Государственное бюджетное профессиональное
образовательное учреждение Московской области
«Физико-технический колледж»

Отчёт по кейсу «Самолёт»:

Работу выполнил:
Студент группы № ИСП-21
Кочкин Никита

Долгопрудный, 2024

Введение: В данном отчёте рассматриваются выводы, полученные с аналитической работы над данными в области “Квартиры в Москве, Новой Москве и Московской области”

Цель: Собрать данные и произвести анализ данных

Задачи:

- Используя открытые источники (Циан) составить список параметров, значительно влияющих на цену квадратного метра жилой площади.
- С учетом выявленных факторов произвести парсинг данных по квартирам на продажу, используя различные парсеры.
- Произвести подготовку данных для анализа.
- Визуализировать взаимосвязь между ними; определить признаки, оказывающие наиболее сильное влияние.

1: Мы импортируем разные библиотеки для разных целей:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import csv
import warnings
warnings.simplefilter('ignore')
import missingno as msn
```

Pandas нам нужен для анализа уже структурированных данных, то есть размещённых не хаотично, а в таблицах.

Numpy нам нужна для работы с многомерными массивами и включает набор математических функций, которые применяются над ними.

Matplotlib нам нужен для визуализации данных в Python. Она используется для создания любых видов графиков: линейных, круговых диаграмм, построчных гистограмм и других — в зависимости от задач.

Seaborn нам нужен для анализа данных и отображения сложных зависимостей с помощью графиков на языке Python.

Missingno нам нужна для поиска и визуализации пропусков в данных.

2: Мы открываем наши необработаные данные:

```
df = pd.read_csv('8K.csv')
```

3: Убираем все не нужные данные:

```
del df['accommodation_type'] # Это тип дома
del df['deal_type'] # Тип сделки, мы покупаем, а не берем в аренду
del df['residential_complex'] # Особо не надо
del df['heating_type'] # Обогрев не нужен так как данных по нему отсутсвует
del df['object_type'] # Не нужно по тем же причинам как и с обогревом
```

4: Чистим отрицательные данные

```
df_replaced = df.replace(-1,np.nan)
df_replaced = df_replaced.replace("-1",np.nan)
df_replaced = df_replaced.replace(-1.0,np.nan)
df_replaced = df_replaced.replace("-1.0",np.nan)
df_replaced.to_csv("cleaned_8K.csv", index=False)
```

5: Открываем уже чистые и хорошие данные

```
df = pd.read_csv('cleaned_8K.csv')
dash_info = pd.read_csv('dash_info_fifth.csv')
dash_info.sort_values(["price_for_meter"], axis=0, ascending=[False], inplace=True)
```

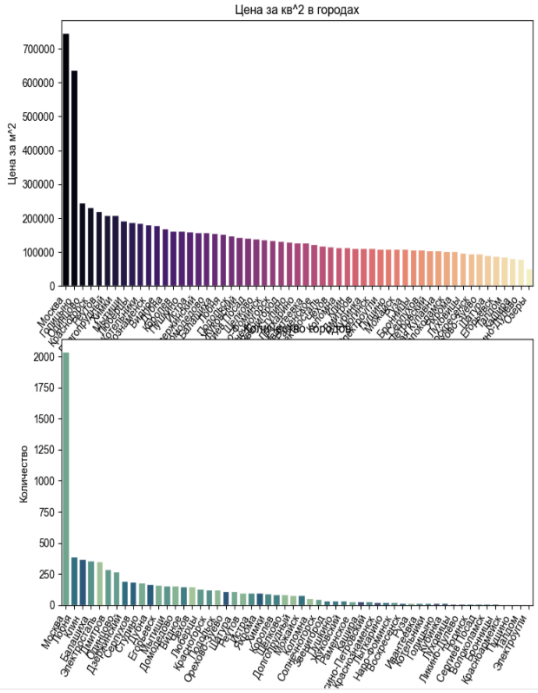
6: Как готовились данные для верхнего графика:

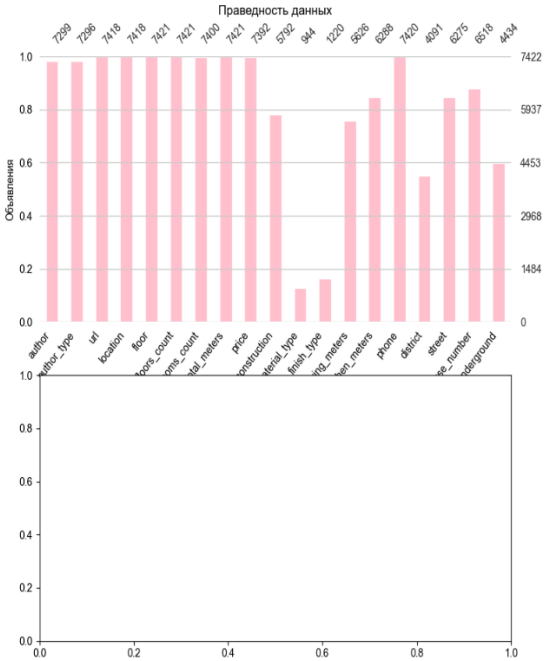
```
list_of_cities = df['location'].unique()

def price_for_meter(location):
    city = df[df['location']==location]
    city_price = city['price'].sum()

    cleaned_data = [x.replace("м²", "").strip() for x in city['total_meters']]
    cleaned_data = [x.replace("-1", "0").strip() for x in cleaned_data]
    cleaned_data = [x.replace("���", "0").strip() for x in cleaned_data]
    cleaned_data = pd.to_numeric([x.replace(",", ".").strip() for x in cleaned_data]).sum()

    # cleaned_data = city['total_meters'].sum()
    return round(city_price/cleaned_data,2)

with open("dash_info_fourth.csv", 'w', newline='', encoding='UTF-8') as csvfile:
    fieldnames = ['city', 'price_for_meter']
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    for city in list_of_cities:
        writer.writerow({'city': city, 'price_for_meter': price_for_meter(city)})
```

7: Вот так выводились графики:

```
figs, axs = plt.subplots(2,2, figsize=(30,14))

sns.set_theme(style='whitegrid', palette='dark:pink')
sns.barplot(hue='city', legend=False, x='city', y='price_for_meter', data=dash_info, ax=axs[0,0], palette='magma')
axs[0,0].set_title('Цена за кв^2 в городах')
axs[0,0].set_xticklabels(axs[0,0].get_xticklabels(), rotation=50, ha='right')
axs[0,0].set_ylabel('Цена за м^2')
axs[0,0].set_xlabel('Города')

sns.set_theme(style='whitegrid', palette='dark:pink')
sns.countplot(x='location', hue='location', legend=False, data=df, ax=axs[1,0], order=df['location'].value_counts().index, palette='crest')
axs[1,0].set_title('Количество городов')
axs[1,0].set_xticklabels(axs[1,0].get_xticklabels(), rotation=60, ha='right')
axs[1,0].set_ylabel('Количество')

colours = ['#000000', '#ff0000']
# sns.heatmap(df[df.columns].isnull(), cmap=sns.color_palette(colours), ax=axs[0,1])
msn.bar(df, ax=axs[0,1], fontsize=10, color=(1, 0.75, 0.8))
axs[0,1].set_title('Праведность данных')
axs[0,1].set_xticklabels(axs[0,1].get_xticklabels(), rotation=50, ha='right')
axs[0,1].set_ylabel('Объявления')
axs[0,1].set_xlabel('')
axs[0,1].grid(False)

plt.show()

dash = pd.read_csv('years_count.csv')

plt.subplots(figsize=(24,5))

plot = sns.barplot(x='year_of_construction', y='count', data=dash, palette='flare', orient='v')
plot.set_xticklabels(plot.get_xticklabels(), rotation=90, ha='right')
sns.despine(left=True, bottom=True)
print(".")

plt.show()
```

8: Результаты (Графики):



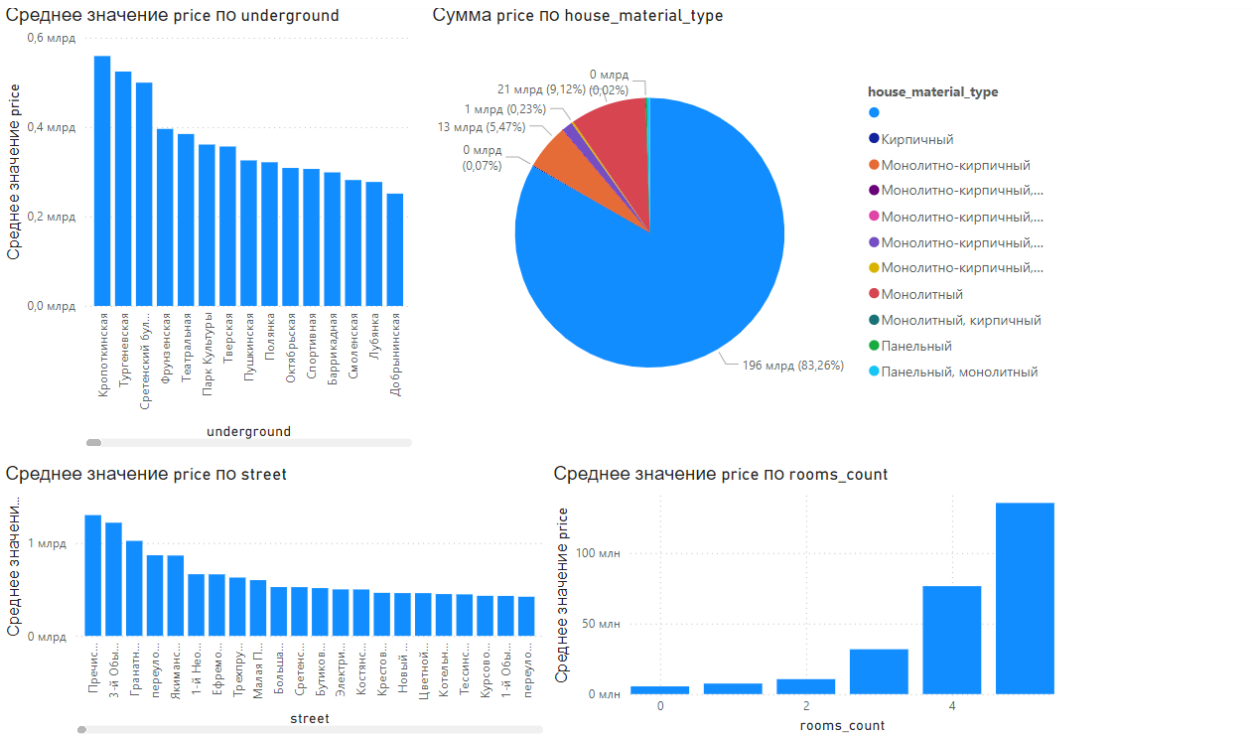


9. графики в power BI:

1) Соотношение цены по локации дома

2) Соотношение цены от материала здания

3) Соотношение цены квартир от станций метро



9: Корреляционная матрица:

```
plt.figure(figsize=(8,4))
sns.heatmap(df.corr(numeric_only=True), cmap='crest', annot=True, vmin=-1, vmax=1)
plt.xticks(rotation=30, ha="right")
plt.show()
```