Modeling Sales Price of Condos and Co-ops in Queens

Final project for Math 342W Data Science at Queens College
25 May 2021

By Elizabeth McHugh

Abstract

This report summarizes the results of modeling sales price of condos and co-ops based on a set of features extracted and/or transformed from a raw data set of condos and co-ops in mainland Queens County, NY, using single tree, OLS, and random forest algorithms trained on 31 features. Surprisingly, the error metrics for the OLS model outperformed the other models, both in sample and out of sample, suggesting that sales prices of co-ops and condos in mainland Queens is linear, based on the features used in modeling. The models produced, specifically the OLS model, coupled with industry knowledge in the demographic may serve to aid both real estate professionals, potential buyers, and sellers alike in predicting future sales prices of condos and co-ops in Queens.

1. Introduction

Given the 2016-2017 raw housing data representation (from MLSI) of apartments on mainland Queens, NY, we have been asked to develop a predictive model of future sales prices for listings in the same demographics. The population has been limited to apartments selling for less than one million dollars sold between February of 2016 and February of 2017. As some online services offer estimates of sales, we wish to build a predictive model which can beat the model(s) employed by such an agency. In this case, specifically, we wish to produce better estimates than those produced by Zillow.com's Zestimates.

Now, what is a predictive model, and why would we wish to take time to develop one in this case? A predictive model is an abstract object which uses historical data (observations with known outcomes) to predict the future outcome of a specific phenomenon, or response. In this case, the response we wish to model is the future sales price of a condo or co-op in Queens County, New York, in US dollars, based upon commonly available information which potential buyers would seek–including zip code, additional charges, tax information, community council districts, number and type of rooms (bedrooms, bathrooms, kitchen, dining), square footage, and age of building. In order to model sales prices, it was necessary to clean up the raw data provided by removing "noise", or information which is certainly unrelated to the response we wish to model, impute (or fill in with a "best guess", in this case using the missForest machine learning algorithm) any missing information in the predictive variables of historical data, and then fit and validate single tree, OLS, and random forest models using the cleaned and imputed data and selected or transformed features.

In my case, the OLS model performed consistently better than either the single tree model or the Random Forest (RF) model, which was surprising, as I had initially expected the RF model to do much better than the OLS model.

2. The Data

The data used in this modelling process is, as mentioned in the introduction, raw data harvested via Mturk from MLSI. The raw data represents apartments sold for less than one million dollars in the mainland zip codes of Queens, NY. The original data contained 2230 listings. Unfortunately, 1702 of the observations had no sales data available, so those observations were unable to be used in the modeling process, though they likely contain valuable information which could help build stronger models. That said, the remaining 528 observations were available for use in creating a predictive model for future sales. While this data might be fairly representative of the target population, unfortunately the data only covers one year and may have less predictive ability than desired in relating to future sales, as housing markets can fluctuate wildly over the course of a year, and most certainly can fluctuate even more-so over the course of several years. Thus, there is a great deal of danger in extrapolation here, as the effect of the passage of time (over more than one year) is not taken into account. Equally, the relatively low representation in the data of certain types of apartments may create limited predictive ability for apartments which are atypical or else built since the raw data was collected.

2.2. Featurization

In all, I used 31 features, based on given 528 observations in order to model the historical data and create future predictions. The vast majority of these features were provided in the raw data. However, there were 12 features used in modeling the data which I transformed, in one way or another, from the raw data, not including features which I converted into factors or numeric values from their original designations otherwise. The factors included in my model are as follows:

(a) **Approximate year built**
Continuous variable. From raw data. Year building was built.
Range: 1915-2016 . **Average**: 1962 .

(b) **Cats allowed**
Factor. From raw data. Two levels. Are cats allowed?
"Yes": 46% , "No": 54%

(c) **Community district number**
Continuous variable. From raw data. Which city council district does the apartment pertain to?
Range: 3 - 30. **Average**: 26 .

(d) **Coop or Condo Designation**
Factor. Raw data. Two levels.
"co-op": 63% , "condo": 37%

(e) **Date of Sale**
Continuous variable. (Raw data, coerced to time series.)
Range: 16847 - 17212 . **Average**: 17035 .

(f) **Dining Room Type**
Factor. Raw data. Four levels.
"combo": 46% , "formal": 22% , "other": 10%, "unknown": 23%

(g) **Dogs allowed**
Factor. Raw data (coerced). Two levels.
"Yes": 28% , "No": 72%

(h) **Fuel type**
Factor. Raw data (modified). Six levels.
"electric": 2% , "gas": 57% , "none": 1% , "oil": 34%, "other": 2% , "unknown": 5%

(i) **Garage exists**
Factor. Raw data (modified). Two levels.
"TRUE": 82% , "FALSE": 18%

(j) **Kitchen type**
Factor. Raw data (modified). Four levels.
"combo": 1% , "eat in": 15%, "efficiency": 40%, "unknown": 43%

(k) **Number of bedrooms**
Continuous variable.
Raw data. Number of bedrooms in apartment.
Range: 0 - 3 . **Average**: 1.5 .

(l) **Number of floors in the building**
Continuous variable.
Raw data.
Range: 1 - 34 . **Average**: 7 .

(m) **Number of full bathrooms**
Continuous variable. Raw data.
Range: 1 - 3 . **Average**: 1 .

(n) **Number of total rooms**
Continuous variable. Raw data.
Range: 1 - 8. **Average**: 4.

(o) **Percent tax deductible**
Continuous variable. Raw data. Percent of sale which is tax deductible.
Range: 20. **Average**: 65 .

(p) **Square footage**
Continuous variable. Raw data. Number of square ft. in apartment.
Range: 375 - 6215 . **Average**: 849 .

(q) **Total taxes**
Continuous variable. Raw data. Property taxes.
Range: 11 - 9300 . **Average**: 2525 .

(r) **Walk score**
Continuous variable. Raw data. Unsure of what this factor does.
Range: 15 - 99 . **Average**: 85 .

(s) **Number of bathrooms**
Continuous variable. Total number of bathrooms. Computed from number of full bathrooms plus number of half bathrooms (in raw data).
Range: 1 - 3.5 . **Average**: 1 .

(t) **Month of the year**
Continuous variable. Extracted from raw data.
Range: 1 - 12 . **Average**: 7 .

(u) **Day of the Week**
Continuous variable. Extracted from raw data.
Range: 1 - 6 . **Average**: 4 .

(v) **Day of the Month**
Continuous variable. Extracted from raw data.
Range: 1 - 31 . **Average**: 16 .

(w) **Year**
Continuous variable. Extracted from raw data.
Range: 2016 - 2017 . **Average**: 2016 .

(x) **Zip code (as a factor)**
Factor. Extracted from raw data.
47 levels, corresponding to Queens mainland zip codes. Percentage not computed.

(y) **Total taxes missing**
Dummy factor catching if total taxes were missing, and thus imputed.
"FALSE": 19% , "TRUE": 81%

(z) **Log of total additional charges**
Continuous variable. The log value of total additional charges (not taxes). Created from raw data.
Range: 1 - 4804 . **Average**: 734.8 .

(aa) **Number of missing**
Continuous variable. Total number of missing markers in the dummified categories (plus 10). Created.
Range: 10 - 16 . **Average**: 12 .

(bb) **Bedroom to Square foot ratio**
Continuous variable. Ratio of number of bedrooms to square foot. Computed.
Range: 0 - 0.003695 . **Average**: 0.01658 .

(cc) **Bedrooms to bathrooms ratio**
Continuous variable. Ratio of bedrooms to bathrooms.
Range: 0 - 3 . **Average**: 1 .

2.3. Errors and Missingness

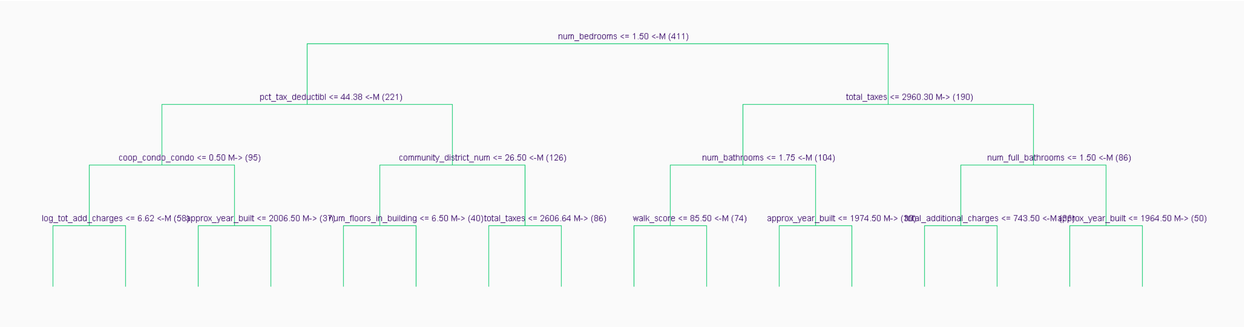
Within the data there were a number of errors caught and corrected as well as a lot of missingness.
Among the errors caught and corrected (which are clearly marked in the attached code) were one observation with the zip code in the wrong column as well as one observation with the year built in the "kitchen type" column. These errors I corrected manually.
When it comes to missingness, there is a good deal of missingness across factors. Most notably, there is greater than 50% missingness in the following factors: total taxes, square footage, and percent tax deductible. Likewise, there was a good deal of missingness among potentially related factors within the data. This missingness was likely related to the type of apartment (coop vs. condo), as the types of charges associated with each type of apartment are different. In general, one should either have maintenance fees or common fees. Instead of worrying about imputing or simply setting missingness to zero, I combined this categories into one variable of total missing values by creating dummy variables for each category and then summing the categories, before dropping those variables all together. As so, I also combined all of these fees into one "total additional charges" variable. These two measures attempted to capture the effects of both the cumulative effect of missing these variables (if a category happened to be missing a lot of these variables, the listing was likely incomplete or poorly promoted, which could easily have an affect on sale price), as well as the effect of total charges above taxes which would be required of the owner.
Apart from this, I used missForest in R to impute the remaining missingness. First, by imputing on my training data, then imputing my test data using the combined
X$_testandX_train.Thishandledtheremainingmissingnessinmydata.Postimputing, Irearrangedandfinc$

3. Modeling

3.1 Regression Tree Modeling

A regression tree model was fit using the YARF package in R. From this singular regression tree, as can be observed in Figure 1, the top 12 predictive features appear to be, in order of levels of the tree:

Figure 1: Single regression tree top layers. Illustrated with YARF.



(a) Number of bedrooms (num_bedrooms)

(b) Percent tax deductible (pct_tax_deductibl)

(c) Total taxes (total_taxes)

(d) Coop or condo (coop_condo)

(e) Community district number (com_dist_num)

(f) Number of bathrooms (num_bathrooms)

(g) Number of full bathrooms (num_full_bathrooms)

(h) Log of total additional charges (log_tot_add_charges)

(i) Approximate year built (approx_year_built)

(j) Number of floors in the building (num_floors_in_building)

(k) Walk score (walk_score)

(l) Total additional charges (total_additional_charges)

Most of these top twelve features are what one might reasonably expect when considering, without the modeling procedure, what features might be important in sales prices. Several of these features are directly related. For instance, the number of bathrooms for each observation includes the number of full bathrooms in that observation, by construction. Also, total additional charges and log of total additional charges are related logarithmically. It makes sense that if one highly correlated feature is important for prediction, then other related features will be, likewise, important for prediction. When thinking about what makes one apartment more (or less desirable than another), the features which come immediately to mind have to do with location, additional fees which will be payed apart from the sale, and the proximity to public transportation and the city. Regardless, there nothing surprising found here. However, as a single tree is known to not have good predictive power, we will continue on, keeping in mind the top predictive features noted within this tree model.

3.2 Linear Modeling

For the linear model, a vanilla OLS model was fit, using the lm() function in R. This OLS model had an in-sample R-squared metric of approximately 0.892 and an RMSE of approximately 64,678. Considering the average sale price of apartments in our supplied set, this model should provide a very decent estimate of future occurances on interpolation. (This can be noted by the in and out of sample metrics seen in the following figure. The OLS model output and summary are also provided in the figures. The most significant factors in this model are fairly straightforward and expected: Condo/coop, number of floors, total additional charges, bedrooms per square foot ratio, and residing in a number of zip codes (whether for good or bad).

Initially, I doubted that an OLS model would be good for prediction, but after modeling the data, and running both in and out of sample tests, I am fairly certain that this OLS model will be the best predictive model of those in this report. It's performance was consistent, even when setting different seeds and choosing different sizes of test and training sets. (Specifics not included in this report. Possibly available at a later date.)

Figure 2: Summary RMSE and R Squared metrics for OLS models (initial trained and final trained)

	RMSE <dbl>	R.Squared <dbl>
In Sample	64677.72	0.8919875
Out-of-Sample	69535.17	0.8625976
Final Model In-Sample	63310.14	0.8940442

Figure 3: OLS Model Output

```
##
## Call:
## lm(formula = y_train ~ ., data = X_train)
##
## Coefficients:
##             (Intercept)         approx_year_built           cats_allowedyes
##              -1.446e+09                 3.831e+02                 1.478e+04
##   community_district_num         coop_condocondo              date_of_sale
##               3.691e+03                 2.213e+05                -1.936e+03
##    dining_room_typeother   dining_room_typeformal   dining_room_typeunknown
##               7.971e+03                 1.898e+04                -3.191e+03
##          dogs_allowedyes             fuel_typegas             fuel_typenone
##              -6.963e+03                 2.068e+04                 5.355e+04
##             fuel_typeoil           fuel_typeother          fuel_typeunknown
##               3.470e+04                 5.636e+04                 2.944e+04
##        garage_existsTRUE        kitchen_typecombo         kitchen_typeeat in
##               6.624e+03                 1.710e+04                -7.028e+02
##   kitchen_typeefficiency             num_bedrooms     num_floors_in_building
##              -9.270e+03                 9.546e+04                 3.255e+03
##       num_full_bathrooms          num_total_rooms         pct_tax_deductibl
##               1.933e+04                 5.545e+03                -1.125e+03
##               sq_footage              total_taxes                walk_score
##              -4.312e+01                 6.032e-02                -7.724e+02
##            num_bathrooms            month_of_year               day_of_week
##               8.853e+04                 6.252e+04                 2.240e+02
##             day_of_month                     year           zip_factor11005
##               1.690e+03                 7.329e+05                 3.230e+04
##          zip_factor11101          zip_factor11102           zip_factor11104
##               1.359e+05                 1.211e+05                 6.448e+04

##          zip_factor11105          zip_factor11106           zip_factor11354
##              -6.936e+03                 1.151e+05                 2.487e+04
##          zip_factor11355          zip_factor11356           zip_factor11357
##              -2.281e+04                -1.402e+05                -5.195e+04
##          zip_factor11358          zip_factor11360           zip_factor11361
##               5.849e+04                -2.299e+04                 1.078e+04
##          zip_factor11362          zip_factor11363           zip_factor11364
##              -5.029e+04                -9.965e+03                -2.801e+04
##          zip_factor11365          zip_factor11367           zip_factor11368
##              -3.576e+04                -2.449e+04                -1.180e+05
##          zip_factor11369          zip_factor11370           zip_factor11372
##              -3.285e+04                -2.531e+04                 6.362e+04
##          zip_factor11373          zip_factor11374           zip_factor11375
##              -8.468e+03                 5.270e+03                 4.913e+04
##          zip_factor11377          zip_factor11378           zip_factor11379
##               3.933e+04                -5.072e+03                -5.762e+04
##          zip_factor11385          zip_factor11413           zip_factor11414
##              -3.403e+04                -6.652e+04                -1.591e+05
##          zip_factor11415          zip_factor11417           zip_factor11421
##              -6.599e+04                -3.246e+05                -8.964e+04
##          zip_factor11422          zip_factor11423           zip_factor11426
##              -7.721e+04                -9.578e+04                -1.097e+04
##          zip_factor11427          zip_factor11432           zip_factor11433
##              -5.776e+04                -8.979e+04                -4.251e+05
##          zip_factor11435  total_taxes_missingTRUE  total_additional_charges
##              -6.729e+04                -1.644e+04                 8.901e+01
##       log_tot_add_charges              num_missing       bedroom_sq_ft_ratio
##              -1.234e+04                 2.224e+02                -1.102e+08
##  bedroom_bathroom_ratio
##               7.974e+04
```

Figure 4: OLS Model Summary Continued

```
Call:
lm(formula = y_train ~ ., data = X_train)

Residuals:
    Min      1Q  Median      3Q     Max 
-219634  -31845   -2881   31266  250622 

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)    
(Intercept)             -1.446e+09  7.100e+09  -0.204 0.838764    
approx_year_built        3.831e+02  3.458e+02   1.108 0.268611    
cats_allowedyes          1.478e+04  1.113e+04   1.328 0.185169    
community_district_num   3.691e+03  1.324e+03   2.787 0.005620 ** 
coop_condocondo          2.213e+05  3.160e+04   7.003 1.40e-11 ***
date_of_sale            -1.936e+03  9.823e+03  -0.197 0.843863    
dining_room_typeother    7.971e+03  1.285e+04   0.621 0.535351    
dining_room_typeformal   1.898e+04  9.563e+03   1.985 0.047963 *  
dining_room_typeunknown -3.191e+03  9.207e+03  -0.347 0.729130    
dogs_allowedyes         -6.963e+03  1.253e+04  -0.556 0.578740    
fuel_typegas             2.068e+04  2.555e+04   0.809 0.418889    
fuel_typenone            5.355e+04  5.066e+04   1.057 0.291250    
fuel_typeoil             3.470e+04  2.591e+04   1.340 0.181310    
fuel_typeother           5.636e+04  3.856e+04   1.462 0.144757    
fuel_typeunknown         2.944e+04  2.961e+04   0.994 0.320919    
garage_existsTRUE        6.624e+03  1.107e+04   0.599 0.549906    
kitchen_typecombo        1.710e+04  3.135e+04   0.546 0.585745    
kitchen_typeeat in      -7.028e+02  3.070e+04  -0.023 0.981749    
kitchen_typeefficiency  -9.270e+03  3.058e+04  -0.303 0.761969    
num_bedrooms             9.546e+04  3.496e+04   2.730 0.006665 ** 
num_floors_in_building   3.255e+03  8.449e+02   3.853 0.000140 ***
num_full_bathrooms       1.933e+04  3.283e+04   0.589 0.556435    
num_total_rooms          5.545e+03  6.054e+03   0.916 0.360356    
pct_tax_deductibl       -1.125e+03  1.188e+03  -0.947 0.344198    
sq_footage              -4.312e+01  1.593e+01  -2.708 0.007126 ** 
total_taxes              6.032e-02  5.232e+00   0.012 0.990808    
walk_score              -7.724e+02  4.442e+02  -1.739 0.083034 .  
num_bathrooms            8.853e+04  4.986e+04   1.776 0.076701 .  
month_of_year            6.252e+04  3.000e+05   0.208 0.835036    
day_of_week              2.240e+02  2.563e+03   0.087 0.930399    
day_of_month             1.690e+03  9.863e+03   0.171 0.864021    
year                     7.329e+05  3.603e+06   0.203 0.838954    
zip_factor11005          3.230e+04  5.062e+04   0.638 0.523854    
zip_factor11101          1.359e+05  4.891e+04   2.778 0.005772 ** 
zip_factor11102          1.211e+05  4.656e+04   2.600 0.009740 ** 
zip_factor11104          6.448e+04  6.015e+04   1.072 0.284513    
zip_factor11105         -6.936e+03  7.212e+04  -0.096 0.923448    
zip_factor11106          1.151e+05  4.789e+04   2.403 0.016828 *  
zip_factor11354          2.487e+04  3.110e+04   0.800 0.424450    
zip_factor11355         -2.281e+04  3.271e+04  -0.697 0.486029    
zip_factor11356         -1.402e+05  5.804e+04  -2.415 0.016266 *  
zip_factor11357         -5.195e+04  3.099e+04  -1.677 0.094580 .  
zip_factor11358          5.849e+04  7.133e+04   0.820 0.412805    
zip_factor11360         -2.299e+04  2.990e+04  -0.769 0.442394    
zip_factor11361          1.078e+04  3.457e+04   0.312 0.755433    
zip_factor11362         -5.029e+04  3.022e+04  -1.664 0.097005 .  
zip_factor11363         -9.965e+03  3.602e+04  -0.277 0.782235    
zip_factor11364         -2.801e+04  3.046e+04  -0.919 0.358523    
zip_factor11365         -3.576e+04  3.856e+04  -0.927 0.354343    
zip_factor11367         -2.449e+04  3.103e+04  -0.789 0.430474    
zip_factor11368         -1.180e+05  3.398e+04  -3.471 0.000586 ***
zip_factor11369         -3.285e+04  4.819e+04  -0.682 0.495896    
zip_factor11370         -2.531e+04  5.352e+04  -0.473 0.636595    
zip_factor11372          6.362e+04  3.119e+04   2.040 0.042185 *  
zip_factor11373         -8.468e+03  3.525e+04  -0.240 0.810306    
zip_factor11374          5.270e+03  3.230e+04   0.163 0.870479    
zip_factor11375          4.913e+04  2.972e+04   1.653 0.099291 .  
zip_factor11377          3.933e+04  3.580e+04   1.099 0.272754    
zip_factor11378         -5.072e+03  7.038e+04  -0.072 0.942599    
zip_factor11379         -5.762e+04  4.103e+04  -1.404 0.161132    
zip_factor11385         -3.403e+04  4.790e+04  -0.710 0.477923    
zip_factor11413         -6.652e+04  7.065e+04  -0.942 0.347071    
zip_factor11414         -1.591e+05  2.953e+04  -5.387 1.36e-07 ***
zip_factor11415         -6.599e+04  3.128e+04  -2.109 0.035650 *  
zip_factor11417         -3.246e+05  7.483e+04  -4.338 1.91e-05 ***
zip_factor11421         -8.964e+04  4.112e+04  -2.180 0.029971 *  
zip_factor11422         -7.721e+04  4.512e+04  -1.711 0.087971 .  
zip_factor11423         -9.578e+04  3.391e+04  -2.825 0.005015 ** 
zip_factor11426         -1.097e+04  4.492e+04  -0.244 0.807126    
zip_factor11427         -5.776e+04  3.585e+04  -1.611 0.108115    
```

Figure 5: OLS Model Summary

```
zip_factor11427          -5.776e+04  3.585e+04  -1.611 0.108115
zip_factor11432          -8.979e+04  3.360e+04  -2.672 0.007913 **
zip_factor11433          -4.251e+05  7.437e+04  -5.716 2.43e-08 ***
zip_factor11435          -6.729e+04  3.392e+04  -1.984 0.048113 *
total_taxes_missingTRUE  -1.644e+04  2.857e+04  -0.575 0.565357
total_additional_charges  8.901e+01  1.760e+01   5.058 7.03e-07 ***
log_tot_add_charges      -1.234e+04  4.101e+03  -3.010 0.002814 **
num_missing               2.224e+02  3.919e+03   0.057 0.954790
bedroom_sq_ft_ratio      -1.102e+08  2.232e+07  -4.936 1.27e-06 ***
bedroom_bathroom_ratio    7.974e+04  3.632e+04   2.195 0.028837 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 64680 on 332 degrees of freedom
Multiple R-squared:  0.892,     Adjusted R-squared:  0.8666
F-statistic: 35.15 on 78 and 332 DF,  p-value: < 2.2e-16
```

3.3 Random Forest Modeling

4. Performance Results for your Random Forest Model

Figure 6: OOB Output for randomForest and YARF RF models

```
rf_mod
```

```
##
## Call:
##  randomForest(formula = y_train ~ ., data = X_train, ntree = 6000,      mtry = 25)
##                Type of random forest: regression
##                      Number of trees: 6000
## No. of variables tried at each split: 25
##
##           Mean of squared residuals: 5977895159
##                     % Var explained: 80.89
```

```
rf_mod_YARF
```

```
## YARF v1.1 for regression
## Missing data feature ON.
## 6000 trees, training data n = 411 and p = 87
## Model construction completed within 0.93 minutes.
## OOB results on all observations:
##   R^2: 0.77309
##   RMSE: 84254.63
##   MAE: 58324.4
##   L2: 2.917625e+12
##   L1: 23971329
```

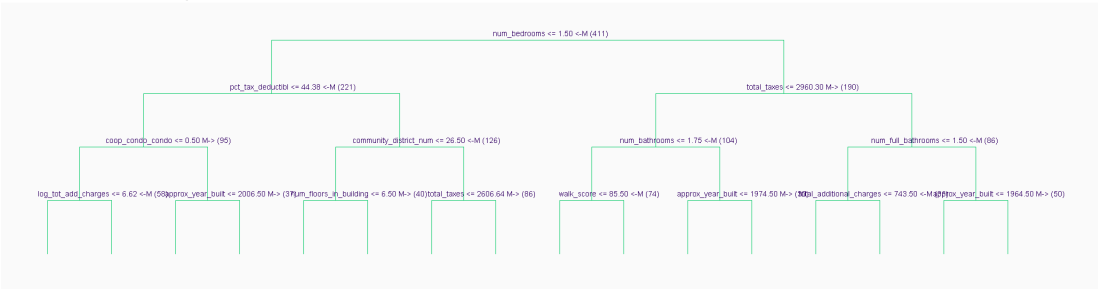
5. Discussion

Acknowledgments

I would like to acknowledge my two amazing children, who not only put up with me being "off" for a week while I worked on this project, but also helped to solve operator errors with my current computing device. I would also like to give credit to my classmate, Janine Lim, who assisted in my completion of this project by hearing out my applied math insecurities enough for me to "get back to work".

Appendix A: Tables, Visualizations, and Figures

3.1 YARF Single Tree



Appendix B: Code

Code used in project (also available on github):

Final Project

Elizabeth McHugh

5/22/2021

Contents

```
if (!require("pacman")){install.packages("pacman")}
```

```
## Loading required package: pacman
## Warning: package 'pacman' was built under R version 4.0.5
```

```
pacman::p_load(knitr, randomForest, dplyr, tidyverse, ggplot2, missForest, stats, readr,
```

##Import and Clean Data Set (2.1)##

Import the data set from drive.

```
library(readr)
housing_data_2016_2017 <- read_csv("C:\\Users\\twiz0\\Downloads\\housing_data_2016_2017.
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   Keywords = col_logical(),
##   MaxAssignments = col_double(),
##   AssignmentDurationInSeconds = col_double(),
##   AutoApprovalDelayInSeconds = col_double(),
##   NumberOfSimilarHITs = col_logical(),
##   LifetimeInSeconds = col_logical(),
##   RejectionTime = col_logical(),
##   RequesterFeedback = col_logical(),
##   WorkTimeInSeconds = col_double(),
##   approx_year_built = col_double(),
##   community_district_num = col_double(),
##   num_bedrooms = col_double(),
##   num_floors_in_building = col_double(),
##   num_full_bathrooms = col_double(),
```

```
##   num_half_bathrooms = col_double(),
##   num_total_rooms = col_double(),
##   pct_tax_deductibl = col_double(),
##   sq_footage = col_double(),
##   walk_score = col_double(),
##   url = col_logical()
## )
## i Use `spec()` for the full column specifications.
```

```
## Warning: 758 parsing failures.
##  row col           expected
## 1473 url 1/0/T/F/TRUE/FALSE http://www.mlsli.com/homes-for-sale/10-Station-Sq-Forest-
## 1474 url 1/0/T/F/TRUE/FALSE http://www.mlsli.com/homes-for-sale/10-01-162nd-St-Beechh
## 1475 url 1/0/T/F/TRUE/FALSE http://www.mlsli.com/homes-for-sale/100-10-67th-Rd-Forest
## 1476 url 1/0/T/F/TRUE/FALSE http://www.mlsli.com/homes-for-sale/100-25-Queens-Blvd-Fo
## 1477 url 1/0/T/F/TRUE/FALSE http://www.mlsli.com/homes-for-sale/10-11-162nd-St-Beechh
## .... ... .................. .........................................................
## See problems(...) for more details.
```

```
#View(housing_data_2016_2017)
```

#Split Data#

Split Test and Training Sets. Retain 20% of data for testing.

```
set.seed(479)

#Split 20% Test/ 80% Train
K = 5

test_indices = sample(1 : nrow(housing_data_2016_2017), round(nrow(housing_data_2016_201
train_indices = setdiff(1 : nrow(housing_data_2016_2017), test_indices)

housing_data_test = housing_data_2016_2017[test_indices, ]
housing_data_train = housing_data_2016_2017[train_indices, ]

#View(housing_data_train)
#View(housing_data_test)

#summary(housing_data_train)
#summary(housing_data_test)

#Count observations with missing target variable.
sum(is.na(housing_data_2016_2017$sale_price))
```

```
## [1] 1702
```

#Initial (Pre-Imputation) Data Clean-up (2.2)#

Data Clean-Up on Training Set

```
#Remove obviously unnecessary columns, reorder with objective variable (sale price) at
housing_data_train = housing_data_train %>%
  select(-(1:28), -url) %>%
    select(sale_price, everything()) %>%
      filter(!is.na(sale_price)) %>%
        select(-listing_price_to_nearest_1000)

#Unformat all prices
housing_data_train = housing_data_train %>%
  mutate(sale_price = parse_number(sale_price)) %>%
  mutate(common_charges = parse_number(common_charges)) %>%
  mutate(maintenance_cost = parse_number(maintenance_cost)) %>%
  mutate(parking_charges = parse_number(parking_charges)) %>%
  mutate(total_taxes = parse_number(total_taxes))

#Add feature for total bathrooms (whole plus half).
housing_data_train = housing_data_train %>%
  mutate(num_half_bathrooms = replace(num_half_bathrooms, is.na(num_half_bathrooms), 0))
  mutate(num_bathrooms = num_full_bathrooms + 0.5 * num_half_bathrooms)

#Separate dates sold as year, date, month, weekdays, and days of month.
housing_data_train = housing_data_train %>%
  mutate(date_of_sale = as_date(mdy(date_of_sale))) %>%
  mutate(month_of_year = month(date_of_sale)) %>%
  mutate(day_of_week = wday(date_of_sale)) %>%
  mutate(day_of_month = as.numeric(day(date_of_sale))) %>%
  mutate(year = year(date_of_sale)) %>%
    mutate(date_of_sale = as.numeric(date_of_sale))

#Extract zip codes from addresses.
housing_data_train = housing_data_train %>%
  mutate(zip_numeric = as.numeric(str_sub(full_address_or_zip_code, -5,-1))) %>%
  mutate(zip_factor = as.factor(zip_numeric))
```

```
## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion
```

```
#Create dummy variables for non-factor variables with potentially significant missing
housing_data_train = housing_data_train %>%
  mutate(common_charges_missing = as.factor(is.na(common_charges))) %>%
    mutate(common_charges = ifelse(is.na(common_charges), 0, common_charges)) %>%
  mutate(approx_year_built_missing = as.factor(is.na(approx_year_built))) %>%
  mutate(maintenance_cost_missing = as.factor(is.na(maintenance_cost))) %>%
    mutate(maintenance_cost = ifelse(is.na(maintenance_cost), 0, maintenance_cost)) %>%
  mutate(num_floors_in_building_missing = as.factor(is.na(num_floors_in_building))) %>%
```

```
  mutate(parking_charges_missing = as.factor(is.na(parking_charges))) %>%
    mutate(parking_charges = ifelse(is.na(parking_charges), 0, parking_charges)) %>%
  mutate(pct_tax_deductibl_missing = as.factor(is.na(pct_tax_deductibl))) %>%
  mutate(sq_footage_missing = as.factor(is.na(sq_footage))) %>%
  mutate(total_taxes_missing = as.factor(is.na(total_taxes)))

#Coerce yes/no to factors.
housing_data_train = housing_data_train %>%
  mutate(cats_allowed = factor(cats_allowed)) %>%
  mutate(dogs_allowed = factor(dogs_allowed))

#Garage exists to factor.
housing_data_train = housing_data_train %>%
  mutate(garage_exists = as.factor(!is.na(garage_exists)))

#Factorize character variables and set NA to "unknown" factor.
housing_data_train = housing_data_train %>%
  mutate(dining_room_type = replace_na(dining_room_type, "unknown")) %>%
  mutate(dining_room_type = factor(dining_room_type)) %>%
  mutate(coop_condo = factor(coop_condo, ordered = FALSE)) %>%
  mutate(fuel_type = ifelse(fuel_type %in% c("other", "Other"), "other", fuel_type)) %>%
  mutate(fuel_type = ifelse(is.na(fuel_type), "unknown", fuel_type)) %>%
  mutate(fuel_type = factor(fuel_type)) %>%
  mutate(kitchen_type = ifelse(kitchen_type %in% c("eat in", "Eat In", "Eat in"), "eat i
  mutate(kitchen_type = replace_na(kitchen_type, "unknown")) %>%
  mutate(kitchen_type = ifelse(kitchen_type == "Combo", "combo", kitchen_type)) %>%
  mutate(kitchen_type = as.factor(kitchen_type))

#Take care of factors with only a few observations.
housing_data_train = housing_data_train %>%
  mutate(dining_room_type = recode(dining_room_type, "dining area" = "other")) %>%
  mutate(kitchen_type = recode(kitchen_type, "1955" = "unknown"))

#Fill in singular missing values in train data (found zip manually in raw data)
housing_data_train$zip_numeric[2] = 11354
housing_data_train$zip_factor[2] = "11354"

#Remove full address, model type, and date of sale
housing_data_train = housing_data_train %>%
  mutate(total_additional_charges = common_charges + maintenance_cost + parking_charges)
  select(-full_address_or_zip_code, -model_type, -common_charges, -parking_charges, -mai

#summary(housing_data_train)
#sapply(housing_data_train, class)
```

Data Clean-Up on Test Set

```
#Remove obviously unnecessary columns, reorder with objective variable (sale price) at
housing_data_test = housing_data_test %>%
  select(-(1:28), -url) %>%
    select(sale_price, everything()) %>%
      filter(!is.na(sale_price)) %>%
        select(-listing_price_to_nearest_1000)

#Unformat all prices
housing_data_test = housing_data_test %>%
  mutate(sale_price = parse_number(sale_price)) %>%
  mutate(common_charges = parse_number(common_charges)) %>%
  mutate(maintenance_cost = parse_number(maintenance_cost)) %>%
  mutate(parking_charges = parse_number(parking_charges)) %>%
  mutate(total_taxes = parse_number(total_taxes))

#Add feature for total bathrooms (whole plus half).
housing_data_test = housing_data_test %>%
  mutate(num_half_bathrooms = replace(num_half_bathrooms, is.na(num_half_bathrooms), 0))
  mutate(num_bathrooms = num_full_bathrooms + 0.5 * num_half_bathrooms)

#Separate dates sold as year, date, month, weekdays, and days of month.
housing_data_test = housing_data_test %>%
  mutate(date_of_sale = as_date(mdy(date_of_sale))) %>%
  mutate(month_of_year = month(date_of_sale)) %>%
  mutate(day_of_week = wday(date_of_sale)) %>%
  mutate(day_of_month = as.numeric(day(date_of_sale))) %>%
  mutate(year = year(date_of_sale)) %>%
    mutate(date_of_sale = as.numeric(date_of_sale))

#Extract zip codes from addresses.
housing_data_test = housing_data_test %>%
  mutate(zip_numeric = as.numeric(str_sub(full_address_or_zip_code, -5,-1))) %>%
  mutate(zip_factor = as.factor(zip_numeric))

#Create dummy variables for non-factor variables with potentially significant missing
housing_data_test = housing_data_test %>%
  mutate(common_charges_missing = as.factor(is.na(common_charges))) %>%
    mutate(common_charges = ifelse(is.na(common_charges), 0, common_charges)) %>%
  mutate(approx_year_built_missing = as.factor(is.na(approx_year_built))) %>%
  mutate(maintenance_cost_missing = as.factor(is.na(maintenance_cost))) %>%
    mutate(maintenance_cost = ifelse(is.na(maintenance_cost), 0, maintenance_cost)) %>%
  mutate(num_floors_in_building_missing = as.factor(is.na(num_floors_in_building))) %>%
  mutate(parking_charges_missing = as.factor(is.na(parking_charges))) %>%
```

```r
    mutate(parking_charges = ifelse(is.na(parking_charges), 0, parking_charges)) %>%
  mutate(pct_tax_deductibl_missing = as.factor(is.na(pct_tax_deductibl))) %>%
  mutate(sq_footage_missing = as.factor(is.na(sq_footage))) %>%
  mutate(total_taxes_missing = as.factor(is.na(total_taxes)))

#Coerce yes/no to factors.
housing_data_test = housing_data_test %>%
  mutate(cats_allowed = factor(cats_allowed)) %>%
  mutate(dogs_allowed = factor(dogs_allowed))

#Garage exists to factor.
housing_data_test = housing_data_test %>%
  mutate(garage_exists = as.factor(!is.na(garage_exists)))

#Factorize character variables and set NA to "unknown" factor.
housing_data_test = housing_data_test %>%
  mutate(dining_room_type = replace_na(dining_room_type, "unknown")) %>%
  mutate(dining_room_type = factor(dining_room_type)) %>%
  mutate(coop_condo = factor(coop_condo, ordered = FALSE)) %>%
  mutate(fuel_type = ifelse(fuel_type %in% c("other", "Other"), "other", fuel_type)) %>%
  mutate(fuel_type = ifelse(is.na(fuel_type), "unknown", fuel_type)) %>%
  mutate(fuel_type = factor(fuel_type)) %>%
  mutate(kitchen_type = ifelse(kitchen_type %in% c("eat in", "Eat In", "Eat in"), "eat i
  mutate(kitchen_type = replace_na(kitchen_type, "unknown")) %>%
  mutate(kitchen_type = ifelse(kitchen_type == "Combo", "combo", kitchen_type)) %>%
  mutate(kitchen_type = as.factor(kitchen_type))

#Take care of factors with only a few observations.
housing_data_test = housing_data_test %>%
  mutate(dining_room_type = recode(dining_room_type, "dining area" = "other")) %>%
  mutate(kitchen_type = recode(kitchen_type, "1955" = "unknown"))

#Fill in singular missing values easily available manually
housing_data_test = housing_data_test %>%
  mutate(dining_room_type = recode(dining_room_type, "dining area" = "other"))

#Remove full address, model type, and date of sale
housing_data_test = housing_data_test %>%
  mutate(total_additional_charges = common_charges + maintenance_cost + parking_charges)
  select(-full_address_or_zip_code, -model_type, -common_charges, -parking_charges, -mai

#summary(housing_data_train)
#sapply(housing_data_test, class)
```

##Missingness in Features (2.3)

Impute using missForest. Check out line 245 issue.

```
#Impute missing values in training data
housing_data_train_imp = missForest(data.frame(housing_data_train))$ximp
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
```

```
#Impute missing values in test data.
housing_data_test_imp = cbind("sale_price" = NA, housing_data_test[2:ncol(housing_data_t
housing_data_test_train_imp = rbind(housing_data_test_imp, housing_data_train_imp)
housing_data_test_train_imp = missForest(data.frame(housing_data_test_train_imp))$ximp
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
```

```
housing_data_test_imp = housing_data_test_train_imp[1:nrow(housing_data_test_imp), ]
```

Playing with visualizations to consider feature transformations.

```
#Not a linear relationship.
ggplot(housing_data_train_imp) +
  aes(x = log(total_additional_charges), y = sale_price) +
  geom_smooth() +
  geom_jitter()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## Warning: Removed 13 rows containing non-finite values (stat_smooth).
```

final_project_files/figure-latex/unnamed-chunk-7-1.pdf

```
#Note how the relative lack of data in zip codes (just two zips?) below 11300 as well



ggplot(housing_data_train_imp) +
  aes(x = zip_factor, y = sale_price) +
  geom_smooth() +
  geom_jitter()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

final_project_files/figure-latex/unnamed-chunk-7-2.pdf

```
ggplot(housing_data_train_imp) +
  aes(x = zip_numeric, y = sale_price) +
  geom_smooth() +
  geom_jitter()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

final_project_files/figure-latex/unnamed-chunk-7-3.pdf

```
#Visualize effect of interactions between #bedrooms and #bathrooms on sale price
ggplot(housing_data_train_imp) +
  aes(x = (num_bedrooms / num_bathrooms)^2, y = sale_price) +
  geom_smooth() +
  geom_jitter()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

final_project_files/figure-latex/unnamed-chunk-7-4.pdf

```
ggplot(housing_data_train_imp) +
  aes(x = (num_bedrooms / num_bathrooms)^2, y = sale_price) +
  geom_smooth() +
  geom_jitter()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

final_project_files/figure-latex/unnamed-chunk-7-5.pdf

Feature Transformations

Add feature transformations to be included in models.

```
#Training Data Transformations

housing_data_train_imp = housing_data_train_imp %>%
  mutate(log_tot_add_charges = log(total_additional_charges)) %>%
  mutate(log_tot_add_charges = ifelse(log_tot_add_charges == -Inf, 0, log_tot_add_charge
    select(-num_half_bathrooms) %>%
    mutate(num_missing = (as.numeric(common_charges_missing) + as.numeric(approx_year_bu
      select(-common_charges_missing, -approx_year_built_missing, -maintenance_cost_miss

housing_data_train_imp = housing_data_train_imp %>%
  mutate(bedroom_sq_ft_ratio = num_bedrooms / sq_footage) %>%
  mutate(bedroom_bathroom_ratio = num_bedrooms / num_bathrooms) %>%
    select(-zip_numeric)

#Test Data Transformations


housing_data_test_imp = housing_data_test_imp %>%
  mutate(log_tot_add_charges = log(total_additional_charges)) %>%
  mutate(log_tot_add_charges = ifelse(log_tot_add_charges == -Inf, 0, log_tot_add_charge
    select(-num_half_bathrooms) %>%
    mutate(num_missing = (as.numeric(common_charges_missing) + as.numeric(approx_year_bu
      select(-common_charges_missing, -approx_year_built_missing, -maintenance_cost_miss

housing_data_test_imp = housing_data_test_imp %>%
  mutate(bedroom_sq_ft_ratio = num_bedrooms / sq_footage) %>%
  mutate(bedroom_bathroom_ratio = num_bedrooms / num_bathrooms) %>%
    select(-zip_numeric)


#head(housing_data_train_imp)
#head(housing_data_test_imp)
```

Split into X, y test and training sets.

```
X_train = housing_data_train_imp[ , 2:ncol(housing_data_train_imp)]
y_train = housing_data_train_imp[ , 1]

X_test = housing_data_test_imp[ , 2:ncol(housing_data_test_imp)]
y_test = housing_data_test[ ,1]
```

##Regression Tree Modeling (3.1)

Load YARF

```
Sys.setenv(JAVA_HOME = '/usr/lib/jvm/jdk1.8.0_65')

if (!pacman::p_isinstalled(YARF)){
  pacman::p_install_gh("kapelner/YARF/YARFJARs", ref = "dev")
  pacman::p_install_gh("kapelner/YARF/YARF", ref = "dev", force = TRUE)
}
options(java.parameters = "-Xmx4000m")
pacman::p_load(YARF)
```

```
## YARF can now make use of 7 cores.
```

```
library(YARF, YARFJARs)
```

Create one tree model.

```
mod_YARF = YARF(y = y_train, X = X_train, num_trees = 1)
```

```
## YARF initializing with a fixed 1 trees...
## YARF factors created...
## YARF after data preprocessed... 87 total features...
## Beginning YARF regression model construction...done.
## Calculating OOB error...done.
```

```
illustrate_trees(mod_YARF, max_depth = 5, length_in_px_per_half_split = 30, font_size =

mod_YARF
```

```
## YARF v1.1 for regression
## Missing data feature ON.
## 1 trees, training data n = 411 and p = 87
## Model construction completed within 0.02 minutes.
## OOB results on 36.74% of the observations (260 missing):
##   R^2: 0.79675
##   RMSE: 131556.1
##   MAE: 91496.32
##   L2: 2.613356e+12
##   L1: 13815945
```

Tree Metrics? Nope.. Just a free space to check out some things.

```
#housing_data_test
#housing_data_train_imp
```

##Linear Modeling (3.2)

Create OLS Model

```
#summary(X_train)
#str(X_train)
```

```r
mod_ols = lm(y_train ~ ., X_train)
mod_ols
```

```
##
## Call:
## lm(formula = y_train ~ ., data = X_train)
##
## Coefficients:
##             (Intercept)         approx_year_built           cats_allowedyes
##              -1.446e+09                 3.831e+02                 1.478e+04
##  community_district_num           coop_condocondo              date_of_sale
##               3.691e+03                 2.213e+05                -1.936e+03
##   dining_room_typeother    dining_room_typeformal   dining_room_typeunknown
##               7.971e+03                 1.898e+04                -3.191e+03
##         dogs_allowedyes              fuel_typegas             fuel_typenone
##              -6.963e+03                 2.068e+04                 5.355e+04
##            fuel_typeoil            fuel_typeother          fuel_typeunknown
##               3.470e+04                 5.636e+04                 2.944e+04
##        garage_existsTRUE        kitchen_typecombo         kitchen_typeeat in
##               6.624e+03                 1.710e+04                -7.028e+02
##  kitchen_typeefficiency              num_bedrooms    num_floors_in_building
##              -9.270e+03                 9.546e+04                 3.255e+03
##      num_full_bathrooms           num_total_rooms         pct_tax_deductibl
##               1.933e+04                 5.545e+03                -1.125e+03
##               sq_footage               total_taxes                walk_score
##              -4.312e+01                 6.032e-02                -7.724e+02
##           num_bathrooms             month_of_year               day_of_week
##               8.853e+04                 6.252e+04                 2.240e+02
##            day_of_month                      year          zip_factor11005
##               1.690e+03                 7.329e+05                 3.230e+04
##         zip_factor11101           zip_factor11102           zip_factor11104
##               1.359e+05                 1.211e+05                 6.448e+04
##         zip_factor11105           zip_factor11106           zip_factor11354
##              -6.936e+03                 1.151e+05                 2.487e+04
##         zip_factor11355           zip_factor11356           zip_factor11357
##              -2.281e+04                -1.402e+05                -5.195e+04
##         zip_factor11358           zip_factor11360           zip_factor11361
##               5.849e+04                -2.299e+04                 1.078e+04
##         zip_factor11362           zip_factor11363           zip_factor11364
##              -5.029e+04                -9.965e+03                -2.801e+04
##         zip_factor11365           zip_factor11367           zip_factor11368
##              -3.576e+04                -2.449e+04                -1.180e+05
##         zip_factor11369           zip_factor11370           zip_factor11372
##              -3.285e+04                -2.531e+04                 6.362e+04
```

```
##          zip_factor11373          zip_factor11374          zip_factor11375
##               -8.468e+03                5.270e+03                4.913e+04
##          zip_factor11377          zip_factor11378          zip_factor11379
##                3.933e+04               -5.072e+03               -5.762e+04
##          zip_factor11385          zip_factor11413          zip_factor11414
##               -3.403e+04               -6.652e+04               -1.591e+05
##          zip_factor11415          zip_factor11417          zip_factor11421
##               -6.599e+04               -3.246e+05               -8.964e+04
##          zip_factor11422          zip_factor11423          zip_factor11426
##               -7.721e+04               -9.578e+04               -1.097e+04
##          zip_factor11427          zip_factor11432          zip_factor11433
##               -5.776e+04               -8.979e+04               -4.251e+05
##          zip_factor11435   total_taxes_missingTRUE  total_additional_charges
##               -6.729e+04               -1.644e+04                8.901e+01
##      log_tot_add_charges              num_missing       bedroom_sq_ft_ratio
##               -1.234e+04                2.224e+02               -1.102e+08
##   bedroom_bathroom_ratio
##                7.974e+04
```

```r
View(data.frame(coefficients(mod_ols)), "OLS Model Coefficients")
```

OLS In-Sample Metrics

```r
RMSE = summary(mod_ols)$sigma
RMSE
```

```
## [1] 64677.72
```

```r
r_squared = summary(mod_ols)$r.square

View(data.frame(cbind("R Squared" = r_squared, "RMSE" = RMSE)),  title = "OLS Model In-S
```

##Random Forest Modeling (3.3)

Create RF Model

```r
rf_mod = randomForest(y_train ~ . , data = X_train, ntree = 6000, mtry = 25)

rf_mod_YARF = YARF(X = X_train, y = y_train, num_trees = 6000, mtry = 25)
```

```
## YARF initializing with a fixed 6000 trees...
## YARF factors created...
## YARF after data preprocessed... 87 total features...
## Beginning YARF regression model construction...done.
## Calculating OOB error...done.
```

##Performance Results for Random Forest (4)

RF Metrics

```r
rf_mod
```

```
##
## Call:
##  randomForest(formula = y_train ~ ., data = X_train, ntree = 6000,      mtry = 25)
##                Type of random forest: regression
##                      Number of trees: 6000
## No. of variables tried at each split: 25
##
##           Mean of squared residuals: 5977895159
##                     % Var explained: 80.89
```

```r
rf_mod_YARF
```

```
## YARF v1.1 for regression
## Missing data feature ON.
## 6000 trees, training data n = 411 and p = 87
## Model construction completed within 0.93 minutes.
## OOB results on all observations:
##   R^2: 0.77309
##   RMSE: 84254.63
##   MAE: 58324.4
##   L2: 2.917625e+12
##   L1: 23971329
```

```r
oob_se = sd(housing_data_train$sale_price - rf_mod$predicted)
oob_se
```

```
## [1] 77279.66
```

```r
View(data.frame(cbind("R-Squared" = max(rf_mod$rsq), "OOB_SE" = oob_se)), "Random Forest
```

#Break open the test data.

Out-of-sample OLS model metrics

```r
y_test = as.matrix(y_test)

y_hat_oos = predict(mod_ols, X_test)
oos_residuals = y_test - y_hat_oos

R_sq_oos = 1 - sum(oos_residuals^2) / sum((y_test - mean(y_test))^2)
RMSE_oos = sqrt(mean(oos_residuals^2))
ooss_e = sd(y_hat_oos - y_test)

RMSE_oos
```

```
## [1] 69535.17
```

```
R_sq_oos
```

```
## [1] 0.8625976
```

```
ooss_e
```

```
## [1] 69821.17
```

Create a final OLS model and compute final in-sample statistics for whole data set.

```
train = cbind(X_train, "sale_price" = y_train)
test = cbind(X_test, y_test)
full = rbind(train, test)

head(train)
```

```
##   approx_year_built cats_allowed community_district_num coop_condo date_of_sale
## 1              1955           no                     25      co-op        16847
## 2              1955           no                     25      co-op        16847
## 3              2004           no                     24      condo        16848
## 4              2002           no                     25      condo        16848
## 5              1949          yes                     26      co-op        16849
## 6              1950           no                     29      co-op        16850
##   dining_room_type dogs_allowed fuel_type garage_exists kitchen_type
## 1            combo           no       gas         FALSE       eat in
## 2           formal           no       oil         FALSE       eat in
## 3            combo           no   unknown         FALSE   efficiency
## 4            combo           no       gas         FALSE       eat in
## 5            combo          yes       gas         FALSE       eat in
## 6            combo           no       gas         FALSE   efficiency
##   num_bedrooms num_floors_in_building num_full_bathrooms num_total_rooms
## 1            2               6.000000                  1               5
## 2            1               7.000000                  1               4
## 3            1               1.000000                  1               3
## 4            3               6.306667                  2               5
## 5            2               2.000000                  1               4
## 6            1               4.490000                  1               3
##   pct_tax_deductibl sq_footage total_taxes walk_score num_bathrooms
## 1            44.290   993.1100     2058.53         82             1
## 2            44.000   890.0000     2663.36         89             1
## 3            42.550   550.0000     5500.00         90             1
## 4            42.120   966.9858     2260.00         94             2
## 5            39.000   675.0000     2641.52         71             1
## 6            41.015   711.8900     2299.87         72             1
##   month_of_year day_of_week day_of_month year zip_factor total_taxes_missing
## 1             2           3           16 2016      11355                TRUE
## 2             2           3           16 2016      11354                TRUE
```

```
## 3            2          4          17 2016      11368               FALSE
## 4            2          4          17 2016      11354               FALSE
## 5            2          5          18 2016      11426                TRUE
## 6            2          6          19 2016      11423                TRUE
##   total_additional_charges log_tot_add_charges num_missing bedroom_sq_ft_ratio
## 1                      767            6.642487          13         0.002013876
## 2                      604            6.403574          12         0.001123596
## 3                      167            5.117994          11         0.001818182
## 4                      275            5.616771          13         0.003102424
## 5                      660            6.492240          11         0.002962963
## 6                      660            6.492240          14         0.001404711
##   bedroom_bathroom_ratio sale_price
## 1                    2.0     228000
## 2                    1.0     235500
## 3                    1.0     137550
## 4                    1.5     545000
## 5                    2.0     241700
## 6                    1.0     145000
```

```
head(test)
```

```
##   approx_year_built cats_allowed community_district_num coop_condo date_of_sale
## 1              1926           no                     25      condo        17123
## 2              1982          yes                     25      condo        17100
## 3              1947          yes                     26      co-op        17058
## 4              1956           no                     28      co-op        17156
## 5              1950          yes                     26      co-op        17106
## 6              1950           no                     24      co-op        17037
##   dining_room_type dogs_allowed fuel_type garage_exists kitchen_type
## 1          unknown           no       oil         FALSE       eat in
## 2            combo           no       gas         FALSE       eat in
## 3            combo          yes       gas         FALSE   efficiency
## 4            combo           no       gas          TRUE       eat in
## 5            combo           no       oil         FALSE       eat in
## 6           formal           no       gas          TRUE       eat in
##   num_bedrooms num_floors_in_building num_full_bathrooms num_total_rooms
## 1            3                      6                  2               6
## 2            2                     22                  3               7
## 3            1                      2                  1               3
## 4            1                      6                  1               3
## 5            2                      2                  1               4
## 6            2                      6                  1               4
##   pct_tax_deductibl sq_footage total_taxes walk_score num_bathrooms
## 1          38.96668  2000.0000    5359.000         96             2
## 2          41.70799  1419.0000    5807.000         82             3
```

```
## 3           43.31925   730.4336    2273.023         74              1
## 4           20.00000   921.6717    2585.406         91              1
## 5           43.11997   903.7003    2685.371         77              1
## 6           43.53132  1100.0000    2557.847         87              1
##   month_of_year day_of_week day_of_month year zip_factor total_taxes_missing
## 1            11           6           18 2016      11355               FALSE
## 2            10           4           26 2016      11360               FALSE
## 3             9           4           14 2016      11004                TRUE
## 4            12           4           21 2016      11375                TRUE
## 5            11           3            1 2016      11362                TRUE
## 6             8           4           24 2016      11355                TRUE
##   total_additional_charges log_tot_add_charges num_missing bedroom_sq_ft_ratio
## 1                      821            6.710523          11         0.001500000
## 2                     1017            6.924612          11         0.001409443
## 3                      497            6.208590          13         0.001369050
## 4                      740            6.606650          11         0.001084985
## 5                      810            6.697034          13         0.002213123
## 6                      886            6.786717          11         0.001818182
##   bedroom_bathroom_ratio sale_price
## 1              1.5000000     830000
## 2              0.6666667     790000
## 3              1.0000000     189000
## 4              1.0000000     205000
## 5              2.0000000     248500
## 6              2.0000000     355000
```

```
X = full[ , 1:(ncol(full) - 1)]
y = full[ , ncol(full)]

ols_mod_final = lm(y ~ ., X)
summary(ols_mod_final)
```

```
##
## Call:
## lm(formula = y ~ ., data = X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -231664  -34006     -26   28740  257163
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -2.008e+09  6.031e+09  -0.333 0.739305
## approx_year_built        4.561e+02  2.830e+02   1.612 0.107656
## cats_allowedyes          1.320e+04  9.561e+03   1.380 0.168178
```

```
## community_district_num    3.243e+03  1.157e+03   2.803 0.005277 **
## coop_condocondo           2.254e+05  2.941e+04   7.663 1.13e-13 ***
## date_of_sale             -2.713e+03  8.344e+03  -0.325 0.745233
## dining_room_typeother     1.260e+04  1.089e+04   1.158 0.247533
## dining_room_typeformal    2.216e+04  8.157e+03   2.717 0.006836 **
## dining_room_typeunknown   1.783e+03  7.878e+03   0.226 0.821088
## dogs_allowedyes          -2.475e+03  1.053e+04  -0.235 0.814328
## fuel_typegas              3.651e+04  2.185e+04   1.671 0.095494 .
## fuel_typenone             7.711e+04  4.680e+04   1.648 0.100099
## fuel_typeoil              4.559e+04  2.238e+04   2.037 0.042216 *
## fuel_typeother            2.750e+04  3.166e+04   0.868 0.385615
## fuel_typeunknown          4.645e+04  2.555e+04   1.818 0.069714 .
## garage_existsTRUE         3.438e+03  8.892e+03   0.387 0.699228
## kitchen_typecombo         1.008e+04  2.731e+04   0.369 0.712176
## kitchen_typeeat in       -8.320e+03  2.664e+04  -0.312 0.754943
## kitchen_typeefficiency   -1.584e+04  2.665e+04  -0.594 0.552569
## num_bedrooms              1.485e+05  2.687e+04   5.526 5.57e-08 ***
## num_floors_in_building    3.122e+03  7.352e+02   4.247 2.64e-05 ***
## num_full_bathrooms        4.243e+04  2.787e+04   1.522 0.128620
## num_total_rooms           5.348e+03  5.108e+03   1.047 0.295702
## pct_tax_deductibl        -7.033e+02  9.524e+02  -0.738 0.460603
## sq_footage               -4.118e+01  1.499e+01  -2.747 0.006264 **
## total_taxes              -6.408e-01  3.889e+00  -0.165 0.869190
## walk_score               -5.379e+02  3.726e+02  -1.444 0.149494
## num_bathrooms             1.761e+04  4.117e+04   0.428 0.668969
## month_of_year             8.602e+04  2.548e+05   0.338 0.735843
## day_of_week              -6.184e+02  2.179e+03  -0.284 0.776703
## day_of_month              2.547e+03  8.370e+03   0.304 0.761039
## year                      1.018e+06  3.061e+06   0.333 0.739536
## zip_factor11005           4.645e+04  4.294e+04   1.082 0.279960
## zip_factor11101           1.481e+05  4.084e+04   3.626 0.000321 ***
## zip_factor11102           1.057e+05  3.718e+04   2.843 0.004666 **
## zip_factor11104           2.968e+04  4.586e+04   0.647 0.517842
## zip_factor11105           8.436e+04  5.139e+04   1.641 0.101403
## zip_factor11106           8.865e+04  3.965e+04   2.236 0.025864 *
## zip_factor11354           1.322e+04  2.601e+04   0.508 0.611436
## zip_factor11355          -1.708e+04  2.669e+04  -0.640 0.522465
## zip_factor11356          -1.607e+05  4.336e+04  -3.706 0.000237 ***
## zip_factor11357          -4.121e+04  2.597e+04  -1.587 0.113323
## zip_factor11358          -2.341e+03  4.261e+04  -0.055 0.956209
## zip_factor11360          -2.826e+04  2.518e+04  -1.122 0.262364
## zip_factor11361           3.818e+02  2.923e+04   0.013 0.989584
## zip_factor11362          -4.514e+04  2.495e+04  -1.809 0.071070 .
## zip_factor11363          -7.976e+03  3.254e+04  -0.245 0.806460
## zip_factor11364          -4.016e+04  2.479e+04  -1.620 0.105917
```

```
## zip_factor11365          -6.191e+04  3.102e+04  -1.996 0.046528 *
## zip_factor11367          -3.881e+04  2.444e+04  -1.588 0.113019
## zip_factor11368          -1.262e+05  2.930e+04  -4.307 2.03e-05 ***
## zip_factor11369          -6.403e+04  3.944e+04  -1.624 0.105174
## zip_factor11370          -1.095e+04  4.323e+04  -0.253 0.800135
## zip_factor11372           6.277e+04  2.551e+04   2.461 0.014243 *
## zip_factor11373          -2.509e+04  3.115e+04  -0.805 0.420987
## zip_factor11374          -7.162e+03  2.692e+04  -0.266 0.790335
## zip_factor11375           3.574e+04  2.444e+04   1.463 0.144272
## zip_factor11377           2.718e+04  3.077e+04   0.883 0.377576
## zip_factor11378          -1.012e+04  6.732e+04  -0.150 0.880545
## zip_factor11379          -7.473e+04  3.707e+04  -2.016 0.044392 *
## zip_factor11385          -6.615e+04  4.014e+04  -1.648 0.100003
## zip_factor11413          -7.259e+04  6.775e+04  -1.071 0.284565
## zip_factor11414          -1.526e+05  2.426e+04  -6.288 7.62e-10 ***
## zip_factor11415          -6.320e+04  2.570e+04  -2.460 0.014286 *
## zip_factor11417          -2.093e+05  5.170e+04  -4.048 6.08e-05 ***
## zip_factor11421          -9.484e+04  3.548e+04  -2.673 0.007791 **
## zip_factor11422          -7.737e+04  4.159e+04  -1.860 0.063514 .
## zip_factor11423          -9.562e+04  3.063e+04  -3.122 0.001913 **
## zip_factor11426          -9.692e+03  4.201e+04  -0.231 0.817661
## zip_factor11427          -7.236e+04  3.078e+04  -2.351 0.019143 *
## zip_factor11432          -9.672e+04  2.890e+04  -3.347 0.000885 ***
## zip_factor11433          -4.365e+05  7.009e+04  -6.228 1.09e-09 ***
## zip_factor11435          -7.904e+04  2.811e+04  -2.812 0.005146 **
## total_taxes_missingTRUE  -6.655e+03  2.692e+04  -0.247 0.804857
## total_additional_charges  8.184e+01  1.557e+01   5.255 2.29e-07 ***
## log_tot_add_charges      -1.101e+04  3.609e+03  -3.050 0.002426 **
## num_missing              -1.874e+03  3.360e+03  -0.558 0.577404
## bedroom_sq_ft_ratio      -1.078e+08  1.868e+07  -5.773 1.46e-08 ***
## bedroom_bathroom_ratio    2.463e+04  2.890e+04   0.852 0.394495
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 63310 on 449 degrees of freedom
## Multiple R-squared:  0.894,  Adjusted R-squared:  0.8756
## F-statistic: 48.57 on 78 and 449 DF,  p-value: < 2.2e-16
```

```
summary(ols_mod_final)$r.sq
```

```
## [1] 0.8940442
```

```
R_sq_final = summary(ols_mod_final)$r.sq
RMSE_final = summary(ols_mod_final)$sigma
```

```
RMSE_Rsq_table = data.frame(cbind("RMSE" = c(RMSE, RMSE_oos, RMSE_final), "R Squared" =

View(RMSE_Rsq_table)
```