

Table of Contents

- 1 Открытие файла, знакомство с данными
- 2 Знакомство с данными. Предобработка
 - 2.1 Предобработка столбцов
 - 2.1.1 name
 - 2.1.2 category
 - 2.1.3 Географические данные: address, district, lat, lng
 - 2.1.4 hours
 - 2.1.5 rating
 - 2.1.6 Цены: price, avg_bill, middle_avg_bill, middle_coffee_cup
 - 2.1.7 chain
 - 2.1.8 seats
 - 2.2 Добавление необходимых столбцов
 - 2.3 Выводы
- 3 Анализ данных
 - 3.1 Категории заведений
 - 3.2 Число посадочных мест по категориям
 - 3.3 Сетевые заведения
 - 3.3.1 Соотношение сетевых и несетевых заведений
 - 3.3.2 Какие категории заведений чаще являются сетевыми?
 - 3.3.3 Топ-15 сетевых заведений
 - 3.3.4 Дополнительный анализ Топ-15 сетей
 - 3.3.4.1 Рейтинг
 - 3.3.4.2 Цены
 - 3.3.4.3 Распределение по округам
 - 3.4 Распределение по административным округам
 - 3.5 Средний рейтинг по категориям
 - 3.6 Средний рейтинг заведений по округам: фоновая картограмма
 - 3.7 Все заведения на карте
 - 3.8 Топ-15 улиц по количеству заведений
 - 3.9 Улицы с только одним объектом общепита
 - 3.10 Средние чеки
 - 3.10.1 Средние чеки по категориям
 - 3.10.2 Средние чеки по округам
 - 3.11 Дополнительный анализ
 - 3.11.1 Круглосуточные заведения
 - 3.11.2 Заведения с низкими рейтингами
 - 3.12 Выводы
- 4 Детализация исследования: открытие кофейни
 - 4.1 Кофейни по округам
 - 4.2 Круглосуточные кофейни
 - 4.3 Рейтинги кофеен по округам
 - 4.4 Дополнительный анализ
 - 4.4.1 Средняя стоимость чашки капучино
 - 4.4.2 Средний чек
 - 4.4.3 Посадочные места

- 4.4.4 Сетевые кофейни
- 4.4.5 Рейтинги
- 4.5 Где лучше открыть кофейню?
- 5 Выводы, рекомендации по открытию кофейни
- 6 Презентация

Рынок заведений общественного питания Москвы

Анализ рынка общепита для инвесторов из фонда «Shut Up and Take My Money», нацеленных на открытие заведения общественного питания в Москве.

Наша задача - подготовить исследование рынка Москвы, найти интересные особенности и презентовать полученные результаты, которые в будущем помогут в выборе подходящего инвесторам места.

План работы:

1. Открытие файла и знакомство с данными

2. Предобработка данных:

- обработка пропусков, дубликатов
- приведение данных к нужному типу
- изучение данных в каждом столбце, анализ возможных выбросов и аномалий
- создание новых столбцов, необходимых для дальнейшей работы

3. Анализ данных

- исследование объектов по категориям
- исследование количества посадочных мест по категориям
- оценка соотношения сетевых и несетевых заведений
 - оценка сетевых заведений по категориям
- определение топ-15 заведений Москвы по количеству заведений
- исследование заведений по административным округам
- оценка распределения средних рейтингов по категориям заведений
- построение картограммы со средним рейтингом заведений каждого района
- построение картограммы со всеми заведениями
- определение топ-15 улиц по числу заведений
- определение улиц, на которых находится только один объект общепита
- расчет медианной стоимости средних чеков для каждого округа
 - анализ зависимости цен от удаленности от центра
- в зависимости от результатов проведем дополнительный анализ по интересующим вопросам

4. Детализация исследования, рекомендации по открытию нового заведения

Ответим на следующие вопросы:

- Сколько всего кофеен в датасете? В каких районах их больше всего, каковы особенности их расположения?
- Есть ли круглосуточные кофейни?
- Какие у кофеен рейтинги? Как они распределяются по районам?

- На какую стоимость чашки капучино стоит ориентироваться при открытии и почему?

5. Подготовка презентации

Для анализа нам доступен датасет с заведениями общественного питания Москвы, составленный на основе данных сервисов Яндекс Карты и Яндекс Бизнес на лето 2022 года.

Описание данных:

- name — название заведения;
- address — адрес заведения;
- category — категория заведения, например «кафе», «пиццерия» или «кофейня»;
- hours — информация о днях и часах работы;
- lat — широта географической точки, в которой находится заведение;
- lon — долгота географической точки, в которой находится заведение;
- rating — рейтинг заведения по оценкам пользователей в Яндекс Картах (высшая оценка — 5.0);
- price — категория цен в заведении, например «средние», «ниже среднего», «выше среднего» и так далее;
- avg_bill — строка, которая хранит среднюю стоимость заказа в виде диапазона, например:
 - «Средний счёт: 1000–1500 ₽»;
 - «Цена чашки капучино: 130–220 ₽»;
 - «Цена бокала пива: 400–600 ₽».
 - и так далее;
- middle_avg_bill — число с оценкой среднего чека, которое указано только для значений из столбца avg_bill, начинающихся с подстроки «Средний счёт»:
 - Если в строке указан ценовой диапазон из двух значений, в столбец войдёт медиана этих двух значений.
 - Если в строке указано одно число — цена без диапазона, то в столбец войдёт это число.
 - Если значения нет или оно не начинается с подстроки «Средний счёт», то в столбец ничего не войдёт.
- middle_coffee_cup — число с оценкой одной чашки капучино, которое указано только для значений из столбца avg_bill, начинающихся с подстроки «Цена одной чашки капучино»:
 - Если в строке указан ценовой диапазон из двух значений, в столбец войдёт медиана этих двух значений.
 - Если в строке указано одно число — цена без диапазона, то в столбец войдёт это число.
 - Если значения нет или оно не начинается с подстроки «Цена одной чашки капучино», то в столбец ничего не войдёт.
- chain — число, выраженное 0 или 1, которое показывает, является ли заведение сетевым (для маленьких сетей могут встречаться ошибки):
 - 0 — заведение не является сетевым
 - 1 — заведение является сетевым
- district — административный район, в котором находится заведение, например Центральный административный округ;
- seats — количество посадочных мест.

Открытие файла, знакомство с данными

In [1]:

```
# импорт библиотек
import pandas as pd
import matplotlib.pyplot as plt
```

```

plt.style.use('ggplot')
%config InlineBackend.figure_format = 'retina'
import seaborn as sns
import plotly
import plotly.express as px
from plotly import graph_objects as go
from plotly.subplots import make_subplots
import plotly.io as io
io.templates.default = 'ggplot2'

import plotly.io as pio
pio.renderers.default = "png"
svg_renderer = pio.renderers["png"]
svg_renderer.scale = 1.2

from pandas.plotting import register_matplotlib_converters
pd.options.display.max_colwidth = 130

!pip install folium
import json
import folium
from folium import Map, Choropleth, Marker
from folium.plugins import MarkerCluster

```

Requirement already satisfied: folium in c:\users\hp\anaconda4\lib\site-packages (0.14.0)
Requirement already satisfied: numpy in c:\users\hp\anaconda4\lib\site-packages (from folium) (1.21.5)
Requirement already satisfied: jinja2>=2.9 in c:\users\hp\anaconda4\lib\site-packages (from folium) (2.11.3)
Requirement already satisfied: branca>=0.6.0 in c:\users\hp\anaconda4\lib\site-packages (from folium) (0.6.0)
Requirement already satisfied: requests in c:\users\hp\anaconda4\lib\site-packages (from folium) (2.28.1)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\hp\anaconda4\lib\site-packages (from jinja2>=2.9->folium) (2.0.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\hp\anaconda4\lib\site-packages (from requests->folium) (3.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\hp\anaconda4\lib\site-packages (from requests->folium) (2022.9.14)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\hp\anaconda4\lib\site-packages (from requests->folium) (1.26.11)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\hp\anaconda4\lib\site-packages (from requests->folium) (2.0.4)

In [2]:

```

try:
    data = pd.read_csv(r'\Users\Hp\Desktop\Практикум\project_may\moscow_places.csv')

except:
    data = pd.read_csv('https://code.s3.yandex.net/datasets/moscow_places.csv')

```

In [3]:

```

display(data.info())
display(data.head())
data.describe()

```

```

<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

	name	category	address	district	hours	lat	lng	rating	price
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	выс средне
2	Хазри	кафе	Москва, Клязьминская улица, 15	Северный административный округ	пн-чт 11:00– 02:00; пт,сб 11:00– 05:00; вс 11:00–02:00	55.889146	37.525901	4.6	средн
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
4	Илья Марко	пиццерия	Москва, Правобережная улица, 15	Северный административный округ	ежедневно, 10:00–22:00	55.881166	37.449357	5.0	средн

Out[3]:

	lat	lng	rating	middle_avg_bill	middle_coffee_cup	chain	seats
count	8406.000000	8406.000000	8406.000000	3149.000000	535.000000	8406.000000	4795.000000
mean	55.750109	37.608570	4.229895	958.053668	174.721495	0.381275	108.421689
std	0.069658	0.098597	0.470348	1009.732845	88.951103	0.485729	122.833396
min	55.573942	37.355651	1.000000	0.000000	60.000000	0.000000	0.000000
25%	55.705155	37.538583	4.100000	375.000000	124.500000	0.000000	40.000000
50%	55.753425	37.605246	4.300000	750.000000	169.000000	0.000000	75.000000
75%	55.795041	37.664792	4.400000	1250.000000	225.000000	1.000000	140.000000
max	55.928943	37.874466	5.000000	35000.000000	1568.000000	1.000000	1288.000000

Набор данных содержит 8406 наблюдений, в некоторых столбцах встречаются пропуски. Среди числовых столбцов встречаются аномалии - максимальные средний чек 35.000 и число

посадочных мест 1288. Приступим к предобработке и познакомимся с данными в столбцах поближе.

Знакомство с данными. Предобработка

Предобработка столбцов

Предстоит работа с 13 столбцами, данные очень разнородные, поэтому есть смысл рассмотреть каждый столбец отдельно. Проверим:

- наличие неявных дубликатов
- корректность хранящейся информации в столбцах
- корректность типов данных
- наличие выбросов и аномалий
- для некоторых столбцов бегло посмотрим на распределение данных

name

In [4]: # функция, вывод числа пропусков в столбце

```
def isna_count (col):  
    return print('Количество пропусков в {col} : {isna_count}'.format(col=col, isna_count=dat  
#функция, вывод числа уникальных значений:  
def uniq_count (col):  
    return print('Количество уникальных значений в {col} : {uniq_count}'.format(col=col, uniq
```

In [5]: isna_count('name')
uniq_count('name')

Количество пропусков в name : 0
Количество уникальных значений в name : 5614

Пропусков в столбце нет, но можем встретить неявные дубликаты - одни и те же названия, написанные в разном регистре. Приведем все значения в столбце к нижнему регистру и вновь посчитаем число уникальных заведений

In [6]: data['name'] = data['name'].str.lower()
uniq_count('name')

Количество уникальных значений в name : 5512

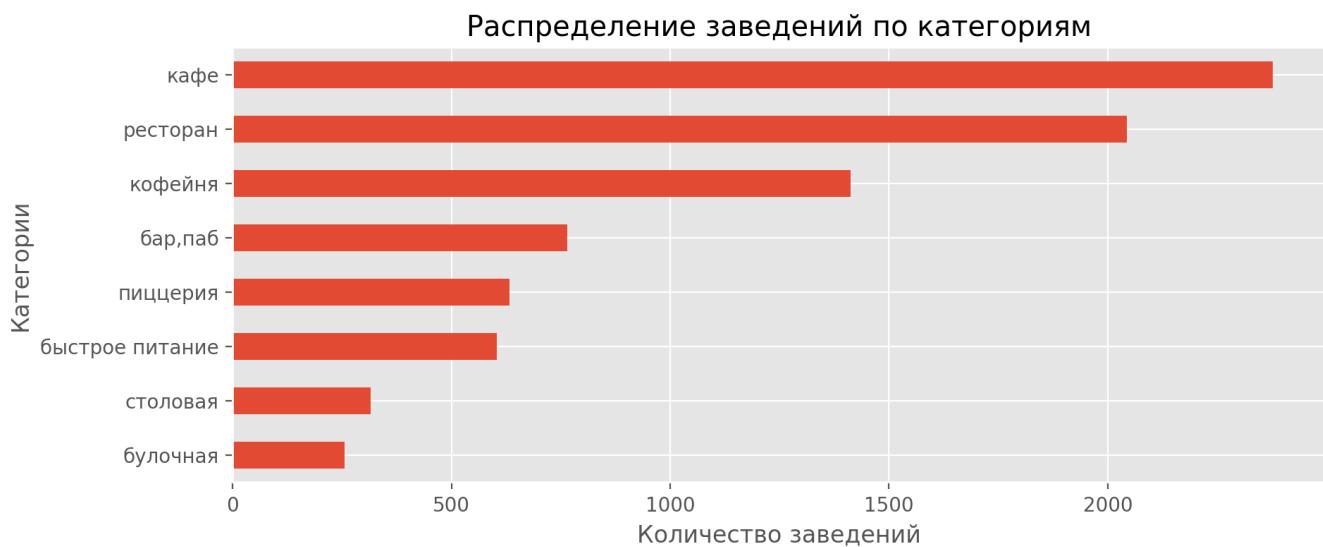
Число названий сократилось. **Столбец готов к работе**

category

In [7]: isna_count('category')
uniq_count('category')

```
data.groupby('category')['name'].count().sort_values().plot(kind='barh',  
                figsize=(10,4),  
                xlabel='Категории',  
                ylabel='Количество заведений',  
                title='Распределение заведений по  
plt.xlabel('Количество заведений');
```

Количество пропусков в category : 0
Количество уникальных значений в category : 8



В датасете 8 категорий заведений, чаще всего встречаются кафе и рестораны, реже всего столовые и булочные. Неявных дубликатов нет, в целом, распределение выглядит нормальным.

Столбец готов к работе

Географические данные: address, district, lat, lng

address

```
In [8]: isna_count('address')
uniq_count('address')
```

Количество пропусков в address : 0
 Количество уникальных значений в address : 5753

Пропусков в столбце address нет, по аналогии со столбцом name приведем все значения к нижнему регистру, чтобы избежать возможных неявных дубликатов

```
In [9]: data['address'] = data['address'].str.lower()
uniq_count('address')
```

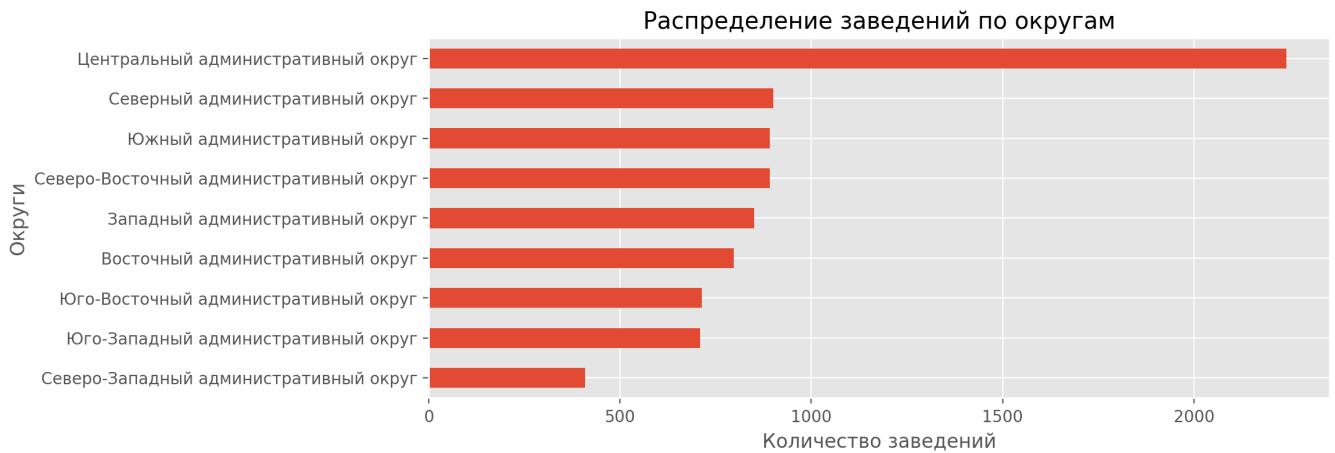
Количество уникальных значений в address : 5752

Встречался один неявный дубликат. Столбец готов к работе

district

```
In [10]: isna_count('district')
uniq_count('district')
data.groupby('district')['name'].count().sort_values().plot(kind='barh',
                                                               figsize=(10,4),
                                                               xlabel='Округи',
                                                               title='Распределение заведений по округам');
plt.xlabel('Количество заведений');
```

Количество пропусков в district : 0
 Количество уникальных значений в district : 9



Пропусков и неявных дубликатов в столбце нет, всего встречается 9 округов . Больше всего заведений в центре, что неудивительно. Распределение выглядит нормально. **Столбец готов к работе**

lat, lng не содержат пропусков, данные хранятся в корректном типе float64. **Столбцы готовы к работе**

hours

```
In [11]: isna_count('hours')
uniq_count('hours')
print()
print('Значения в столбце hours:', data['hours'].sort_values().unique())
```

Количество пропусков в hours : 536
Количество уникальных значений в hours : 1308

Значения в столбце hours: ['Нет информации'
'вт 08:30-17:00;ср,чт 12:00-20:30;пт 08:30-17:00;сб 09:00-16:30'
'вт 13:00-21:00;ср 11:00-20:00;чт 13:00-21:00;пт-вс 11:00-20:00' ...
'чт круглосуточно,перерыв 10:00-20:00;сб круглосуточно'
'чт-вс 20:00-06:00' nan]

Графики работы у заведений сильно отличаются. В столбце есть 536 пропусков , заменим из на значение-заглушку, которое уже встречается в столбце - "Нет информации"

```
In [12]: data['hours'].fillna('Нет информации', inplace = True)
isna_count('hours')
```

Количество пропусков в hours : 0

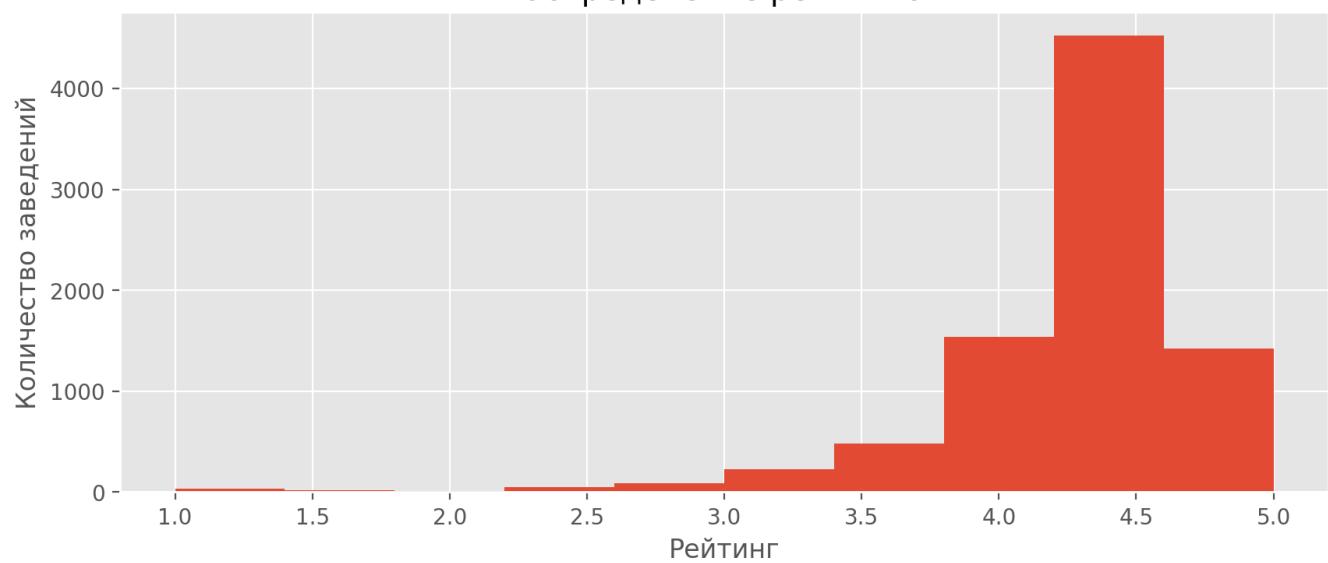
Пропуски обработаны. **Столбец готов к работе**

rating

```
In [13]: isna_count('rating')
uniq_count('rating')
data['rating'].plot(kind='hist',
                     figsize=(10,4),
                     title='Распределение рейтингов');
plt.xlabel('Рейтинг');
plt.ylabel('Количество заведений');
```

Количество пропусков в rating : 0
Количество уникальных значений в rating : 41

Распределение рейтингов



Пропусков в столбце нет, данные хранятся в корректном типе. График выглядит нормально, заведения оцениваются от 1 до 5, большинство заведений имеет оценку ~4.3, очень мало заведений с оценкой ниже 3

Столбец готов к работе.

Цены: price, avg_bill, middle_avg_bill, middle_coffee_cup

price

```
In [14]:  
isna_count('price')  
uniq_count('price')  
print('Уникальные значения в столбце:', data['price'].unique().tolist())
```

Количество пропусков в price : 5091

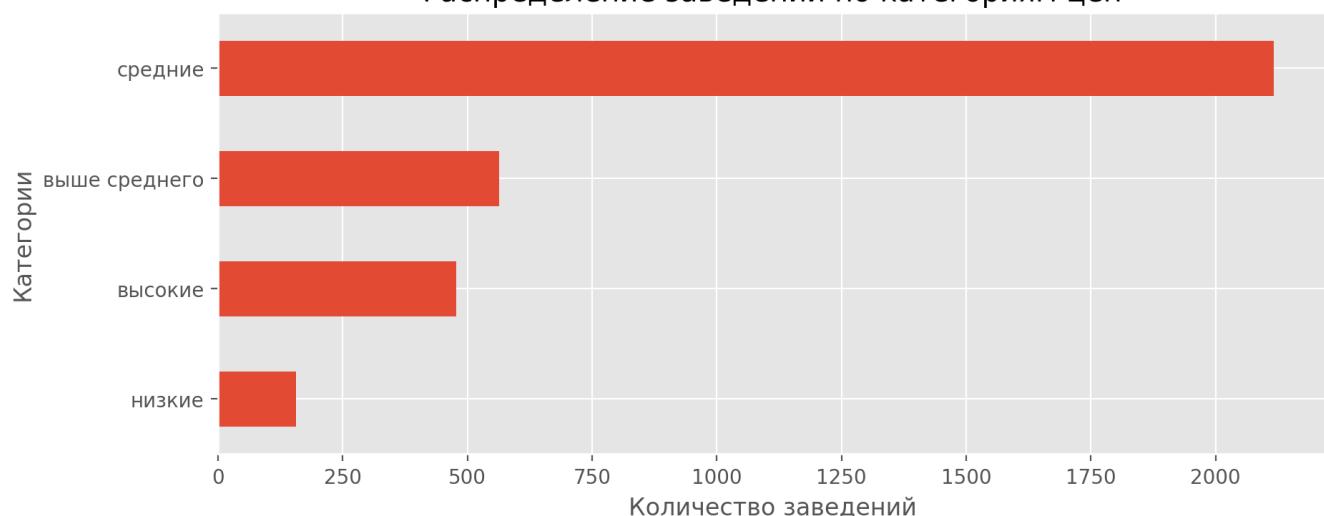
Количество уникальных значений в price : 5

Уникальные значения в столбце: [nan, 'выше среднего', 'средние', 'высокие', 'низкие']

Столбец содержит **очень большое число пропусков** - больше 50%. Встречаются 4 категории средних цен. Заменим пропуски на значение-заглушку и посмотрим как распределены заведения по ценовой категории.

```
In [15]:  
data['price'].fillna('Нет информации', inplace = True)  
data.query('price != "Нет информации"').groupby('price')[['name']]\  
    .count().sort_values().plot(kind='barh',  
                                figsize=(10,4),  
                                title='Распределение заведений по категориям цен');  
plt.xlabel('Количество заведений');  
plt.ylabel('Категории');
```

Распределение заведений по категориям цен



Абсолютное большинство заведений относятся к среднему бюджету. **Столбец готов к работе**

avg_bill

```
In [16]: isna_count('avg_bill')  
uniq_count('avg_bill')
```

Количество пропусков в avg_bill : 4590
Количество уникальных значений в avg_bill : 898

Снова пропущено более 50% значений, заменим пропуски на заглушку

```
In [17]: data['avg_bill'].fillna('Нет информации', inplace = True)  
isna_count('avg_bill')
```

Количество пропусков в avg_bill : 0

middle_avg_bill, middle_coffee_cup

```
In [18]: isna_count('middle_avg_bill')  
isna_count('middle_coffee_cup')
```

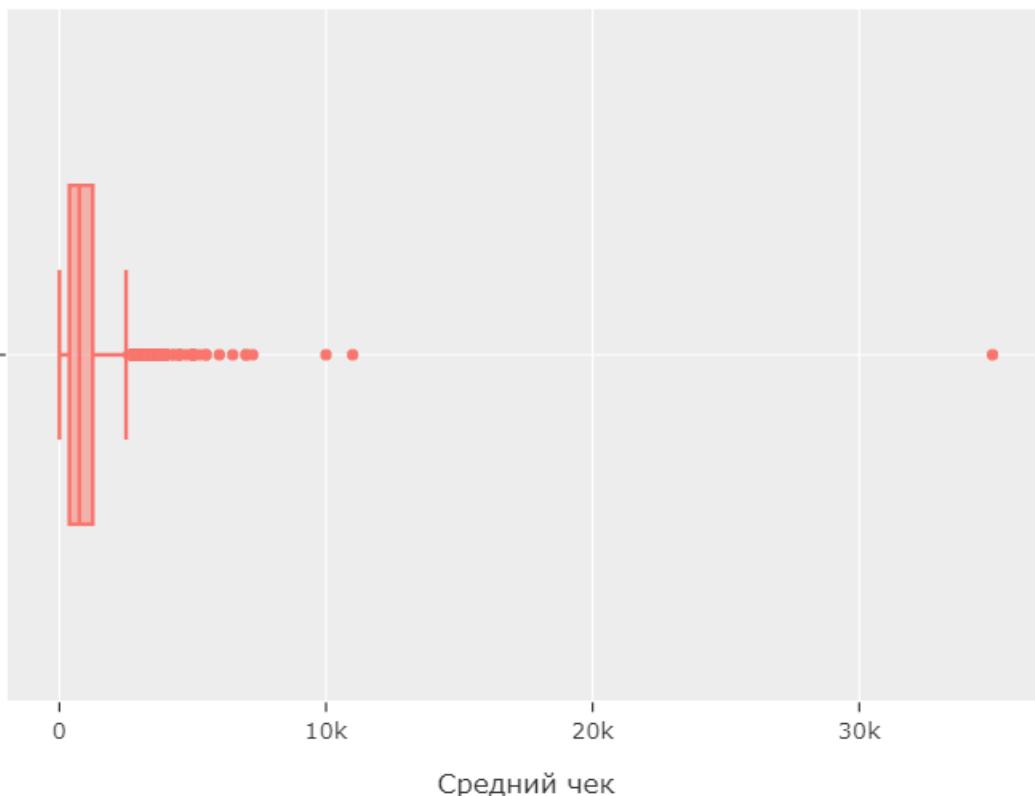
Количество пропусков в middle_avg_bill : 5257
Количество пропусков в middle_coffee_cup : 7871

Пропуски в обоих столбцах связаны с пропусками в avg_bill, из которого и "подтягиваюся" значения из этих столбцов. Не стоит обрабатывать такое большое число пропусков медианными значениями. Данные хранятся в корректном типе float, оставим NaN, этот тип данных позволит проводить математические операции со столбцом.

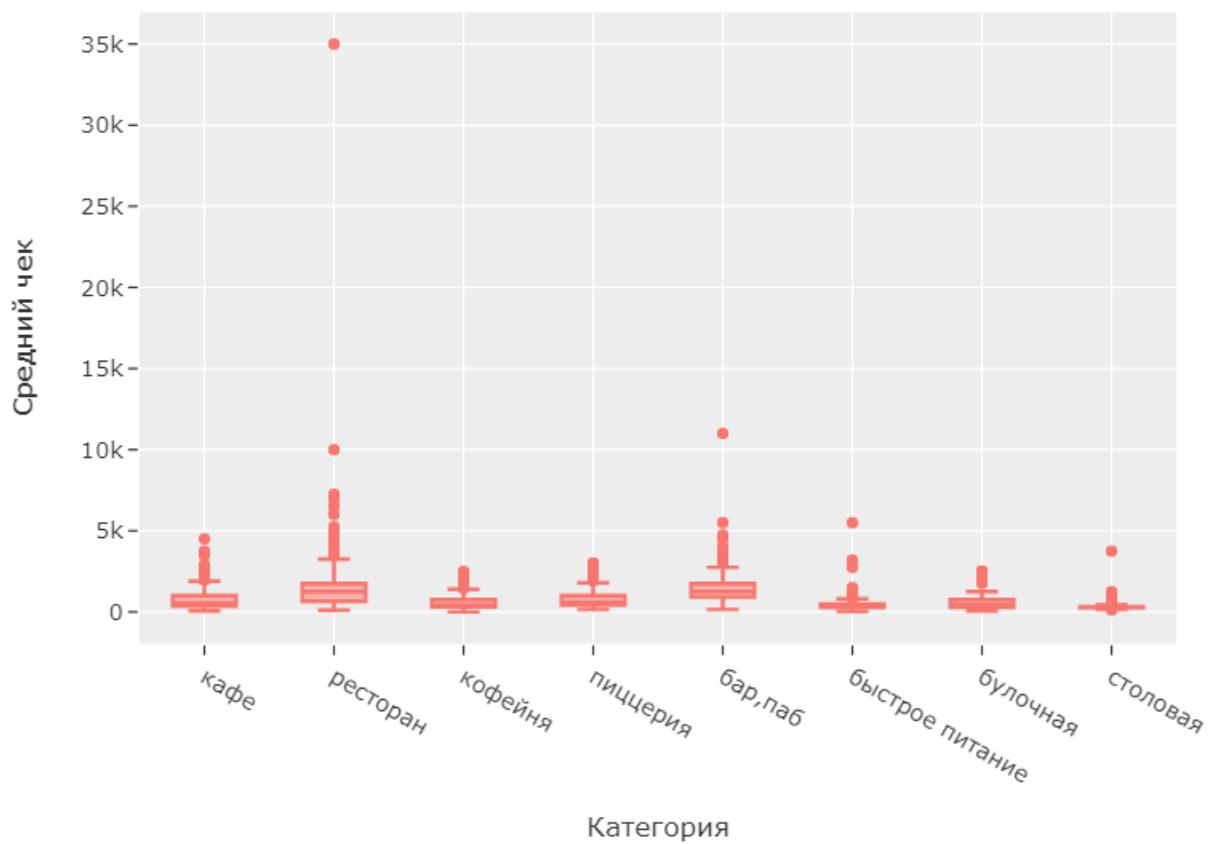
На этапе знакомства с датасетом, мы обратили внимание, что столбцы содержат выбросы. Построим "ящики с усами" по категориям заведений для столбца middle_avg_bill

```
In [19]: fig = px.box(data, x='middle_avg_bill')  
fig.update_layout(title='Средний чек: размах данных')  
fig.update_xaxes(title='Средний чек')  
fig.update_yaxes(title='')  
fig.show()  
fig = px.box(data,x='category', y='middle_avg_bill')  
fig.update_layout(title='Средний чек по категориям: размах данных')  
fig.update_xaxes(title='Категория')  
fig.update_yaxes(title='Средний чек')  
fig.show()
```

Средний чек: размах данных



Средний чек по категориям: размах данных



Видим ресторан со средним чеком 35.000, что, конечно, мало похоже на правду. Также можем отметить, что встречаются выбросы и в других категориях: бары с чеком 11.000, "быстрое питание" 5500, столовая 3750.

На данном этапе зафиксируем наличие выбросов, но не будем удалять заведения (в строках может содержаться другая интересующая нас информация), при более детальном анализе

отбросим выбивающиеся значения (ориентируемся на верхнюю границу в 10.000).

Средняя цена чашки кофе указана всего у 535 заведений, построим общую диаграмму размаха по столбцу middle_coffee_cup

```
In [20]: fig = px.scatter(data, y="middle_coffee_cup")
fig.update_layout(title='Средняя стоимость чашки кофе: размах данных')
fig.update_xaxes(title='Заведения')
fig.update_yaxes(title='Средняя стоимость')
fig.show()
```



Здесь информация уже выглядит правдоподобнее, цена чашки кофе в пределах 400Р, есть один выброс со средней ценой 1568, поступим так же, как и со столбцом middle_avg_bill

Столбцы готовы к работе

chain

```
In [21]: isna_count('chain')
data['chain'].value_counts()
```

Количество пропусков в chain : 0

```
Out[21]: 0    5201
1    3205
Name: chain, dtype: int64
```

Большинство наших заведений несетевые, пропусков в столбце нет, каких-то неожиданных значений тоже. Заменим тип данных на boolean.

```
In [22]: data['chain'] = data['chain'].astype('bool')
```

Столбец готов к работе

seats

In [23]:

```
isna_count('seats')  
uniq_count('seats')
```

Количество пропусков в seats : 3611

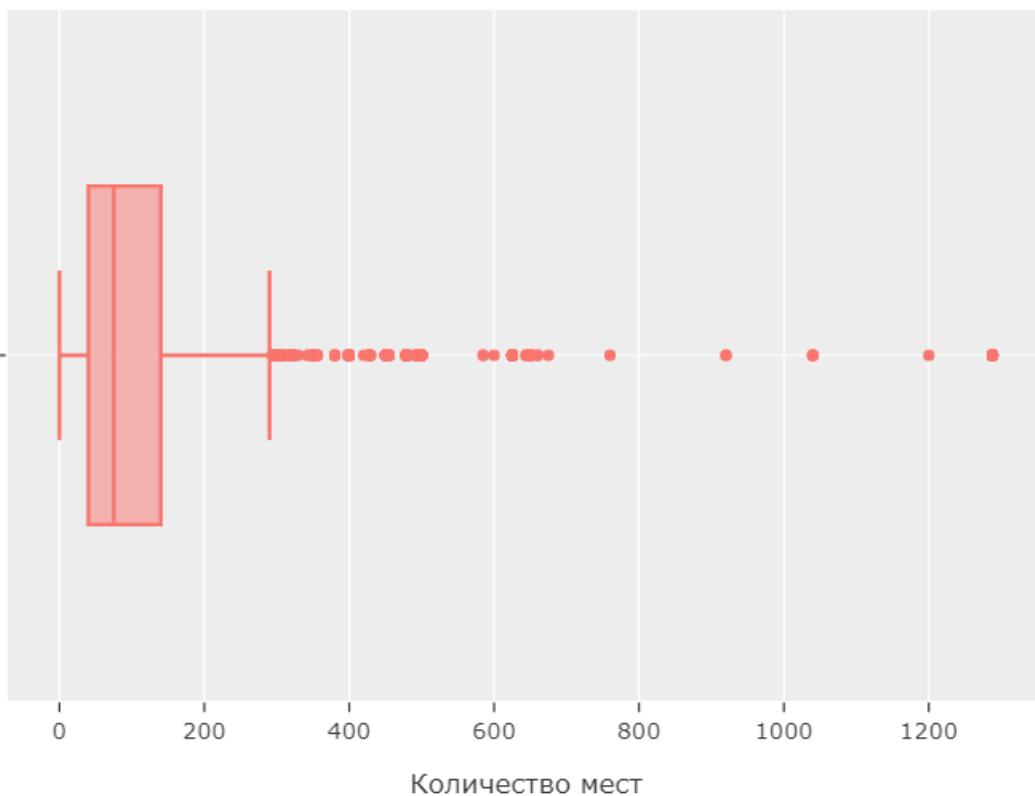
Количество уникальных значений в seats : 230

Достаточно большое число пропусков. Оставим NaN, чтобы иметь возможность проводить мат.операции со столбцом. Ранее заметили, что столбец содержит выбросы, построим "ящик с усами"

In [24]:

```
fig = px.box(data, x="seats")  
fig.update_layout(title='Количество посадочных мест: размах данных')  
fig.update_xaxes(title='Количество мест')  
fig.update_yaxes(title='')  
fig.show()
```

Количество посадочных мест: размах данных

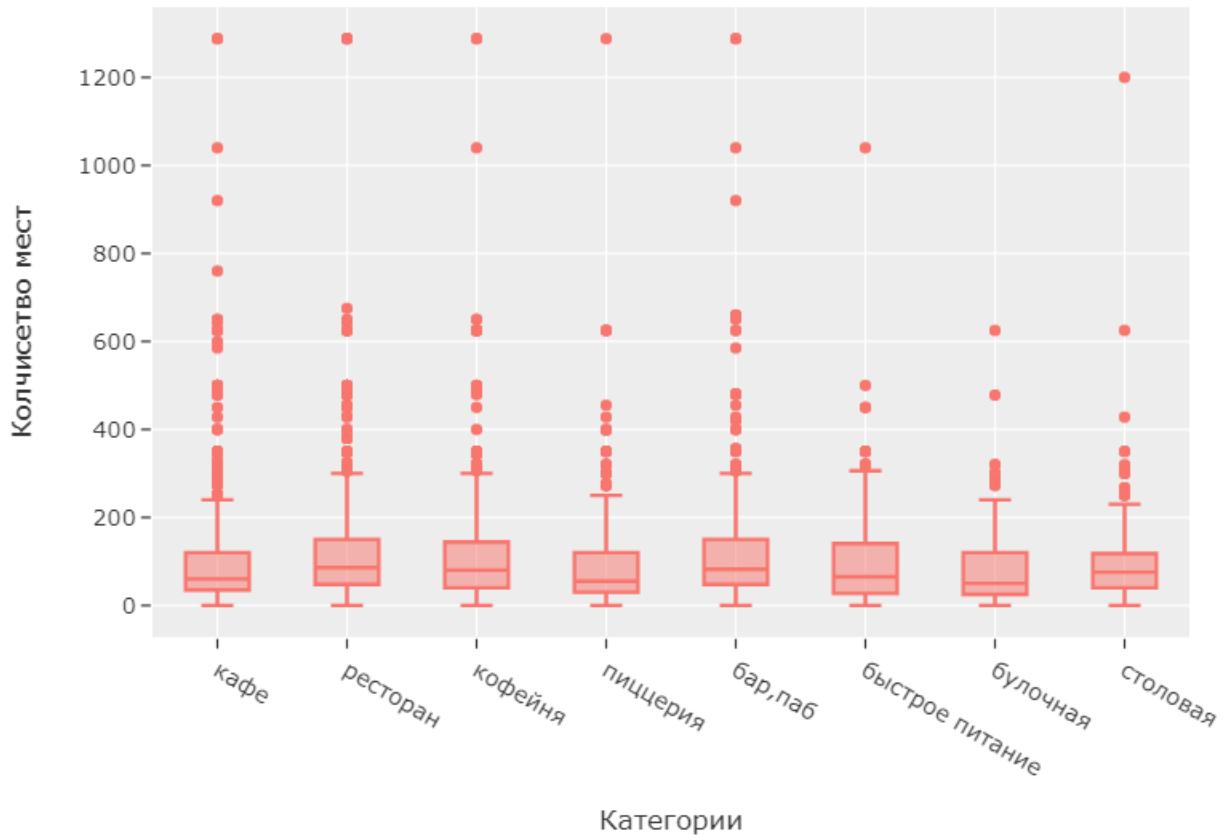


Верхняя граница нормальных значений 290, все что выше считается выбросами. Действительно, такие большие заведения в реальности встречаются редко. Посмотрим на диаграмму размаха по категориям

In [25]:

```
fig = px.box(data, x= "category", y="seats")  
fig.update_layout(title='Количество посадочных мест по категориям: размах данных')  
fig.update_xaxes(title='Категории')  
fig.update_yaxes(title='Количество мест')  
fig.show()
```

Количество посадочных мест по категориям: размах данных



Аномалии встречаются в каждой категории: настораживает как 1288 посадочных мест в ресторане, так и 625 в булочной, у большинства заведений достаточно много выбросов.

Предполагаю, что часть заведений находятся на фудкортах и в столбце `seats` указана вместимость самого футкорта. Сгруппируем данные по адресу, посчитаем количество заведений и отберем те адреса, по которым располагается три и более заведений с вместимостью > 290

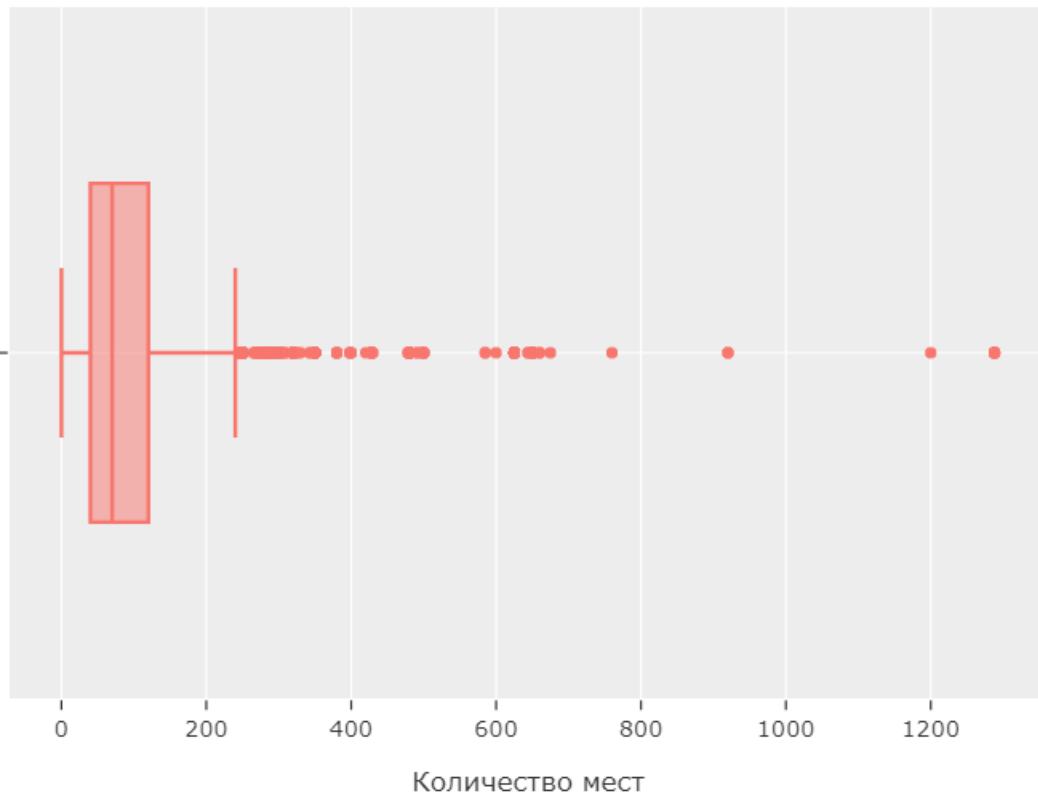
```
In [26]: unnormal_seats_rest = data.query('seats > 290').groupby('address')['name'].count().sort_value unnormal_seats_rest = unnormal_seats_rest.query('name > 2') unnormal_seats_rest_adress = unnormal_seats_rest['address'].tolist() print('Количество адресов, по которым расположено более двух больших заведений:', len(unnorma print('Суммарно заведений по таким адресам:', unnormal_seats_rest['name'].sum()))
```

Количество адресов, по которым расположено более двух больших заведений: 28
Суммарно заведений по таким адресам: 171

Два ресторана в одном здании с числом посадочных мест ~300 еще можно представить, но три и более заведений в одном доме и более - скорее аномалия. Такие заведения лучше отсечь. Удалять их не стоит, в сроках хранится другая интересующая нас информация. Установим для этих заведений условный маркер 9999, чтобы оставить тип данных в столбце числовым, но при этом в дальнейшем мы могли легко сделать срез по этому маркеру.

```
In [27]: data.loc[(data['address'].isin(unnormal_seats_rest_adress)) & (data['seats']>290), 'seats']=9 fig = px.box(data.query('seats != 9999'), x="seats") fig.update_layout(title='Количество посадочных мест: размах данных') fig.update_xaxes(title='Количество мест') fig.update_yaxes(title='') fig.show()
```

Количество посадочных мест: размах данных

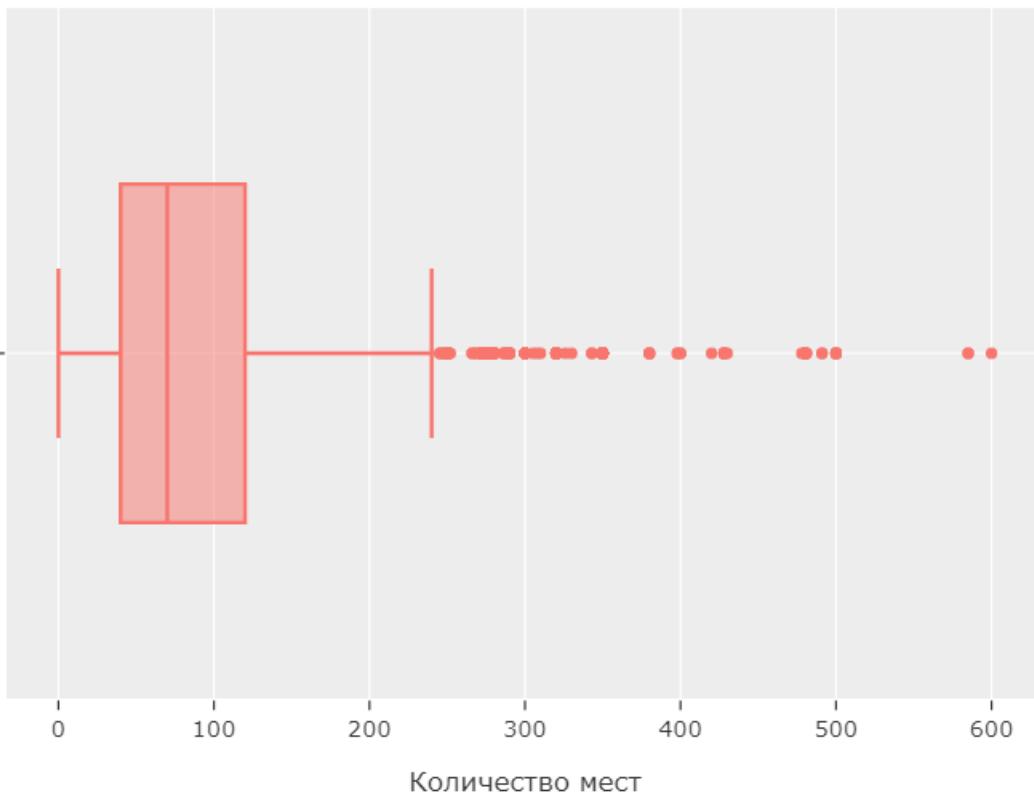


Даже установив маркер на предполагаемые фудкорты встречаем экстремально высокие значения. Рестораны с числом посадочных мест > 600 в наборе данных встречаются реже всего, даже с учетом банкетных залов сложно представить настолько большие заведения. Таким заведениям тоже установим условный маркер 9999

```
In [28]: data.loc[(data['seats']>600), 'seats']=9999
```

```
In [29]: fig = px.box(data.query('seats!=9999'), x='seats')
fig.update_layout(title='Количество посадочных мест: размах данных')
fig.update_xaxes(title='Количество мест')
fig.update_yaxes(title='')
fig.show()
```

Количество посадочных мест: размах данных



В дальнейшем анализе и построении визуализаций будем учитывать именно этот срез данных.

Столбец готов к работе

Проверим наличие полных дубликатов и вновь вызовем инфо для проверки внесенных изменений

```
In [30]: print('Число полных дубликатов:', data.duplicated().sum())
print()
data.info()
```

Число полных дубликатов: 0

```
<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              8406 non-null   object  
 5   lat                8406 non-null   float64 
 6   lng                8406 non-null   float64 
 7   rating             8406 non-null   float64 
 8   price              8406 non-null   object  
 9   avg_bill           8406 non-null   object  
 10  middle_avg_bill    3149 non-null   float64 
 11  middle_coffee_cup  535 non-null    float64 
 12  chain              8406 non-null   bool    
 13  seats              4795 non-null   float64 
dtypes: bool(1), float64(6), object(7)
memory usage: 862.1+ KB
```

Добавление необходимых столбцов

Для дальнейшей работы создадим два столбца:

- `street` с названиями улиц из столбца с адресом.
- `is_24_7` с обозначением, что заведение работает ежедневно и круглосуточно (24/7):
 - логическое значение `True` — если заведение работает ежедневно и круглосуточно;
 - логическое значение `False` — в противоположном случае.

```
In [31]: data['street'] = data['address'].str.split(', ').str[1]
data['is_24_7'] = data['hours'].str.contains('ежедневно, круглосуточно')
print('Круглосуточных заведений:', len(data.query('is_24_7 == True')))

data.sample(5)
```

Круглосуточных заведений: 730

Out[31]:

		name	category	address	district	hours	lat	lng	rating
6947		entree	булочная	москва, профсоюзная улица, 68, корп. 3	Юго-Западный административный округ	пн-пт 08:00– 22:00; сб,вс 09:00–22:00	55.664786	37.545599	4.3
8351		mr. john's	пиццерия	москва, 2-й грайвороновский проезд, 42к4	Юго-Восточный административный округ	пн-чт 10:00– 22:00; пт,сб 10:00– 23:00; вс 10:00–22:00	55.724671	37.744333	4.5
3169		cha cha	бар,паб	москва, улица мнёвники, 21	Северо-Западный административный округ	ежедневно, 10:00–23:30	55.773850	37.481051	4.4
1698		давайте карбонару	ресторан	москва, большая новодмитровская улица, 36, стр. 7	Северо-Восточный административный округ	ежедневно, 10:00–23:00	55.805816	37.584972	4.4
6230		больше, чем кофе	кофейня	москва, донская улица, 8	Центральный административный округ	пн-пт 08:00– 20:00; сб,вс 09:00–21:00	55.723466	37.606850	4.2

Столбцы успешно созданы.

Ранее мы привели все названия и адреса к нижнему регистру, проверим наличие неявных дубликатов по этим двум столбцам

In [32]:

```
print('Количество неявных дубликатов', len(data[data.duplicated(subset=['name', 'address'])]))
```

Количество неявных дубликатов 4

Удалим дубликаты

In [33]:

```
data = data.drop_duplicates(subset=['name', 'address'])
print('Оставшееся число записей в data:', len(data))
```

Оставшееся число записей в data: 8402

Выводы

- значения в столбцах с адресом и названием приведены к нижнему регистру для избежания неявных дубликатов
- все столбцы приведены к необходимому типу данных
- обработаны пропуски: в столбцах с категориальными данными вместо пропусков установлены значения-заглушки
- обработаны аномалии в столбце seats, выбивающиеся экстремально высокие значения заменены на условный маркер 9999
- дубликатов нет, столбцы названы корректно и хранят корректные данные
- добавлены столбцы street и is_24_7

В таблице содержатся аномалии и выбросы. Обнаружены аномальные высокие значения в столбцах middle_avg_bill, middle_coffee_cup, seats, **необходимо учитывать при дальнейшем анализе.**

Таблица готова к дальнейшему исследованию.

Анализ данных

- Какие категории заведений представлены в данных? Исследуйте количество объектов общественного питания по категориям: рестораны, кофейни, пиццерии, бары и так далее. Постройте визуализации. Ответьте на вопрос о распределении заведений по категориям.
- Исследуйте количество посадочных мест в местах по категориям: рестораны, кофейни, пиццерии, бары и так далее. Постройте визуализации. Проанализируйте результаты и сделайте выводы.
- Рассмотрите и изобразите соотношение сетевых и несетевых заведений в датасете. Каких заведений больше?
- Какие категории заведений чаще являются сетевыми? Исследуйте данные и ответьте на вопрос графиком.
- Сгруппируйте данные по названиям заведений и найдите топ-15 популярных сетей в Москве. Под популярностью понимается количество заведений этой сети в регионе. Постройте подходящую для такой информации визуализацию. Знакомы ли вам эти сети? Есть ли какой-то признак, который их объединяет? К какой категории заведений они относятся?
- Какие административные районы Москвы присутствуют в датасете? Отобразите общее количество заведений и количество заведений каждой категории по районам. Попробуйте проиллюстрировать эту информацию одним графиком.
- Визуализируйте распределение средних рейтингов по категориям заведений. Сильно ли различаются усреднённые рейтинги в разных типах общепита?
- Постройте фоновую картограмму (хороплет) со средним рейтингом заведений каждого района. Границы районов Москвы, которые встречаются в датасете, хранятся в файле admin_level_geotmap.geojson (скачать файл для локальной работы).
- Отобразите все заведения датасета на карте с помощью кластеров средствами библиотеки folium.
- Найдите топ-15 улиц по количеству заведений. Постройте график распределения количества заведений и их категорий по этим улицам. Попробуйте проиллюстрировать эту информацию одним графиком.
- Найдите улицы, на которых находится только один объект общепита. Что можно сказать об этих заведениях?

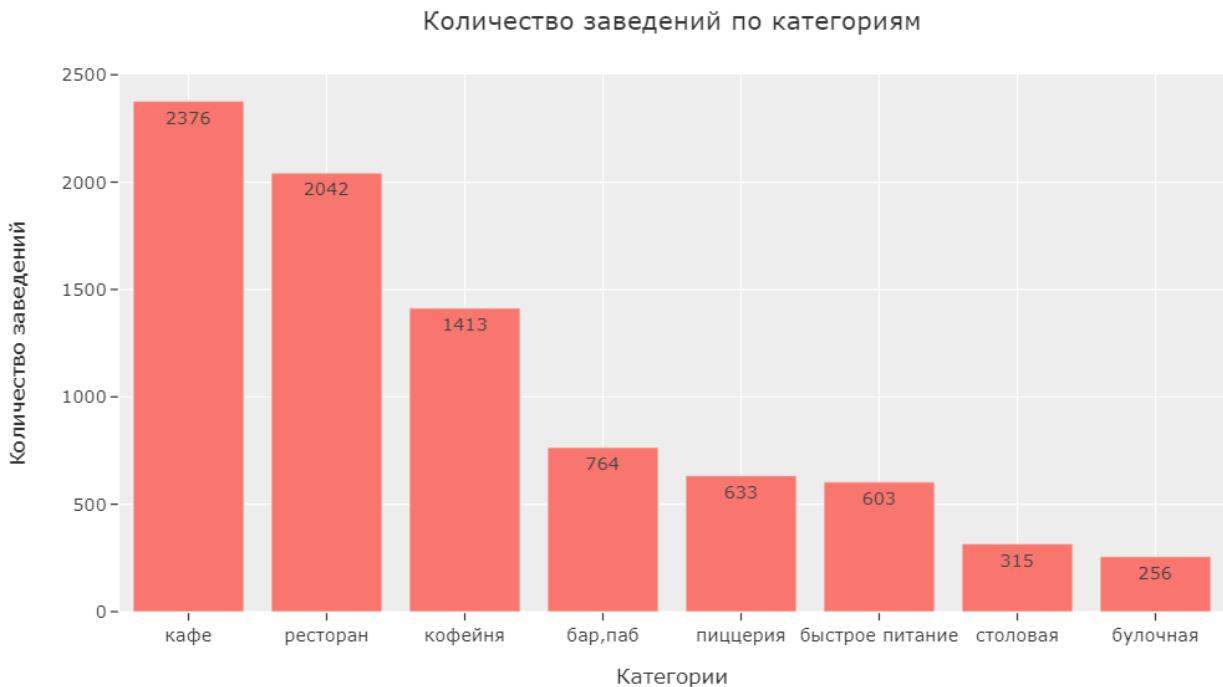
- Значения средних чеков заведений хранятся в столбце `middle_avg_bill`. Эти числа показывают примерную стоимость заказа в рублях, которая чаще всего выражена диапазоном. Посчитайте медиану этого столбца для каждого района. Используйте это значение в качестве ценового индикатора района. Постройте фоновую картограмму (хороплет) с полученными значениями для каждого района.
- Проанализируйте цены в центральном административном округе и других. Как удалённость от центра влияет на цены в заведениях?
- Необязательное задание: проиллюстрируйте другие взаимосвязи, которые вы нашли в данных. Например, по желанию исследуйте часы работы заведений и их зависимость от расположения и категории заведения. Также можно исследовать особенности заведений с плохими рейтингами, средние чеки в таких местах и распределение по категориям заведений.
- Соберите наблюдения по вопросам выше в один общий вывод.

Категории заведений

```
In [34]: # таблица: число заведений по категориям:
category_group = data.groupby('category')[['name']].count().sort_values(ascending=False).reset_index()
```

```
In [35]: fig = px.bar(category_group,
                  x='category', y='name', text='name')
fig.update_layout(title='Количество заведений по категориям',
                  title_x=0.5,
                  height=500, width=900)
fig.update_yaxes(title_text="Количество заведений")
fig.update_xaxes(title_text="Категории")

fig.show()
```



- **Кафе и рестораны - самые популярные категории заведений**, 2376 и 2042 заведения соответственно
- **Кофейня** - третий по популярности тип заведения, 1413 заведений

Доли остальных заведений существенно ниже, меньше 10%. Реже всего встречаются столовые и булочные

Число посадочных мест по категориям

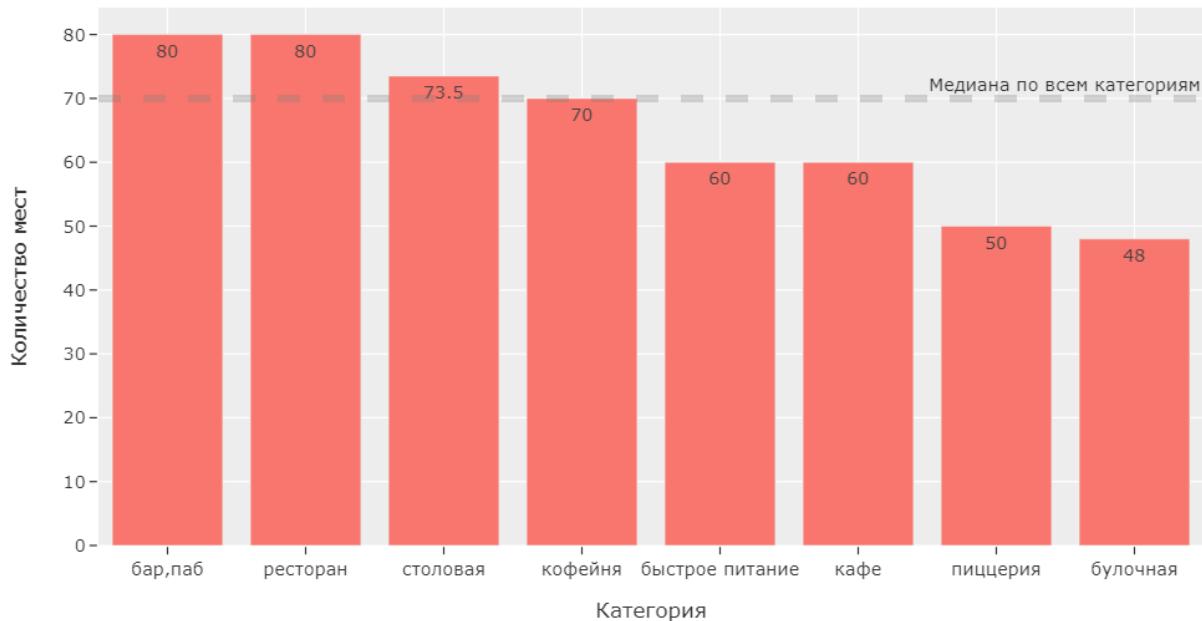
In [36]:

```
#графики по срезанным данным:
fig = px.bar(data.query('seats != 9999').groupby('category')['seats'].median().sort_values(as
    x='category', y='seats', text='seats')

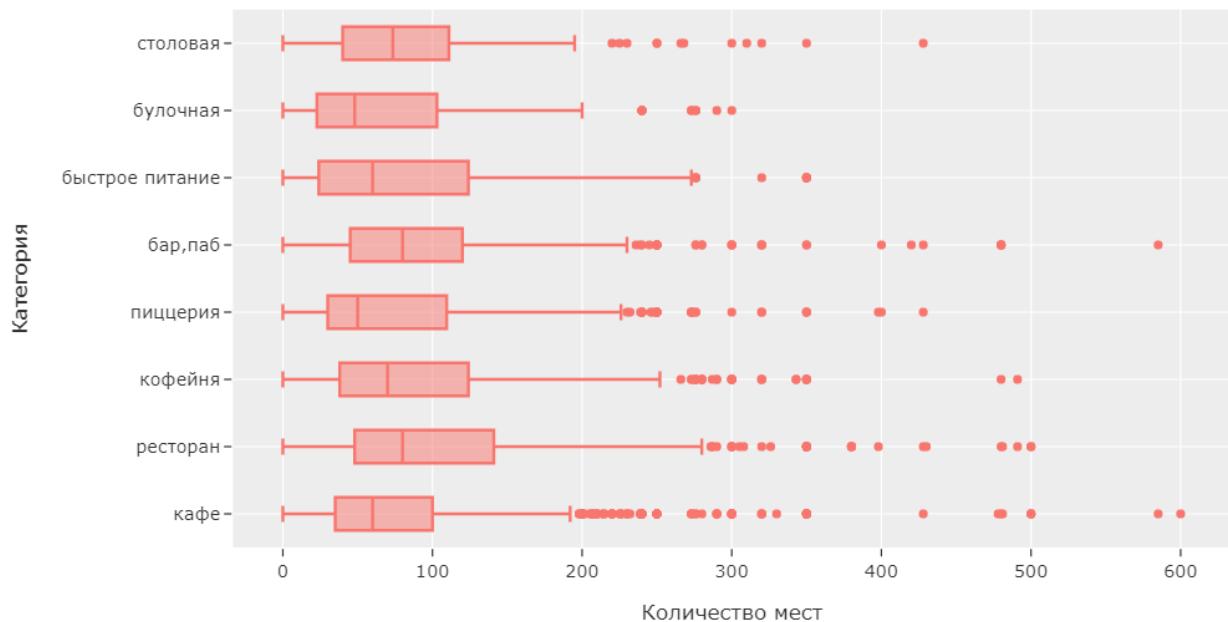
fig.update_layout(title='Медианное число посадочных мест по категориям',
                  title_x = 0.5,
                  height=500, width=900)
fig.update_yaxes(title_text = "Количество мест")
fig.update_xaxes(title_text = "Категория")
fig.add_hline(y=data.query('seats != 9999')['seats'].median(), line_width=5, line_dash="dash"
                  annotation_text="Медиана по всем категориям")
fig.show()

fig = px.box(data.query('seats !=9999'), y= "category", x="seats")
fig.update_layout(title='Количество посадочных мест по категориям',
                  title_x = 0.5,
                  height=500, width=900)
fig.update_yaxes(title_text = "Категория")
fig.update_xaxes(title_text = "Количество мест")
#fig.add_vline(x=data.query('seats != 9999')['seats'].median(), line_width=5, line_dash="dash"
fig.show()
```

Медианное число посадочных мест по категориям



Количество посадочных мест по категориям



Больше всего посадочных мест в ресторанах, барах(пабах) и столовых - медианные значение 80, 80, 73.5 соответственно, что выше общего для всех категорий медианного значения. Следом по вместимости идут кофейни - по 70 посадочных мест, меньше всего мест у булочных - медианное значение 48.

Большинство заведений имеют следующие диапазоны посадочных мест

- столовая: 40-111
- булочная: 22-103
- быстрое питание: 24-124
- бар, паб: 45-120
- пиццерия: 30-109
- кофейня 38-124
- ресторан: 48-141
- кафе: 35-100

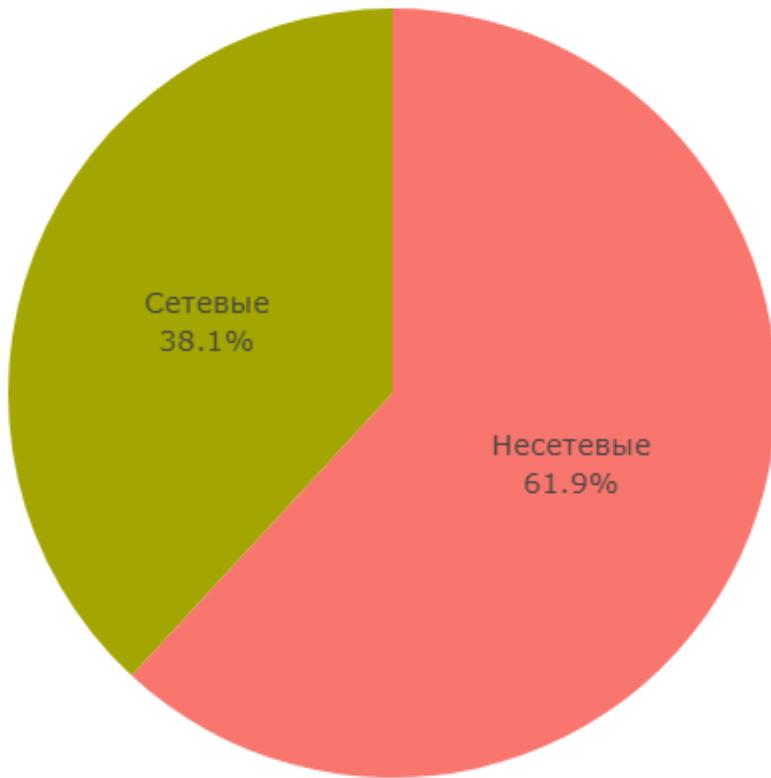
Более мелкие и более крупные заведения встречаются реже.

Сетевые заведения

Соотношение сетевых и несетевых заведений

```
In [37]: fig = go.Figure(data=[go.Pie(labels=['Несетевые', 'Сетевые'],
                                     values=data.groupby('chain')[['name']].count().reset_index()['name'].values)])
fig.update_layout(title='Соотношение сетевых и несетевых заведений',
                  title_x=0.5,
                  height=500, width=500, showlegend=False)
fig.update_traces(textinfo='label+percent')
fig.show()
```

Соотношение сетевых и несетевых заведений



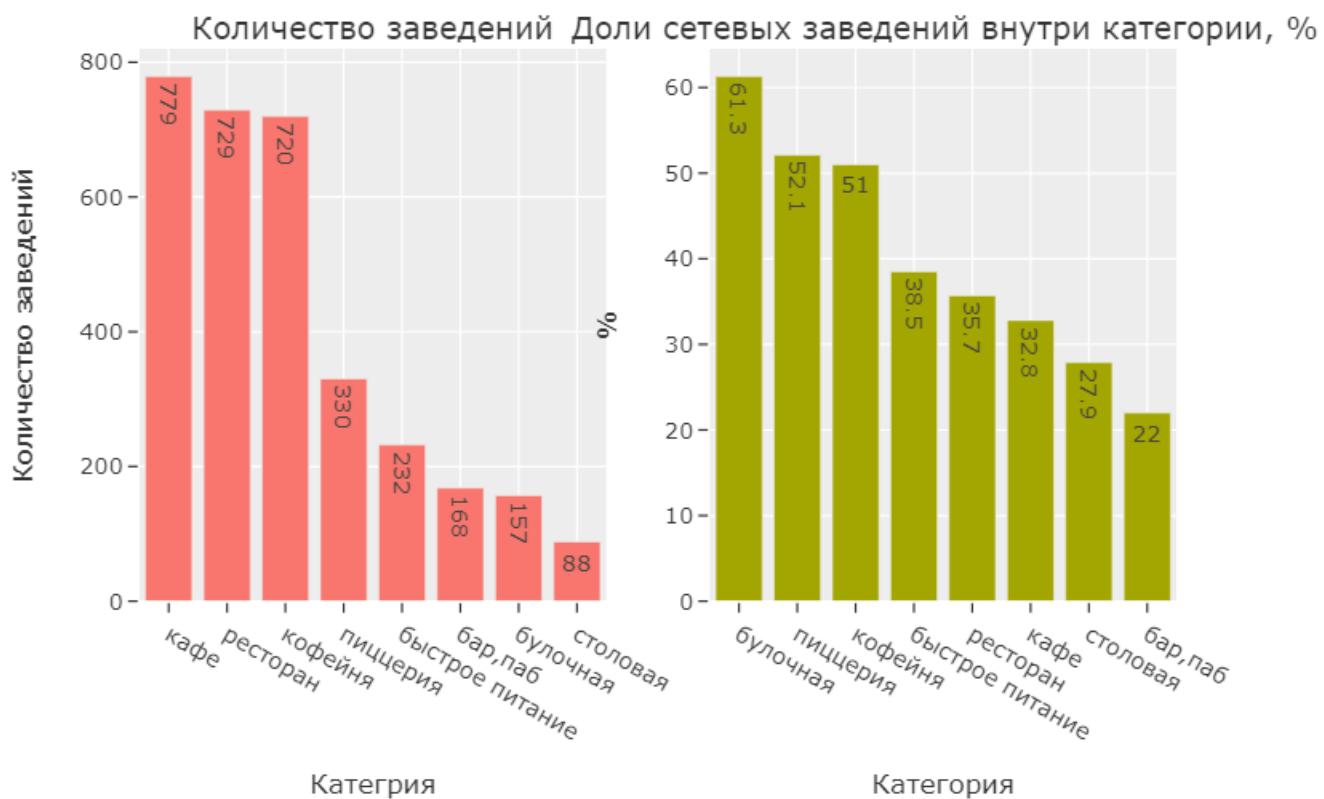
Доля сетевых заведений достаточно внушительная: 38.1%

Какие категории заведений чаще являются сетевыми?

```
In [38]: #таблица: общее число заведений, число сетевых и их долей по категориям:  
chain_rest = data.groupby('category').agg({'name':'count', 'chain':'sum'}).reset_index()  
chain_rest['ratio'] = round(((chain_rest['chain'] / chain_rest['name'])*100),1)  
chain_rest = chain_rest.rename(columns={'name':'total'})
```

```
In [39]: fig = make_subplots(rows=1, cols=2, specs=[[{'type':'bar'}, {'type':'bar'}]],  
                        subplot_titles=['Количество заведений', 'Доли сетевых заведений внутри ка  
  
fig.add_trace(go.Bar(x=chain_rest.sort_values(by='chain', ascending=False)[ 'category'],  
                     y=chain_rest.sort_values(by='chain', ascending=False)[ 'chain'],  
                     text=chain_rest.sort_values(by='chain', ascending=False)[ 'chain']),  
                     1, 1)  
  
fig.add_trace(go.Bar(x=chain_rest.sort_values(by='ratio', ascending=False)[ 'category'],  
                     y=chain_rest.sort_values(by='ratio', ascending=False)[ 'ratio'],  
                     text=chain_rest.sort_values(by='ratio', ascending=False)[ 'ratio']),  
                     1, 2)  
fig.update_layout(title="Сети по категориям", title_x = 0.5, showlegend=False,)  
fig.update_yaxes(title='%', col=2, row=1)  
fig.update_xaxes(title='Категория', col=2, row=1)  
fig.update_yaxes(title='Количество заведений', col=1, row=1)  
fig.update_xaxes(title='Категория', col=1, row=1)  
fig.show()
```

Сети по категориям



Численно больше сетевых:

- кафе - 779 заведений
- ресторанов - 730 заведений
- кофеен - 720 заведений

Меньше всего сетевых столовых, но их в принципе мало в нашем наборе данных.

Доли сетевых заведений внутри категорий распределены в другом порядке. **Чаще всего сетевыми являются:**

- булочная - 61%
- пиццерия - 52%
- кофейня - 51%

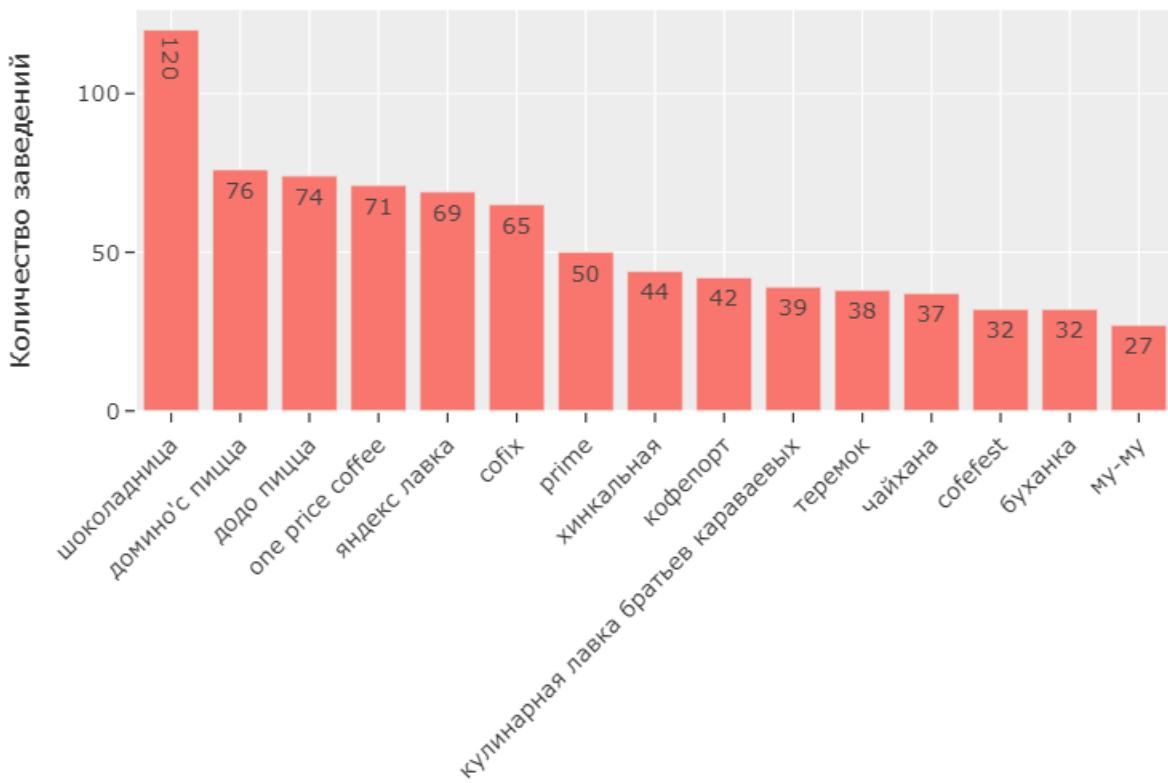
Т.к. нас в большей степени интересуют кофейни, сконцентрируем внимание на том, что **каждая вторая кофейня в Москве - сетевая**

Топ-15 сетевых заведений

```
In [40]: #таблица: название сетей и число их заведений, среди 15-ти самых крупных:  
chain_top_count = data.query('chain==True').groupby('name').agg({'address' : 'count'})\n    .sort_values(by='address', ascending=False).reset_index().head(15)  
chain_top_count = chain_top_count.rename(columns={'address':'count'})  
#сохранение названий в отдельной переменной:  
chain_top_name = chain_top_count['name'].tolist()
```

```
In [41]: fig = px.bar(chain_top_count.sort_values(by='count', ascending=False), x='name', y='count', t  
fig.update_layout(xaxis_tickangle=-45, title='Топ-15 сетей по количеству заведений', title_x  
fig.update_yaxes(title_text = "Количество заведений")  
fig.update_xaxes(title_text = "")  
fig.show()
```

Топ-15 сетей по количеству заведений



Самая крупная сеть - кофейня "Шоколадница", 120 заведений

В тройку входят две пиццерии: "**Домино'с пицца**" и "**Додо пицца**" - 76 и 74 заведения соответственно

Посмотрим к каким категориям относятся остальные сети Топ-15

```
In [42]: # срез исходного датасета, группировка по категории
chain_top = data.query('name in @chain_top_name')
chain_top_category = chain_top.groupby('category')[['name']].count().sort_values().reset_index()
#таблица для подсчета категорий
chain_top_category_total = chain_top.groupby(['name', 'category']).agg({'district':'count'})
chain_top_category_total = chain_top_category_total.groupby('category')[['name']].count().reset_index()

In [43]: fig = make_subplots(rows=1, cols=2, specs=[[{'type':'Bar'}, {'type':'Bar'}]],
                        subplot_titles=['Категории Топ-15', 'Количество заведений по категориям'])

fig.add_trace(go.Bar(y=chain_top_category['category'], x=chain_top_category['name'], showlegend=False,
                     text=chain_top_category['name'], orientation='h'),
              1, 2)
fig.add_trace(go.Bar(y=chain_top_category_total['category'], x=chain_top_category_total['name'],
                     text=chain_top_category_total['name'], showlegend=False, orientation='h'),
              1, 1)

fig.update_layout(xaxis_tickangle=-45, title='Сети Топ-15: категории', title_x = 0.5)
fig.update_xaxes(title='Количество сетей', col=1, row=1)
fig.update_xaxes(title='Количество заведений', col=2, row=1)
fig.show()
```

Сети Топ-15: категории



Заведения одной и той же сети могут относиться к разным категориям (например, заведения сети 'Му-му' относятся аж к 7 разным категориям). Вероятно, одна и та же сеть может практиковать распространение разных форматов своих заведений, либо загружать в открытые источники разную информацию - одну точку обозначить как ресторан, другую как кафе и т.п.

Исходя из этого можем определить, что чаще всего **сети Топ-15 определяют свои заведения как:**

- кафе
- кофейня
- ресторан

Также принимая в расчет все вышесказанное, можем определить **следующий Топ-3 категорий заведений по количеству:**

- кофейня: 337 заведений, 41.2% от всех заведений топ-15
- ресторан: 188 заведений, 22.7%
- пиццерия: 152 заведения, 18.6%

Есть ли что-то общее у этих сетей? Детально рассмотрим их рейтинги, цены, расположение и сравним с остальным сетями

Дополнительный анализ Топ-15 сетей

Рейтинг

```
In [44]: # таблица, срез сетей, не входящих в top-15
chain_not_top = data.query('name not in @chain_top_name')
```

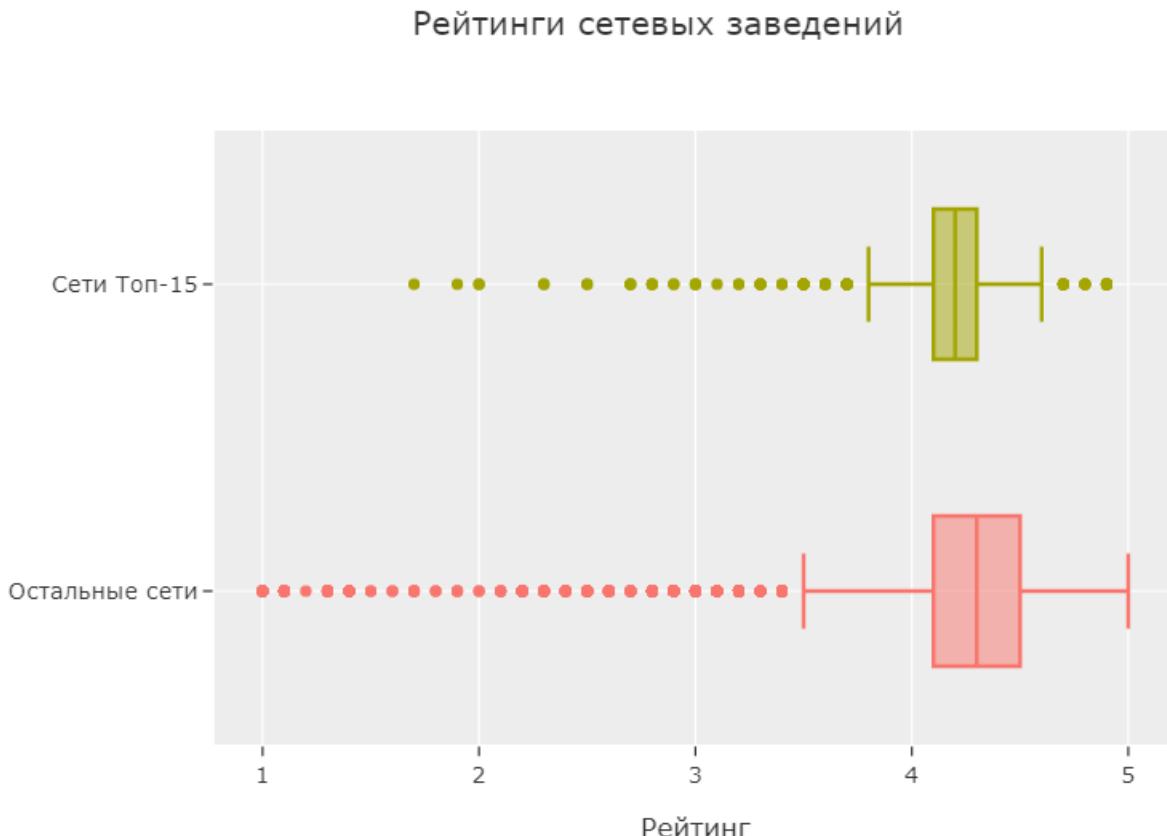
```
fig = go.Figure()
fig.add_trace(go.Box(x=chain_not_top['rating'], name='Остальные сети'))
```

```

fig.add_trace(go.Box(x=chain_top['rating'], name='Сети Топ-15'))

fig.update_layout(title='Рейтинги сетевых заведений',
                  xaxis_title="Рейтинг",
                  showlegend=False,
                  title_x = 0.5)
fig.show()

```



Медианный рейтинг топ-15 сетей чуть ниже, чем у остальных сетей - 4.2 против 4.3.

Из минусов: Наши сети-лидеры **реже получают очень высокие оценки** - граница "ящика с усами" упирается в 4.6, в то время как у остальных сетей в 5. Ни одно заведение Топа-15 не получило такую высокую оценку

Из плюсов: можем отметить, что сети из Топ-15 **намного реже получают крайне низкие оценки**: оценок ниже 3 значительно меньше, чем у других сетей, а минимальная оценка - 1.7

Цены

Посмотрим как отличаются сети Топ-15 от остальных сетей по стоимости среднего чека.

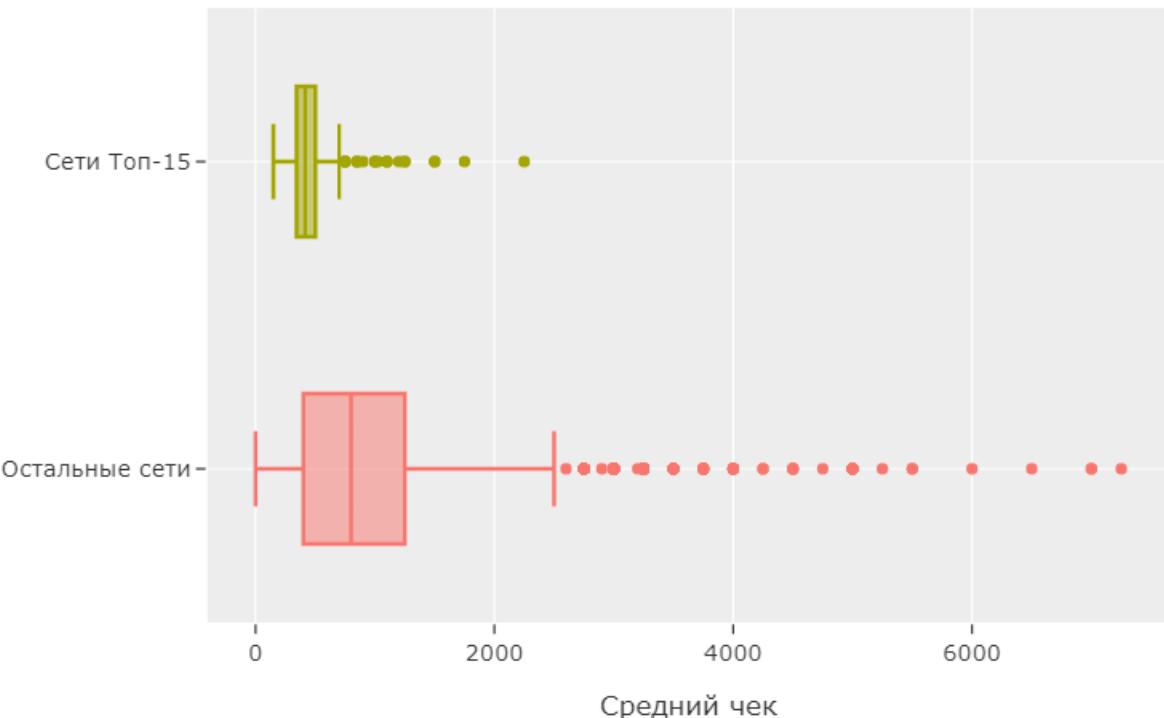
```

In [45]: #здесь срезаем экстремально высокие чеки, обнаруженные на предобработке
fig = go.Figure()
fig.add_trace(go.Box(x=chain_not_top.query('middle_avg_bill < 10000')['middle_avg_bill'].dropna())
fig.add_trace(go.Box(x=chain_top.query('middle_avg_bill < 10000')['middle_avg_bill'].dropna())

fig.update_layout(title='Средние чеки сетевых заведений',
                  xaxis_title="Средний чек",
                  showlegend=False,
                  title_x = 0.5)
fig.show()

```

Средние чеки сетевых заведений



Можем заметить, что средний чек у Топ-15 значительно ниже , чем у остальных сетей.

Большинство значений среднего чека у Топ-15 расположено в диапазоне 150-700Р, в то время как у остальных сетей только медиана расположена на 750.

Очевидно, здесь сети Топ-15 выигрывают у остальных сетей, **распространенность заведений и узнаваемость позволяет удерживать средний чек на достаточно низком уровне**

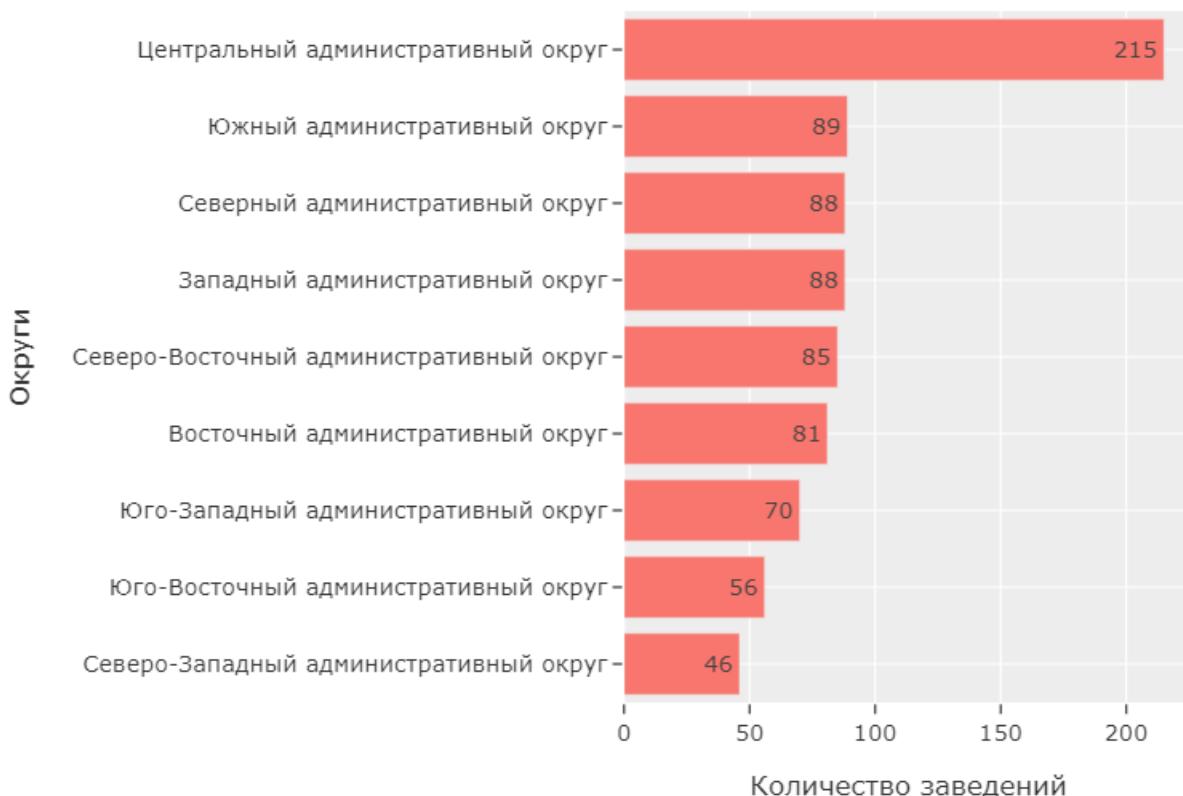
Распределение по округам

```
In [46]: # таблица: сети топ-15 по округам, подсчет числа заведений
district_chains_top = chain_top.groupby('district')['address'].count().sort_values().reset_index()
# таблица: сети топ-15, подсчет занятых округов
chain_top_located = chain_top.groupby('name')['district'].nunique().sort_values().reset_index()
```

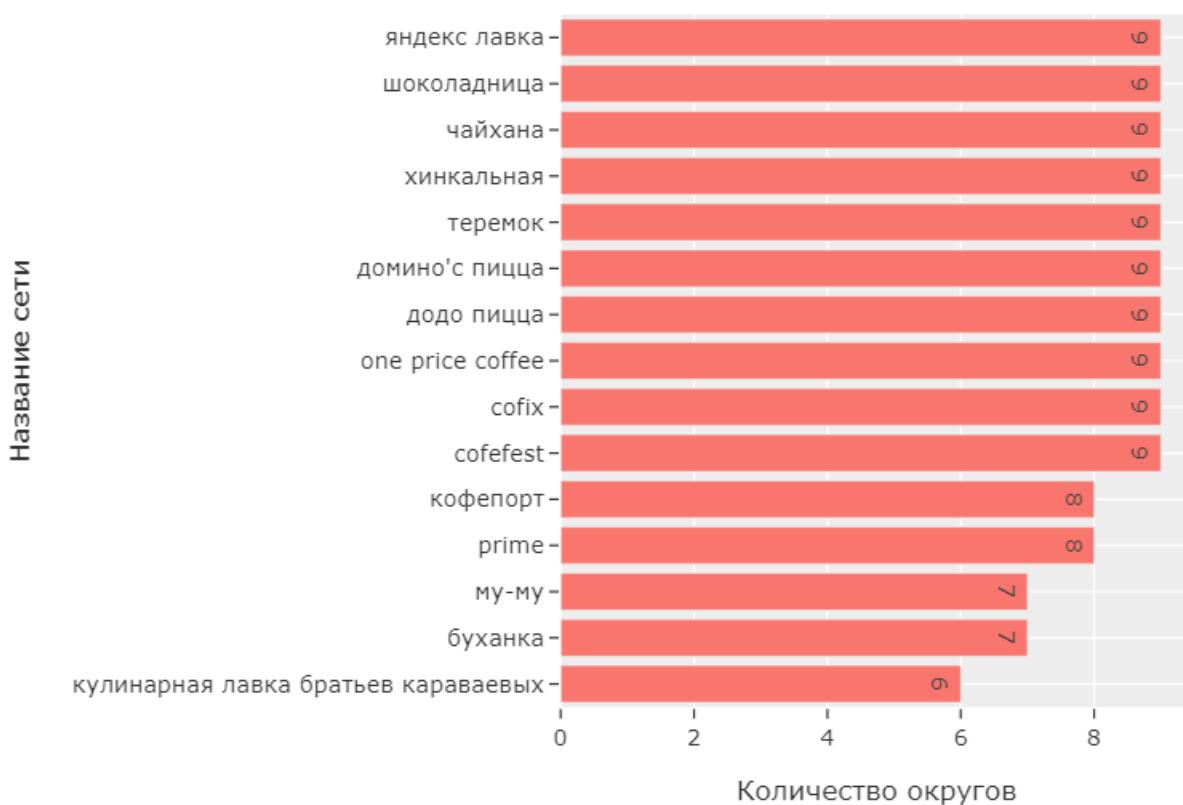
```
In [47]: fig = px.bar(y=district_chains_top['district'], x=district_chains_top['address'], orientation='v',
                  text=district_chains_top['address'])
fig.update_layout(title="Заведения Топ-15 сетей по округам", title_x = 0.5,
                  showlegend=False,
                  xaxis_title = 'Количество заведений',
                  yaxis_title = 'Округи')
fig.show()

fig = px.bar(y=chain_top_located['name'], x=chain_top_located['district'], orientation='h',
              text=chain_top_located['district'])
fig.update_layout(title="Количество занятых округов заведениями Топ-15", title_x = 0.5,showle
                  xaxis_title = 'Количество округов',
                  yaxis_title = 'Название сети')
fig.show()
```

Заведения Топ-15 сетей по округам



Количество занятых округов заведениями Топ-15



Большинство заведений из Топ-15 расположены в ЦАО, что неудивительно - в округе в принципе больше всего заведений. Топовые сети расположены по остальным московским округам в схожих долях, за исключением ЮЗАО, ЮВАО и СЗАО - там заведений меньше.

Также отметим, что **10 из 15 сетей имеют заведения в каждом московском округе**. Самый низкий показатель у "Кулинарной лавки братьев Караваевых" - но и у этой сети заняты 6 округов

из 9.

Подводя итог, выделим **две основные объединяющие черты ТОП-15 сетей**:

- Низкий средний чек относительно других сетей
- Масштаб распространения: лидирующие сети не концентрируются на конкретных округах, а распространяют свои заведения по всей Москве

Распределение по административным округам

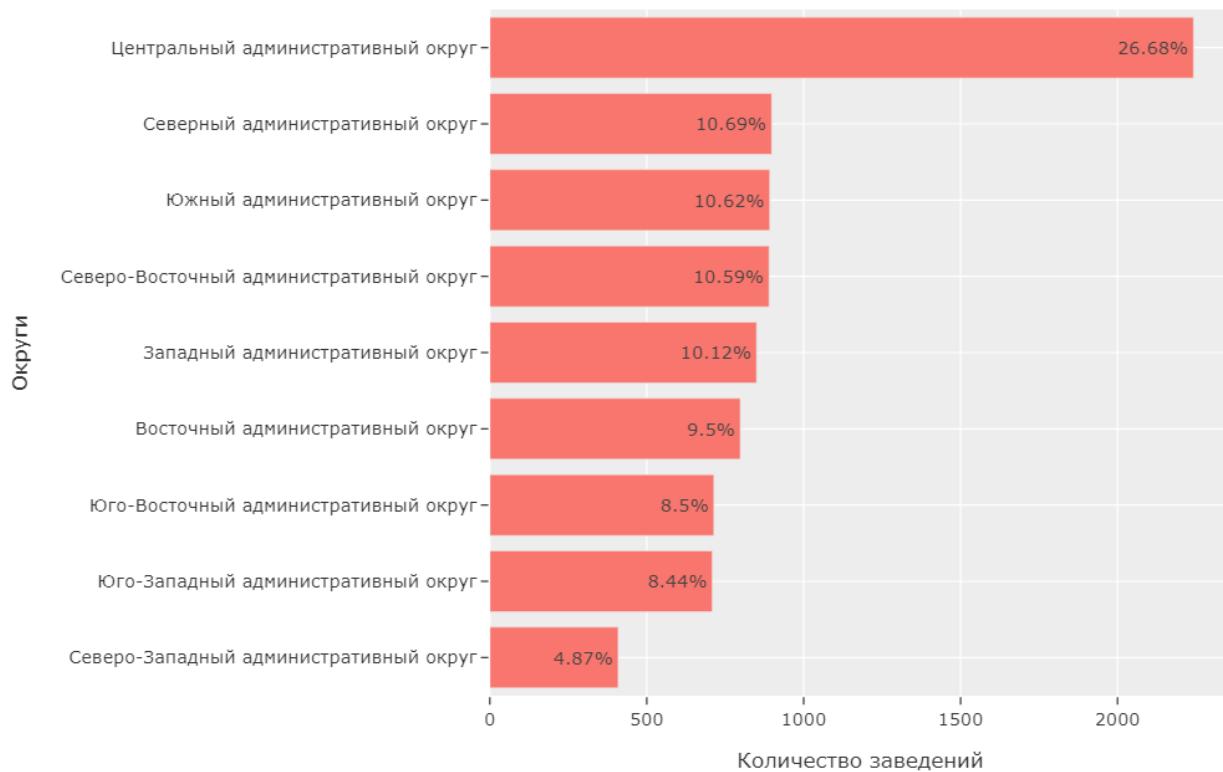
In [48]:

```
#таблица: группировка по району и категории, подсчет заведений:  
district_rest = data.groupby(['district', 'category'])\n    .agg({'name': 'count'}).sort_values('name', ascending=False).reset_index()  
district_rest_group = district_rest.groupby('district')['name'].sum().reset_index()  
district_rest_group['ratio'] = round((district_rest_group['name'] / district_rest_group['name'] * 100).round(2))  
district_rest_group['ratio'] = district_rest_group['ratio'].astype('string')+'%'
```

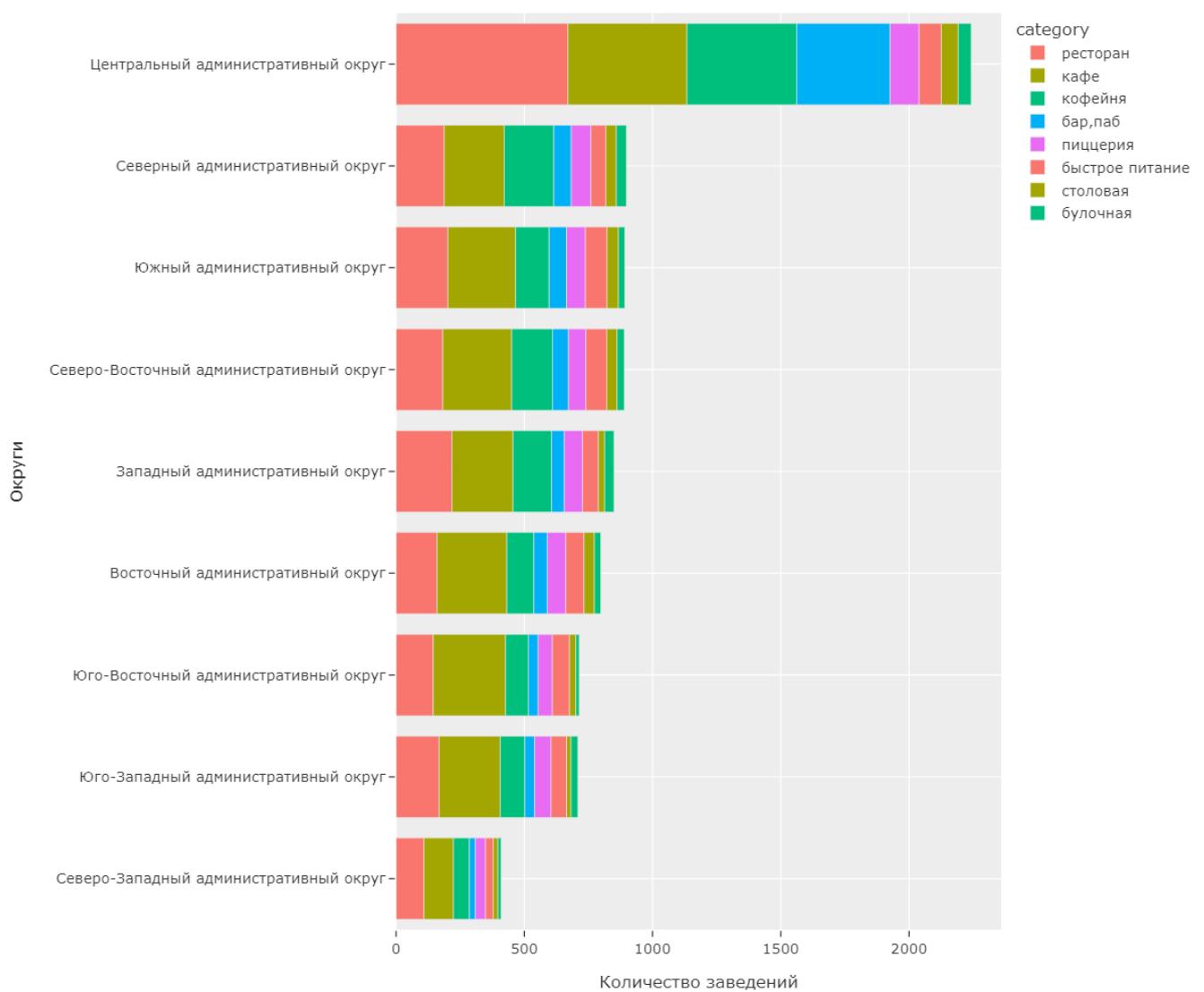
In [49]:

```
fig = px.bar(district_rest_group,\n              x='name',\n              y='district',\n              text=district_rest_group['ratio'])\nfig.update_layout(title = 'Количество заведений по округам и их доли от общего числа заведений',\n                  xaxis_title = 'Количество заведений',\n                  yaxis_title = 'Округи',\n                  yaxis={'categoryorder': 'total ascending'},\n                  height=600, width=900)\nfig.show()\n\nfig = px.bar(district_rest,\n              x='name',\n              y='district',\n              color='category')\nfig.update_layout(title = 'Количество заведений в округах по категориям',\n                  xaxis_title = 'Количество заведений',\n                  yaxis_title = 'Округи',\n                  yaxis={'categoryorder': 'total ascending'},\n                  height=900, width=1000)\nfig.show()
```

Количество заведений по округам и их доля от общего числа заведений



Количество заведений в округах по категориям



- Больше всего заведений в ЦАО

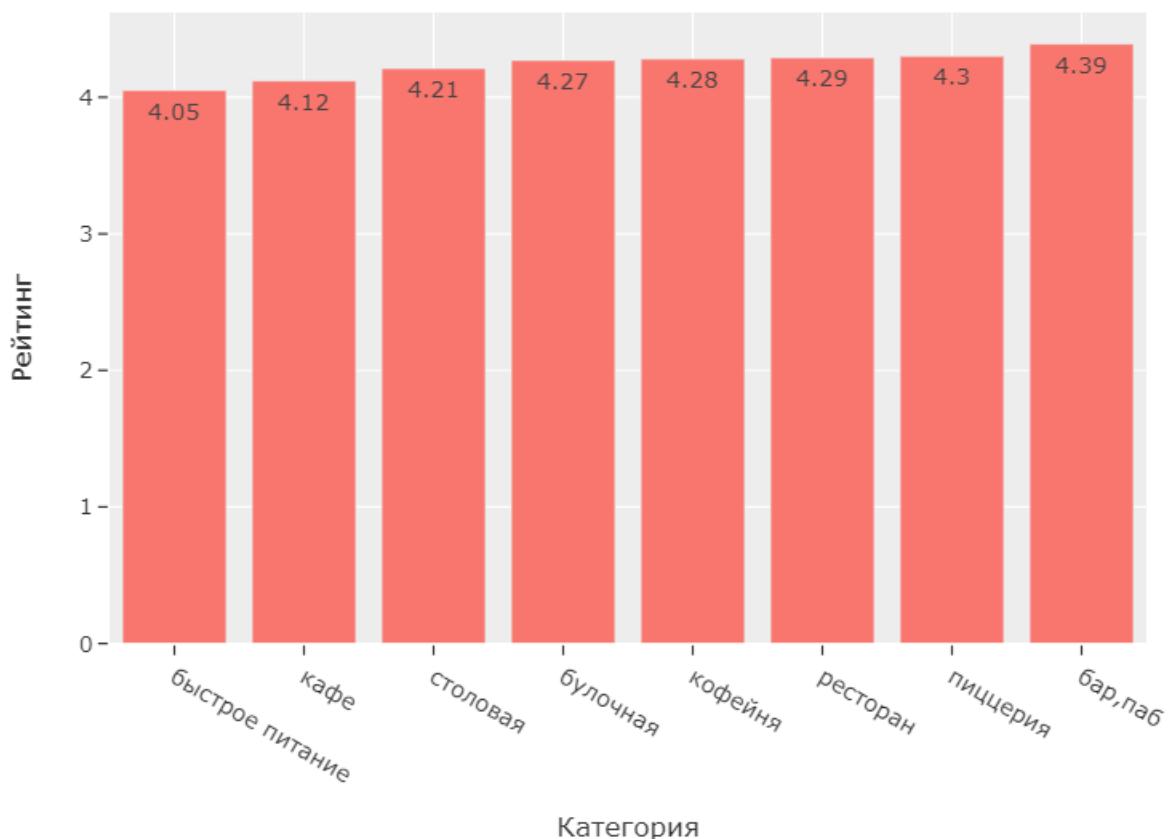
- единственный округ, где самое популярное заведение - ресторан
- в схожих внушительных долях распределены кафе, кофейни и бары
- Во всех округах численное преимущество имеют **кафе и рестораны**
 - за пределами ЦАО рестораны уступают более бюджетному типу - кафе
- В каждом округе **заметная доля кофеен** (взде на третьем месте по распространенности)
- Все малочисленные категории распределены по округам +- в одинаковых долях

Средний рейтинг по категориям

```
In [50]: #таблица: медианные и средние рейтинги по категориям:
category_rating = data.pivot_table(index='category', values='rating', aggfunc={'median': 'median', 'mean': 'mean'})
category_rating['mean']=round(category_rating['mean'],2)
```

```
In [51]: fig = px.bar(x=category_rating['category'],
                  y=category_rating['mean'],
                  text=category_rating['mean'])
fig.update_layout(title = 'Средние рейтинги заведений по категориям',
                  xaxis_title = 'Категория',
                  yaxis_title = 'Рейтинг',
                  yaxis={'categoryorder': 'total ascending'})
fig.show()
```

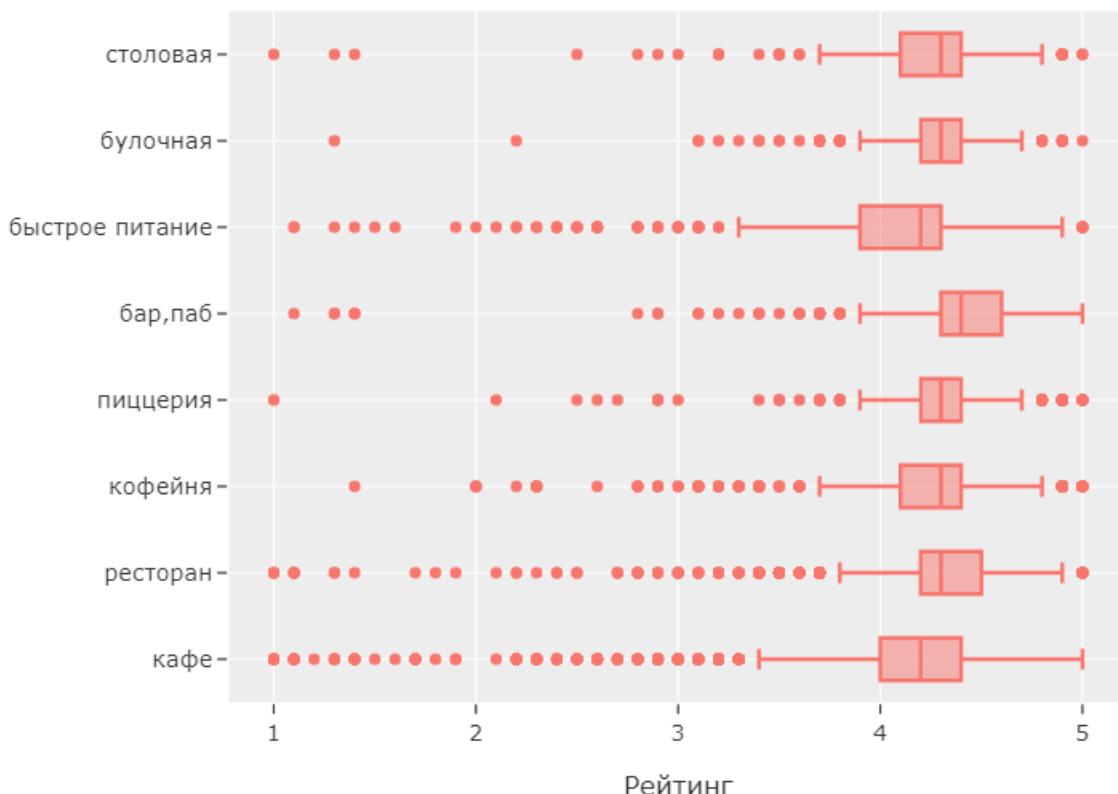
Средние рейтинги заведений по категориям



Самый низкий средний рейтинг у категории "быстрое питание" - 4.05 **самый высокий - у баров(пабов), 4.39**

```
In [52]: fig = px.box(data, y= "category", x="rating")
fig.update_layout(title='Рейтинги по категориям заведений: разброс данных',
                  yaxis_title="",
                  xaxis_title="Рейтинг",
                  showlegend=False,
                  title_x = 0.5)
fig.show()
```

Рейтинги по категориям заведений: разброс данных



Резюмируем график:

- **Отличные оценки (5)** чаще остальных категорий получают бары и кафе
- **Низкие оценки (<3)** чаще всего получают заведения быстрого питания, рестораны и кафе. Эти же три категории чаще остальных имеют экстремально низкие оценки <2
- **Самый высокий срединный рейтинг** у баров, 4.39, у этой же категории самая
- **По самым распространенным категориям:**
 - кафе: большинство значений располагаются в диапазоне 3.4-5, медиана 4.2. Чаще остальных категорий имеют очень низкие средние рейтинги
 - ресторан: большинство значений в диапазоне 3.8-4.9, медиана 4.3, относительно кафе имеют меньше заведений с низким рейтингом
 - кофейня: большинство значений в диапазоне 3.7-4.8, заведения этой категории реже, чем у ресторанов и кафе, имеют максимально высокие оценки, при этом и низкие оценки (<3) встречаются тоже значительно реже

Средний рейтинг заведений по округам: фоновая картограмма

```
In [53]: #создание таблицы: средние рейтинги по округам
district_rating = data.groupby('district', as_index=False)[['rating']].mean()
```

```
In [54]: # загружаем JSON-файл с границами округов Москвы
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 = Map(location=[moscow_lat, moscow_lng], zoom_start=10)

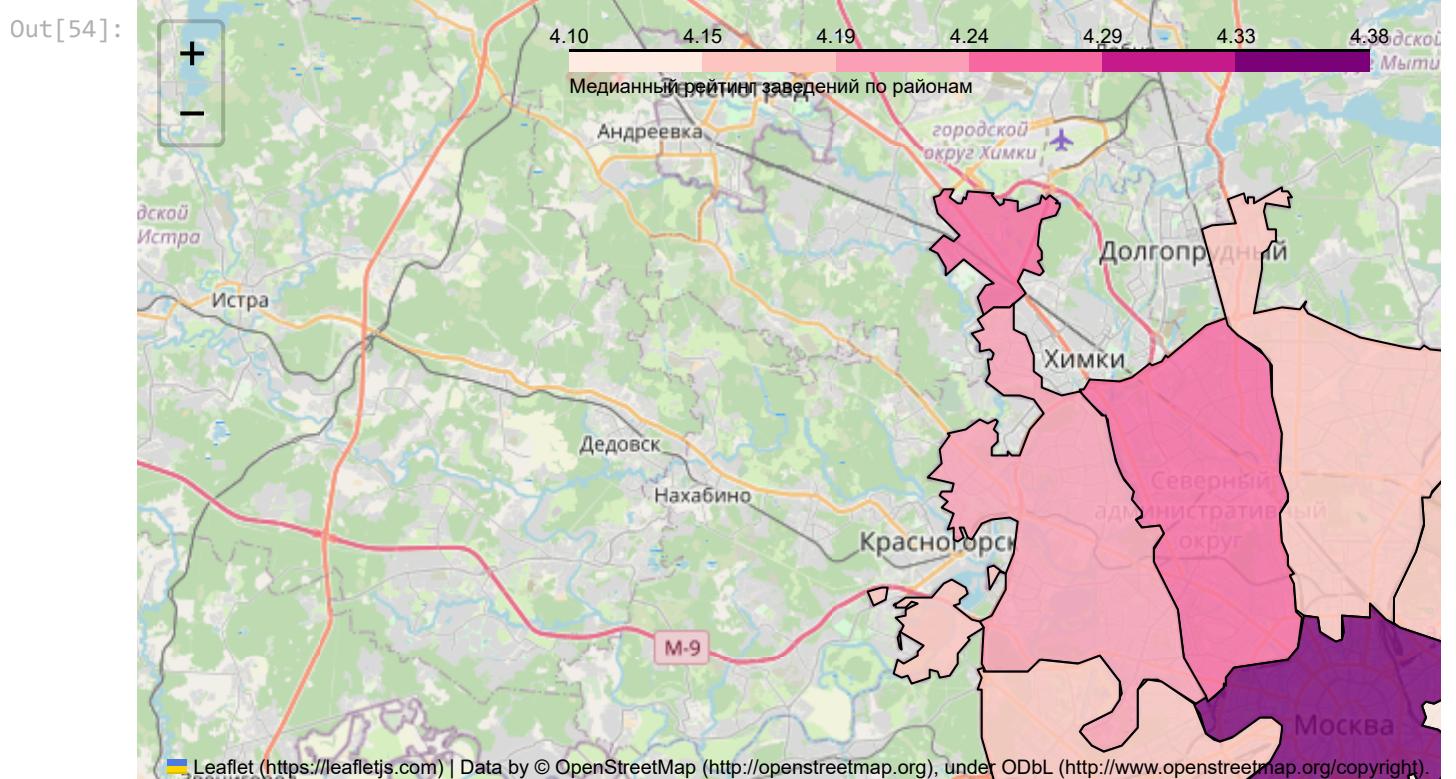
# создаем хороплет с помощью конструктора Choropleth и добавляем его на карту
Choropleth(
    geo_data=state_geo,
```

```

    data=district_rating,
    columns=['district', 'rating'],
    key_on='feature.name',
    fill_color='RdPu',
    fill_opacity=0.8,
    legend_name='Медианный рейтинг заведений по районам',
).add_to(m)

# выводим карту
m

```



- **ЦАО:** самый высокий средний рейтинг у заведений, 4.38
- **САО** - на втором месте, 4.23
- **СЗАО** - замыкает топ-3, 4.2
- все остальные, за исключением ЮВАО, имеют близкие значения среднего рейтинга, 4.15-4.18
- **ЮВАО:** самый низкий рейтинг, 4.1

Все заведения на карте

```

In [55]: m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
marker_cluster = MarkerCluster().add_to(m)

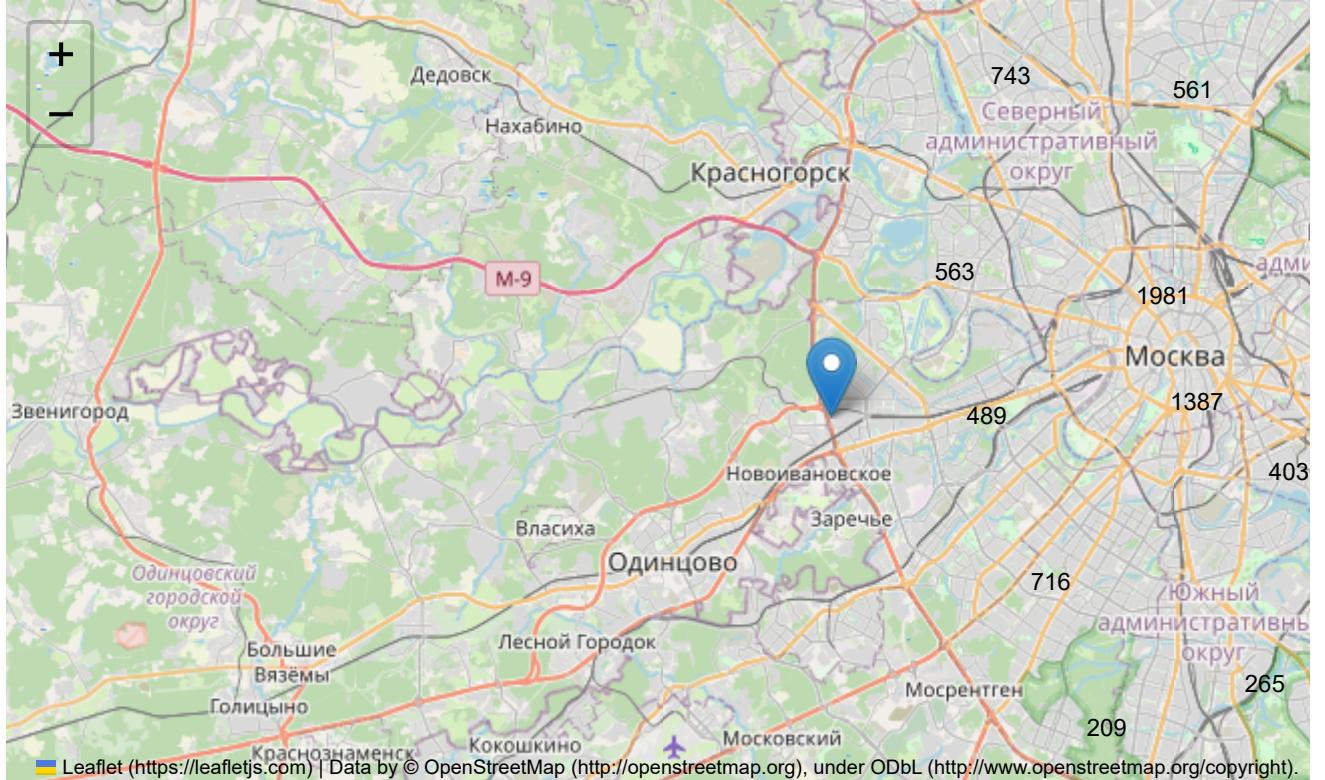
# функция, принимающая строку датафрейма, создает маркер в текущей точке и добавляет его в кл
def create_cluster(row):
    Marker(
        [row['lat'], row['lng']],
        popup=f'{row['name']} {row['rating']}',
    ).add_to(marker_cluster)

# к каждой строке датафрейма применяем функцию
data.apply(create_cluster, axis=1)

# выводим карту
m

```

Out[55]:



Топ-15 улиц по количеству заведений

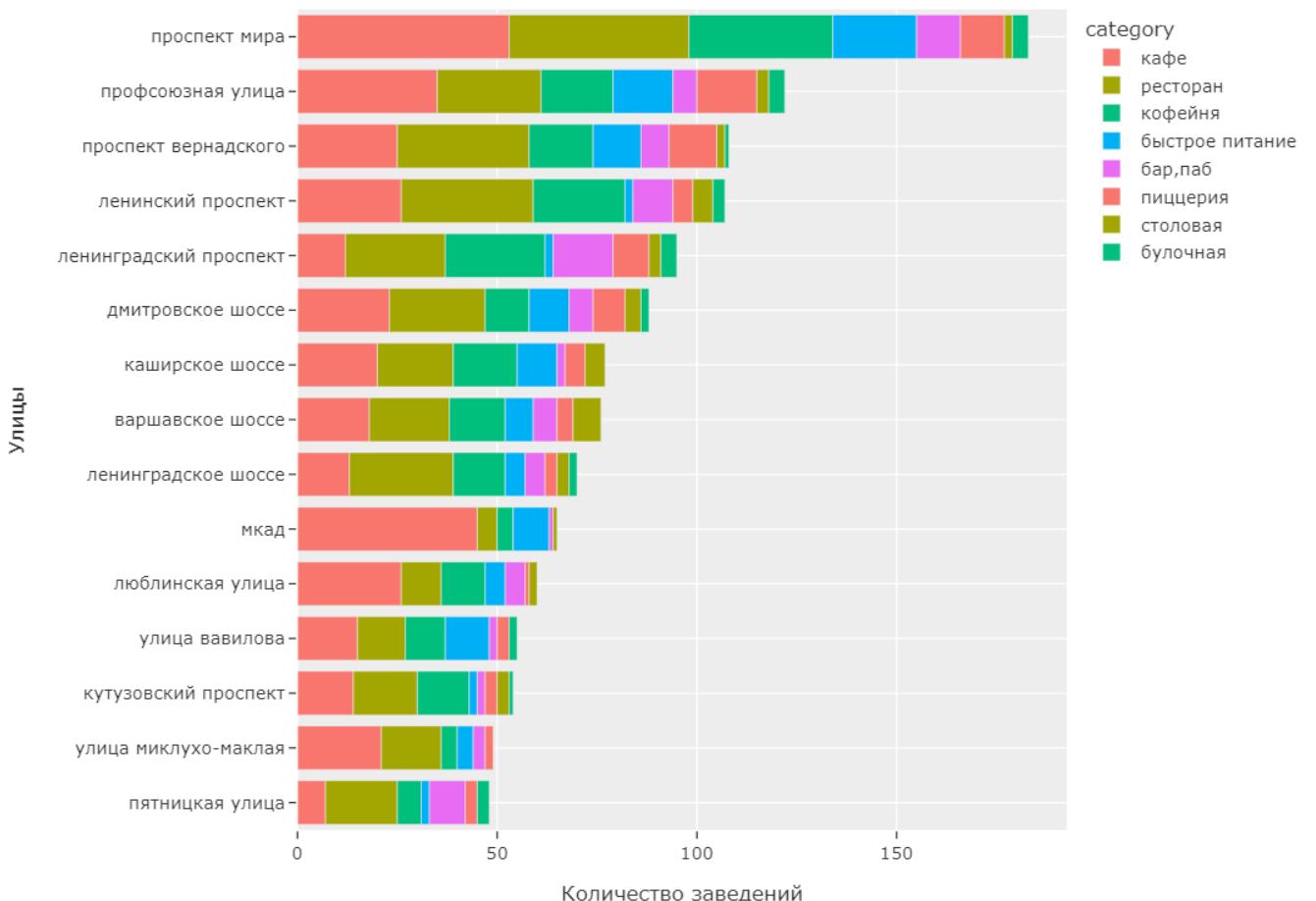
```
In [56]: # таблица: группировка по улице, подсчет заведений, срез первых 15 по числу заведений
top_streets = data.groupby('street').agg({'name': 'count'}).sort_values(by='name', ascending=False)
#сохранение списка топ-15 улиц в переменной
top_streets_name = top_streets['street'].tolist()

#срез исходного датасета по топовым улицам, группировка среза по категориям
top_street = data.query('street in @top_streets_name')
top_streets_cat = top_street.groupby(['street', 'category'])['name'].count().reset_index().sort_values(by='name', ascending=False)
```

```
In [57]: fig = px.bar(top_streets_cat,
                  x='name',
                  y='street',
                  color='category')

fig.update_layout(title='Топ-15 улиц: количество заведений',
                  xaxis_title='Количество заведений',
                  yaxis_title='Улицы',
                  yaxis={'categoryorder':'total ascending'},
                  height=700, width=900)
fig.show()
```

Топ-15 улиц: количество заведений



Неудивительно, что в **Топ-15 вошли одни из самых длинных улиц, проспектов и шоссе Москвы**, логично, что чем длиннее улица, тем больше заведений может там расположиться.

Можем отметить, что **численное преимущество для каждой улицы остается за кафе или ресторанами**, но в целом распределение заведений по категориям для улиц нельзя назвать схожим: например, на Ленинградском шоссе чаще остальных категорий располагаются рестораны, а на МКАД - кафе

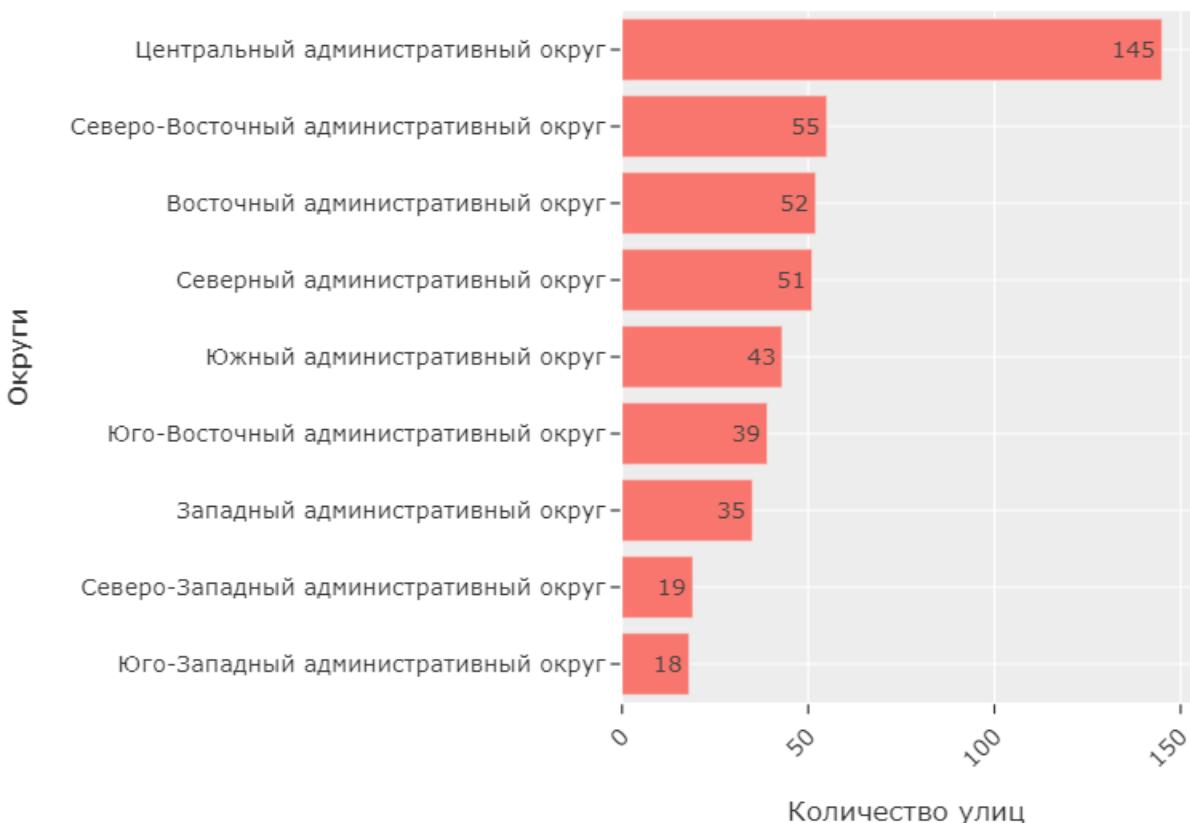
Улицы с только одним объектом общепита

```
In [58]: #таблица группировка по улице, подсчет заведений, срез 'только одно заведение'
unpop_street = data.groupby('street').agg({'district' : pd.Series.mode, 'name': 'count'}).sort_index()
unpop_street = unpop_street.query('name == 1')
#сохранение улиц с одним объектом в переменную, срез датасета по этим улицам
unpop_street_name = unpop_street['street'].tolist()
unpop_street_data = data.query('street in @unpop_street_name')
print('Количество улиц с только одним заведением:', len(unpop_street_data))
```

Количество улиц с только одним заведением: 457

```
In [59]: unpop_street_data_district = unpop_street_data.groupby('district')\ .agg({'name':'count','rating':'mean', 'middle_avg_bill':'mean'}).reset_index()
unpop_street_data_district = unpop_street_data_district.rename(columns={'name':'total_rest'})
unpop_street_data_district['rating_total']=data.groupby('district')['rating'].mean().reset_index()
unpop_street_data_district['middle_avg_bill_total'] =data.groupby('district')['middle_avg_bill'].mean().reset_index()['middle_avg_bill']
unpop_street_data_district = unpop_street_data_district.round({'rating':2, 'rating_total':2, 'middle_avg_bill':2})
fig = px.bar(unpop_street_data_district.sort_values(by='total_rest'), y='district', x='total_rest', title='Улицы с только одним заведением: распределение по округам', title_text = "Округи")
fig.update_yaxes(title_text = "Количество улиц")
```

Улицы с только одним заведением: распределение по округам



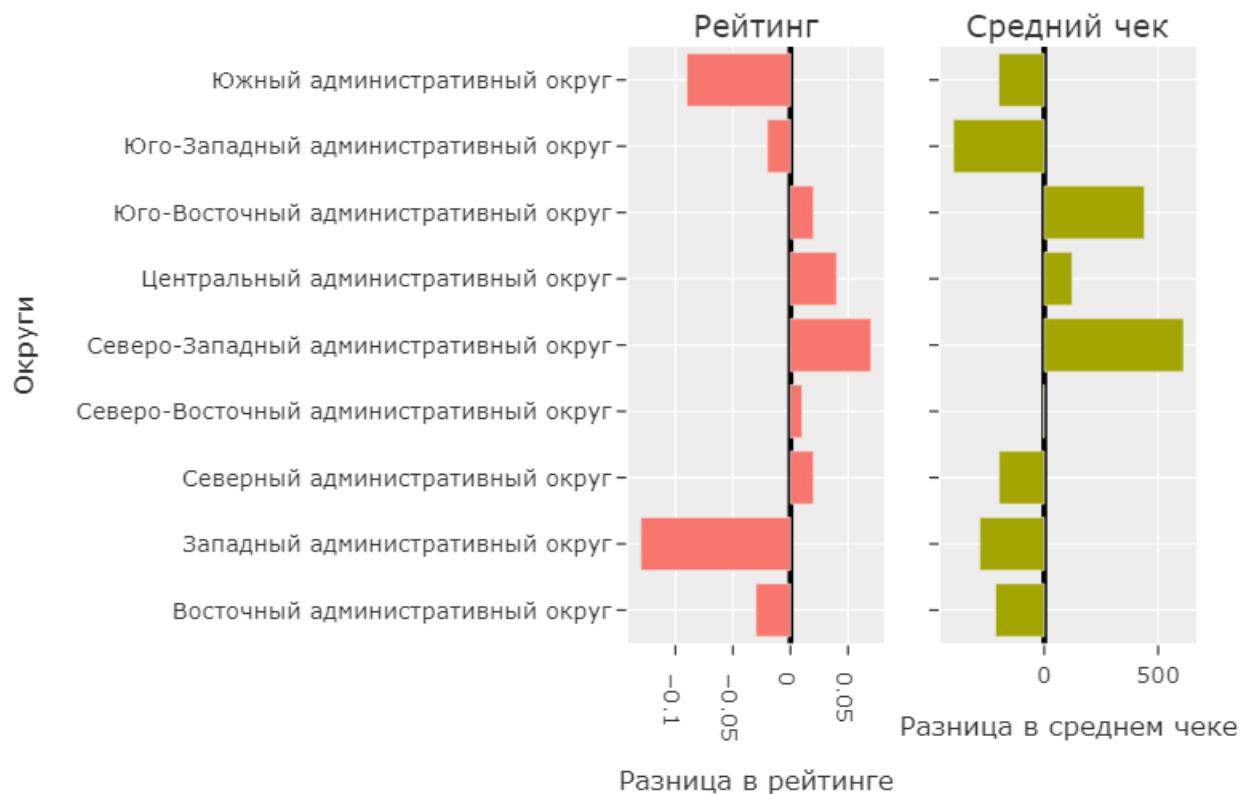
Из 457 таких улиц 145 находится в ЦАО. Немного отстранимся и отметим, что в центре Москвы часто встречаются очень короткие улицы - для столицы не редкость разного рода "тупики" и "переулки" на 4 дома) Поэтому одно заведение на улице вполне может быть обосновано размером этой улицы.

Проанализируем такие улицы детальнее.

```
In [60]: fig = make_subplots(rows=1, cols=2, shared_yaxes=True, subplot_titles=['Рейтинг', 'Средний чек'])
trace0 = go.Bar(y = unpop_street_data_district['district'],
                 x = unpop_street_data_district['rating'] - unpop_street_data_district['rating'].mean())
trace1 = go.Bar(y = unpop_street_data_district['district'],
                 x = unpop_street_data_district['middle_avg_bill'] - unpop_street_data_district['middle_avg_bill'].mean())
fig.update_xaxes(zeroline=True, zerolinewidth=3, zerolinecolor='black')
fig.update_layout(showlegend=False, title='Единственные на улице заведения в сравнении с остальными')

fig.append_trace(trace0, 1, 1)
fig.append_trace(trace1, 1, 2)
fig.update_xaxes(title='Разница в рейтинге', col=1, row=1)
fig.update_yaxes(title='Округи', col=1, row=1)
fig.update_xaxes(title='Разница в среднем чеке', col=2, row=1)
fig.show()
```

Единственные на улице заведения в сравнении с остальными заведениями округа



Отметим, что:

- в ЮАО, ЮЗАО, ЗАО и ВАО если заведение на улице одно, то, как правило, оно имеет **рейтинг и средний чек ниже, чем остальные заведения в округе**
- в ЮВАО, ЦАО, СЗАО - наоборот, **чаще всего это заведение с более высокими рейтингом и средним чеком**
- САО выделяется: **рейтинг выше, но ниже средний чек.** В ЮВАО разница между двумя этими показателями незначительна.

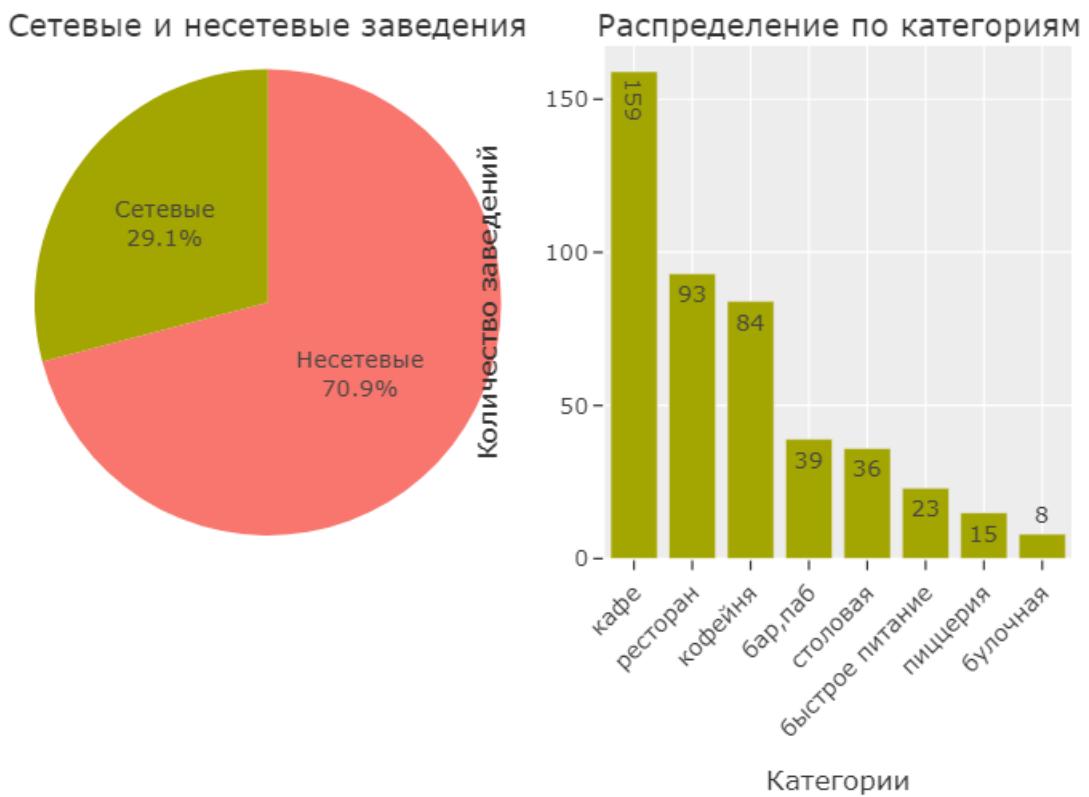
```
In [61]: unpop_chain = unpop_street_data.groupby('chain')['name'].count().reset_index()
unpop_chain['chain'][['Несетевые', 'Сетевые']]
unpop_category = unpop_street_data.groupby('category')['name'].count().reset_index().sort_values(
    by='name', ascending=False)

fig = make_subplots(rows=1, cols=2, specs=[[{'type': 'domain'}, {'type': 'Bar'}]],
                     subplot_titles=['Сетевые и несетевые заведения', 'Распределение по категориям'])

fig.add_trace(go.Pie(labels=unpop_chain['chain'], values=unpop_chain['name']),
              1, 1)
fig.update_traces(textinfo='label+percent')
fig.add_trace(go.Bar(x=unpop_category['category'], y=unpop_category['name'],
                     text=unpop_category['name'], ),
              1, 2)

fig.update_layout(xaxis_tickangle=-45, title='Единственные на улице заведения', title_x = 0.5)
fig.update_yaxes(title='Количество заведений', col=2, row=1)
fig.update_xaxes(title='Категории', col=2, row=1)
fig.show()
```

Единственные на улице заведения



Если на улице только одно заведение, то **чаще всего оно:**

- несетевое (71%)
- относится к кафе, ресторану или бару (35%, 20%, 18%)

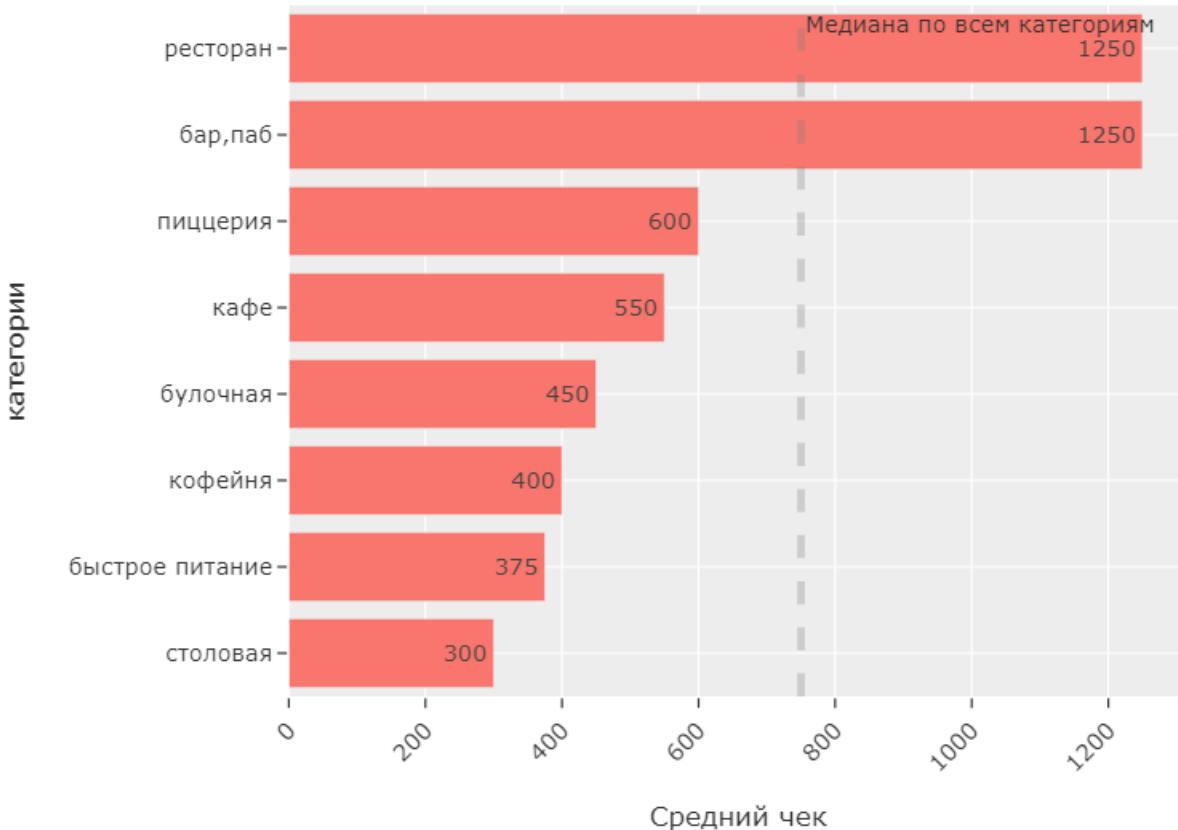
Средние чеки

Средние чеки по категориям

```
In [62]: category_median_bill = data.query('middle_avg_bill < 10000')\
    .groupby('category')['middle_avg_bill'].median().sort_values().reset_index()
```

```
In [63]: fig = px.bar(category_median_bill, y='category', x='middle_avg_bill', text='middle_avg_bill')
fig.update_layout(xaxis_tickangle=-45, title='Медианный средний чек по категориям', title_x =
fig.update_yaxes(title_text = "категории")
fig.update_xaxes(title_text = "Средний чек")
fig.add_vline(x=data.query('middle_avg_bill < 10000')['middle_avg_bill'].median(), line_width
                annotation_text="Медиана по всем категориям")
fig.show()
```

Медианный средний чек по категориям



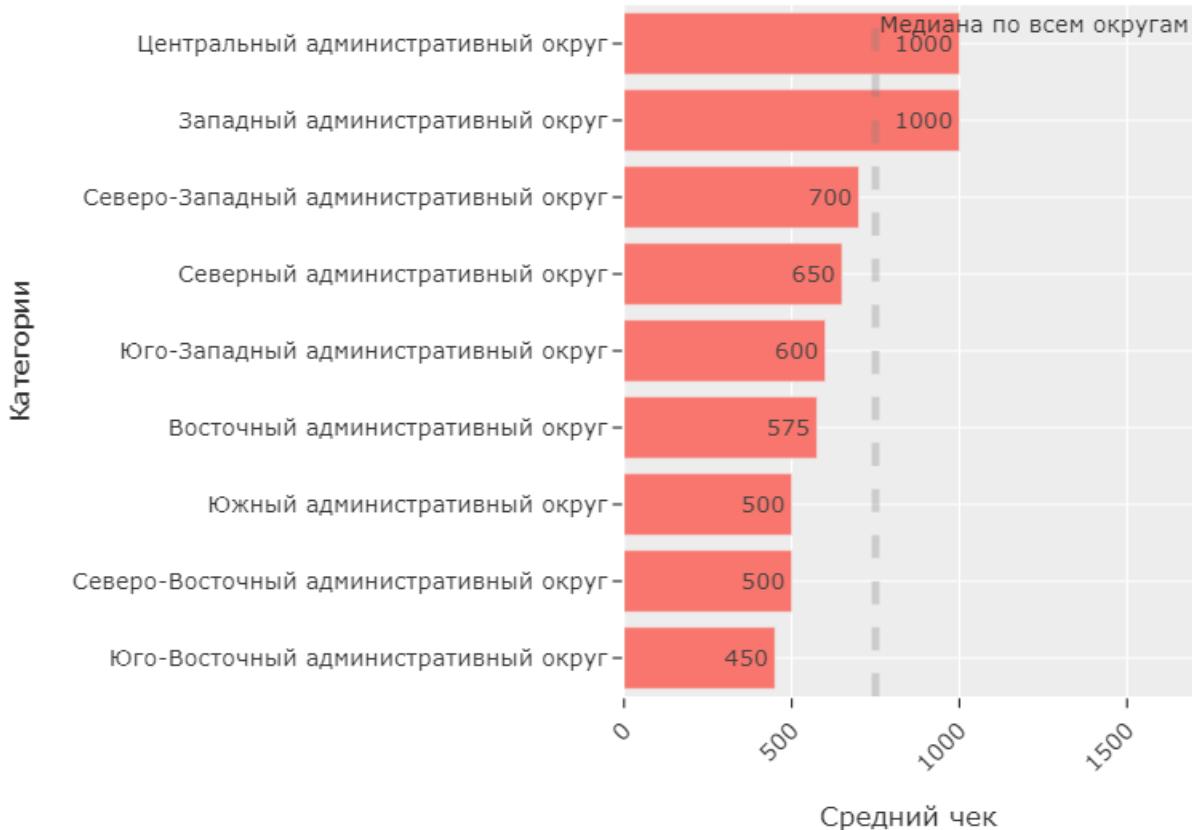
Рестораны и бары - дорогие заведения, медианные значения у обоих категорий 1250. Самые низкие средние чеки у столовых, быстрого питания и кофеен

Средние чеки по округам

```
In [64]: #таблица: медианные средние чеки по округам  
district_median_bill = data.groupby('district')['middle_avg_bill'].median().reset_index()
```

```
In [65]: fig = px.bar(district_median_bill.sort_values(by='middle_avg_bill'), y='district', x='middle_avg_bill'  
fig.update_layout(xaxis_tickangle=-45, title='Медианный средний чек по округам', title_x = 0.  
fig.update_yaxes(title_text = "Категории")  
fig.update_xaxes(title_text = "Средний чек")  
fig.add_vline(x=data['middle_avg_bill'].median(), line_width=4, line_dash="dash", line_color=  
annotation_text="Медиана по всем округам")  
fig.show()
```

Медианный средний чек по округам



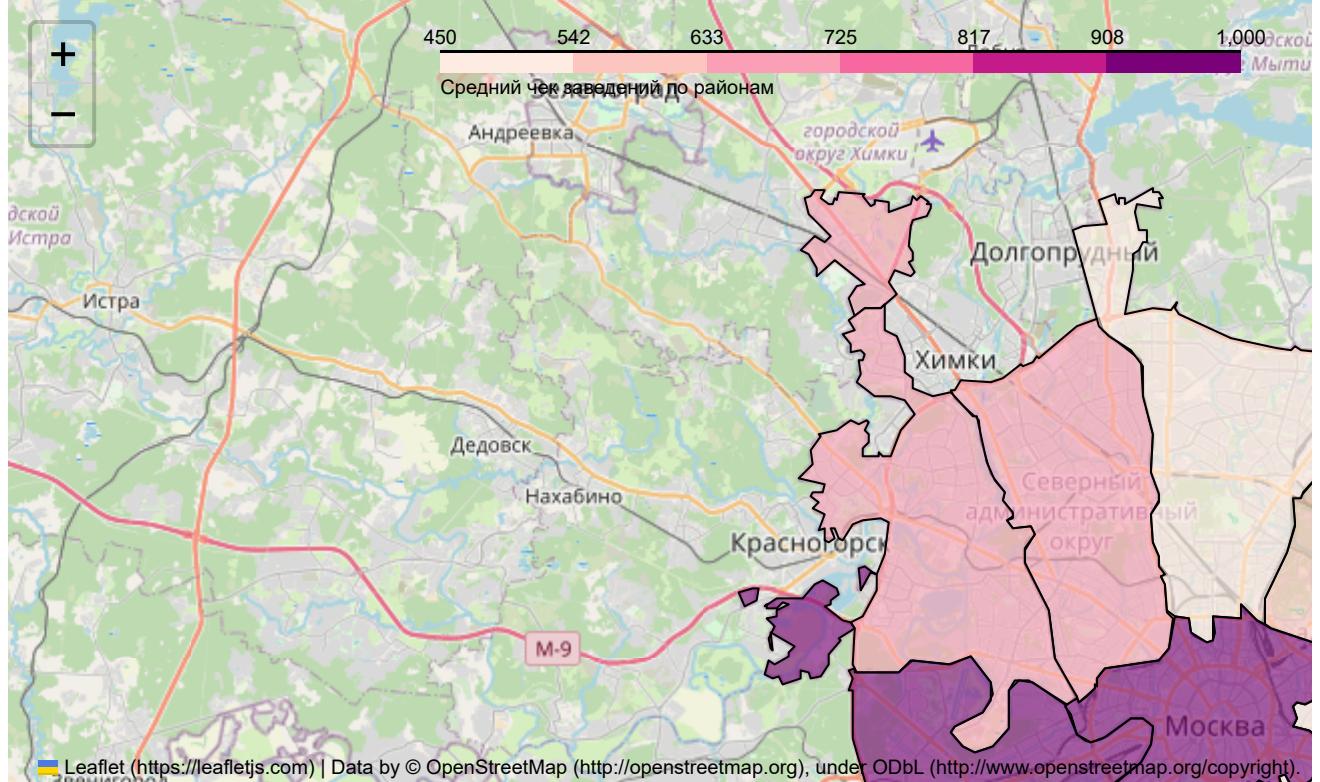
Самые "дорогие" округи - ЦАО и ЗАО, медианные средние чеки выше, чем общая медиана по Москве. Построим карту с распределением рейтингов по округам

```
In [66]: m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)

Choropleth(
    geo_data=state_geo,
    data = district_median_bill,
    columns = ['district', 'middle_avg_bill'],
    key_on = 'feature.name',
    legend_name = 'Средний чек заведений по районам',
    fill_color='RdPu'
).add_to(m)

m
```

Out[66]:



В ЮАО, СВАО, ЮВАО наоборот, - самый низкий средний чек

Дополнительный анализ

Исследуем круглосуточные заведения и заведения с низким рейтингом.

Круглосуточные заведения

In [67]:

```
#срез таблицы по круглосуточным заведениям, группировка по категории
data_24_7 = data.query('is_24_7 == True')
data_24_7_cat = data_24_7.groupby('category')[['name']].count().sort_values(ascending=False).reset_index()
data_24_7_count = data.groupby('is_24_7')[['name']].count().reset_index()
data_24_7_count['is_24_7']=[ 'Некруглосуточные', 'Круглосуточные']
```

In [68]:

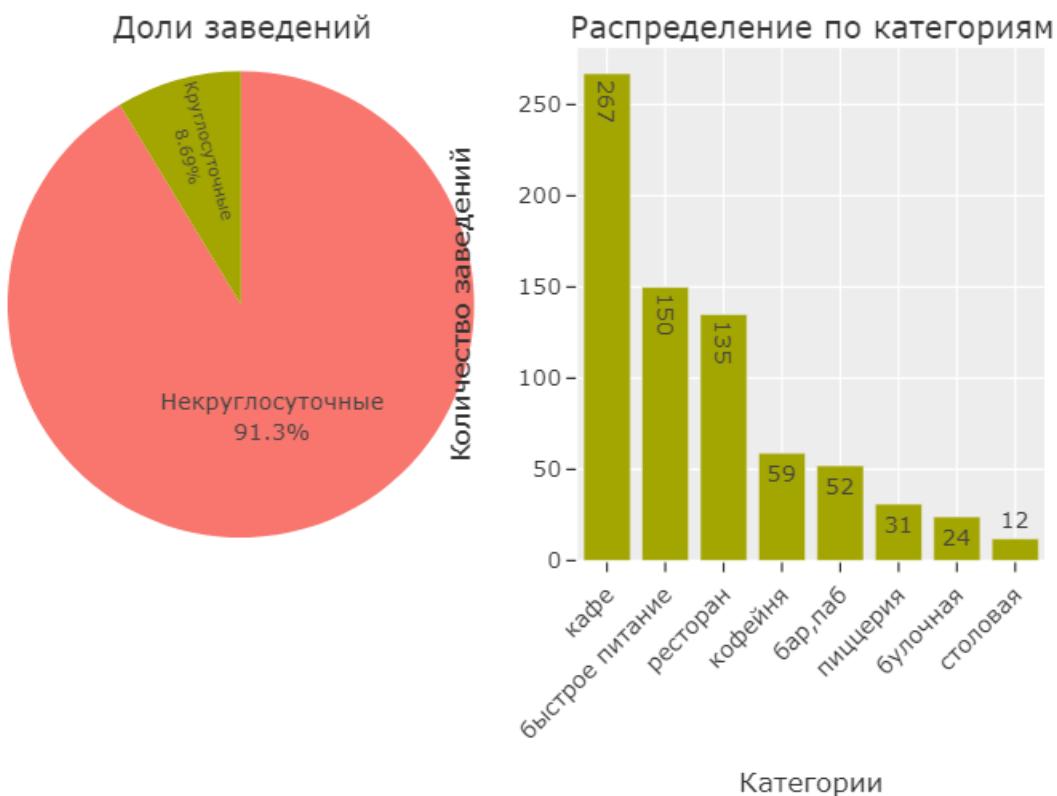
```
fig = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'}, {'type':'Bar'}]],
                     subplot_titles=['Доли заведений', 'Распределение по категориям'])

fig.add_trace(go.Pie(labels=data_24_7_count['is_24_7'], values=data_24_7_count['name'], showlegend=False,
                     1, 1))
fig.update_traces(textinfo='label+percent', textposition='inside')

fig.add_trace(go.Bar(x=data_24_7_cat['category'], y=data_24_7_cat['name'], showlegend=False,
                     1, 2))

fig.update_yaxes(title='Количество заведений', col=2, row=1)
fig.update_xaxes(title='Категории', col=2, row=1)
fig.update_layout(xaxis_tickangle=-45, title='Круглосуточные заведения', title_x = 0.5)
fig.show()
```

Круглосуточные заведения



Круглосуточных заведений немного - всего 730 (8.7%). Чаще всего круглосуточными являются кафе, быстрое питание и рестораны.

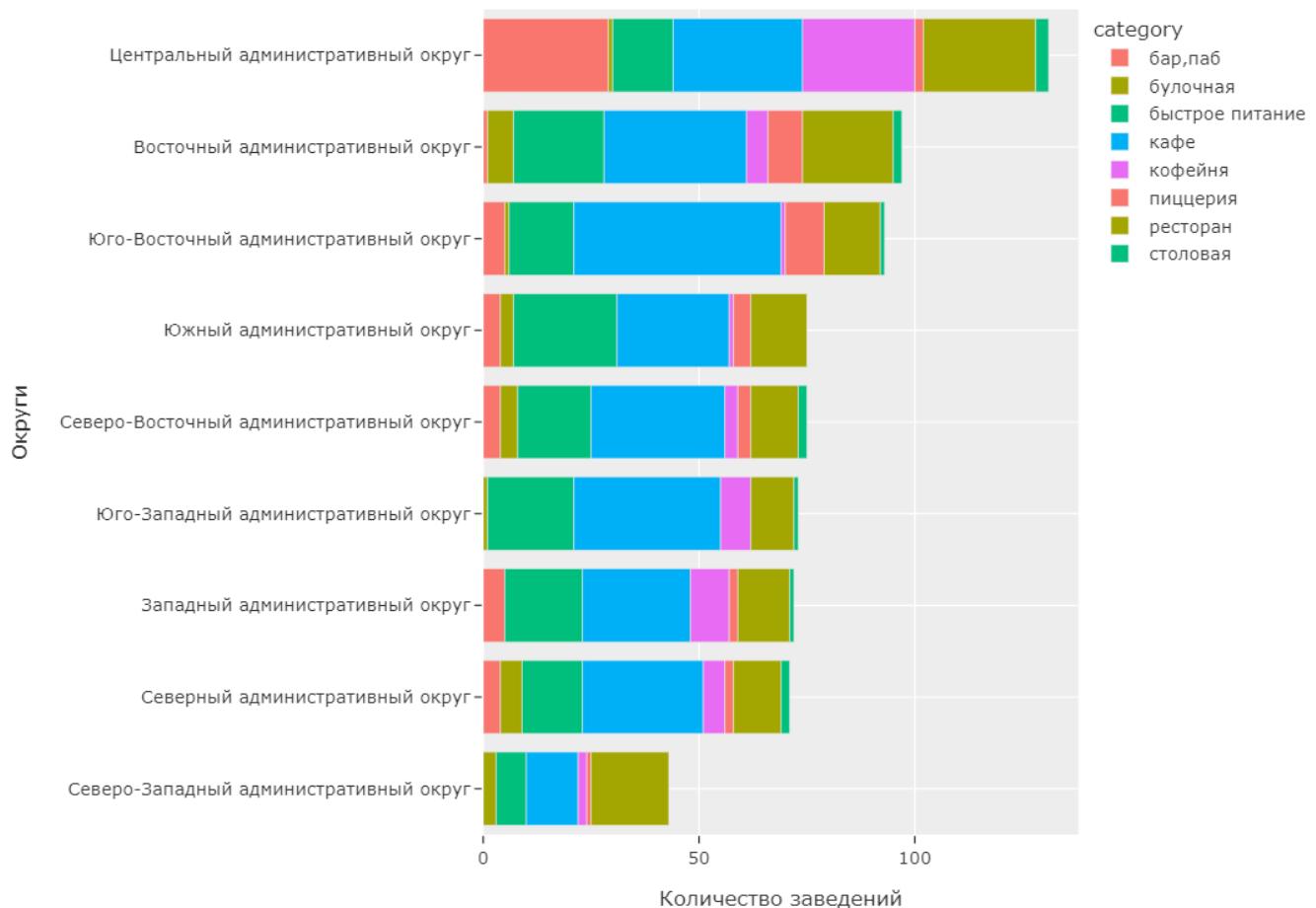
```
In [69]: #таблица: группировка по району и категории
district_24_7 = data_24_7.groupby(['district', 'category'])['name'].count().reset_index()
```

```
In [70]: fig = px.bar(district_24_7,
                  y='district',
                  x='name',
                  color='category')

fig.update_layout(title='Категории круглосуточных заведений по округам',
                  xaxis_title='Количество заведений',
                  yaxis_title='Округи',
                  yaxis={'categoryorder':'total ascending'},
                  height=700, width=900)

fig.show()
```

Категории круглосуточных заведений по округам



И снова ЦАО выделяется среди других округов: только в центре внушительная доля круглосуточных заведений - бары. Также относительно остальных округов выделяются доли кофеен и ресторанов.

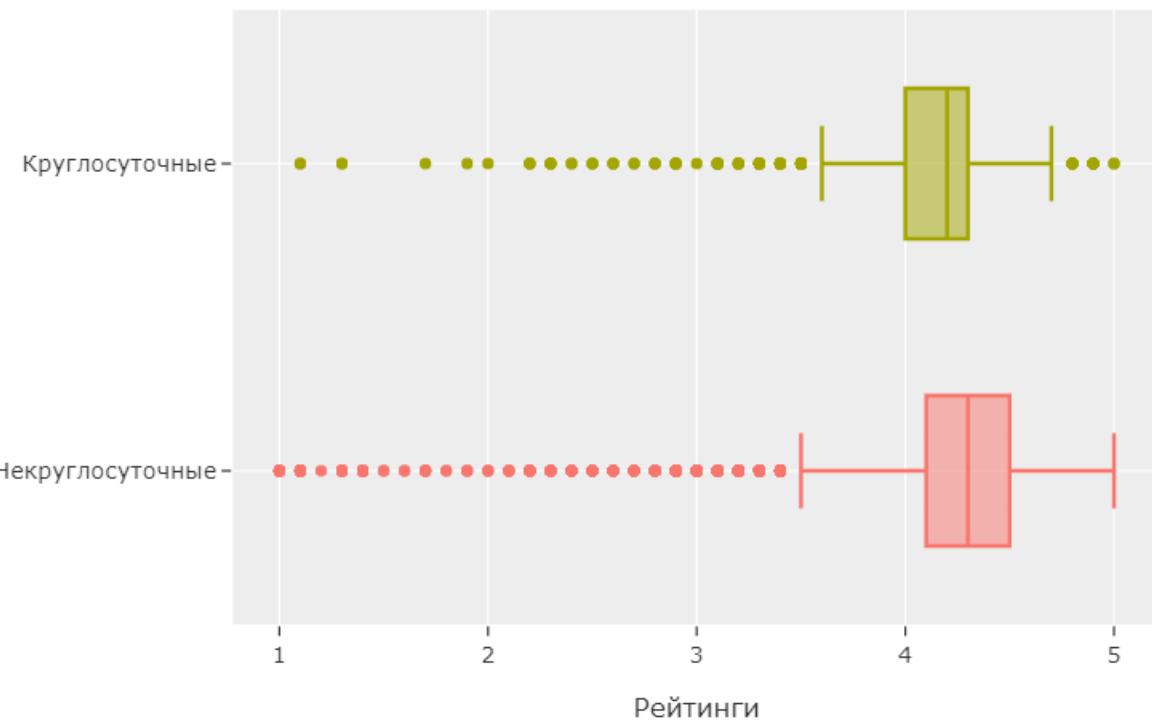
У всех округов (кроме ЮЗАО) **основные типы круглосуточных заведений - кафе и быстрое питание**, что логично.

Посмотрим на средние рейтинги и средние чеки

```
In [71]: fig = go.Figure()
fig.add_trace(go.Box(x=data.query('is_24_7 == False')['rating'], name='Некруглосуточные'))
fig.add_trace(go.Box(x=data_24_7['rating'], name='Круглосуточные'))

fig.update_layout(title='Рейтинги круглосуточных и некруглосуточных заведений',
                  xaxis_title="Рейтинг",
                  showlegend=False,
                  title_x = 0.5)
fig.show()
```

Рейтинги круглосуточных и некруглосуточных заведений



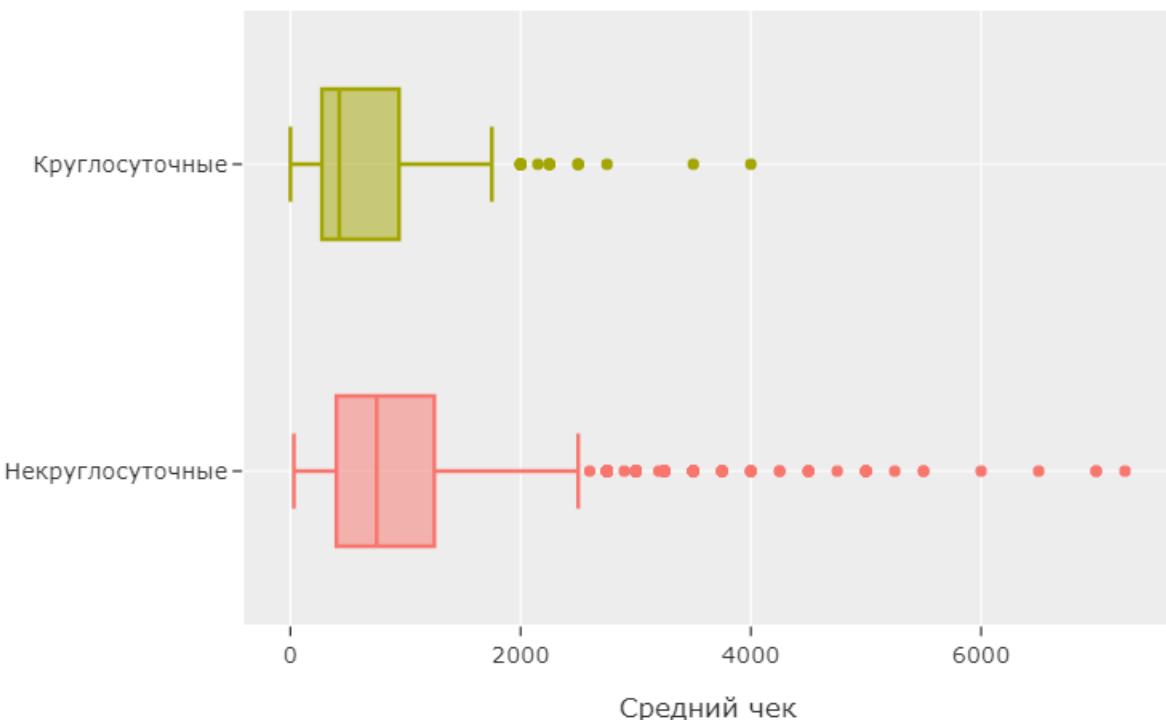
Круглосуточные заведения реже получают максимально высокие рейтинги, и имеют медианный рейтинг ниже, чем у некруглосуточных заведений

In [72]:

```
fig = go.Figure()
#срезаем аномально высокие средние чеки
fig.add_trace(go.Box(x=data.query('is_24_7 == False and middle_avg_bill < 10000')['middle_avg_bill'])
fig.add_trace(go.Box(x=data_24_7.query('middle_avg_bill < 10000')['middle_avg_bill'], name='Круглосуточные')

fig.update_layout(title='Средние чеки круглосуточных и некруглосуточных заведений',
                  xaxis_title="Средний чек",
                  showlegend=False,
                  title_x = 0.5)
fig.show()
```

Средние чеки круглосуточных и некруглосуточных заведений



Круглосуточные заведения имеют более низкий средний чек, обусловлено засильем более "дешевых" заведений

Итог: круглосуточный режим работы нельзя называть очевидным преимуществом (более низкий рейтинг и средний чек), но т.к. доля таких заведений мала, выбор такого графика при открытии может выделить заведение среди остальных

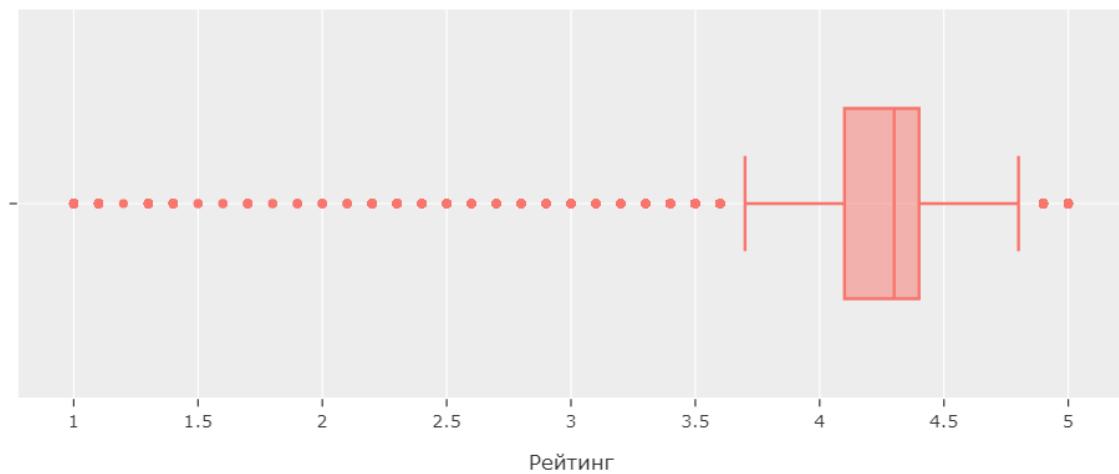
Заведения с низкими рейтингами

Чтобы определить какой рейтинг считать "плохим", посмотрим на диаграмму размаха

```
In [73]: fig = px.box(data, x="rating")
fig.update_layout(title='Рейтинги заведений',
                  yaxis_title="",
                  xaxis_title="Рейтинг",
                  showlegend=False,
                  title_x = 0.5,
                  height=400, width=900)

fig.show()
```

Рейтинги заведений



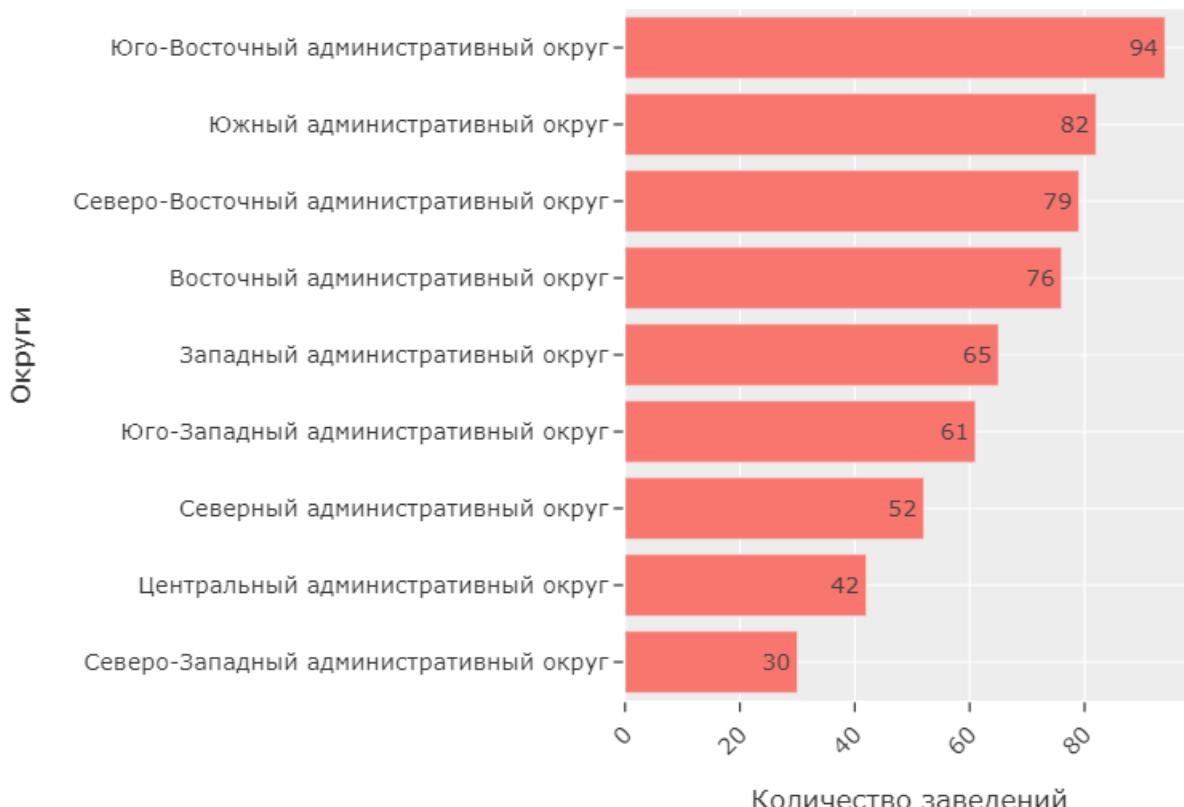
Нижняя граница 3.7 при медиане 4.3. Отберем заведения с рейтингом <3.7 и изучим их особенности

```
In [74]: # через 'плохих' заведений  
bad_rest = data.query('rating <3.7 ')
```

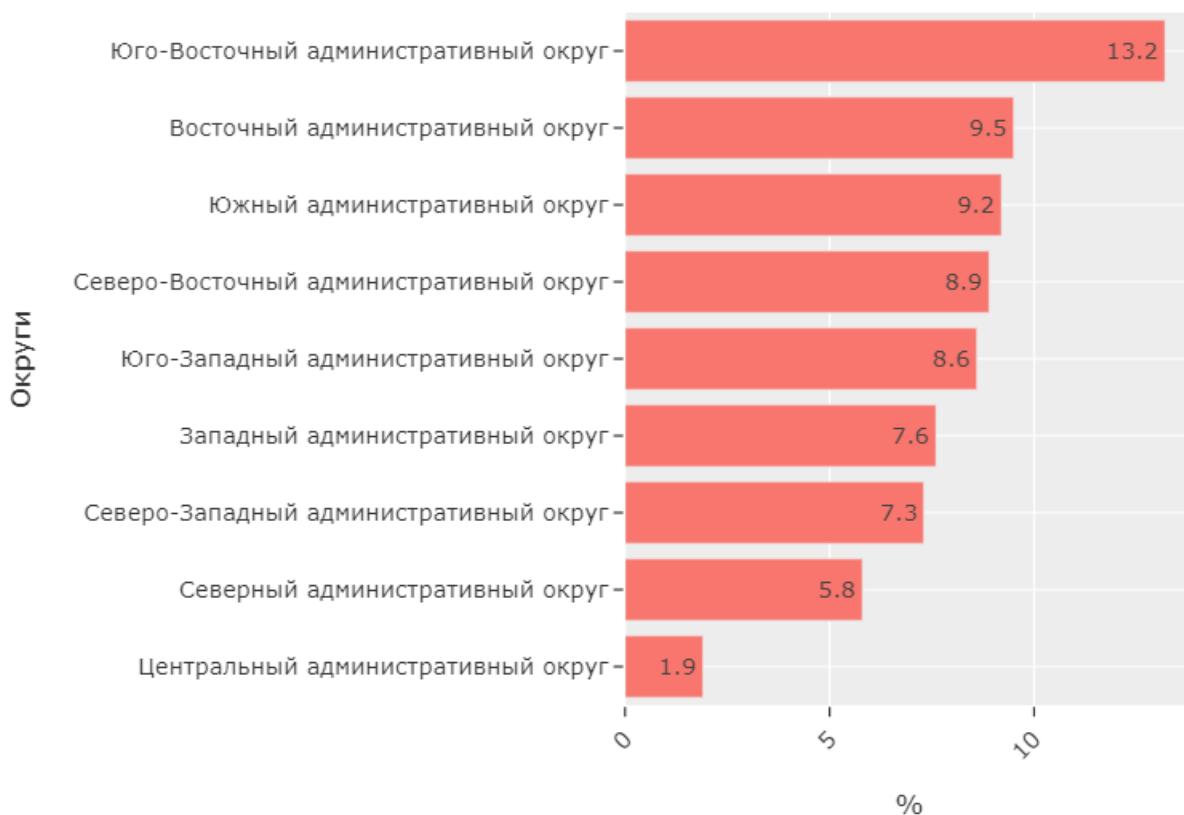
```
In [75]: #таблица: подсчет 'плохих' заведений по округам, расчет доли таких заведений  
bad_rest_district = bad_rest.groupby('district')['name'].count().sort_values().reset_index()  
total_district = data.groupby('district')['name'].count().sort_values().reset_index()  
bad_rest_district = bad_rest_district.merge(total_district[['district','name']], on = 'district')  
bad_rest_district.rename(columns={'name_x':'bad_rest', 'name_y':'total_rest'}, inplace=True)  
bad_rest_district['bad_ratio'] = (bad_rest_district['bad_rest']/bad_rest_district['total_rest'])  
bad_rest_district['bad_ratio']=bad_rest_district['bad_ratio'].round(1)
```

```
In [76]: fig = px.bar(bad_rest_district.sort_values(by='bad_rest'), y='district', x='bad_rest', text='bad_rest'  
fig.update_layout(xaxis_tickangle=-45, title='Количество заведений с низким рейтингом по округам')  
fig.update_yaxes(title_text = "Округи")  
fig.update_xaxes(title_text = "Количество заведений")  
fig.show()  
  
fig = px.bar(bad_rest_district.sort_values(by='bad_ratio'), y='district', x='bad_ratio',text='bad_ratio'  
fig.update_layout(xaxis_tickangle=-45, title='Доли заведений с низким рейтингом внутри округа')  
fig.update_yaxes(title_text = "Округи")  
fig.update_xaxes(title_text = "%")  
fig.show()
```

Количество заведений с низким рейтингом по округам



Доли заведений с низким рейтингом внутри округа, %



Больше всего заведений с низким рейтингом расположены в ЮВАО, ЮАО, СВАО и ВАО. Но доли таких заведений среди остальных упорядочены по округам иначе.

Топ округов по доле "плохих" заведений:

- ЮВАО - 12%
- ВАО - 9%

- ЮАО - 8.2%
- СВАО - 8.1%

В центре заведений с низким рейтингом меньше всего (1.16%)

In [77]:

```
bad_rest_category = bad_rest.groupby('category')['name'].count().sort_values(ascending=False)

fig = go.Figure(data=[go.Bar(x=bad_rest_category['category'], y=bad_rest_category['name'], te
fig.update_layout(title='Категории заведений с рейтингом <3.7')
fig.update_xaxes(title='Категории')
fig.update_yaxes(title='Количество заведений')
fig.show()
```



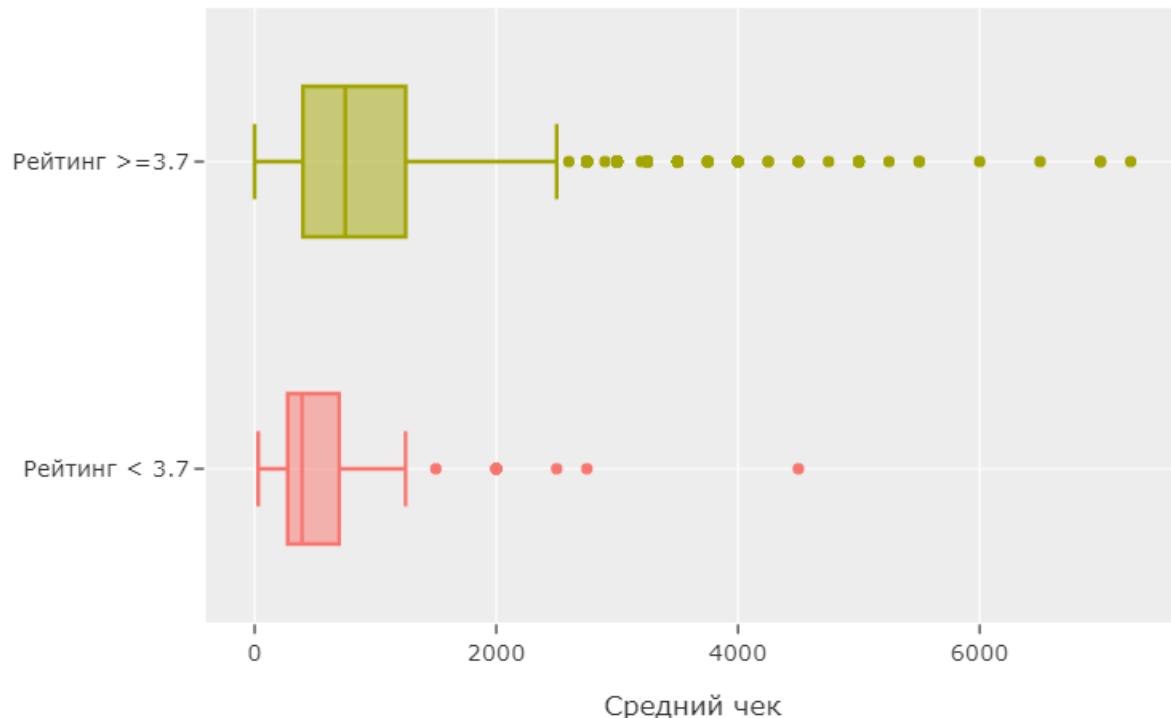
Большая часть заведений с низким рейтингом относятся к категории **кафе**

In [78]:

```
fig = go.Figure()
fig.add_trace(go.Box(x=bad_rest['middle_avg_bill'], name='Рейтинг < 3.7'))
fig.add_trace(go.Box(x=data.query('rating >=3.7 and middle_avg_bill<10000')['middle_avg_bill'])

fig.update_layout(title='Средние чеки заведений: заведения с низким рейтингом и остальные',
                  xaxis_title="Средний чек",
                  showlegend=False,
                  title_x = 0.5)
fig.show()
```

Средние чеки заведений: заведения с низким рейтингом и остальные



Как правило, заведения с низким рейтингом имеют низкий средний чек (медиана 392.5). Однако видим, что и дорогие заведения тоже могут получать низкую оценку

```
In [79]: fig = go.Figure(data=[go.Pie(labels=bad_rest['chain'])])
fig.update_layout(title='Заведения с низким рейтингом: сетевые и несетевые', showlegend=False
fig.update_traces(textinfo='label+percent', textposition='inside')
fig.show()
```

Заведения с низким рейтингом: сетевые и несетевые



Интересное наблюдение: чуть более трети таких заведений - сетевые. Несмотря на то, что как правило у сетей есть определенный стандарт и кухни, и обслуживания, их заведения вполне могут получать низкие оценки.

Итог: "лидер" по наличию заведений с низким рейтингом - Юго-Восточный округ(12% заведений имеют рейтинг <3.6).

Среди заведений с низким рейтингом больше всего кафе, как правило у таких заведений достаточно низкий средний чек. Внутренние стандарты качества сетевых заведений не гарантируют удовлетворенность клиентов, треть "плохих" заведений - сетевые

Выходы

Проведен исследовательский анализ заведений. Основные наблюдения:

- **Категории**
 - всего восемь категорий кафе, ресторан, кофейня, бар(паб), пиццерия, быстрое питание, столовая, булочная
 - самые распространенные заведения: кафе(2378, 28.3%), ресторан(2043, 24.3%), кофейня(1413, 16.8%)
- **Посадочные места**
 - медианное значение по всем категориям: 70 мест
 - самые большие медианные значения: ресторан (80 мест), бар/паб(80), столовая (73), кофейня(70)
- **Рейтинги по категориям**
 - самый высокий средний рейтинг у баров и пабов (4.38)
 - самый низкий у быстрого питания (4.0)
- **Сетевые заведения**

- 38.1% заведений - сетевые
- численно больше остальных встречаются сетевые кафе(779), рестораны(730) и кофейни(720)
- чаще всего сетевыми являются: булочные(61% от всех заведений в категории), пиццерии(52%), кофейни(51%)

- **Топ-15 сетей**

- самые распространенные типы заведений: кафе, кофейня, ресторан, быстрое питание, пиццерия
- больше всего заведений: кофейни(337), ресторан(186), пиццерия(152)
- относительно других сетей имеют более низкие средние чеки
- большинство из этих сетей распространены по всем округам. Численно больше всего заведений расположено в ЦАО

- **Административные округа**

- Всего 9 округов. Больше всего заведений в:
 - Центральный: 2242
 - Северный: 900
 - Южный: 892
 - Северо-Восточный: 891
 - Западный: 851
 - Восточный: 798
- топ-3 категории в каждом округе: кафе, ресторан, кофейня (доли внутри округа могут отличаться)
- самые "дорогие" округи: ЦАО и ЗАО, у обоих медианный средний чек 1000, что больше медианы по всем округам
- самые низкие средние чеки: ЮАО, СВАО, ЮВАО (500, 500, 450 соответственно)
- по средним рейтингам:
 - лучшие округи: ЦАО, САО (4.38, 4.23 соответственно)
 - худший округ: ЮВАО, 4.1

- **Топ-15 улиц по количеству заведений:**

- входят одни из самых длинных московских улиц, шоссе и проспектов
- чаще всего располагаются рестораны и кафе, но в разных долях

- **Улицы только с одним заведением:**

- чаще всего расположены кафе, рестораны, кофейни, 29% заведений - сетевые
- в ЮАО, ЮЗАО, ЗАО и ВАО обычно располагаются заведения с рейтингом и средним чеком ниже, чем остальные заведения в округе
- в ЮВАО, ЦАО и СЗАО - наоборот, обычно более дорогие заведения с лучшим рейтингом

- **Круглосуточные заведения**

- имеют более низкие рейтинг и средний чек
- суммарно всего 8.7% заведений

- **Заведения с низким рейтингом**

- Округи с большими долями "плохих" заведений: ЮВАО(12%), ВАО(9%), ЮАО(8.2%), СВАО(8.1%)
- чаще остальных категорий встречаются кафе
- 32% сетевых заведений

Детализация исследования: открытие кофейни

- Сколько всего кофеен в датасете? В каких районах их больше всего, каковы особенности их расположения?

- Есть ли круглосуточные кофейни?
- Какие у кофеен рейтинги? Как они распределяются по районам?
- На какую стоимость чашки капучино стоит ориентироваться при открытии и почему?

Кофейни по округам

In [80]:

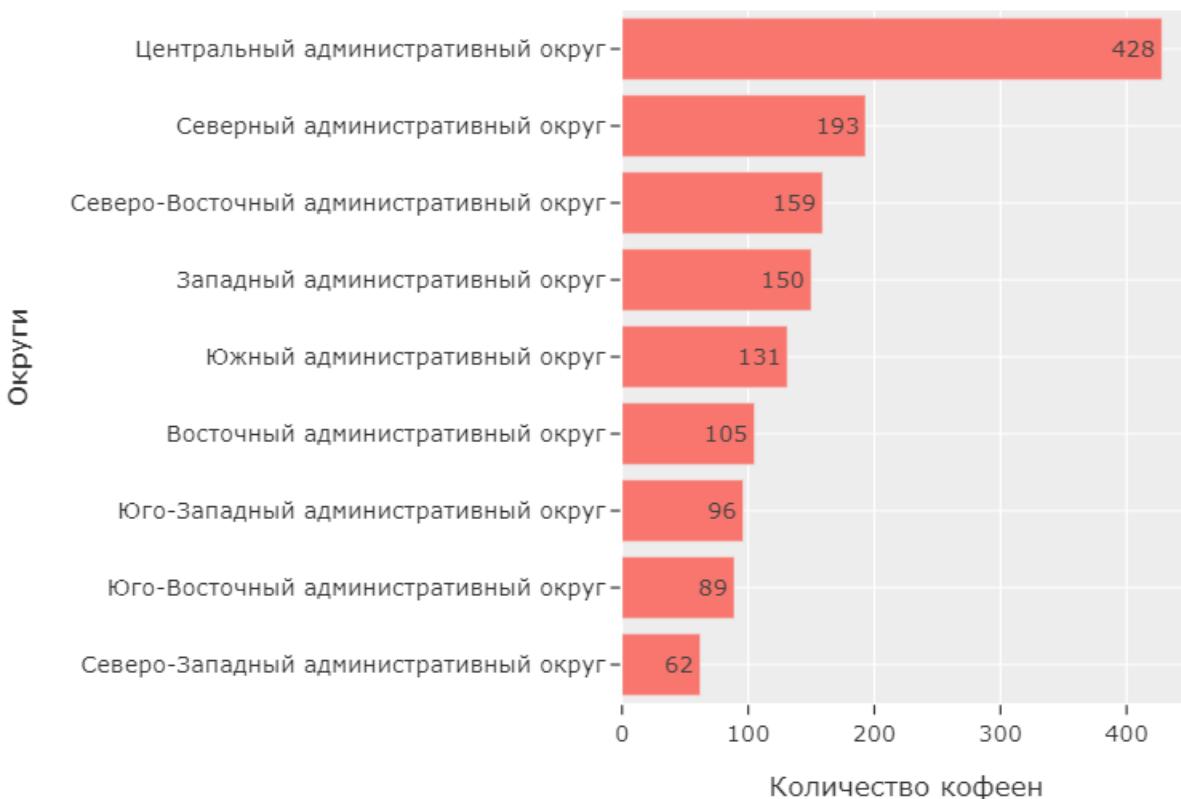
```
#сохранение всех кафе в переменной:
cafe = data.query('category=="кофейня"')
# таблица с группировкой по регионам, подсчет кафе, добавление столбца с долей кафе
cafe_district = cafe.groupby('district')['name'].count().reset_index()
total_rest_distrct = data.groupby('district')['name'].count().reset_index()
cafe_district = cafe_district.merge(total_rest_distrct[['district','name']], on = 'district')
cafe_district = cafe_district.rename(columns={'name_x':'cafe_count', 'name_y':'total_count'})
cafe_district['cafe_ratio'] = (cafe_district['cafe_count']/cafe_district['total_count'])*100
cafe_district = cafe_district.round({'cafe_ratio':2})
```

In [81]:

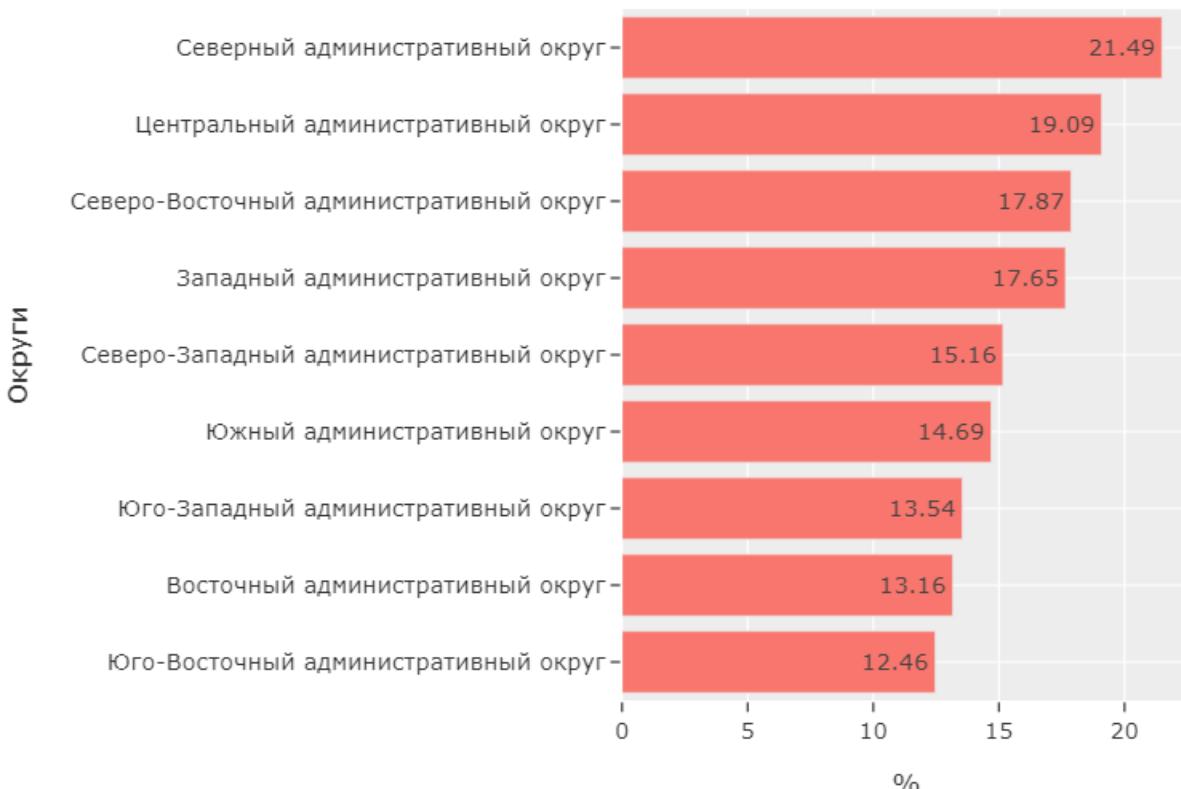
```
fig = px.bar(cafe_district,
              y='district',
              x='cafe_count',
              text='cafe_count')
fig.update_layout(title='Количество кофеен по округам',
                  xaxis_title='Количество кофеен',
                  yaxis_title='Округи',
                  yaxis={'categoryorder':'total ascending'})
fig.show()

fig = px.bar(cafe_district,
              y='district',
              x='cafe_ratio',
              text='cafe_ratio')
fig.update_layout(title='Доли кофеен среди остальных заведений по округам, %',
                  xaxis_title='%',
                  yaxis_title='Округи',
                  yaxis={'categoryorder':'total ascending'})
fig.show()
```

Количество кофеен по округам



Доли кофеен среди остальных заведений по округам, %



И по числу кофеен, и по доле среди других категорий заведений выделяются три округа:

- **ЦАО** - 428 кофеен, 19.1% от всех заведений в округе
- **САО** - 193 кофейни, 21.44% от всех заведений (самое большое значение)
- **СВАО** - 159 кофеен, 17.9% от всех заведений

Круглосуточные кофейни

```
In [82]: cafe_24 = cafe.query('is_24_7==True')
cafe_24_district = cafe_24.groupby('district')['name'].count().reset_index()

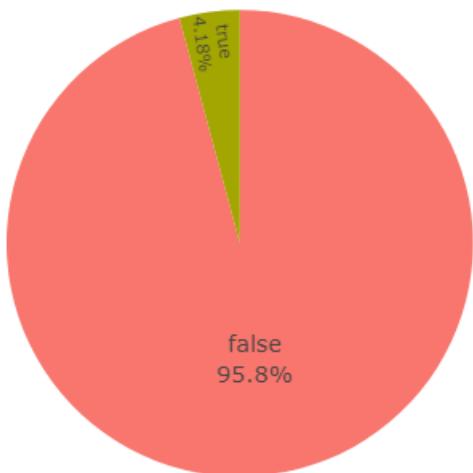
fig = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'}, {'type':'domain'}]],
                     subplot_titles=['Доля круглосуточных', 'Из них сетевых'])

fig.add_trace(go.Pie(labels=cafe['is_24_7']),
              1, 1)
fig.add_trace(go.Pie(labels=cafe_24['chain']),
              1, 2)

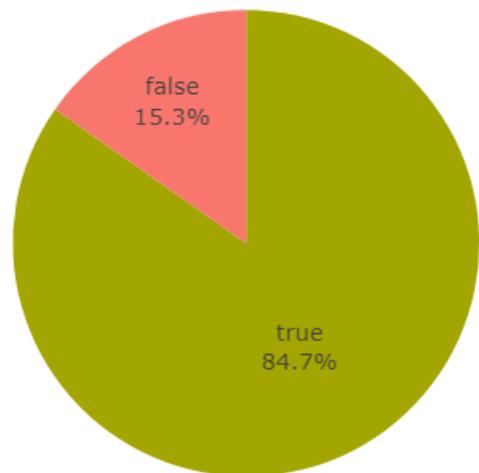
fig.update_layout(xaxis_tickangle=-45, title='Круглосуточные кофейни', title_x = 0.5, showleg
fig.update_traces(textinfo='label+percent', textposition='inside')
fig.show()
```

Круглосуточные кофейни

Доля круглосуточных



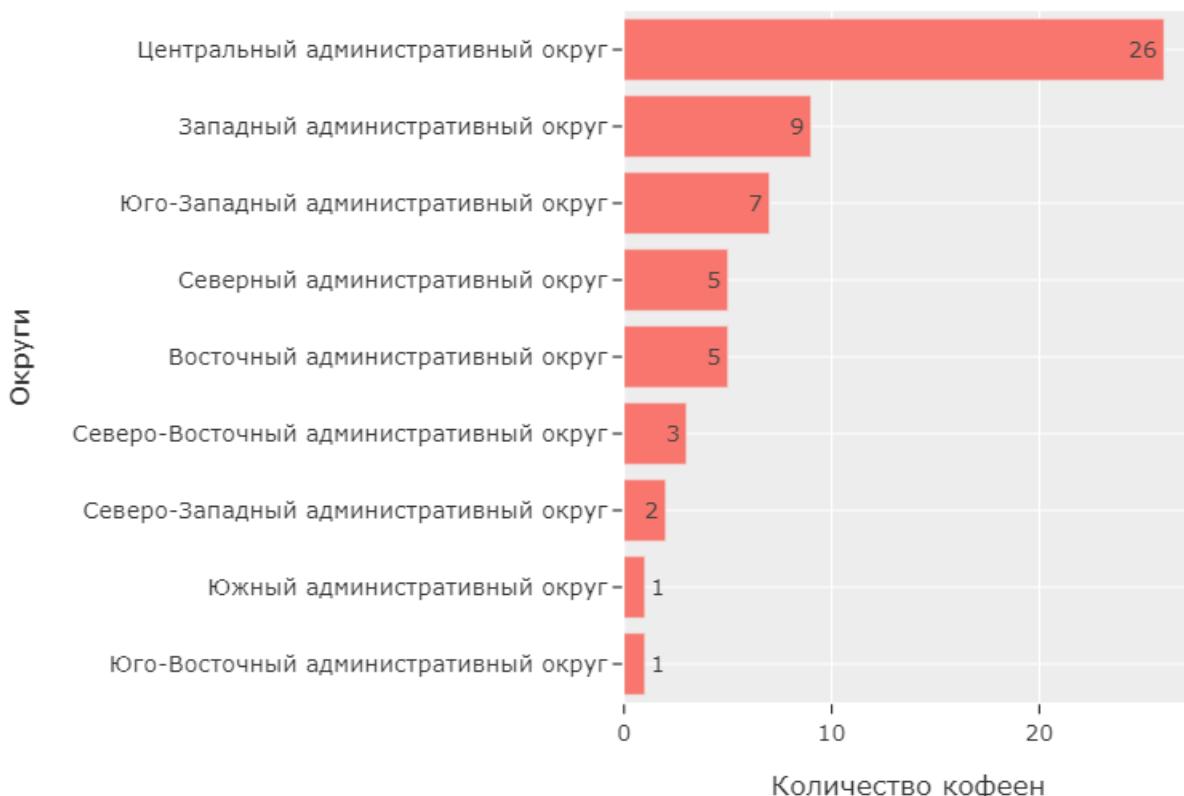
Из них сетевых



Круглосуточных кофеен немного, всего 59. Как правило, круглосуточными являются сетевые кофейни

```
In [83]: fig = px.bar(cafe_24_district,
                  y='district',
                  x='name',
                  text='name')
fig.update_layout(title='Круглосуточные кофейни по округам',
                  xaxis_title='Количество кофеен',
                  yaxis_title='Округи',
                  yaxis={'categoryorder':'total ascending'})
fig.show()
```

Круглосуточные кофейни по округам



Больше всего круглосуточных кофеен в ЦАО, что логично: люди гуляют в центре и днем, и ночью.

Больше всего несетевых кофеен не работают круглосуточно - возможно, стоит перенять их опыт и не ориентироваться на рабочий график 24/7

Рейтинги кофеен по округам

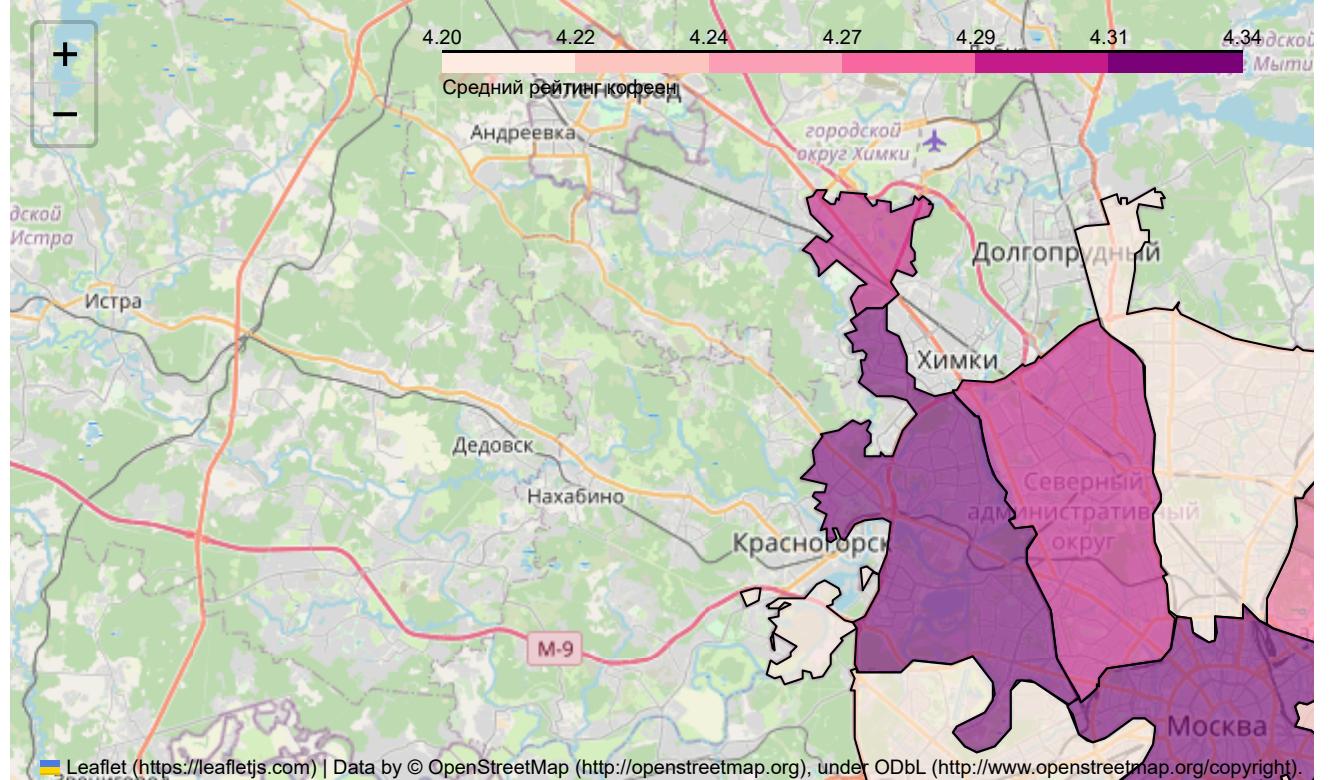
```
In [84]: # таблица: средние рейтинги кофеен по округам
cafe_district_mean = cafe.groupby('district')['rating'].mean().reset_index()
```

```
In [85]: m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
```

```
Choropleth(
    geo_data=state_geo,
    data = cafe_district_mean,
    columns = ['district', 'rating'],
    key_on = 'feature.name',
    legend_name = 'Средний рейтинг кофеен',
    fill_color='RdPu'
).add_to(m)

m
```

Out[85]:



- Самые высокие рейтинги у ЦАО и СЗАО
- Самые низкие у СЭАО и ЗАО

Дополнительный анализ

Рассчитаем среднюю стоимость чашки кофе, средний чек и количество посадочных мест по всей Москве.

По результатам всех предыдущих исследований мы определили, что **ЦАО сильно выделяется среди остальных округов** - и по количеству заведений, и по средним чекам, и по рейтингам. Есть смысл рассматривать округ отдельно от других.

Средняя стоимость чашки капучино

In [86]:

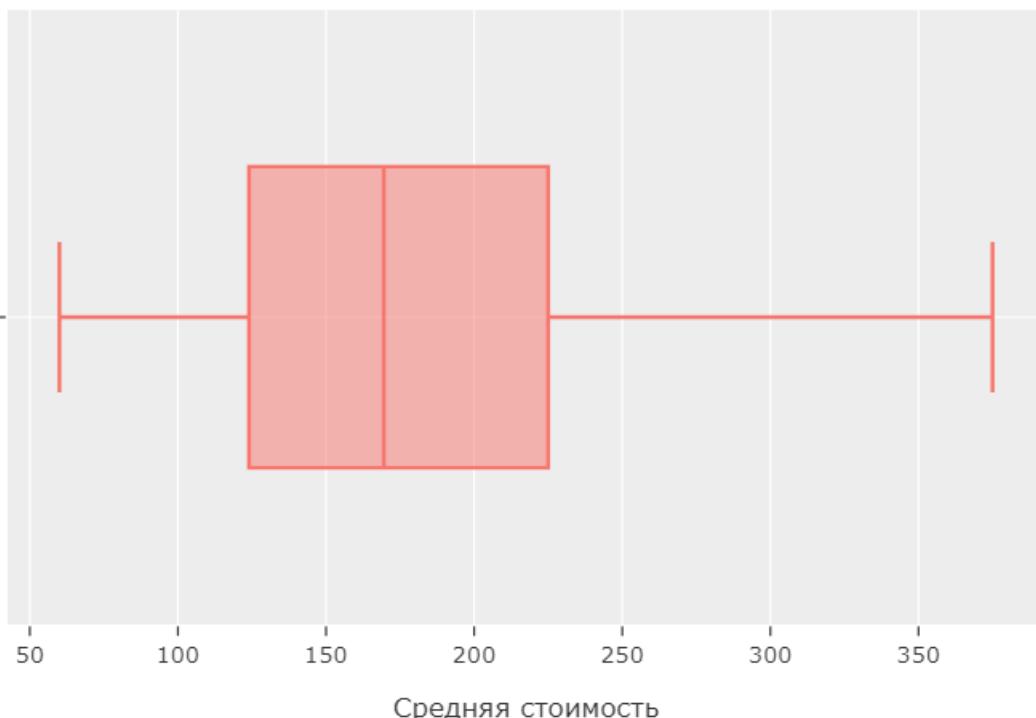
```
# удаляем аномально высокую стоимость в обоих графиках:
fig = go.Figure()
fig.add_trace(go.Box(x=cafe.query('middle_coffee_cup<1500')['middle_coffee_cup'], name=''))
fig.update_layout(title='Средняя стоимость чашки кофе: все кофейни Москвы',
                  xaxis_title="Средняя стоимость",
                  showlegend=False,
                  title_x = 0.5)
fig.show()

fig = go.Figure()
fig.add_trace(go.Box(x=cafe.query('district != "Центральный административный округ" \
                                    and middle_coffee_cup<1500')['middle_coffee_cup'],
                     name='Остальные'))

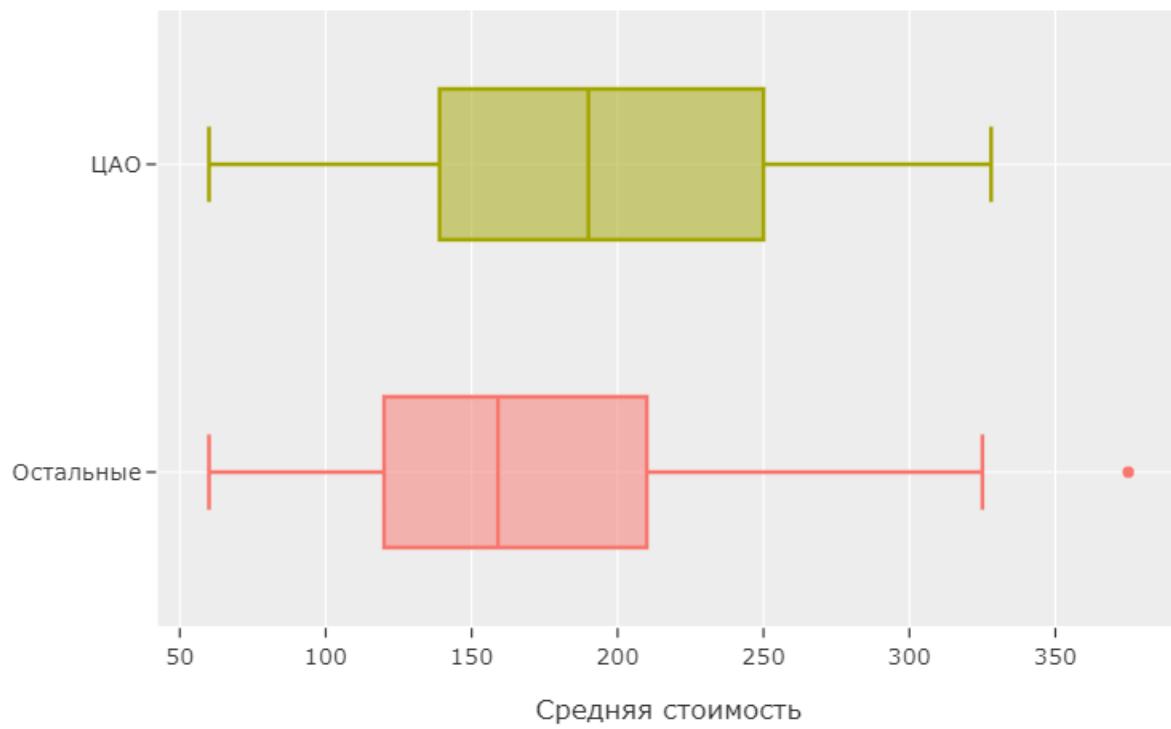
fig.add_trace(go.Box(x=cafe.query('district == "Центральный административный округ"' )['middle_coffee_cup'],
                     name='ЦАО'))

fig.update_layout(title='Средняя стоимость чашки кофе по округам',
                  xaxis_title="Средняя стоимость",
                  showlegend=False,
                  title_x = 0.5)
fig.show()
```

Средняя стоимость чашки кофе: все кофейни Москвы



Средняя стоимость чашки кофе по округам



Медианная средняя стоимость чашки кофе по Москве: 169.5. Большинство значений расположены в диапазоне 124-225

Если мы ориентируемся на открытие в ЦАО: рекомендуемая стоимость чашки кофе **139-250 рублей**

В других округах: **120-210 рублей**

Средний чек

Если мы ориентируемся на заведение, в котором можно не только выпить кофе, но и провести время, определим границы среднего чека.

In [87]:

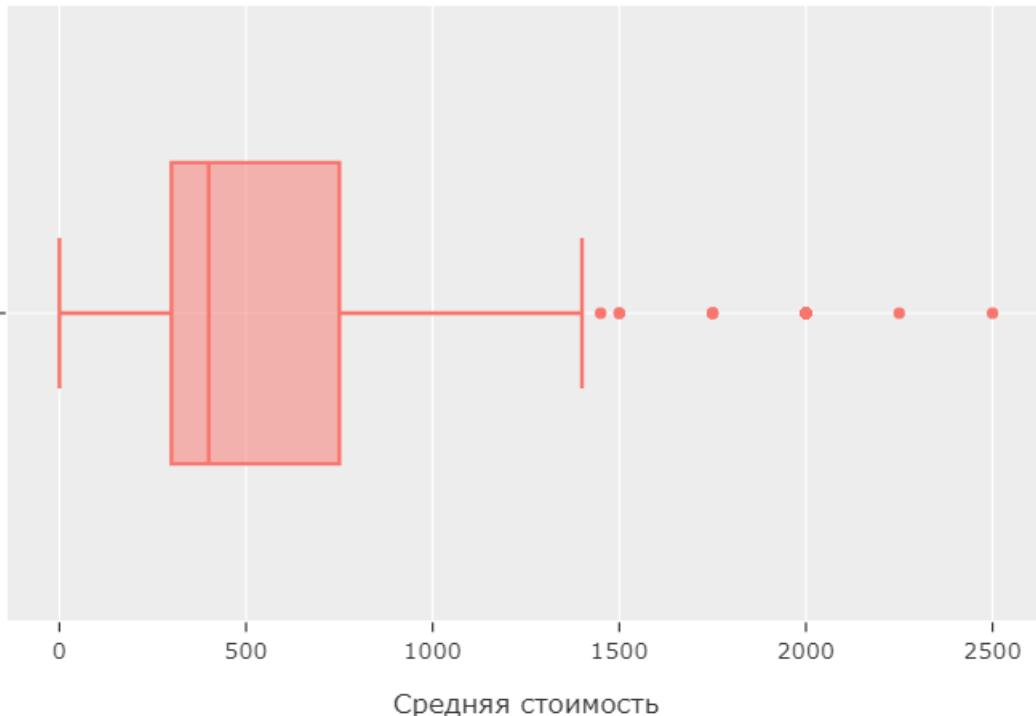
```
fig = go.Figure()
fig.add_trace(go.Box(x=cafe['middle_avg_bill'], name=''))
fig.update_layout(title='Средний чек: все кофейни Москвы',
                  xaxis_title="Средняя стоимость",
                  showlegend=False,
                  title_x = 0.5)
fig.show()

fig = go.Figure()
fig.add_trace(go.Box(x=cafe.query('district != "Центральный административный округ"' )['middle']
name='Остальные'))

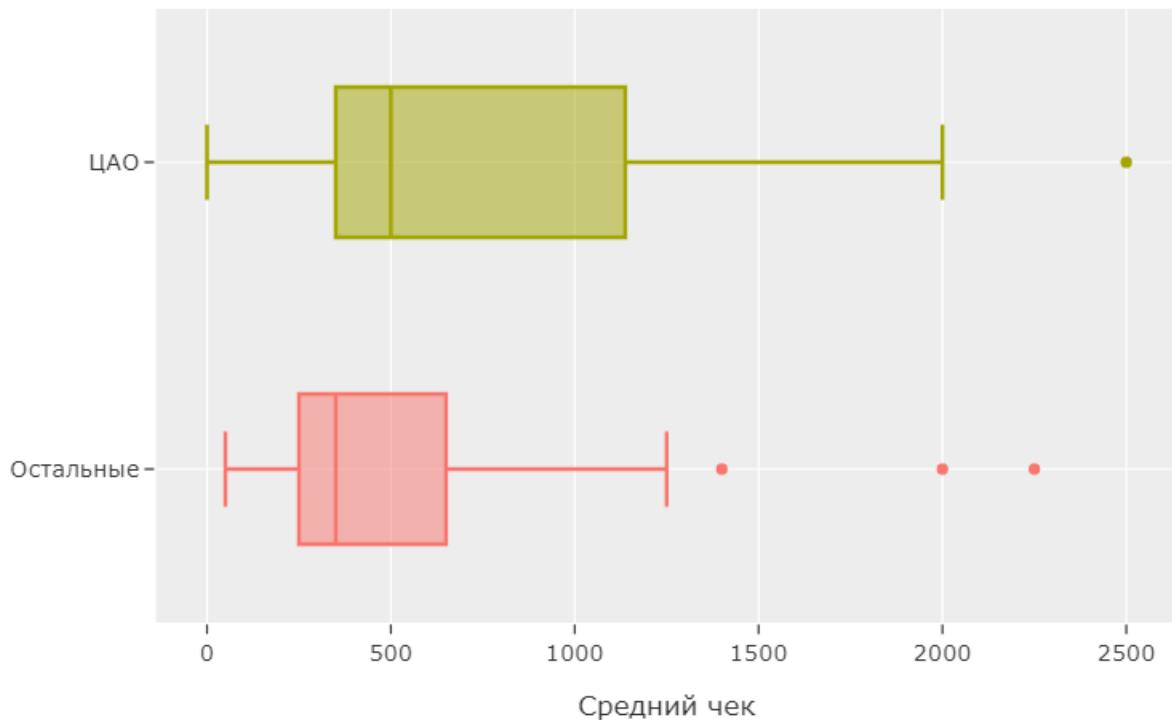
fig.add_trace(go.Box(x=cafe.query('district == "Центральный административный округ"' )['middle']
name='ЦАО'))

fig.update_layout(title='Средний чек по округам',
                  xaxis_title="Средний чек",
                  showlegend=False,
                  title_x = 0.5)
fig.show()
```

Средний чек: все кофейни Москвы



Средний чек по округам



Медианная средний чек кофеен по всей Москве: 400. Большинство значений расположены в диапазоне 300-750

- В ЦАО: ориентируемся на средний чек **350-1137**, медиана 500
- в остальных округах: **250-650**, медиана 350

Посадочные места

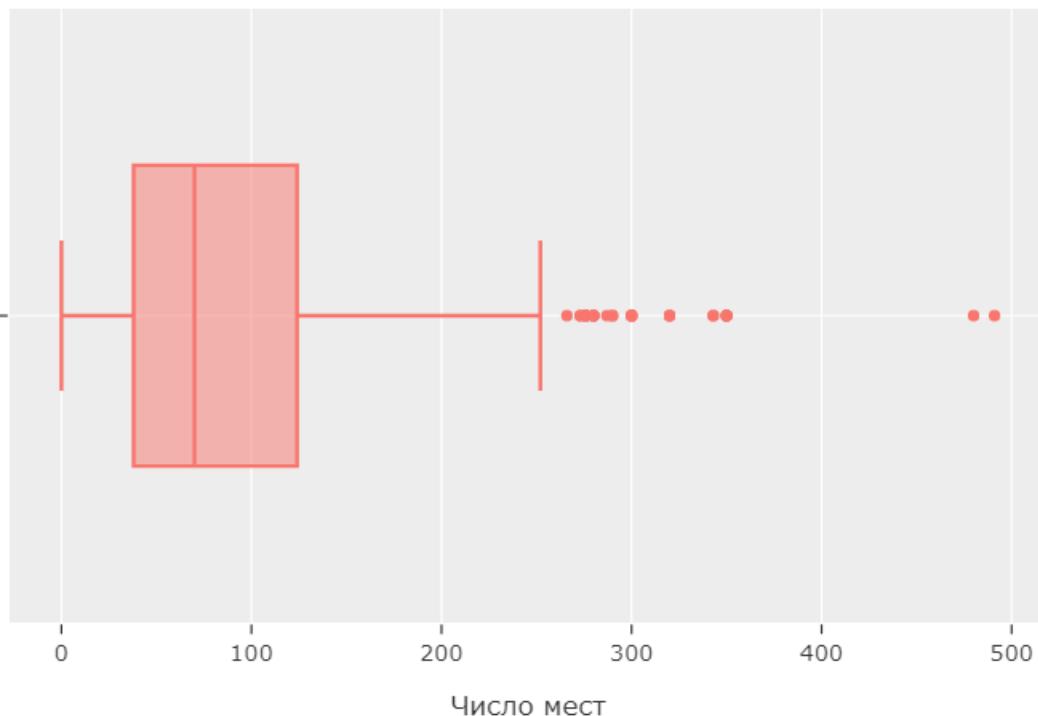
```
In [88]: fig = go.Figure()
fig.add_trace(go.Box(x=cafe.query('seats !=9999')['seats'], name=''))
fig.update_layout(title='Посадочные места: все кофейни Москвы',
                  xaxis_title="Число мест",
                  showlegend=False,
                  title_x = 0.5)
fig.show()

fig = go.Figure()
fig.add_trace(go.Box(x=cafe.query('district != "Центральный административный округ" \
                                    and seats!=9999')['seats'],
                     name='Остальные'))

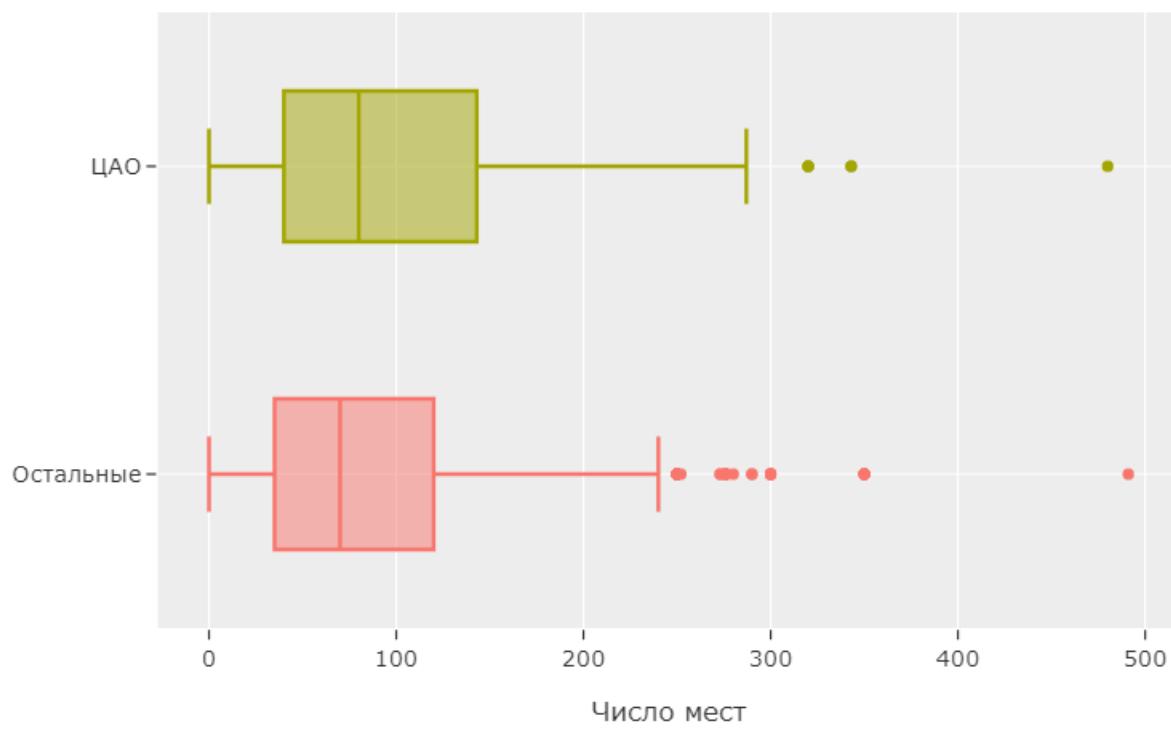
fig.add_trace(go.Box(x=cafe.query('district == "Центральный административный округ" and seats \
name="ЦАО"'))

fig.update_layout(title='Посадочные места: разброс данных по округам',
                  xaxis_title="Число мест",
                  showlegend=False,
                  title_x = 0.5)
fig.show()
```

Посадочные места: все кофейни Москвы



Посадочные места: разброс данных по округам



Медианное число посадочных мест в кофейнях Москвы: 70. Большинство значений расположены в диапазоне 38-124

Если мы не ориентируемся на кофейню формата "на вынос", то:

- В ЦАО лучше ориентироваться на диапазон **40-143 посадочных мест**.
- В остальных округах: компактнее, **35-120 мест**.

Сетевые кофейни

```
In [89]: #таблица: группировка по имени, подсчет заведений, срез первых десяти по числу заведений
cafe_top = cafe.groupby('name').agg({'category':'count'}).reset_index()
cafe_top = cafe_top.sort_values(by='category', ascending=False).head(10)
cafe_top['ratio'] = (cafe_top['category'] / len(cafe))*100
cafe_top_name = cafe_top['name'].tolist()
```

```
In [90]: #таблица с количеством заведений топ-10 и остальных кофеен
cafe_top_sum = cafe_top['category'].sum()
cafe_not_top = len(cafe) - cafe_top_sum
cafe_top_ratio = pd.DataFrame({'name': ['Топ-10', 'Остальные'], 'total': [cafe_top_sum, cafe
```

```
In [91]: print('Доля сетевых кофеен:', "{0:.1f}%".format((cafe['chain'].sum()/len(cafe))*100))
print('В среднем на одну сеть кофеен приходится', \
      round(cafe.query('chain==True')['name'].count() / cafe.query('chain==True')['name'].nunique()))
      'заведений')

fig = make_subplots(rows=1, cols=2, specs=[[{'type':'Bar'}, {'type':'domain'}]],
                     subplot_titles=['Количество заведений', 'Доля Топ-10 среди всех кофеен'])

fig.add_trace(go.Bar(y=cafe_top['category'],
                      x=cafe_top['name'],
                      showlegend=False),
              1, 1)
fig.add_trace(go.Pie(labels=cafe_top_ratio['name'], values=cafe_top_ratio['total'], legendgroup=1,
                     1, 2)

fig.update_traces(hoverinfo="all")
fig.update_layout(xaxis_tickangle=-45, title='Топ-10 кофеен по числу заведений', title_x = 0.
fig.update_yaxes(title='Количество заведений', col=1, row=1)
fig.update_xaxes(title='Названия сетей', col=1, row=1)
fig.show()
```

Доля сетевых кофеен: 51.0%

В среднем на одну сеть кофеен приходится 4.6 заведений

Топ-10 кофеен по числу заведений



Среди всех кофеен чуть более половины - **сетевые**. На одну сеть в среднем приходится 4.6 заведений.

Стоит обратить внимание на топ-10 кофеен: они занимают 28.5% от всего рынка кофеен и распространены намного шире. С распространенными и узнаваемыми сетями **конкурировать сложнее** - обычно они имеют постоянный поток клиентов и могут держать цены на определенном уровне. При выборе места для открытия кофейни **необходимо учитывать расположение кофеен из топ-10**, не выбирать места вблизи них

Рейтинги

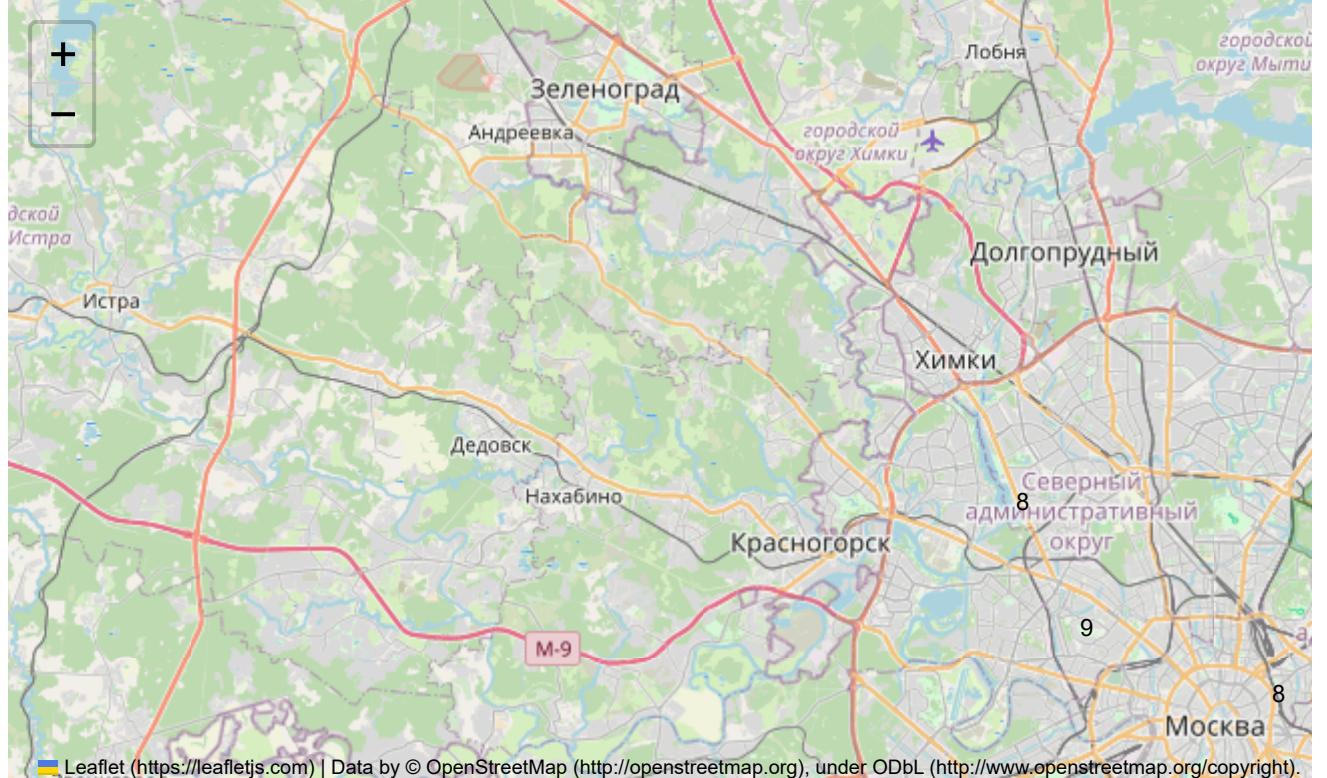
```
In [92]: best_cafe = cafe.query('rating == 5')
```

```
In [93]: m = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
# создаем пустой кластер, добавляем его на карту
marker_cluster = MarkerCluster().add_to(m)
# функция, принимающая строку датафрейма, создает маркер в текущей точке и
# добавляет его в кластер marker_cluster
def create_cluster(row):
    Marker(
        [row['lat'], row['lng']],
        popup=f'{row['name']} {row['rating']}',
    ).add_to(marker_cluster)

# к каждой строке датафрейма применяем функцию
best_cafe.apply(create_cluster, axis=1)

# выводим карту
m
```

Out[93]:



На карте отображены лучшие кофейни - с рейтингом 5. Рекомендуется провести **дополнительный анализ этих заведений**, выяснить, что именно привлекает пользователей в этих заведениях, какие у них особенности в интерьере, кухне, как они ведут соц.сети и т.д.

Посмотрим на средние цены чашки кофе для кофеен с **рейтингом выше медианного и для кофеен, недотягивающих до оценки "хорошо"**

In [94]:

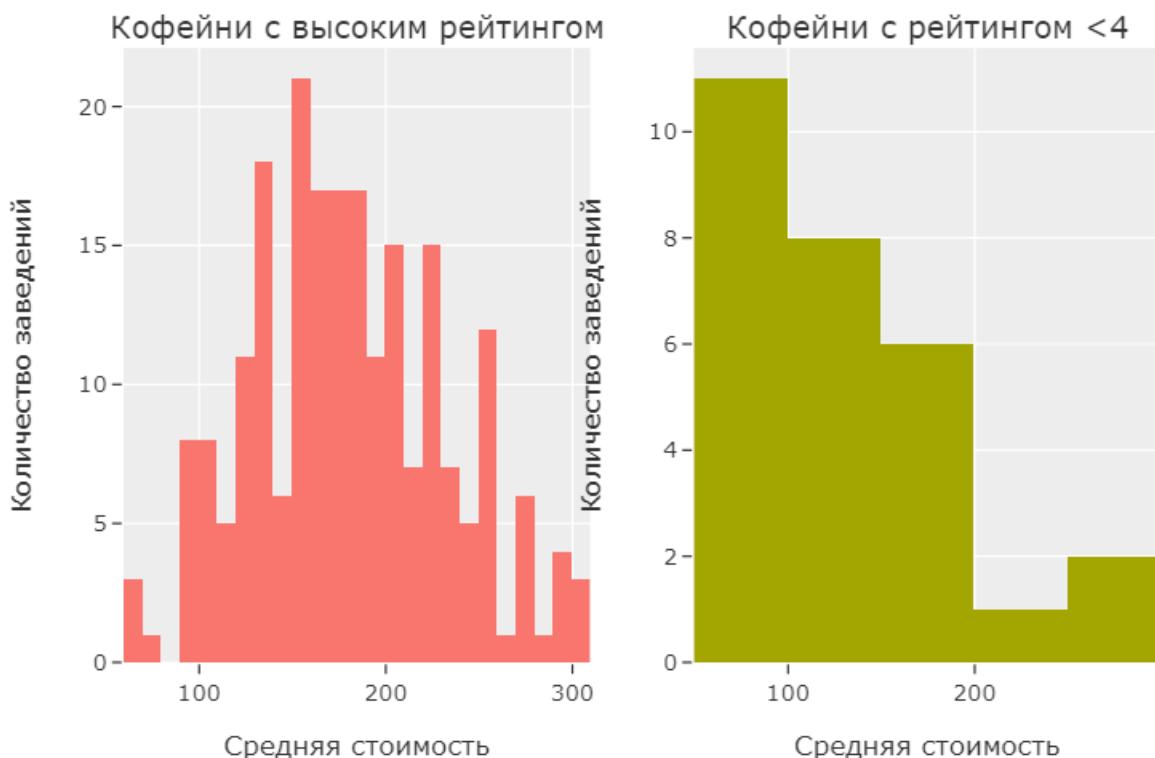
```
cafe_median_rating = cafe['rating'].median()
good_cafe = cafe.query('rating > @cafe_median_rating')
bad_cafe = cafe.query('rating < 4')
```

In [95]:

```
fig = make_subplots(rows=1, cols=2, subplot_titles=['Кофейни с высоким рейтингом', 'Кофейни с низким рейтингом'])
trace0 = go.Histogram(x=good_cafe['middle_coffee_cup'], nbinsx=30)
trace1 = go.Histogram(x=bad_cafe['middle_coffee_cup'], nbinsx = 5)
fig.update_layout(title='Средняя стоимость чашки кофе', showlegend=False)

fig.append_trace(trace0, 1, 1)
fig.append_trace(trace1, 1, 2)
fig.update_yaxes(title='Количество заведений', col=1, row=1)
fig.update_xaxes(title='Средняя стоимость', col=1, row=1)
fig.update_yaxes(title='Количество заведений', col=2, row=1)
fig.update_xaxes(title='Средняя стоимость', col=2, row=1)
fig.update_layout(showlegend=False)
fig.show()
```

Средняя стоимость чашки кофе



Самый частый диапазон цен у кофеен с высоким рейтингом 150-209р. Кофейни с рейтингом ниже 4 обычно продают кофе дешевле.

Возможно, кофейни с низкими рейтингами получают плохие оценки из-за дешевизны и, соответственно, вкусовых качеств кофе. Не стоит экономить на основном продукте

Где лучше открыть кофейню?

Лучший округ для открытия кофейни - Центральный. Почему стоит ориентироваться на ЦАО:

- **самый популярный округ** как для москвичей, так и для туристов. Больше всего заведений находятся именно в ЦАО, конкуренция высока, но и поток людей выше, чем в других округах
- среди всех остальных округов - **самый высокий средний рейтинг заведений**
- **самый высокий средний чек** - в зависимости от фин.плана, можно установить цены на том же уровне, что поможет покрыть издержки, либо установить цены ниже, чем у конкурентов, и привлечь посетителей

Расположить новую кофейню лучше на **популярных улицах** или рядом с ними. Под популярностью понимаем большое число заведений - очевидно, эти улицы отличаются большой проходимостью + люди скорее всего уже знают, что на улице много заведений общепита.

Однако **хотелось бы избежать прямых конкурентов в непосредственной близости**. Стоит ли открываться на улице, где уже есть три кофейни?

Отберем Топ-20 улиц по следующему принципу:

- на улице много заведений (улица популярна)
- на улице не более одной кофейни.

```
In [96]: # группировка по улицам, где есть кофейни, подсчет кофеен на каждой из этих улиц
streets_with_cafe = data.query('category == "кофейня"]').groupby('street')['name'].count().sort_values(ascending=False).reset_index()
# отбираем улицы, на которых расположено больше одной кофейни, сохраняем список этих улиц в переменную
streets_with_many_cafes = streets_with_cafe.query('name > 1')
streets_with_many_cafes_list = streets_with_many_cafes['street'].tolist()
```

```
In [97]: # df для ЦАО, оставляем только те улицы, на которых либо нет, либо только одна кофейня
cao_without_cafes = data.query('district == "Центральный административный округ" \
and street not in @streets_with_many_cafes_list')
top_cao_street_without_cafe_count = cao_without_cafes.groupby('street')['name'].count().sort_values(ascending=False).reset_index().head(20)
top_cao_street_without_cafe_count = top_cao_street_without_cafe_count.rename(columns={'name': 'rest_count'})
top_cao_street_without_cafe_count
```

Out[97]:

	street	rest_count
0	малая бронная улица	20
1	сущёвская улица	13
2	бакунинская улица	12
3	улица большая дмитровка	11
4	улица сергия радонежского	10
5	никольская улица	10
6	таганская площадь	10
7	кривоколенный переулок	9
8	улица большие каменщики	9
9	лужнецкая набережная	8
10	лубянский проезд	8
11	олимпийский проспект	8
12	улица талалихина	8
13	улица рогожский вал	7
14	люсиновская улица	7
15	улица сергея макеева	7
16	улица малая дмитровка	7
17	улица гиляровского	7
18	новинский бульвар	7
19	нижегородская улица	7

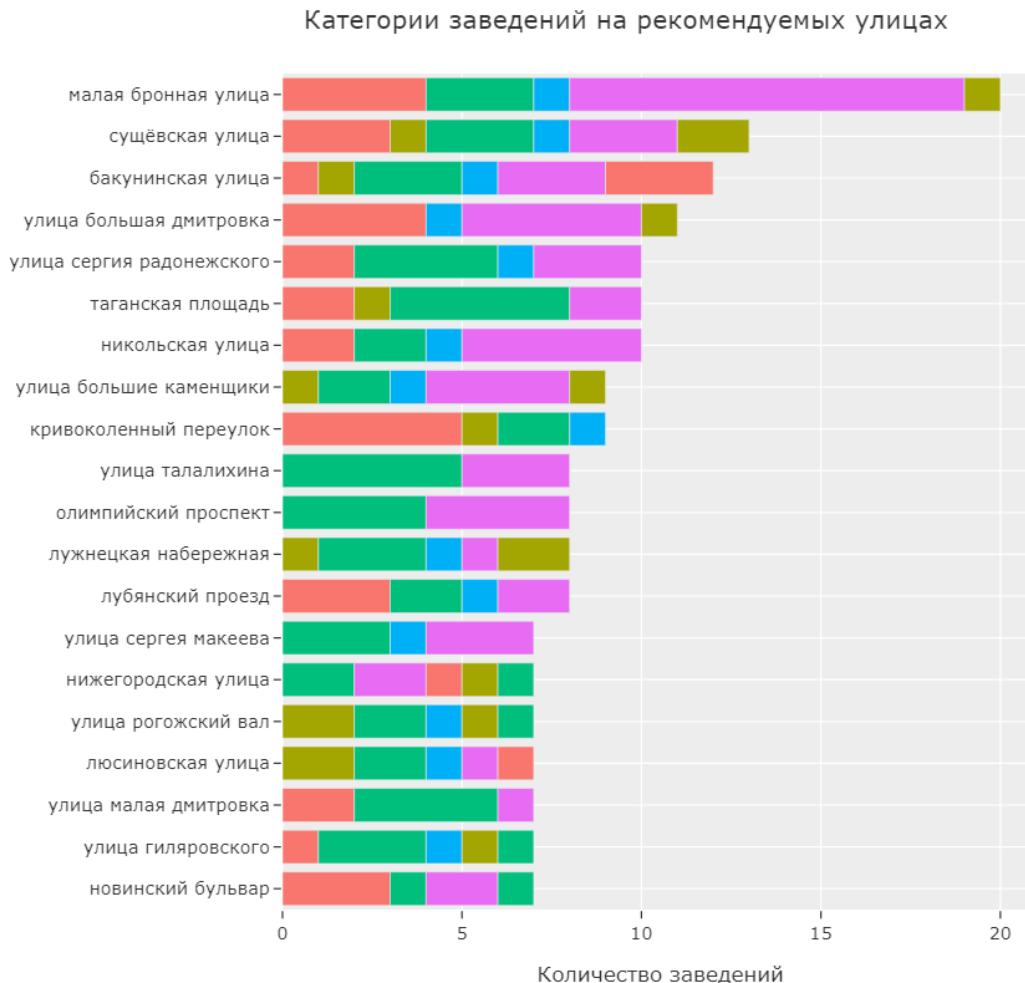
Собрали Топ-20 улиц ЦАО по числу заведений, при этом не более одной кофейни. Рассмотрим эти улицы детальнее.

```
In [98]: top_cao_street_without_cafe_count_list = top_cao_street_without_cafe_count['street'].tolist()
recommend_cao_street = cao_without_cafes.query('street in @top_cao_street_without_cafe_count_list')
recommend_cao_street_grouped_cat = recommend_cao_street.groupby(['street', 'category'])['name'].count()

fig = px.bar(recommend_cao_street_grouped_cat,
              y='street',
              x='name',
              color='category')

fig.update_layout(title='Категории заведений на рекомендуемых улицах',
                  xaxis_title='Количество заведений',
                  yaxis_title='',
```

```
yaxis={'categoryorder':'total ascending'},
height=700, width=900)
fig.show()
```



Чаще всего на этих улицах расположены кафе и рестораны. С небольшой настороженностью стоит отнестись к улицам с засильем баров, либо изучить эти бары поближе - комфортно ли такое соседство для нашей кофейни?

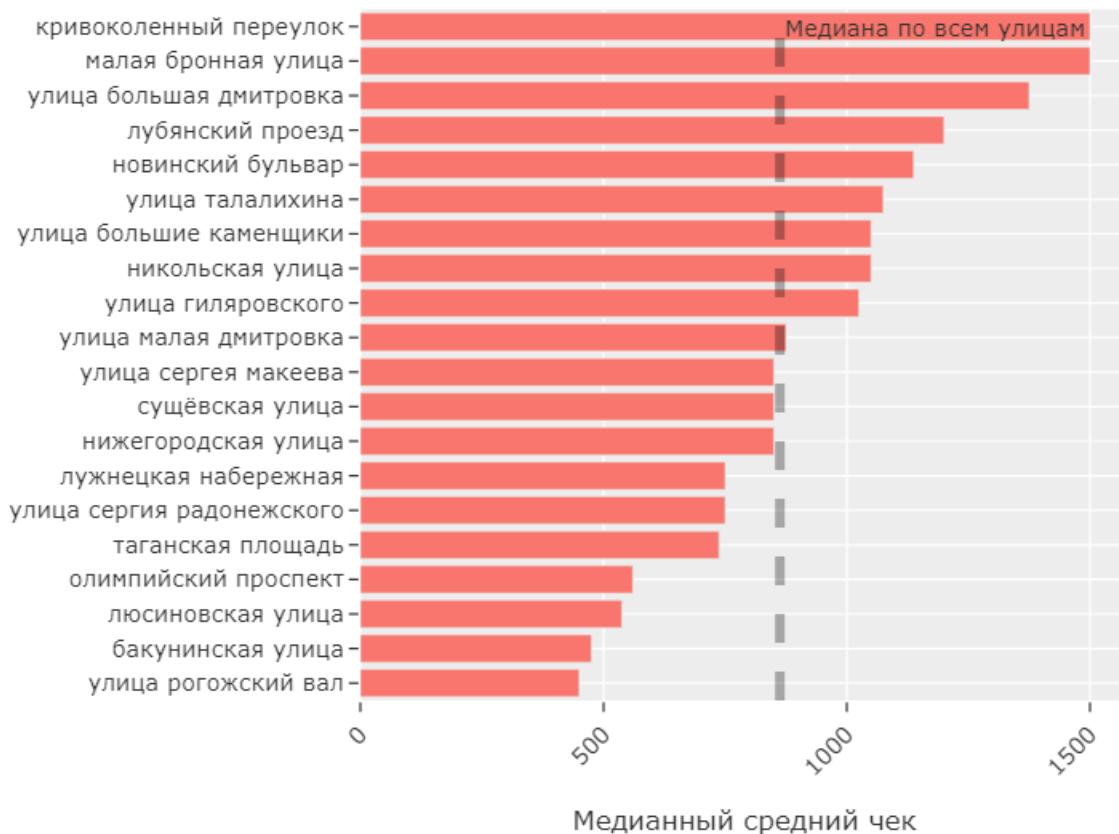
Посмотрим на средние рейтинги заведений по улицам из Топ-20, а также на медианный средний чек

```
In [99]: recommend_cao_street_median = recommend_cao_street.groupby('street')\
    .agg({'rating':'mean', 'middle_avg_bill':'median'}).reset_index()
```

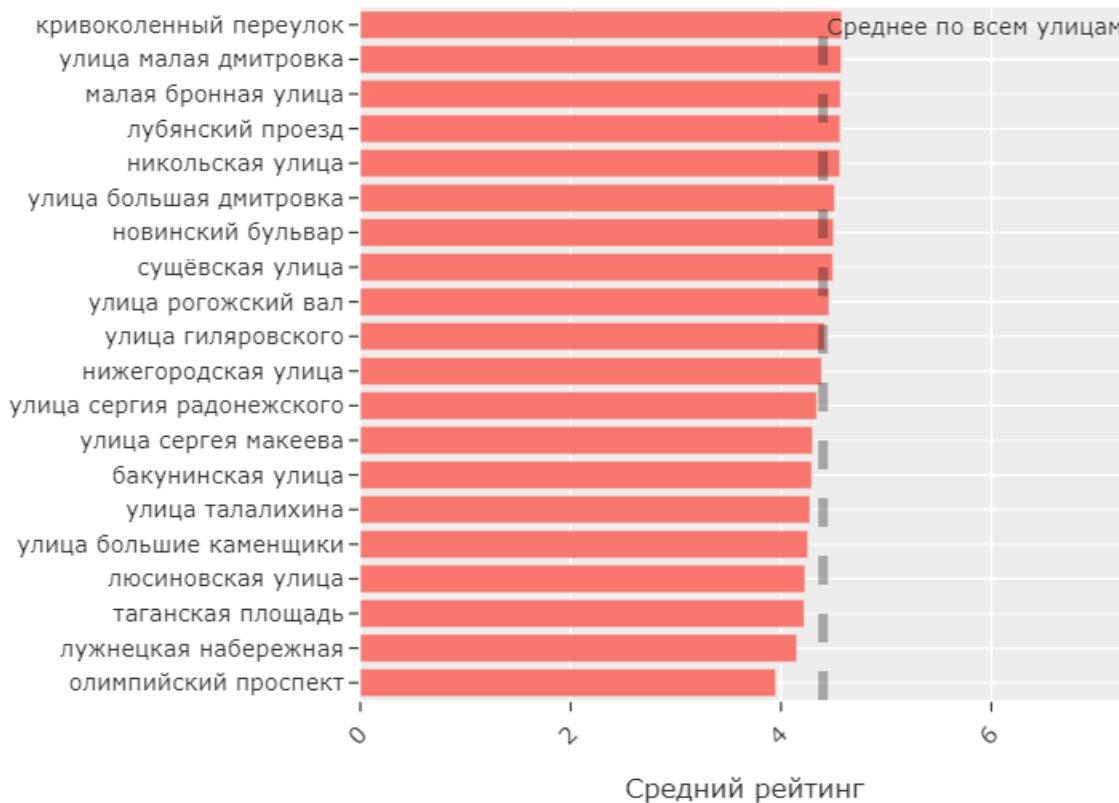
```
In [100...]: fig = px.bar(recommend_cao_street_median.sort_values(by='middle_avg_bill'), y='street', x='middle_avg_bill')
fig.update_layout(xaxis_tickangle=-45, title='Топ-20 улиц ЦАО без кофеен: средний чек', title_x=50)
fig.update_yaxes(title_text = "")
fig.update_xaxes(title_text = "Медианный средний чек")
fig.add_vline(x=recommend_cao_street_median['middle_avg_bill'].median(), line_width=5, line_dash="dashdot", annotation_text="Медиана по всем улицам")
fig.show()

fig = px.bar(recommend_cao_street_median.sort_values(by='rating'), y='street', x='rating')
fig.update_layout(xaxis_tickangle=-45, title='Топ-20 улиц ЦАО без кофеен: средний рейтинг', title_x=50)
fig.update_yaxes(title_text = "")
fig.update_xaxes(title_text = "Средний рейтинг")
fig.add_vline(x=recommend_cao_street_median['rating'].median(), line_width=5, line_dash="dashdot", annotation_text="Среднее по всем улицам")
fig.show()
```

Топ-20 улиц ЦАО без кофеен: средний чек



Топ-20 улиц ЦАО без кофеен: средний рейтинг



10 улиц имеют средний чек выше медианного, 9 улиц имеют более высокий средний рейтинг. Ограничим наш список двумя этими условиями: отберем улицы, которые подходят хотя бы под одно из этих условий и создадим карту

In [101...]

```
cao_rating_mean = recommend_cao_street_median['rating'].mean()
cao_middle_avg_bill_medain = recommend_cao_street_median['middle_avg_bill'].median()
final_recommended_street = recommend_cao_street_median.query('rating > @cao_rating_mean \\\n& middle_avg_bill > @cao_middle_avg_bill_medain')
```

```
final_recommmed_street = final_recommmed_street.round({'rating':2})
final_recommmed_street['middle_avg_bill'] = final_recommmed_street['middle_avg_bill'].astype('i')
final_recommmed_street_list = final_recommmed_street['street'].tolist()
```

In [102...]

```
coordinates_group = data.query('street in @final_recommmed_street_list')
coordinates_group_2=coordinates_group.groupby(['street', 'lat', 'lng'])['name'].count().reset_index()
coordinates = coordinates_group_2.groupby('street').agg({'lat':'first', 'lng':'first'}).reset_index()
```

In [103...]

```
lat = coordinates['lat']
lng = coordinates['lng']
street = coordinates['street']

m = folium.Map(location=[moscow_lat, moscow_lng], zoom_start=10)

for lat, lng, street in zip(lat, lng, street):
    folium.Marker(location=[lat, lng], popup=street).add_to(m)
```

Выводы, рекомендации по открытию кофейни

Общие рекомендации по открытию кофейни:

- **Круглосуточный режим работы.** Большинство несетевых заведений не работают в таком формате, скорее всего, это оправдано экономически, стоит перенять их опыт
- Стоит разграничивать ЦАО и остальные округи - они сильно отличаются по числу заведений и по средним ценам.
 - **ориентиры при открытии в ЦАО**
 - средняя стоимость чашки кофе: 140-250 руб.
 - средний чек: 350-1130 (медиана 500)
 - число посадочных мест: 40-140
 - **ориентиры при открытии в других округах**
 - средняя стоимость чашки кофе: 120-210 руб.
 - средний чек: 250-650 (медиана 350)
 - число посадочных мест: 35-120
- **Конкуренты.** 51% кофеен - сетевые. Стоит обратить внимание на топ-10 сетей кофеен, которые занимают 29% всего рынка; не располагаться в непосредственной близи от заведений этой сети. Дополнительно проанализировать кофейни с отличным рейтингом, выявить их преимущества

Почему стоит ориентироваться на ЦАО?

- самый популярный округ: москвичи чаще проводят досуг именно в центре, также, разумеется, болеещий, относительно остальных округов, поток туристов
- самый высокий средний рейтинг как у всех заведений, так и конкретно у кофеен
- самый высокий средний чек - в зависимости от фин.плана, можно установить цены на том же уровне, что поможет покрыть издержки, либо установить цены ниже, чем у конкурентов, и привлечь посетителей

Лучше выбрать улицу, на которой **уже много заведений(улица популярна), но при этом прямых конкурентов(кофеен) - не больше одного**. На карте маркерами отмечены такие улицы, помимо описанных выше условий, у выбранных улиц выше остальных средний чек и/или рейтинг.

In [104...]

```
m
```

Out[104]:



Помня о том, что на этих улицах расположено достаточно большое количество заведений, можем столкнуться с банальным отсутствием незанятых помещений. **Поэтому рекомендуется использовать эти координаты как ориентир для выбора расположения новой кофейни.** Можно рассмотреть, например, примыкающие и ближайшие к ним улицы

Презентация

Ссылка на презентацию: <https://disk.yandex.ru/i/MFAv6fcbOcH3VQ>