Predicting Home Sale Prices in NYC

Pavel Gladkevich (pg2255), Ye Eun Jeong (yej208), Jiawen Wu (jw1562), Duey Xu (dx2028)

Center for Data Science, New York University

December 2020

Abstract

Predicting the price of housing generates significant investment opportunities for organizations that buy, sell, or manage property. In this project, we attempt to develop a model for predicting the sale price of individual New York City properties sold between 2011 and 2018 using data from various sources, including publicly available property sales and appraisal data, and macroeconomic and demographic data. During the analysis, we assess the usefulness of the various features engineered, and compare the performance of different configurations of hyperparameters and regression algorithms.

1. Business Understanding

The New York City real estate market consistently ranks among the most expensive in the world[1]. Its extreme urban density, high cost of living, and high median income all contribute to its status as one of the toughest places to own. It is a market playing host to many players with diverse interests, and knowing how much a property will sell for is an ongoing concern for all involved. For example:

- Real estate developers want to know which locations and property types are becoming more marketable in order to know where and what to build,
- Prospective home buyers want to know whether it is a good time to buy, or whether it's best to rent for a while longer,

[1] Statista Research Department. "Most expensive residential property markets worldwide in 2020" Statista. June, 2020. Accessed Nov 23, 2020. https://www.statista.com/statistics/1040698/most-expensive-property-markets-worldwide/#:~:text=In%202020%2C%20Hong%20Kong%20had,public%20permanent%20housing%20in%202018.

- The city government needs to understand market prices so that it can calculate fair property taxes each year.

With the total value of property traded each year being approximately $50 billion[2], significant opportunities are presented to those who can accurately predict how much properties will sell for in the near future. For a real estate investor, discriminating a too-early purchase from a too-late purchase is the difference between gaining and losing millions. For a first-time home buyer, this means knowing whether to continue bidding on a house that may have already reached or surpassed its market value; certainly an expensive mistake to make. For a city government, understanding property values factors into property tax assessments and budget planning for overall city government services as well as social services that help those in need. In general, we consider this to be a widely relevant business problem with considerable financial stakes. When scanning the landscape for similar tools in our problem space, we found that organizations such as Zillow[3] are already providing property valuation tools to assist consumers in understanding property prices. However, these tools are powered by proprietary algorithms that are not available to the public.

This project's objective was to build models that can accurately predict the sale price of any given NYC home in a given month, taking into account overall macroeconomic conditions, the property's appraisal data provided by the NYC government, and information about the listed property itself. For this project, we chose to focus exclusively on predicting the sales of single unit homes in NYC. Therefore, data on all commercial or public property sales were excluded from analysis, along with multi-unit sales. Given the uniqueness of each home and multiple factors affecting home prices, machine learning is the ideal tool for tackling this problem by not only predicting prices but also identifying features that impact price most.

2. Data Understanding

The following data sources were used in this project to predict home sale prices: NYC Property Appraisals, NYC Property Sales, U.S. Macroeconomic data, NYC socio-economic data, NYC crimes data. Below we provide detail on each of these sources.

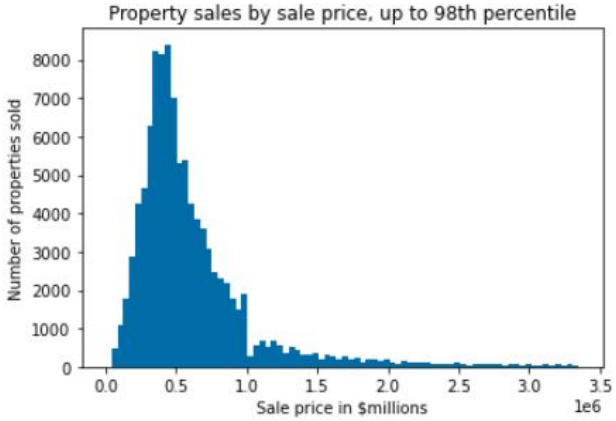
[2] Real Estate Board of New York. "NYC Home Sales Total a Record $50 Billion in 2017" RENBY. February, 2018. Accessed Dec 1, 2020.

[3] United States Home Values. (n.d.). Zillow. Retrieved August 23, 2020 from https://www.zillow.com/home-values/.

2.1. NYC Property Sales

A rolling record of all property sales (tax classes 1, 2, and 4) in NYC is maintained and made publicly available by the NYC Department of Finance. Our starting point was the 689,961 property sales records that occurred between 2011 and 2018 with 21 columns. Each row in this dataset is an instance of a property sale on a specific date (e.g. 610 East 9th Street, sold for $4,000,000 on February 10, 2012). It contains features that could be potentially joined against external datasets, such as borough, neighborhood, building category, zip code, street address, property's square footage, land square footage, and the unique property identifier.

This sales dataset is the main source of property category and includes our target variable for the prediction model, which is the sale price of the property. As seen in the below figures, the distribution of the sale price has a median of $497,000 and a mean of $748,945, hinting at a long right tail in the histogram. This makes sense as NYC is known for having an expensive luxury real estate market. For example, one instance in our dataset was an Upper East Side townhouse that sold for $51 million!

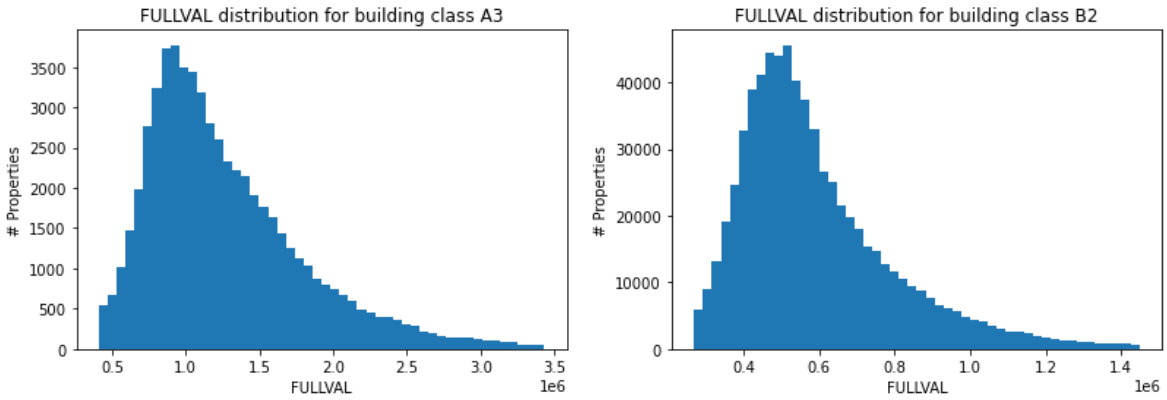


2.2. NYC Property Appraisals

Data on annual NYC property appraisals is also made publicly available by the NYC Department of Finance. The DOF completes a yearly market value assessment of every NYC property in order to determine the property tax that must be paid on it. Each row in this dataset is one property appraisal for a given financial year (e.g. 20 Apricot Court, appraised at $895,000 for FY16/17). The catalogue of property types appraised is comprehensive, covering over 200 unique building classes ranging from small residential homes to large

commercial structures and public interest buildings. Given the completeness of this data, and the other interesting property-related features it contains (e.g. number of stories, building depth and width), this was considered a highly valuable data source to use.

An analysis of appraisal records across the different building class categories revealed that Classes A (one family dwellings), B (two family dwellings), and R (condominium) are the most common properties in NYC. We also observed that each building class followed its own appraisal value ("FULLVAL") distribution centred at different means. Below we compare the appraisal distributions for building classes A3 ("Large Suburban Residence") and B2 ("2 Family Frame-type Dwelling").



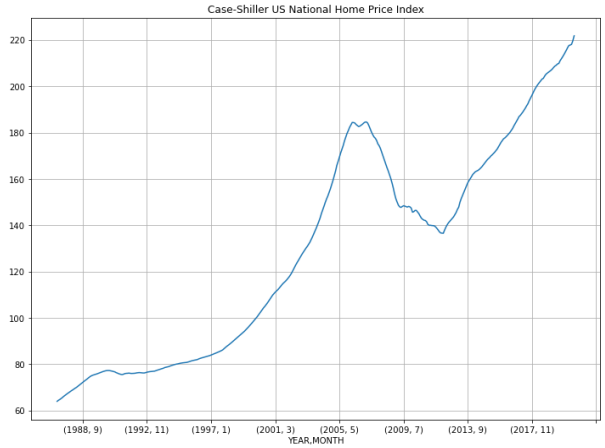
The team's hypothesis was that appraisal value, along with building class, would be among the most informative features for predicting sale prices.

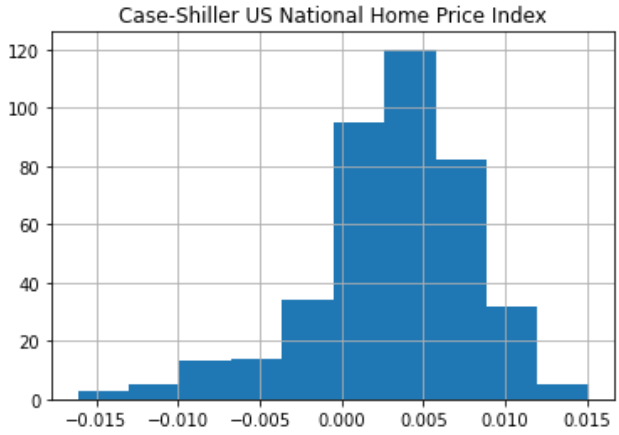
2.3. Macroeconomic, Geographical, Socio-economic Data

Macroeconomic data is tracked and made publicly available by the Research division of the Federal Reserve Bank of St. Louis via their Federal Reserve Economic Database ("FRED"). This database has more than 767,000 economic time series from 102 sources, including national central banks, government agencies, and private companies[4]. In pricing NYC homes, we used a broad set of U.S. economic indicators, some prominent ones going as far back as 1947. Based on domain knowledge, we chose the economic indicators we thought were ones that were highly likely to directly impact real estate, including real GDP, inflation, interest rates, and S&P 500 index.

[4] https://fred.stlouisfed.org/

These time series have various frequencies, ranging from daily to annually. Given we are looking at the NYC housing market, we thought the monthly Case-Shiller US National Home Price Index would be of particular interest.





Looking at the index, it is evident that the housing market crash of 2008 had a large impact on prices. This is confirmed by the the left tail of the histogram of the monthly percent change, as the four worst monthly returns of the entire series were the months between October 2008 and January 2009, when the index dropped at least 1.2% each month, the largest of which was almost a 2.7 standard deviation corresponding to a 1% probability event. As such, we restricted the time interval of our dataset to the period after 2009 which is more representative of "normal" market conditions, specifically from 2011 to 2018.

In addition to US macroeconomic data, we used localized data on populations, demographics, crimes, education, and housing in the five boroughs of New York City as it makes sense intuitively that different conditions of New York City would affect home prices there. This data is publically available through various state and municipal governmental agency websites. One striking feature of the data that we discovered through exploratory data analysis was the distribution of income across the different boroughs. In the 2018 estimate of household incomes, the median household income was $63.799 and the mean was $101,892 across all of New York City, with 11.4% of households earning $200,000 or more. But there is a massive disparity between boroughs, with median and mean household incomes of $38,467 and $57,617, respectively, in the Bronx and only 3.0% of households earning $200,000 or more, compared to median and mean household incomes of $85,066 and $156,633,

respectively, in Manhattan and 22.7% of households earning $200,000 or more.[5] We'd expect almost triple the income between Manhattan and the Bronx would factor into the home prices in those boroughs.

3. Data Preparation

3.1. Combining All Datasets

Having selected our datasets, we proceeded to merge all data into a single data frame using each property's unique identifier and the sale date in the NYC Property Sales dataset as keys. This required some feature engineering since each dataset came in different levels of time and geographical granularity. For example, property sale dates in the Sales dataset were binned into financial years used in the Appraisal dataset in order to join the two together; several million individual instances of arrests and complaints in the NYC Crime dataset had to be binned into discrete time periods and geographies, and then associated with each property.

Macroeconomic features that were reported more than a monthly frequency were averaged into a monthly figure, while those that were reported less than a monthly frequency were imputed assuming linear change in the months between reported times. In addition to the levels of macroeconomic data, we engineered additional features to show the inter-month change, as the amount of change can be more significant than the level itself. A prefix was then added to each feature to identify its source ("v_" for Appraisal data, "s_" for Sales data, "m_" and "d_" for macroeconomic and demographic data). At the end of the merging process, our data frame had 315,069 instances of data and 146 features.

3.2. Data Cleaning, Imputation, and Feature Cleaning

Clearly erroneous sales records were removed on the basis of what constituted a realistic sale price. Any property transfers (with sales price of zero) were removed, and any property sales for less than $50,000 were removed, as this was considered far below what could reasonably be expected in NYC.

[5] https://www1.nyc.gov/site/planning/planning-level/nyc-population/american-community-survey.page

Certain features were found to be highly incomplete, and required some imputation to fill in missing data. "Lot Depth" and "Lot Width", which were missing in different places for both the Sales data and the Appraisal data, were imputed two-ways (one data source was filled in using the other and vice versa) to produce a complete feature set. Lot Depth and Width were then engineered into a single "Lot Area" feature as this was considered sufficient and more interesting as a measure of property size. Unfortunately, sales records for NYC co-ops were so incomplete that they needed to be removed from the analysis altogether. Specifically, because of missing data in the "Total Units" feature, there was no way to distinguish between a single-unit co-op sale and a multi-unit co-op sale, greatly impacting prediction error. Empirical analysis (through simple Linear Regression) indicated that the presence of low-quality co-op data reduced R-squared score by 20%. With no robust way to impute this missing data, the decision was made to exclude co-op sales from the analysis.

To ensure our data was limited to project scope (sales of NYC homes only), the data was thoroughly explored in order to identify and remove any instances that were not strictly related to the sale of a single-unit residential property. Firstly, by filtering on "Building Class Category"[6], we were able to remove commercial, mixed-use and public buildings. Next, we noticed that the data included sales of entire apartment buildings or residential blocks (in some cases containing over 8,000 residential units). For example, the most expensive record we found in the data was the $5 billion sale of Stuyvesant Town in 2015. Such instances were removed by filtering on the "Total Units" feature, which contained the number of residential units contained in each property sale.

As a final cleanup of the columns, 26 more features were removed from the master data frame as they were meaningless to the analysis, or duplicates of other features. Regression algorithms cannot process categorical data, and features on very different scales can impact analysis. To remedy this, the two categorical features of "Neighborhood Tabulation Area" and "Building Class Code" were converted to numerical values via ordinal encoding. Lastly, all input features were transformed by centering and scaling to unit variance. Following Data Preparation our data frame consisted of 103,382 instances and 118 features excluding the target variable.

[6] NYC Building Classes & Building Classification. (n.d.). PropertyShark. Accessed November 11, 2020 from https://www.propertyshark.com/mason/text/nyc_building_class.html.

4. Modeling & Evaluation

4.1. Baseline Data Model and Evaluation Framework

For a baseline model, we used Scikit-learn's Linear Regression algorithm, and compared it to the government's valuation assessment. For our evaluation metrics, we used R-squared, mean absolute error, and mean squared error. R-squared is a measure of the proportion of variance that is explained by the model, so a higher R-squared value implies a better fit. Mean absolute error and mean square error both measure the prediction error as the difference between the actual value and the prediction. The difference between the two is that mean square error squares the prediction error, giving extra penalty for outliers, whereas mean absolute error is less sensitive to outliers. Additionally, from a business perspective, since it represents the dollar amount of the prediction error, the mean absolute error can directly be interpreted as the potential cost of an incorrect prediction. This could translate to potential losses for an investment firm or possible missed property tax revenue opportunity for a local city government agency.

Evaluating our baseline models using the R-squared, mean absolute error, mean square error measures produced the following performance metrics:

	Linear Regression	Valuation Assessment
Mean Absolute Error	268853.197965	312511.494119
Mean Squared Error	838115888923.691162	1126286124080.504639
R^2	0.528284	0.373594

The addition of our features already resulted in substantial improvements to the DOF value. Having set a baseline, we sought to optimize performance and computational effort by:

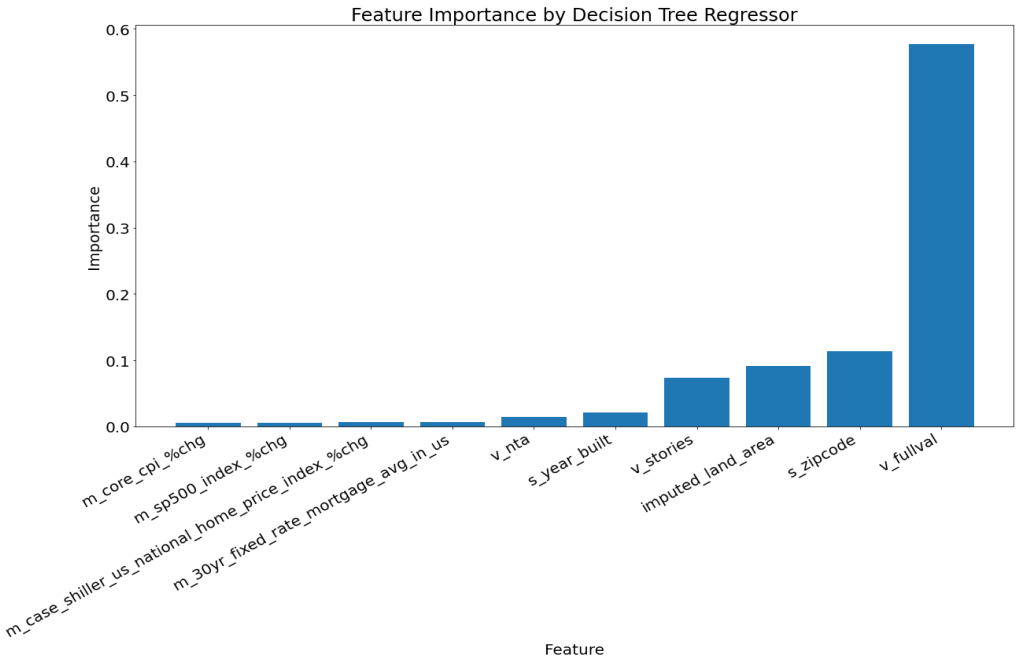
1. Reducing the dimensionality of our data through feature selection to focus only on the most impactful features and decrease the risk of overfitting.
2. Downsampling the data frame to save computational time and resources while empirically testing performance tradeoff.
3. Applying a selection of alternative algorithms and executing a thorough grid search for each algorithm's optimal hyperparameters.

4.2. Feature Selection

With 120 features in the data frame, we conducted feature importance analysis using correlations, decision trees, and PCA to understand how we could reduce the dimensionality of the data.

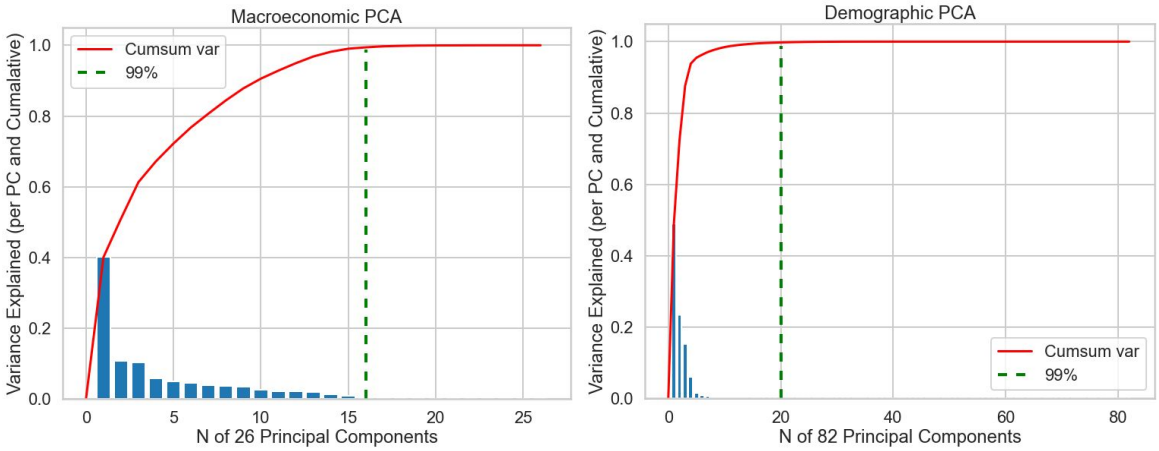
An analysis of feature correlations on the remaining data yielded the following correlation 'heatmap' (Figure 6.1.3) of 118 features. Additionally, the top 20 correlated features (Figure 6.1.2). From the observable structure within the heatmap we see that there are clusters of highly correlated features in macroeconomic and demographic datasets. The 20 most correlated features to sales price are also highly correlated with each other, and come from the same macroeconomic data source. Since highly correlated features can mask each other in some models, it is prudent to apply feature selection methods. By Occam's razor a simpler model is preferable; moreover, dimensionality reduction yields great improvements in speed of learning algorithms, and is justified if it does not result in a substantial decrease in performance.

Decision Tree Regression is commonly used for analyzing feature importance. Feature importance is calculated as the decrease in node impurity provided by a feature weighted by the probability of reaching that feature's node. Using Scikit-learn's Decision Tree Regressor algorithm (6.1.1), the following top ten feature importances were obtained:



While it is expected for the valuation assessment to be first, it is interesting that the zipcode was ranked higher than our area metric. Additionally, the macroeconomic and demographic features (prefixed “m_” and “d_”) collectively carried the lesser portion of feature importances, leading us to the conclusion that a reduction of the dimensionality of our data frame using Principal Component Analysis was warranted.

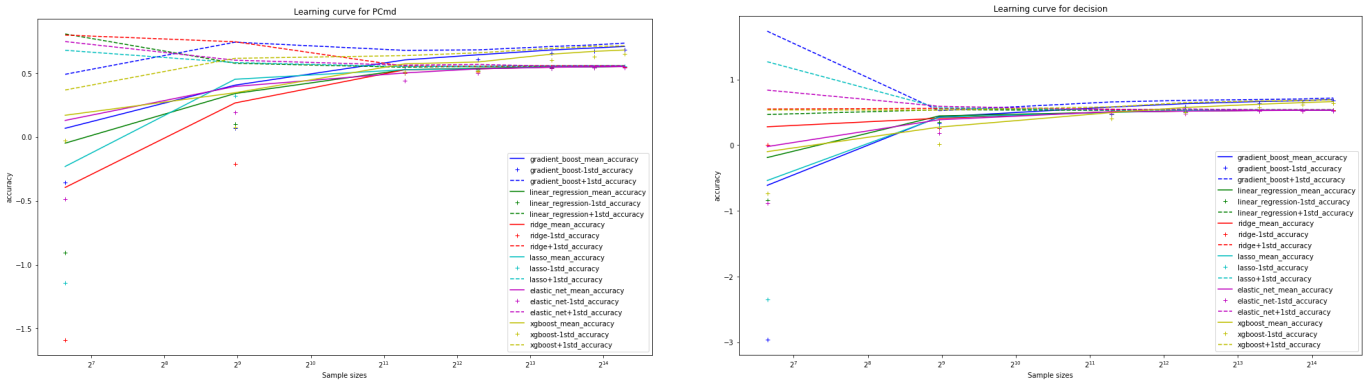
Due to the aforementioned structure discovered in the correlation analysis, we applied grouped PCA for the macroeconomic and separately for the demographic variables. To ascertain a cutoff we took the cumulative sum of the components explained variance (plot macro pca, plot demo pca), and plotted it over the variance explained by each individual component.



The plots here show that for the macroeconomic features, 16 principal components explained 99% of the variation in the target variable. On the other hand, 20 principal components explained 99% of the target variable variation for the demographic features. We selected these 36 principal components as dimension-reduced features of their respective data sets, and decreased the number of features to 46 from 120. There was no significant change in the baseline. The correlation heatmap on the reduced dataset (Figure 6.1.4) shows the elimination of the prior structure and confirms that the PCA successfully reduced the number of highly correlated features.

4.3. Learning Curve Analysis

With over 100,000 instances in our data frame, computational effort and timing had become noticeable. We are aware that there is a marginally decreasing utility to data, so we conducted a learning curve analysis to understand the tradeoff between sample size and model performance.



Based on the resulting learning curves, model performance measured by R-squared improved little past approximately 15,000 instances with only small variances. Additionally, the resulting learning curves showed a lower variability in the PCA dataset as compared to the decision tree reduced dataset. After confirming that baselines were indeed slightly better in the PCA dataset, we moved forward with only using it in our model hyperparameter tuning and comparison.

We decided that roughly 8,000 instances is a point at which there is no marginal utility gain to learning performance, and decided that the smaller sample size would be useful for hyperparameter tuning; we were aware that downsampling the data changes the base rate, and updated our baseline linear regression model R-squared score accordingly to 0.526, which is slightly lower than our original baseline R-squared value as expected, but not by much.

4.4. Creating Leakage-Free Holdout Sets

Prior to starting the modeling and tuning process, when creating training and testing splits, we noticed a possible source of leakage: for data instances with the same unique property identifier (BBL) but with different sale dates, the half of the associated features that are property-dependent would be the same, while the other half of

features that were time-dependent would be variable. To prevent overfitting and control the leakage, we randomly sampled unique BBL IDs instead of instances of data. Then, when splitting the data into train and test sets and placing all instances with the same BBLs into the same dataset.

Out of the leakage-controlled data, we produced a holdout set of 92,482 instances that would be used for the final model evaluation, and a set of 10,900 instances for tuning via cross-validation. We made the size of the tuning dataset consistent with our findings in our learning curve analysis above; since we saw that our target X_train subsample size is 8,192, we knew that in a 4-fold cross-validation setup in which 25% of the data would be set aside as a test set, we would need the total sample size to then be 2^13/(.75) = 10.9k instances. This also addressed a computational bottleneck that we faced when trying to perform hyperparameter optimization by reducing the training dataset.

4.5. Modeling and Hyperparameter Tuning

A family of linear regression models (Ridge, Lasso, Elastic Net) was selected for comparison. Ridge and Lasso regression add penalty terms (L2 and L1 respectively) and can prevent multicollinearity and reduce model complexity, while Elastic Net regression applies a mixture of both regularization approaches. We considered these to be relevant and interesting algorithms to apply given the high dimensionality and known collinearity in our data frame.

Additionally, in addition to the baseline decision tree model, two decision tree-based ensemble learning algorithms (Gradient Boosting Regression, XGBoost Regressor) were selected under the hypothesis that the underlying relationships could be better described through non-linear models, though we are aware that the increased flexibility introduces risk of overfitting. Both are built on individually weak models and sequentially put more weight on instances with wrong predictions and high errors. This allows for the models to learn more difficult cases, which is important for the upper range of sales where the errors are magnified. While both use gradient descent, XGboost additionally computes second-order gradients as similarly to Newton's method which uses the Hessian, and it also uses advanced regularization.

We started by fitting the models with default hyperparameters to assess baseline performance. Prior to tuning the models, we preemptively decided to rule out Support Vector Regressor and Random Forests Regresor

from our candidates due to the extremely long runtime of the training process and significantly subpar performance relative to other models. For the rest of the models, we arrived at the following main hyperparameters that are considered popular candidates for optimization:

Estimator	**Hyperparameters**
XGBoost Regressor	learning_rate: Boosting learning rate n_estimators: Number of gradient boosted trees subsample: Subsample ratio of the training instance
Gradient Boosting Regressor	learning_rate: Boosting learning rate n_estimators: Number of gradient boosted trees subsample: Subsample ratio of the training instance
Lasso Regressor	alpha: L1 penalty
Elastic Net Regressor	alpha: L1 / L2 mixed penalty
Ridge Regressor	alpha: L2 penalty

We used grid search cross-validation to search through all combinations of hyperparameter ranges of each estimator. We first started with a wide range that spanned the possible values for each hyperparameter, then iterated within a narrower window based on the best R-squared score retrieved from each combination. Since we were dealing with a relatively small set of hyperparameters and iterated on the ranges, we chose to not go with random search of hyperparameter combinations.

After some iteration, we arrived at the following best parameters and best R-squared scores when performing 4-fold cross validation on all models:

Estimator	**Best Hyperparameters**
XGBoost Regressor	learning_rate = 0.1 n_estimators = 300 subsample = 0.8
Gradient Boosting Regressor	learning_rate = 0.05 n_estimators = 300 subsample = 0.8
Lasso Regressor	alpha = 22222.223
Elastic Net Regressor	alpha = 0.309
Ridge Regressor	alpha = 20

The learning rate for our optimized ensemble models decreased from 0.3 and 0.1 for XGBoost and Gradient Boosting Regressors, respectively, while the number of trees in the ensemble (n_estimators) increased from 100. The learning rate corresponds to the weights of new features, and a lower rate makes the model less conservative. The number of trees increasing reduces the variability in the construction of trees, and while this could result in overfitting the gradient boosting algorithms are known to be fairly robust in this regard. Subsampling rate also decreased from the default value of 1 to 0.8 for both estimators, meaning both models randomly sampled 80% of the training set in order to prevent overfitting.

For the linear regression models, we observe that some regularization is optimal in each case. While Ridge Regression optimizes at alpha = 20, we find that Lasso Regression optimizes at a much higher alpha. This was likely due to the high dimensionality of the data frame, since an L1 penalty scales with the number of coefficients in the data. This confirmed our hypothesis that there was some multicollinearity still present in the data frame.

4.6. Final Results with Optimized Models

Once we had tuned the hyperparameters of our models, we constructed optimized versions of each model and tested the performance against the never-before-seen holdout set data that we had prepared of 92,482 instances. All models were cross-validated when fitting, in order to account for variations in performance arising from data sampling. The following table summarizes the resulting R-squared and error measurements.

	Decision Tree Regressor (Untuned)	Extreme Gradient Boosting	Gradient Boosting Regressor	Lasso Regressor	Elastic Net Regressor	Ridge Regressor	Linear Regressor (Untuned)
Mean Absolute Error	284576.327	200016.555	210051.735	269901.915	279776.768	276455.158	276981.864
Mean Squared Error	1234888129063.508	596905594605.810	600161723925.394	880373111750.449	902070039434.158	869653042089.012	869216136160.582
R-Squared	0.309	0.674	0.672	0.520	0.508	0.525	0.526

We can see here that the R-squared measure is the highest for the Extreme Gradient Boosting (XGB) Regressor, which is known to be more robust to overfitting. This may be the result of our data not being linear,

which may explain why the tree-based ensemble models performed relatively better over the variations of the Linear Regression family of models, which do not improve drastically in performance from the baseline.

5. Deployment

As with any market, the spread between a property's appraisal value and the market value at any given point in time brings opportunities to each participating party. With our predictive model that would take in a given property's features, its appraisal value, the geographical and socio-economic features associated with the property's surrounding neighborhood, and the overall macroeconomic indicators of the current market, we could predict with reasonable accuracy the market price of the property.

This information could potentially be of benefit to multiple stakeholders in the market, including New York City government, to predict overall market prices by property in each neighborhood for urban planning and producing better estimates of property taxes for budgeting; realtors and real estate firms that need to assess how much a property would sell for; prospective homebuyers who would like to be informed about which properties have the smallest or largest gap from the appraised value.

One point of ethical concern in the model is that it utilizes a property's geographical demographic data, some of which includes racial makeup and the household median income of the neighborhood. Depending on how our predictive model is used, and by whom it is used, it could potentially lead to ethically sensitive repercussions; one can imagine a scenario where real estate firms use our model to survey estimated profits from property sales and potentially evicting tenants, which may disproportionately affect neighborhoods with racial minorities or neighborhoods with lower median household incomes depending on the predicted sale prices from the model.

Since our predictive model is trained on data points that range up to 2018, the results of our model would be subject to concept drift over time, especially in the case of disruptive market-shifting events such as the one we are living through right now due to COVID-19 and the outflux of New York City resident population. This would cause fundamental changes to the underlying features as well as a decay in both prediction performance. Therefore, in realistic deployment scenarios, we would need to constantly feed in the latest data points to train on and continue to update the model in order to make predictions about current-day market prices of properties.

6. Appendix

6.1. Figures

Figure 6.1.1

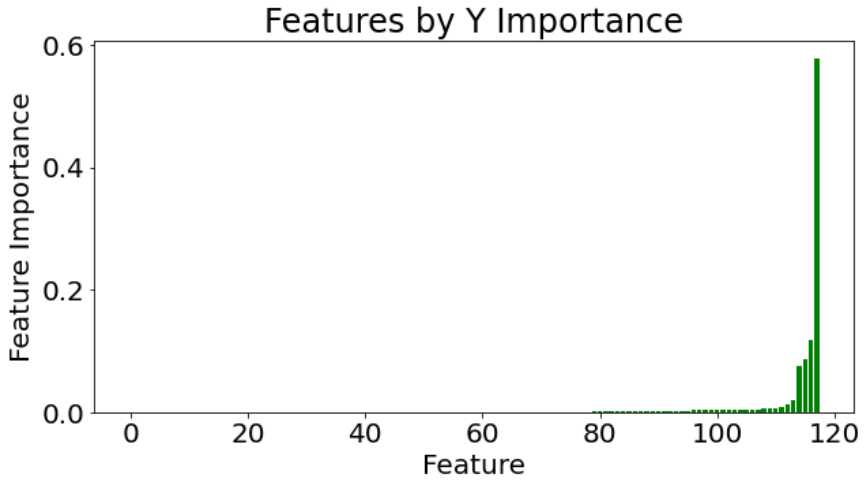


Figure 6.1.2

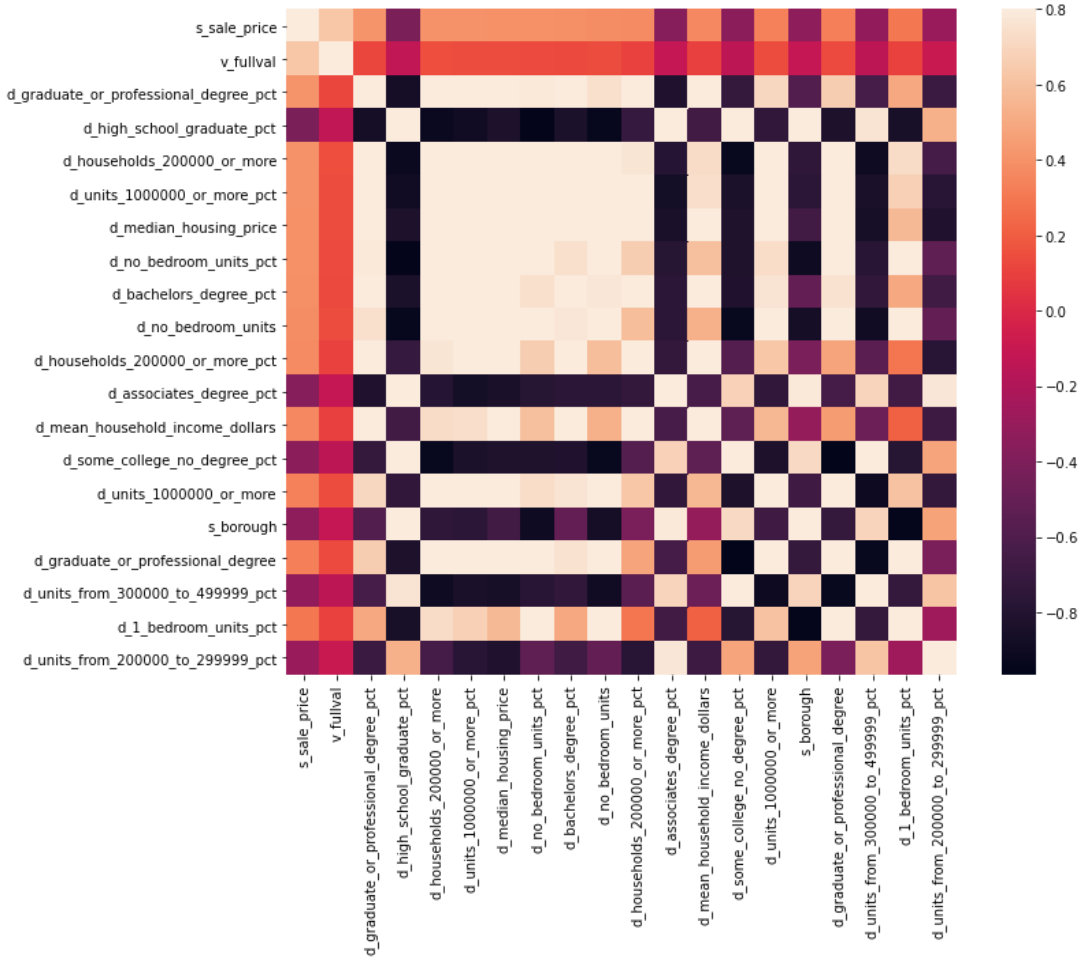


Figure 6.1.3

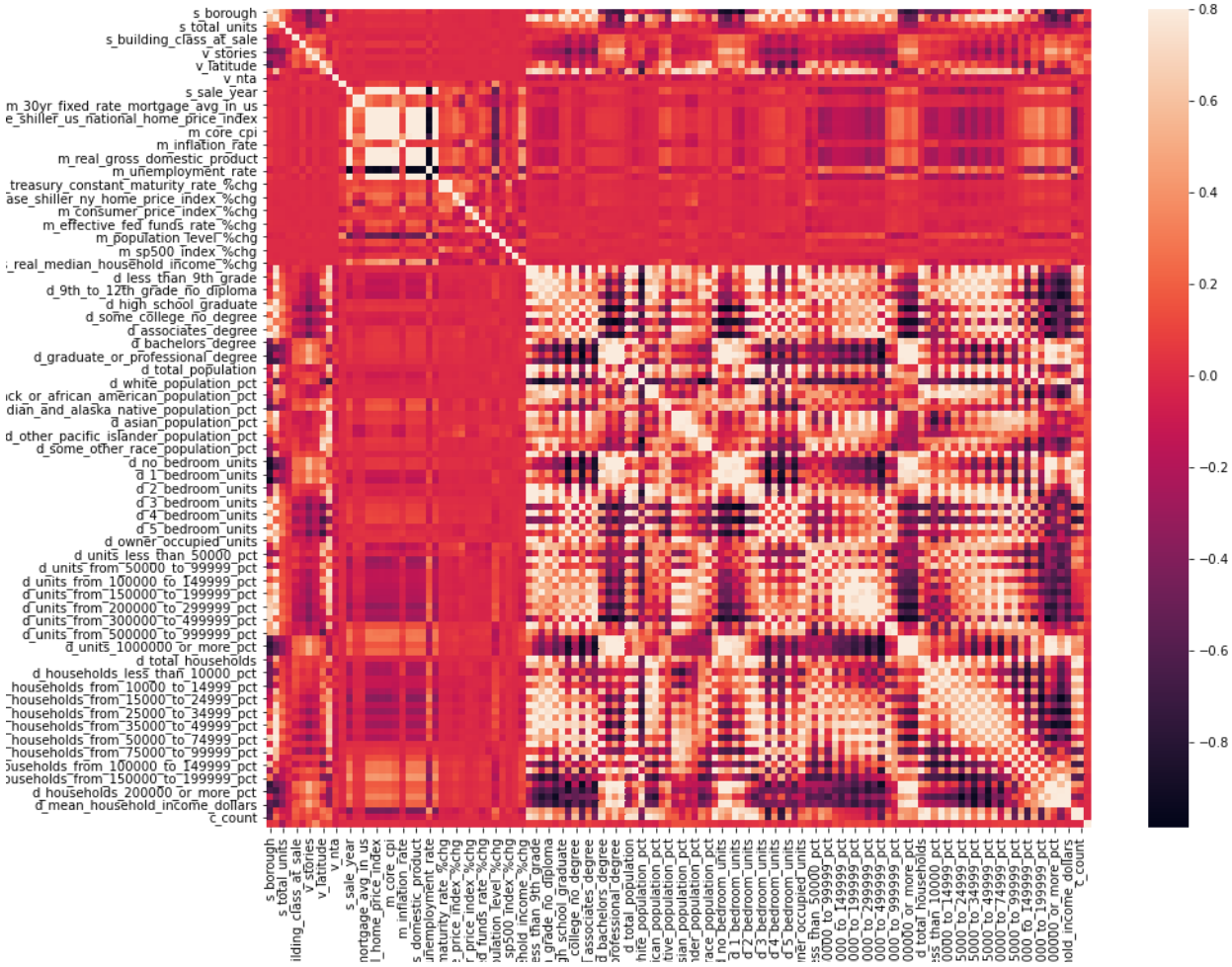
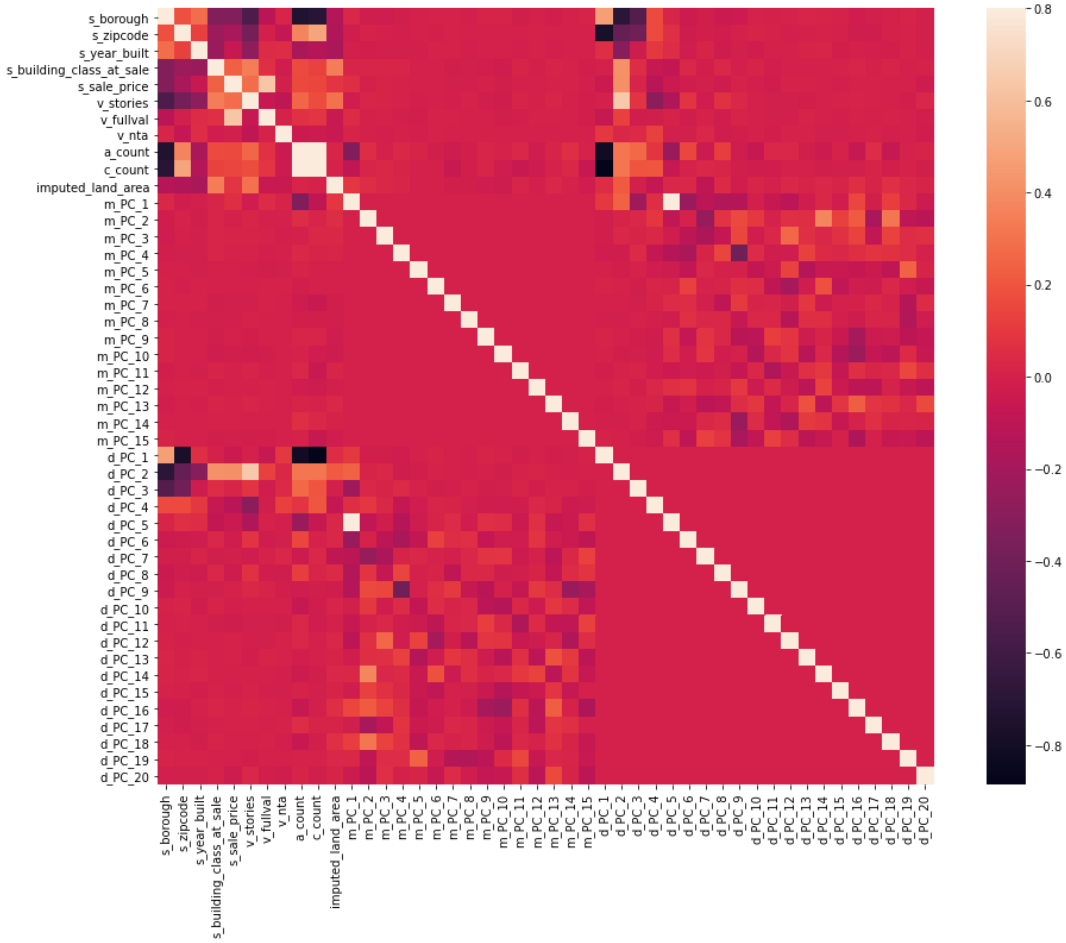

s_borough
s_total_units
s_building_class_at_sale
v_stories
v_latitude
v_nta
s_sale_year
m_30yr_fixed_rate_mortgage_avg_in_us
e_shiller_us_national_home_price_index
m_core_cpi
m_inflation_rate
m_real_gross_domestic_product
m_unemployment_rate
_treasury_constant_maturity_rate_%chg
ase_shiller_ny_home_price_index_%chg
m_consumer_price_index_%chg
m_effective_fed_funds_rate_%chg
m_population_level_%chg
m_sp500_index_%chg
_real_median_household_income_%chg
d_less_than_9th_grade
d_9th_to_12th_grade_no_diploma
d_high_school_graduate
d_some_college_no_degree
d_associates_degree
d_bachelors_degree
d_graduate_or_professional_degree
d_total_population
d_white_population_pct
ıck_or_african_american_population_pct
dian_and_alaska_native_population_pct
d_asian_population_pct
d_other_pacific_islander_population_pct
d_some_other_race_population_pct
d_no_bedroom_units
d_1_bedroom_units
d_2_bedroom_units
d_3_bedroom_units
d_4_bedroom_units
d_5_bedroom_units
d_owner_occupied_units
d_units_less_than_50000_pct
d_units_from_50000_to_99999_pct
d_units_from_100000_to_149999_pct
d_units_from_150000_to_199999_pct
d_units_from_200000_to_299999_pct
d_units_from_300000_to_499999_pct
d_units_from_500000_to_999999_pct
d_units_1000000_or_more_pct
d_total_households
d_households_less_than_10000_pct
_households_from_10000_to_14999_pct
_households_from_15000_to_24999_pct
_households_from_25000_to_34999_pct
_households_from_35000_to_49999_pct
_households_from_50000_to_74999_pct
_households_from_75000_to_99999_pct
ɔuseholds_from_100000_to_149999_pct
ɔuseholds_from_150000_to_199999_pct
d_households_200000_or_more_pct
d_mean_household_income_dollars
c_count
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8

Figure 6.1.4



6.2 Contributions

- Pavel Gladkevich (pg2255): Merging of valuation data; exploratory data analysis; imputation of missing values/feature engineering; ordination of categorical variables; leakage prevention; SVR baseline; PCA and dimensionality reduction, feature selection; XGBoost implementation and hyper parameter tuning; contributing to final report
- Ye Eun Jeong (yej208): Sourcing and merging sales data across all boroughs and years; preliminary feature engineering, exploration, and cleaning of the sales data; data engineering across all datasets to stitch them into one dataframe; exploratory modeling; conducting hyperparameter tuning on all models and running final model evaluation; contributing to writing final report

- Jiawen Wu (jw1562): Source, analyze, and prepare macroeconomic, geographical, socio-economic data for modeling; create correlation heatmaps for feature analysis; conduct learning curve analysis to find optimal sample size for the training set; leakage prevention; contribute to writing final report
- Duey Xu (dx2028): Sourcing and analysis of property appraisal data; feature engineering of crime data; cross-validation code; decision tree regressor feature importance analysis; research and implementation of linear regression models and hyperparameter tuning; assistance with cleaning and preparation of master data frame, collaboration on final report

6.3 Code

All code for this project can be found on: `https://github.com/moongu/dsga1001`