Starting this project we knew we wanted to focus on the house price evaluation problem (see problem formulation). What we knew coming into the project, something which was a big focus point for us, is that the location of an apartment or a property has massive influence on its price. One of our goals was to add GIS (geographic information system) layers to whatever dataset we find, and thus we assumed we'll get better results.
We came into the project with an understanding that house price evaluation has many quarks and fine points, and we wanted to try and understand them in addition to coming up with interesting insights about evaluating the price of a property or apartment.

The Dataset

In the beginning (before any work was done) we wanted to focus on evaluating the price of a house or an apartment (not a property). We figured out that evaluating the price of a 3 bedroom apartment where we know the size of the living room, what floor it is on and whether the apartment has a balcony or not (and a lot of other data on the apartment) will be a manageable problem. Unfortunately we couldn't find a dataset with apartments data. We tried numerous sources such as the Tax Authority website which had this data, but we couldn't download it (had a captcha that would let you get out 150 at a time and some data was missing). After about 3 weeks of searching we found a property sales dataset from New York and decided to go with it.
We chose this New York dataset for many reasons:

1. It was open and ready to download. Also most fields didn't have a lot of missing data.
2. There's massive business interest in New York. It's a place where evaluating a house is actually an important problem.
3. New York has high density of property sales. This is a major advantage since properties are usually appraised by looking at similar properties nearby.
4. There's a massive interest in New York's property prices, as New York is one of the biggest and most influential cities in the world.
5. We had to make sure we will be able to add GIS layers into our model. Therefore as New York is so popular we could also find GIS data for the properties in the dataset (and New York in general)

D**escription of Dataset:**

The dataset included information on property deals and transactions around NYC.
A property (different from an apartment or a house) can be many things. It can be a museum, an asylum, a skyscraper, a one family house, an apartment building and more.

As you can probably see there's incredible variance in the kinds of properties there are, so we'll start by looking at the dataset and describing it, and then continue to what we focused on.

The fields and columns of the dataset:
BOROUGH - numbered (1-5) each number represents a different borough in NYC. The boroughs are: Manhattan, Bronx, Brooklyn, Queens, Staten Island.
NEIGHBORHOOD - NYC is "partitioned" into neighborhoods (such as Tribeca or Chelsea), this field included the neighborhood which the property is located in (written in words)
BUILDING CLASS CATEGORY - indicates (in words i.e. a string) the kind of property. this field takes a lot of values, the properties can be factories, store buildings, one family homes, two family homes and many more.
TAX CLASS AT PRESENT - a tax class determines the amount of taxes (in percentage of assessed property value) that the property owner has to pay yearly. There are 4 classes where 1 is the highest tax percentage and 4 is the lowest.
BLOCK, LOT - New York is partitioned into blocks and these blocks are partitioned into lots. This is just the block and lot numbers of the property.
BUILDING CLASS AT PRESENT - New York has many building classes. every one has a very specific definition. Follow this link https://www1.nyc.gov/assets/finance/jump/hlpbldgcode.html for a more detailed explanation about each of the 100 building classes. (Note: building class is different from building class category)

Also, address, zip codes area of the land, and area of the property in addition to the year built, sale date and of course SALE PRICE (and more).
Each row in the dataset contained general information about the property (see notebook)

Dataset Initial Cleaning:

Our initial dataset contained 600,000 transactions which occurred between 2010 and 2017. At the first look we took at the dataset we noticed that a lot of SALE PRICEs (what we are trying to assess) are missing. after deleting every row with missing sale price we were left with about 300,000 transactions. We also had to remove every entry with missing GROSS SQUARE FEET and LAND SQUARE FEET (since it's impossible to evaluate the value of a property without knowing its area).
After dealing with this major null value problems we had another one to attend to which was the variance in kinds of properties. There's a huge difference between evaluating the price of a factory and the price of an apartment building. We figured out that it will be better

for our model to focus on specific kinds of properties rather than every kind of property (some of which have nothing to do with evaluating the other ones). This principle of evaluating a property by looking at the prices of similar properties in the area is one of the key principles in the theory of appraisal, and this was the main reason we decided to focus on specific kinds of properties.
Therefore we decided to focus on evaluating family homes (private houses) and apartment buildings. There are 3 types of family homes - one family home, two family home, three family homes. (see appendix-2 for more information about these types of properties)

After filtering our data to these building categories and eliminating our null values in the SALE PRICE, GROSS SQUARE FEET and LAND SQUARE FEET columns we stayed with about 200,000 transactions (190,699 to be precise). The amount of transactions we were left with was big enough for a machine learning data science problem (which was another reason we chose to go with these building categories).
Every step we would describe in the next lines was done starting from this filtered dataset.

Problem Formulation:

Our problem was a property evaluation problem. Given data such as area, location and time of sale of a property we wanted to be able to evaluate the price for which it sold.
To clarify, say a one family home in queens was sold for 950,000$ in 2016 then we would like our model to evaluate the house (given data about the house) at about 950,000$ (essentially “predicting” the SALE PRICE column)

Data Exploration & Analysis:

Checking out correlations (pearson correlation) between the numerical fields, we found a correlation of 0.78 between SALE PRICE and TOTAL UNITS. 0.5 between SALE PRICE and LAND SQUARE FEET. 0.846 between SALE PRICE and GROSS SQUARE FEET. (all of these correlations are before cleaning outliers)
We started out by boxploting the SALE PRICE, GROSS SQUARE FEET and LAND SQUARE FEET which were the main parameters in our dataset. As soon as we did that we discovered that our data contained massive outliers (to the degree that we can’t even see the box in the boxplot). Taking another look at our dataset we decided to keep all the prices above 100,000$ and below 4,000,000$. We decided upon those numbers because they seemed to make a sensible sale price (a house sold for 100$ raises some suspicions) and they also created a nice looking boxplot while not loosing too much data.

We dropped from the dataset properties with GROSS SQUARE FEET greater than 20,000 and LAND SQUARE FEET greater than 25,000 for the same reasons.
As we looked at the year built column with a boxplot we discovered outliers again (a building built at year 0 or so). We decided to keep buildings who were built after 1800.
Plotting the distribution plot we discovered a close to normal distribution of the SALE PRICE and GROSS SQUARE FEET fields. Overall the data seemed to be distributed quite well.
The most interesting insight we had from that stage (something which was kind of obvious but nice to see it in the data) is that prices differ by borough. Plotting a boxplot of the SALE PRICE in the 5 different boroughs we saw that Manhattan was much more expensive than the other 4 boroughs and that the differences between these 4 were also substantial. All of this reinforced our belief that location is a crucial parameter in house price evaluation, thus we started a search expedition for any GIS data we could find.
Another discovery we had is that SALE PRICE is much more correlated with GROSS SQUARE FEET when looking inside a certain neighborhood (location again).
By boxploting TOTAL UNITS to SALE PRICE we discovered that TOTAL UNITS have a real strong affect on the price too when inside the neighborhood.
In addition we found an incredibly strong correlation (0.99) between TOTAL UNITS and RESIDENTIAL UNITS and thus we decided to drop the RESIDENTIAL UNITS column.

Checking correlations after the cleaning described above we got a different result than before. Correlation between SALE PRICE and TOTAL UNITS went down to 0.395 and between SALE PRICE and GROSS SQUARE FEET stayed about the same at 0.48. While correlation between SALE PRICE and LAND SQUARE FEET was completely abolished at 0.09. the highest correlation for SALE PRICE was with GROSS SQUARE FEET and TOTAL UNITS. (see appendix-3 for our explanation for this drop in correlation)

Initial Results:

Firstly, since our problem is a regression problem, we had to define what's a successful prediction. Looking at international companies who are in the business of house price evaluation using machine learning algorithms (essentially what we're trying to do) such as Zillow we discovered they use deviation from the actual price (in percentages) as a way to measure the quality of an evaluation. Zillow uses 5%, 10% and 20% as measuring parameters, thus we decided to take 10% as a measure for a successful evaluation.
If a property was sold for X dollars than and our model evaluated it's price between

[X-X/10, X+X/10] we would call it a successful evaluation, otherwise (if the model's prediction is not within that segment) it will be an unsuccessful evaluation.
We wanted to see what percentage of our validation/test set falls our model evaluates successfully.
Our initial models:

1. **Linear Regression:**
   ratio of predictions within 10%: 0.28062636562272397
   Mean Absolute Error: 180153.44962331574
2. **Ridge Regression:**
   ratio of predictions within 10%: 0.2802257829570284
   Mean Absolute Error: 176171.3899014095
3. **Lasso Regression:**
   ratio of predictions within 10%: 0.27982520029133284
   Mean Absolute Error: 176161.52416823225
4. **Random Forest:**
   ratio of predictions within 10%: 0.3083029861616897
   Mean Absolute Error: 166278.0216751639

As you can see Random Forest is the model which preforms best on our dataset, with a "success rate" of about 30% (30% of the validation set is accounted to as a successful prediction) and a MAE of 168,379$ which means that on average our model predicts the house price with an error of 168,000$.
Now, let's check out the feature importance. GROSS SQUARE FEET is the most important feature with importance of 0.27, second we have BLOCK at 0.18. After those two we have SALE DATE and LAND SQUARE FEET and then LOT, borough_2 (which is manhattan's one hot encoding). We can see that the boroughs rank quite high on importance. Also the one-hot encoding of neighborhoods such as "Sunset Park" and "Park Slope" rank high with them, even before the BUILDING CLASS CATEGORY's one-hot encodings.
The fact that BLOCK (which is a locational feature) ranked second along with the high rankings of the boroughs and neighborhoods made us realize that location is indeed an important characteristic in property price evaluation and that leads us to the next step…

GIS & Feature Engineering:

In the beginning we wished to add GIS layers to our dataset. GIS layers, to put simply, are additional datasets that contain relations of coordinates to geographical objects. This additional information, according to our anticipations, should have increased the accuracy

of our prediction, since it is a well-known fact in the world of real estate estimators that the environment of the property, heavily affects its price.
The first challenge, was to translate all the addresses in our data set to a set of coordinates, so we could combine it to the other GIS layers. We chose to use Google's API, since it ensured us a high quality translation. (other services, which are free, translated the addresses with mistakes).
Next, we found an open source datasets, shared by the city of New York and we downloaded them [1]. They contained a lot of GIS layers, but because of time limitations, we chose to go with four. The GIS layers that we chose are - parks, theaters, homeless shelters and museums. We chose those layers, since we believed that they correlate strongly with the quality of life in the areas that these institutes/places could be found. We needed to extract the information about the quality of each area in the city, which partially was encompassed in the GIS layers that we chose. We saw from looking at correlations, that not necessarily the **closeness** of the property, to, for example, a theater, is what affected its price significantly, but the **amount** of theaters in the vicinity of the property are what could help us determine its price. Generally speaking, areas with high theater density, are more prestigious, and therefore households that reside in those places, should be more expansive. Because of that reasoning, we unified the GIS layers In the following matter -
Parks – count of parks in a radius of 500 meter, 1000 meters and 3000 meters. As well as the total acres (area) of the those parks, also, in 500 meters, 1000 meters and 3000 meter.
Theaters – count of theater in a radius of 500 meters, 1000 meters and 3000 meters. In addition, we put the count of theater that belong to Broadway, in a radius of 500 meters, 1000 meters and 3000 meters.
Homeless shelters – only four shelters, we chose to put the distance to then nearest.
Museums – count of museums in a radius of 500 meters, 1000 meters, and 3000 meters.

Final Results:

After adding the GIS layers we first checked correlations between our GIS features and the SALE PRICE and got pretty high correlations. 0.46 with museum_cnt_3000 (almost as high as GROSS SQUARE FEET) and 0.27 with theaters_cnt_3000.
Our results:

1. **Linear Regression:**

    ratio of predictions within 10%: 0.281142586227332

Mean Absolute Error: 162102.57769799914

2. **Ridge Regression:**

ratio of predictions within 10%: 0.2814389000829679

Mean Absolute Error: 162098.8725620929

3. **Lasso Regression:**

ratio of predictions within 10%: 0.28209079056536684

Mean Absolute Error: 162090.05085330724

4. **Random Forest:**

ratio of predictions within 10%: 0.3209079056536684

Mean Absolute Error: 151236.35757378218

Random Forest is still the model which performs the best with a rise of 2% in success rate. Therefore we decided to focus our efforts on it. we tried changing the n_estimators parameter to the model (which controls the number of tress in the forest) and got that setting it to a 100 gives the best result at 0.34 of predictions are in the 10% range and a MAE of 143,498.784 playing with another parameter: max_features which controls the number of features each tree takes into account we discovered that the default gets the best results. Looking at feature importance again we saw a change, GROSS SQUARE FEET is still in the lead at 0.2 but right after that we get museum_cnt_3000 at 0.18 and lng (longitude) ranking right after it. Indeed location played a role in setting the property's price but not as much as we expected.

Thus we resorted to other models who may be able to take location into account a little better. (see appendix-1 for an explanation on why these models)

1. **KNN Regression:**

ratio of predictions within 10%: 0.213583027142349 17

Mean Absolute Error: 218754.93487021452

2. **XGBoost:**

ratio of predictions within 10%: 0.2886689581604836

Mean Absolute Error: 161871.12642601042

3. **Neural Net:**

ratio of predictions within 10%: 0.28149816285409507

Mean Absolute Error: 176123.4891928781

(More statistical measures can be found in the third presentation)

At the end random forest gave the best results. (see appendix-4 for NNs experimentation)

Applications & Insights:

- The house price evaluation problem is a tough problem which still can't be solved by current algorithms, big companies such as Zillow and Urban Compass have tried their power in the problem but still did not get to the accuracy of human appraisals
- House price evaluation problem is important for many reasons. First, it's important to know the value of a house in order to sell it at a fair price. Moreover, appraiser are people whose sole job is to appreciate the value of the house, thus an algorithm who could achieve the same results will be incredibly valuable.
- We were able to establish a business partnership with a realtors office in NYC. We agreed that he will assign us a contact person so we can ask questions and achieve better accuracy for our model and in return he will get one year of free use of the algorithm when it's ready. This shows that there is indeed a need for this kind of algorithm/software.
- GIS and price have very high corrs, this means a property's location is very important for its value, though our models couldn't really get much out of the GIS information we added we believe the right people with the right expertise could make it work.
- After conversing with people in the real estate business we found that every country or region have their own real estate quirks which affect the price. For example in Israel balcony area wasn't accounted when calculating the overall area of an apartment, in the mid 1980's that law was changed and balcony area was counted as part of the apartment area. As a result after this turning point properties with larger areas were sold for the same price as properties with smaller areas did before the change. A model which would not have taken this change into account would have been very confused. Another conversation with Brian (the realtor we partnered with) had made it clear that these kinds of quirks also exist in NYC. As a result we started searching our data for inconsistencies, We found out that some properties have different tax classes at present and at the time of their sale, we decided to remove all such properties from our data and indeed our success rate grew by half a percent.

Related Work:

- The city of New York uploaded its 2017 data to kaggle just to see what people would do with it, we took this notebook as reference: https://www.kaggle.com/akosciansky/how-to-become-a-property-tycoon-in-new-york
- In addition there are 4 companies which we know of who are trying to solve the very same problem: Zillow, Urban Compass, Madlan and Skyline, all 4 companies have proprietary rights on their evaluation models and will not disclose them.

Appendix:

app-1: Consulting with a few sources and searching the internet we found out that properties are usually valued by looking at similar properties in the area, therefore we decided to try KNN in addition we heard XGBoost is a good learner, Moreover, in class Prof. Deutsch said that neural networks might perform well on our problem.
Here are the results:

app-2: Two and three family homes are essentially one property which has one “structure” which is partitioned into two or three private homes.
In addition there are two importantly different types of apartment buildings - elevator apartments, walkup apartments. In these two apartment buildings also differ in whether they are rentals or coops. Coops is a situation where multiple people buy the building together, each one gets an apartment and the buyers live there in cooperation (coop is short for cooperation).

app-3: the sudden drop in correlation probably came because the outliers which we cut had large areas and high prices which contributed to a better correlation, these outliers would have misled our model (or, in fact, any model) and it’s good we eradicated them before they caused any damage, even if it means that our correlations had to suffer.

app-4: We assumed NN’s will have a good potential thus we tried many different settings with them, we experienced with the number of layers (tried 2, 3, 4, 5) and the number of neurons in each layer in addition to the activation function of each layer. We saw that 2 hidden layers with relu perform the best, We also noticed that the more layers we add from 2 makes the performance go down. we tried standardizing and normalizing the data which also didn’t help.