NYC Property Sales

Sale Price Predictions

Context

The NYC Property Sales dataset is a record of every building unit sold in New York City.

This is a record from the New York City Department of Finance's over a 12-month period between 2016 and 2017.

In this project we're going to build a model to predict the sale price of a property.

Data Exploration

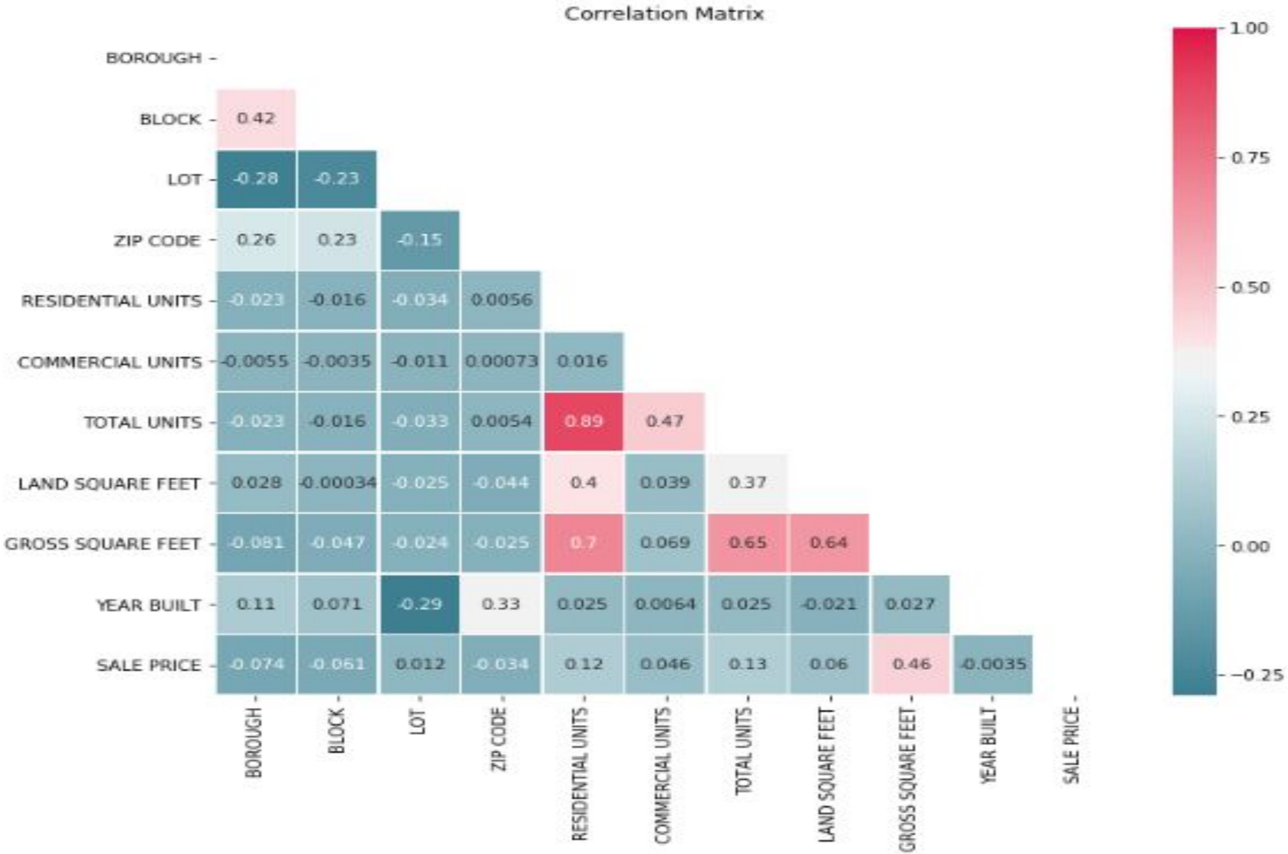
```
RangeIndex: 84548 entries, 0 to 84547
Data columns (total 21 columns):
BOROUGH                           84548 non-null int64
NEIGHBORHOOD                      84548 non-null object
BUILDING CLASS CATEGORY           84548 non-null object
TAX CLASS AT PRESENT              84548 non-null object
BLOCK                             84548 non-null int64
LOT                               84548 non-null int64
EASE-MENT                         84548 non-null object
BUILDING CLASS AT PRESENT         84548 non-null object
ADDRESS                           84548 non-null object
APARTMENT NUMBER                  84548 non-null object
ZIP CODE                          84548 non-null int64
RESIDENTIAL UNITS                 84548 non-null int64
COMMERCIAL UNITS                  84548 non-null int64
TOTAL UNITS                       84548 non-null int64
LAND SQUARE FEET                  84548 non-null object
GROSS SQUARE FEET                 84548 non-null object
YEAR BUILT                        84548 non-null int64
TAX CLASS AT TIME OF SALE         84548 non-null int64
BUILDING CLASS AT TIME OF SALE    84548 non-null object
SALE PRICE                        84548 non-null object
SALE DATE                         84548 non-null object
```

Data Cleaning

The column EASEMENT is completely empty and was deleted.

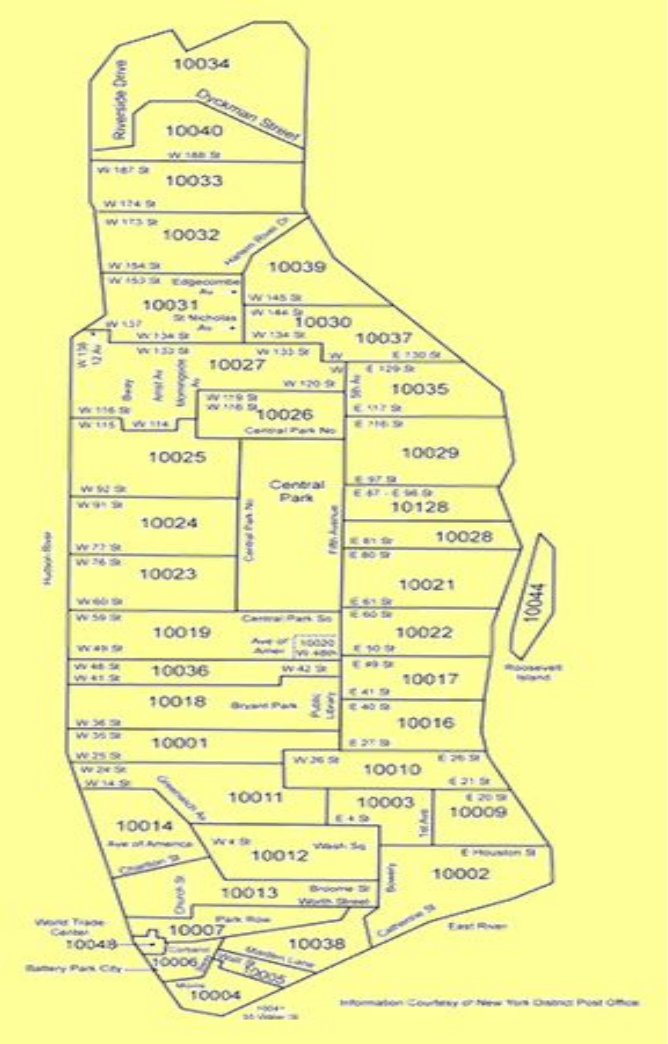
'ADDRESS', 'APARTMENT NUMBER' and 'NEIGHBORHOOD' are irrelevant in our perspective.

'BUILDING CLASS AT PRESENT' was dropped because it shared the same values with 'BUILDING CLASS AT TIME OF SALE’. Thus, only the latter was kept.



Correlation Matrix
BOROUGH
BLOCK
LOT
ZIP CODE
RESIDENTIAL UNITS
COMMERCIAL UNITS
TOTAL UNITS
LAND SQUARE FEET
GROSS SQUARE FEET
YEAR BUILT
SALE PRICE
0.42
-0.28 -0.23
0.26 0.23 -0.15
-0.023 -0.016 -0.034 0.0056
-0.0055 -0.0035 -0.011 0.00073 0.016
-0.023 -0.016 -0.033 0.0054 0.89 0.47
0.028 -0.00034 -0.025 -0.044 0.4 0.039 0.37
-0.081 -0.047 -0.024 -0.025 0.7 0.069 0.65 0.64
0.11 0.071 -0.29 0.33 0.025 0.0064 0.025 -0.021 0.027
-0.074 -0.061 0.012 -0.034 0.12 0.046 0.13 0.06 0.46 -0.0035
1.00
0.75
0.50
0.25
0.00
−0.25


1
2
3
4
5
1. Manhattan
2. Brooklyn
3. Queens
4. The Bronx
5. Staten Island


10034
Riverside Drive
Dyckman Street
10040
10033
10032
10039
10031
10030
10037
10027
10035
10026
10025
10029
Central Park
10128
10024
10028
10023
10021
10044
10019
10022
10036
10017
10018
10016
10001
10010
10011
10003
10009
10014
10012
10002
10013
10007
10038
10048
10006
10005
10004
World Trade Center
Battery Park City
East River
Roosevelt Island
Information Courtesy of New York District Post Office

Data Cleaning

We had the choice of working with either BBL ('BOROUGH', 'BLOCK', 'LOT') or ‘ZIP CODE’ ; we chose ‘ZIP CODE’ because it gives a good idea of the location of the property without being too specific (too much variables).

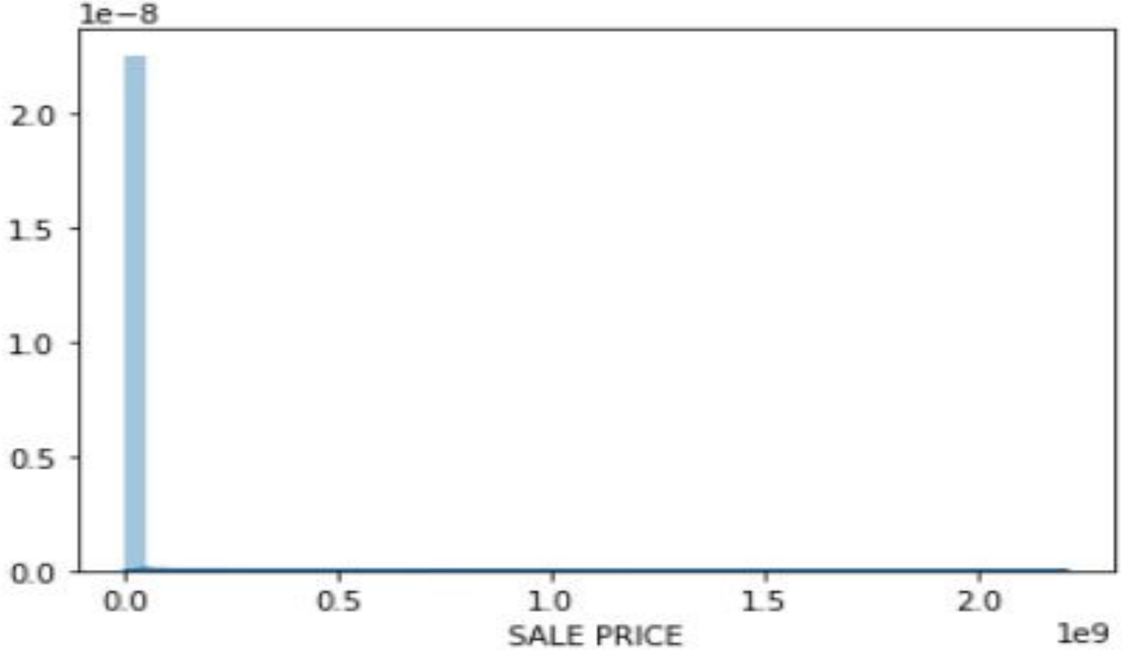
We added a new feature ‘AGE’ to represent the building age using 'YEAR BUILT' and 'SALE DATE'. This new information will replace ‘YEAR BUILT'.

The 'SALE DATE' feature was replaced by its corresponding ‘JULIAN DATE’ value.

We found missing values in ‘SALE PRICE', 'TAX CLASS AT PRESENT', 'ZIP CODE', 'YEAR BUILT', 'LAND SQUARE FEET', 'SALE DATE' that are presented in many different ways: ‘ -’, ‘ ‘, 0

Data Cleaning

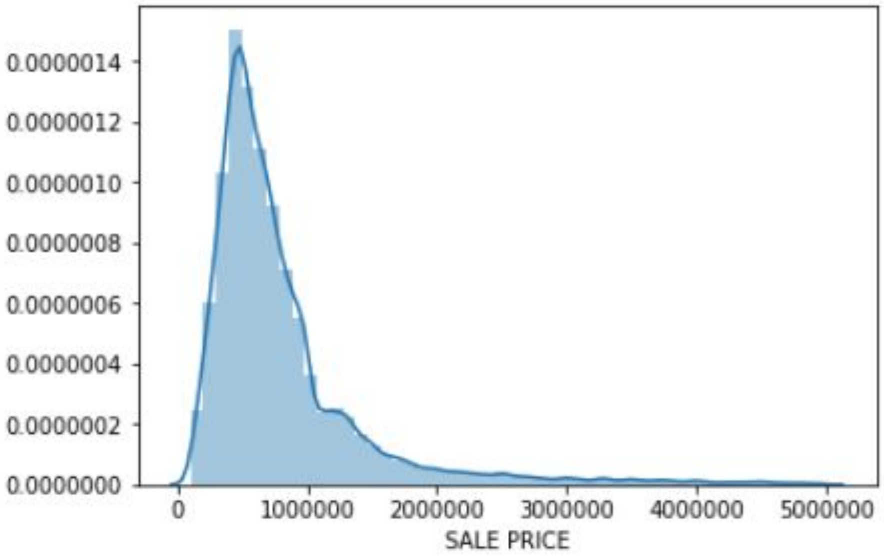
There are a lot of outliers in Sale Price variable:



Data Cleaning

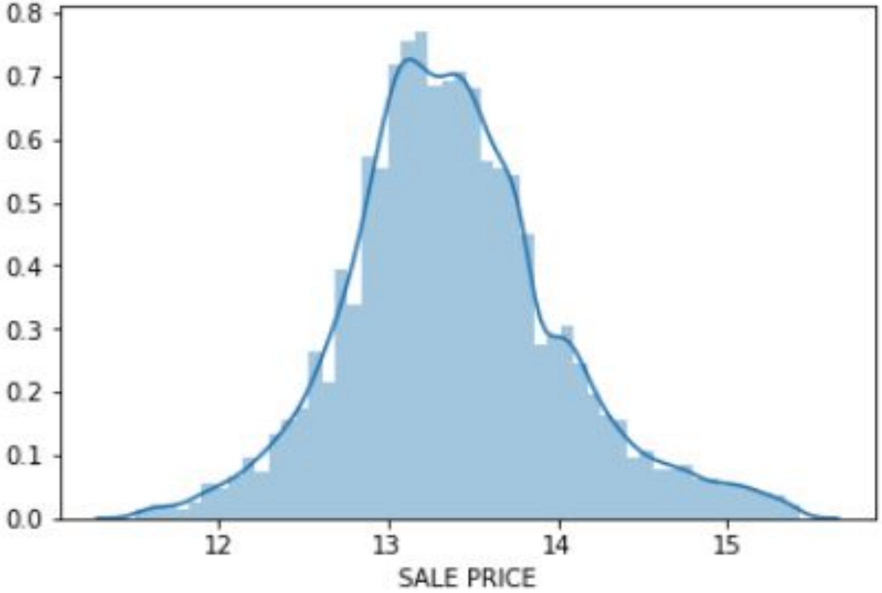
After making some inspection in sale prices we chose:

1) property needs to be more expensive than 100,000 USD
2) property needs to be cheaper than 5,000,000 USD



Data Cleaning

As we saw the Sale Price is positively skewed. So we used a non-linear transformation (log) to reduce skewness.



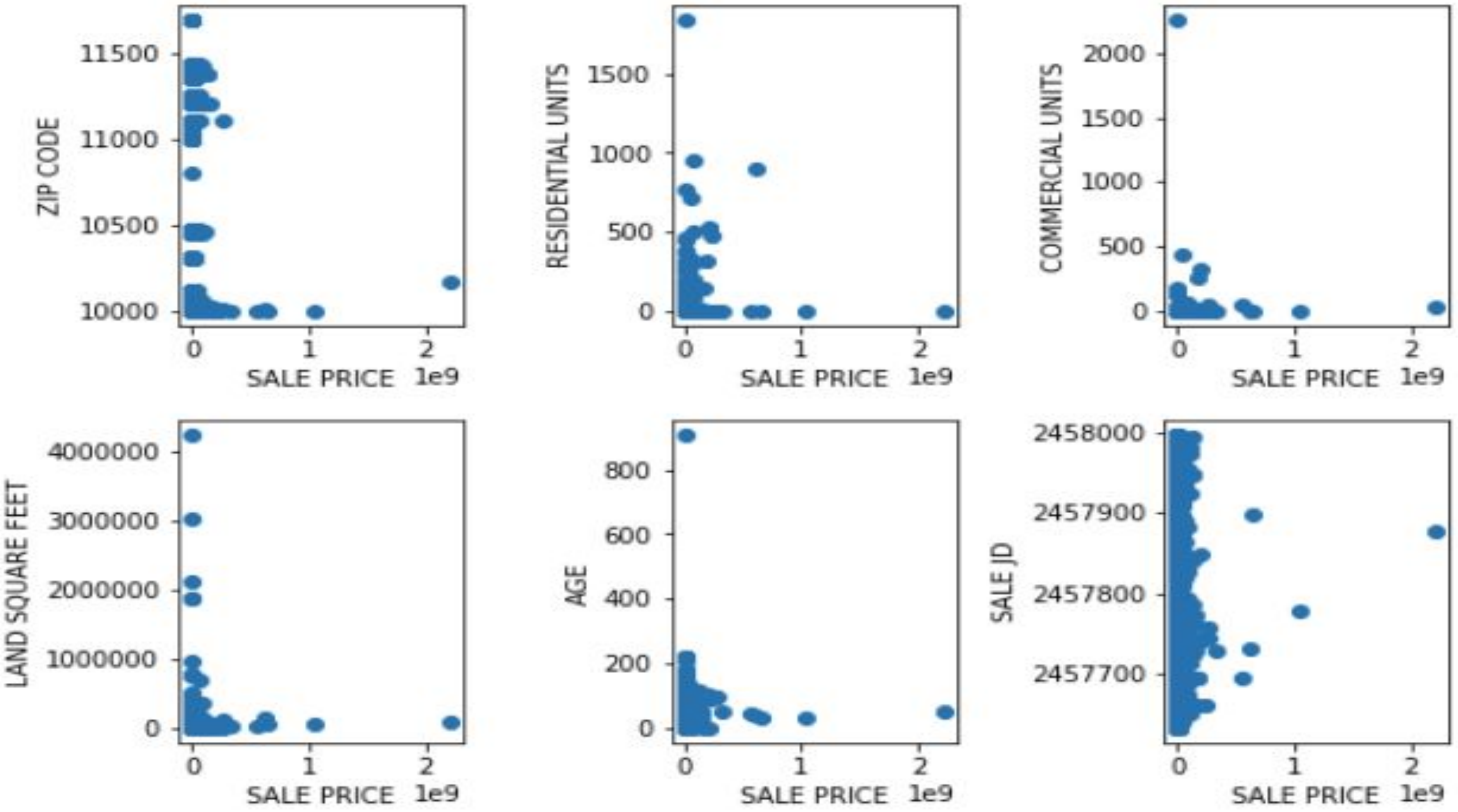
Data Cleaning

We found some duplicates that we had to eliminate.

Finally, these features were converted into dummy variables:

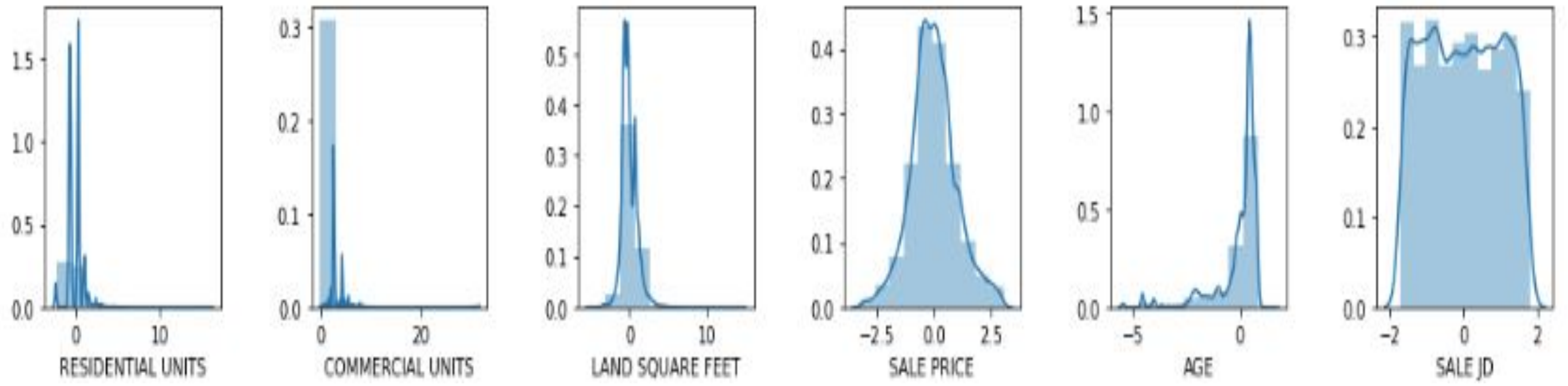
‘BUILDING CLASS CATEGORY', 'TAX CLASS AT PRESENT', 'ZIP CODE', 'TAX CLASS AT TIME OF SALE','BUILDING CLASS AT TIME OF SALE'.

Linear relationship

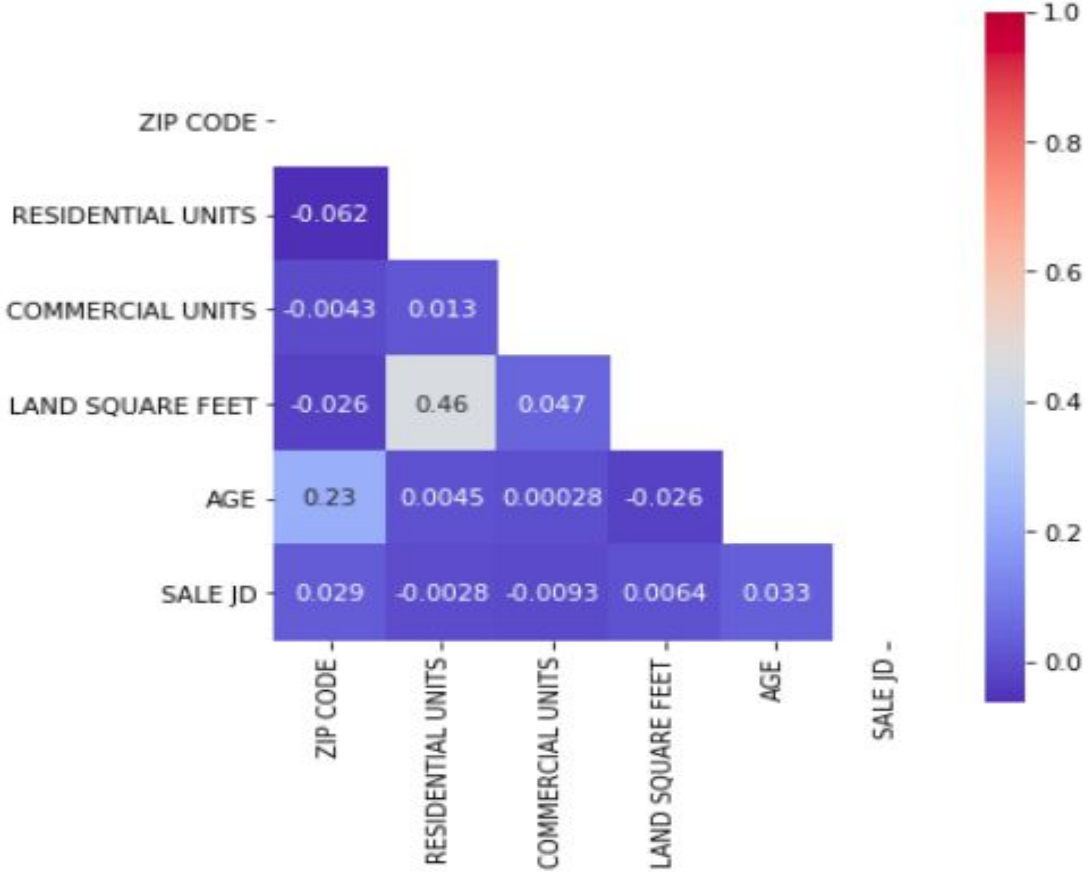


Multivariate normality and homoscedasticity

To make sure that our variables are multivariate normal and homoscedastic, we did a data scaling (with sklearn StandardScaler) by removing the mean and scaling to unit variance.



Multicollinearity



Linear Regression Models

We tested 4 different models to see which one gives us the best performance:

1. Linear regression (scikit-learn)
2. Ordinary Least Square Regression (OLS) (statsmodels)
3. Ridge Regression (l2-norm) (scikit-learn)
4. Lasso Regression (l1-norm) (scikit-learn)

Splitted the data into Training/Testing sets (70% 30%):

- Total sample size = 26845;
  - training sample size = 18791
  - testing sample size = 8054

Linear regression (scikit-learn)

There is a huge difference between train and test performance, hence our model is overfitting the data.

```
R² test:   -1.06704310659e+26
R² train:   0.566164475142
```

Ordinary Least Square Regression (OLS)

The problem of overfitting can also be seen in the OLS model.

OLS Regression Results

Dep. Variable:	y	R-squared:	0.568
Model:	OLS	Adj. R-squared:	0.562
Method:	Least Squares	F-statistic:	92.67
Date:	Mon, 09 Jul 2018	Prob (F-statistic):	0.00
Time:	12:43:10	Log-Likelihood:	-18872.
No. Observations:	18791	AIC:	3.827e+04
Df Residuals:	18527	BIC:	4.034e+04
Df Model:	263		

R^2 test: -9.68140200824e+23

Avoid overfitting: Reducing number of features

Our approach is to eliminate features whose p-values are greater than alpha level=0.05

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.0068	0.006	1.142	0.254	-0.005	0.018
x1	-0.0087	0.005	-1.688	0.091	-0.019	0.001
x2	-0.0522	0.007	-7.714	0.000	-0.065	-0.039
x3	0.0458	0.006	8.087	0.000	0.035	0.057
x4	-2.27e+12	1.66e+12	-1.366	0.172	-5.53e+12	9.87e+11
x5	-4.248e+10	3.11e+10	-1.366	0.172	-1.03e+11	1.85e+10
x6	0.1453	0.006	24.829	0.000	0.134	0.157
x7	0.0357	0.006	5.732	0.000	0.024	0.048
x8	0.0056	0.005	1.183	0.237	-0.004	0.015
x9	-0.0216	0.005	-4.554	0.000	-0.031	-0.012
x10	-0.1121	0.006	-19.514	0.000	-0.123	-0.101
x11	-0.1406	0.006	-23.590	0.000	-0.152	-0.129
x12	-0.1549	0.006	-26.021	0.000	-0.167	-0.143

Avoid overfitting: Reducing number of features

We reiterated the same step several times until we got:

OLS Regression Results

Dep. Variable:	y	R-squared:	0.411
Model:	OLS	Adj. R-squared:	0.408
Method:	Least Squares	F-statistic:	173.8
Date:	Mon, 09 Jul 2018	Prob (F-statistic):	0.00
Time:	12:43:12	Log-Likelihood:	-21794.
No. Observations:	18791	AIC:	4.374e+04
Df Residuals:	18715	BIC:	4.434e+04
Df Model:	75		

R^2 test: 0.396452843174

Linear regression (scikit-learn) after dropping the features

R^2 train: 0.410591281822
R^2test: 0.396452843174

Avoid overfitting: Regularization

1. Ridge Regression (l2-norm) (scikit-learn):

```
Lambda:  1882.4679033962318
R² test:  0.5147958742044085
R² train:  0.5396838046649428
MSE:  0.486765245367367
```

2. Lasso Regression (l1-norm) (scikit-learn):

```
Lambda:  0.01155064850041579
R² test:  0.5002804633826277
R² train:  0.5147207118725904
MSE:  0.5013149888367329
```

Ridge Regression coefficients

RESIDENTIAL UNITS	-3.811552e-03
COMMERCIAL UNITS	-1.152619e-02
LAND SQUARE FEET	4.013693e-02
AGE	-5.999720e-02
SALE JD	3.912290e-02
BUILDING CLASS CATEGORY_02	3.714218e-02
BUILDING CLASS CATEGORY_03	4.150925e-02
BUILDING CLASS CATEGORY_05	3.409526e-03
BUILDING CLASS CATEGORY_06	-6.157468e-03
BUILDING CLASS CATEGORY_07	4.509864e-02
BUILDING CLASS CATEGORY_08	1.426721e-02
BUILDING CLASS CATEGORY_09	-1.718196e-02
BUILDING CLASS CATEGORY_10	-3.094141e-02
BUILDING CLASS CATEGORY_11	0.000000e+00
BUILDING CLASS CATEGORY_14	1.771530e-02
BUILDING CLASS CATEGORY_21	1.369856e-02
BUILDING CLASS CATEGORY_22	2.391960e-02
BUILDING CLASS CATEGORY_23	8.516023e-03
BUILDING CLASS CATEGORY_25	0.000000e+00
BUILDING CLASS CATEGORY_26	-8.633022e-04
BUILDING CLASS CATEGORY_27	2.289951e-02
BUILDING CLASS CATEGORY_29	2.307751e-02
BUILDING CLASS CATEGORY_30	2.574141e-02

Lasso Regression coefficients

RESIDENTIAL UNITS	-0.000000
COMMERCIAL UNITS	-0.008405
LAND SQUARE FEET	0.012389
AGE	-0.062112
SALE JD	0.038114
BUILDING CLASS CATEGORY_02	0.021099
BUILDING CLASS CATEGORY_03	0.069500
BUILDING CLASS CATEGORY_05	0.000000
BUILDING CLASS CATEGORY_06	-0.000000
BUILDING CLASS CATEGORY_07	0.000000
BUILDING CLASS CATEGORY_08	0.000000
BUILDING CLASS CATEGORY_09	-0.020444
BUILDING CLASS CATEGORY_10	-0.032067
BUILDING CLASS CATEGORY_11	0.000000
BUILDING CLASS CATEGORY_14	0.001544
BUILDING CLASS CATEGORY_21	-0.000000
BUILDING CLASS CATEGORY_22	0.000000
BUILDING CLASS CATEGORY_23	0.008480
BUILDING CLASS CATEGORY_25	0.000000
BUILDING CLASS CATEGORY_26	-0.000000
BUILDING CLASS CATEGORY_27	0.013407
BUILDING CLASS CATEGORY_29	0.009605
BUILDING CLASS CATEGORY_30	0.012151

Ridge vs Lasso Regression

Ridge:

ZIP CODE_11215	0.116063
ZIP CODE_11217	0.105102
ZIP CODE_11231	0.092597
BUILDING CLASS AT TIME OF SALE_C1	0.086444
BUILDING CLASS AT TIME OF SALE_A3	0.079681
ZIP CODE_11238	0.077143
ZIP CODE_11216	0.074952
ZIP CODE_11219	0.071130
ZIP CODE_11230	0.070355
ZIP CODE_11201	0.069346

Lasso:

TAX CLASS AT TIME OF SALE_4	0.160202
ZIP CODE_11215	0.133402
BUILDING CLASS AT TIME OF SALE_C1	0.106102
TAX CLASS AT PRESENT_2	0.104679
ZIP CODE_11231	0.097996
ZIP CODE_11217	0.095919
ZIP CODE_11238	0.088104
ZIP CODE_11219	0.081278
ZIP CODE_11230	0.079508
ZIP CODE_11201	0.075725
BUILDING CLASS AT TIME OF SALE_A3	0.075318