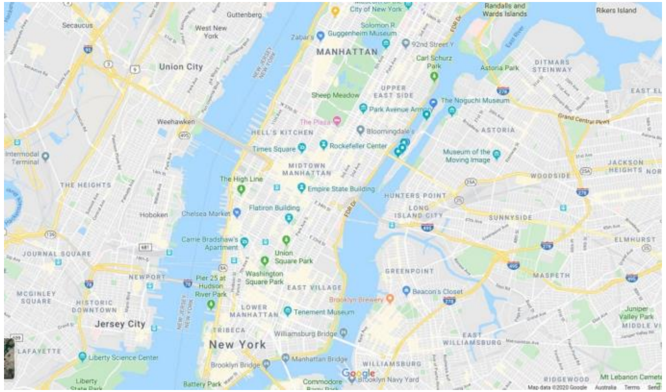
Manhattan Rent Prices

Michelle Wang, Ella Wanner, Sofia Kolobaev

Background

- NYC known to be notoriously expensive
- % of people who rent instead of owning in NYC -> 67.2%
- Manhattan - popular place with very high rent



Motivation/Broad Questions

- Explanatory modeling:
  - What factors impact the rent prices in Manhattan?
    - Out of these factors, which one(s) impact the price the most?
  - We explored other small questions along the way as well
- Predictive modeling:
  - Make models to predict rent prices given the factors
- Goal: help those who are looking for a place to rent in Manhattan

Techniques for Research and Findings

- Explanatory modeling
  - Excel
    - Correlation matrix
    - Multiple linear regression
  - Python
    - Multiple linear regression
    - Visualizations
  - Tableau
    - Visualizations
- Predictive modeling
  - Python
    - Random Forest
      - Feature importance graph
    - Multiple linear regression
    - K-nearest neighbors regression

Defining and Adjusting Dataset

rental_id rent bedrooms bathroom size_sqft min_to_su floor building_a no_fee has_roofd has_wash has_doorr has_eleva has_dishw has_patio has_gym neighborh borough

rental_id	rent	bedrooms	bathroom	size_sqft	min_to_su	floor	building_a	no_fee	has_roofd	has_wash	has_doorr	has_eleva	has_dishw	has_patio	has_gym	neighborhood
1545	2550	0	1	480	9	2	17	1	1	0	0	1	1	0	1	Upper East Side
2472	11500	2	2	2000	4	1	96	0	0	0	0	0	0	0	0	Greenwich Village
2919	4500	1	1	916	2	51	29	0	1	0	1	1	1	0	0	Midtown
2790	4795	1	1	975	3	8	31	0	0	0	1	1	1	0	1	Greenwich Village
3946	17500	2	2	4800	3	4	136	0	0	0	1	1	1	0	1	Soho
10817	3800	3	2	1100	3	5	101	0	0	0	0	0	0	0	0	Central Harlem
9077	1995	0	0	600	6	1	115	0	0	0	0	0	0	0	0	Midtown East
5150	2995	0	1	579	6	21	33	0	0	0	0	0	0	0	0	Battery Park City
9507	15000	2	2	1715	0	30	2	0	0	0	0	0	0	0	0	Flatiron
1437	4650	1	1	915	5	5	106	0	0	0	0	0	0	0	0	Upper East Side
404	2950	1	1	550	43	17	14	1	1	0	1	1	0	0	0	Upper East Side
8293	6920	3	2	1439	7	9	39	1	0	0	0	0	0	0	0	Midtown East
6594	4875	1	1	900	1	14	52	1	0	1	1	1	1	0	1	East Village
2964	4850	1	1	789	2	40	11	0	0	0	0	0	0	0	0	Midtown West
5405	3700	1	1	947	5	5	85	1	0	1	0	1	0	1	0	Upper East Side
5635	4200	2	1	900	4	8	111	1	0	0	0	0	0	0	0	Upper West Side
5832	2195	1	1	500	3	3	106	1	0	0	0	0	0	0	0	Lower East Side
7050	4200	1	1	640	3	15	52	1	0	0	0	0	0	0	0	Tribeca
476	9000	2	2	1749	4	15	10	0	0	0	1	0	0	0	0	Midtown East

Explanation of the x and y variables

- x-variables
  - neighborhood – neighborhood in Manhattan of the place
  - bedrooms – count of bedrooms of the place
  - bathrooms – count of bathrooms of the place
  - size_sqft – square footage of the place
  - min_to_subway – minutes away from the subway
  - floor – count of floors of the place
  - building_age_yrs – age of the building
  - no_fee – 0 for fee, 1 for no fee
  - has_roofdeck – 0 for no roof deck, 1 for has roof deck
  - has_washer_dryer – 0 for no washer dryer, 1 for has washer dryer
  - has_doorman – 0 for has no doorman, 1 for has doorman
  - has_elevator - 0 for has no elevator, 1 for has elevator
  - has_dishwasher - 0 for has no dishwasher, 1 for has dishwasher
  - has_patio - 0 for has no patio, 1 for has patio
  - has_gym - 0 for has no gym, 1 for has gym

- y-variable
  - rent - rent price per month of the place

Explanatory Modeling - Excel

	rent	bedrooms	bathrooms	size_sqft	min_to_subway	floor	building_age_yrs	no_fee	has_roofdeck	has_washer_dryer	has_doorman	has_elevator	has_dishwasher	has_patio	has_gym
rent	1														
bedrooms	0.638336	1													
bathrooms	0.769474	0.720885	1												
size_sqft	0.857954	0.771263	0.803627	1											
min_to_subway	0.035164	0.076543	0.086932	0.039448	1										
floor	0.215867	0.043539	0.127969	0.107186	0.082445369	1									
building_age_yrs	-0.12889	0.037228	-0.09542	0.014489	-0.18468244	-0.4	1								
no_fee	-0.1015	-0.10035	-0.06221	-0.14145	0.080087628	0.1	-0.221428548	1							
has_roofdeck	0.035165	0.002938	0.019556	0.024822	-0.020693437	0.1	-0.041305463	-0.0957	1						
has_washer_dryer	0.053873	0.008721	0.025752	0.038263	-0.001327204	0	-0.030013952	-0.0703	0.31345903	1					
has_doorman	0.031302	-0.01733	0.014745	0.026098	-0.009011783	0.1	-0.047265067	-0.1825	0.48983583	0.328291049	1				
has_elevator	0.05186	-0.00677	0.02115	0.040916	-0.000409704	0.1	-0.060627315	-0.1615	0.51653367	0.379998557	0.717929182	1			
has_dishwasher	0.052241	0.005467	0.038829	0.050364	-0.012243695	0	-0.027420171	-0.0787	0.3319987	0.455165684	0.343158787	0.4198125	1		
has_patio	0.029302	0.003037	0.042304	0.021921	0.00149951	0.1	-0.050321392	-0.0497	0.1225676	0.140979241	0.140967663	0.1345356	0.133127365	1	
has_gym	0.040609	-0.00411	0.029739	0.029347	-0.004315481	0.1	-0.063110456	-0.1012	0.5616258	0.348433148	0.633628002	0.6420993	0.34258959	0.123524	1

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.882783521
R Square	0.779306745
Adjusted R Square	0.778429984
Standard Error	1488.780337
Observations	3539

ANOVA

	df	SS	MS	F	Significance F	
Regression	14	27581413781	1970100984	888.8474677	0	Significance F is less than 0.05 so the model is significant
Residual	3524	7810829328	2216466.892			
Total	3538	35392243108				

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%	
Intercept	-422.16058	96.09549458	-4.393136035	1.15016E-05	-610.5689995	-233.7521606	-610.5689995	-233.7521606	
bedrooms	-315.4939511	42.60401118	-7.40526402	1.62904E-13	-399.0249683	-231.9629339	-399.0249683	-231.9629339	
bathrooms	1181.064349	74.49751781	15.85374096	9.89612E-55	1035.001731	1327.126968	1035.001731	1327.126968	
size_sqft	4.917379044	0.101254571	48.56451392	0	4.718855545	5.115902542	4.718855545	5.115902542	
min_to_subway	-16.41727249	4.64929521	-3.53113144	0.000419086	-25.5328545	-7.301690473	-25.5328545	-7.301690473	
floor	23.35722973	2.518139825	9.275588868	2.99721E-20	18.42007064	28.29438882	18.42007064	28.29438882	
building_age_yrs	-7.480691272	0.724996415	-10.31824588	1.30104E-24	-8.902146349	-6.059236195	-8.902146349	-6.059236195	
no_fee	-130.3259857	54.2076263	-2.404200195	0.016259198	-236.6074845	-24.0444869	-236.6074845	-24.0444869	
has_roofdeck	31.12172859	87.5360981	0.355530224	0.722213685	-140.5048181	202.7482753	-140.5048181	202.7482753	not significant because p-value >= 0.05
has_washer_dryer	152.6843884	79.76593857	1.914155228	0.055681232	-3.707693055	309.0764698	-3.707693055	309.0764698	not significant because p-value >= 0.05
has_doorman	-159.6798667	85.7650944	-1.861828146	0.062710503	-327.8341173	8.474383959	-327.8341173	8.474383959	not significant because p-value >= 0.05
has_elevator	87.30387301	87.58835155	0.996752096	0.318953291	-84.42512377	259.0328698	-84.42512377	259.0328698	not significant because p-value >= 0.05
has_dishwasher	-26.79488303	76.4687761	-0.350402928	0.726057272	-176.7224244	123.1326584	-176.7224244	123.1326584	not significant because p-value >= 0.05
has_patio	-103.0680672	111.9519074	-0.920645924	0.357298335	-322.5651624	116.4290281	-322.5651624	116.4290281	not significant because p-value >= 0.05
has_gym	-11.80939601	95.99639118	-0.123019166	0.902098936	-200.0235096	176.4047176	-200.0235096	176.4047176	not significant because p-value >= 0.05

SUMMARY OUTPUT	
Regression Statistics	
Multiple R	0.882496
R Square	0.778799
Adjusted R Square	0.778361
Standard Error	1489.014
Observations	3539

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*	
Regression	7	2.76E+10	3937634903	1775.979844	0	Significance F is less than 0.05 so the model is significant
Residual	3531	7.83E+09	2217161.933			
Total	3538	3.54E+10				

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	-426.394	93.28174034	-4.571031261	5.01958E-06	-609.2852943	-243.502208	-609.2852943	-243.502208
bedrooms	-313.554	42.517932	-7.374630825	2.04304E-13	-396.9162422	-230.1918616	-396.9162422	-230.1918616
bathrooms	1175.333	74.40178763	15.79711038	2.27402E-54	1029.458424	1321.208079	1029.458424	1321.208079
size_sqft	4.92447	0.101081218	48.71795052	0	4.726286275	5.122653234	4.726286275	5.122653234
min_to_subway	-16.3554	4.647974751	-3.518828022	0.000438926	-25.46841066	-7.242436933	-25.46841066	-7.242436933
floor	23.02461	2.503595245	9.196619294	6.15322E-20	18.11597324	27.93325143	18.11597324	27.93325143
building_age_yrs	-7.48416	0.722757473	-10.35500995	8.95263E-25	-8.901225178	-6.067096462	-8.901225178	-6.067096462
no_fee	-120.383	53.00953852	-2.270972733	0.023208587	-224.3156289	-16.45080423	-224.3156289	-16.45080423

Rent price = -426.394 - 313.554(bedrooms) + 1175.333(bathrooms) + 4.92447(size_sqft) - 16.3554(min_to_subway) + 23.02461(floor) - 7.48416((building_age_yrs) - 120.383(no_fee)

Explanatory Modeling - Python

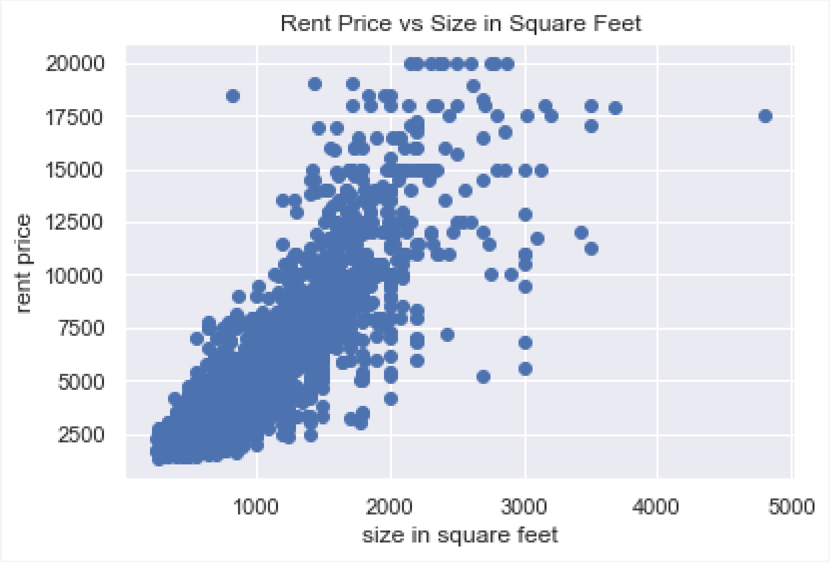
```python
# EXPLANATORY MODELING
# multiple linear regression - find the equation model
import pandas as pd
from sklearn.linear_model import LinearRegression
import numpy as np
df_manhattan_subset = pd.read_excel('manhattansubset.xlsx')


def disp_regress(df, x_feat_list, y_feat, verbose=True):
    """ linear regression, displays model w/ coef

    Args:
        df (pd.DataFrame): dataframe
        x_feat_list (list): list of all features in model
        y_feat (list): target feature
        verbose (bool): toggles command line output

    Returns:
        reg (LinearRegression): model fit to data
    """
    # initialize regression object
    reg = LinearRegression()

    # get target variable
    x = df.loc[:, x_feat_list].values
    y = df.loc[:, y_feat].values

    # fit regression
    reg.fit(x, y)
```

```python
    # predict with model
    y_pred = reg.predict(x)

    if verbose:
        # print model
        model_str = y_feat + f' = {reg.intercept_:.2f}'
        for feat, coef in zip(x_feat_list, reg.coef_):
            s_sign = ' - ' if coef < 0 else ' + '
            model_str += s_sign + f'{np.abs(coef):.2f} {feat}'
        print(model_str)

    return reg

disp_regress(df=df_manhattan_subset,
             x_feat_list=df_manhattan_subset.columns[1:],
             y_feat='rent');
```

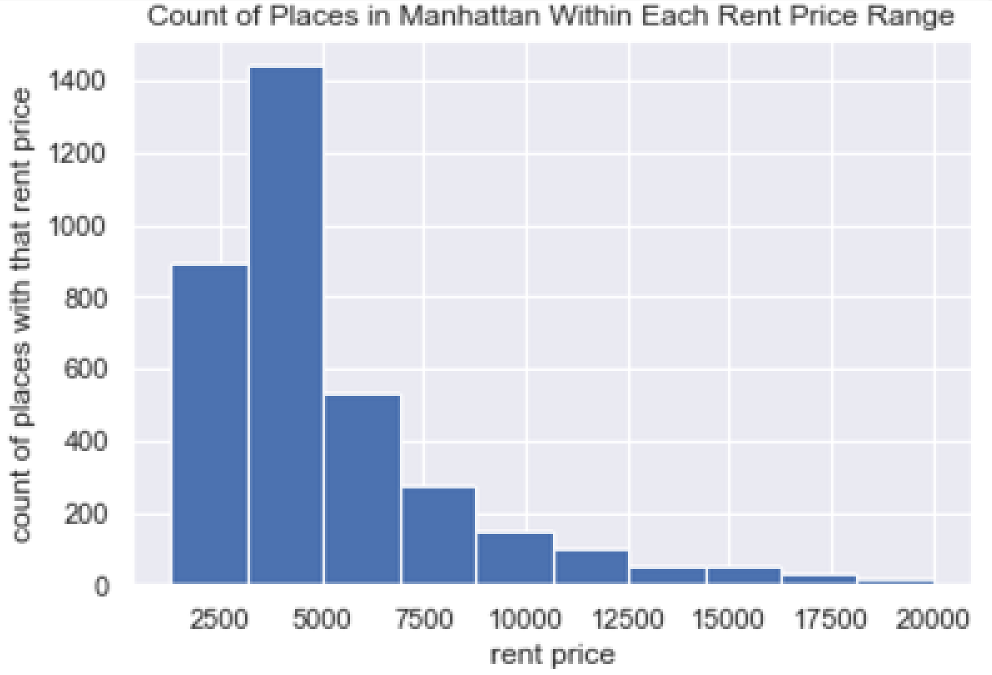
```
rent = -501.41 + 4.76 size_sqft - 6.70 building_age_yrs + 23.84
floor - 14.98 min_to_subway + 1130.02 bathrooms - 1956.17
neighborhood_Washington Heights - 1639.82 neighborhood_Central
Harlem - 194.16 bedrooms + 1691.73 neighborhood_West Village +
1357.01 neighborhood_Soho + 809.88 neighborhood_Chelsea
```

Explanatory Modeling - Python

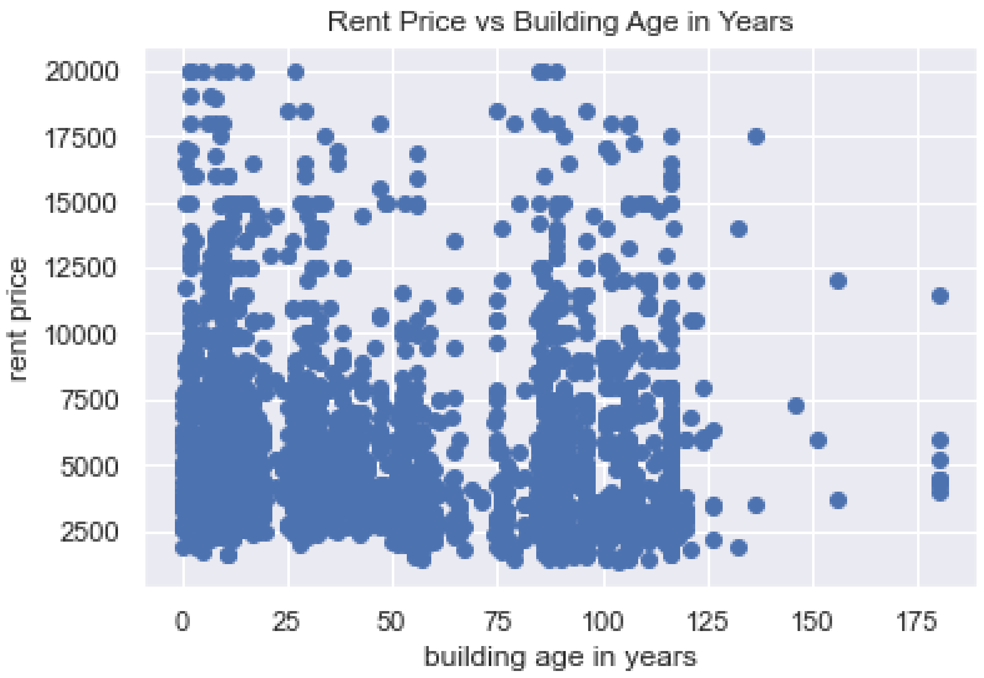
```
# square feet vs rent - scatterplot
plt.scatter(new_df_manhattan['size_sqft'], new_df_manhattan['rent'])
plt.xlabel('size in square feet')
plt.ylabel('rent price')
plt.title('Rent Price vs Size in Square Feet')
```



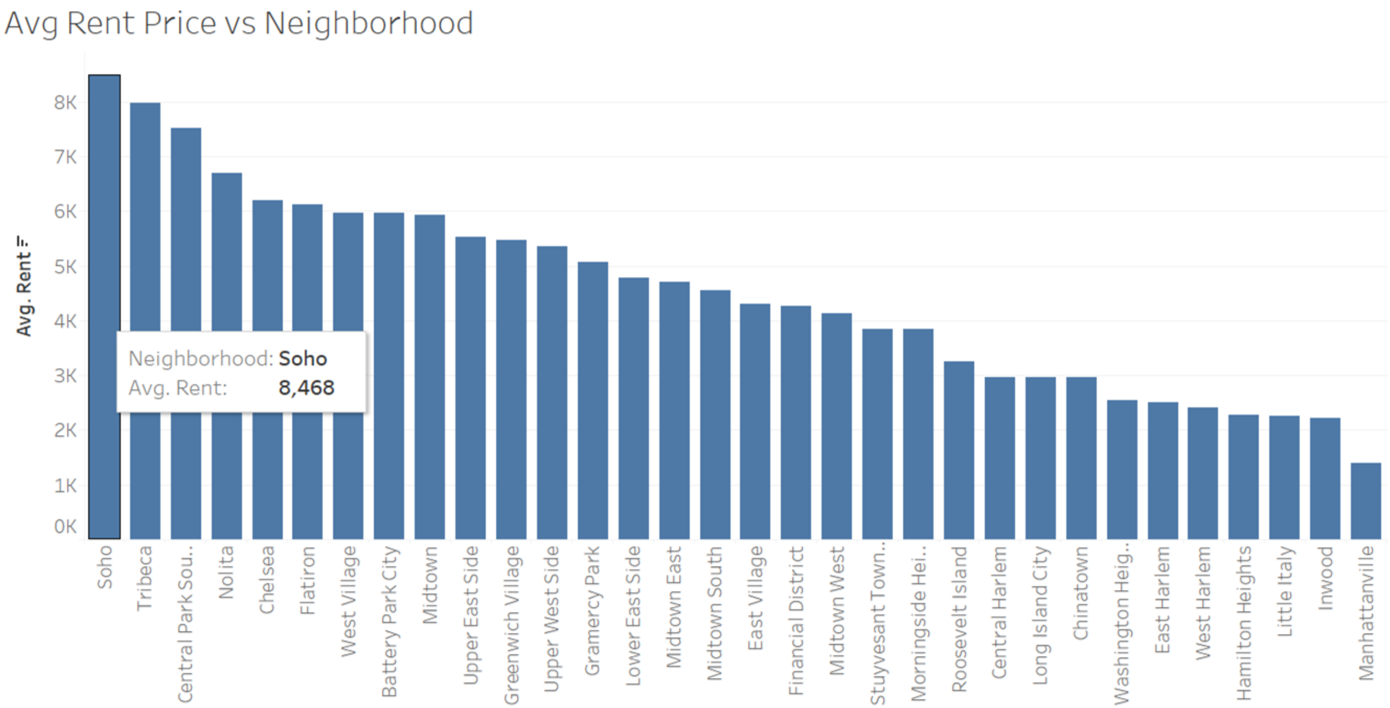
```
# histogram of rent price
plt.hist(new_df_manhattan['rent'])
plt.xlabel('rent price')
plt.ylabel('count of places with that rent price')
plt.title('Count of Places in Manhattan Within Each Rent Price Range')
```



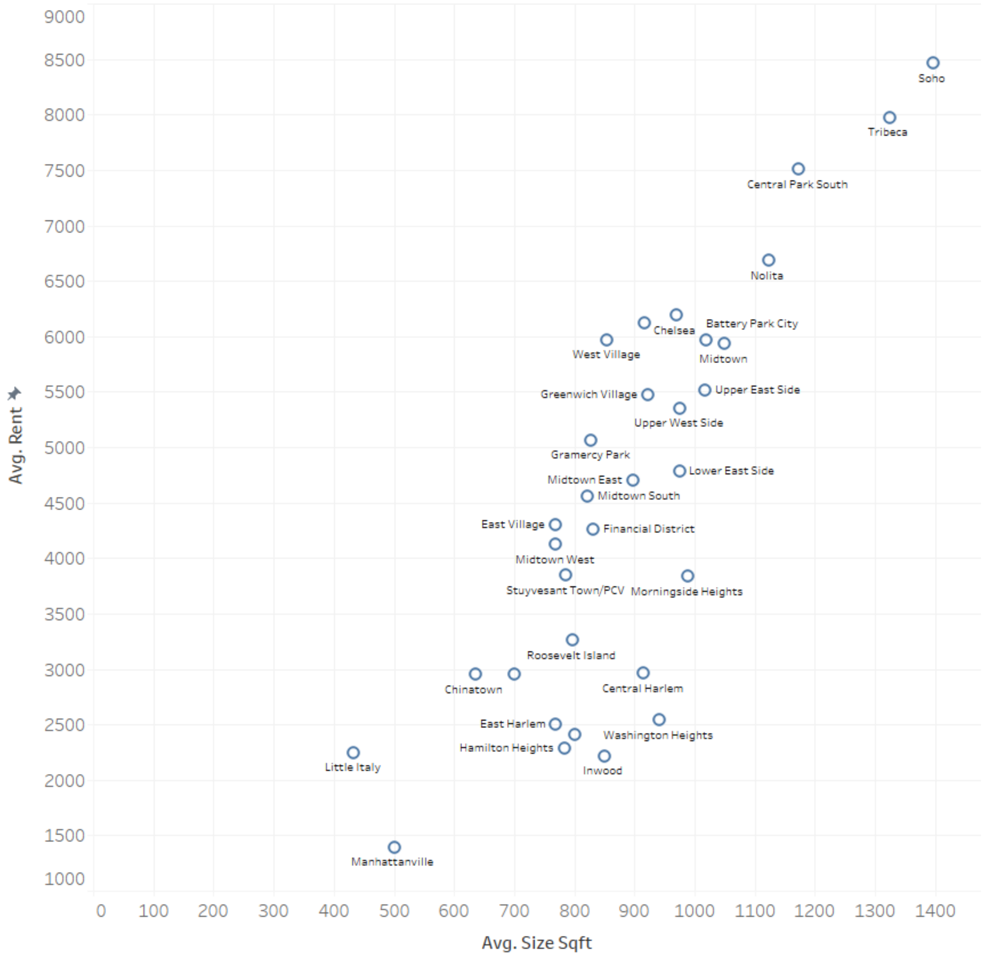
```python
# building age in years vs rent - scatterplot
plt.scatter(new_df_manhattan['building_age_yrs'],
            new_df_manhattan['rent'])
plt.xlabel('building age in years')
plt.ylabel('rent price')
plt.title('Rent Price vs Building Age in Years')
```

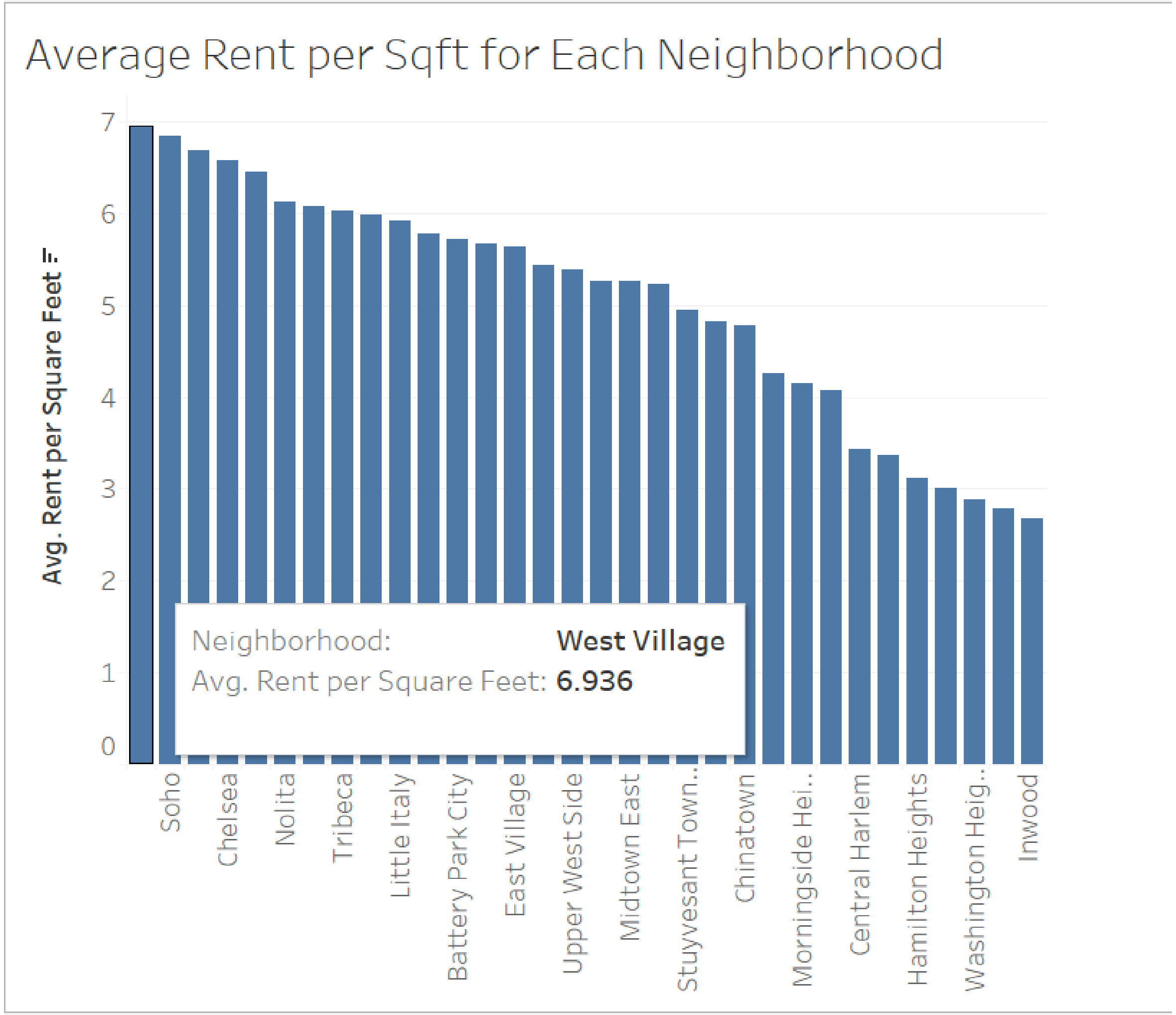


Explanatory Modeling - Tableau

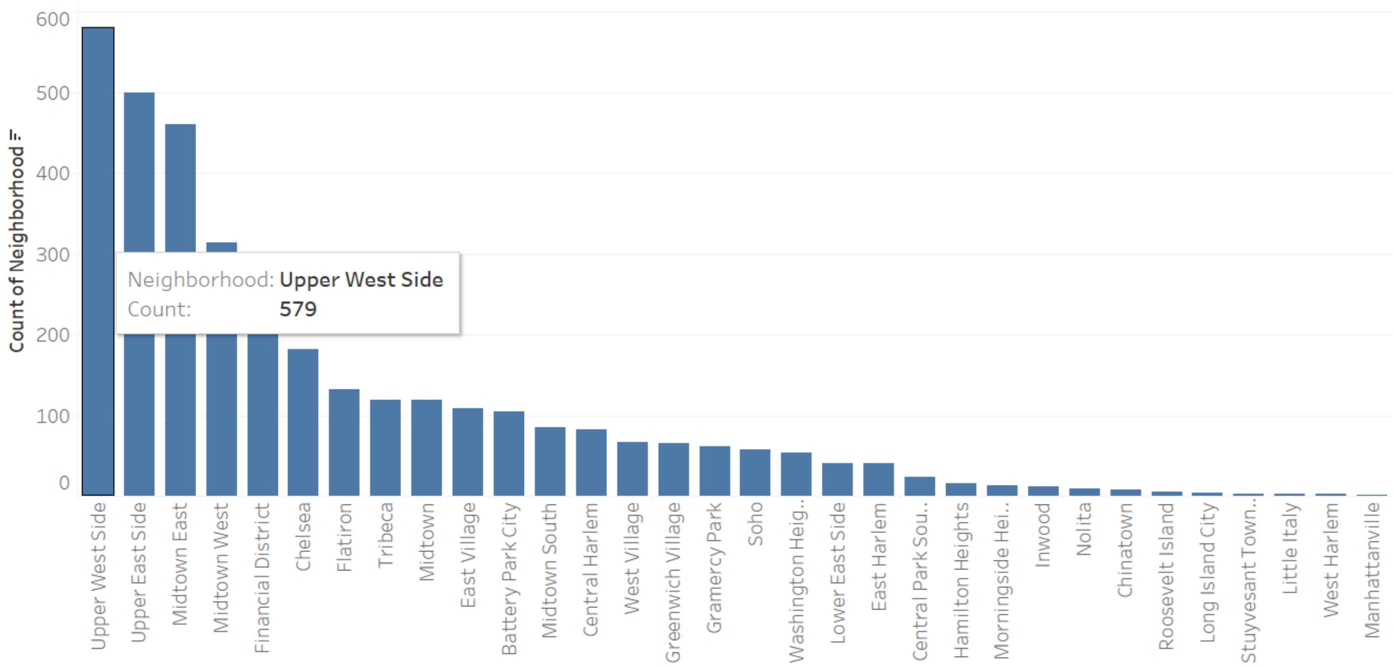


Average Rent vs Average Size Sqft

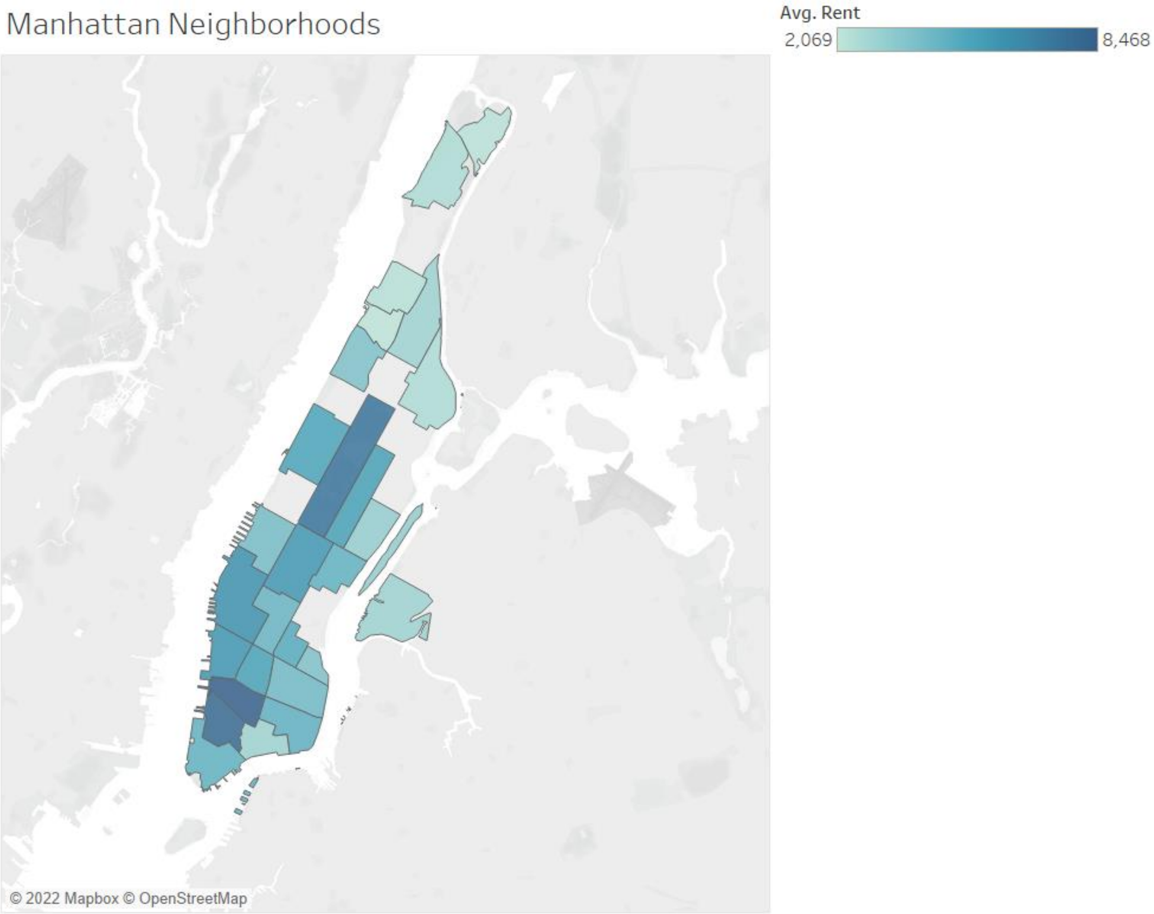



Average Rent per Sqft for Each Neighborhood
Avg. Rent per Square Feet
7
6
5
4
3
2
1
0
Soho
Chelsea
Nolita
Tribeca
Little Italy
Battery Park City
East Village
Upper West Side
Midtown East
Stuyvesant Town..
Chinatown
Morningside Hei..
Central Harlem
Hamilton Heights
Washington Heig..
Inwood
Neighborhood: West Village
Avg. Rent per Square Feet: 6.936

Count for Each Neighborhood



Manhattan Neighborhoods



Predictive Modeling - Python

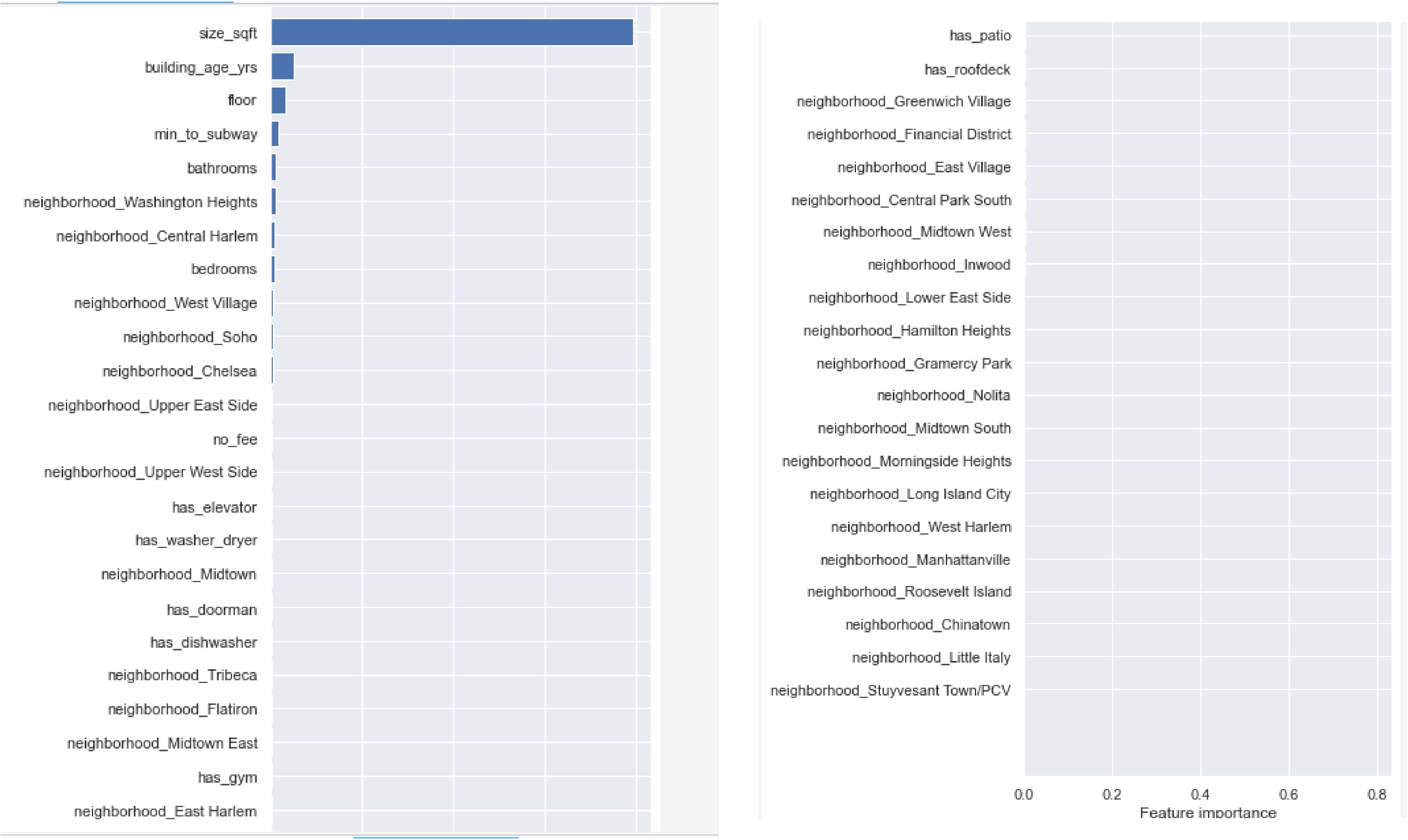
```python
# Random Forest - feature importance graph
# import necessary libraries
from sklearn.model_selection import KFold
import numpy as np
from sklearn.metrics import r2_score
from sklearn.ensemble import RandomForestRegressor

x_feat_list = new_df_manhattan.columns[2:]
y_feat = 'rent'

x = new_df_manhattan.loc[:, x_feat_list].values
y_true = new_df_manhattan.loc[:, y_feat].values

# initialize random forest
rf_reg = RandomForestRegressor()

# initialize k fold
skfold = KFold(shuffle=True)

# initialize y_pred, stores predictions of y
y_pred = np.empty_like(y_true)
```

```python
for train_idx, test_idx in skfold.split(x, y_true):
    # get training data
    x_train = x[train_idx, :]
    y_train = y_true[train_idx]

    # get test data
    x_test = x[test_idx, :]

    # fit data
    rf_reg = rf_reg.fit(x_train, y_train)

    # estimate on test data
    y_pred[test_idx] = rf_reg.predict(x_test)
r2 = r2_score(y_true, y_pred)
print(r2)
```

r2 = 0.84

```python
# Feature importance graph
import matplotlib.pyplot as plt
def plot_feat_import(feat_list, feat_import, sort=True):
    """ plots feature importances in a horizontal bar chart

    Args:
        feat_list (list): str names of features
        feat_import (np.array): feature importances (MSE reduce)
        sort (bool): if True, sorts features in decreasing importance
            from top to bottom of plot
    """

    if sort:
        # sort features in decreasing importance
        idx = np.argsort(feat_import).astype(int)
        feat_list = [feat_list[_idx] for _idx in idx]
        feat_import = feat_import[idx]

    # plot and label feature importance
    plt.barh(feat_list, feat_import)
    plt.gcf().set_size_inches(5, len(feat_list) / 2)
    plt.xlabel('Feature importance\n(Mean decrease in r2 across all Decision Trees)')
```

```python
# import libraries
import seaborn as sns

# call seaborn to make the plot nicer
sns.set()

# fit on entire dataset
rf_reg.fit(x, y_true)

# call the plot_feat_import to plot the plot
plot_feat_import(x_feat_list, rf_reg.feature_importances_)
```


size_sqft
building_age_yrs
floor
min_to_subway
bathrooms
neighborhood_Washington Heights
neighborhood_Central Harlem
bedrooms
neighborhood_West Village
neighborhood_Soho
neighborhood_Chelsea
neighborhood_Upper East Side
no_fee
neighborhood_Upper West Side
has_elevator
has_washer_dryer
neighborhood_Midtown
has_doorman
has_dishwasher
neighborhood_Tribeca
neighborhood_Flatiron
neighborhood_Midtown East
has_gym
neighborhood_East Harlem
has_patio
has_roofdeck
neighborhood_Greenwich Village
neighborhood_Financial District
neighborhood_East Village
neighborhood_Central Park South
neighborhood_Midtown West
neighborhood_Inwood
neighborhood_Lower East Side
neighborhood_Hamilton Heights
neighborhood_Gramercy Park
neighborhood_Nolita
neighborhood_Midtown South
neighborhood_Morningside Heights
neighborhood_Long Island City
neighborhood_West Harlem
neighborhood_Manhattanville
neighborhood_Roosevelt Island
neighborhood_Chinatown
neighborhood_Little Italy
neighborhood_Stuyvesant Town/PCV
0.0
0.2
0.4
0.6
0.8
Feature importance

Predictive Modeling - Python

```
# Multiple linear regression - only using the significant x variables
# (based on feature importance graph findings!)
x = df_manhattan_subset[df_manhattan_subset.columns[1:]]
y = df_manhattan_subset['rent']

# split into training and testing
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,
                                                    random_state=1)

# implement the linear regression model
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(x_train, y_train)

# coefficients
lm.coef_
lm.intercept_

# evaluate the model: using the testing dataset
predictions = lm.predict(x_test)
```

```python
# to evaluate, we compare predictions with y_test (actual data)
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, predictions)
rmse = mse ** 0.5
print(rmse)
# rmse = 1371.25

from sklearn.metrics import r2_score
r2 = r2_score(y_test, predictions)
print(r2)
# r2 = 0.80
```

Predictive Modeling - Python

```
# k-nearest neighbors regressor
from sklearn.neighbors import KNeighborsRegressor
from copy import copy
from sklearn.model_selection import KFold
from sklearn.metrics import r2_score

# Using x variables that are important based on feature importance graph
# x and y variables of interest
x_feat_list = df_manhattan_subset.columns[1:]
y_feat = 'rent'

# scale normalization
for feat in x_feat_list:
    df_manhattan_subset[feat] = df_manhattan_subset[feat] / df_manhattan_subset[feat].std()

# get the x and y from the dataset
x = df_manhattan_subset.loc[:, x_feat_list].values
y_true = df_manhattan_subset.loc[:, y_feat].values

# initialize a knn_regressor
knn_regressor = KNeighborsRegressor()

# cross validation
kfold = KFold(shuffle=True)

# allocate an empty array to store predictions in
y_pred = copy(y_true)
```

```python
for train_idx, test_idx in kfold.split(x, y_true):
    # build arrays which correspond to x, y train /test
    x_test = x[test_idx, :]
    x_train = x[train_idx, :]
    y_true_train = y_true[train_idx]

    # fit on training data
    knn_regressor.fit(x_train, y_true_train)

    # estimate rent
    y_pred[test_idx] = knn_regressor.predict(x_test)
r2 = r2_score(y_true, y_pred)
print(r2)
# r2 = 0.80
```

Conclusions

- Size in sq footage is the factor that is most important in predicting rent prices
- Multiple linear regression model based on only the most important x variables

```
rent = -501.41 + 4.76 size_sqft - 6.70 building_age_yrs + 23.84
floor - 14.98 min_to_subway + 1130.02 bathrooms - 1956.17
neighborhood_Washington Heights - 1639.82 neighborhood_Central
Harlem - 194.16 bedrooms + 1691.73 neighborhood_West Village +
1357.01 neighborhood_Soho + 809.88 neighborhood_Chelsea
```

- Other smaller conclusions:
  - Upper West side has the most listings
  - Avg rent price is most expensive in SoHo
  - As age of building increase, so does rent price
  - Most common price category for Manhattan rent - $2500-$5000 price category

Other Conclusions

- Strong predictive models can be made to predict rent price - mostly focused on using most importance x-variables, as dictated by the feature importance graph
- Challenges
  - Only 3539 records from one source, one dataset
  - Not sure when the data was collected
  - May be other x-variables even more important in predicting rent prices in Manhattan

Resources Used

- https://www.kaggle.com/datasets/zohaib30/streeteasy-dataset?resource=download)
- https://www.valuepenguin.com/new-york-city-renters-statistics
- https://www.nyc.gov/site/planning/data-maps/open-data/census-download-metadata.page
- Some of the predictive modeling code used were functions from DS 2500.