New York housing price analysis

In this notebook, we analyze a dataset containing housing prices in New York city based on different attributes with the aim of constructing a regression model with a subset of of these features for future housing price predictions.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

plt.rc('text', usetex=True)
plt.rc('font', family='serif')
plt.rc('xtick',labelsize=12)
plt.rc('ytick',labelsize=12)

sns.set_theme()

df_housing = pd.read_csv('NY-House-Dataset.csv')

df_housing.head()
```

Out[1]:

	BROKERTITLE	TYPE	PRICE	BEDS	BATH	PROPERTYSQFT	ADDRESS
0	Brokered by Douglas Elliman -111 Fifth Ave	Condo for sale	315000	2	2.000000	1400.0	2 E 55th St Unit 803
1	Brokered by Serhant	Condo for sale	195000000	7	10.000000	17545.0	Central Park Tower Penthouse-217 W 57th New Yo...
2	Brokered by Sowae Corp	House for sale	260000	4	2.000000	2015.0	620 Sinclair Ave
3	Brokered by COMPASS	Condo for sale	69000	3	1.000000	445.0	2 E 55th St Unit 908W33
4	Brokered by Sotheby's International Realty - E...	Townhouse for sale	55000000	7	2.373861	14175.0	5 E 64th St

There are no missing values in any of the columns. However, most columns in the dataframe are strings. But most of the string features might as well be redundant.

In [13]:
```
df_housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4801 entries, 0 to 4800
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   BROKERTITLE                  4801 non-null   object 
 1   TYPE                         4801 non-null   object 
 2   PRICE                        4801 non-null   int64  
 3   BEDS                         4801 non-null   int64  
 4   BATH                         4801 non-null   float64
 5   PROPERTYSQFT                 4801 non-null   float64
 6   ADDRESS                      4801 non-null   object 
 7   STATE                        4801 non-null   object 
 8   MAIN_ADDRESS                 4801 non-null   object 
 9   ADMINISTRATIVE_AREA_LEVEL_2  4801 non-null   object 
 10  LOCALITY                     4801 non-null   object 
 11  SUBLOCALITY                  4801 non-null   object 
 12  STREET_NAME                  4801 non-null   object 
 13  LONG_NAME                    4801 non-null   object 
 14  FORMATTED_ADDRESS            4801 non-null   object 
 15  LATITUDE                     4801 non-null   float64
 16  LONGITUDE                    4801 non-null   float64
dtypes: float64(4), int64(2), object(11)
memory usage: 637.8+ KB
```

In [14]:
```
df_housing.TYPE.unique()
```

Out[14]:
```
array(['Condo for sale', 'House for sale', 'Townhouse for sale',
       'Co-op for sale', 'Multi-family home for sale', 'For sale',
       'Contingent', 'Land for sale', 'Foreclosure', 'Pending',
       'Coming Soon', 'Mobile house for sale', 'Condop for sale'],
      dtype=object)
```

Exploratory Data Analysis

We pay particular attention to the 'PRICE' attribute and its statistics

$$\mathrm{E}\left(\mathrm{PRICE}\right) = 2.36 \times 10^6,\ \mathrm{PRICE}_{\mathrm{min}} = 2494.00,\ \mathrm{PRICE}_{\mathrm{max}} = 2.15 \times 10^9,\ \sigma = 3.14$$

There is great disparity between the minimum and maximum prices and there may be many outliers present. Importantly, we will see that prices are quite dependent on the NYC borough in which they are located as well as the community district of the particular borough.

In [15]:
```
df_housing.describe()
```

Out[15]:

	PRICE	BEDS	BATH	PROPERTYSQFT	LATITUDE	LONGITUDI
count	4.801000e+03	4801.000000	4801.000000	4801.000000	4801.000000	4801.00000(
mean	2.356940e+06	3.356801	2.373861	2184.207862	40.714227	-73.94160
std	3.135525e+07	2.602315	1.946962	2377.140894	0.087676	0.10108:
min	2.494000e+03	1.000000	0.000000	230.000000	40.499546	-74.25303:
25%	4.990000e+05	2.000000	1.000000	1200.000000	40.639375	-73.98714:
50%	8.250000e+05	3.000000	2.000000	2184.207862	40.726749	-73.94918(
75%	1.495000e+06	4.000000	3.000000	2184.207862	40.771923	-73.87063(
max	2.147484e+09	50.000000	50.000000	65535.000000	40.912729	-73.70245(

In [2]:

```
'''
The PRICE column has an extremly wide range of values from 10^3 to 10^9. For
'''
df_housing['LOG_PRICE'] = df_housing['PRICE'].apply(lambda x: np.log10(x))

'''
Another important quantity is PRICE per SQFT
'''
df_housing['PRICE_SQFT'] = df_housing.PRICE / df_housing.PROPERTYSQFT
```

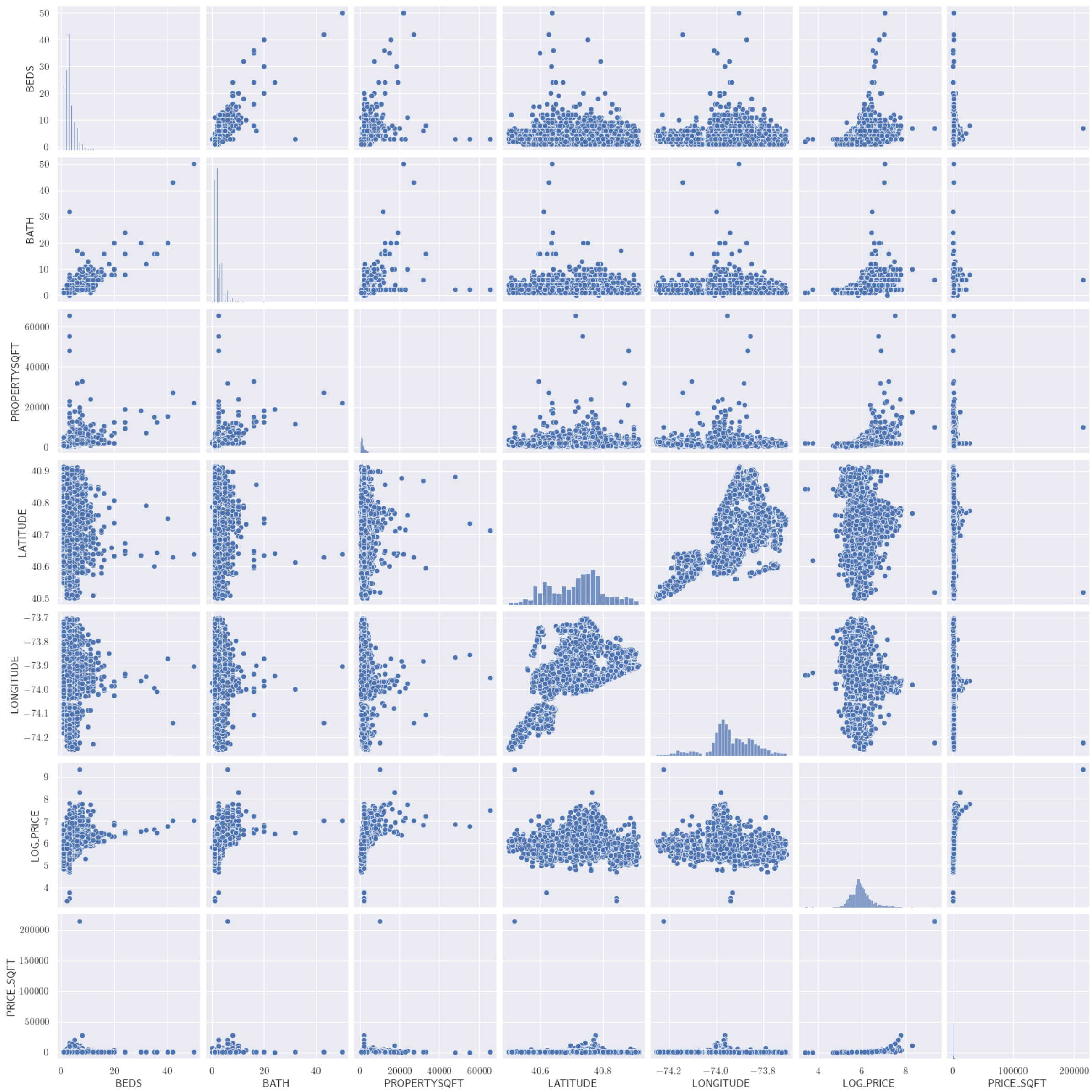
Correlations

How are the different numerical attributes correlated? We can obtain some relevant information regarding this through the scatterplot matrix (pairplot) and a correlation matrix heatmap. The most obvious linear correlation is between number of beds and baths with $r^2 = 0.78$. 'BEDS' and 'PROPERTYSQFT' also display some lienar correlation. However, it is not as strong since the properties that are the largest also have fewer beds. Furthermore, the logarithmic price has some positive correlation with all of these attributes. The relation between 'PRICE' with 'LATITUDE' and 'LONGITUDE' may as well depend on particular borough location of the property.

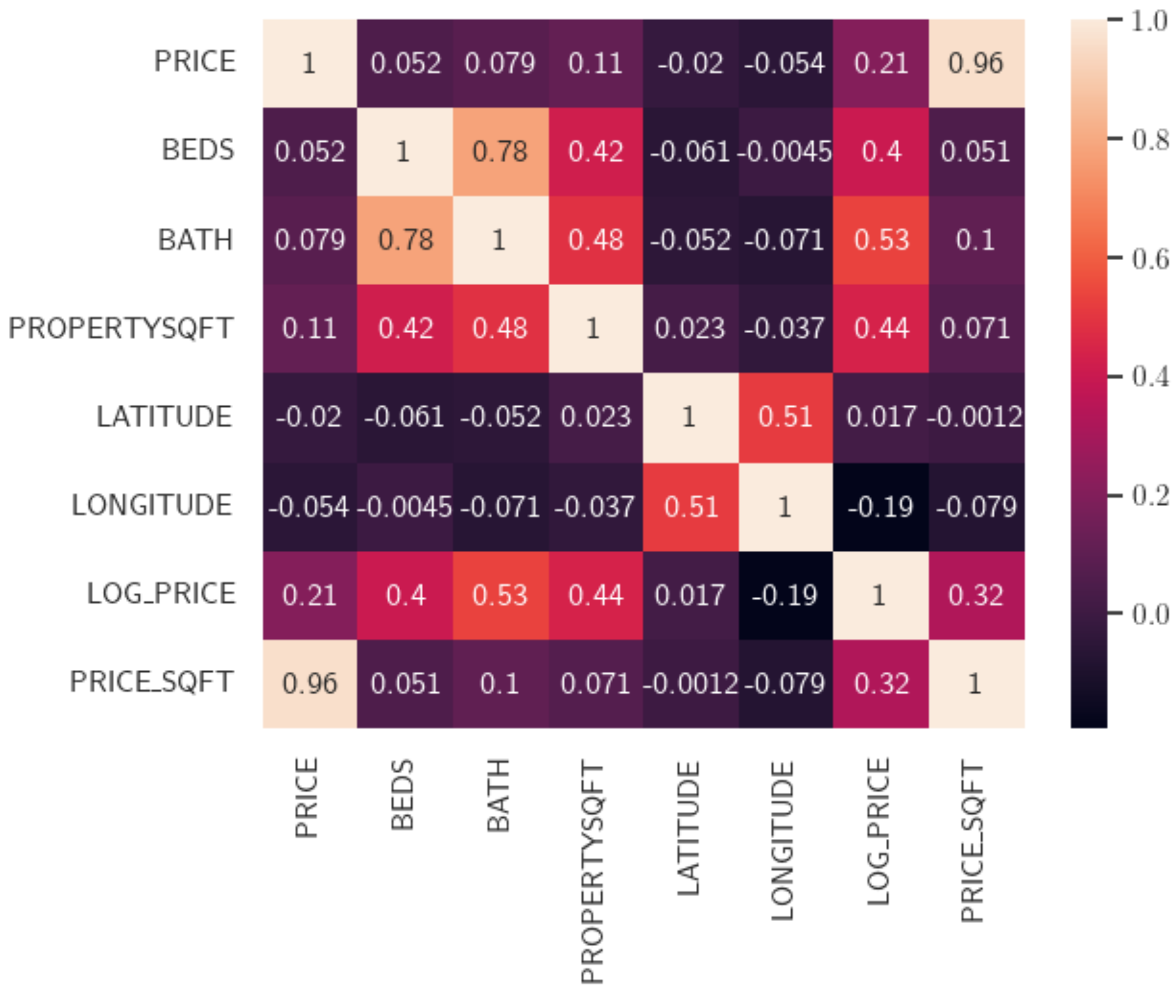
In [17]:

```
sns.pairplot( data=df_housing.iloc[:,3:], diag_kind='hist'   )
plt.show()
```

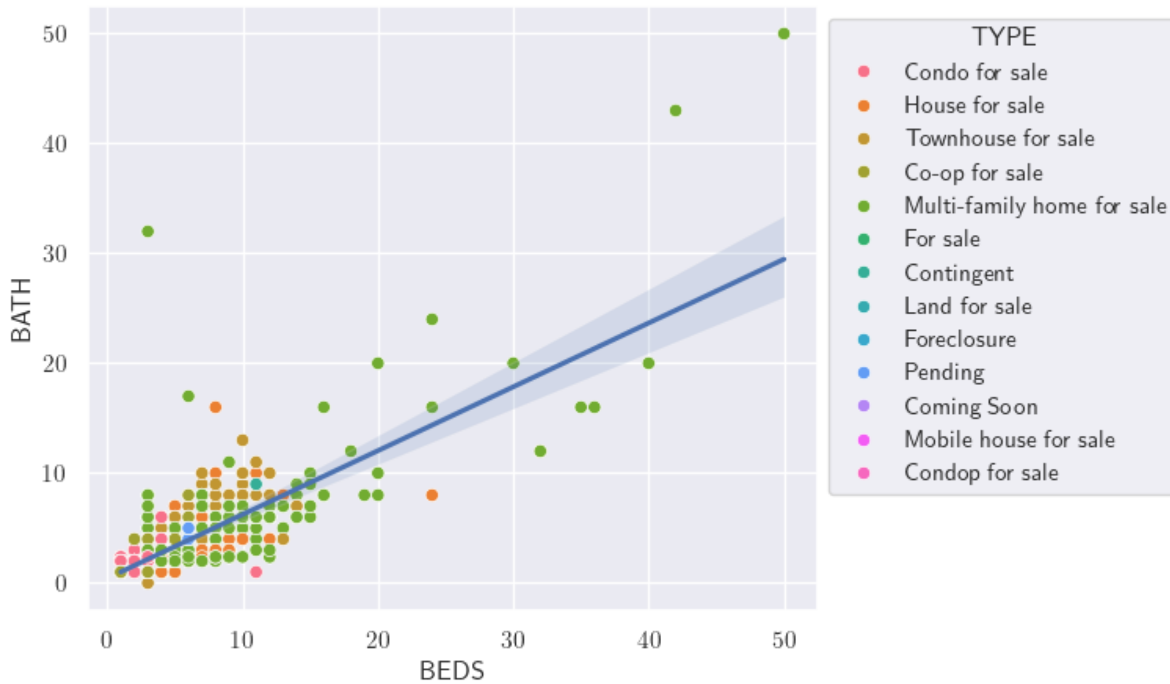
```
/Users/rafidmahbub/anaconda3/lib/python3.11/site-packages/seaborn/axisgri
d.py:123: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [3]:
```
sns.heatmap(df_housing.select_dtypes(['int64', 'float64']).corr(), annot=Tru
plt.show()
```



In [4]:
```
fig, ax = plt.subplots(figsize=(6,5))

ax = sns.scatterplot( x=df_housing.BEDS, y=df_housing.BATH, hue=df_housing.T
sns.move_legend(ax, 'upper left', bbox_to_anchor=(1,1))
sns.regplot(x=df_housing.BEDS, y=df_housing.BATH, scatter=False)
plt.show()
```



Types of properties, possible errors in data and outliers

Important statistics can be gleamed from the relationship between price and the type of property. From the dataset, we find thirteen unique types of properties. Among them, we observe that certain types of properties for sale, there are also contingent properties and certain properties for sale without descriptors and foreclosures. These types of properties without fixed description maybe problematic for machine learning purposes since they cannot be really classified under the labels of most frequently occuring properties.

In [5]:
```
df_housing.groupby('TYPE')[['PRICE', 'PROPERTYSQFT']].describe()
```

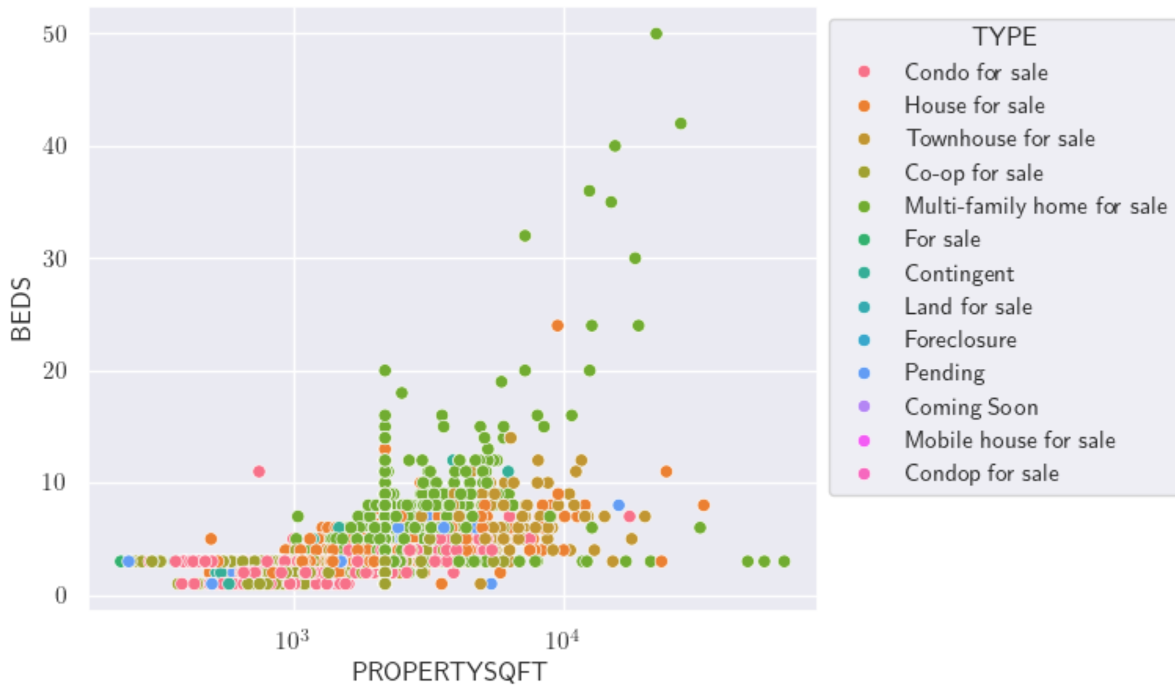
Out[5]:

	count	mean	std	min	25%	50%	
TYPE							
Co-op for sale	1450.0	1.100418e+06	3.251499e+06	49500.0	279000.00	425000.0	7990
Coming Soon	2.0	1.172000e+06	7.396337e+05	649000.0	910500.00	1172000.0	14335
Condo for sale	891.0	2.630710e+06	7.791476e+06	60000.0	575000.00	899000.0	21500
Condop for sale	5.0	9.986000e+05	2.255378e+05	598000.0	1065000.00	1080000.0	11250
Contingent	88.0	8.825717e+05	1.365915e+06	193999.0	486749.75	682450.0	854
For sale	20.0	1.954536e+06	1.875709e+06	2494.0	811000.00	1044500.0	28125
Foreclosure	14.0	1.343010e+06	2.518298e+06	249900.0	484000.00	592450.0	864
House for sale	1012.0	3.684216e+06	6.754506e+07	130000.0	659000.00	859000.0	12890
Land for sale	49.0	1.073021e+06	1.212969e+06	5800.0	285000.00	650000.0	13950
Mobile house for sale	1.0	1.288000e+06	NaN	1288000.0	1288000.00	1288000.0	12880
Multi-family home for sale	727.0	1.680428e+06	2.465464e+06	250000.0	890000.00	1199000.0	16900
Pending	243.0	1.340867e+06	2.698604e+06	90000.0	544500.00	800000.0	12875
Townhouse for sale	299.0	6.365925e+06	8.368261e+06	315000.0	1231500.00	2950000.0	79500

In general, we observe an increase in the number of beds with property size. However, there are several properies with very large square footage by very little beds (mostly occuring for 'Multi-family homes'). For quite a few of these occurences, the number of baths exceeds the number of beds. For example, the property positioned at index 622 is reported to have 3 beds but 32 baths. Moreover, there are numerous properties where the number of baths is fractional, which probably were previously NaN values filled in with mean/median values. These might be examples of erroneous reporting.

In [6]:

```
fig, ax = plt.subplots(figsize=(6,5))

ax = sns.scatterplot( x=df_housing.PROPERTYSQFT, y=df_housing.BEDS, hue=df_h
sns.move_legend(ax, 'upper left', bbox_to_anchor=(1,1))
ax.set_xscale('log')
plt.show()
```



Here, we see that three properties have a great disparity between the numbers of beds and baths

1. idx = 7, # of beds = 8, # of baths = 16
2. idx = 622, # of beds = 3, # of baths = 32
3. idx = 4691, # of beds = 6, # of baths = 17

For these properties, the no. of baths will be set equal to the no. of beds.

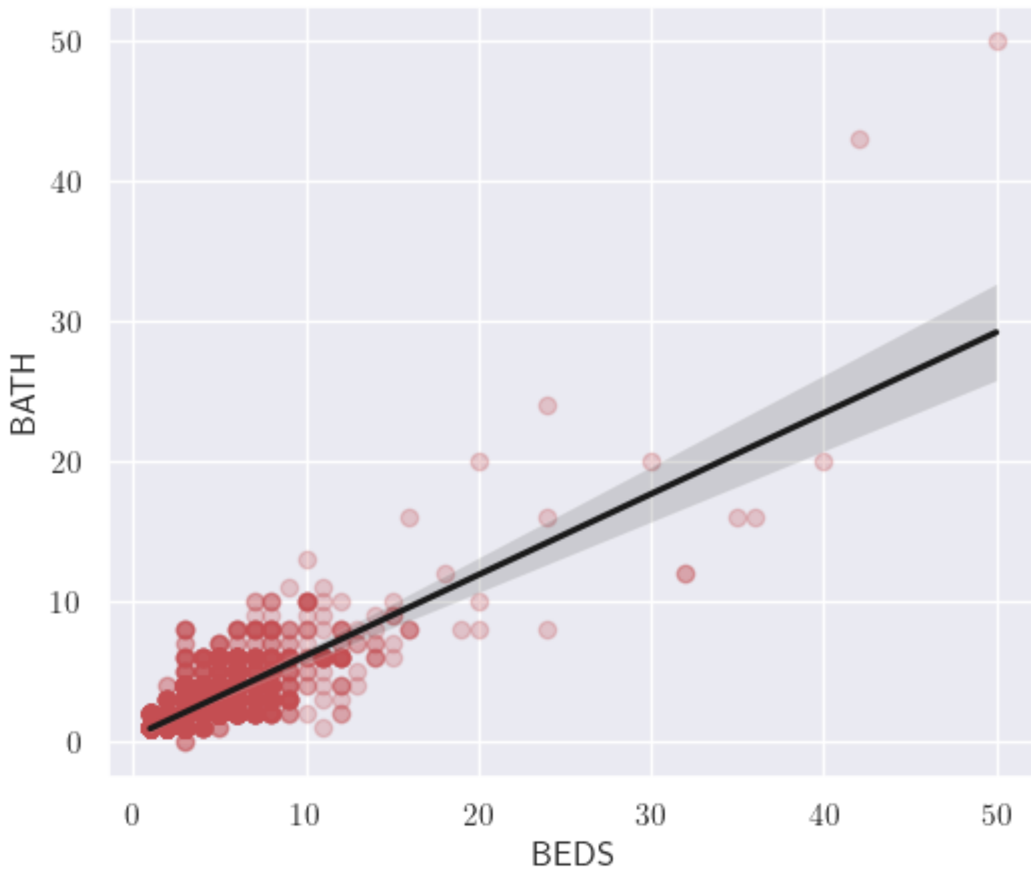
In [7]:
```
df_housing[
    (df_housing.PROPERTYSQFT >= 10**4)
    &
    (df_housing.BEDS < 10) ][['TYPE', 'BEDS', 'BATH', 'PROPERTYSQFT']]
```

Out[7]:

	TYPE	BEDS	BATH	PROPERTYSQFT
1	Condo for sale	7	10.000000	17545.0
4	Townhouse for sale	7	2.373861	14175.0
7	House for sale	8	16.000000	33000.0
69	Townhouse for sale	3	2.373861	15200.0
99	House for sale	8	8.000000	12000.0
181	Townhouse for sale	4	2.373861	10582.0
304	House for sale	7	6.000000	10000.0
601	Townhouse for sale	7	9.000000	12300.0
622	Multi-family home for sale	3	32.000000	11760.0
823	Multi-family home for sale	3	2.373861	48000.0
917	Townhouse for sale	7	2.373861	20000.0
969	House for sale	7	6.000000	11250.0
972	House for sale	7	6.000000	11250.0
1063	House for sale	7	8.000000	10100.0
1295	Multi-family home for sale	3	2.373861	12200.0
1520	Townhouse for sale	9	2.373861	10500.0
1733	House for sale	4	4.000000	10000.0
1823	Townhouse for sale	5	4.000000	17860.0
2054	Multi-family home for sale	3	2.373861	21000.0
2107	Pending	8	10.000000	16000.0
2146	Multi-family home for sale	3	2.373861	55300.0
2148	Multi-family home for sale	3	2.373861	55300.0
2171	Townhouse for sale	4	2.373861	13000.0
2932	House for sale	3	2.373861	23027.0
3007	Townhouse for sale	8	6.000000	10940.0
3073	Townhouse for sale	7	5.000000	12000.0
3097	Townhouse for sale	7	5.000000	12000.0
3130	Multi-family home for sale	6	6.000000	32000.0
4353	Multi-family home for sale	3	2.373861	17000.0
4623	Multi-family home for sale	3	2.373861	65535.0
4691	Multi-family home for sale	6	17.000000	12733.0

In [14]:
```
df_housing.loc[[7, 622, 4691], 'BATH'] = df_housing.loc[[7, 622, 4691], 'BED

df_housing.loc[[7, 622, 4691]]

df_housing.BATH = df_housing.BATH.apply(np.rint) # rounding number of baths
```

In [15]:
```
fig, ax = plt.subplots(figsize=(6,5))

sns.regplot(x=df_housing.BEDS, y=df_housing.BATH, line_kws=dict(color='k'),
plt.show()
```



It appears that

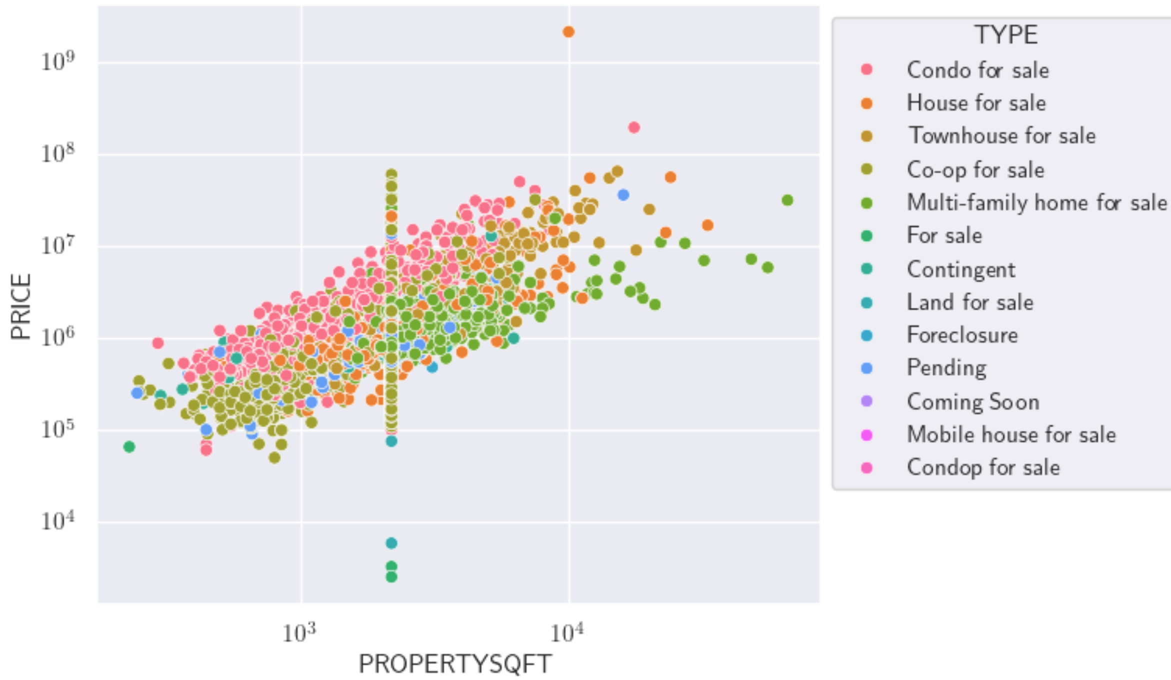
$$\log(\mathrm{PRICE}) \sim \log(\mathrm{PROPERTYSQFT})$$

The Pearson correlation between these logarithmic quantities is $r \approx 0.59$, which indicates more significant correlation compared to the base quantities. The linear correlation is worsened by the large number of points that have the same property size but increasing prices.

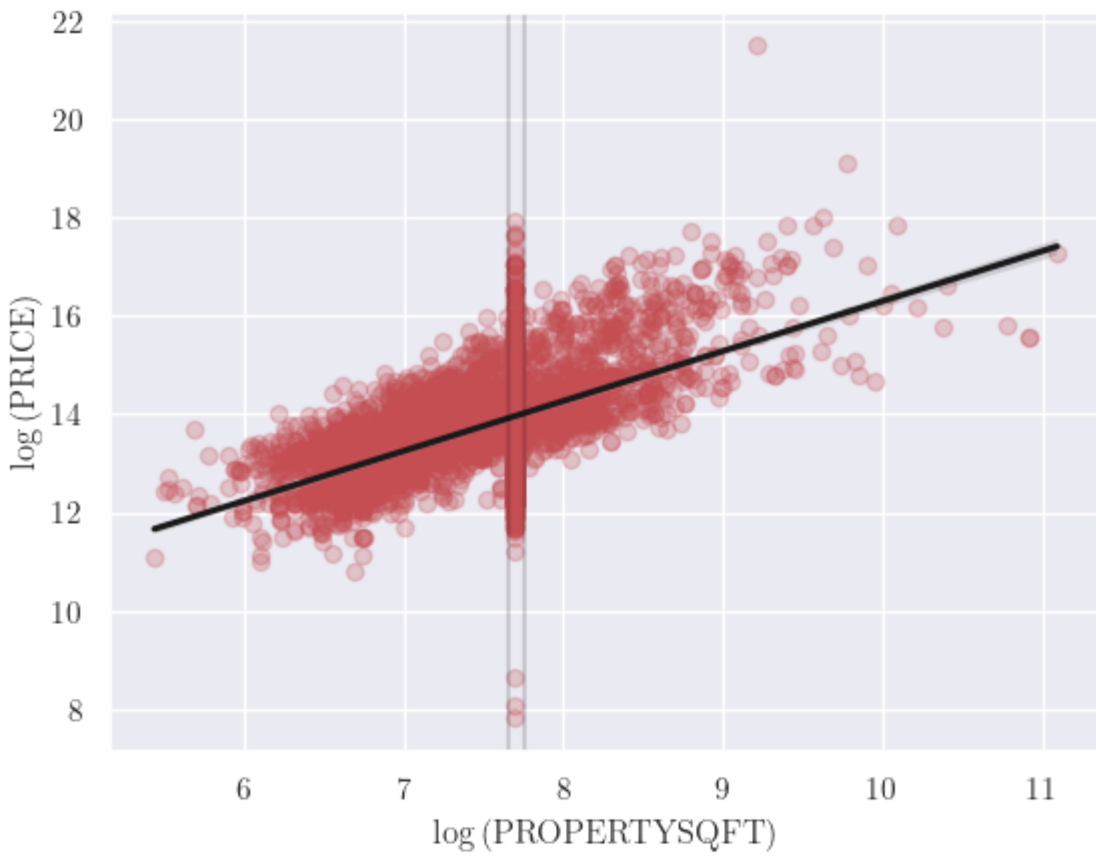
In [16]:
```
log_size = df_housing.PROPERTYSQFT.apply(np.log)
log_price = df_housing.PRICE.apply(np.log)

print('Correlation between logarithms of property size and price ' + str(log
```

```
Correlation between logarithms of property size and price 0.585257523769982
9
```

In [17]:
```
fig, ax = plt.subplots( figsize=(6,5))

ax = sns.scatterplot( x=df_housing.PROPERTYSQFT, y=df_housing.PRICE, hue=df_
sns.move_legend(ax, 'upper left', bbox_to_anchor=(1,1))
ax.set_xscale('log')
ax.set_yscale('log')

plt.show()
```



In [18]:
```
sns.regplot(x=log_size, y=log_price, fit_reg=True, line_kws=dict(color='k'),
plt.axvline(7.65, color='black', alpha=0.15)
plt.axvline(7.75, color='black', alpha=0.15)
plt.xlabel(r'$ \log\left( \rm{PROPERTYSQFT} \right) $')
plt.ylabel(r'$ \log\left( \rm{PRICE} \right) $')
plt.show()
```



There are a large number of entries (1621) that have been listed as having area of 2,184 sqft. This is clearly erroneous data, which, again, were former NaN values filled in with median square footage. However, they also represent a significant proportion of the total number of data points and cannot be safely ignored either. One way these can fixed is to isolate the subset of data and use the mean or median square footage of similar properties and replace the previous values with these. This will be done later when we include geographical data.

In [19]:
```
area = 2184.207862 # this is where these is a vertical line in the price vs

df_same_sqft = df_housing[ df_housing.PROPERTYSQFT == area ]
df_same_sqft.shape
```

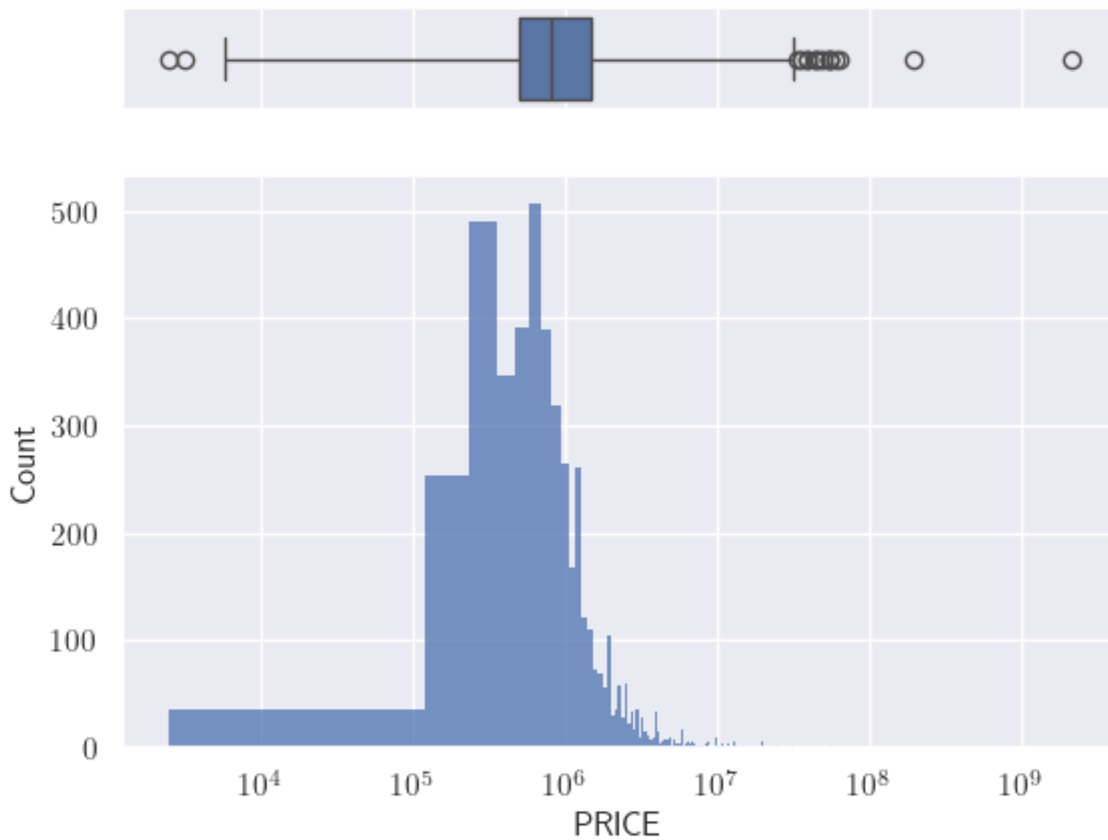
Out[19]: (1621, 19)

There are two rather big outliers in terms of price. These prices are 2.147 billion USD and 195 million USD, which are of the type 'House' and 'Condo' respectively. We will see that the most expensive one is apparently an erroneous entry. This will be fixed later. The whiskers on the boxplot here are plotted such that the outliers lie beyond the 3σ interval.

In [20]:
```
df_housing.PRICE.sort_values(ascending=False).head()
```

Out[20]:
```
304     2147483647
1        195000000
69        65000000
1075      60000000
141       56000000
Name: PRICE, dtype: int64
```

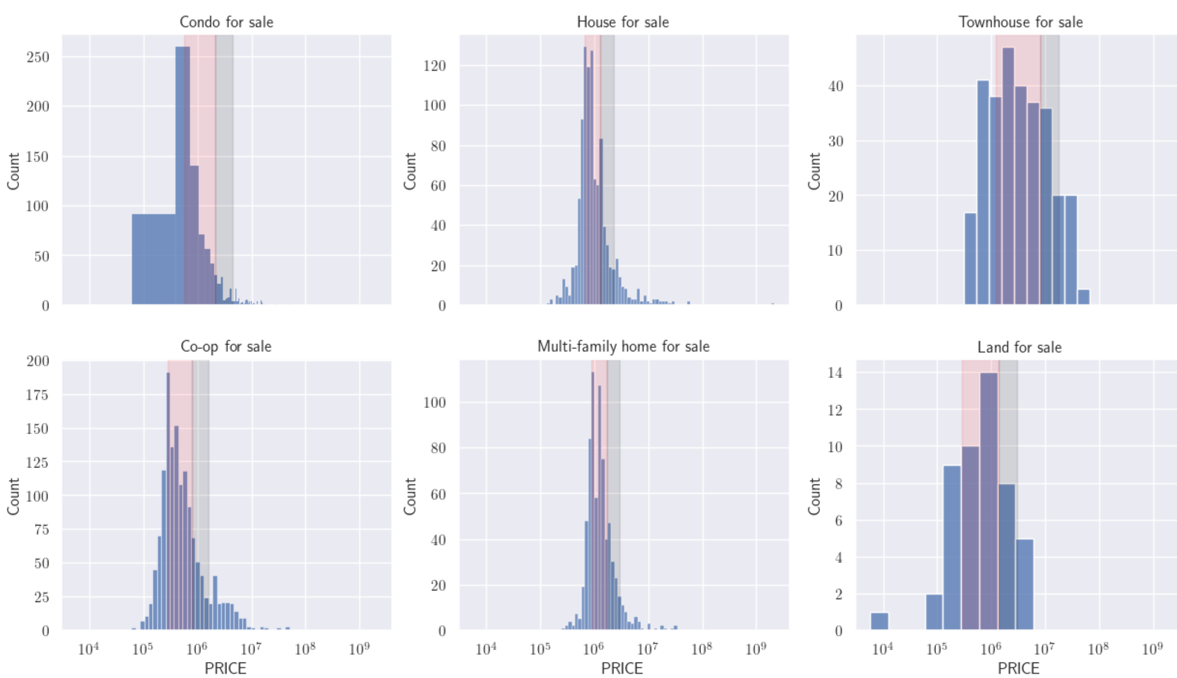
In [23]:
```
fig, (ax_box, ax_hist) = plt.subplots(2, sharex=True,
                                    gridspec_kw={"height_ratios": (.15, .85)

sns.boxplot(x=df_housing.PRICE, ax=ax_box, whis=[0.03, 99.7]) # Boxplot with
sns.histplot(x=df_housing.PRICE, ax=ax_hist)
ax_box.set_xscale('log')
ax_hist.set_xscale('log')

plt.show()
```



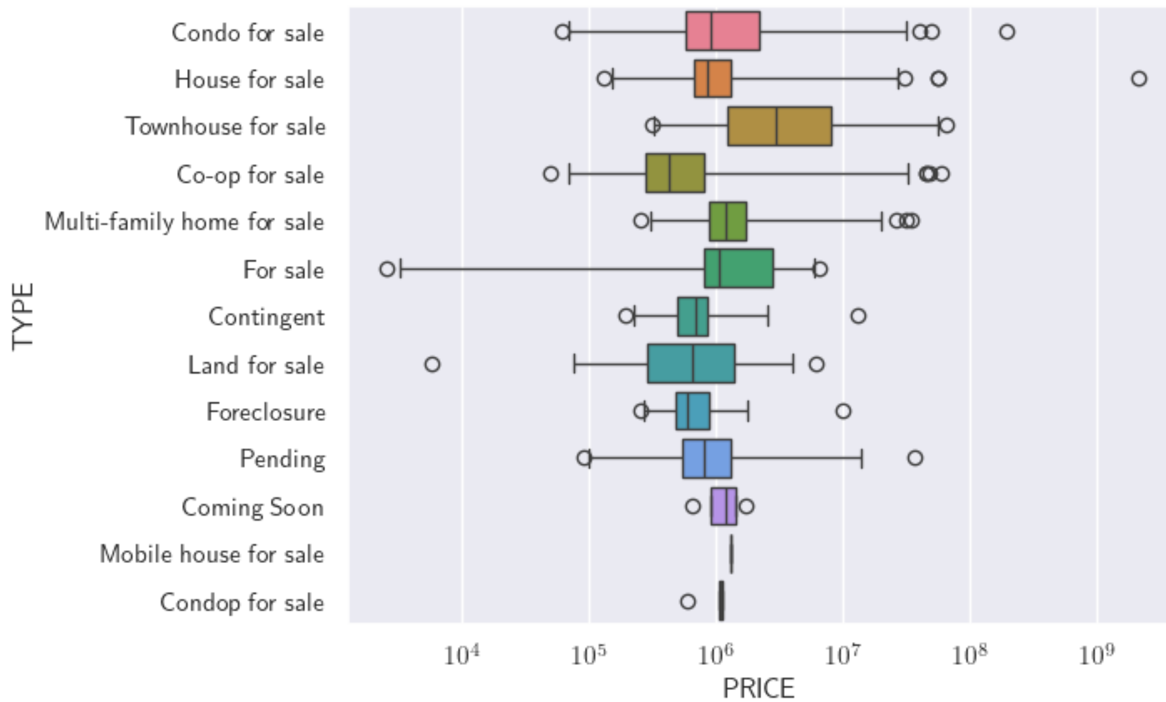
In [26]:
```
whisker_low, whisker_high = df_housing.PRICE.quantile([0.0003, 0.997])

print( 'Number of listings below 3-sigma of mean: ' + str(df_housing[ df_hou
print( 'Number of listings above 3-sigma of mean: ' + str(df_housing[ df_hou
```

```
Number of listings below 3-sigma of mean: 2
Number of listings above 3-sigma of mean: 15
```

In [29]:
```
TYPE = ['Condo for sale',
         'House for sale',
         'Townhouse for sale',
         'Co-op for sale',
         'Multi-family home for sale',
         'Land for sale']

fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(15,8), sharex=True)

for i, ax in enumerate(axes.flatten()):

    df_type = df_housing[df_housing.TYPE == TYPE[i]]
    lq, uq = df_type.PRICE.quantile([0.25, 0.75])
    iqr = uq - lq

    sns.histplot( x=df_type.PRICE, bins='auto', ax=ax )
    ax.axvspan(lq, uq, color='red', alpha=0.1)
    ax.axvspan(uq, uq + 1.5*iqr, color='black', alpha=0.1)
    ax.set_xscale('log')
    ax.set_title(TYPE[i])

plt.show()
```



In [30]:
```
sns.boxplot( y=df_housing.TYPE, x = df_housing.PRICE, hue=df_housing.TYPE, w
plt.xscale('log')
```



Geography dependence

The price of property in any city is dependent on the location and the same is also true for NYC. Here we encode the price information on a map of NYC that has been segmented into boroughs and community disctricts (CDs) using geopandas. For easier interpretability, the prices have been separated into the following ranges

1. CLASS 1: $10^3 \text{ US} \leq \text{PRICE} < 10^4 \text{ USD}$
2. CLASS 2: $10^4 \text{ US} \leq \text{PRICE} < 10^5 \text{ USD}$
3. CLASS 3: $10^5 \text{ US} \leq \text{PRICE} < 10^6 \text{ USD}$
4. CLASS 4: $10^6 \text{ US} \leq \text{PRICE} < 10^7 \text{ USD}$
5. CLASS 5: $\text{PRICE} \geq 10^7 \text{ USD}$

In [31]:
```
import geopandas as gpd
from geodatasets import get_path

path_to_data = get_path('nybb') # built-in geopandas dataframe on NYC
ny_boroughs = gpd.read_file(path_to_data)
ny_CD = gpd.read_file('./Community_Districts/geo_export_50afcaa7-d801-43f3-b

ny_boroughs
```

Out[31]:

	BoroCode	BoroName	Shape_Leng	Shape_Area	geometry
0	5	Staten Island	330470.010332	1.623820e+09	MULTIPOLYGON (((970217.022 145643.332, 970227....
1	4	Queens	896344.047763	3.045213e+09	MULTIPOLYGON (((1029606.077 156073.814, 102957...
2	3	Brooklyn	741080.523166	1.937479e+09	MULTIPOLYGON (((1021176.479 151374.797, 102100...
3	1	Manhattan	359299.096471	6.364715e+08	MULTIPOLYGON (((981219.056 188655.316, 980940....
4	2	Bronx	464392.991824	1.186925e+09	MULTIPOLYGON (((1012821.806 229228.265, 101278...

In [32]:

```
ny_CD.head()
```

Out[32]:

	boro_cd	shape_area	shape_leng	geometry
0	308.0	4.560379e+07	38232.886649	POLYGON ((-73.95829 40.67983, -73.95596 40.679...
1	205.0	3.831698e+07	29443.048056	POLYGON ((-73.89138 40.86170, -73.89142 40.861...
2	311.0	1.032083e+08	51534.144746	POLYGON ((-73.97299 40.60881, -73.97296 40.608...
3	410.0	1.720774e+08	105822.376549	MULTIPOLYGON (((-73.85722 40.65028, -73.85902 ...
4	164.0	3.831238e+07	32721.097627	POLYGON ((-73.94923 40.79687, -73.94942 40.796...

In [33]:

```
# Gathering the coordinates (longitude/latitude) of all the CDs

ny_CD['coord'] = ny_CD['geometry'].apply( lambda x: x.representative_point()

ny_CD['coord'] = [coords[0] for coords in ny_CD.coord]

ny_CD['CD'] = ny_CD.boro_cd.astype(np.int64).astype(str)

ny_CD.head()
```

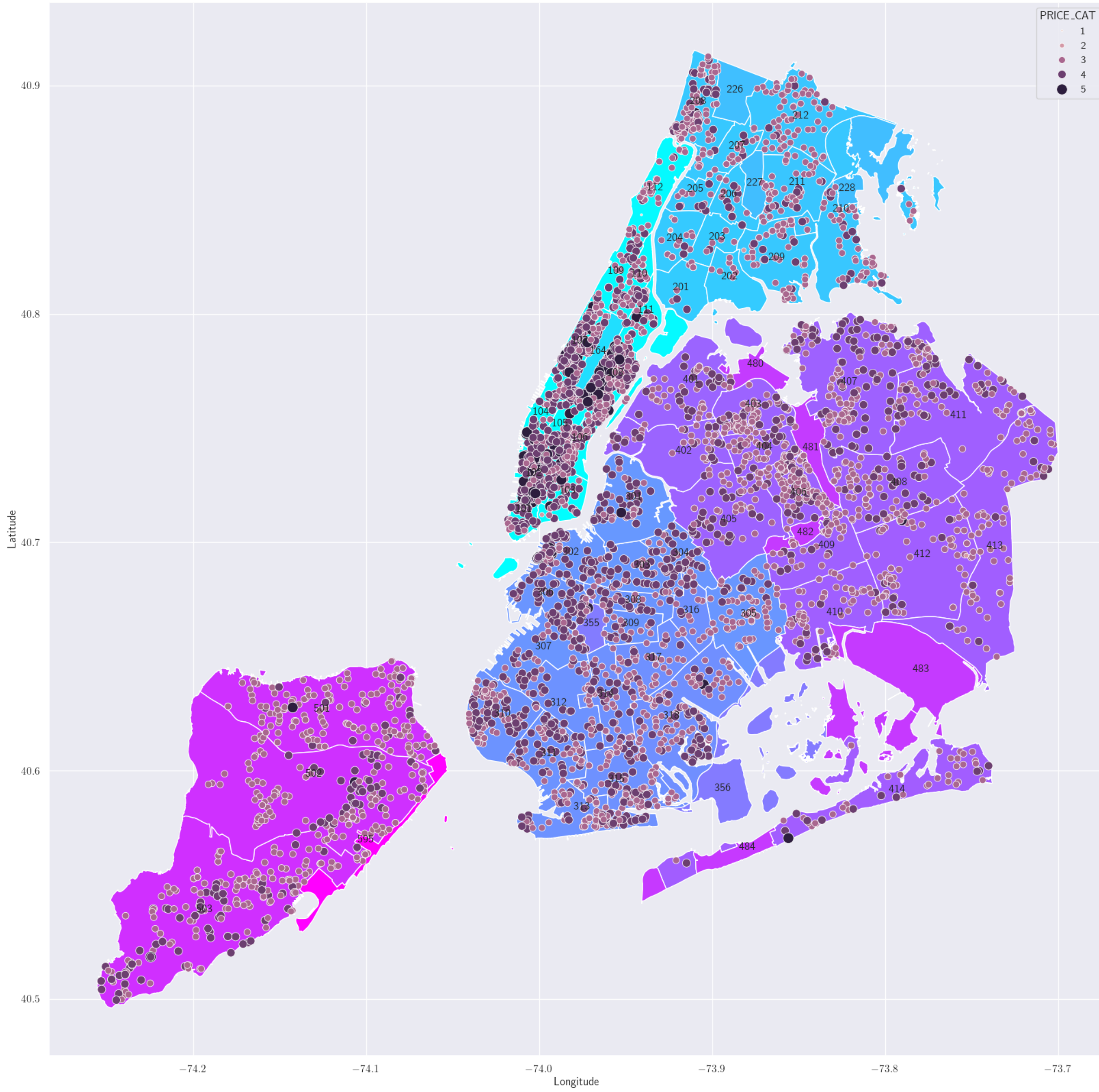
Out[33]:

	boro_cd	shape_area	shape_leng	geometry	coord	CD
0	308.0	4.560379e+07	38232.886649	POLYGON ((-73.95829 40.67983, -73.95596 40.679...	(-73.94545913768121, 40.67338767399892)	308
1	205.0	3.831698e+07	29443.048056	POLYGON ((-73.89138 40.86170, -73.89142 40.861...	(-73.9097177437186, 40.85365289770583)	205
2	311.0	1.032083e+08	51534.144746	POLYGON ((-73.97299 40.60881, -73.97296 40.608...	(-73.99428263554637, 40.60632372934502)	311
3	410.0	1.720774e+08	105822.376549	MULTIPOLYGON (((-73.85722 40.65028, -73.85902 ...	(-73.82900441292807, 40.66797529939021)	410
4	164.0	3.831238e+07	32721.097627	POLYGON ((-73.94923 40.79687, -73.94942 40.796...	(-73.96557217401126, 40.782459599544914)	164

In [59]:

```
ny_boroughs['area'] = ny_boroughs.area
ny_boroughs = ny_boroughs.set_geometry('geometry').to_crs('EPSG:4326')

price = df_housing['PRICE']

def price_categorized(x):
    if 10**3 <= x < 10**4:
        return 1
    elif 10**4 <= x < 10**5:
        return 2
    elif 10**5 <= x < 10**6:
        return 3
    elif 10**6 <= x < 10**7:
        return 4
    else:
        return 5
price_cat = price.apply( price_categorized )
df_housing['PRICE_CAT'] = price_cat


fig, ax = plt.subplots(figsize=(20,20))
cmap = plt.get_cmap('jet')

ny_CD.plot(ax=ax, column='boro_cd', edgecolor='white', cmap='cool')
sns.scatterplot(x=df_housing.LONGITUDE, y=df_housing.LATITUDE, hue=df_housir

for index, row in ny_CD.iterrows():
    plt.annotate( text=row['CD'], xy=row['coord'], horizontalalignment='cent

ax.set_xlabel(r'Longitude')
ax.set_ylabel(r'Latitude')
plt.show()
```

```
/var/folders/tn/5r78nf7j5lq1jcy9q50tq8dw0000gn/T/ipykernel_6154/422283535.p
y:1: UserWarning: Geometry is in a geographic CRS. Results from 'area' are
likely incorrect. Use 'GeoSeries.to_crs()' to re-project geometries to a pr
ojected CRS before this operation.

  ny_boroughs['area'] = ny_boroughs.area
```



As expected, the price distribution is highly dependent on the borough in which the property is located. Manhattan, Brooklyn an Queens have a similar total number of listings. However, the most expensive ones are located in Manhattan. On the otherhand, however, the data also shows that the cheapest listings are also located in Manhattan. These are obviously erroneous additions and might mean monthly rent and not the property price.

In [51]:
```
from shapely.geometry import Point

'''
Create a geopandas Point object using 'LATITUDE' and 'LONGITUDE' informatior
property is in.
'''
geo_points = df_housing.apply( lambda row: Point(row.LONGITUDE, row.LATITUDE
df_housing['GEO_POINTS'] = geo_points

df_housing.head()
```

Out[51]:

	BROKERTITLE	TYPE	PRICE	BEDS	BATH	PROPERTYSQFT	ADDRESS	
0	Brokered by Douglas Elliman -111 Fifth Ave	Condo for sale	315000	2	2.0	1400.0	2 E 55th St Unit 803	Ne NY
1	Brokered by Serhant	Condo for sale	195000000	7	10.0	17545.0	Central Park Tower Penthouse-217 W 57th New Yo...	Ne NY
2	Brokered by Sowae Corp	House for sale	260000	4	2.0	2015.0	620 Sinclair Ave	Isla
3	Brokered by COMPASS	Condo for sale	69000	3	1.0	445.0	2 E 55th St Unit 908W33	Man NY
4	Brokered by Sotheby's International Realty - E...	Townhouse for sale	55000000	7	2.0	14175.0	5 E 64th St	Ne NY

5 rows × 21 columns

In [52]:

```
def point_to_borough(point):
    check = point.within(ny_boroughs.geometry)
    idx = check[check].index[0]

    return ny_boroughs.BoroName.loc[idx]

def point_to_CD(point):
    check = point.within(ny_CD.geometry)
    idx = check[check].index[0]

    return ny_CD.boro_cd.loc[idx]

df_housing['BOROUGH'] = df_housing.GEO_POINTS.apply( point_to_borough )
df_housing['COMMUNITY_DISTRICT'] = df_housing.GEO_POINTS.apply( point_to_CD
```

In [53]:

```
df_housing['COMMUNITY_DISTRICT'] = df_housing['COMMUNITY_DISTRICT'].astype(r
df_housing.head()
```

Out[53]:

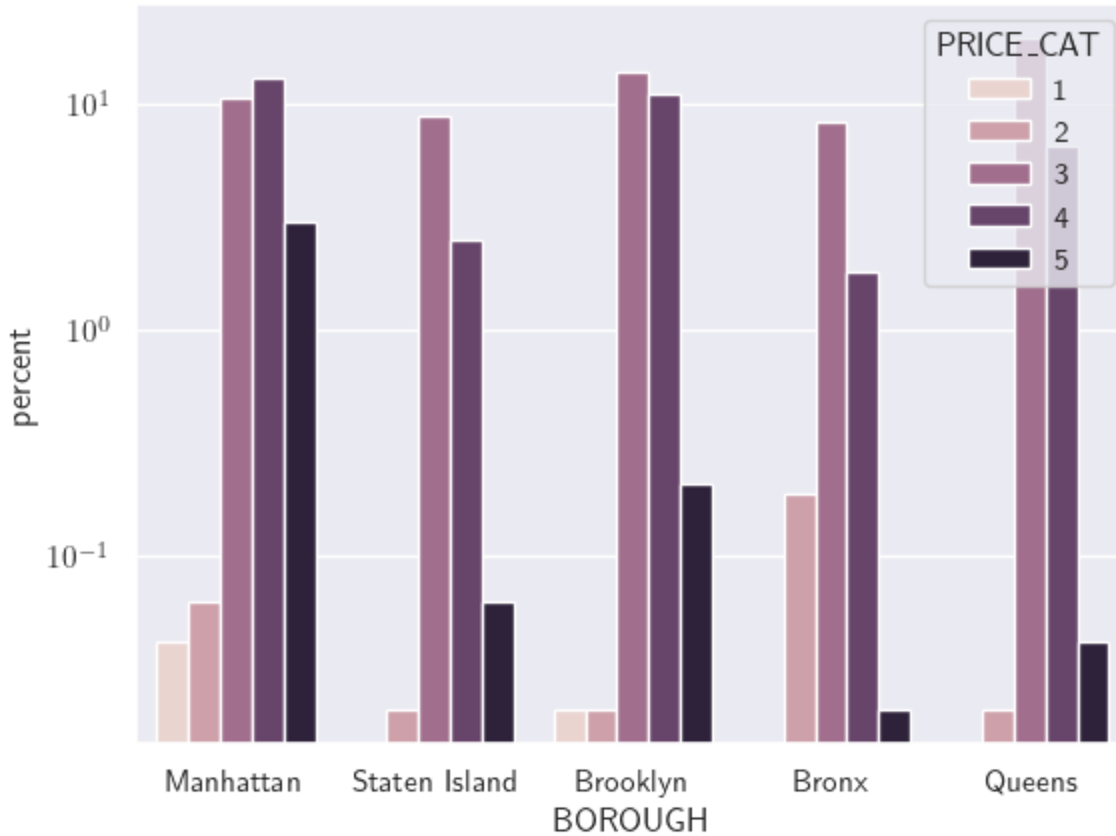
	BROKERTITLE	TYPE	PRICE	BEDS	BATH	PROPERTYSQFT	ADDRESS	
0	Brokered by Douglas Elliman -111 Fifth Ave	Condo for sale	315000	2	2.0	1400.0	2 E 55th St Unit 803	Ne NY
1	Brokered by Serhant	Condo for sale	195000000	7	10.0	17545.0	Central Park Tower Penthouse-217 W 57th New Yo...	Ne NY
2	Brokered by Sowae Corp	House for sale	260000	4	2.0	2015.0	620 Sinclair Ave	Isla
3	Brokered by COMPASS	Condo for sale	69000	3	1.0	445.0	2 E 55th St Unit 908W33	Man NY
4	Brokered by Sotheby's International Realty - E...	Townhouse for sale	55000000	7	2.0	14175.0	5 E 64th St	Ne NY

5 rows × 23 columns

Here we see a borough and community district breakdown of listings in the five price categories that have been created. Unsurprisingly, Manhattan contains the highest proportion of listings that are priced greater than 10 million USD. In particular, CD #108 appears to be the most expensive in Manhattan. This makes sense, since the `Upper East Side' falls within this community district.

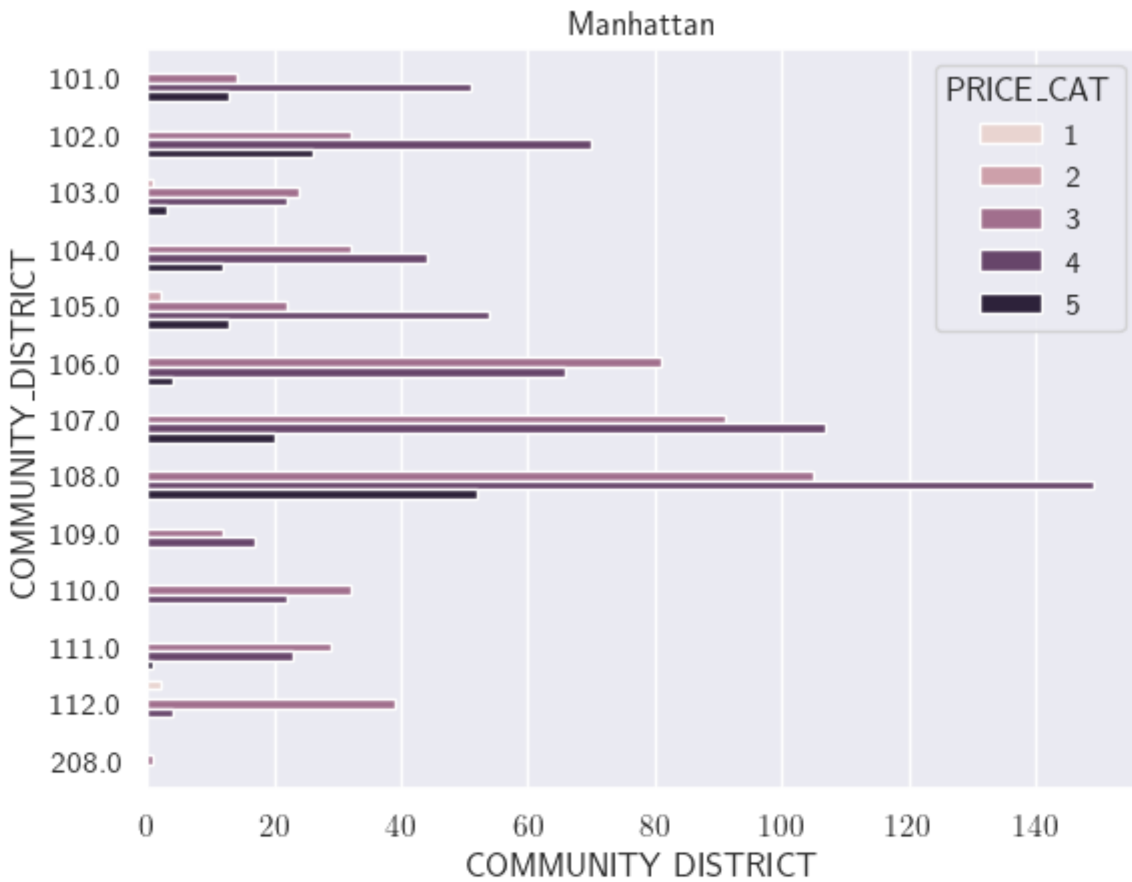
In [54]:

```
sns.countplot( x=df_housing.BOROUGH, hue=df_housing.PRICE_CAT, stat='percent
plt.yscale('log')
plt.show()
```



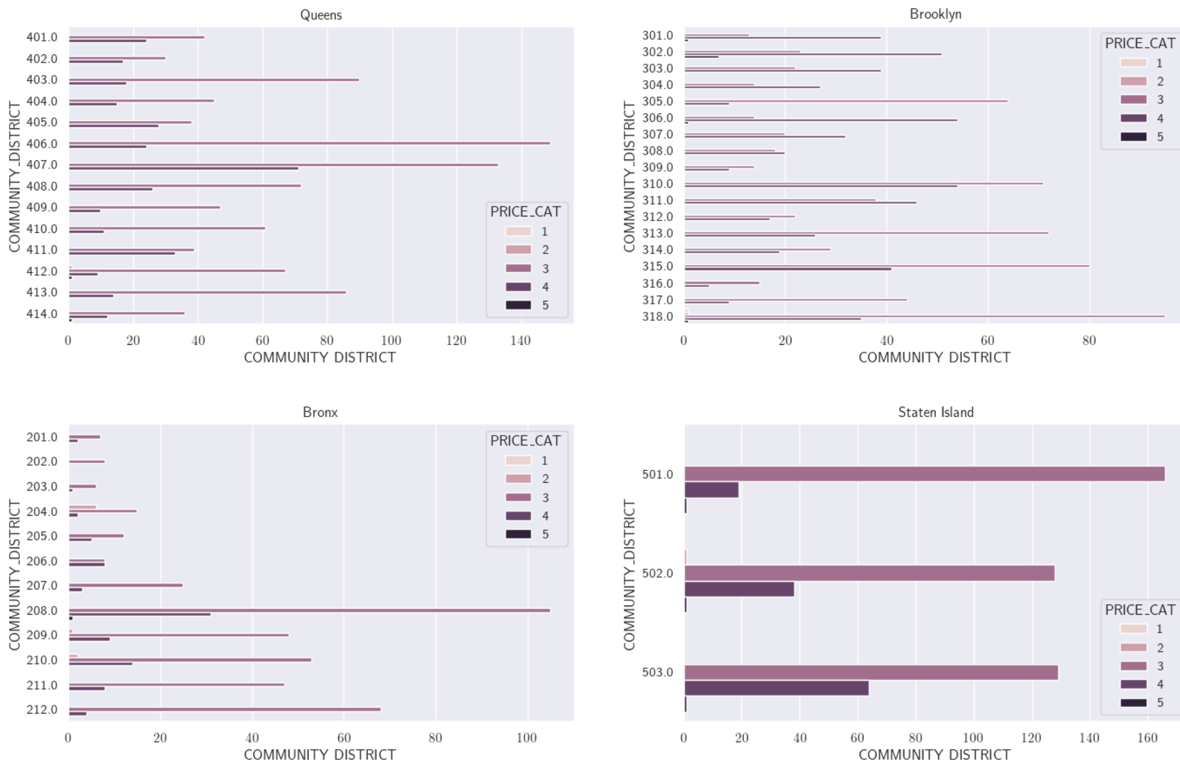
In [55]:

```
def boroughCD(df, borough_name):
    df_borough = df[ df['BOROUGH'] == borough_name ]
    return df_borough

df_MHT = boroughCD(df_housing, 'Manhattan')

sns.countplot(y=df_MHT.COMMUNITY_DISTRICT, hue=df_housing.PRICE_CAT )
plt.xlabel(r'COMMUNITY DISTRICT')
plt.title('Manhattan')
plt.show()
```



In [56]:

```
df_QUEENS = boroughCD(df_housing, 'Queens')
df_BKLN = boroughCD(df_housing, 'Brooklyn')
df_BRNX = boroughCD(df_housing, 'Bronx')
df_STAT = boroughCD(df_housing, 'Staten Island')

fig, ax = plt.subplots(2,2, figsize=(15,10))
fig.tight_layout(pad=5)

sns.countplot(ax=ax[0,0], y=df_QUEENS.COMMUNITY_DISTRICT, hue=df_housing.PRI
ax[0,0].set_xlabel(r'COMMUNITY DISTRICT')
ax[0,0].set_title(r'Queens')

sns.countplot(ax=ax[0,1], y=df_BKLN.COMMUNITY_DISTRICT, hue=df_housing.PRICE
ax[0,1].set_xlabel(r'COMMUNITY DISTRICT')
ax[0,1].set_title(r'Brooklyn')

sns.countplot(ax=ax[1,0], y=df_BRNX.COMMUNITY_DISTRICT, hue=df_housing.PRICE
ax[1,0].set_xlabel(r'COMMUNITY DISTRICT')
ax[1,0].set_title(r'Bronx')

sns.countplot(ax=ax[1,1], y=df_STAT.COMMUNITY_DISTRICT, hue=df_housing.PRICE
ax[1,1].set_xlabel(r'COMMUNITY DISTRICT')
ax[1,1].set_title(r'Staten Island')

plt.show()
```



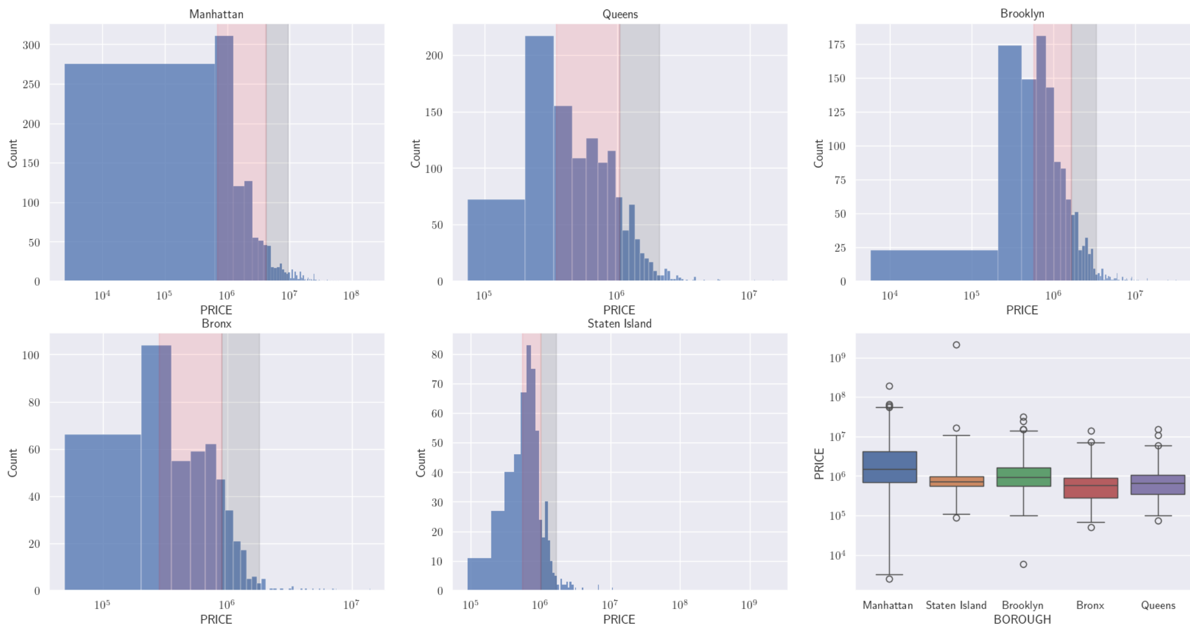
In [60]:
```
df_housing.groupby('BOROUGH').describe()['PRICE']
```

Out[60]:

BOROUGH	count	mean	std	min	25%	50%	75%
Bronx	499.0	7.583150e+05	1.008273e+06	49500.0	282500.0	590000.0	894500.0
Brooklyn	1212.0	1.431187e+06	1.904929e+06	5800.0	567250.0	949000.0	1650000.0
Manhattan	1292.0	4.277375e+06	8.805249e+06	2494.0	699000.0	1495000.0	4146250.0
Queens	1250.0	8.228747e+05	8.032906e+05	75000.0	349000.0	668000.0	1044250.0
Staten Island	548.0	4.831589e+06	9.170286e+07	90000.0	549000.0	734900.0	989000.0

In [61]:
```
def price_histplot_borough(df, borough_name, ax):
    df_borough = df[ df.BOROUGH == borough_name ]
    lq, uq = df_borough.PRICE.quantile([0.25, 0.75])
    iqr = uq - lq

    sns.histplot( x=df_borough.PRICE, ax=ax, bins='auto' )
    ax.axvspan(lq, uq, 0, 500, color='red', alpha=0.1)
    ax.axvspan(uq, uq + 1.5*iqr, 0, 500, color='black', alpha=0.1)

    ax.set_xscale('log')
    ax.set_title(borough_name)

    return(ax)
```

In [62]:
```
fig, ax = plt.subplots(2, 3, figsize=(20,10))

price_histplot_borough(df_housing, 'Manhattan', ax=ax[0,0])
price_histplot_borough(df_housing, 'Queens', ax=ax[0,1])
price_histplot_borough(df_housing, 'Brooklyn', ax=ax[0,2])
price_histplot_borough(df_housing, 'Bronx', ax=ax[1,0])
price_histplot_borough(df_housing, 'Staten Island', ax=ax[1,1])

sns.boxplot( x=df_housing.BOROUGH, y=df_housing.PRICE, hue=df_housing.BOROUG
ax[1,2].set_yscale('log')

plt.show()
```



Fixing outliers and other possibly misrepresented data

One way we can check the high price outlier listings is to use the stated addresses and cross-check with a real estate website like Zillow. We take a look at the ten most expensive listings in the dataframe. Now, a multi-billion dollar home should sound alarm bells from the start. Checking with Zillow, the 2.147 billion USD property in Staten Island is actually listed at 2.495 million USD with total square footage of 4,950.

In [52]:
```
df_housing.sort_values(by=['PRICE'], ascending=False).head(10)
```

Out[52]:

	BROKERTITLE	TYPE	PRICE	BEDS	BATH	PROPERTYSQFT	ADDRI
304	Brokered by ANNE LOPA REAL ESTATE	House for sale	2147483647	7	6.000000	10000.000000	6659-6 Amboy
1	Brokered by Serhant	Condo for sale	195000000	7	10.000000	17545.000000	Central F Tc Penthouse- W 57th I
69	Brokered by Sotheby's International Realty - E...	Townhouse for sale	65000000	3	2.373861	15200.000000	4 E 79t
1075	Brokered by COMPASS	Co-op for sale	60000000	8	8.000000	2184.207862	960 5th Uni
141	Brokered by Douglas Elliman - 575 Madison Ave	House for sale	56000000	11	10.000000	24000.000000	9 W 54t
99	Brokered by Douglas Elliman - 575 Madison Ave	House for sale	55000000	8	8.000000	12000.000000	25 River
4	Brokered by Sotheby's International Realty - E...	Townhouse for sale	55000000	7	2.373861	14175.000000	5 E 64t
626	Brokered by Nest Seekers International, Midtown	Condo for sale	50000000	6	6.000000	6569.000000	100 Vandar Apt
1453	Brokered by Corcoran East Side	Co-op for sale	48000000	5	2.373861	2184.207862	740 Park Av 4 8
3388	Brokered by Sotheby's International Realty - E...	Co-op for sale	45000000	5	2.373861	2184.207862	4 E 66th S

10 rows × 23 columns

In [63]:

```
# Correcting this entry

df_housing.loc[304, 'PRICE'] = 2495000
df_housing.loc[304, 'PROPERTYSQFT'] = 4950.000

df_housing.loc[304, :]
```

Out[63]:

```
BROKERTITLE                                  Brokered by ANNE LOPA REAL ESTATE
TYPE                                                        House for sale
PRICE                                                              2495000
BEDS                                                                     7
BATH                                                                   6.0
PROPERTYSQFT                                                        4950.0
ADDRESS                                                6659-6675 Amboy Rd
STATE                                                  New York, NY 10309
MAIN_ADDRESS                                6659-6675 Amboy RdNew York, NY 10309
ADMINISTRATIVE_AREA_LEVEL_2                                  United States
LOCALITY                                                          New York
SUBLOCALITY                                                Richmond County
STREET_NAME                                                  Staten Island
LONG_NAME                                                       Amboy Road
FORMATTED_ADDRESS                    6659 Amboy Rd, Staten Island, NY 10309, USA
LATITUDE                                                         40.518484
LONGITUDE                                                       -74.224418
LOG_PRICE                                                          9.33193
PRICE_SQFT                                                     214748.3647
PRICE_CAT                                                                5
GEO_POINTS                                 POINT (-74.2244185 40.5184841)
BOROUGH                                                      Staten Island
COMMUNITY_DISTRICT                                                     503
Name: 304, dtype: object
```

Moreover, we should also check the three least expensive listings. The first two turns out to be apartments with monthly rent information that are similar to the price listings. However, no overall property price can be found. For the the third entry, we find that its 2023 assessed price is 179,000 USD with square footage of 4,000. With these information, the lowest two price listings will be dropped from the dataframe and the the third entry below modified.

In [64]:

```
df_housing.sort_values(by=['PRICE']).head(3)
```

Out[64]:

	BROKERTITLE	TYPE	PRICE	BEDS	BATH	PROPERTYSQFT	ADDRESS	STATE	MAIN
317	Brokered by Living NY - Main Office	For sale	2494	2	1.0	2184.207862	635 W 170th St Apt 4F	New York, NY 10032	635 Apt 4
310	Brokered by Living NY - Main Office	For sale	3225	3	1.0	2184.207862	635 W 170th St Apt 2C	New York, NY 10032	635 Apt 2
360	Brokered by Century 21 Realty First	Land for sale	5800	3	2.0	2184.207862	4515 Avenue N Lot 5	Brooklyn, NY 11234	451 Lo

3 rows × 23 columns

In [65]:

```
df_housing.drop( index=[310, 317], inplace=True )

df_housing.sort_values(by=['PRICE']).head(3)
```

Out[65]:

	BROKERTITLE	TYPE	PRICE	BEDS	BATH	PROPERTYSQFT	ADDRESS	STATE	
360	Brokered by Century 21 Realty First	Land for sale	5800	3	2.0	2184.207862	4515 Avenue N Lot 5	Brooklyn, NY 11234	4
463	Brokered by Morris Park Realty Group	Co-op for sale	49500	3	2.0	800.000000	150 City Island Ave Unit E3	Bronx, NY 10464	1!
979	Brokered by COMPASS	Condo for sale	60000	3	1.0	445.000000	2 E 55th St Unit 809W35	Manhattan, NY 10022	80

3 rows × 23 columns

Now, we handle those listings that have the same 2184 sqft area. Since they represent a statistically significant proportion of the data, they cannot simply be dropped. This was also the reasoning when this dataset was created in the first place. Here, we replace the sqft value of each of these by looking at properties in the same community district and taking a mean. But before, let us see how these data points are representative of each borough.

In [66]:

```
df_same_sqft = df_housing[ df_housing.PROPERTYSQFT == area ]

df1 = df_same_sqft.groupby('BOROUGH')[['TYPE']].agg('count')
df2 = df_housing.groupby('BOROUGH')[['TYPE']].agg('count')

df_merged = df1.merge( df2, on='BOROUGH' )
df_merged.rename( columns={'TYPE_x': 'mislabeled', 'TYPE_y': 'total'}, inpla

df_merged
```

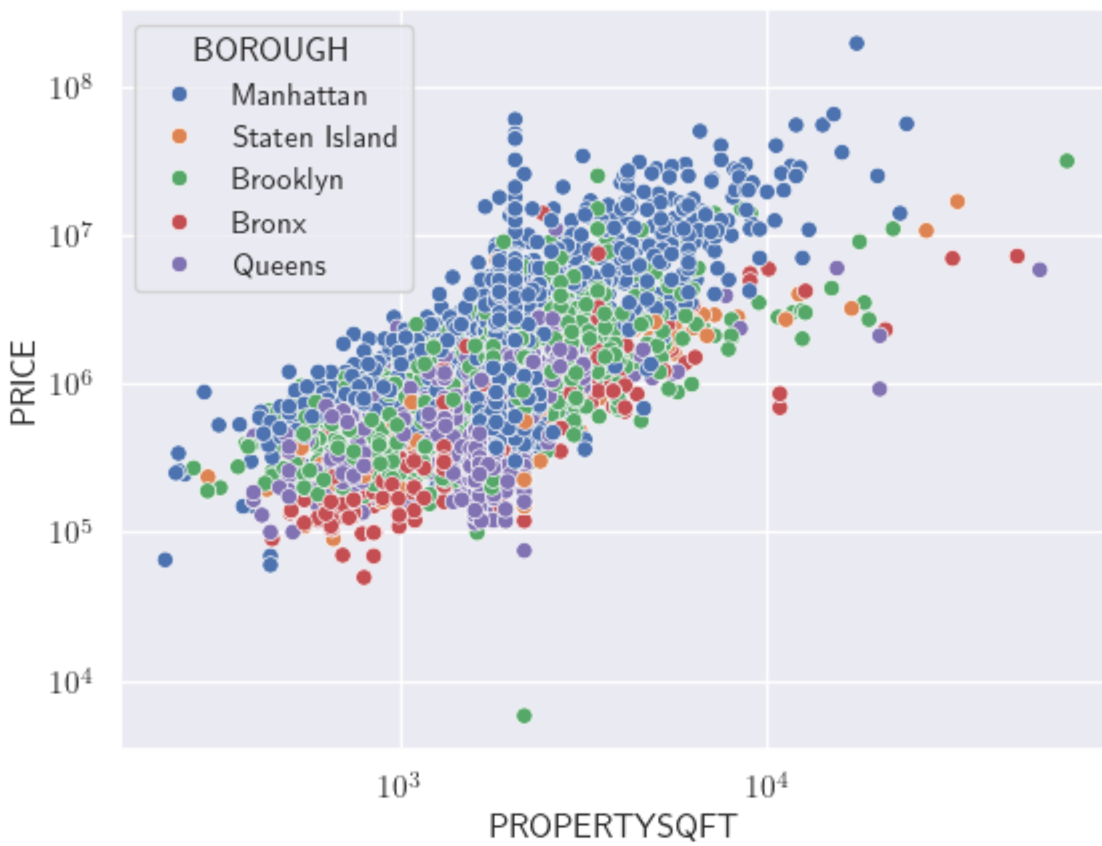
Out[66]:

BOROUGH	mislabeled	total
Bronx	95	499
Brooklyn	297	1212
Manhattan	520	1290
Queens	680	1250
Staten Island	27	548

In [70]:
```
'''
Create a function that fixes the sqft values
'''
def fix_sqft(df, area):
    df1 = df[ df['PROPERTYSQFT'] != area ]
    df_sub = df[ df['PROPERTYSQFT'] == area ] # part of dataset that has the

    df_grouped = df.groupby(['TYPE', 'COMMUNITY_DISTRICT'])[['PROPERTYSQFT']

    for index, row in df_sub.iterrows():
        cd = row['COMMUNITY_DISTRICT']
        type = row['TYPE']

        mean_sqft = df_grouped.loc[type, cd][-1]
        df_sub.loc[index, 'PROPERTYSQFT'] = mean_sqft

    return pd.concat([df1, df_sub])

df_fixed = fix_sqft(df_housing, area)
```

The function somewhat fixes the issue.

In [71]:
```
sns.scatterplot( x=df_fixed.PROPERTYSQFT, y=df_fixed.PRICE, hue=df_fixed.BOF
plt.xscale('log')
plt.yscale('log')
plt.show()
```



Regression analysis

Here we perform a regression analysis on the cleaned dataframe. We note that the did not remove the outliers which were above the 3σ interval purely because such expensive listings were not erroneous and removing them would be unwarranted. As a result, we will use tree-based regressors - (i) Random Forest and (ii) Extreme Gradient Boosted trees to regression since these generalize well in the presence of outliers. In what follows, we will drop the 'COMMUNITY_DISTRICT' column (among others) to reduce the number of categorical features, with the expectation that the longitude and latitude data will serve as suitable replacements. Furthermore, we will also use 'LOG_PRICE' in the hopes that the smaller range of values will help with training and the predicted LOG_PRICE can always be converted back to PRICE as

$$\text{PRICE} = 10^{\text{LOGPRICE}}$$

In [73]:
```
cols = ['TYPE', 'LOG_PRICE', 'BEDS', 'BATH', 'PROPERTYSQFT', 'LATITUDE', 'LO

df_housing_new = df_fixed[cols]
df_housing_new.head()
```

Out[73]:

	TYPE	LOG_PRICE	BEDS	BATH	PROPERTYSQFT	LATITUDE	LONGITUDE	BOROUGI
0	Condo for sale	5.498311	2	2.0	1400.0	40.761255	-73.974483	Manhatta
1	Condo for sale	8.290035	7	10.0	17545.0	40.766393	-73.980991	Manhatta
2	House for sale	5.414973	4	2.0	2015.0	40.541805	-74.196109	State Islan
3	Condo for sale	4.838849	3	1.0	445.0	40.761398	-73.974613	Manhatta
4	Townhouse for sale	7.740363	7	2.0	14175.0	40.767224	-73.969856	Manhatta

In [74]:
```
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer

X = df_housing_new.drop('LOG_PRICE', axis=1)
y = df_housing_new.LOG_PRICE

ct = ColumnTransformer(
    transformers=[
        ('ohe', OneHotEncoder(sparse_output=False), ['TYPE', 'BOROUGH']),
        ('standard_scaler', StandardScaler(), ['BEDS', 'BATH', 'PROPERTYSQFT
    ]
)
ct.set_output(transform='pandas')

X_scaled = ct.fit_transform(df_housing_new)
X_scaled.head()
```

Out[74]:

	ohe__TYPE_Co-op for sale	ohe__TYPE_Coming Soon	ohe__TYPE_Condo for sale	ohe__TYPE_Condop for sale	ohe__TYPE_
0	0.0	0.0	1.0	0.0	
1	0.0	0.0	1.0	0.0	
2	0.0	0.0	0.0	0.0	
3	0.0	0.0	1.0	0.0	
4	0.0	0.0	0.0	0.0	

5 rows × 23 columns

We perform regression on the scaled and encoded dataset using (i) Random Forest and (ii) Extreme Gradient Boosting (XGB) regressors. First, we implement a simple version of there without performing any cross-validation and hyperparameter tuning to check performance and calculate the R^2 and MSE values.

In [108…

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0

RF_reg = RandomForestRegressor()
RF_reg.fit(X_train, y_train)
y_pred_RF = RF_reg.predict(X_test)
rmse_RF = np.sqrt(mean_squared_error(y_test, y_pred_RF))

xgb_reg = XGBRegressor()
xgb_reg.fit(X_train, y_train)
y_pred_xgb = xgb_reg.predict(X_test)
rmse_xgb = np.sqrt(mean_squared_error(y_test, y_pred_xgb))

print( 'The R-squared score for Random Forest Regressor is ' + str(RF_reg.sc
print( 'The R-squared score for XGBoosted Regressor is ' + str(xgb_reg.score
print( 'The RMSE for Random Forest Regressor is ' + str(rmse_RF) )
print( 'The RMSE for XGBoosted Regressor is ' + str(rmse_xgb) )
```

```
The R-squared score for Random Forest Regressor is 0.8205283986967793
The R-squared score for XGBoosted Regressor is 0.8091529195177253
The RMSE for Random Forest Regressor is 0.19069411284029397
The RMSE for XGBoosted Regressor is 0.19664466890061968
```

Even with the default hyperparameters, the use of LOG*PRICE gives us very good* R^2 *scores for both Random Forest and XGB. The RMSE values are also smaller than $ |sigma*{\rm{LOGPRICE}}$, which is indicative of good predictive power of the model. We now perform cross-validation on these two Regressors with cv=5 before performing hyperparameter fine-tuning.

In [90]:
```
from sklearn.model_selection import cross_val_score

cv_score_RF = cross_val_score(RF_reg, X_train, y_train, scoring='neg_mean_sc
cv_score_xgb = cross_val_score(xgb_reg, X_train, y_train, scoring='neg_mean_

print( 'The mean RMSE for Random Forest Regression over 5-folds is ' + str(
print( 'The mean RMSE for XGBoosted Regression over 5-folds is ' + str( np.s
```

```
The mean RMSE for Random Forest Regression over 5-folds is 0.18695538007107
7
The mean RMSE for XGBoosted Regression over 5-folds is 0.19392952931284574
```

In [109…
```
from sklearn.model_selection import GridSearchCV

params_RF = [
    {
        'n_estimators' : [50, 100, 150, 200],
        'max_depth'    : [5, 10, 15]
    }
]

gs_RF = GridSearchCV(
    RandomForestRegressor(),
    param_grid = params_RF,
    scoring = ['r2', 'neg_root_mean_squared_error'],
    refit = 'r2',
    cv=5,
    verbose=4
)

gs_RF.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits
[CV 1/5] END max_depth=5, n_estimators=50; neg_root_mean_squared_error: (te
st=-0.189) r2: (test=0.799) total time=   0.1s
[CV 2/5] END max_depth=5, n_estimators=50; neg_root_mean_squared_error: (te
st=-0.229) r2: (test=0.737) total time=   0.1s
[CV 3/5] END max_depth=5, n_estimators=50; neg_root_mean_squared_error: (te
st=-0.224) r2: (test=0.755) total time=   0.1s
[CV 4/5] END max_depth=5, n_estimators=50; neg_root_mean_squared_error: (te
st=-0.197) r2: (test=0.786) total time=   0.1s
[CV 5/5] END max_depth=5, n_estimators=50; neg_root_mean_squared_error: (te
st=-0.221) r2: (test=0.773) total time=   0.1s
[CV 1/5] END max_depth=5, n_estimators=100; neg_root_mean_squared_error: (t
est=-0.188) r2: (test=0.800) total time=   0.2s
[CV 2/5] END max_depth=5, n_estimators=100; neg_root_mean_squared_error: (t
est=-0.229) r2: (test=0.736) total time=   0.2s
[CV 3/5] END max_depth=5, n_estimators=100; neg_root_mean_squared_error: (t
est=-0.223) r2: (test=0.758) total time=   0.2s
[CV 4/5] END max_depth=5, n_estimators=100; neg_root_mean_squared_error: (t
est=-0.197) r2: (test=0.788) total time=   0.2s
[CV 5/5] END max_depth=5, n_estimators=100; neg_root_mean_squared_error: (t
est=-0.219) r2: (test=0.776) total time=   0.2s
[CV 1/5] END max_depth=5, n_estimators=150; neg_root_mean_squared_error: (t
est=-0.189) r2: (test=0.799) total time=   0.4s
[CV 2/5] END max_depth=5, n_estimators=150; neg_root_mean_squared_error: (t
est=-0.228) r2: (test=0.739) total time=   0.4s
[CV 3/5] END max_depth=5, n_estimators=150; neg_root_mean_squared_error: (t
est=-0.222) r2: (test=0.759) total time=   0.4s
[CV 4/5] END max_depth=5, n_estimators=150; neg_root_mean_squared_error: (t
est=-0.197) r2: (test=0.787) total time=   0.4s
[CV 5/5] END max_depth=5, n_estimators=150; neg_root_mean_squared_error: (t
est=-0.220) r2: (test=0.775) total time=   0.4s
[CV 1/5] END max_depth=5, n_estimators=200; neg_root_mean_squared_error: (t
est=-0.188) r2: (test=0.801) total time=   0.5s
[CV 2/5] END max_depth=5, n_estimators=200; neg_root_mean_squared_error: (t
est=-0.228) r2: (test=0.739) total time=   0.5s
[CV 3/5] END max_depth=5, n_estimators=200; neg_root_mean_squared_error: (t
est=-0.222) r2: (test=0.760) total time=   0.5s
[CV 4/5] END max_depth=5, n_estimators=200; neg_root_mean_squared_error: (t
est=-0.196) r2: (test=0.789) total time=   0.5s
[CV 5/5] END max_depth=5, n_estimators=200; neg_root_mean_squared_error: (t
est=-0.220) r2: (test=0.775) total time=   0.5s
[CV 1/5] END max_depth=10, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.162) r2: (test=0.852) total time=   0.2s
[CV 2/5] END max_depth=10, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.194) r2: (test=0.812) total time=   0.2s
[CV 3/5] END max_depth=10, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.188) r2: (test=0.827) total time=   0.2s
[CV 4/5] END max_depth=10, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.158) r2: (test=0.862) total time=   0.2s
[CV 5/5] END max_depth=10, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.188) r2: (test=0.836) total time=   0.2s
[CV 1/5] END max_depth=10, n_estimators=100; neg_root_mean_squared_error:
(test=-0.162) r2: (test=0.852) total time=   0.4s
[CV 2/5] END max_depth=10, n_estimators=100; neg_root_mean_squared_error:
(test=-0.192) r2: (test=0.815) total time=   0.4s
[CV 3/5] END max_depth=10, n_estimators=100; neg_root_mean_squared_error:
(test=-0.190) r2: (test=0.824) total time=   0.4s
[CV 4/5] END max_depth=10, n_estimators=100; neg_root_mean_squared_error:
(test=-0.158) r2: (test=0.863) total time=   0.4s
```

```
[CV 5/5] END max_depth=10, n_estimators=100; neg_root_mean_squared_error:
(test=-0.186) r2: (test=0.839) total time=   0.4s
[CV 1/5] END max_depth=10, n_estimators=150; neg_root_mean_squared_error:
(test=-0.162) r2: (test=0.853) total time=   0.6s
[CV 2/5] END max_depth=10, n_estimators=150; neg_root_mean_squared_error:
(test=-0.192) r2: (test=0.814) total time=   0.6s
[CV 3/5] END max_depth=10, n_estimators=150; neg_root_mean_squared_error:
(test=-0.190) r2: (test=0.823) total time=   0.6s
[CV 4/5] END max_depth=10, n_estimators=150; neg_root_mean_squared_error:
(test=-0.157) r2: (test=0.865) total time=   0.6s
[CV 5/5] END max_depth=10, n_estimators=150; neg_root_mean_squared_error:
(test=-0.187) r2: (test=0.838) total time=   0.6s
[CV 1/5] END max_depth=10, n_estimators=200; neg_root_mean_squared_error:
(test=-0.162) r2: (test=0.852) total time=   0.8s
[CV 2/5] END max_depth=10, n_estimators=200; neg_root_mean_squared_error:
(test=-0.193) r2: (test=0.814) total time=   0.8s
[CV 3/5] END max_depth=10, n_estimators=200; neg_root_mean_squared_error:
(test=-0.191) r2: (test=0.823) total time=   0.8s
[CV 4/5] END max_depth=10, n_estimators=200; neg_root_mean_squared_error:
(test=-0.156) r2: (test=0.867) total time=   0.9s
[CV 5/5] END max_depth=10, n_estimators=200; neg_root_mean_squared_error:
(test=-0.187) r2: (test=0.837) total time=   0.8s
[CV 1/5] END max_depth=15, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.161) r2: (test=0.855) total time=   0.3s
[CV 2/5] END max_depth=15, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.187) r2: (test=0.824) total time=   0.3s
[CV 3/5] END max_depth=15, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.186) r2: (test=0.831) total time=   0.3s
[CV 4/5] END max_depth=15, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.153) r2: (test=0.871) total time=   0.3s
[CV 5/5] END max_depth=15, n_estimators=50; neg_root_mean_squared_error: (t
est=-0.184) r2: (test=0.842) total time=   0.3s
[CV 1/5] END max_depth=15, n_estimators=100; neg_root_mean_squared_error:
(test=-0.161) r2: (test=0.854) total time=   0.5s
[CV 2/5] END max_depth=15, n_estimators=100; neg_root_mean_squared_error:
(test=-0.188) r2: (test=0.823) total time=   0.5s
[CV 3/5] END max_depth=15, n_estimators=100; neg_root_mean_squared_error:
(test=-0.187) r2: (test=0.829) total time=   0.5s
[CV 4/5] END max_depth=15, n_estimators=100; neg_root_mean_squared_error:
(test=-0.154) r2: (test=0.870) total time=   0.5s
[CV 5/5] END max_depth=15, n_estimators=100; neg_root_mean_squared_error:
(test=-0.185) r2: (test=0.841) total time=   0.5s
[CV 1/5] END max_depth=15, n_estimators=150; neg_root_mean_squared_error:
(test=-0.160) r2: (test=0.856) total time=   0.8s
[CV 2/5] END max_depth=15, n_estimators=150; neg_root_mean_squared_error:
(test=-0.189) r2: (test=0.821) total time=   0.8s
[CV 3/5] END max_depth=15, n_estimators=150; neg_root_mean_squared_error:
(test=-0.187) r2: (test=0.829) total time=   0.8s
[CV 4/5] END max_depth=15, n_estimators=150; neg_root_mean_squared_error:
(test=-0.154) r2: (test=0.870) total time=   0.8s
[CV 5/5] END max_depth=15, n_estimators=150; neg_root_mean_squared_error:
(test=-0.183) r2: (test=0.844) total time=   0.8s
[CV 1/5] END max_depth=15, n_estimators=200; neg_root_mean_squared_error:
(test=-0.160) r2: (test=0.855) total time=   1.1s
[CV 2/5] END max_depth=15, n_estimators=200; neg_root_mean_squared_error:
(test=-0.188) r2: (test=0.822) total time=   1.1s
[CV 3/5] END max_depth=15, n_estimators=200; neg_root_mean_squared_error:
(test=-0.186) r2: (test=0.831) total time=   1.1s
[CV 4/5] END max_depth=15, n_estimators=200; neg_root_mean_squared_error:
```

```
(test=-0.154) r2: (test=0.871) total time=   1.1s
[CV 5/5] END max_depth=15, n_estimators=200; neg_root_mean_squared_error:
(test=-0.183) r2: (test=0.845) total time=   1.1s
```

Out[109]:

```
▸ GridSearchCV
▸ estimator: RandomForestRegressor
  ▸ RandomForestRegressor
```

In [114…

```
print(gs_RF.best_params_)
print(gs_RF.best_score_)
```

```
{'max_depth': 15, 'n_estimators': 200}
0.8446693444786701
```

In [116…

```
params_xgb = [
    {
        'n_estimators' : [50, 100, 150, 200],
        'max_depth'    : [5, 10, 15],
        'gamma'        : [0.01, 0.05, 0.1],
        'learning_rate': [0.01, 0.1, 1]
    }
]

gs_xgb = GridSearchCV(
    XGBRegressor(),
    param_grid = params_xgb,
    scoring = ['r2', 'neg_root_mean_squared_error'],
    refit = 'r2',
    cv=5,
    verbose=4
)

gs_xgb.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 108 candidates, totalling 540 fits
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.302) r2: (test=0.485) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.329) r2: (test=0.456) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.325) r2: (test=0.485) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.307) r2: (test=0.483) total time=
0.0s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.339) r2: (test=0.465) total time=
0.0s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.238) r2: (test=0.680) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.271) r2: (test=0.632) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.260) r2: (test=0.670) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.241) r2: (test=0.680) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.275) r2: (test=0.648) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.205) r2: (test=0.764) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.240) r2: (test=0.712) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.227) r2: (test=0.748) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.207) r2: (test=0.764) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.241) r2: (test=0.729) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.187) r2: (test=0.802) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.223) r2: (test=0.750) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.210) r2: (test=0.785) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.806) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=5, n_estimators=200;
```

```
neg_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.298) r2: (test=0.501) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.319) r2: (test=0.491) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.314) r2: (test=0.518) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.297) r2: (test=0.517) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.331) r2: (test=0.491) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.230) r2: (test=0.701) total time=
0.2s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.252) r2: (test=0.681) total time=
0.2s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.246) r2: (test=0.704) total time=
0.2s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.226) r2: (test=0.721) total time=
0.2s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.261) r2: (test=0.683) total time=
0.2s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.197) r2: (test=0.782) total time=
0.3s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.218) r2: (test=0.761) total time=
0.3s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.214) r2: (test=0.776) total time=
0.3s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.190) r2: (test=0.803) total time=
0.3s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.227) r2: (test=0.760) total time=
0.3s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.180) r2: (test=0.818) total time=
0.4s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.203) r2: (test=0.794) total time=
0.4s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.200) r2: (test=0.806) total time=
0.4s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.173) r2: (test=0.837) total time=
0.4s
```

```
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.211) r2: (test=0.792) total time=
0.4s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.298) r2: (test=0.498) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.319) r2: (test=0.489) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.315) r2: (test=0.516) total time=
0.2s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.296) r2: (test=0.519) total time=
0.2s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.330) r2: (test=0.494) total time=
0.2s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.232) r2: (test=0.696) total time=
0.3s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.252) r2: (test=0.680) total time=
0.3s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.247) r2: (test=0.703) total time=
0.3s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.224) r2: (test=0.724) total time=
0.3s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.260) r2: (test=0.687) total time=
0.3s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.199) r2: (test=0.776) total time=
0.4s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.219) r2: (test=0.759) total time=
0.4s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.217) r2: (test=0.771) total time=
0.5s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.188) r2: (test=0.806) total time=
0.4s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.225) r2: (test=0.764) total time=
0.5s
[CV 1/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.183) r2: (test=0.811) total time=
0.6s
[CV 2/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.204) r2: (test=0.791) total time=
0.6s
[CV 3/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.205) r2: (test=0.795) total time=
0.6s
[CV 4/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.172) r2: (test=0.838) total time=
```

```
0.6s
[CV 5/5] END gamma=0.01, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.209) r2: (test=0.797) total time=
0.6s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.0s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.199) r2: (test=0.802) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.188) r2: (test=0.828) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.158) r2: (test=0.863) total time=
0.0s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.196) r2: (test=0.822) total time=
0.0s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.159) r2: (test=0.858) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.817) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.183) r2: (test=0.836) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.154) r2: (test=0.870) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.832) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.158) r2: (test=0.859) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.818) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.182) r2: (test=0.838) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.153) r2: (test=0.872) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.835) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.158) r2: (test=0.859) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.818) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.182) r2: (test=0.838) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=200;
```

```
neg_root_mean_squared_error: (test=-0.153) r2: (test=0.872) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.835) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.162) r2: (test=0.852) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.819) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.866) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.820) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.162) r2: (test=0.852) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.189) r2: (test=0.820) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.821) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.866) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.820) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.162) r2: (test=0.852) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.189) r2: (test=0.820) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.821) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.866) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.820) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.162) r2: (test=0.852) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.189) r2: (test=0.820) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.821) total time=
0.1s
```

```
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.866) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.820) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.166) r2: (test=0.845) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.816) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.811) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.158) r2: (test=0.863) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.196) r2: (test=0.822) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.165) r2: (test=0.846) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.816) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.811) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.158) r2: (test=0.864) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.165) r2: (test=0.846) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.816) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.811) total time=
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.158) r2: (test=0.864) total time=
0.1s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.165) r2: (test=0.846) total time=
0.1s
[CV 2/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.816) total time=
0.1s
[CV 3/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.811) total time=
```

```
0.1s
[CV 4/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.158) r2: (test=0.864) total time=
0.2s
[CV 5/5] END gamma=0.01, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.185) r2: (test=0.806) total time=   0.0s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.212) r2: (test=0.782) total time=   0.0s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.196) r2: (test=0.790) total time=   0.0s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.214) r2: (test=0.787) total time=   0.0s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.185) r2: (test=0.806) total time=   0.0
s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0
s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.212) r2: (test=0.782) total time=   0.0
s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.196) r2: (test=0.790) total time=   0.0
s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.214) r2: (test=0.787) total time=   0.0
s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.185) r2: (test=0.806) total time=   0.0
s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0
s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.212) r2: (test=0.782) total time=   0.0
s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.196) r2: (test=0.790) total time=   0.0
s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.214) r2: (test=0.787) total time=   0.0
s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.185) r2: (test=0.806) total time=   0.0
s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0
s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.212) r2: (test=0.782) total time=   0.0
s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.196) r2: (test=0.790) total time=   0.0
s
```

```
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.214) r2: (test=0.787) total time=   0.0
s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.194) r2: (test=0.788) total time=   0.0
s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.217) r2: (test=0.765) total time=   0.0
s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.229) r2: (test=0.745) total time=   0.0
s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.202) r2: (test=0.776) total time=   0.0
s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.228) r2: (test=0.758) total time=   0.0
s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.194) r2: (test=0.788) total time=
0.0s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.217) r2: (test=0.765) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.229) r2: (test=0.745) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.202) r2: (test=0.776) total time=
0.0s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.228) r2: (test=0.758) total time=
0.0s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.194) r2: (test=0.788) total time=
0.0s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.217) r2: (test=0.765) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.229) r2: (test=0.745) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.202) r2: (test=0.776) total time=
0.0s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.228) r2: (test=0.758) total time=
0.0s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.194) r2: (test=0.788) total time=
0.0s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.217) r2: (test=0.765) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.229) r2: (test=0.745) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.202) r2: (test=0.776) total time=
```

```
0.0s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.228) r2: (test=0.758) total time=
0.0s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.196) r2: (test=0.784) total time=   0.0
s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.225) r2: (test=0.746) total time=   0.0
s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.231) r2: (test=0.740) total time=   0.0
s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=   0.0
s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.227) r2: (test=0.761) total time=   0.0
s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.196) r2: (test=0.784) total time=
0.0s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.225) r2: (test=0.746) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.231) r2: (test=0.740) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=
0.0s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.227) r2: (test=0.761) total time=
0.0s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.196) r2: (test=0.784) total time=
0.0s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.225) r2: (test=0.746) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.231) r2: (test=0.740) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=
0.0s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.227) r2: (test=0.761) total time=
0.0s
[CV 1/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.196) r2: (test=0.784) total time=
0.0s
[CV 2/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.225) r2: (test=0.746) total time=
0.0s
[CV 3/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.231) r2: (test=0.740) total time=
0.0s
[CV 4/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=200; n
```

```
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=
0.0s
[CV 5/5] END gamma=0.01, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.227) r2: (test=0.761) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.302) r2: (test=0.485) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.329) r2: (test=0.456) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.325) r2: (test=0.485) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.307) r2: (test=0.483) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=50;
neg_root_mean_squared_error: (test=-0.339) r2: (test=0.465) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.238) r2: (test=0.680) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.271) r2: (test=0.632) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.260) r2: (test=0.670) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.241) r2: (test=0.680) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.275) r2: (test=0.649) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.205) r2: (test=0.764) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.239) r2: (test=0.712) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.227) r2: (test=0.748) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.207) r2: (test=0.764) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.241) r2: (test=0.730) total time=
0.2s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.187) r2: (test=0.802) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.223) r2: (test=0.750) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.210) r2: (test=0.784) total time=
0.1s
```

```
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.806) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.223) r2: (test=0.770) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.296) r2: (test=0.504) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.317) r2: (test=0.495) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.314) r2: (test=0.520) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.296) r2: (test=0.518) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.331) r2: (test=0.492) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.230) r2: (test=0.702) total time=
0.2s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.250) r2: (test=0.686) total time=
0.2s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.246) r2: (test=0.706) total time=
0.2s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.224) r2: (test=0.724) total time=
0.2s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=100
; neg_root_mean_squared_error: (test=-0.260) r2: (test=0.687) total time=
0.2s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.196) r2: (test=0.783) total time=
0.2s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=
0.3s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.213) r2: (test=0.778) total time=
0.3s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.187) r2: (test=0.807) total time=
0.2s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=150
; neg_root_mean_squared_error: (test=-0.225) r2: (test=0.765) total time=
0.3s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.179) r2: (test=0.819) total time=
0.3s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.202) r2: (test=0.796) total time=
0.3s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.200) r2: (test=0.805) total time=
```

```
0.3s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.171) r2: (test=0.840) total time=
0.4s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=10, n_estimators=200
; neg_root_mean_squared_error: (test=-0.209) r2: (test=0.798) total time=
0.3s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.296) r2: (test=0.507) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.317) r2: (test=0.494) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.313) r2: (test=0.521) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.295) r2: (test=0.522) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.330) r2: (test=0.495) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.229) r2: (test=0.705) total time=
0.2s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.250) r2: (test=0.687) total time=
0.2s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.245) r2: (test=0.708) total time=
0.2s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.223) r2: (test=0.726) total time=
0.3s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=100
; neg_root_mean_squared_error: (test=-0.259) r2: (test=0.689) total time=
0.2s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.195) r2: (test=0.786) total time=
0.4s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.216) r2: (test=0.765) total time=
0.3s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.213) r2: (test=0.778) total time=
0.3s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.186) r2: (test=0.809) total time=
0.3s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=150
; neg_root_mean_squared_error: (test=-0.224) r2: (test=0.766) total time=
0.3s
[CV 1/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.178) r2: (test=0.820) total time=
0.3s
[CV 2/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.201) r2: (test=0.797) total time=
0.3s
[CV 3/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=200
```

```
; neg_root_mean_squared_error: (test=-0.200) r2: (test=0.805) total time=
0.3s
[CV 4/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.170) r2: (test=0.841) total time=
0.3s
[CV 5/5] END gamma=0.05, learning_rate=0.01, max_depth=15, n_estimators=200
; neg_root_mean_squared_error: (test=-0.209) r2: (test=0.798) total time=
0.3s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.163) r2: (test=0.850) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.198) r2: (test=0.804) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.189) r2: (test=0.826) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.159) r2: (test=0.862) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.195) r2: (test=0.822) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.161) r2: (test=0.854) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.194) r2: (test=0.811) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.828) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.867) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.193) r2: (test=0.826) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.161) r2: (test=0.854) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.194) r2: (test=0.811) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.828) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.867) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.193) r2: (test=0.826) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.161) r2: (test=0.854) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.194) r2: (test=0.811) total time=
0.1s
```

```
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.828) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.867) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.193) r2: (test=0.826) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.819) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.821) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.194) r2: (test=0.825) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.819) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.821) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.194) r2: (test=0.825) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.819) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.821) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.194) r2: (test=0.825) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.819) total time=
```

```
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.821) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.194) r2: (test=0.825) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.818) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.823) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.867) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.818) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.823) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.867) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.818) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.823) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.867) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=200;
```

```
neg_root_mean_squared_error: (test=-0.190) r2: (test=0.818) total time=
0.1s
[CV 3/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.191) r2: (test=0.823) total time=
0.1s
[CV 4/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.156) r2: (test=0.867) total time=
0.1s
[CV 5/5] END gamma=0.05, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.218) r2: (test=0.760) total time=   0.0s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.211) r2: (test=0.782) total time=   0.0s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.194) r2: (test=0.794) total time=   0.0s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=50; neg
_root_mean_squared_error: (test=-0.210) r2: (test=0.795) total time=   0.0s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0
s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.218) r2: (test=0.760) total time=   0.0
s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.211) r2: (test=0.782) total time=   0.0
s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.194) r2: (test=0.794) total time=   0.0
s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.210) r2: (test=0.795) total time=   0.0
s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0
s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.218) r2: (test=0.760) total time=   0.0
s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.211) r2: (test=0.782) total time=   0.0
s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.194) r2: (test=0.794) total time=   0.0
s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.210) r2: (test=0.795) total time=   0.0
s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0
s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.218) r2: (test=0.760) total time=   0.0
s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.211) r2: (test=0.782) total time=   0.0
```

```
s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.194) r2: (test=0.794) total time=   0.0
s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=5, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.210) r2: (test=0.795) total time=   0.0
s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.188) r2: (test=0.802) total time=   0.0
s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.215) r2: (test=0.767) total time=   0.0
s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.227) r2: (test=0.748) total time=   0.0
s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.191) r2: (test=0.800) total time=   0.0
s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=   0.0
s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.188) r2: (test=0.802) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.215) r2: (test=0.767) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.227) r2: (test=0.748) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.191) r2: (test=0.800) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.188) r2: (test=0.802) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.215) r2: (test=0.767) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.227) r2: (test=0.748) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.191) r2: (test=0.800) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.188) r2: (test=0.802) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.215) r2: (test=0.767) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=200; n
```

```
eg_root_mean_squared_error: (test=-0.227) r2: (test=0.748) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.191) r2: (test=0.800) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=10, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.191) r2: (test=0.794) total time=   0.0
s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.222) r2: (test=0.752) total time=   0.0
s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.228) r2: (test=0.746) total time=   0.0
s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.192) r2: (test=0.797) total time=   0.0
s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=   0.0
s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.191) r2: (test=0.794) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.222) r2: (test=0.752) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.228) r2: (test=0.746) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.797) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.191) r2: (test=0.794) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.222) r2: (test=0.752) total time=
0.0s
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.228) r2: (test=0.746) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.797) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=
0.0s
[CV 1/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.191) r2: (test=0.794) total time=
0.0s
[CV 2/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.222) r2: (test=0.752) total time=
0.0s
```

```
[CV 3/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.228) r2: (test=0.746) total time=
0.0s
[CV 4/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.797) total time=
0.0s
[CV 5/5] END gamma=0.05, learning_rate=1, max_depth=15, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.224) r2: (test=0.767) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.302) r2: (test=0.485) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.329) r2: (test=0.456) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.325) r2: (test=0.485) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.307) r2: (test=0.483) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.339) r2: (test=0.465) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.238) r2: (test=0.680) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.271) r2: (test=0.632) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.260) r2: (test=0.670) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.242) r2: (test=0.680) total time=
0.1s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=100;
neg_root_mean_squared_error: (test=-0.275) r2: (test=0.648) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.205) r2: (test=0.764) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.240) r2: (test=0.712) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.227) r2: (test=0.749) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.208) r2: (test=0.764) total time=
0.1s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=150;
neg_root_mean_squared_error: (test=-0.241) r2: (test=0.729) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.187) r2: (test=0.803) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.223) r2: (test=0.750) total time=
```

```
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.210) r2: (test=0.785) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.806) total time=
0.1s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=5, n_estimators=200;
neg_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.296) r2: (test=0.506) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.317) r2: (test=0.495) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.315) r2: (test=0.517) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.296) r2: (test=0.520) total time=
0.1s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=50;
neg_root_mean_squared_error: (test=-0.331) r2: (test=0.490) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.230) r2: (test=0.703) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.250) r2: (test=0.686) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.246) r2: (test=0.705) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.224) r2: (test=0.724) total time=
0.1s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.261) r2: (test=0.684) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.785) total time=
0.2s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.218) r2: (test=0.761) total time=
0.2s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.215) r2: (test=0.774) total time=
0.2s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.189) r2: (test=0.805) total time=
0.2s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.227) r2: (test=0.760) total time=
0.2s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.180) r2: (test=0.818) total time=
0.2s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=200;
```

```
neg_root_mean_squared_error: (test=-0.204) r2: (test=0.791) total time=
0.2s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.202) r2: (test=0.801) total time=
0.2s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.172) r2: (test=0.838) total time=
0.2s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.211) r2: (test=0.793) total time=
0.2s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.295) r2: (test=0.508) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.317) r2: (test=0.495) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.314) r2: (test=0.518) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.295) r2: (test=0.522) total time=
0.1s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=50;
neg_root_mean_squared_error: (test=-0.330) r2: (test=0.493) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.229) r2: (test=0.705) total time=
0.2s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.250) r2: (test=0.685) total time=
0.2s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.246) r2: (test=0.705) total time=
0.2s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.224) r2: (test=0.725) total time=
0.2s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.260) r2: (test=0.686) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.786) total time=
0.2s
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.218) r2: (test=0.761) total time=
0.2s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.215) r2: (test=0.774) total time=
0.2s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.188) r2: (test=0.806) total time=
0.2s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.226) r2: (test=0.763) total time=
0.2s
[CV 1/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.180) r2: (test=0.818) total time=
0.2s
```

```
[CV 2/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.204) r2: (test=0.790) total time=
0.2s
[CV 3/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.202) r2: (test=0.800) total time=
0.2s
[CV 4/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.172) r2: (test=0.838) total time=
0.3s
[CV 5/5] END gamma=0.1, learning_rate=0.01, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.210) r2: (test=0.795) total time=
0.2s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.163) r2: (test=0.850) total time=   0.0
s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.199) r2: (test=0.801) total time=   0.0
s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.189) r2: (test=0.826) total time=   0.0
s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.158) r2: (test=0.863) total time=   0.0
s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=50; ne
g_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=   0.0
s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.163) r2: (test=0.850) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.199) r2: (test=0.801) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.189) r2: (test=0.826) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.157) r2: (test=0.864) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=100; n
eg_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.163) r2: (test=0.850) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.199) r2: (test=0.801) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.189) r2: (test=0.826) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.157) r2: (test=0.864) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=150; n
eg_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.163) r2: (test=0.850) total time=
```

```
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.199) r2: (test=0.801) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.189) r2: (test=0.826) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.157) r2: (test=0.864) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=5, n_estimators=200; n
eg_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.165) r2: (test=0.847) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.193) r2: (test=0.813) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.154) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.165) r2: (test=0.847) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.193) r2: (test=0.813) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.154) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=100;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.165) r2: (test=0.847) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.193) r2: (test=0.813) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.154) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=150;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=200;
```

```
neg_root_mean_squared_error: (test=-0.165) r2: (test=0.847) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.193) r2: (test=0.813) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.154) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=10, n_estimators=200;
neg_root_mean_squared_error: (test=-0.197) r2: (test=0.819) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.816) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=50; n
eg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.816) total time=
0.0s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=100;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.0s
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.0s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.816) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.0s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.0s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=150;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.0s
```

```
[CV 1/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.164) r2: (test=0.849) total time=
0.1s
[CV 2/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.816) total time=
0.1s
[CV 3/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.192) r2: (test=0.820) total time=
0.1s
[CV 4/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.155) r2: (test=0.869) total time=
0.1s
[CV 5/5] END gamma=0.1, learning_rate=0.1, max_depth=15, n_estimators=200;
neg_root_mean_squared_error: (test=-0.195) r2: (test=0.823) total time=
0.1s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=50; neg_
root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=50; neg_
root_mean_squared_error: (test=-0.216) r2: (test=0.765) total time=   0.0s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=50; neg_
root_mean_squared_error: (test=-0.211) r2: (test=0.783) total time=   0.0s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=50; neg_
root_mean_squared_error: (test=-0.194) r2: (test=0.793) total time=   0.0s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=50; neg_
root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=100; neg
_root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=100; neg
_root_mean_squared_error: (test=-0.216) r2: (test=0.765) total time=   0.0s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=100; neg
_root_mean_squared_error: (test=-0.211) r2: (test=0.783) total time=   0.0s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=100; neg
_root_mean_squared_error: (test=-0.194) r2: (test=0.793) total time=   0.0s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=100; neg
_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=150; neg
_root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=150; neg
_root_mean_squared_error: (test=-0.216) r2: (test=0.765) total time=   0.0s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=150; neg
_root_mean_squared_error: (test=-0.211) r2: (test=0.783) total time=   0.0s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=150; neg
_root_mean_squared_error: (test=-0.194) r2: (test=0.793) total time=   0.0s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=150; neg
_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=200; neg
_root_mean_squared_error: (test=-0.181) r2: (test=0.815) total time=   0.0s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=200; neg
_root_mean_squared_error: (test=-0.216) r2: (test=0.765) total time=   0.0s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=200; neg
_root_mean_squared_error: (test=-0.211) r2: (test=0.783) total time=   0.0s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=200; neg
_root_mean_squared_error: (test=-0.194) r2: (test=0.793) total time=   0.0s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=5, n_estimators=200; neg
_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=50; neg
_root_mean_squared_error: (test=-0.184) r2: (test=0.809) total time=   0.0s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=50; neg
_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0s
```

```
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=50; neg
_root_mean_squared_error: (test=-0.223) r2: (test=0.758) total time=   0.0s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=50; neg
_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=   0.0s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=50; neg
_root_mean_squared_error: (test=-0.222) r2: (test=0.771) total time=   0.0s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.184) r2: (test=0.809) total time=   0.0
s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0
s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.223) r2: (test=0.758) total time=   0.0
s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=   0.0
s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.222) r2: (test=0.771) total time=   0.0
s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.184) r2: (test=0.809) total time=   0.0
s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0
s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.223) r2: (test=0.758) total time=   0.0
s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=   0.0
s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.222) r2: (test=0.771) total time=   0.0
s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.184) r2: (test=0.809) total time=   0.1
s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.217) r2: (test=0.764) total time=   0.0
s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.223) r2: (test=0.758) total time=   0.0
s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.192) r2: (test=0.798) total time=   0.0
s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=10, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.222) r2: (test=0.771) total time=   0.0
s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=50; neg
_root_mean_squared_error: (test=-0.185) r2: (test=0.808) total time=   0.0s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=50; neg
_root_mean_squared_error: (test=-0.218) r2: (test=0.761) total time=   0.0s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=50; neg
_root_mean_squared_error: (test=-0.225) r2: (test=0.753) total time=   0.0s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=50; neg
_root_mean_squared_error: (test=-0.188) r2: (test=0.807) total time=   0.0s
```

```
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=50; neg
_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.185) r2: (test=0.808) total time=   0.0
s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.218) r2: (test=0.761) total time=   0.0
s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.225) r2: (test=0.753) total time=   0.0
s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.188) r2: (test=0.807) total time=   0.0
s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=100; ne
g_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0
s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.185) r2: (test=0.808) total time=   0.0
s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.218) r2: (test=0.761) total time=   0.0
s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.225) r2: (test=0.753) total time=   0.0
s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.188) r2: (test=0.807) total time=   0.0
s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=150; ne
g_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0
s
[CV 1/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.185) r2: (test=0.808) total time=   0.0
s
[CV 2/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.218) r2: (test=0.761) total time=   0.0
s
[CV 3/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.225) r2: (test=0.753) total time=   0.0
s
[CV 4/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.188) r2: (test=0.807) total time=   0.0
s
[CV 5/5] END gamma=0.1, learning_rate=1, max_depth=15, n_estimators=200; ne
g_root_mean_squared_error: (test=-0.223) r2: (test=0.769) total time=   0.0
s
```

Out[116]:

```
▸ GridSearchCV
▸ estimator: XGBRegressor
  ▸ XGBRegressor
```

In [117…

```
print(gs_xgb.best_params_)
print(gs_xgb.best_score_)
```

```
{'gamma': 0.01, 'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200}
0.8443316349972807
```

After performing hyperparameter fine-tuning, we see that both regressors perform similarly, yielding $R^2 \approx 0.84$.

Conclusions

Here, we analyzed the NYC housing dataset that is available on Kaggle. After performing a thorough exploratory data analysis, where the different features where visually studied, we were able to determine the various trends and correlations present between them. For better interpretation of the latitude/longitude information in the dataset, we use, in addition to the already present dataset, geopandas to plot the listings on a geographical map of NYC with boundaries delineating the different boroughs and community districts. Using this, we performed further analysis on the dependence of the listing prices on borough and community districts in each borough.

Part of the data cleaning required identification of outliers. The listings with the lowest three prices were removed since it was reasonable to assume that the price listed were actually monthly rents and not the actual price of the properties. Furthermore, the most expensive listing turned out to be an incorrect entry as verified using Zillow information. The high price outliers were not removed since these were not incorrect prices and there would be no justification in removing them. Interestingly enough, the dataset did not have any missing values. However, we realized that they were filled in by the uploader, as a consequence of which there were numerous fractional numbers of bathrooms and over 1,600 listings with square footage of approximately 2,184. The former was further cleaned by rounding up all the bathrooms to integers. For the later problem, the community district of each of these listings was identified and instead of 2,184 sft, the average property size of the corresponding community district was used.

The community district information was mostly used for visualization and data cleaning purposes. For training a regression model, we the only attributes that were kept were 'TYPE', 'PROPERTYSQFT', 'BEDS', 'BATH', 'LONGITUDE', 'LATITUDE', and 'BOROUGH'. For better training of models, the 'LOG_PRICE' column was used as the target variable since it effectively scales the data to a smaller range and it can be easily converted to the actual price. We used 'Random Forest' and 'Extreme Gradient Boosing' regressors. Through hyperparameter fine-tuning, we found that both models perform similarly, yielding $R^2 \approx 0.84$.