Рынок заведений общественного питания Москвы

`Презентация:` https://drive.google.com/drive/folders/1PHVE4oazK2yAu-wkLdGttVnp4IoOp8Xh

`Цель проекта:` подготовить исследование рынка Москвы, найти интересные особенности и презентовать полученные результаты.

`Задачи проекта`

1)исследование рынка общественного питания

2)подготовка презентации

`Описание данных` Исследование будет проводиться основе данных сервисов Яндекс Карты и Яндекс Бизнес на лето 2022 года.

Проект будет состоять из четырех частей:

1)предобработка данных

2)анализ данных

3)ответ на вопрос, возможно-ли открытие кофейни?

4)подготовка презентации

Подготовка данных

Загрузим необходимые библиотеки

```
import pandas as pd
import numpy as np
import math as mth
import matplotlib.pyplot as plt
from datetime import datetime
from scipy import stats as st
from plotly import graph_objects as go
from folium import Marker, Map, Choropleth
from folium.plugins import MarkerCluster
from matplotlib import transforms

import json

df = pd.read_csv('/datasets/moscow_places.csv')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8406 entries, 0 to 8405
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   name               8406 non-null   object 
 1   category           8406 non-null   object 
 2   address            8406 non-null   object 
 3   district           8406 non-null   object 
 4   hours              7870 non-null   object 
 5   lat                8406 non-null   float64
 6   lng                8406 non-null   float64
 7   rating             8406 non-null   float64
 8   price              3315 non-null   object 
 9   avg_bill           3816 non-null   object 
 10  middle_avg_bill    3149 non-null   float64
 11  middle_coffee_cup  535 non-null    float64
 12  chain              8406 non-null   int64  
 13  seats              4795 non-null   float64
dtypes: float64(6), int64(1), object(7)
memory usage: 919.5+ KB
```

```
df['hours'].head()
```

```
0                                ежедневно, 10:00–22:00
1                                ежедневно, 10:00–22:00
2    пн-чт 11:00–02:00; пт,сб 11:00–05:00; вс 11:00...
3                                ежедневно, 09:00–22:00
4                                ежедневно, 10:00–22:00
Name: hours, dtype: object
```

```
df['avg_bill'].head()
```

```
0                               NaN
1         Средний счёт:1500–1600 ₽
2           Средний счёт:от 1000 ₽
3    Цена чашки капучино:155–185 ₽
4           Средний счёт:400–600 ₽
Name: avg_bill, dtype: object
```

Всего представленно 8406 заведений

часы работы заведений выведены в виде обьекта (вначале дни работы, потом часы)

Цена также выводится строкой (среднее-выше среднего)

Средний счет зависит от типа заведения(кофейня, бар, ресторан), чашка кофе - кружка пива - общий счет соответственно

Остальные типы столбцов не вызывают вопросов

Выполните предобработку данных

Проверим данные на наличие пропусков и дубликатов

```
print(pd.isnull(df).sum())
```

```
name                    0
category                0
address                 0
district                0
hours                 536
lat                     0
lng                     0
rating                  0
price                5091
avg_bill             4590
middle_avg_bill      5257
middle_coffee_cup    7871
chain                   0
seats                3611
dtype: int64
```

```
print('всего дубликатов:', df.duplicated().sum())
print('доля дубликатов:',
df.duplicated().sum()/df.duplicated().count())
```

```
всего дубликатов: 0
доля дубликатов: 0.0
```

Проверим неявные дубликаты

```
df_columns = df.columns

for elem in df_columns:
    print(elem, abs((df.duplicated(subset=[elem]).sum())-8600))
```

```
name 5808
category 202
address 5947
district 203
hours 1502
lat 8403
lng 8452
rating 235
price 199
avg_bill 1092
middle_avg_bill 425
middle_coffee_cup 291
chain 196
seats 424
```

```
df = df.loc[df['name'] != 'Кафе']
```

Не до конца понимаю, что делать с неявными дубликатами. Например кол-во мест, там ведь цифры, они могут повторяться.

Пример про "данные на глазок" как их побороть, их же так не найти, разве нет?

Пока следует оставить пропуски как есть, во время исследования их можно будет удалить или заполнить.

Дубликаты отсутствуют

С помощью функции создадим столбец с улицами заведений

```
def find_street(df):
    for elem in df['address'].split(','):
           if elem != 'Москва':
                 if len(elem)>7:
                        return elem.replace(',', '')

df['street'] = df.apply(find_street, axis=1)

len(df.loc[pd.isnull(df['street'])])
```

```
1
```

С помощью функции создадим с обозначением, что заведение работает ежедневно и круглосуточно

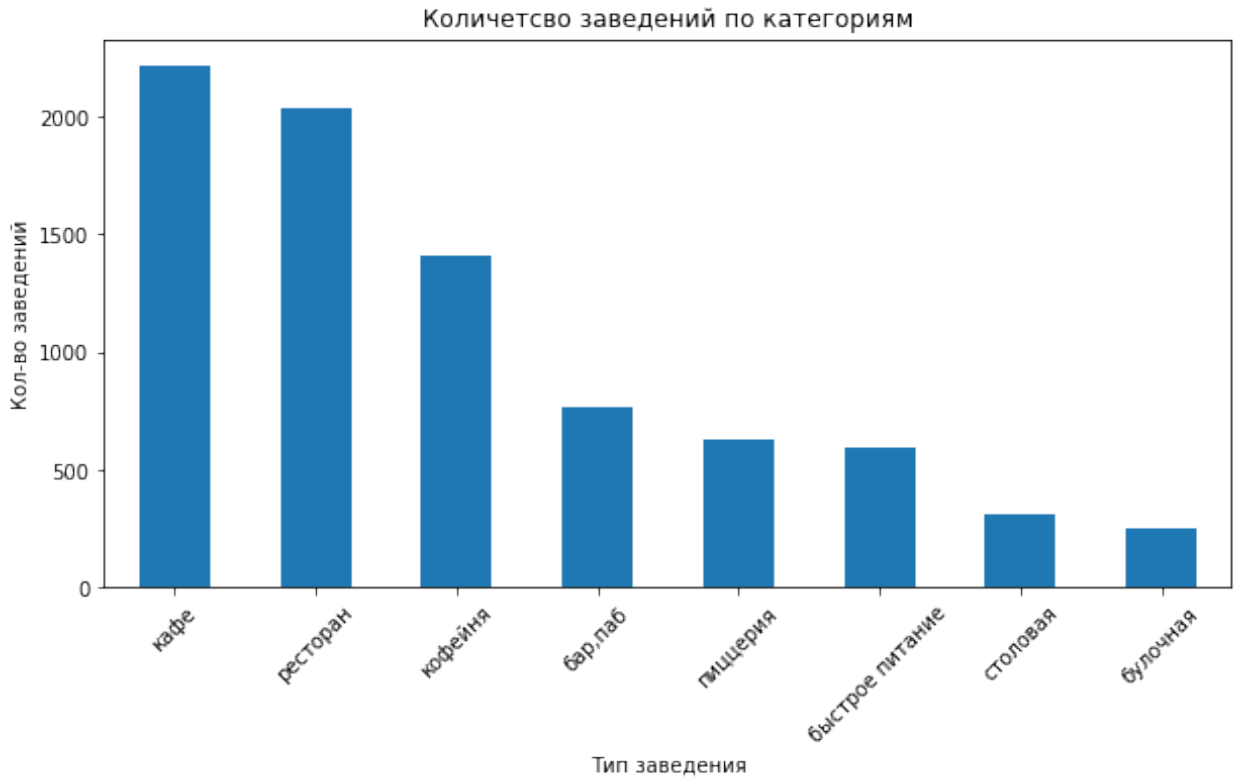
```
df['hours'] = df['hours'].apply(str)

def work_time (df):
    x = 'ежедневно'
    y = 'круглосуточно'
    if x and y in df:
        return True
    else:
        return False

df['is_24/7'] = df['hours'].apply(work_time)
```

Анализ данных

Рассмотрим количетсво заведений по категориям

```
plt.figure(figsize=(10,5))
categ_place = df.pivot_table(index='category',
aggfunc='count').sort_values(by='address', ascending=False)
categ_place['address'].plot(x='address', kind="bar")
```

```
plt.title('Количетсво заведений по категориям')
plt.xticks(rotation=45)
plt.xlabel('Тип заведения')
plt.ylabel('Кол-во заведений')
plt.show()
```



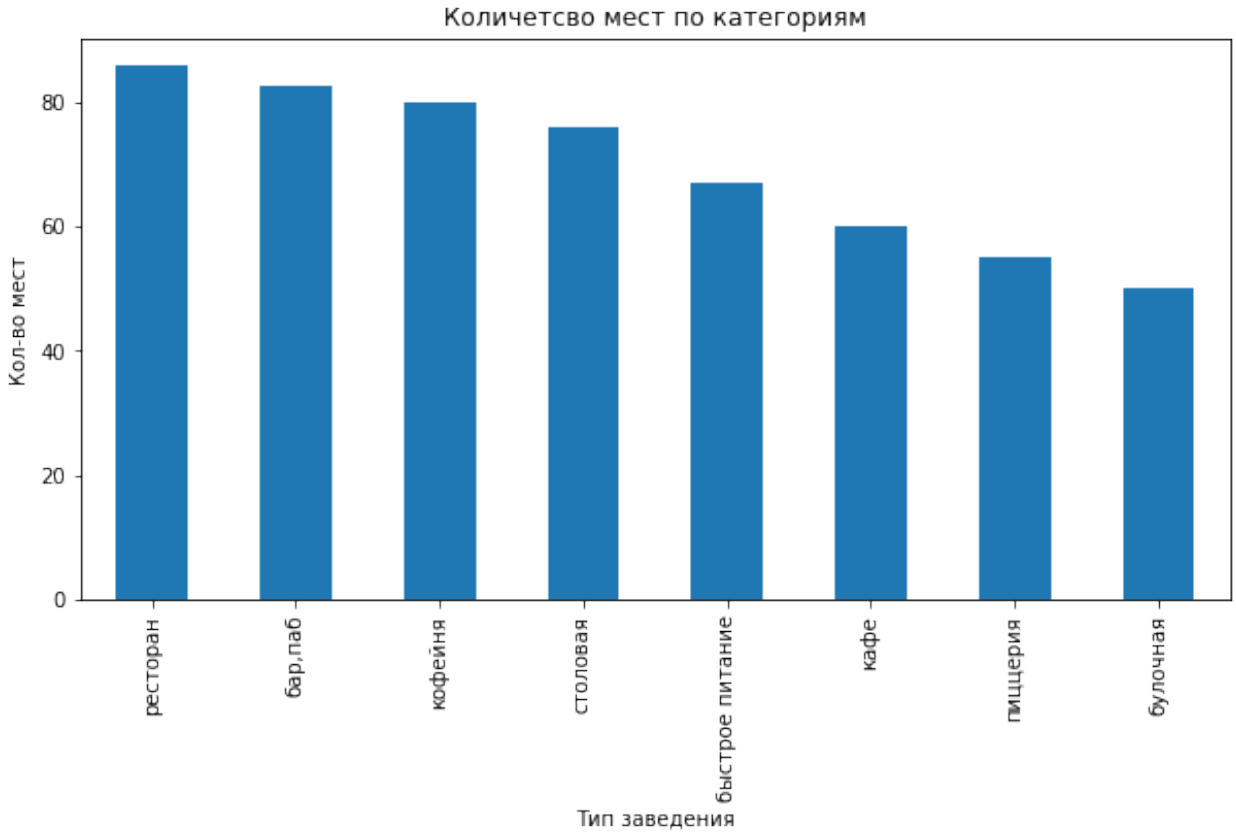

```
df.groupby(['category'])['address'].count()/8406*100
```

```
category
бар,паб              9.076850
булочная             3.045444
быстрое питание      7.090174
кафе                26.397811
кофейня             16.738044
пиццерия             7.518439
ресторан            24.208898
столовая             3.675946
Name: address, dtype: float64
```

Больше всего заведений из категории кафе-ресторан(52%), меньше всего столовых и булочных (6%)

Сколько посадочных мест в заведениях, по категории

```
plt.figure(figsize=(10,5))
seats_place = df.pivot_table(index='category',
aggfunc='median').sort_values(by='seats', ascending=False)
seats_place['seats'].plot(x='seats', kind="bar")
plt.title('Количетсво мест по категориям')
plt.xlabel('Тип заведения')
plt.ylabel('Кол-во мест')
plt.show()
```



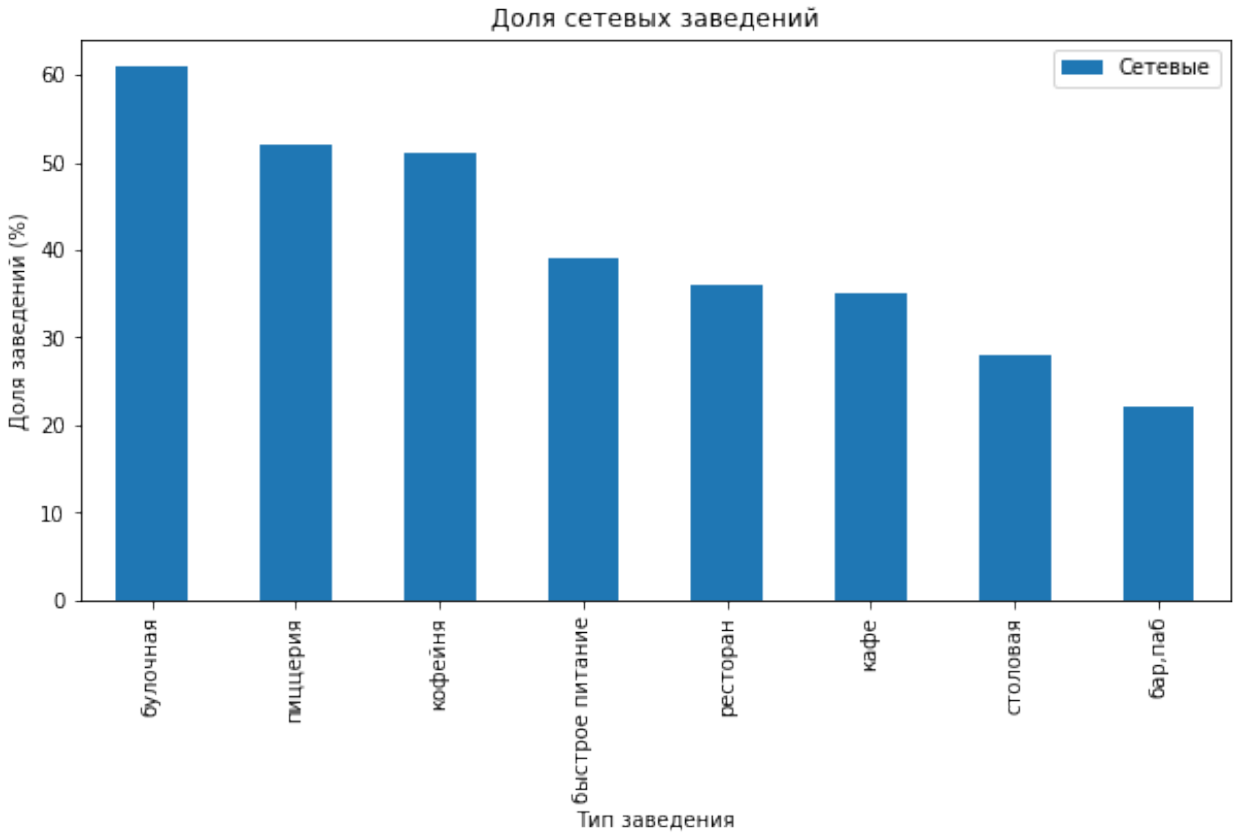
Больше всего мест в заведениях, подразумевающих долгое посещение(ресторан-бар), заведения где можно забрать заказ с собой(пиццерия, булочная, быстрое питание), имеют меньше мест.

```
print('Сетевых заведений:', df['chain'].sum())
print('Не сетевых заведений:', df['chain'].count() -
df['chain'].sum())
```

```
Сетевых заведений: 3205
Не сетевых заведений: 5012
```

```
plt.figure(figsize=(10,5))
chain_first = df.pivot_table(index='category', aggfunc='sum')
chain_second = df.pivot_table(index='category', aggfunc='count')
chain_second['total_place'] = chain_second['address']
```

```
chain_unite =
pd.concat([chain_first['chain'],chain_second['total_place']],
axis=1).sort_values(by='chain')
chain_unite['part'] =
round(chain_unite['chain']/chain_unite['total_place']*100).astype(int)
chain_unite['part'].sort_values(ascending=False).plot(kind='bar')
plt.legend(["Сетевые"])
plt.title('Доля сетевых заведений')
plt.xlabel('Тип заведения')
plt.ylabel('Доля заведений (%)')
plt.show()
```



Больше всего сетевых заведений среди булочных и пиццерий. Уникальные заведения представлены барами и столовыми.

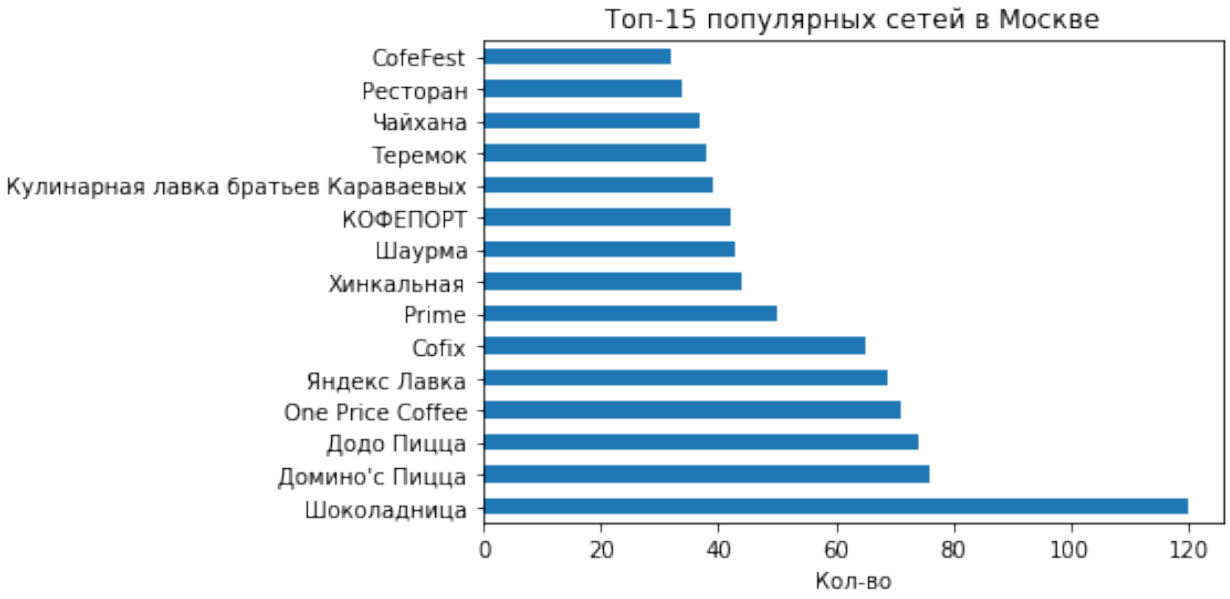
Рассмотрим самые популярные заведения

```
top_chain = df.groupby(['name','category'])
['name'].count().sort_values(ascending=False).head(15)
top_chain
```

name	category	
Шоколадница	кофейня	119
Домино'с Пицца	пиццерия	76

```
Додо Пицца                                  пиццерия            74
One Price Coffee                            кофейня             71
Яндекс Лавка                                ресторан            69
Cofix                                       кофейня             65
Prime                                       ресторан            49
КОФЕПОРТ                                    кофейня             42
Кулинарная лавка братьев Караваевых         кафе                39
Теремок                                     ресторан            36
Ресторан                                    ресторан            33
Шаурма                                      быстрое питание     32
CofeFest                                    кофейня             31
Чайхана                                     кафе                26
Буханка                                     булочная            25
Name: name, dtype: int64
```

```
df.groupby('name')
['name'].count().sort_values(ascending=False).head(15).plot(kind='barh
')
plt.title('Топ-15 популярных сетей в Москве')
plt.ylabel('')
plt.xlabel('Кол-во')
plt.show()
```



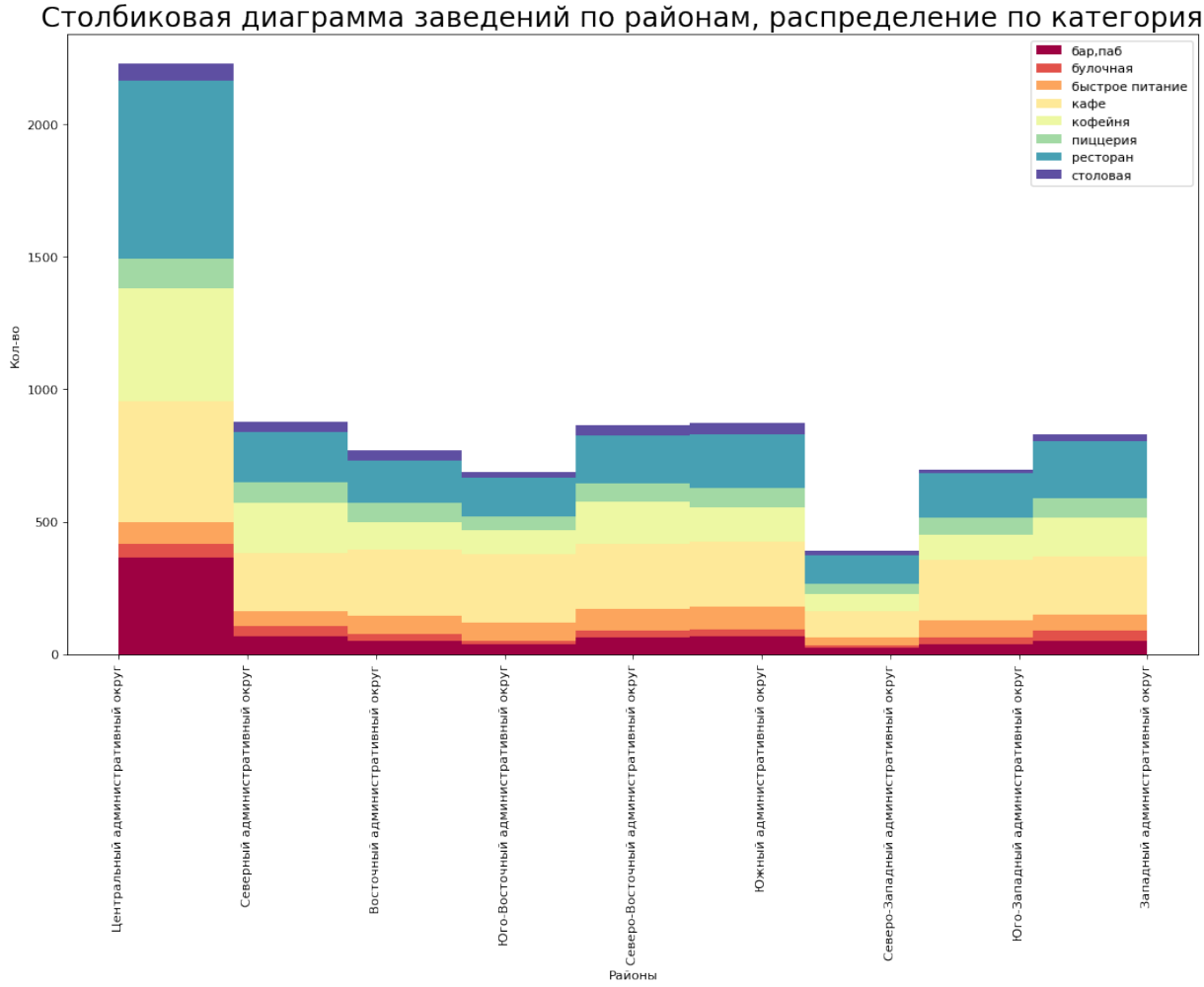
Скорее всего Кафе это дубликат, удалим его

Чаще всего это сетевые заведения (кафе, пиццерии, рестораны)

Какие административные районы Москвы присутствуют в датасете?

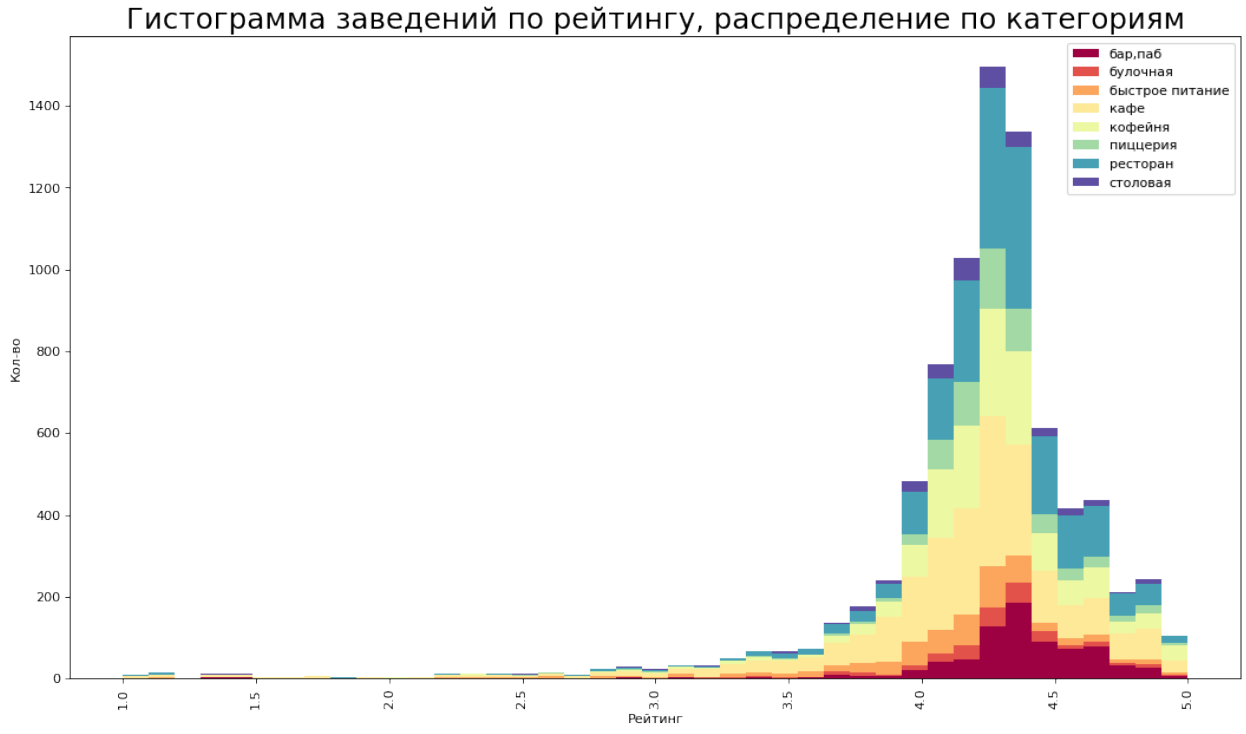
```
x_var = 'district'
groupby_var = 'category'
```

```
df_agg = df.loc[:, [x_var, groupby_var]].sort_values(by='category',
ascending=True).groupby(groupby_var)
vals = [df[x_var].values.tolist() for i, df in df_agg]

plt.figure(figsize=(16,9), dpi= 80)
colors = [plt.cm.Spectral(i/float(len(vals)-1)) for i in
range(len(vals))]
n, bins, patches = plt.hist(vals, df[x_var].unique().__len__(),
stacked=True, density=False, color=colors[:len(vals)])


plt.legend({group:col for group, col in
zip(np.unique(df[groupby_var]).tolist(), colors[:len(vals)])})
plt.title('Столбиковая диаграмма заведений по районам, распределение
по категориям', fontsize=22)
plt.xlabel('Районы')
plt.xticks(rotation=90)
plt.ylabel("Кол-во")
plt.show()
```



Распределение средних рейтингов по категориям заведений

```
# Prepare data
x_var = 'rating'
groupby_var = 'category'
df_agg = df.loc[:, [x_var, groupby_var]].groupby(groupby_var)
vals = [df[x_var].values.tolist() for i, df in df_agg]

# Draw
plt.figure(figsize=(16,9), dpi= 80)
colors = [plt.cm.Spectral(i/float(len(vals)-1)) for i in
range(len(vals))]
n, bins, patches = plt.hist(vals, df[x_var].unique().__len__(),
stacked=True, density=False, color=colors[:len(vals)])

# Decoration
plt.legend({group:col for group, col in
zip(np.unique(df[groupby_var]).tolist(), colors[:len(vals)])})
plt.title('Гистограмма заведений по рейтингу, распределение по
категориям', fontsize=22)
plt.xlabel('Рейтинг')
plt.xticks(rotation=90)
plt.ylabel("Кол-во")
plt.show()
```



Распределение рейтингов по типам заведений практически идентично

Фоновая картограмма (хороплет) со средним рейтингом заведений каждого района

```
rating_df = df.groupby('district', as_index=False)
['rating'].agg('mean')

with open('/datasets/admin_level_geomap.geojson', 'r') as f:
    geo_json = json.load(f)

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

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


# пишем функцию, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер
marker_cluster
Choropleth(
    geo_data=state_geo,
    data=rating_df,
    columns=['district', 'rating'],
    key_on='feature.name',
    fill_color='YlGn',
    fill_opacity=0.8,
    legend_name='Средний рейтинг заведений по районам',
).add_to(m)

# применяем функцию create_clusters() к каждой строке датафрейма

# выводим карту
m
```

```
<folium.folium.Map at 0x7f449503d490>
```

Все заведения датасета на карте с помощью кластеров

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

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


# пишем функцию, которая принимает строку датафрейма,
```

```
# создаёт маркер в текущей точке и добавляет его в кластер
marker_cluster
def create_clusters(row):
    Marker(
        [row['lat'], row['lng']],
        popup=f"{row['name']} {row['rating']}",
    ).add_to(marker_cluster)

# применяем функцию create_clusters() к каждой строке датафрейма
df.apply(create_clusters, axis=1)

# выводим карту
m
```

```
<folium.folium.Map at 0x7f4494f3c4f0>
```

Топ-15 улиц по количеству заведений

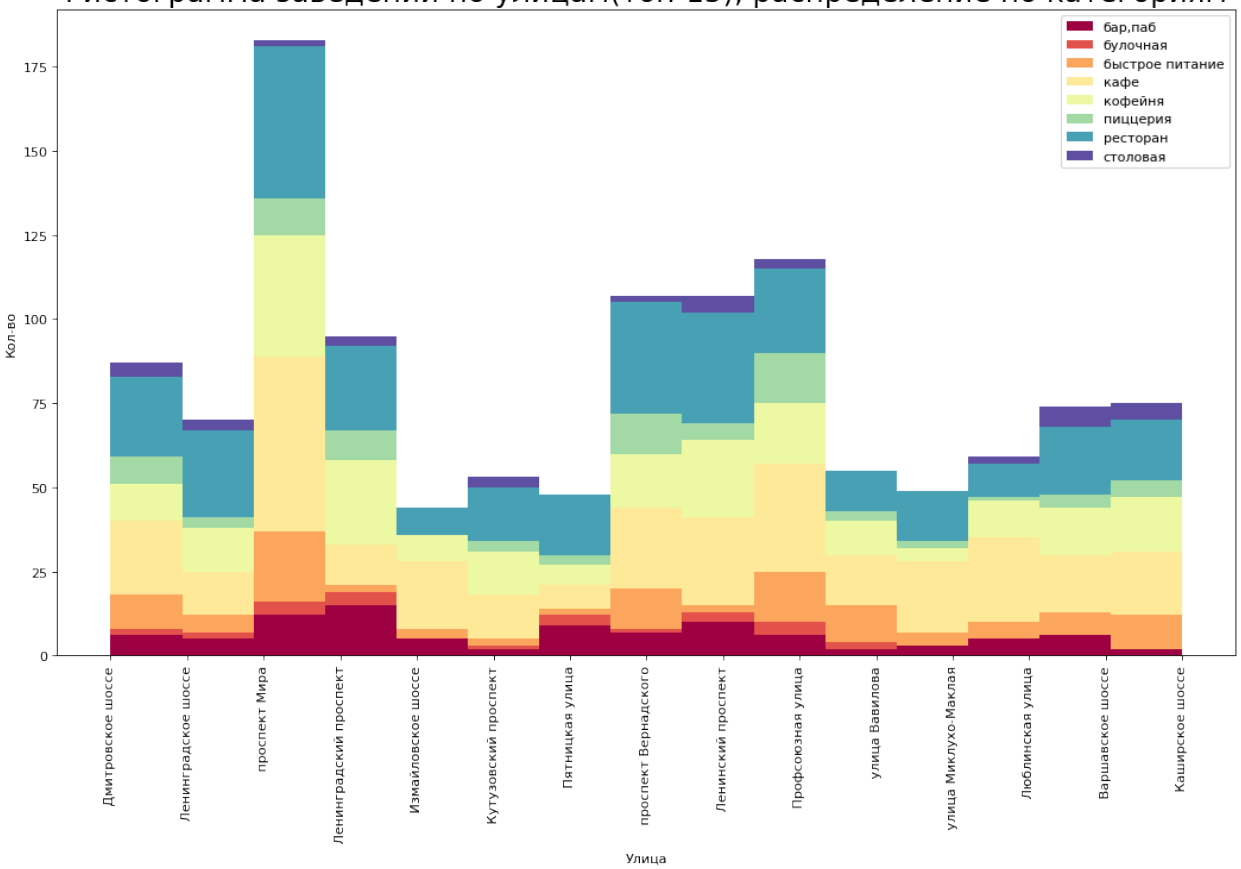
```
exemp = df.pivot_table(index='street',values='address',
aggfunc='count')
exemp = exemp.sort_values(by=['address'], ascending=True).tail(15)
top_streets = []
for elem in exemp.T:
    top_streets.append(elem)

exemp
```

street	address
Измайловское шоссе	44
Пятницкая улица	48
улица Миклухо-Маклая	49
Кутузовский проспект	53
улица Вавилова	55
Люблинская улица	59
Ленинградское шоссе	70
Варшавское шоссе	74
Каширское шоссе	75
Дмитровское шоссе	87
Ленинградский проспект	95
Ленинский проспект	107
проспект Вернадского	107
Профсоюзная улица	118
проспект Мира	183

```
def ret(df):
    if df in top_streets:
        return True
    else:
        return False
```

```
df['top_street'] = df['street'].apply(ret)
exemp = df.loc[df['top_street'] == 1]

# Prepare data
x_var = 'street'
groupby_var = 'category'
exemp_agg = exemp.loc[:, [x_var, groupby_var]].groupby(groupby_var)
vals = [exemp[x_var].values.tolist() for i, exemp in exemp_agg]

# Draw
plt.figure(figsize=(16,9), dpi= 80)
colors = [plt.cm.Spectral(i/float(len(vals)-1)) for i in
range(len(vals))]
n, bins, patches = plt.hist(vals, exemp[x_var].unique().__len__(),
stacked=True, density=False, color=colors[:len(vals)])

# Decoration
plt.legend({group:col for group, col in
zip(np.unique(exemp[groupby_var]).tolist(), colors[:len(vals)])})
plt.title('Гистограмма заведений по улицам(топ-15), распределение по
категориям', fontsize=22)
plt.xlabel('Улица')
plt.xticks(rotation=90)
plt.ylabel("Кол-во")
plt.show()
```

Гистограмма заведений по улицам(топ-15), распределение по категориям



Улицы с одним обьектом общепита

```
exemp = df.pivot_table(index='street',values='address',
aggfunc='count')
exemp = exemp.sort_values(by=['address'], ascending=True).head(15)
bad_streets = []
for elem in exemp.T:
    bad_streets.append(elem)
def ret(df):
    if df in bad_streets:
        return True
    else:
        return False
df['bad_streets'] = df['street'].apply(ret)
exemp = df.loc[df['bad_streets'] == 1]

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

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

```
marker_cluster = MarkerCluster().add_to(m)

# пишем функцию, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер
marker_cluster
def create_clusters(row):
    Marker(
        [row['lat'], row['lng']],
        popup=f"{row['name']} {row['rating']}",
    ).add_to(marker_cluster)

# применяем функцию create_clusters() к каждой строке датафрейма
exemp.apply(create_clusters, axis=1)

# выводим карту
m
```

```
<folium.folium.Map at 0x7f448d67e580>
```

Достаточно маленькие места, слаобо связанные между собой.

```
bill_df = df.groupby('district', as_index=False)
['middle_avg_bill'].agg('median')

with open('/datasets/admin_level_geomap.geojson', 'r') as f:
    geo_json = json.load(f)

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

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


# пишем функцию, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер
marker_cluster
Choropleth(
    geo_data=state_geo,
    data=bill_df,
    columns=['district', 'middle_avg_bill'],
    key_on='feature.name',
    fill_color='YlGn',
    fill_opacity=0.8,
    legend_name='Медианный счет заведений по районам',
).add_to(m)
```

```
# применяем функцию create_clusters() к каждой строке датафрейма

# выводим карту
m
```

```
<folium.folium.Map at 0x7f448d4d52e0>
```

Удаленность от центра на прямую влияет на цену, ближе к центру - больше цены и наоборот. Исключением можно считать западный административный округ

Больше всего заведений из категории кафе-ресторан, меньше всего столовых и булочных.

Больше мест в заведениях, подразумевающих долгое посещение(ресторан-кафе), заведения, где можно забрать заказ с собой(пиццерия, булочная, быстрое питание) + бары(можно постоять за стойкой) имеют меньше мест.

Не сетевых заведений больше, чаще всего сетевыми являются кофейни, пиццерии и булочные.

Распределение рейтингов по типам заведений практически идентично

Детализируем исследование: открытие кофейни

```
cafe = df.loc[df['category'] == 'кофейня']
print('всего кофеен:', cafe['address'].count())
```

```
всего кофеен: 1407
```

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

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


# пишем функцию, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер marker_cluster
def create_clusters(row):
    Marker(
        [row['lat'], row['lng']],
        popup=f"{row['name']} {row['rating']}",
    ).add_to(marker_cluster)
```

```
# применяем функцию create_clusters() к каждой строке датафрейма
cafe.apply(create_clusters, axis=1)

# выводим карту
m
```

```
<folium.folium.Map at 0x7f4494f52340>
```

```
cafe.groupby('district')
['district'].count().sort_values(ascending=False)
```

```
district
Центральный административный округ          428
Северный административный округ             191
Северо-Восточный административный округ     158
Западный административный округ              149
Южный административный округ                 131
Восточный административный округ             105
Юго-Западный административный округ           95
Юго-Восточный административный округ          89
Северо-Западный административный округ        61
Name: district, dtype: int64
```

Больше всего кофеен в центре, на Севере и Северо-Востоке. Меньше всего на Юге и Северо-Западе

```
 print('Всего кофеен 24/7:',len(df.loc[df['is_24/7'] == True]))
```

```
Всего кофеен 24/7: 753
```

Какие у кофеен рейтинги? Как они распределяются по районам?

```
rating_cafe = cafe.groupby('district', as_index=False)
['rating'].agg('mean')

with open('/datasets/admin_level_geomap.geojson', 'r') as f:
    geo_json = json.load(f)

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

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


# пишем функцию, которая принимает строку датафрейма,
```

```
# создаёт маркер в текущей точке и добавляет его в кластер
marker_cluster
Choropleth(
    geo_data=state_geo,
    data=rating_cafe,
    columns=['district', 'rating'],
    key_on='feature.name',
    fill_color='YlGn',
    fill_opacity=0.8,
    legend_name='Медианный рейтинг кафе по районам',
).add_to(m)

# применяем функцию create_clusters() к каждой строке датафрейма

# выводим карту
m
```

```
<folium.folium.Map at 0x7f448d4daca0>
```

Разница практически отсутствует, медианный рейтинг чуть меньше в Западном адм округе.

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

```
cud_df = df.groupby('district', as_index=False)
['middle_coffee_cup'].agg('median')

with open('/datasets/admin_level_geomap.geojson', 'r') as f:
    geo_json = json.load(f)

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

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


# пишем функцию, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер
marker_cluster
Choropleth(
    geo_data=state_geo,
    data=cud_df,
    columns=['district', 'middle_coffee_cup'],
    key_on='feature.name',
    fill_color='YlGn',
    fill_opacity=0.8,
```

```
    legend_name='Медианная кружка кофе заведений по районам',
).add_to(m)

# применяем функцию create_clusters() к каждой строке датафрейма

# выводим карту
m
```

```
<folium.folium.Map at 0x7f449525b820>
```

Средняя чашка дороже всего в центре и в Западном административном округе.

Наиболее важным параметром при выборе расположения кофейни будет будет являться цена средний чашки кофе и цена аренды, зависящая от расположения. Можно предположить, что цена аренда зависит от приблеженности к центру. Колличество кофеен в центре достаточно велико, в тоже время центр не является единственным районом с дорогой средней кружкой, стоит присмотреться к Западному административному округу. Медианный рейтинг заведений в этом районе, хоть и незначительно, но отличается от остальных, он хуже. Если рассмотреть карту можно найти ряд мест в районе, приблеженных к центру, имеющих высокую цену за среднюю кружку. Таким образом, следует рассмотреть возможность открытия кофейни с высоким рейтингом в Западном административном округе, с целью совместить меньшую арендную плату чем в центре, и высокую цену средней кружки кофе.