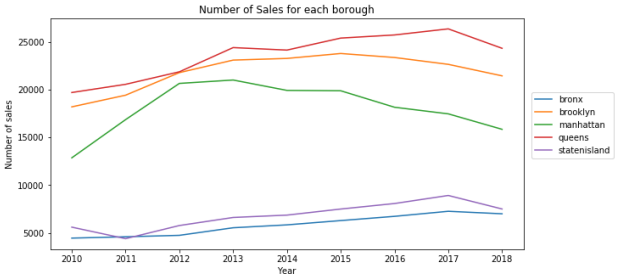
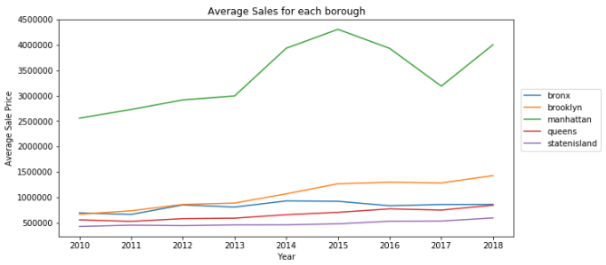
Real estate is a thriving business in New York City, there are countless investors and home buyers that want to purchase properties either as an investment or as a place to live in. However, purchasing a home is often times a very stressful event; Besides things like credit reports/scores, escrows, mortgages, or zoning issues first time home buyers or property investors wonder if the property they are purchasing are worth the price they are paying for. Getting a property evaluation is unreliable short of getting a professional real estate appraisal. I started out wanting to see if there was a better way to determine whether some properties were undervalued or overvalued, this would allow home buyers or property investors to find deals that would often go unnoticed. There are many real estate companies that deploy machine learning models to find a price, most of them tend to use nearby properties that were recently sold and average the prices of ones that are similar to them. Although, that does not seem like a bad idea, I believed that there was a better way to do it if I included nuanced features/variables that I knew increased property prices. Specifically, I wanted to see if nearby construction data would help achieve a better model than the ones other real estate companies use. However, due to data and time limitations I had to compromise and eventually, my end project was quite different from the one I envisioned at the beginning.

My original dataset was one that I scrapped from propertyshark.com using a python library called scrapy. Scrapy basically allowed me to harvest webpage data from a website. One of the reasons why I scrapped PropertyShark as opposed to something like Trulia or Redfin was that the later were very unfriendly for scraping, using a bot captcha to prevent them. At the time I had the option of using an API to get pass the captcha however that would have taken an enormous amount of time due to the fact that I had to get hundreds of thousands of property entries. PropertyShark, on the other hand, was less strict and the only thing I had to be careful of was the number of requests at any given time—which was easily bypassable with request throttling and IP spoofing. The original dataset I collected had a bunch of information that I thought was—at the time—very important, such as features like distance to a police station, park, estimated



price, etc. However, it was only after downloading everything did I realize the first set of problems with the data collected.

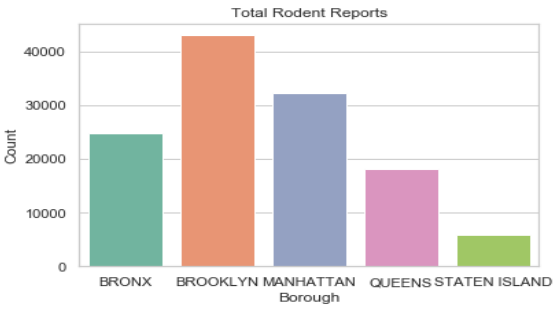
The main issue I had with the original dataset was that the price they used for a property was the tax assessor's market value, more often than not that price often lags behind the actual sales price. So instead of using that dataset I found another dataset that fitted my needs better—the rolling sales for NYC for 2010-2018. The data there were actually prices that people were actually willing to pay for the particular property and therefore, it was more useful for a predictive model on if a property price was being undervalued or overvalued.



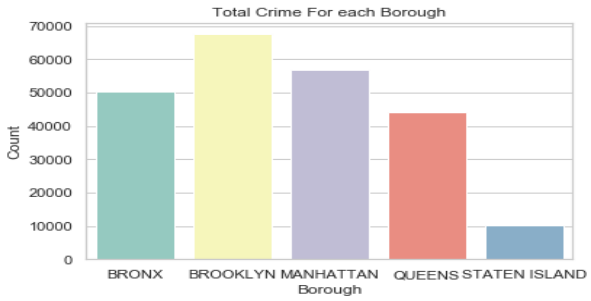

Looking at the initial data, we can see that most of the property sales happen in Queens, Brooklyn, and Manhattan. Manhattan had a much higher average price compared with the other 4 boroughs, there also seems to be a trend upwards for property prices for Brooklyn and debatably Queens. As for specific reason I bought in extra features from other datasets to supplement the rolling sales dataset. Specifically, they are rodent infestation report, crime sightings report, and construction plan datasets.

The original datasets for each category were quite messy so I had to do a little data wrangling. For the rolling sales dataset, there are quite a number of 0s for gross area. My initial solution was to get the average of that property's gross area and use that for the missing data. However, I found that there were many zip codes that had very few sales or some zip codes that had too many missing gross areas as well. In the end, I decided that using an average of the property's borough was the most generalized approach given the circumstances. However, I understand that it might have skewed the data a little bit. In addition to that, I also removed certain properties with sales prices of under $50,000. A majority of those sales were from outlier reasons. One example might be because the property was part of a deal and selling it for low a low price might be a way to game taxes.

For the rodent dataset, I used rodent reports from 2010-2018 just like the property rolling sales. There were some data points that were missing, however, there were not many of them and being that I was only interested in the quantity of rodent reports and their locations I just removed the unclean data points entirely. I imagined it would not cause any issues because the removed points were spread out quite evenly over multiple zip codes.




The crime dataset on the other hand, I only managed to find a reliable 2018 data. However, looking at compstat's year to year report of crimes, there did not seem to be a big variance between the crime activity from 2010-2018. Due to that fact, I decided to generalize a bit and extrapolate the crime reports of 2018 to the rest of the years. Although this might be generalizing a bit too much, at the time I didn't think it would be too much of a big deal. I also found that violent crimes such as felony assault or rape had a bigger correlation to property prices than nonviolent white-collar crimes did. After confirming that, I made a list of violent crimes arbitrarily throwing in a few that I thought might seem bad as well such as prostitution, criminal trespassing, arson, and a few others that sounded bad but were extremely rare. Interestingly, the rodent reports and crime reports seem to have some parallels. The rankings for rodent reports and crime reports seem to mirror each other. Intuitively, it seems there might be a correlation between the two.




```
["ARSON", "ASSAULT 3 & RELATED OFFENSES", "DANGEROUS DRUGS", "DANGEROUS WEAPONS"
  "FELONY ASSAULT", "GRAND LARCENY", "GRAND LARCENY OF MOTOR VEHICLE", "KIDNAPPING",
  "KIDNAPPING & RELATED OFFENSES", "MURDER & NON-NEGL. MANSLAUGHTER", "RAPE" , "ROBBERY"
  "PROSTITUTION & RELATED OFFENSES", "BURGLARY", "CRIMINAL TRESPASS", "FELONY ASSAULT"]
```

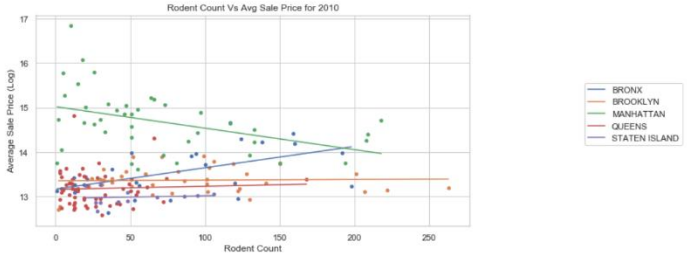
*Figure above: Crime types used for crime report dataset*

As for the construction data, I found construction plan datasets for 2017-2018. The original dataset only had zip codes of the owner's property. Often times the property building plans and the owner's home address were not even in the same state so I had to call several APIs to find the property's zip code from their street address. I found that there were some addresses

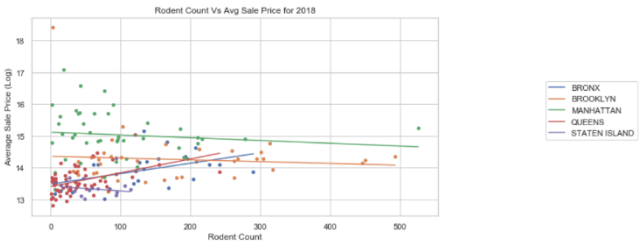
that were not found from geoclient's API. A few of those were relatively near my home, so after I going to those property plan locations I found that some locations were recently build or not build yet, so their addresses have not finished registering. Unfortunately, the first thought in my head was to correct each empty zip code data entry manually by finding the building plans and inputting the zip codes that way. Just deleting those entries were not an option because sometimes the property building plans were multimillion-dollar ventures; deleting them would have possibly skewed the data for certain zip codes. I decided to use the total price instead of the average price just because of the huge variances between building plans and because several building plans were split into multiple projects. One thing to note is that the datasets for construction plans are from 2017-2018, but extrapolating the data like how it was done with the crime data is certainly a bad idea because property investments have risen in recent years.

The main data features that I focused on for the prediction model are the rodent infestation count, crime report count, and the average prices of the construction costs for properties. For these features, I grouped them by their borough and zip code and did a simple linear regression to see the relationship between those features and the average sale prices of the properties. On the right, you can see a simple regression of rodent data for 2010. It seems that some boroughs have a correlation of some kind with a good p score while other boroughs seem to have no correlation at all. It seems to be the same for the other dates as well, like the example of 2018 right below the 2010 one. These seem to show that it is possible for rodent data to influence the sale prices of some boroughs. It also changes depending on the data. In 2010 rodent reports seem to have

```
Correlation for BRONX is: 0.522256 with a p value of 0.007405
Correlation for BROOKLYN is: 0.030956 with a p value of 0.853626
Correlation for MANHATTAN is: -0.418309 with a p value of 0.006495
Correlation for QUEENS is: 0.048196 with a p value of 0.719392
Correlation for STATEN ISLAND is: 0.100614 with a p value of 0.755707
```




```
Correlation for BRONX is: 0.458785 with a p value of 0.021068
Correlation for BROOKLYN is: -0.088937 with a p value of 0.595428
Correlation for MANHATTAN is: -0.109781 with a p value of 0.478085
Correlation for QUEENS is: 0.451622 with a p value of 0.000229
Correlation for STATEN ISLAND is: -0.248796 with a p value of 0.435529
```




```
Correlation for bronx is: 0.634319 with a p value of 0.000501
Correlation for brooklyn is: -0.151978 with a p value of 0.355694
Correlation for manhattan is: 0.263885 with a p value of 0.073083
Correlation for queens is: 0.321417 with a p value of 0.009604
Correlation for statenisland is: -0.039920 with a p value of 0.896978
```

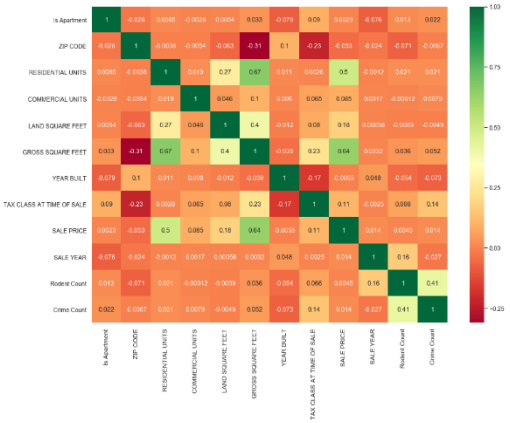
little effect on the sales price for Queens, however that changes in 2018.Similar to the rodent data, the crime data also only seems to show correlation for only a few of the boroughs.

For all three main datasets, they were analyzed using the Pearson correlation (calculated below), however it doesn't really show anything that could be extremely meaningful. It really only shows that there might be a meaningful connection between those features and the sales price of properties. However, it is better that there is some correlation instead of no correlation.

$$r_{pb} = \frac{\sum(x - m_x)(y - m_y)}{\sqrt{\sum(x - m_x)^2 \sum(y - m_y)^2}}$$

*Pearson's correlation*

There was about a dozen features that were not classification data that seemed to have some effect on the sales price of the properties. After all the data cleaning and one hot encoding was done, there were roughly 500 features for the datasets. Because there are so many features in the dataset and since we do not have a clear understanding of which of those features are important, Lasso regression helps automatically determine which coefficients and features should be selected or removed. Lasso regression removes coefficients that are unnecessary and "shrinks" features without a big risk of increasing bias. At the time, it seemed Lasso was the way to go, however, after scoring the Lasso regression model, Lasso did not do as well as I hoped.




*Correlation heatmap, before one hot encoding*

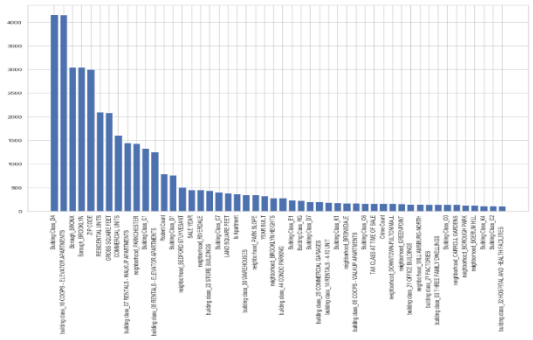
Moving on from that, I tried out a Random Forest model, it also does somewhat of an automatic (albeit random) feature selection using bootstrap resampling. Due to time constraints of running the model I could not hyper tune it to be extremely precise however I did attempt a few parameter tunings


*Residual Plot: Random Forest*

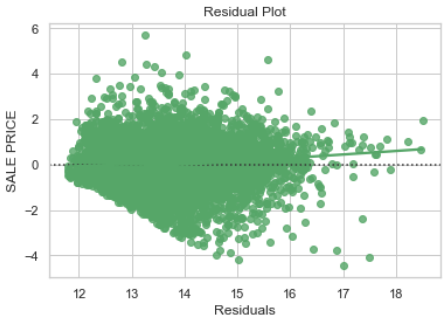

*Predicted vs actual: Random Forest*

to determine what number of trees and number of features would generate a good model score. After plotting the predicted vs actual, I noticed that there were too many points on the plot, I had to add an alpha value to each point so that I could see where most of the points were. There did not seem to be too many outliers, which was a good sign, however I noticed that there was a large variance between the predicted and actual. I felt like this was due to either an incomplete feature cleaning or it could be due to other external features that were not available in the dataset that was constructed.

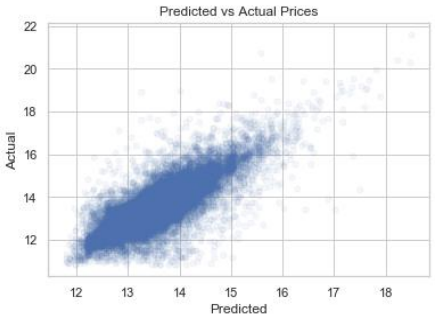
Along with random forest, I also tried a Jackknife resampling method for random forest. It was a new technique that was gaining popularity and was mentioned to me. However, I could not find an easy to use implementation of it, so unfortunately, I had to implement it myself with a twist. So instead of doing a complete random jackknife resampling method (which is just basically a bootstrap resampling without replacement) I did an average between gotten from a normal jackknife random forest regression and a decision tree regression of the best n (equal to the rest) features. I thought it would be appropriate since I already formulated the features that were most impactful using SelectKBest from sklearn. Unfortunately, the R^2 score was just slightly lower than the bootstrapped random forest one, a 63% vs a 73% for random forest. This could be for several reasons, but I personally believed that the implementation of the jackknife random forest was not perfect.



*SelectKBest Plot: best 50 features*




*Residual Plot: Jackknife Random Forest*




*Predicted vs actual plot: Jackknife random forest*

As always, more data and more feature cleaning would probably help the model better. There was a lot of generalization that was done to make data handling easier. For example, I generalized rodent data (count), crime data(count), construction data (average sales) into zip codes. Doing that did not account for several things like how far the crime, rodent infestation, or construction plan was from the specific address. I think if I had better processing power I would probably do that instead of just lumping them into groups.

The end result was great for a learning experience however using it to actually find good deals would be a not so great idea. There more features that would probably help, however actually getting the features would be the difficult part.