In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
%matplotlib inline

df = pd.read_csv('data_flats.csv',sep=';')
df
```

Out[1]:

	id	full_sq	life_sq	floor	sub_area	preschool_quota	preschool_education_centers_raion	school_quota
0	1	43	27.0	4.0	Bibirevo	5001.0	5	11065.0
1	2	34	19.0	3.0	Nagatinskij Zaton	3119.0	5	6237.0
2	3	43	29.0	2.0	Tekstil'shhiki	1463.0	4	5580.0
3	4	89	50.0	9.0	Mitino	6839.0	9	17063.0
4	5	77	77.0	4.0	Basmannoe	3240.0	7	7770.0
...	...	...	...	...	...	...	...	..
30464	30469	44	27.0	7.0	Otradnoe	5088.0	4	12721.0
30465	30470	86	59.0	3.0	Tverskoe	1874.0	4	6772.0
30466	30471	45	NaN	10.0	Poselenie Vnukovskoe	NaN	0	NaN
30467	30472	64	32.0	5.0	Obruchevskoe	2372.0	6	6083.0
30468	30473	43	28.0	1.0	Novogireevo	2215.0	4	5824.0

30469 rows × 56 columns

In [2]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30469 entries, 0 to 30468
Data columns (total 56 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   id                                     30469 non-null  int64
 1   full_sq                                30469 non-null  int64
 2   life_sq                                24086 non-null  float64
 3   floor                                  30302 non-null  float64
 4   sub_area                               30469 non-null  object
 5   preschool_quota                        23781 non-null  float64
 6   preschool_education_centers_raion      30469 non-null  int64
 7   school_quota                           23784 non-null  float64
 8   school_education_centers_raion         30469 non-null  int64
 9   school_education_centers_top_20_raion  30469 non-null  int64
 10  hospital_beds_raion                    16029 non-null  float64
 11  healthcare_centers_raion               30469 non-null  int64
 12  university_top_20_raion                30469 non-null  int64
 13  sport_objects_raion                    30469 non-null  int64
 14  additional_education_raion             30469 non-null  int64
 15  culture_objects_top_25_raion           30469 non-null  int64
 16  shopping_centers_raion                 30469 non-null  int64
 17  office_raion                           30469 non-null  int64
 18  metro_min_avto                         30469 non-null  float64
 19  metro_km_avto                          30469 non-null  float64
```

```
 20  metro_min_walk                       30444 non-null  float64
 21  metro_km_walk                        30444 non-null  float64
 22  kindergarten_km                      30469 non-null  float64
 23  school_km                            30469 non-null  float64
 24  park_km                              30469 non-null  float64
 25  green_zone_km                        30469 non-null  float64
 26  industrial_km                        30469 non-null  float64
 27  railroad_station_walk_km             30444 non-null  float64
 28  railroad_station_walk_min            30444 non-null  float64
 29  public_transport_station_km          30469 non-null  float64
 30  public_transport_station_min_walk    30469 non-null  float64
 31  mkad_km                              30469 non-null  float64
 32  ttk_km                               30469 non-null  float64
 33  sadovoe_km                           30469 non-null  float64
 34  bulvar_ring_km                       30469 non-null  float64
 35  kremlin_km                           30469 non-null  float64
 36  big_market_km                        30469 non-null  float64
 37  market_shop_km                       30469 non-null  float64
 38  fitness_km                           30469 non-null  float64
 39  swim_pool_km                         30469 non-null  float64
 40  ice_rink_km                          30469 non-null  float64
 41  stadium_km                           30469 non-null  float64
 42  basketball_km                        30469 non-null  float64
 43  hospice_morgue_km                    30469 non-null  float64
 44  university_km                        30469 non-null  float64
 45  workplaces_km                        30469 non-null  float64
 46  shopping_centers_km                  30469 non-null  float64
 47  office_km                            30469 non-null  float64
 48  additional_education_km              30469 non-null  float64
 49  preschool_km                         30469 non-null  float64
 50  big_church_km                        30469 non-null  float64
 51  church_synagogue_km                  30469 non-null  float64
 52  theater_km                           30469 non-null  float64
 53  museum_km                            30469 non-null  float64
 54  ecology                              30469 non-null  object
 55  price_doc                            30469 non-null  int64
dtypes: float64(41), int64(13), object(2)
memory usage: 13.0+ MB
```

In [3]:

```
df = df.dropna()
df
```

Out[3]:

	id	full_sq	life_sq	floor	sub_area	preschool_quota	preschool_education_centers_raion	school_quota
0	1	43	27.0	4.0	Bibirevo	5001.0	5	11065.0
1	2	34	19.0	3.0	Nagatinskij Zaton	3119.0	5	6237.0
2	3	43	29.0	2.0	Tekstil'shhiki	1463.0	4	5580.0
4	5	77	77.0	4.0	Basmannoe	3240.0	7	7770.0
6	7	25	14.0	10.0	Sokol'niki	933.0	5	5050.0
...	...	...	...	...	...	...	...	...
30461	30466	56	29.0	13.0	Severnoe Tushino	4116.0	5	9891.0
30462	30467	56	51.0	19.0	Sviblovo	2057.0	1	3741.0
30465	30470	86	59.0	3.0	Tverskoe	1874.0	4	6772.0
30467	30472	64	32.0	5.0	Obruchevskoe	2372.0	6	6083.0
30468	30473	43	28.0	1.0	Novogireevo	2215.0	4	5824.0

13652 rows × 56 columns

Выявить наличие ошибочных данных

В качестве зависимого признака выступает стоимость квартиры Y (цена квартиры, руб). В качестве независимых выбраны следующие:

1. full_sq — общая площадь;+
2. life_sq — жилая площадь;+
3. floor — этаж;-
4. sub_area — служебные площади;-
5. preschool_quota — дошкольные образовательные центры;-
6. preschool_education_centers_raion — районные дошкольные образовательные центры;+
7. school_quota — школы;-
8. school_education_centers_raion — районные школы;+
9. school_education_centers_top_20_raion — рейтинг районных школ;-
10. hospital_beds_raion — районные больницы;-
11. healthcare_centers_raion — районные центры здоровья;-
12. university_top_20_raion — рейтинг районных институтов;-
13. sport_objects_raion — районные спортивные объекты;+
14. additional_education_raion — дополнительные районные образовательные учереждения;-
15. culture_objects_top_25_raion — рейтинг районных культурных объектов;-
16. shopping_centers_raion — районные шоппинг центров;-
17. office_raion — районные учереждения;-
18. metro_min_avto — время до метро на машине;+
19. metro_km_avto — расстояние до метро на машине;+
20. metro_min_walk — время до метро пешком;+
21. metro_km_walk — расстояние до метро пешком;+
22. kindergarten_km — расстояние до детского сада;+
23. school_km — расстояние до школы;+
24. park_km — расстояние до парка;+
25. green_zone_km — расстояние до зеленой зоны;-
26. industrial_km — расстояние до промышленного предприятия;-
27. railroad_station_walk_km — расстояние до станции;+
28. railroad_station_walk_min — минимальное расстояние до станции;+
29. public_transport_station_km — расстояние до остановки общественного транспорта;+
30. public_transport_station_min_walk — минимальное расстояние до остановки общественного транспорта;+
31. mkad_km — расстояние до МКАДа;+
32. ttk_km — расстояние до ТРК;-
33. sadovoe_km — расстояние до Садового кольца;+
34. bulvar_ring_km — расстояние до Бульварного кольца;+
35. kremlin_km — расстояние до Кремля;+
36. big_market_km — расстояние до крупного рынка;+
37. market_shop_km — расстояние до рынка;+
38. fitness_km — расстояние до фитнеса;+
39. swim_pool_km — расстояние до бассейна;-
40. ice_rink_km — расстояние до катка;+

41. stadium_km — расстояние до стадиона;+
42. basketball_km — расстояние до секции баскетбола;-
43. hospice_morgue_km — расстояние до хосписа;+
44. university_km — расстояние до университета;-
45. workplaces_km — расстояние до работы;-
46. shopping_centers_km — расстояние до торгового центра;+
47. office_km — расстояние до офиса;-
48. additional_education_km — расстояние до центра доп.образования;+
49. preschool_km — расстояние до дошкольного учреждения;+
50. big_church_km — расстояние до церкви;-
51. church_synagogue_km — расстояние до синагоги;+
52. theater_km — расстояние до театра;-
53. museum_km — расстояние до театра;-
54. ecology экололология;-

Следует отметить, что переменные, Y, X2 — X4 непрерывные, X1, X5, X8 — категориальные переменные.

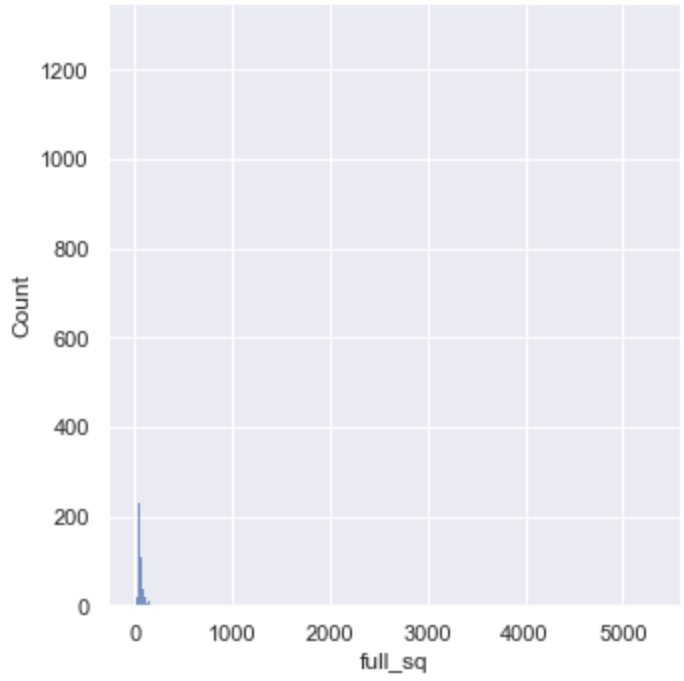
Задача работы состоит в построении уравнения множественной регрессии для предложенных данных в виде:

$$Y = f(X_1, X_2, X_3, \ldots, X_{54})$$

Исследуя распределение значиний каждого столбца, выяснили что некотороые данные не подчиняются нормальному закону распределения, поэтому мы исключаем их из выборки. Далее строим попарную корреляционную матрицу для дальнейшего исследования выбросов данных.

In [15]:
```
sns.set()
sns.displot(df['full_sq'])
```

Out[15]:
```
<seaborn.axisgrid.FacetGrid at 0x1e6c6e4b9a0>
```



In [4]:
```
df_cut = df[["full_sq","life_sq","preschool_education_centers_raion","school_education_cer
             "sport_objects_raion","metro_min_avto","metro_km_avto","metro_min_walk","metr
             "kindergarten_km","school_km","park_km","railroad_station_walk_km","railroad_
             "public_transport_station_km","public_transport_station_min_walk","mkad_km","
```

```
                "bulvar_ring_km","kremlin_km","big_market_km","market_shop_km","fitness_km","
                "hospice_morgue_km","shopping_centers_km","additional_education_km","preschoo
               "price_doc"]].copy()
df_cut
```

Out[4]:

	full_sq	life_sq	preschool_education_centers_raion	school_education_centers_raion	sport_objects_raion	metro_
0	43	27.0	5	5	7	
1	34	19.0	5	8	6	
2	43	29.0	4	7	5	
4	77	77.0	7	9	25	
6	25	14.0	5	5	17	
...	...	...	...	...	...	
30461	56	29.0	5	5	1	
30462	56	51.0	1	2	4	
30465	86	59.0	4	4	29	
30467	64	32.0	6	8	11	
30468	43	28.0	4	4	7	

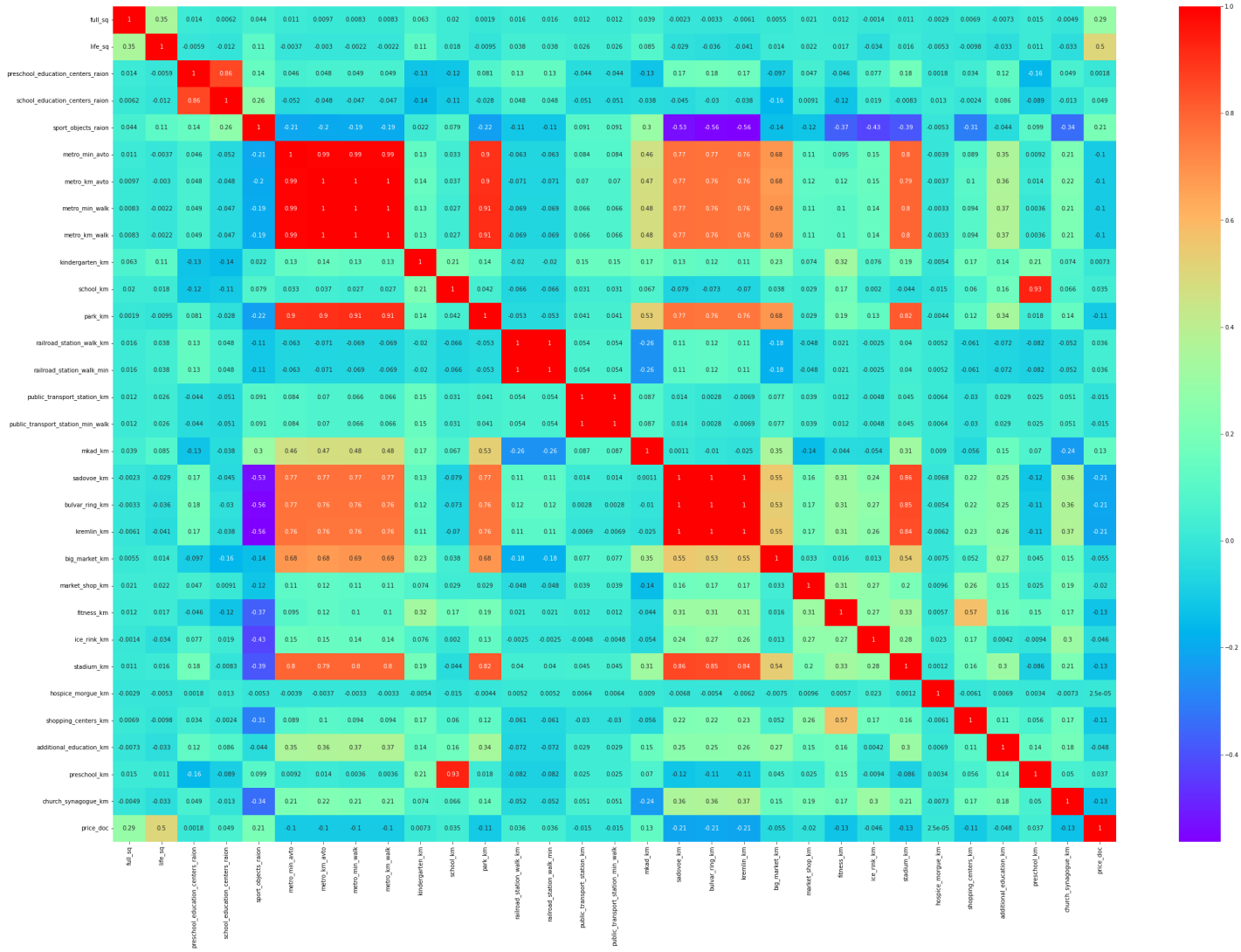
13652 rows × 31 columns

In [5]:

```
correlation = df_cut.corr()
fig = plt.figure()
axes = fig.add_axes([0,0,5,5])
sns.heatmap(correlation, annot=True, cmap='rainbow')
```

Out[5]:

```
<Axes:>
```



In [6]:
```
np.linalg.matrix_rank(correlation)
```

Out[6]:
```
28
```

In [7]:
```
np.linalg.det(correlation)
```

Out[7]:
```
-8.510628290780297e-58
```

По получившейся матрице видно, что требуется убрать зависящие друг от друга данные: это - metro_min_avto, metro_km_avto, metro_min_walk, metro_km_walk, park_km;

In [8]:
```
df_cut = df[["full_sq","life_sq",
             "sport_objects_raion","metro_min_avto",
             "mkad_km",
             "kremlin_km","fitness_km","shopping_centers_km","stadium_km",
             "church_synagogue_km",
            "price_doc"]].copy()
df_cut
```

Out[8]:

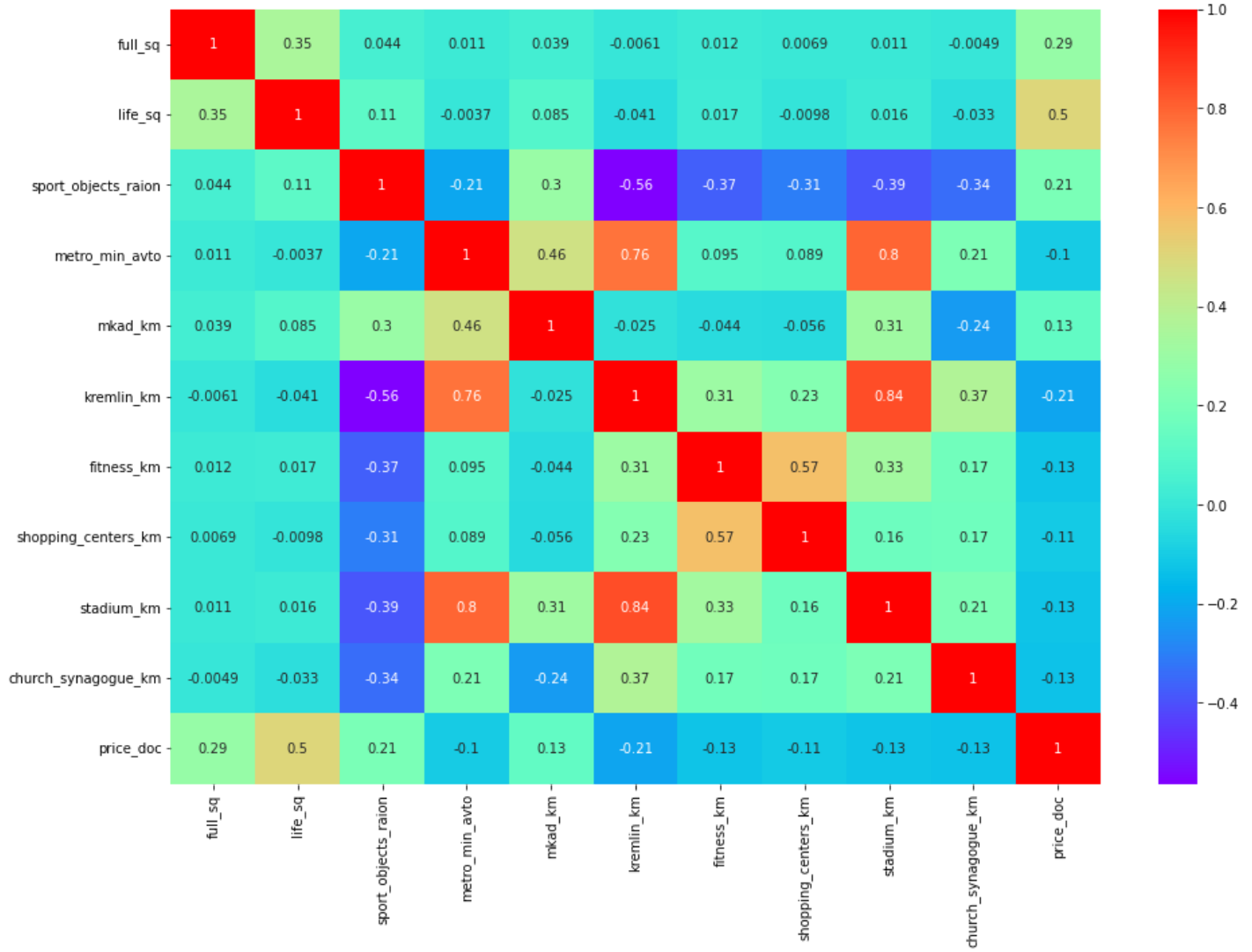
	full_sq	life_sq	sport_objects_raion	metro_min_avto	mkad_km	kremlin_km	fitness_km	shopping_centers_km
0	43	27.0	7	2.590241	1.422391	15.156211	0.485841	0.648488
1	34	19.0	6	0.936700	9.503405	8.698054	0.668364	0.519311
2	43	29.0	5	2.120999	5.604800	9.067885	0.733101	1.486533

	full_sq	life_sq	sport_objects_raion	metro_min_avto	mkad_km	kremlin_km	fitness_km	shopping_centers_km
4	77	77.0	25	1.257186	11.616653	2.578671	0.220288	0.429052
6	25	14.0	17	1.453762	8.618597	6.468719	0.132256	0.513689
...	...	...	...	...	...	...	...	...
30461	56	29.0	1	2.622565	1.486707	16.626186	1.003262	0.232778
30462	56	51.0	4	0.815305	5.363124	10.514468	0.378930	0.187828
30465	86	59.0	29	1.060577	13.100989	3.269284	0.398831	0.540003
30467	64	32.0	11	3.377814	2.327138	13.622569	0.412813	1.108672
30468	43	28.0	7	0.584636	1.920884	11.812614	0.819001	0.224601

13652 rows × 11 columns

In [9]:
```
correlation = df_cut.corr()
fig = plt.figure()
axes = fig.add_axes([0,0,2,2])
sns.heatmap(correlation, annot=True, cmap='rainbow')
```

Out[9]:
```
<Axes:>
```



In [10]:
```
np.linalg.matrix_rank(correlation)
```

Out[10]: 11

In [11]:
```
np.linalg.det(correlation)
```

Out[11]: 0.00415845664671564

In [12]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics

X = df_cut.iloc[:,0:10].values
Y = df_cut.iloc[:,10].values

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3)
my_model = LinearRegression()
my_model.fit(X_train, Y_train)

y_pred = my_model.predict(X_test)
print(my_model.intercept_,my_model.coef_)

print('MAE:', metrics.mean_absolute_error(Y_test,y_pred))

print('MSE:', metrics.mean_squared_error(Y_test, y_pred))

print('R_2:', metrics.r2_score(Y_test,y_pred))
```

```
4635337.874932846 [  10239.38405155  129535.52205714    3439.8884422   -26550.48497499
   93980.24797157 -110508.22765728 -712115.79411557 -134789.69964381
    6861.51310649 -259882.8599155 ]
MAE: 2789188.533963208
MSE: 26381757876418.07
R_2: 0.29691794125069515
```

In [28]:
```
X
```

Out[28]:
```
array([[43.        , 27.        ,  7.        , ...,  0.64848764,
         8.14859077,  0.62818655],
       [34.        , 19.        ,  6.        , ...,  0.51931132,
         6.12707278,  0.47144652],
       [43.        , 29.        ,  5.        , ...,  1.48653302,
         0.76756877,  0.75594602],
       ...,
       [86.        , 59.        , 29.        , ...,  0.54000281,
         2.96573222,  0.31002056],
       [64.        , 32.        , 11.        , ...,  1.1086716 ,
         2.37410568,  0.47602088],
       [43.        , 28.        ,  7.        , ...,  0.22460143,
         3.64165629,  0.86733182]])
```

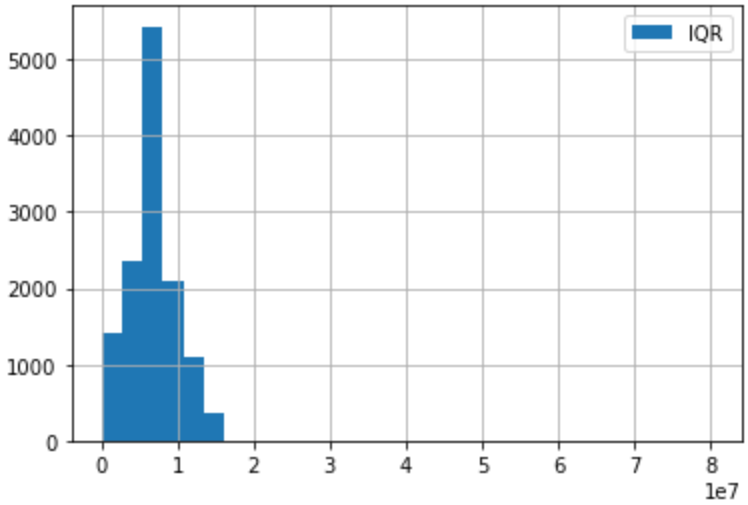
In [13]:
```
median = df_cut.price_doc.median()
print(median)
IQR = df_cut.price_doc.quantile(0.75, interpolation='midpoint') - df_cut.price_doc.quantil
perc25 = df_cut.price_doc.quantile(0.25, interpolation='midpoint')
perc75 = df_cut.price_doc.quantile(0.75, interpolation='midpoint')
print('25-й перцентиль:{},'.format(perc25),
      '75-й перцентиль: {},'.format(perc75),
      "IQR: {}, ".format(IQR),"Гарницы выбросов: [{f}, {l}].".format(f=perc25 - 1.5*IQR,
                                                                     l=perc75+1.5*IQR))

df_cut.price_doc.loc[df_cut.price_doc.between(perc25-1.5*IQR,
                                              perc75+1.5*IQR)].hist(bins=30,
                                                                    range=(1e+5,8e+7),
```

```
                                                                label='IQR')
plt.legend()
df_cut = df_cut.loc[df_cut.price_doc.between(perc25-1.5*IQR,
                                  perc75+1.5*IQR)]
df_cut.price_doc.describe()
```

```
6700000.0
25-й перцентиль:5250000.0, 75-й перцентиль: 9100000.0, IQR: 3850000.0,  Гарницы выбросов:
[-525000.0, 14875000.0].
```

Out[13]:

```
count    1.273600e+04
mean     6.770514e+06
std      3.054684e+06
min      1.000000e+05
25%      5.100000e+06
50%      6.500000e+06
75%      8.400000e+06
max      1.485000e+07
Name: price_doc, dtype: float64
```



In [14]:

```
X = df_cut.iloc[:,0:10].values
Y = df_cut.iloc[:,10].values

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.4)
my_model = LinearRegression()
my_model.fit(X_train, Y_train)

y_pred = my_model.predict(X_test)
print(my_model.intercept_,my_model.coef_)

print('MAE:', metrics.mean_absolute_error(Y_test,y_pred))

print('MSE:', metrics.mean_squared_error(Y_test, y_pred))

print('R_2:', metrics.r2_score(Y_test,y_pred))
```

```
6273505.037948447 [   2909.45938085   48012.33698831  -31236.81489099  -40051.35287605
   44389.19661411  -45061.0936616  -411071.365728    -216257.75805989
    7514.55780977  -66484.0761289 ]
MAE: 2108057.6841243743
MSE: 8031448007010.996
R_2: 0.15343000540758012
```

In [19]:

```
len(X)
```

Out[19]:

```
12736
```

In [15]:

```
df_cut = df[["full_sq","life_sq",
             "sport_objects_raion",
             "kremlin_km",
             "price_doc"]].copy()
df_cut
```

Out[15]:

	full_sq	life_sq	sport_objects_raion	kremlin_km	price_doc
0	43	27.0	7	15.156211	5850000
1	34	19.0	6	8.698054	6000000
2	43	29.0	5	9.067885	5700000
4	77	77.0	25	2.578671	16331452
6	25	14.0	17	6.468719	5500000
...	...	...	...	...	...
30461	56	29.0	1	16.626186	12000000
30462	56	51.0	4	10.514468	10262010
30465	86	59.0	29	3.269284	25000000
30467	64	32.0	11	13.622569	13500000
30468	43	28.0	7	11.812614	5600000

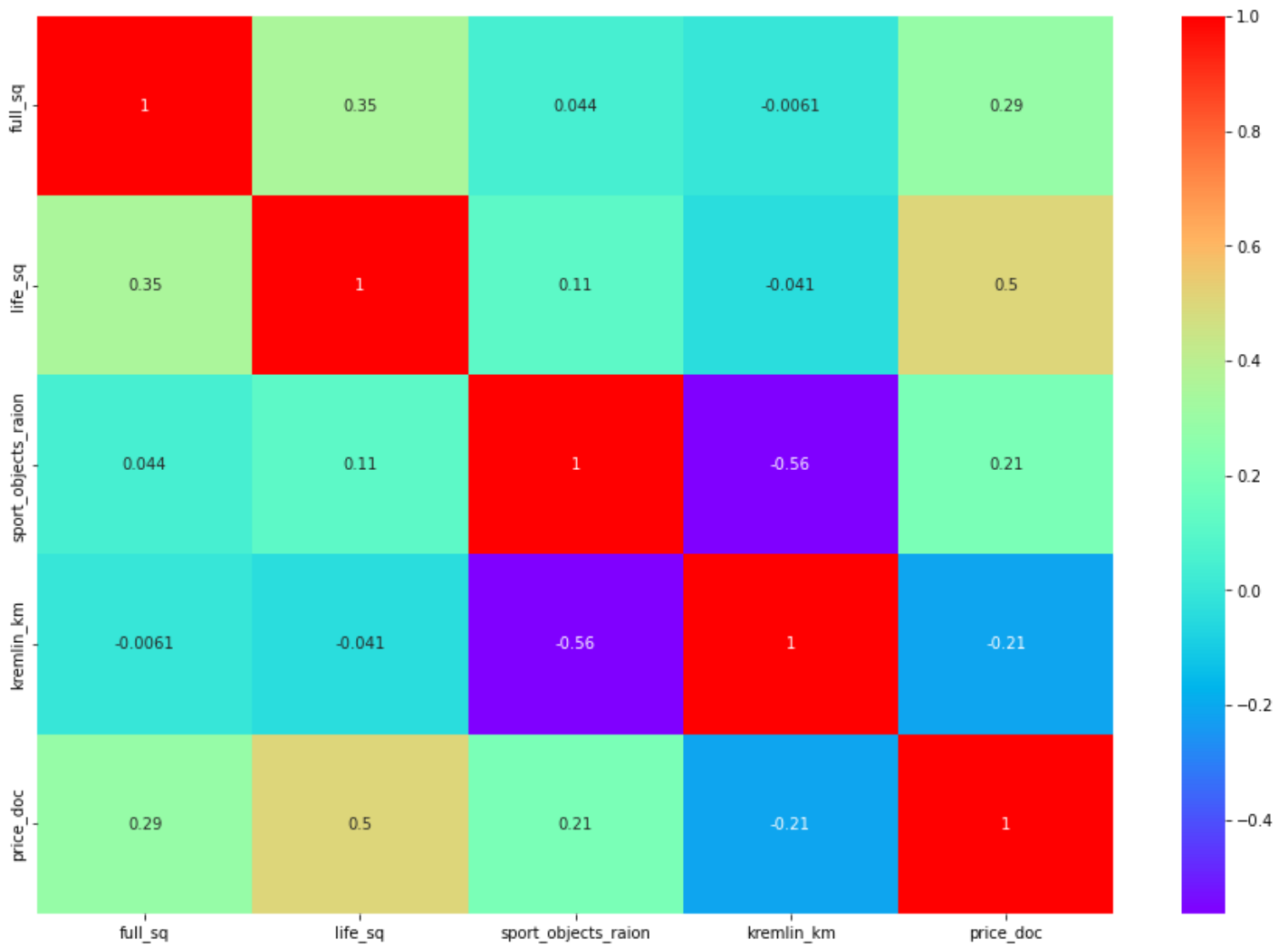
13652 rows × 5 columns

In [16]:

```
correlation = df_cut.corr()
fig = plt.figure()
axes = fig.add_axes([0,0,2,2])
sns.heatmap(correlation, annot=True, cmap='rainbow')
```

Out[16]:

```
<Axes:>
```



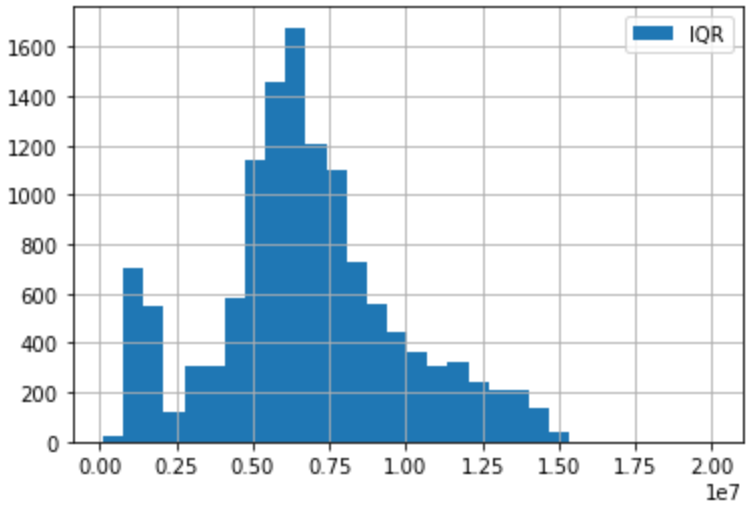
In [17]:

```
median = df_cut.price_doc.median()
print(median)
IQR = df_cut.price_doc.quantile(0.75, interpolation='midpoint') - df_cut.price_doc.quantil
perc25 = df_cut.price_doc.quantile(0.25, interpolation='midpoint')
perc75 = df_cut.price_doc.quantile(0.75, interpolation='midpoint')
print('25-й перцентиль:{},'.format(perc25),
      '75-й перцентиль: {},'.format(perc75),
      "IQR: {}, ".format(IQR),"Гарницы выбросов: [{f}, {l}].".format(f=perc25 - 1.5*IQR,
                                                                     l=perc75+1.5*IQR))

df_cut.price_doc.loc[df_cut.price_doc.between(perc25-1.5*IQR,
                                              perc75+1.5*IQR)].hist(bins=30,
                                                                    range=(100000,20000000
                                                                    label='IQR')

plt.legend()
df_cut = df_cut.loc[df_cut.price_doc.between(perc25-1.5*IQR,
                                   perc75+1.5*IQR)]
df_cut.price_doc.describe()
```

```
6700000.0
25-й перцентиль:5250000.0, 75-й перцентиль: 9100000.0, IQR: 3850000.0,  Гарницы выбросов:
[-525000.0, 14875000.0].
```

Out[17]:

```
count    1.273600e+04
mean     6.770514e+06
std      3.054684e+06
min      1.000000e+05
25%      5.100000e+06
50%      6.500000e+06
75%      8.400000e+06
max      1.485000e+07
Name: price_doc, dtype: float64
```



In [18]:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics

X = df_cut.iloc[:,0:4].values
Y = df_cut.iloc[:,4].values

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3)
my_model = LinearRegression()
my_model.fit(X_train, Y_train)

y_pred = my_model.predict(X_test)
print(my_model.intercept_,my_model.coef_)

print('MAE:', metrics.mean_absolute_error(Y_test,y_pred))

print('MSE:', metrics.mean_squared_error(Y_test, y_pred))

print('R_2:', metrics.r2_score(Y_test,y_pred))
```

```
5988651.062510242 [  3298.70480272  51856.61088341  -5809.9270071  -71888.46582746]
MAE: 2105834.8919098536
MSE: 7993092993899.628
R_2: 0.1331818786509401
```

In [29]:

```
df.describe(include=['bool','object'])
```

Out[29]:

	sub_area	ecology
count	13652	13652
unique	83	5
top	Nekrasovka	poor
freq	585	4766

In [30]:

```
df.describe()
```

Out[30]:

	id	full_sq	life_sq	floor	preschool_quota	preschool_education_centers_raion
count	13652.000000	13652.000000	13652.000000	13652.000000	13652.000000	13652.000000
mean	15019.892616	52.851084	33.052959	6.943525	2743.145327	4.723923

	id	full_sq	life_sq	floor	preschool_quota	preschool_education_centers_raion
std	8856.988104	50.719559	19.660150	5.017495	1459.098589	1.863767
min	1.000000	1.000000	0.000000	0.000000	0.000000	1.000000
25%	7578.750000	38.000000	20.000000	3.000000	1768.000000	4.000000
50%	15198.500000	45.000000	29.000000	6.000000	2508.000000	5.000000
75%	22367.250000	61.000000	42.000000	10.000000	3494.000000	6.000000
max	30473.000000	5326.000000	637.000000	77.000000	7610.000000	10.000000

8 rows × 54 columns

In [ ]: