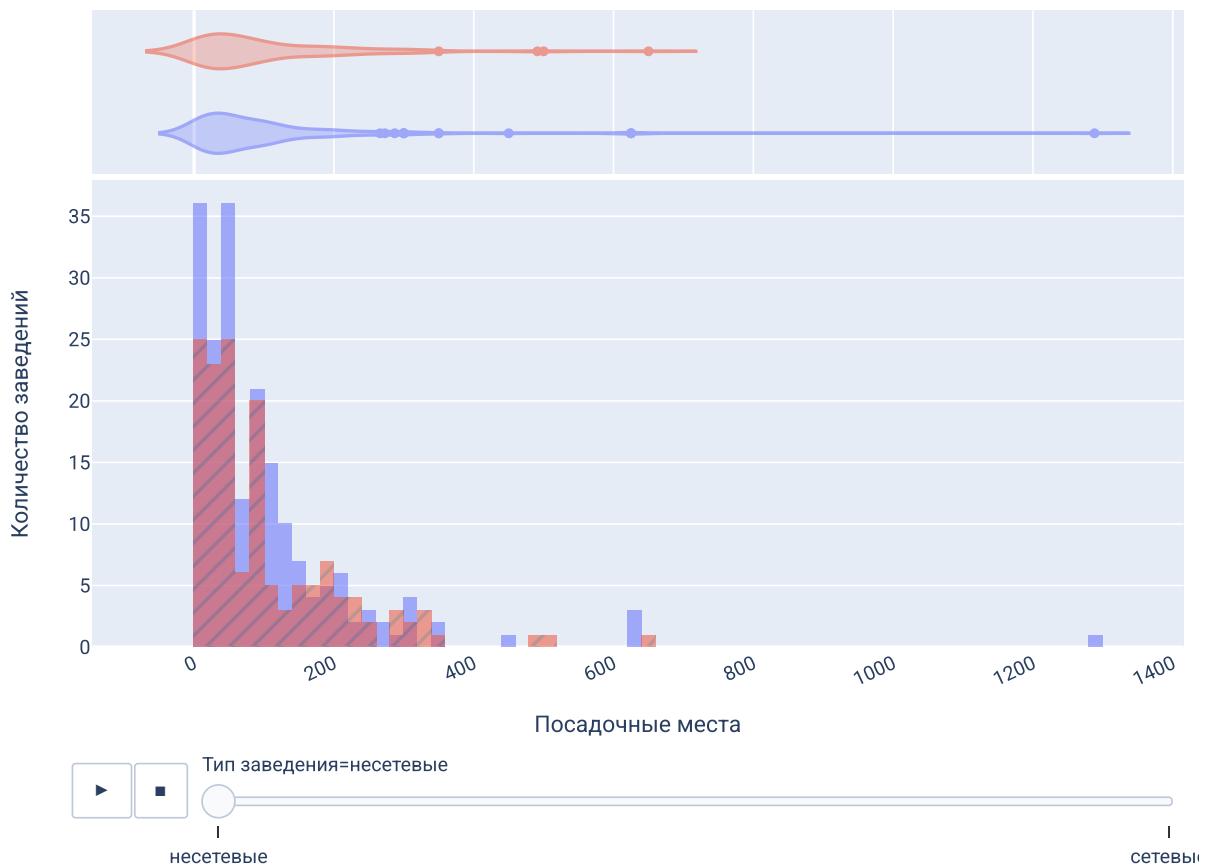
```
*****
```

```
new_chain
несетевые    0.47
сетевые      0.65
Name: near_is_mall, dtype: float64
```

- Доля сетевых заведений имеющих по близости торговый центр равна 65%, тогда как у несетевых этот показатель равен 47%

```
In [144]: cofee_house['Наличие торгового центра'] = cofee_house.near_is_mall_str
cofee_house['Тип заведения'] = cofee_house.new_chain
```

```
In [146]: # построим график
fig = px.histogram(cofee_house, x='seats', marginal='violin', \
                    color='Наличие торгового центра', pattern_shape='Наличие торгового центра', animation_group='Наличие торгового центра')
fig.update_xaxes(tickangle=-25)
fig.update_layout(width=950, height=600)
fig.update_layout(barmode='overlay', xaxis_title='Посадочные места',
                  yaxis_title='Количество заведений')
fig.update_traces(opacity=0.55)
fig.show();
```



- При более глубоком анализе количества посадочных мест можно заметить резкое увеличение их медианного значения с 66 без торгового центра proximity до 96 с его наличием в целевой группе сетевых кофеен, что свидетельствует о том что генерируемый торговым центром трафик положительно сказывается на проходимости в них

[к содержанию](#)

Вывод по анализу кофеен:

- Кофейни составляют 16,8% от общего числа заведений
- Количество сетевых и несетевых кофеен распределились примерно поровну: 720 и 693 соответственно
- Основная часть кофеен, 426, расположилась в центральном округе, Северо-Западный административный округ имеет самое низкое количество 62
- На 15 популярных улицах размещены 240 кофеен из общего числа 1413, что составляет порядка 17%. В топе по количеству заведений лидируют проспекты и шоссе, на первом месте проспект Мира, а на последнем Береговой проезд. Количество сетевых и несетевых заведений на этих улицах распределены неравномерно, так на Ново-Дмитровской только 3 сетевых заведения из 12, тогда как на Ленинградском шоссе наоборот 11 сетевых и только 2 - нет.
- На карте можно заметить особенности расположения кофеен на топ 15 улицах. Во-первых: все улицы крупные с развитой инфраструктурой что обеспечивает хорошую транспортную и пешую доступность к заведениям, во вторых: можно отметить что улицы идущие в восточном и северо-восточном направлении не попали в наш топ, в третьих: на самих улицах кофейни в основном сосредотачиваются в точках которые генерируют трафик – это торговые центры, деловые центры, крупные муниципальные учреждения, станции метро и прочее.
- В топе по количеству находятся крупные сетевые заведения, Шоколадница имеет самое большое количество заведений.

- Доля круглосуточных сетевых кофеен 7% против 1% у несетевых
- Доля круглосуточных кофеен довольно мала и составляет порядка 4% от всех заведений. Распределение круглосуточных кофеен по округам неравномерно и располагается между 1% и 7%, где 7% в Юго-Западном административном округе, 1% в Южном и Юго-Восточном
- Рейтинги сетевых кофеен сдвинуты в меньшую сторону, хотя имеют более стабильные показатели, тогда как несетевые кофейни еще получают более высокие оценки
- Рейтинг кофеен практически не меняется по округам
- Стоимость чашки капучино в несетевых заведениях имеет нормальную форму распределения со средним значением около 160 рублей, тогда как в сетевых мы четко видим сегментацию по ценовым категориям с пиками которые соответствуют стандартной стоимости капучино, где в верхней категории мы имеем пик на уровне 250 рублей в нижней два пика в 60 и 90 рублей.
- Средняя стоимость чашки капучино неравномерно распределена по округам Москвы, так Центральный и Юго-западный округа имеют более высокую среднюю стоимость в районе 190 рублей, тогда как средняя стоимость в Восточном округе равна 135 рублям
- Стоимость чашки капучино резко снижается с 220 до 160 рублей первые 6 километров из центра, далее стабилизируется на уровне своего медианного значения, причем изменение цены в несетевых заведения происходит более плавно, тогда как в сетевых мы видим ступени, которые отражают отсутствие или наличие заведений с дорогим кофе в указанном радиусе.
- Среднее количество посадочных мест в кофейнях равно 80. Сетевые заведения имеют небольшой сдвиг в большую сторону в распределении количества посадочных мест
- Доля сетевых заведений имеющих по близости торговый центр равна 65%, тогда как у несетевых этот показатель равен 47%
- При более глубоком анализе количества посадочных мест можно заметить резкое увеличение их медианного значения с 66 без торгового центра proximity до 96 с его наличием в целевой группе сетевых кофеен, что свидетельствует о том что генерируемый торговым центром трафик положительно сказывается на проходимости в них

Рекомендации по выбору места для открытия заведений общественного питания:

- Первый шаг заключается в создании концепта и определении целевой аудитории нашего заведения
- Вторым шагом необходимо определить коммуникационную стратегию по продаже нашего концепта целевой аудитории, привлечь маркетологов для этих целей
- Далее необходимо локализовать место до района
- После этого необходимо определить точки максимальной концентрации целевой аудитории, это могут быть парки, муниципальные заведения, остановки метро, торговые центры и прочее.
- Далее локализуем наш выбор на лучшей улице в желаемом районе с точки зрения инфраструктуры и транспортной доступности
- Затем необходимо провести анализ на наличие конкурентов в нашей целевой аудитории
- Последний этап выбор помещения. Приоритет отдаем помещениям на первой линии нашей улицы, его фасад должен, по возможности, выходить на нее, для лучшего привлечения клиентов. Так же желательно наличие парковочных мест
- Также параметры как количество посадочных мест, средний чек на который можно рассчитывать, а также средняя стоимость чашки капучино закладываются после выбора места на основе нашего анализа

Рекомендации по выбору помещения для концептуальной идеи кофейни "Central Perk":

- Целевой аудиторией концепции "Central Perk" будет типичный среднестатистический житель города, округа либо района, то есть самая широкая аудитория

- Для определения правильной стратегии коммуникации будет верным провести локальное социологическое исследование на анализ состава населения желаемой локации. Мы должны попасть в центровое место притяжения для таких людей в указанном районе
- Для примера возьмем Ломоносовский район. Здесь одним из лучших мест будет Ленинский проспект между остановками "Строителей" и "Ленинской 93". Где большое количество учебных заведений, два парка, хорошая пешеходная зона, транспортная доступность и наличие парковки.
- По нашим данным на этом участке нет сетевых заведений с которыми тяжело конкурировать, а из несетевых мы имеем порядка 3 заведений, что потенциально большого трафика не так много
- Помещение выберем с фасадом на Ленинский проспект или, что лучше, в сторону парка "Надежда", плюс, по возможности, для такого помещения будет наличие витринных окон. Площадь должна позволять установить порядка 90 посадочных мест. Концептуально будет верно сделать теплый ламповый интерьер который будет подталкивать к дружеским беседам. График желаемой работы будет с 7 до 7 в будни и с 10 до 10 в выходные. Стоимость чашки капучино в нашем месте

[к содержанию](#)