Research on the Real Estate Market for Apartment Sales

- to determine market value of real estate

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:
```
data=pd.read_csv("real_estate_data.csv", sep='\t')
pd.set_option('display.max_columns', None)
```

dataset overview

In [ ]:
```
data.head(10)
```

In [ ]:
```
data.info()
```

In [ ]:
```
data.shape
```

renaming columns

In [ ]:
```
data = data.rename(columns={'studio': 'is_studio','open_plan': 'is_open_pla
data.columns
```

converting datatypes of columns

- first_day_exposition(object) publication date. Requires conversion to DateTime
- floors_total(float64) Requires conversion to int
- is_apartment(object) requires conversion to boolean
- balcony(float64) Requires conversion to int
- num_of_parks_around_3000m(float64) number of parks within a 3 km radius. Requires conversion to int
- num_of_ponds_around_3000m(float64) number of bodies of water within a 3 km radius. Requires conversion to inton to int

In [ ]:
```
data['first_day_exposition'] = pd.to_datetime(data['first_day_exposition'],
data['floors_total'] = data['floors_total'].astype('Int64')
data['is_apartment'] = data['is_apartment'].astype('bool')
data['balcony'] = data['balcony'].astype('Int64')
data['num_of_parks_around_3000m'] =  data['num_of_parks_around_3000m'].asty
data['num_of_ponds_around_3000m'] =  data['num_of_ponds_around_3000m'].asty
data.head()
```

In [ ]:
```
# adding all indexes of rows that will be deleted
idx_to_be_removed = []
```

Removing anomalies from dataset

deleting total images column

In [ ]:
```
# we dont need total_images
data.drop('total_images',axis=1,inplace=True)
```

removing extremely expensive house from data

In [ ]:
```
# checking last price column for outliers
data['last_price'].describe()
```

In [ ]:
```
last_price_anomalies = data['last_price'].loc[(data['last_price']>50000000)
last_price_anomalies_p = np.round(float((last_price_anomalies *100) /len(da
print("there are",last_price_anomalies,"houses with abnormal values which i
```

In [ ]:
```
#removing house which costs more than 50000000
idx_to_be_removed.extend(data['last_price'].loc[(data['last_price']>5000000
```

removing very cheap houses from dataset

In [ ]:
```
last_price_small_anomalies = data['last_price'].loc[(data['last_price'] <10
last_price_small_anomalies_p =np.round(float((last_price_small_anomalies *1
print("there are",last_price_small_anomalies,"houses with abnormal values w
```

In [ ]:
```
# removing houses which cost less than 1000000
idx_to_be_removed.extend(data['last_price'].loc[(data['last_price']<1000000
```

removing outliers from total floors

In [ ]:
```
data['floors_total'].describe()
```

In [ ]:
```
data.loc[:,['floors_total']].boxplot()
plt.show()
```

In [ ]:
```
floors_total_anomalies =data['floors_total'].loc[(data['floors_total'] >30)
floors_total_anomalies_p = np.round(float((floors_total_anomalies * 100) /l
print("Total records with extreme number of floors is",floors_total_anomali
```

In [ ]:
```
# Mark for deletion
idx_to_be_removed.extend(data['floors_total'].loc[(data['floors_total' ] >3
idx_to_be_removed.extend(data[data['floors_total'].isna()].index)
```

removing outliers from rooms

In [ ]:
```
data['rooms'].describe()
```

In [ ]:
```
data.loc[:,['rooms']].boxplot()
plt.show()
```

In [ ]:
```
# records with high number of rooms
rooms_anomalies = data['rooms'].loc[(data['rooms'] >8)].count()
rooms_anomalies_p = np.round(float((rooms_anomalies *100) / len(data)),
decimals =1)
print('total records with high number of rooms is',rooms_anomalies,'which i
```

In [ ]:
```
# Mark for deletion
idx_to_be_removed.extend(data['rooms'].loc[(data['rooms'] >8)].index)
```

In [ ]:
```
# records with zero rooms
zero_rooms_anomalies = data['rooms'].loc[(data['rooms'] == 0)].count()
# mark for deletion
idx_to_be_removed.extend(data['rooms'].loc[(data['rooms'] == 0)].index)
```

removing outliers from total area

In [ ]:
```
data['total_area'].describe()
```

In [ ]:
```
# recommended apartment area is 28-30
data.loc[:,['total_area']].boxplot()
plt.show()
```

For apartments with a total area of more than 300 square meters, we will set the decimal separator, since it was most likely lost when entering

In [ ]:
```
for anomaly in data.loc[(data['total_area'] > 300),'total_area'] :
    data.loc[(data['total_area'] == anomaly) ,'total_area'] = anomaly / 10
```

deleting rows with less than 20 area

In [ ]:
```
total_area_anomalies = data['total_area'].loc[(data['total_area'] <= 20)].c
# Mark for deletion
idx_to_be_removed.extend(data['total_area'].loc[(data['total_area'] <=20)].
```

removing outliers from ceiling height

replacing null ceiling height values with median ceiling height of same value in rooms

In [ ]:
```
median_heights = data.groupby('rooms')['ceiling_height'].median()
data['ceiling_height'] = data.apply(lambda row: median_heights[row['rooms']
```

In [ ]:
```
data['ceiling_height'].describe()
```

In [ ]:
```
data.loc[:,['ceiling_height']].boxplot()
plt.show()
```

for ceilings over 20 meters we set the decimal separator

In [ ]:
```
for anomaly in data.loc[(data['ceiling_height'] >= 20),'ceiling_height'] :
    data.loc[(data['ceiling_height'] == anomaly) ,'ceiling_height'] = anoma
```

deleting houses with ceiling height more than 4.5 and less than 2

In [ ]:
```
idx_to_be_removed.extend(data['ceiling_height'].loc[(data['ceiling_height']
idx_to_be_removed.extend(data['ceiling_height'].loc[(data['ceiling_height']
```

removing outliers from kitchen area

In [ ]:
```
data['kitchen_area'].describe()
```

we replace negetive values with median of same total area

In [ ]:
```
for anomaly in data.loc[(data['kitchen_area'] < 0)].index :
    data.loc[(data.index == anomaly),'kitchen_area'] = data.loc[(data['tota
```

when kitchen area is larger than total area we assume misplacement of decimal and add it

In [ ]:
```
for anomaly in data.loc[(data['kitchen_area'] >  data['total_area'])].index
    data.loc[(data.index == anomaly),'kitchen_area'] = data.loc[(data.index
```

removing rows wit kitchen area greater than 50

In [ ]:
```
idx_to_be_removed.extend(data.loc[(data['kitchen_area'] >50)].index)
```

removing entries with kitchen area less than 4

In [ ]:
```
idx_to_be_removed.extend(data.loc[(data['kitchen_area'] <4)].index)
```

removing outliers from living area

In [ ]:
```
data['living_area'].describe()
```

when living area exceeedes total area we calculate it as total area-kitchen area

In [ ]:
```
for anomaly in data.loc[(data['living_area'] > data['total_area']) &
                        (data['kitchen_area'] < data['total_area'])]:

    data.loc[(data.index == anomaly),'living_area'] = data.loc[(data.index
```

remove reows with living area less than 5

In [ ]:
```
idx_to_be_removed.extend(data.loc[(data['living_area'] <5)].index)
```

Handling missing values

In [ ]:
```
data.isna().sum()
```

filling missing values in living and kitchen area

In [ ]:
```
# replacing na with median in living area
k_median_living_area = data['living_area'].median() / data['total_area'].me
data['living_area'] = data['living_area'].fillna(k_median_living_area*data[
```

In [ ]:
```
# replacing na with with median in kitchen area
k_median_kitchen_area = data['kitchen_area'].median() / data['total_area'].
data['kitchen_area'] = data['kitchen_area'].fillna(k_median_kitchen_area*da
```

filling missing values in balcony column

na means there is no balcony. we replace na with 0

In [ ]:
```
data['balcony'] = data['balcony'].fillna(0)
data['balcony'].describe()
```

In [ ]:
```
data.loc[:,['balcony']].boxplot()
plt.show()
```

In [ ]:
```
data
```

Pre processing

number of unique localities

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

removing rows with missing names

In [ ]:
```
idx_to_be_removed.extend(data['locality_name'].loc[data['locality_name'].is
```

Removing rows

checking how may rows to remove

In [ ]:
```
to_be_removed_count = len(pd.Series(idx_to_be_removed).unique())
to_be_removed_prc = np.round(float((to_be_removed_count * 100) / len(data))

print(f"Total records marked for deletion:{to_be_removed_count} ({to_be_rem
```

creating new dataframe with clean data while keeping original intact

In [ ]:
```
data_cleaned = data.copy()
data_cleaned.drop(labels = idx_to_be_removed, axis =0, inplace = True)
```

In [ ]:
```
data_cleaned.head(2)
```

In [ ]:
```
data_cleaned.info()
```

In [ ]:
```
data_cleaned[['rooms', 'total_area', 'ceiling_height', 'days_exposition', '
```

adding new coluns to dataset

price per square meter

In [ ]:
```
data_cleaned['sqm_cost'] = data_cleaned['last_price']/data['total_area'] /1
data_cleaned['sqm_cost'].describe()
```

day,month and year the ad was published

In [ ]:
```
data_cleaned['day'] = pd.to_datetime(data['first_day_exposition'], format='
data_cleaned['month'] = data_cleaned['first_day_exposition'].dt.month
data_cleaned['year'] = data_cleaned['first_day_exposition'].dt.year
```

Exploratory Analysis

total area

In [ ]:
```
data_cleaned.plot(kind='hist',y='total_area',histtype='step',range=(0, 200)
    grid=True,
    legend=True,
    figsize = (15,3))
plt.show()
```

most apartment on the market are from 25 to 40 sq meters

ratio of total area to living area

In [ ]:
```
ax=data_cleaned.plot(kind='hist',y='living_area',histtype='step',range=(0,2
data_cleaned.plot(ax=ax,kind='hist',y='total_area',histtype='step',range=(0
plt.show()
```

kitchen area

In [ ]:
```
# after pre-processing
ax = data_cleaned.plot(kind='hist',y='kitchen_area',histtype='step',range=(

# before pre-processing
data.plot(kind='hist',y='kitchen_area',histtype='step',range=(0, 50),bins=1
plt.show()
```

The graph shows that the results of preprocessing did not affect the shape of the distribution of the parameter under study. This means that the basic statistical parameters of the population remained unchanged.

relationship between last price and total area

In [ ]:
```
data_cleaned.plot(x='total_area',y='last_price',kind='scatter',grid=True,al
plt.show()
```

The price of a property is closely related to its total area, however, as these parameters increase, the relationship between them weakens significantly.

distribution of number of rooms

In [ ]:
```
data_cleaned['count'] = 1
data_cleaned.pivot_table(index='rooms', values='count', aggfunc='count').pl
plt.show()
```

floors

In [ ]:
```
data_cleaned.plot(x='floors_total', y='floor', kind='hexbin',gridsize=15, f
grid=True)
plt.show()
```

Most often offered for sale 1st and 2nd floors

Floors total

In [ ]:
```
data_cleaned['floors_total'].hist(bins=150, figsize=(15,3))
plt.show()
```

the market is dominated by series of houses with 5 and 9 storey buildings

day of week

In [ ]:
```
data_cleaned['day'].hist(bins=7,figsize=(20,3))
plt.show()
```

most ads are posted on weekdays

month

In [ ]:
```
data_cleaned['month'].hist(bins=12,figsize=(20,3))
plt.show()
```

there is a decline in ads during the summer

real estate sales speed

To analyze the speed of apartment sales, we will create an auxiliary dataframe, which will contain only those real estate objects that have information in the 'days_exposition' column

In [ ]:
```
data_sold = data[~data['days_exposition'].isna()]
```

In [ ]:
```
data_sold['days_exposition'].describe()
```

the duration of the sale is normal is a period of up to six months from six months to a year and a half is long, but within normal limits. There are cases when a property has been on sale for more than a year and a half.

In [ ]:
```
data_sold.loc[:,['days_exposition']].boxplot()
plt.show()
```

Factors that most influenced the total cost

To analyze the influence of factors on the price of a real estate property, we will construct a matrix of scatterplots for each of the factors in order to visually assess the presence of a relationship between the price and each of the factors

In [ ]:
```
list=(['last_price','total_area','living_area','kitchen_area','rooms'])
```

In [ ]:
```
pd.plotting.scatter_matrix(data_cleaned.loc[:,list], figsize=(9, 9))
plt.show()
```

In [ ]:
```
data_cleaned.loc[:,list].corr()
```

The price of a property is significantly influenced by its total area. The influence that other characteristics of the property have (living area, kitchen area, number of rooms) is a consequence of the fact that there is a direct correlation between these factors and the total area, and therefore they also show a correlation with the price.

In [ ]:
```
```