house_prices_data_prep

May 26, 2021

[1]:
```
import numpy as np
import pandas as pd
from scipy import stats

from sklearn.metrics import mean_absolute_error, make_scorer
from sklearn.model_selection import train_test_split, cross_val_score

from tqdm import tqdm
tqdm.pandas()

import seaborn as sns
sns.set(font_scale=1.5)

import matplotlib.pyplot as plt
import matplotlib.style as style
%matplotlib inline

import warnings
warnings.filterwarnings("ignore")
```

```
C:\Users\HaKKe\anaconda3\lib\site-packages\tqdm\std.py:658: FutureWarning: The
Panel class is removed from pandas. Accessing it from the top-level namespace
will also be removed in the next version
  from pandas import Panel
```

[2]:
```
import zipfile
with zipfile.ZipFile('introml-2020-property-prices.zip', 'r') as zip_ref:
    zip_ref.extractall('dataset/property-prices/')
```

[3]:
```
train_df = pd.read_csv('dataset/property-prices/Train.csv')
test_df = pd.read_csv('dataset/property-prices/Test.csv')
```

[4]:
```
# All_zeros = pd.read_csv('dataset/property-prices/SampleSubmission.csv')
# All_zeros['price'] = np.zeros(len(All_zeros))
# All_zeros.to_csv('submission_0.csv', index=None)
mean_absolute_value = 5851954.56518
```

[5]:
```
train_df.shape, test_df.shape
```

[5]:
```
((100000, 25), (100000, 24))
```

[6]:
```
train_df
```

[6]:
```
           id    date  street_id  build_tech  floor  area  rooms  balcon  \
0           0  2011-1        616         0.0      4    43      2       0
1           1  2011-1        112         0.0      3    33      1       0
2           2  2011-1        230         NaN      9    34      1       0
3           3  2011-1        302         1.0      4    60      3       0
4           4  2011-1        578         0.0      3    49      2       0
…         …     …          …          …    …   …    …      …
99995   99995  2012-3        612         0.0      3    36      1       0
99996   99996  2012-3        573         0.0      4    51      2       0
99997   99997  2012-3        550         NaN      9    48      2       0
99998   99998  2012-3        595         1.0     10    51      2       1
99999   99999  2012-3        388         1.0      2    34      1       1

       metro_dist  g_lift  …  kw5  kw6  kw7  kw8  kw9  kw10  kw11  kw12  \
0            30.0     1.0  …    0    0    0    0    0     0     0     0
1            15.0     1.0  …    0    0    0    0    0     0     0     0
2            25.0     NaN  …    0    0    0    0    0     0     0     0
3            15.0     0.0  …    0    0    0    0    0     0     0     0
4            30.0     NaN  …    0    0    0    0    0     0     0     0
…           …     …   …  …  …  …  …  …  …   …   …   …
99995        30.0     NaN  …    0    0    0    0    0     0     0     0
99996        30.0     NaN  …    0    0    0    0    0     0     0     0
99997        30.0     0.0  …    0    0    0    0    0     0     0     0
99998        15.0     1.0  …    0    0    0    0    0     0     0     0
99999        15.0     0.0  …    0    0    0    0    0     0     0     0

       kw13    price
0         0  1738000
1         0  1169000
2         0  2821000
3         0  5714000
4         0  1660000
…     …       …
99995     0  3898000
99996     0  8698000
99997     0  6498000
99998     0  9436000
99999     0  3230000

[100000 rows x 25 columns]
```

[7]:
```
train_df.dtypes
```

[7]:
```
id              int64
date           object
street_id       int64
build_tech    float64
floor           int64
area            int64
rooms           int64
balcon          int64
metro_dist    float64
g_lift        float64
n_photos        int64
kw1             int64
kw2             int64
kw3             int64
kw4             int64
kw5             int64
kw6             int64
kw7             int64
kw8             int64
kw9             int64
kw10            int64
kw11            int64
kw12            int64
kw13            int64
price           int64
dtype: object
```

[8]:
```
train_df.describe().transpose()
```

[8]:

	count	mean	std	min	25%	\
id	100000.0	4.999950e+04	2.886766e+04	0.0	24999.75	
street_id	100000.0	3.346366e+02	1.939479e+02	0.0	166.00	
build_tech	69817.0	5.298566e-01	5.823403e-01	0.0	0.00	
floor	100000.0	5.375840e+00	4.045109e+00	1.0	2.00	
area	100000.0	5.220378e+01	1.755998e+01	29.0	40.00	
rooms	100000.0	2.113640e+00	8.261554e-01	1.0	1.00	
balcon	100000.0	3.966000e-01	5.529480e-01	0.0	0.00	
metro_dist	94943.0	2.191936e+01	8.377479e+00	0.0	15.00	
g_lift	70133.0	4.997932e-01	5.000035e-01	0.0	0.00	
n_photos	100000.0	2.518600e+00	1.974278e+00	0.0	1.00	
kw1	100000.0	5.757000e-02	2.329297e-01	0.0	0.00	
kw2	100000.0	6.273100e-01	4.835230e-01	0.0	0.00	
kw3	100000.0	7.790000e-03	8.791697e-02	0.0	0.00	
kw4	100000.0	6.580000e-03	8.085029e-02	0.0	0.00	
kw5	100000.0	4.450000e-03	6.656006e-02	0.0	0.00	
kw6	100000.0	1.560000e-03	3.946621e-02	0.0	0.00	
kw7	100000.0	1.000000e-03	3.160712e-02	0.0	0.00	

```
kw8         100000.0  7.370000e-03  8.553219e-02       0.0         0.00
kw9         100000.0  7.390000e-03  8.564731e-02       0.0         0.00
kw10        100000.0  1.980000e-03  4.445334e-02       0.0         0.00
kw11        100000.0  3.100000e-04  1.760418e-02       0.0         0.00
kw12        100000.0  1.510000e-03  3.882956e-02       0.0         0.00
kw13        100000.0  2.000000e-04  1.414079e-02       0.0         0.00
price       100000.0  5.100166e+06  4.228087e+06  490000.0  2583000.00

                 50%         75%         max
id           49999.5    74999.25     99999.0
street_id      335.0      503.00       671.0
build_tech       0.0        1.00         2.0
floor            4.0        7.00        25.0
area            52.0       60.00       217.0
rooms            2.0        3.00         6.0
balcon           0.0        1.00         2.0
metro_dist      25.0       30.00        30.0
g_lift           0.0        1.00         1.0
n_photos         2.0        4.00        11.0
kw1              0.0        0.00         1.0
kw2              1.0        1.00         1.0
kw3              0.0        0.00         1.0
kw4              0.0        0.00         1.0
kw5              0.0        0.00         1.0
kw6              0.0        0.00         1.0
kw7              0.0        0.00         1.0
kw8              0.0        0.00         1.0
kw9              0.0        0.00         1.0
kw10             0.0        0.00         1.0
kw11             0.0        0.00         1.0
kw12             0.0        0.00         1.0
kw13             0.0        0.00         1.0
price       4063000.0  6220250.00  83634000.0
```

[9]:
```
mis_cols = train_df.columns[train_df.isna().any(axis=0)]
train_df[mis_cols].isna().sum(axis=0)
```

[9]:
```
build_tech    30183
metro_dist     5057
g_lift        29867
dtype: int64
```

[10]:
```
test_df[mis_cols].isna().sum(axis=0)
```

[10]:
```
build_tech    30213
metro_dist     4913
g_lift        29940
```

```
dtype: int64
```

[11]:
```
train_df.loc[train_df.isna().any(axis=1), :].head(10)
```

[11]:

	id	date	street_id	build_tech	floor	area	rooms	balcon	metro_dist	\
2	2	2011-1	230	NaN	9	34	1	0	25.0	
4	4	2011-1	578	0.0	3	49	2	0	30.0	
7	7	2011-1	427	0.0	4	41	2	0	30.0	
8	8	2011-1	74	NaN	12	70	2	1	30.0	
9	9	2011-1	365	0.0	6	50	2	0	30.0	
10	10	2011-1	555	NaN	24	39	1	0	25.0	
11	11	2011-1	55	0.0	7	63	3	0	15.0	
12	12	2011-1	666	0.0	3	34	1	0	15.0	
16	16	2011-1	25	0.0	3	59	3	0	NaN	
18	18	2011-1	145	NaN	5	52	2	1	25.0	

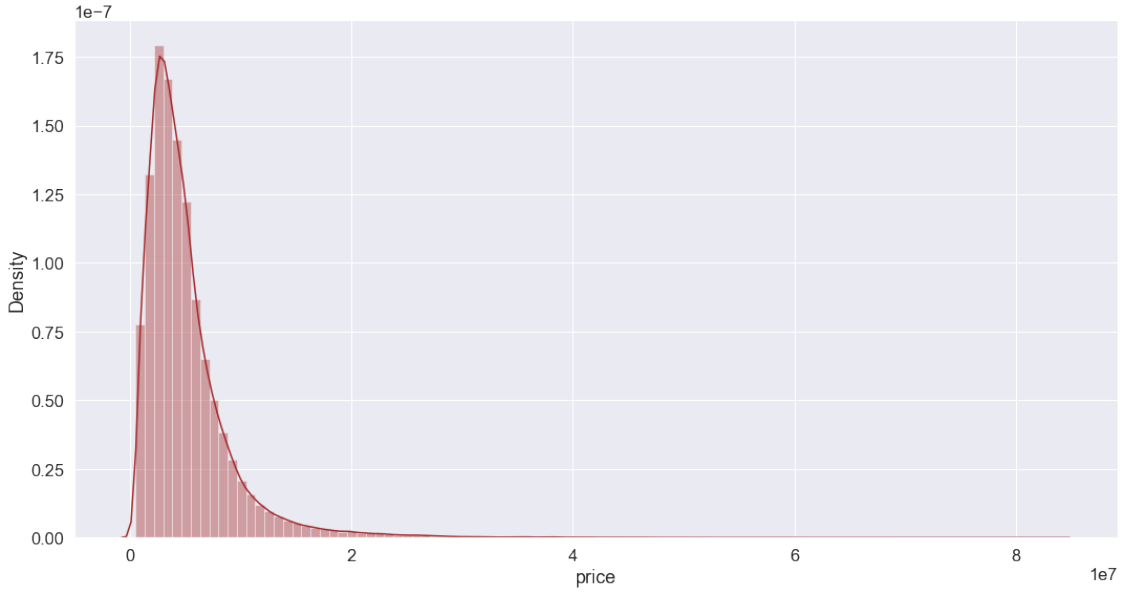
	g_lift	…	kw5	kw6	kw7	kw8	kw9	kw10	kw11	kw12	kw13	price
2	NaN	…	0	0	0	0	0	0	0	0	0	2821000
4	NaN	…	0	0	0	0	0	0	0	0	0	1660000
7	NaN	…	0	0	0	0	0	0	0	0	0	1102000
8	1.0	…	0	0	0	0	0	0	0	0	0	1000000
9	NaN	…	0	0	0	0	0	0	0	0	0	2910000
10	0.0	…	0	0	0	0	0	0	0	0	0	2744000
11	NaN	…	0	0	0	0	0	0	0	0	0	3588000
12	NaN	…	0	0	0	0	0	0	0	0	0	2774000
16	0.0	…	0	0	0	0	0	0	0	0	0	7181000
18	NaN	…	0	0	0	0	0	0	0	0	0	7455000

```
[10 rows x 25 columns]
```

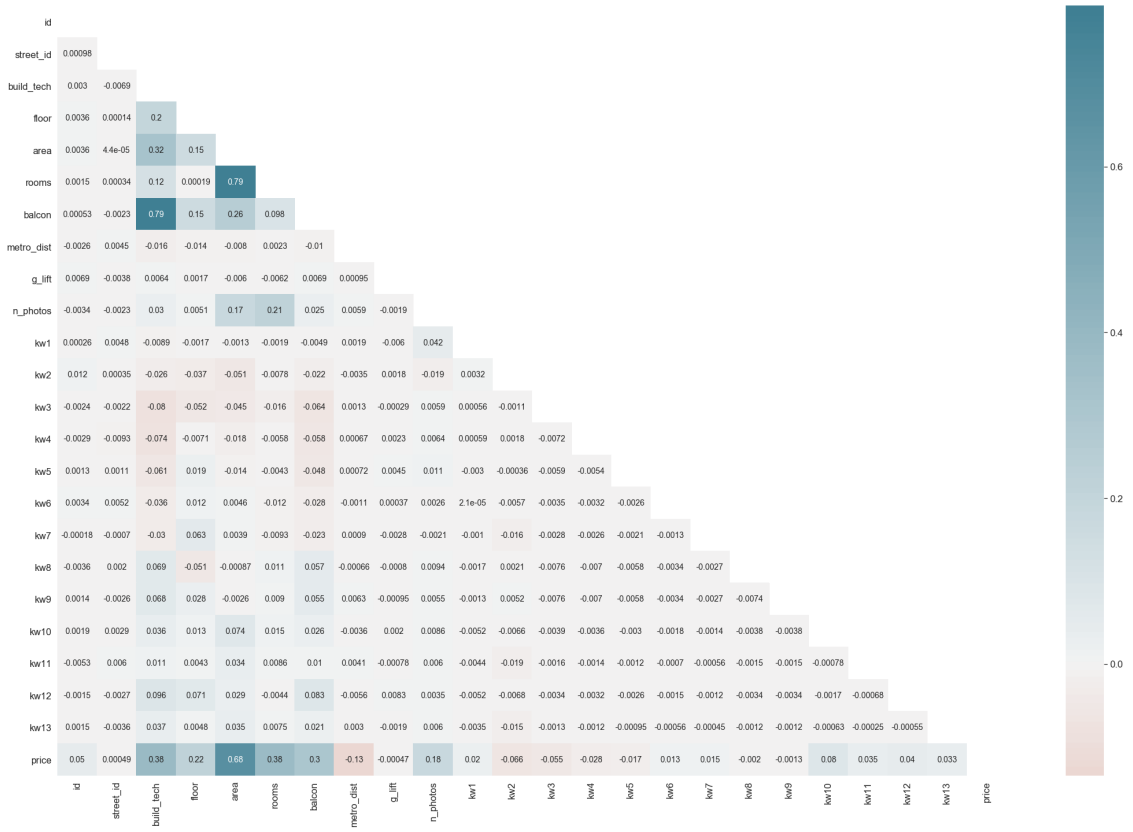
[12]:
```
train_df.isna().sum(axis=1).value_counts()
```

[12]:
```
0    46575
1    42221
2    10726
3      478
dtype: int64
```

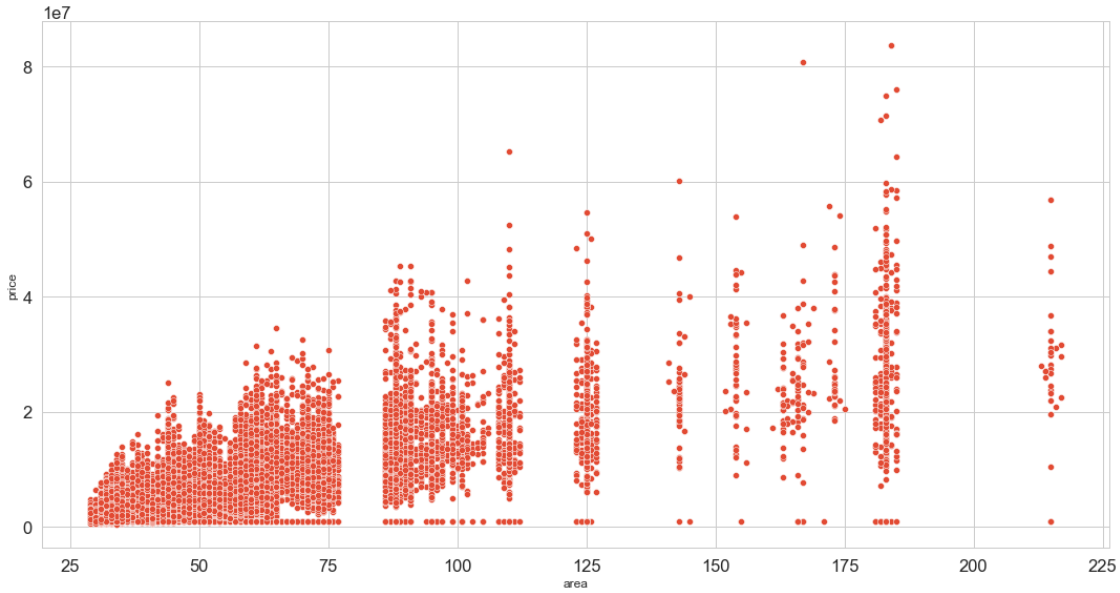
[13]:
```
fig = plt.figure(figsize=(15,8))
sns.distplot(train_df['price'],bins=100,color="brown")
sns.set_style("white")
plt.tight_layout()
```



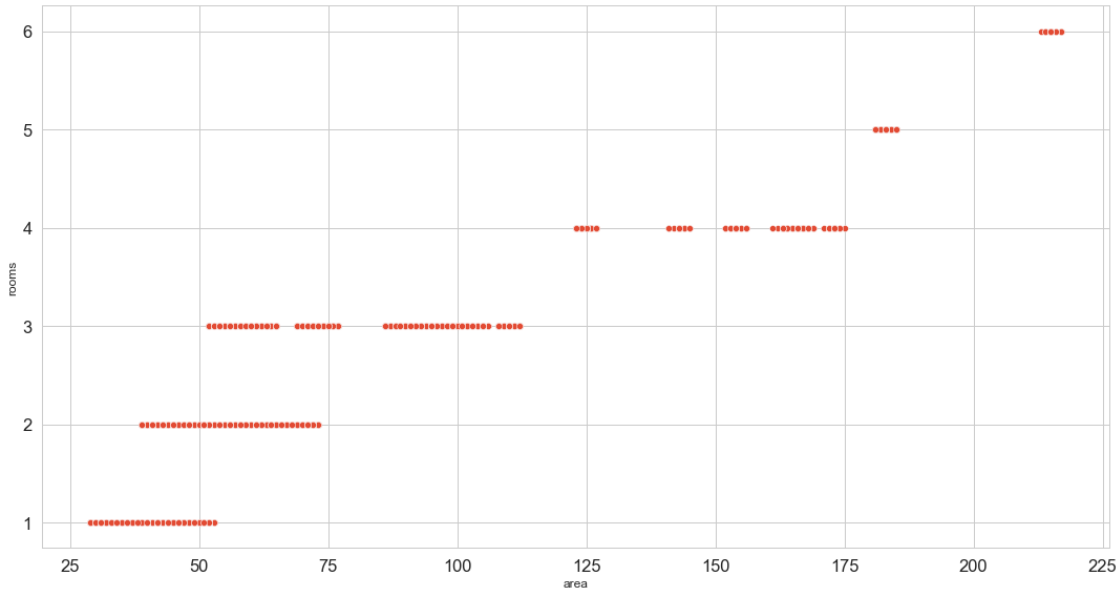
[14]:
```
style.use('ggplot')
sns.set_style('whitegrid')
plt.subplots(figsize = (30,20))

mask = np.zeros_like(train_df.corr(), dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

sns.heatmap(train_df.corr(),
            cmap=sns.diverging_palette(20, 220, n=200),
            mask = mask,
            annot=True,
            center = 0,
            annot_kws={"size": 14},
            cbar="coolwarm",
           );
plt.tight_layout()
```



[15]:
```
fig = plt.figure(figsize=(15,8))
sns.scatterplot(x="area",y="price",data=train_df)
sns.set_style("whitegrid")
plt.tight_layout()
```

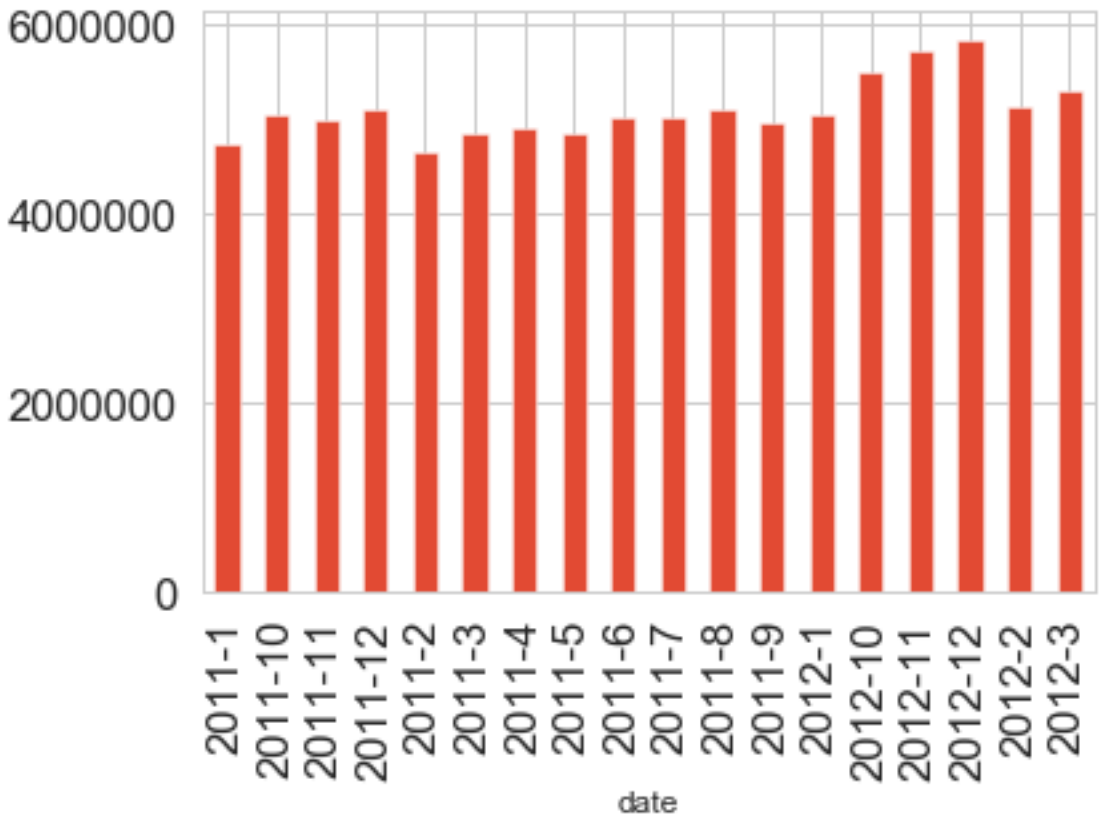


[16]:
```
fig = plt.figure(figsize=(15,8))
sns.scatterplot(x="area",y="rooms",data=train_df)
sns.set_style("whitegrid")
plt.tight_layout()
```



[17]:
```
prev_train = train_df.copy()
```

[18]:
```
train_df.groupby('date')['price'].mean().plot.bar()
plt.show()
```



[19]:
```
train_df.price.value_counts().head(10)
```

[19]:
```
1000000    3968
2686000      37
1946000      31
4421000      30
2113000      30
2614000      30
2420000      29
2676000      29
3077000      28
3305000      28
Name: price, dtype: int64
```

[20]:
```
outliers = train_df[train_df.price == 1e6][train_df[train_df.price == 1e6].area␣
↪> 60].id.values # changable param
train_df.drop(outliers, inplace=True)
```

[21]:
```
indexes = train_df[train_df['price'] == 1e6]['id'].values
train_df.drop(np.random.choice(indexes, size=len(indexes)-400, replace=False),␣
↪inplace=True)
```

[22]:
```
all_data = pd.concat((train_df.iloc[:, :-1], test_df)).reset_index(drop=True)
```

[23]:
```
def fill_missing(dataframe):
    dataframe['metro_dist'] = dataframe.groupby('street_id')['metro_dist'].
 ↪transform(lambda x: x.fillna(x.mean()))
    dataframe['g_lift'] = dataframe.groupby('street_id')['g_lift'].
 ↪transform(lambda x: x.fillna(np.round(x.mean())))
    for index, row in dataframe.iterrows(): # better to rewrite if used again
        if np.isnan(row['build_tech']):
            dataframe.at[index, 'build_tech'] = row['balcon']
    return dataframe

train_df = fill_missing(train_df)
all_data = fill_missing(all_data)
```

[24]:
```
train_df.date.value_counts().index.sort_values()
```

[24]:
```
Index(['2011-1', '2011-10', '2011-11', '2011-12', '2011-2', '2011-3', '2011-4',
       '2011-5', '2011-6', '2011-7', '2011-8', '2011-9', '2012-1', '2012-10',
       '2012-11', '2012-12', '2012-2', '2012-3'],
      dtype='object')
```

[25]:
```
test_df.date.value_counts().index.sort_values()
```

[25]:
```
Index(['2012-3', '2012-4', '2012-5', '2012-6', '2012-7', '2012-8', '2012-9',
       '2013-1', '2013-10', '2013-11', '2013-12', '2013-2', '2013-3', '2013-4',
       '2013-5', '2013-6', '2013-7', '2013-8', '2013-9'],
      dtype='object')
```

[26]:
```
X_test_df = train_df[(train_df.date == '2012-10') |
                     (train_df.date == '2012-11') |
                     (train_df.date == '2012-12')]
test_indexes = X_test_df.id.values
```

[27]:
```
print('mean test target:', X_test_df.price.mean())
```

```
mean test target: 5869762.4686716795
```

[28]:
```
train_indexes = np.setxor1d(train_df.id.values, test_indexes)
X_train_df = train_df.loc[train_indexes]
```

[29]:
```
print('mean train target:', X_train_df.price.mean())
```

```
mean train target: 5129327.480365842
```

[30]:
```
mean_price_0 = X_train_df.groupby('street_id')['price'].mean()
```

```
X_train_df['mean_square_root_price'] = pd.
 ↪Series(data=(mean_price_0[X_train_df['street_id']].reset_index(drop=True) /␣
 ↪X_train_df['area'].reset_index(drop=True)).values, index=X_train_df.id)

X_test_df['mean_square_root_price'] = pd.
 ↪Series(data=(mean_price_0[X_test_df['street_id']].reset_index(drop=True) /␣
 ↪X_test_df['area'].reset_index(drop=True)).values, index=X_test_df.id)
```

[31]:
```
X_train_df['avg_room_area'] = X_train_df['area'] / X_train_df['rooms']
X_train_df['area_and_balcon'] = X_train_df['area'] + 5. * X_train_df['balcon']

X_test_df['avg_room_area'] = X_test_df['area'] / X_test_df['rooms']
X_test_df['area_and_balcon'] = X_test_df['area'] + 5. * X_test_df['balcon']
```

[32]:
```
mean_floor_price_0 = X_train_df.groupby(['street_id', 'floor'])['price'].mean()
```

[33]:
```
X_train_df['mean_street_floor_square_price'] = mean_floor_price_0[
    list(zip(X_train_df.street_id, X_train_df.floor))].values / X_train_df.area
```

[34]:
```
def getClosest_0(street, floor):
    ex_floor = min(mean_floor_price_0[street].keys(), key=lambda x:abs(x-floor))
    return mean_floor_price_0[street, ex_floor]
```

[35]:
```
temp_series = pd.Series()
for index, row in tqdm(X_test_df.iterrows()):
    try:
        temp_series.at[index] = mean_floor_price_0[row['street_id'],␣
 ↪row['floor']] / row['area']
    except KeyError:
        temp_series.at[index] = getClosest_0(row['street_id'], row['floor']) /␣
 ↪row['area']
```

```
15960it [00:27, 583.28it/s]
```

[36]:
```
X_test_df['mean_street_floor_square_price'] = temp_series
```

[37]:
```
X_train_df['metro_dist'] = X_train_df['metro_dist'].div(5).round(0) * 5.
X_train_df['metro_dist'] = 30.0 - X_train_df['metro_dist']

X_test_df['metro_dist'] = X_test_df['metro_dist'].div(5).round(0) * 5.
X_test_df['metro_dist'] = 30.0 - X_test_df['metro_dist']
```

[38]:
```
mean_price_1 = train_df.groupby('street_id')['price'].mean()
all_data['mean_square_root_price'] = mean_price_1[all_data['street_id']].
 ↪reset_index(drop=True) / all_data['area']
```

[39]:
```
all_data['avg_room_area'] = all_data['area'] / all_data['rooms']
all_data['area_and_balcon'] = all_data['area'] + 5. * all_data['balcon']
```

[40]:
```
mean_floor_price_1 = train_df.groupby(['street_id', 'floor'])['price'].mean()
```

[41]:
```
all_data['mean_street_floor_square_price'] = np.zeros(len(all_data))
all_data.mean_street_floor_square_price.iloc[:len(train_df)] =␣
↪mean_floor_price_1[
    list(zip(train_df.street_id, train_df.floor))].values / train_df.area.
↪reset_index(drop=True)
```

[42]:
```
def getClosest_1(street, floor):
    ex_floor = min(mean_floor_price_1[street].keys(), key=lambda x:abs(x-floor))
    return mean_floor_price_1[street, ex_floor]
```

[43]:
```
temp_series = pd.Series()
for i, (index, row) in enumerate(all_data[len(train_df):].iterrows()):
    try:
        temp_series.at[index] = mean_floor_price_1[row['street_id'],␣
↪row['floor']] / row['area']
    except KeyError:
        temp_series.at[index] = getClosest_1(row['street_id'], row['floor']) /␣
↪row['area']
    if i % 1e4 == 0:
        print(i)
```

```
0
10000
20000
30000
40000
50000
60000
70000
80000
90000
```

[44]:
```
all_data.mean_street_floor_square_price.iloc[len(train_df):] = temp_series
```

[45]:
```
all_data['metro_dist'] = all_data['metro_dist'].div(5).round(0) * 5.
all_data['metro_dist'] = 30.0 - all_data['metro_dist']
```

[46]:
```
Train_filled = all_data.iloc[:len(train_df)]
Train_filled = pd.concat((Train_filled, train_df.iloc[:, -1].
↪reset_index(drop=True)), axis=1)
```

[47]:
```
X_train_df = X_train_df[[c for c in X_train_df if c not in ['price']] +␣
↪['price']]
X_test_df = X_test_df[[c for c in X_test_df if c not in ['price']] + ['price']]
```

[48]:
```
X_train_df.to_csv('dataset/X_train.csv', index=None)
X_test_df.to_csv('dataset/X_test.csv', index=None)
```