Bestimates

Final project for Math 390 Data Science at Queens College

May 24, 2019

By Adriana Sham

In collaboration with:

Burhan Haanif

Diego Astudillo

Sakib Salim

Vincent Miceli

Abstract

We use sophisticated algorithms to predict the value of individual coops and condos. We contend that these estimates are far more accurate and efficient than zillow's predictions and Multiple Listing Services (mlsi).

We identify the most significant features of apartment dynamically, supplement the data set to smooth over missing data, and run our data in several algorithms that feature select, validate, and ship a model useful toward the goal of estimating future prices.

1. Introduction

Machine learning is based on the study of algorithms and statistical models, using computer systems to create algorithms and statistical models to gradually improve their performance in specific tasks. The process of applying machine learning to real world problems is called predictive analysis.

In this project, a predicting model is used estimate the selling price of aparments in Queens, NY area; this project pointedly studies the condominium and coops. It utilizes the raw data listed on Multiple Listing Services, for coops and condominiums that are sold during the year of 2016 to 2017 which consists of n = 2230 (observations) and p = 55 (features).

In the model training process, it uses $D = x_n, y_n$ to train the predictive model, to get the $yhats$ (the output). Three different algorithms are used to make a model for the sale price of the apartments: Linear, Regression Tree, and Random Forests Modeling.

Our group used Google's vitual machine to obtain better estimates, and bu doing so it allowed us to make more iterations in the algorithms.

2. The Data

The raw data that was utilized in tho make the model was harvested from Multiple Listing Services. The data set was harvested with MTurk and it is a raw download from their website. It is mainly zip codes from mainland Queens, leaving out the Rockaways, a peninsula near JFK airport that is geographically distinct from the rest of the neighborhoods, with a total of 55 zipcodes from Queens, NY. The raw size of the dataset is n x p which is 2230 x 55. However, some of the irrelevant features were excluded in order to improve the dataset.

The raw dataset began with 55 features and it was cut down to 25 features, because 30 of them were irrelevant. The features that were kept in the original data are:

The population of interest is the entire the coops and condos that are sold in Queens. This dataset partly represents the population of interest because this is a fairly small sample size of Queens. Certain zipcodes have just a few observations in comparison to the other zipcodes that have a larger size of obsevations.

External sources were not use in this project, but we featurization was done using the provided dataset. There are dangers of extrapolation because the prediction of this data can only predict on the zipcodes bedrrom, bathrroms, within the range provided and not outside of that and that might make extrapolation anything outside of the range of these feature. Now that out model is predicting for studios, one, two, three bedrooms home, using the same model to predict home that have over 3 bedrooms would be extrapolation. In our dataset there are outliers, for example the house that worth close to $1M or in the total taxes that ranges from $11 to $9300. It does not take account on weird cases, it will predict poorly on them.

2.2. Featurization

There were 55 features at the beginnng and only 24 features were kept, and they are:

approx_year_built: interger; prewar built would be concrete walls, and brick outside, old elevators and modern ones that are built would be wood community_district_num: interger; determines school districts *coop_condo: factor; appartments that are either coops or condos. coops are those that have more community charges, you don't own the appartment, however, you own a stock in the cooporation that owns the building. In order to buy coops you have to be aproved by the coorperation board. Coops also have less fredom in renting or subletting. Co-ops are also cheaper then condos. In Comparison to co_ops a person actully owns the apartment , and has the freedom to make the changes in the floor plans and almost 2x as expensive* dining_room_type: factor; this was factorial and was combo, formal or other *garage_exists: factor; existence of garage in the coop/condo* kitchen_type: factor; the feature was factorial but we made it into two different kinds "eat in" and "efficeny" or none *num_bedrooms: interger; number of bedrroms in the coop/condo* num_floors_in_building: interger; number of floors in the coop/condo *num_full_bathrooms: interger; number of full bathrooms in the coop/condo* num_half_bathrooms: interger; number of half bathrooms in the coop/condo *num_total_rooms: interger; number of total rooms in the coop/condo*

parking_charges: factor; the charge for the parking space in the coop/condo *pct_tax_deductibl: interger; the percentage of reducable taxable income* sq_footage: interger; the size of the apartment by square feet *total_taxes: factor; the property tax charged by the local goverment* walk_score: interger; it is the walk to the nearby stores

Within the 25 features, we created more features from raw data that help facilitate the model making process, such as:

pets_allowed: cats_allowed and dog_allowed were combined into one column because they are collinear. montly charges: maintenance_cost and common_charges are combine into one column since they are mutually exclusive. *price_persqft: using listing_price to divide by sqr_footage, then dropping listing_price and sqr_footage* lat: latitude of addresses from full_address_or_zip_code. We used the package ggmap to get the coordinates from the given feature full_address_or_zip_code, then we dropped the feature after using. *lon: longitude of addresses from full_address_or_zip_code. We used the package ggmap to get the coordinates from the given feature full_address_or_zip_code, then we dropped the feature after using.* Distance to the closest LIRR: we used ggmap to get the latitude and longitude of each LIRR station in Queens, then we ran a code to find the nearest distance of each coop/condo to the closest LIRR station.

we featurize some of the data in accordance to how we needed it in excel and in R such as deleting the dollar sign($) from some columns because Rwa sreading the column as a string, then we made as numeric, another exaple would be kitchen_type we made into eaten and efficiency, etc.

2.3. Errors and Missingness

The dataset was full of NAs, misspelling and wrong entries. One of them was in the parking garage exists such as “eys”, “yes”, “underground”, “Yes”, we corrected them into “yes” through excel by replacing each misspelled entry and turning NAs into “no” assuming they have no garage. There were errors in the total_taxes entries some of the entries for the taxes were uncertain such as entries under a $1000 dollars. There were community district numbers that were missing and we used search engine to correct each entry.

In total_taxes, the entries that were under a $1000 dollars we decided to make it into NA’s and let missforest fill the entries because total_taxes lower than $1000 did not make sense. Entries that have over 1000 entries of Nas, we decided not to impute them because it would give us an arbitrary number which might worsen our prediction. And most of the columns that contains NAs we used missForest to impute the data, so that we don’t drop some of the important features.

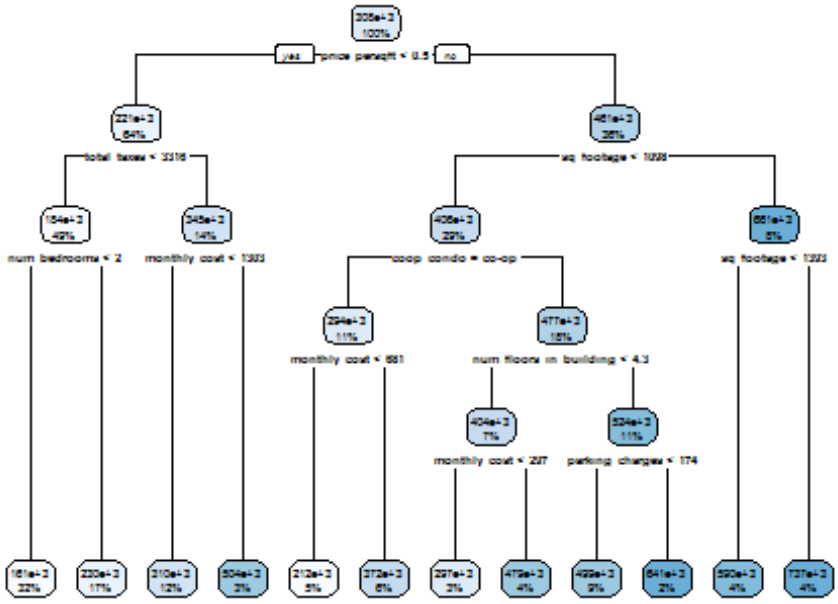
We also included a dummy matrix M, to take account of the NAs so that when running the regression the missingness would also be taken account, and see the effect on the regression line.

3. Modeling

Below are three different model to predict on prices. It shows which of the model gives the best result to ship to ship to the world. The dataset used n = 523 observations and the rest of the observation were dropped because they did not have the y (the output).

3.1 Regression Tree Modeling

The were only six variable used for spliting, they are coop/condo, sq_footage, price_per_sq_foot, parking_charges, latitude, montly_cost. The tree starts from the most important feature to start splitting which in this case is coop/condo because the feature that affect the price the most is coop/condo, then going down to the size of the apartment determine by sq_footage, then by price_per_sq_foot, parking_charges, latitude, montly_cost which is the montly maintenance charges by the apprtment.



3.2 Linear Modeling

We fitted a linear regression model using xs (features) and some of them were statistically signficant, which are coop_condocondo, dining_room_typeformal, num_bedrooms, num_floors_in_building, num_total_rooms, parking_charges, lat, monthly_cost, price_persqft, is_missing_community_district_num

R^2 in sample = 84 is higher than out of sample, it is higher because it is prompt to overfitting. out of sample R^2 = 85 is lower than in sample since since it is working with the testing set. To interpret the coefficients once we set everything in constant, once we

compare two naturally observed observations PRICE and coop/condo we can see that price will go up by $172000 dollars. yet if we take in considerration latitude PRICE will increase to $677900

This linear regression did well, but it could be done better using RandomForest. This is because the data is not completly linear, and therefore it is not yielding a desirable R^2 and the RMSE is $83215.95 which means that our linear model is sample estimates is plus or minus $83215.95.

```
##
## Call:
## lm(formula = Ytrain ~ ., data = Xtrain)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -295653  -31899    2725   33423  314214
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     -3.794e+07  1.017e+07  -3.730 0.000220
## approx_year_built               -3.775e+01  2.690e+02  -0.140 0.888450
## community_district_num           2.001e+03  1.243e+03   1.610 0.108301
## coop_condocondo                  1.720e+05  1.977e+04   8.703  < 2e-16
## dining_room_typedining area      1.950e+04  5.172e+04   0.377 0.706350
## dining_room_typeformal           2.424e+04  9.331e+03   2.598 0.009733
## dining_room_typeother            1.246e+04  1.221e+04   1.020 0.308268
## garage_exists                    3.292e+03  1.023e+04   0.322 0.747841
## kitchen_typeeat in               5.448e+03  1.092e+04   0.499 0.618207
## kitchen_typeefficiency          -1.649e+04  1.078e+04  -1.530 0.126950
## num_bedrooms                     3.235e+04  8.779e+03   3.685 0.000261
## num_floors_in_building           2.363e+03  7.932e+02   2.980 0.003069
## num_full_bathrooms               1.706e+04  5.152e+04   0.331 0.740723
## num_half_bathrooms               2.510e+03  3.594e+04   0.070 0.944363
## num_total_rooms                  2.179e+04  5.748e+03   3.791 0.000174
## parking_charges                  3.950e+02  1.032e+02   3.827 0.000151
## pct_tax_deductibl                5.616e+01  1.417e+03   0.040 0.968404
## sq_footage                       2.578e+01  1.316e+01   1.958 0.050927
## total_taxes                      7.102e+00  5.832e+00   1.218 0.224030
## walk_score                      -5.797e+02  3.738e+02  -1.551 0.121804
## lat                              6.779e+05  1.472e+05   4.606 5.59e-06
## lon                             -1.359e+05  9.529e+04  -1.426 0.154661
## pets_allowed                     5.998e+03  7.596e+03   0.790 0.430213
## monthly_cost                     1.630e+02  1.876e+01   8.688  < 2e-16
## price_persqft                    4.754e+05  7.376e+04   6.445 3.44e-10
## is_missing_approx_year_built     1.324e+04  3.309e+04   0.400 0.689335
## is_missing_community_district_num -2.695e+05  7.382e+04  -3.651 0.000297
## is_missing_dining_room_type      5.126e+02  8.653e+03   0.059 0.952789
## is_missing_kitchen_type         -5.278e+03  2.832e+04  -0.186 0.852252
## is_missing_num_floors_in_building  6.807e+02  9.305e+03   0.073 0.941719
## is_missing_num_half_bathrooms    8.973e+02  1.691e+04   0.053 0.957718
```

```
## is_missing_parking_charges        -5.735e+03  8.255e+03  -0.695 0.487666
## is_missing_pct_tax_deductibl       1.016e+04  9.496e+03   1.070 0.285327
## is_missing_sq_footage              2.216e+03  7.544e+03   0.294 0.769094
## is_missing_total_taxes             6.810e+02  1.000e+04   0.068 0.945737
## is_missing_monthly_cost           -7.148e+03  2.288e+04  -0.312 0.754886
##
## (Intercept)                       ***
## approx_year_built
## community_district_num
## coop_condocondo                   ***
## dining_room_typedining area
## dining_room_typeformal            **
## dining_room_typeother
## garage_exists
## kitchen_typeeat in
## kitchen_typeefficiency
## num_bedrooms                      ***
## num_floors_in_building            **
## num_full_bathrooms
## num_half_bathrooms
## num_total_rooms                   ***
## parking_charges                   ***
## pct_tax_deductibl
## sq_footage                        .
## total_taxes
## walk_score
## lat                               ***
## lon
## pets_allowed
## monthly_cost                      ***
## price_persqft                     ***
## is_missing_approx_year_built
## is_missing_community_district_num ***
## is_missing_dining_room_type
## is_missing_kitchen_type
## is_missing_num_floors_in_building
## is_missing_num_half_bathrooms
## is_missing_parking_charges
## is_missing_pct_tax_deductibl
## is_missing_sq_footage
## is_missing_total_taxes
## is_missing_monthly_cost
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 70340 on 386 degrees of freedom
## Multiple R-squared:  0.855,  Adjusted R-squared:  0.8418
## F-statistic: 65.03 on 35 and 386 DF,  p-value: < 2.2e-16
```

3.3 Random Forest Modeling

Random forest is like a bunch of decision trees. When building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. A fresh sample of m predictors is taken at each split, and typically we choose the number of predictors considered at each split is approximately equal to the square root of the total number of predictors.

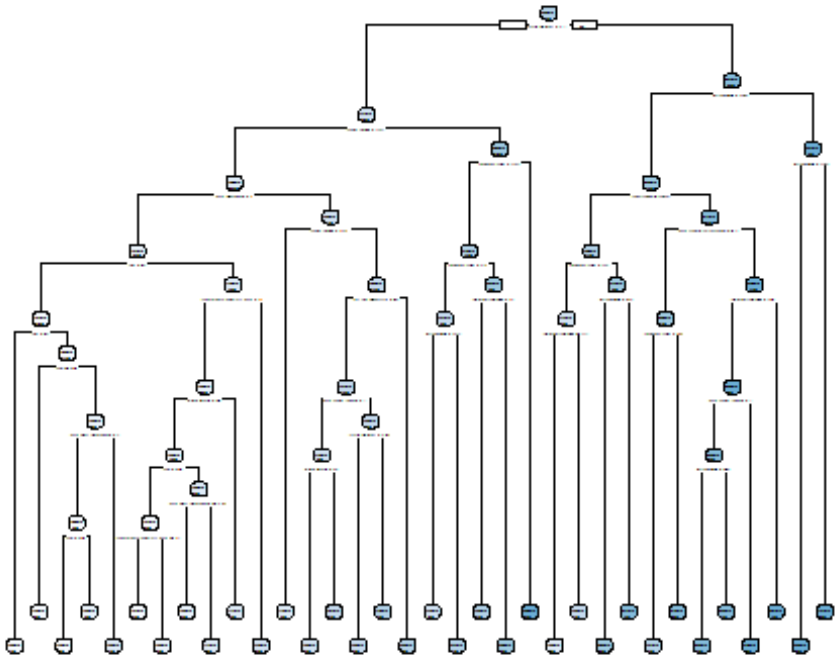
Random Forest is a non-parametric model. Where the function that is used to predict can be any function; the data decides on how the function looks like.

We believe random forest is the best because When the number of trees produced by random forests approaches infinity, theoretically, according to the large number theorem, it can be proved that the training error and the test error converge. In the actual process, since it is impossible to generate an infinite number of decision trees, the setting problem of the model parameters will affect the degree of over-fitting of the fitting results in the same running time.

I believe that coop_condo, sq_footage, and latitude have an effect on sale price that is truly causal. Whether the apartment is coop or condo is important because if it's a condominium then it means that it would predict higher sale price, and if it's a coop it would have a lower price. sq_footage is the size of the house the bigger the house the more expensive its is sale price and latitude because the more north is the apartment the more expensive it is.

4. Performance Results (for Random Forest)

in sample R^2 = 84 and out of sample R^2 = 85 which is very close to out linear model. The RMSE is plus or minu $70000. which is way better than the RMSE in the linear model. It is a $13000 of difference. This is a valid estimate of model because the data was split into train and test set, and performing validation from the model created from the train set and tested it on the test set.



5. Discussion

While trying to use missForest to fill the NAs in the dataset. At first the y's in the training set was use to used the y's, giving was giving perfect x's the moment out of sample was done, were having bad results. Due to short amount of time given, this prediction is the best that we can have for now, if given time was longer our group would have done the forward stepwise to do feature selecting and have a better prediction model. I think our prediction is better than the listed estimates that Zillow has (with the given zipcodes) because the model that we carefully futurized. If we were given more time we would have also included distance from the apartment to midtown Manhattan.

##Code Appendix

```
pacman::p_load(dplyr, tidyr, ggplot2, magrittr, stringr, mlr)
housing_data = read.csv("housing_data_2016_2017.csv")
```

##Delete variables that we dont need

```
housing_data %<>%
  select(-c(HITId, HITTypeId, Title, Description, Keywords, Reward, CreationTime, MaxAssignments, RequesterAnnotation, AssignmentDurationInSeconds, AutoApprovalDelayInSeconds, Expiration, NumberOfSimilarHITs, LifetimeInSeconds, AssignmentId, WorkerId, AssignmentStatus, AcceptTime, SubmitTime, AutoApprovalTime, ApprovalTime, RejectionTime, RequesterFeedback, WorkTimeInSeconds, LifetimeApprovalRate, Last30DaysApprovalRate, Last7DaysApprovalRate, URL, url, date_of_sale))
```

Clean Data

```
housing_data %<>%
  mutate( zip_code = str_extract(full_address_or_zip_code, "[0-9]{5}"))

housing_data %<>%
  mutate(dogs_allowed = ifelse(substr(housing_data$dogs_allowed, 1, 3) == "ye
s", 1, 0)) %>%
  mutate(cats_allowed = ifelse(substr(housing_data$cats_allowed, 1, 3) == "ye
s", 1, 0)) %>%
  mutate( pets_allowed = ifelse( cats_allowed + dogs_allowed > 0, 1, 0)) %>%
  mutate(coop_condo = factor(tolower(coop_condo)))

housing_data %<>%
  select(-c(dogs_allowed,cats_allowed, fuel_type))

d = housing_data

d %<>%
  mutate(maintenance_cost = sjmisc::rec(maintenance_cost, rec = "NA = 0 ; els
e = copy")) %<>%
  mutate(common_charges = sjmisc::rec(common_charges, rec = "NA = 0 ; else =
copy"))##recode from NA to 0.


# combine maintaince cost and common charges
d %<>%
  mutate( monthly_cost = common_charges + maintenance_cost)

d %<>%
  mutate(monthly_cost = sjmisc::rec(monthly_cost, rec = "0 = NA ; else = copy
"))

## Garage exists conver it to binary

d %<>%
  mutate(garage_exists = sjmisc::rec(garage_exists, rec = "NA = 0 ; else = co
py")) ##recode from NA to 0.

d %<>%
  mutate(garage_exists = sjmisc::rec(garage_exists, rec = " eys = 1; UG = 1 ;
Underground = 1; yes = 1 ; Yes = 1 ; else = copy")) ##recode from NA to 0.

d %<>%
  select(-c(maintenance_cost , common_charges, model_type))
```

##Change variable type

```
d %<>%
  mutate( dining_room_type = as.factor(dining_room_type)) %>%
  mutate(garage_exists = as.character(garage_exists)) %>%
  mutate(garage_exists = as.numeric(garage_exists)) %>%
  mutate( parking_charges = as.character(parking_charges)) %>%
  mutate( parking_charges = as.numeric(parking_charges)) %>%
  mutate(sale_price = as.character(sale_price)) %>%
  mutate(sale_price = as.numeric(sale_price)) %>%
  mutate(total_taxes = as.character(total_taxes)) %>%
  mutate(total_taxes = as.numeric(total_taxes)) %>%
  mutate(price_persqft = listing_price_to_nearest_1000 / sq_footage)

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

#Added latitude and longitude features using ggmap

```
#Already run and included in the data
#pacman::p_load(ggmap)
#d %<>%
#  mutate(lat = geocode(full_address_or_zip_code)$lat, lon = #geocode(full_ad
dress_or_zip_code)$lon )
#geocoordinates for relevant LIRR stations

lirr_coord = coord

## Error in eval(expr, envir, enclos): object 'coord' not found

RAD_EARTH = 3958.8
degrees_to_radians = function(angle_degrees){
  for(i in 1:length(angle_degrees))
    angle_degrees[i] = angle_degrees[i]*pi/180
  return(angle_degrees)
}
compute_globe_distance = function(destination, origin){
  destination_rad = degrees_to_radians(destination)
  origin_rad = degrees_to_radians(origin)
  delta_lat = destination_rad[1] - origin_rad[1]
  delta_lon = destination_rad[2] - origin_rad[2]
  h = (sin(delta_lat/2))^2 + cos(origin_rad[1]) * cos(destination_rad[1]) * (
sin(delta_lon/2))^2
  central_angle = 2 * asin(sqrt(h))
  return(RAD_EARTH * central_angle)
}
#find the closest LIRR station and compute distance
shortest_lirr_distance = function(all_lirr_coords, house_coords){
  shortest_dist = Inf
  for (i in 1: nrow(all_lirr_coords)){
```

```
    ith_lirr = c(all_lirr_coords$lat[i], all_lirr_coords$lon[i])
    new_dist = compute_globe_distance(ith_lirr, house_coords)
    if( new_dist < shortest_dist){
      shortest_dist = new_dist
    }
  }
  return(shortest_dist)
}
d %<>%
  rowwise() %>%
  mutate(shortest_dist = shortest_lirr_distance(lirr_coord, c(lat, lon)) )

## Error in nrow(all_lirr_coords): object 'lirr_coord' not found

#makes any other addresses redundant
d %<>%
  select(-c(zip_code, full_address_or_zip_code, listing_price_to_nearest_1000
))
```

We are trying to predict `sale_price`. So let's section our dataset:

```
####CREATE A COLUMN ID

d %<>%
  ungroup(d) %>%
  mutate(id = 1 : 2230)
d %<>%
  mutate(total_taxes = ifelse(d$total_taxes < 1000, NA, total_taxes))
real_y = data.frame(d$id, d$sale_price)
real_d = subset(d, (!is.na(d$sale_price)))
fake_d = subset(d, (is.na(d$sale_price)))
real_d$sale_price = NULL
fake_d$sale_price = NULL
```

#Split the data that has y into train and test sets

```
train_indices = sample(1 : nrow(real_d), nrow(real_d)*4/5)
training_data = real_d[train_indices, ]
testing_data = real_d[-train_indices, ]

X = rbind(training_data, testing_data, fake_d)

M = tbl_df(apply(is.na(X), 2, as.numeric))
colnames(M) = paste("is_missing_", colnames(X), sep = "")
```

#Some of these missing indicators are collinear because they share all the rows they are missing on. Let's filter those out:

```
M = tbl_df(t(unique(t(M))))
```

#Some featuers did not have missingness so let's remove them:

```r
M %<>% select_if(function(x){sum(x) > 0})

pacman::p_load(missForest)
Ximp = missForest(data.frame(X), sampsize = rep(172, ncol(X)))$ximp

##   missForest iteration 1 in progress...

## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you wan
t
## to do regression?

## done!
##   missForest iteration 2 in progress...

## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you wan
t
## to do regression?

## done!
##   missForest iteration 3 in progress...

## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you wan
t
## to do regression?

## done!
##   missForest iteration 4 in progress...

## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you wan
t
## to do regression?

## done!

Ximp %<>%
  arrange(id)

Xnew = data.frame(cbind(Ximp, M, real_y))

Xnew %<>%
  mutate(price = d.sale_price) %>%
  select(-c(id, d.id, d.sale_price))

linear_mod_impute_and_missing_dummies = lm(price ~ ., data = Xnew)
summary(linear_mod_impute_and_missing_dummies)

##
## Call:
```

```
## lm(formula = price ~ ., data = Xnew)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -296854  -36454     -20   36523  332799
##
## Coefficients: (3 not defined because of singularities)
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -3.664e+07  9.154e+06  -4.003 7.23e-05
## approx_year_built               3.249e+01  2.453e+02   0.132 0.894659
## community_district_num          2.337e+03  1.169e+03   2.000 0.046099
## coop_condocondo                 1.419e+05  1.729e+04   8.208 1.98e-15
## dining_room_typedining area     1.559e+04  5.259e+04   0.297 0.766940
## dining_room_typeformal          2.332e+04  8.519e+03   2.738 0.006405
## dining_room_typeother           1.297e+04  1.126e+04   1.152 0.249739
## garage_exists                   5.128e+02  9.017e+03   0.057 0.954675
## kitchen_typeeat in             -2.854e+03  1.000e+04  -0.285 0.775535
## kitchen_typeefficiency         -1.767e+04  9.750e+03  -1.813 0.070508
## num_bedrooms                    4.236e+04  7.904e+03   5.359 1.29e-07
## num_floors_in_building          2.535e+03  7.163e+02   3.539 0.000440
## num_full_bathrooms              1.727e+04  5.242e+04   0.329 0.741999
## num_half_bathrooms             -1.230e+04  3.258e+04  -0.377 0.706010
## num_total_rooms                 1.853e+04  5.175e+03   3.581 0.000376
## parking_charges                 3.493e+02  9.717e+01   3.595 0.000357
## pct_tax_deductibl               4.671e+01  1.026e+03   0.046 0.963700
## sq_footage                      3.027e+01  1.268e+01   2.386 0.017398
## total_taxes                     1.314e+01  5.344e+00   2.458 0.014296
## walk_score                     -6.554e+02  3.382e+02  -1.938 0.053253
## lat                             6.530e+05  1.354e+05   4.822 1.90e-06
## lon                            -1.301e+05  8.466e+04  -1.537 0.125057
## pets_allowed                    1.112e+04  6.886e+03   1.615 0.106929
## monthly_cost                    1.286e+02  1.416e+01   9.080  < 2e-16
## price_persqft                   5.546e+05  6.661e+04   8.326 8.36e-16
## is_missing_approx_year_built    6.389e+03  3.368e+04   0.190 0.849625
## is_missing_community_district_num -2.312e+05  7.469e+04  -3.095 0.002080
## is_missing_dining_room_type     5.110e+03  7.818e+03   0.654 0.513701
## is_missing_kitchen_type        -1.343e+04  2.871e+04  -0.468 0.640234
## is_missing_num_bedrooms                NA         NA      NA       NA
## is_missing_num_floors_in_building  2.070e+03  8.315e+03   0.249 0.803508
## is_missing_num_half_bathrooms  -1.221e+04  1.407e+04  -0.867 0.386093
## is_missing_num_total_rooms             NA         NA      NA       NA
## is_missing_parking_charges     -2.146e+03  7.632e+03  -0.281 0.778657
## is_missing_pct_tax_deductibl    1.291e+04  8.669e+03   1.489 0.137226
## is_missing_sq_footage          -2.267e+03  6.700e+03  -0.338 0.735281
## is_missing_total_taxes          6.496e+02  9.155e+03   0.071 0.943463
## is_missing_monthly_cost        -2.973e+03  1.991e+04  -0.149 0.881388
## is_missing_price_persqft               NA         NA      NA       NA
##
## (Intercept)                    ***
## approx_year_built
```

```
## community_district_num             *
## coop_condocondo                    ***
## dining_room_typedining area
## dining_room_typeformal             **
## dining_room_typeother
## garage_exists
## kitchen_typeeat in
## kitchen_typeefficiency             .
## num_bedrooms                       ***
## num_floors_in_building             ***
## num_full_bathrooms
## num_half_bathrooms
## num_total_rooms                    ***
## parking_charges                    ***
## pct_tax_deductibl
## sq_footage                         *
## total_taxes                        *
## walk_score                         .
## lat                                ***
## lon
## pets_allowed
## monthly_cost                       ***
## price_persqft                      ***
## is_missing_approx_year_built
## is_missing_community_district_num **
## is_missing_dining_room_type
## is_missing_kitchen_type
## is_missing_num_bedrooms
## is_missing_num_floors_in_building
## is_missing_num_half_bathrooms
## is_missing_num_total_rooms
## is_missing_parking_charges
## is_missing_pct_tax_deductibl
## is_missing_sq_footage
## is_missing_total_taxes
## is_missing_monthly_cost
## is_missing_price_persqft
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 72110 on 492 degrees of freedom
##   (1702 observations deleted due to missingness)
## Multiple R-squared:  0.8494, Adjusted R-squared:  0.8387
## F-statistic: 79.28 on 35 and 492 DF,  p-value: < 2.2e-16
```

REMOVING MISSING Y SECTION

```
Data = Xnew
### sale price is our imputed Y

Y = Data$price
```

```
Data %<>%
  filter(!is.na(price)) %>%
  select(-price)

Xtrain = Data[1:422, ]
Xtest = Data[423:528, ]

Ytrain = Y[1:422]
Ytest = Y[423:528]

dtrain = cbind(Xtrain, Ytrain) ## combine x train with y train, x test with y
test
dtest = cbind(Xtest, Ytest)
```

Dropping colinear features

```
Xtrain %<>%
  select(-c(is_missing_num_total_rooms, is_missing_num_bedrooms, is_missing_p
rice_persqft))
```

Linear Regression

```
linear = lm(Ytrain ~ ., data = Xtrain)## simple linear model
summary(linear)

##
## Call:
## lm(formula = Ytrain ~ ., data = Xtrain)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -295653  -31899    2725   33423  314214
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -3.794e+07  1.017e+07  -3.730 0.000220
## approx_year_built              -3.775e+01  2.690e+02  -0.140 0.888450
## community_district_num          2.001e+03  1.243e+03   1.610 0.108301
## coop_condocondo                 1.720e+05  1.977e+04   8.703  < 2e-16
## dining_room_typedining area     1.950e+04  5.172e+04   0.377 0.706350
## dining_room_typeformal          2.424e+04  9.331e+03   2.598 0.009733
## dining_room_typeother           1.246e+04  1.221e+04   1.020 0.308268
## garage_exists                   3.292e+03  1.023e+04   0.322 0.747841
## kitchen_typeeat in              5.448e+03  1.092e+04   0.499 0.618207
## kitchen_typeefficiency         -1.649e+04  1.078e+04  -1.530 0.126950
## num_bedrooms                    3.235e+04  8.779e+03   3.685 0.000261
## num_floors_in_building          2.363e+03  7.932e+02   2.980 0.003069
## num_full_bathrooms              1.706e+04  5.152e+04   0.331 0.740723
## num_half_bathrooms              2.510e+03  3.594e+04   0.070 0.944363
## num_total_rooms                 2.179e+04  5.748e+03   3.791 0.000174
```

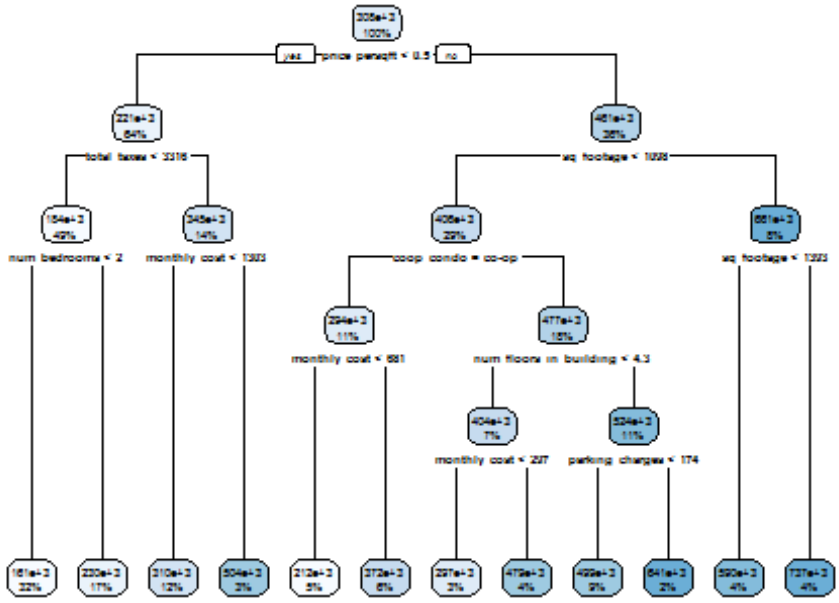
```
## parking_charges                     3.950e+02  1.032e+02   3.827 0.000151
## pct_tax_deductibl                   5.616e+01  1.417e+03   0.040 0.968404
## sq_footage                          2.578e+01  1.316e+01   1.958 0.050927
## total_taxes                         7.102e+00  5.832e+00   1.218 0.224030
## walk_score                         -5.797e+02  3.738e+02  -1.551 0.121804
## lat                                 6.779e+05  1.472e+05   4.606 5.59e-06
## lon                                -1.359e+05  9.529e+04  -1.426 0.154661
## pets_allowed                        5.998e+03  7.596e+03   0.790 0.430213
## monthly_cost                        1.630e+02  1.876e+01   8.688  < 2e-16
## price_persqft                       4.754e+05  7.376e+04   6.445 3.44e-10
## is_missing_approx_year_built        1.324e+04  3.309e+04   0.400 0.689335
## is_missing_community_district_num  -2.695e+05  7.382e+04  -3.651 0.000297
## is_missing_dining_room_type         5.126e+02  8.653e+03   0.059 0.952789
## is_missing_kitchen_type            -5.278e+03  2.832e+04  -0.186 0.852252
## is_missing_num_floors_in_building   6.807e+02  9.305e+03   0.073 0.941719
## is_missing_num_half_bathrooms       8.973e+02  1.691e+04   0.053 0.957718
## is_missing_parking_charges         -5.735e+03  8.255e+03  -0.695 0.487666
## is_missing_pct_tax_deductibl        1.016e+04  9.496e+03   1.070 0.285327
## is_missing_sq_footage               2.216e+03  7.544e+03   0.294 0.769094
## is_missing_total_taxes              6.810e+02  1.000e+04   0.068 0.945737
## is_missing_monthly_cost            -7.148e+03  2.288e+04  -0.312 0.754886
##
## (Intercept)                        ***
## approx_year_built
## community_district_num
## coop_condocondo                    ***
## dining_room_typedining area
## dining_room_typeformal             **
## dining_room_typeother
## garage_exists
## kitchen_typeeat in
## kitchen_typeefficiency
## num_bedrooms                       ***
## num_floors_in_building             **
## num_full_bathrooms
## num_half_bathrooms
## num_total_rooms                    ***
## parking_charges                    ***
## pct_tax_deductibl
## sq_footage                         .
## total_taxes
## walk_score
## lat                                ***
## lon
## pets_allowed
## monthly_cost                       ***
## price_persqft                      ***
## is_missing_approx_year_built
## is_missing_community_district_num ***
## is_missing_dining_room_type
```

```
## is_missing_kitchen_type
## is_missing_num_floors_in_building
## is_missing_num_half_bathrooms
## is_missing_parking_charges
## is_missing_pct_tax_deductibl
## is_missing_sq_footage
## is_missing_total_taxes
## is_missing_monthly_cost
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 70340 on 386 degrees of freedom
## Multiple R-squared:  0.855,  Adjusted R-squared:  0.8418
## F-statistic: 65.03 on 35 and 386 DF,  p-value: < 2.2e-16
```

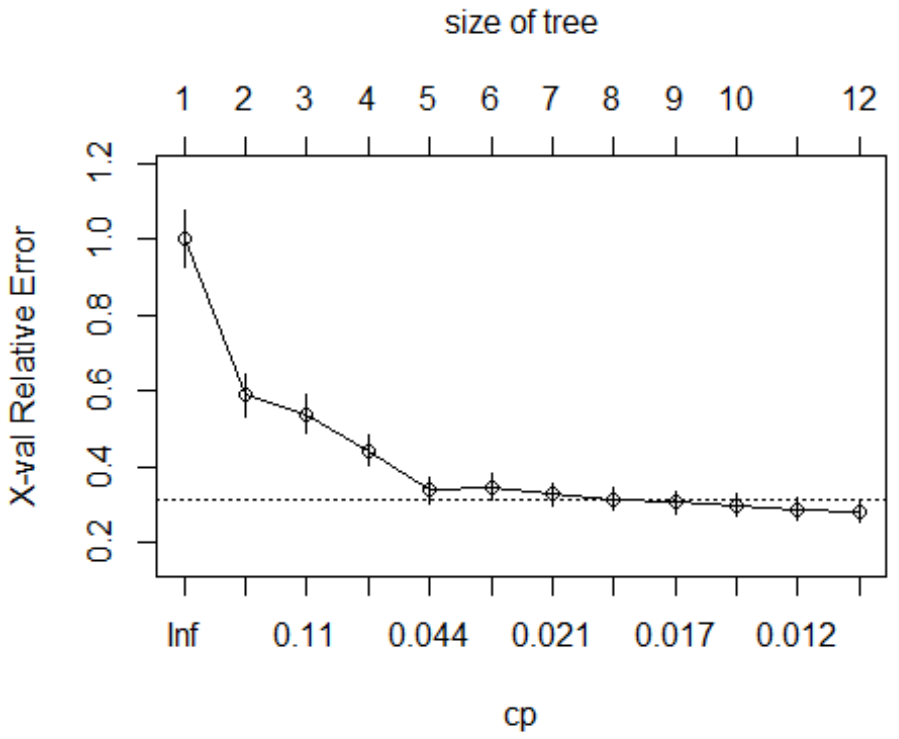
```
yhat = predict(linear, Xtest)

e = yhat - Ytest

sqrt(sum(e^2) / nrow(Xtest))
```

```
## [1] 83215.95
```

```
#REGRESSION TREE

pacman::p_load(rsample)#data spliting
pacman::p_load(rpart) #performing reg tree
pacman::p_load(rpart.plot) #ploting reg tree
pacman::p_load(ipred) #bagging
pacman::p_load(caret) #bagging



m1 = rpart(
  formula = Ytrain ~ .,
  data    = Xtrain,
  method  = "anova"
  )

rpart.plot(m1)
```




```
plotcp(m1)
```




```
summary(m1)
```

```
## Call:
## rpart(formula = Ytrain ~ ., data = Xtrain, method = "anova")
##   n= 422
##
##            CP nsplit rel error    xerror       xstd
## 1  0.42758149      0 1.0000000 1.0034864 0.07563004
## 2  0.12824221      1 0.5724185 0.5888809 0.05677089
## 3  0.09266116      2 0.4441763 0.5377683 0.05130348
## 4  0.07319492      3 0.3515151 0.4417798 0.04077420
## 5  0.02585520      4 0.2783202 0.3364894 0.03348648
## 6  0.02291037      5 0.2524650 0.3462884 0.03336475
## 7  0.01939333      6 0.2295546 0.3283003 0.02936593
## 8  0.01776872      7 0.2101613 0.3146267 0.03040719
## 9  0.01663924      8 0.1923926 0.3055096 0.02989080
## 10 0.01348814      9 0.1757533 0.2983826 0.03004550
## 11 0.01014447     10 0.1622652 0.2876490 0.02931209
## 12 0.01000000     11 0.1521207 0.2813368 0.02881401
##
## Variable importance
##          price_persqft           monthly_cost             coop_condo
##                     20                     15                     13
##      approx_year_built             sq_footage            total_taxes
##                     12                      9                      8
##        parking_charges                    lon           num_bedrooms
##                      8                      5                      3
##        num_total_rooms      pct_tax_deductibl community_district_num
##                      3                      2                      1
## num_floors_in_building                    lat
##                      1                      1
##
## Node number 1: 422 observations,    complexity param=0.4275815
##   mean=308191.7, MSE=3.121006e+10
##   left son=2 (268 obs) right son=3 (154 obs)
##   Primary splits:
##       price_persqft     < 0.5036155 to the left,  improve=0.4275815, (0 mi
ssing)
##       coop_condo        splits as  LR, improve=0.3754617, (0 missing)
##       approx_year_built < 1970.5    to the left,  improve=0.3463094, (0 mi
ssing)
##       total_taxes       < 3389.792  to the left,  improve=0.3270625, (0 mi
ssing)
##       num_total_rooms   < 4.5       to the left,  improve=0.2781774, (0 mi
ssing)
##   Surrogate splits:
##       coop_condo        splits as  LR, agree=0.848, adj=0.584, (0 split)
##       approx_year_built < 1970.5    to the left,  agree=0.844, adj=0.571,
(0 split)
##       parking_charges   < 135.1167  to the left,  agree=0.794, adj=0.435,
(0 split)
##       monthly_cost      < 471.5     to the right, agree=0.789, adj=0.422,
```

```
(0 split)
##       lon               < -73.87994 to the right, agree=0.744, adj=0.299,
(0 split)
##
## Node number 2: 268 observations,    complexity param=0.09266116
##   mean=220622.8, MSE=1.036057e+10
##   left son=4 (207 obs) right son=5 (61 obs)
##   Primary splits:
##       total_taxes     < 3316.375  to the left,  improve=0.4395279, (0 miss
ing)
##       sq_footage      < 940.6025  to the left,  improve=0.4138876, (0 miss
ing)
##       monthly_cost    < 1019      to the left,  improve=0.3937581, (0 miss
ing)
##       num_total_rooms < 4.5       to the left,  improve=0.3399684, (0 miss
ing)
##       num_bedrooms    < 1.5       to the left,  improve=0.3103121, (0 miss
ing)
##   Surrogate splits:
##       sq_footage             < 1050.357  to the left,  agree=0.858, adj=0.
377, (0 split)
##       monthly_cost           < 985.5     to the left,  agree=0.851, adj=0.
344, (0 split)
##       lat                    < 40.77906  to the left,  agree=0.832, adj=0.
262, (0 split)
##       num_total_rooms        < 5.5       to the left,  agree=0.810, adj=0.
164, (0 split)
##       num_floors_in_building < 7.36      to the left,  agree=0.806, adj=0.
148, (0 split)
##
## Node number 3: 154 observations,    complexity param=0.1282422
##   mean=460584.3, MSE=3.092526e+10
##   left son=6 (121 obs) right son=7 (33 obs)
##   Primary splits:
##       sq_footage        < 1098      to the left,  improve=0.3546534, (0 mi
ssing)
##       num_total_rooms   < 4.5       to the left,  improve=0.3330176, (0 mi
ssing)
##       num_bedrooms      < 1.5       to the left,  improve=0.3201549, (0 mi
ssing)
##       total_taxes       < 3690.024  to the left,  improve=0.3156214, (0 mi
ssing)
##       approx_year_built < 1963.5    to the left,  improve=0.2232692, (0 mi
ssing)
##   Surrogate splits:
##       total_taxes       < 4120.488  to the left,  agree=0.890, adj=0.485,
(0 split)
##       num_bedrooms      < 2.5       to the left,  agree=0.857, adj=0.333,
(0 split)
##       monthly_cost      < 1478.5    to the left,  agree=0.844, adj=0.273,
```

```
(0 split)
##       num_total_rooms   < 5.5       to the left,  agree=0.805, adj=0.091,
(0 split)
##       pct_tax_deductibl < 35.53167  to the right, agree=0.799, adj=0.061,
(0 split)
##
## Node number 4: 207 observations,    complexity param=0.01663924
##   mean=183990.5, MSE=3.585262e+09
##   left son=8 (137 obs) right son=9 (70 obs)
##   Primary splits:
##       num_bedrooms    < 1.5       to the left,  improve=0.2952904, (0 miss
ing)
##       monthly_cost    < 764       to the left,  improve=0.2228783, (0 miss
ing)
##       sq_footage      < 940.6025  to the left,  improve=0.2185199, (0 miss
ing)
##       total_taxes     < 2953.945  to the left,  improve=0.2160151, (0 miss
ing)
##       num_total_rooms < 4.5       to the left,  improve=0.2120814, (0 miss
ing)
##   Surrogate splits:
##       sq_footage         < 853       to the left,  agree=0.874, adj=0.629,
(0 split)
##       num_total_rooms    < 3.5       to the left,  agree=0.845, adj=0.543,
(0 split)
##       monthly_cost       < 805.5     to the left,  agree=0.773, adj=0.329,
(0 split)
##       total_taxes        < 3140.35   to the left,  agree=0.744, adj=0.243,
(0 split)
##       num_half_bathrooms < 0.965     to the left,  agree=0.691, adj=0.086,
(0 split)
##
## Node number 5: 61 observations,    complexity param=0.0258552
##   mean=344932.6, MSE=1.334551e+10
##   left son=10 (50 obs) right son=11 (11 obs)
##   Primary splits:
##       monthly_cost    < 1302.5    to the left,  improve=0.4183022, (0 miss
ing)
##       sq_footage      < 1102.955  to the left,  improve=0.3724499, (0 miss
ing)
##       total_taxes     < 4024.79   to the left,  improve=0.2772907, (0 miss
ing)
##       price_persqft   < 0.4499131 to the left,  improve=0.2409971, (0 miss
ing)
##       parking_charges < 64.775    to the left,  improve=0.2147612, (0 miss
ing)
##   Surrogate splits:
##       num_total_rooms    < 6.5       to the left,  agree=0.885, adj=0.364,
(0 split)
##       sq_footage         < 1339.5    to the left,  agree=0.869, adj=0.273,
```

```
(0 split)
##       total_taxes        < 4381.68   to the left,  agree=0.852, adj=0.182,
(0 split)
##       num_half_bathrooms < 0.72      to the right, agree=0.836, adj=0.091,
(0 split)
##
## Node number 6: 121 observations,    complexity param=0.07319492
##   mean=405892.4, MSE=2.159689e+10
##   left son=12 (47 obs) right son=13 (74 obs)
##   Primary splits:
##       coop_condo        splits as  LR, improve=0.3689025, (0 missing)
##       approx_year_built < 2004.5    to the left,  improve=0.2996678, (0 mi
ssing)
##       price_persqft     < 0.5847979 to the left,  improve=0.2978430, (0 mi
ssing)
##       num_total_rooms   < 3.5       to the left,  improve=0.2590936, (0 mi
ssing)
##       sq_footage        < 679.415   to the left,  improve=0.2478039, (0 mi
ssing)
##   Surrogate splits:
##       approx_year_built      < 1971      to the left,  agree=0.884, adj=0.
702, (0 split)
##       monthly_cost           < 514.5     to the right, agree=0.851, adj=0.
617, (0 split)
##       price_persqft          < 0.6130331 to the left,  agree=0.818, adj=0.
532, (0 split)
##       community_district_num < 27.5      to the right, agree=0.769, adj=0.
404, (0 split)
##       pct_tax_deductibl      < 48.45333  to the right, agree=0.752, adj=0.
362, (0 split)
##
## Node number 7: 33 observations,    complexity param=0.01348814
##   mean=661121.2, MSE=1.394647e+10
##   left son=14 (17 obs) right son=15 (16 obs)
##   Primary splits:
##       sq_footage             < 1392.81   to the left,  improve=0.3859944,
(0 missing)
##       total_taxes            < 3874.715  to the left,  improve=0.3050610,
(0 missing)
##       monthly_cost           < 1326      to the left,  improve=0.2652581,
(0 missing)
##       num_bedrooms           < 2.5       to the left,  improve=0.2378804,
(0 missing)
##       num_floors_in_building < 21.5      to the left,  improve=0.1998063,
(0 missing)
##   Surrogate splits:
##       total_taxes   < 4366.515  to the left,  agree=0.818, adj=0.625, (0 s
plit)
##       monthly_cost  < 1326      to the left,  agree=0.788, adj=0.562, (0 s
plit)
```

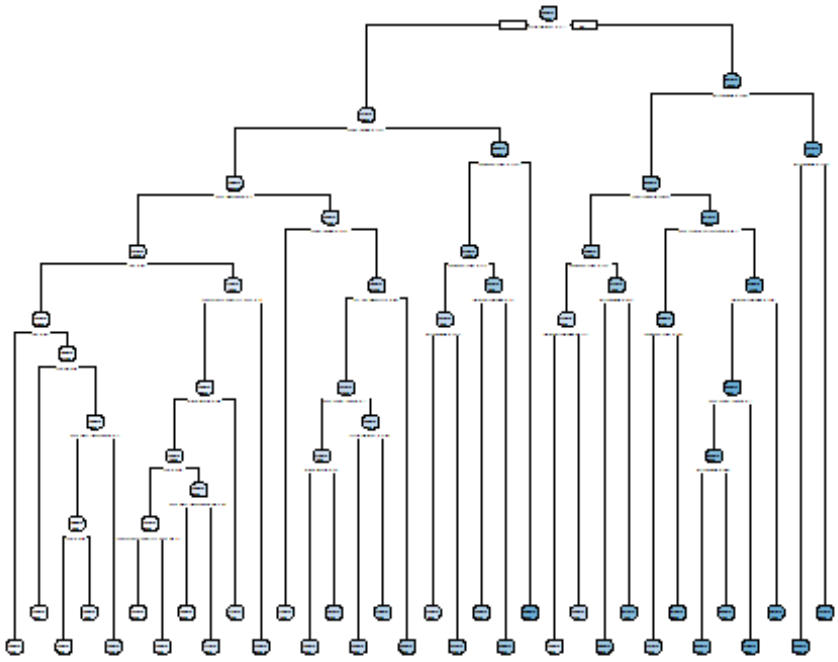
```
##       num_bedrooms  < 2.5       to the left,  agree=0.727, adj=0.438, (0 s
plit)
##       lon           < -73.83932 to the right, agree=0.727, adj=0.438, (0 s
plit)
##       price_persqft < 0.5384526 to the right, agree=0.727, adj=0.438, (0 s
plit)
##
## Node number 8: 137 observations
##   mean=160732.4, MSE=1.951881e+09
##
## Node number 9: 70 observations
##   mean=229509.8, MSE=3.651313e+09
##
## Node number 10: 50 observations
##   mean=309887.8, MSE=7.481011e+09
##
## Node number 11: 11 observations
##   mean=504227.3, MSE=9.045062e+09
##
## Node number 12: 47 observations,    complexity param=0.02291037
##   mean=293892.3, MSE=1.181581e+10
##   left son=24 (23 obs) right son=25 (24 obs)
##   Primary splits:
##       monthly_cost    < 681       to the left,  improve=0.5433477, (0 miss
ing)
##       sq_footage      < 743.825   to the left,  improve=0.5232991, (0 miss
ing)
##       total_taxes     < 2752.269  to the left,  improve=0.4687540, (0 miss
ing)
##       num_total_rooms < 3.5       to the left,  improve=0.4296906, (0 miss
ing)
##       num_bedrooms    < 0.5       to the left,  improve=0.3338656, (0 miss
ing)
##   Surrogate splits:
##       sq_footage        < 743.825   to the left,  agree=0.894, adj=0.783,
(0 split)
##       total_taxes       < 3129.924  to the left,  agree=0.872, adj=0.739,
(0 split)
##       num_total_rooms   < 3.5       to the left,  agree=0.766, adj=0.522,
(0 split)
##       price_persqft     < 0.5539046 to the left,  agree=0.745, adj=0.478,
(0 split)
##       pct_tax_deductibl < 43.415    to the right, agree=0.723, adj=0.435,
(0 split)
##
## Node number 13: 74 observations,    complexity param=0.01939333
##   mean=477027.5, MSE=1.478183e+10
##   left son=26 (29 obs) right son=27 (45 obs)
##   Primary splits:
##       num_floors_in_building < 4.285     to the left,  improve=0.2335069,
```

```
(0 missing)
##       monthly_cost           < 211       to the left,  improve=0.2193306,
(0 missing)
##       total_taxes            < 2143      to the left,  improve=0.2122442,
(0 missing)
##       approx_year_built      < 2006.5    to the left,  improve=0.2079009,
(0 missing)
##       parking_charges        < 174.5133  to the left,  improve=0.1971597,
(0 missing)
##   Surrogate splits:
##       price_persqft          < 0.5718783 to the left,  agree=0.730, adj=0.
310, (0 split)
##       total_taxes            < 2143      to the left,  agree=0.716, adj=0.
276, (0 split)
##       parking_charges        < 114.89    to the left,  agree=0.703, adj=0.
241, (0 split)
##       community_district_num < 24.5      to the left,  agree=0.676, adj=0.
172, (0 split)
##       pct_tax_deductibl      < 46.875    to the right, agree=0.676, adj=0.
172, (0 split)
##
## Node number 14: 17 observations
##   mean=589941.2, MSE=3.455467e+09
##
## Node number 15: 16 observations
##   mean=736750, MSE=1.399019e+10
##
## Node number 24: 23 observations
##   mean=212043.5, MSE=3.314759e+09
##
## Node number 25: 24 observations
##   mean=372330.8, MSE=7.389972e+09
##
## Node number 26: 29 observations,    complexity param=0.01776872
##   mean=403842.7, MSE=1.358682e+10
##   left son=52 (12 obs) right son=53 (17 obs)
##   Primary splits:
##       monthly_cost < 297       to the left,  improve=0.5939465, (0 missing
)
##       kitchen_type splits as  LRL-, improve=0.2566576, (0 missing)
##       total_taxes  < 2087.5    to the left,  improve=0.2457086, (0 missing
)
##       sq_footage   < 698.275   to the left,  improve=0.2435420, (0 missing
)
##       num_bedrooms < 1.5       to the left,  improve=0.2292115, (0 missing
)
##   Surrogate splits:
##       num_bedrooms    < 1.5       to the left,  agree=0.793, adj=0.500, (0
split)
##       num_total_rooms < 3.5       to the left,  agree=0.793, adj=0.500, (0
```

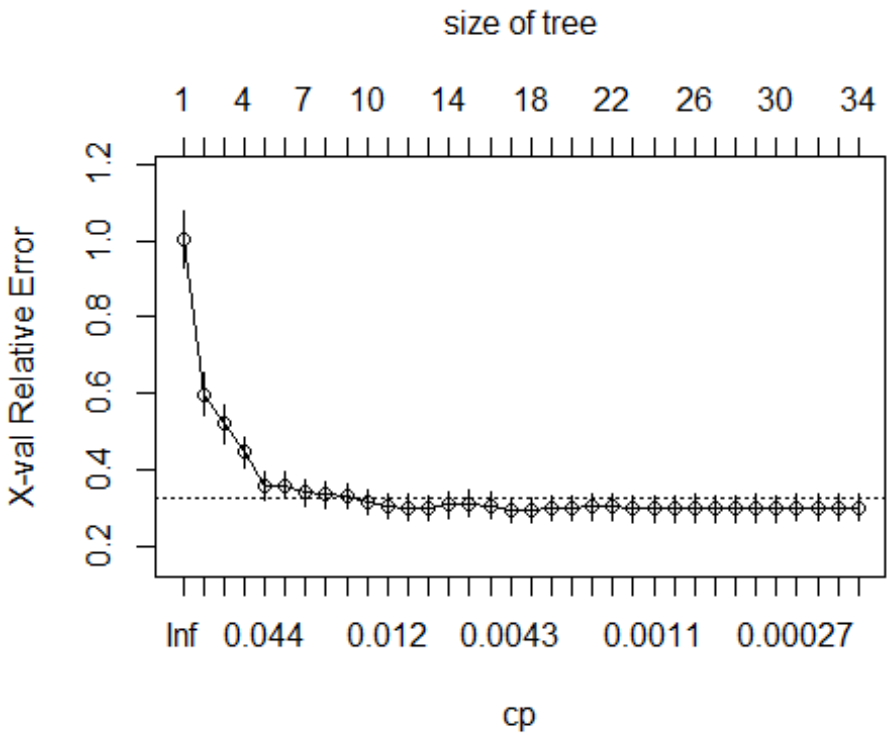
```
split)
##       sq_footage      < 669.49    to the left,  agree=0.793, adj=0.500, (0
split)
##       price_persqft   < 0.6518208 to the right, agree=0.793, adj=0.500, (0
split)
##       total_taxes     < 2306.5    to the left,  agree=0.759, adj=0.417, (0
split)
##
## Node number 27: 45 observations,    complexity param=0.01014447
##   mean=524191.1, MSE=9.875883e+09
##   left son=54 (37 obs) right son=55 (8 obs)
##   Primary splits:
##       parking_charges   < 174.1833  to the left,  improve=0.3006410, (0 mi
ssing)
##       monthly_cost      < 630.5     to the left,  improve=0.2754588, (0 mi
ssing)
##       approx_year_built < 2005.5    to the left,  improve=0.2286395, (0 mi
ssing)
##       total_taxes       < 3674.215  to the left,  improve=0.1941959, (0 mi
ssing)
##       sq_footage        < 878.5     to the left,  improve=0.1876615, (0 mi
ssing)
##   Surrogate splits:
##       price_persqft          < 0.8290999 to the left,  agree=0.956, adj=0.
750, (0 split)
##       lon                    < -73.92553 to the right, agree=0.933, adj=0.
625, (0 split)
##       approx_year_built      < 2009.5    to the left,  agree=0.889, adj=0.
375, (0 split)
##       community_district_num < 29.5      to the left,  agree=0.867, adj=0.
250, (0 split)
##       sq_footage             < 591       to the right, agree=0.867, adj=0.
250, (0 split)
##
## Node number 52: 12 observations
##   mean=296920.8, MSE=1.016437e+10
##
## Node number 53: 17 observations
##   mean=479316.9, MSE=2.236462e+09
##
## Node number 54: 37 observations
##   mean=498854.1, MSE=6.488097e+09
##
## Node number 55: 8 observations
##   mean=641375, MSE=8.843234e+09

yhat = predict(m1, Xtest)
e = yhat - Ytest
sqrt(sum(e^2)/106)
```

```
## [1] 111454.6
```

```
m2 <- rpart(
    formula = Ytrain ~ .,
    data    = Xtrain,
    method  = "anova",
    control = list(cp = 0, xval = 10)
)

rpart.plot(m2)
```



```
plotcp(m2)
```




```
yhat = predict(m2, Xtest)
e = yhat - Ytest
sqrt(sum(e^2)/106)

## [1] 107028.4

jpeg(file = "save_m2.jpeg")

###Tuning
m3 <- rpart(
    formula = Ytrain ~ .,
    data    = Xtrain,
    method  = "anova",
    control = list(minsplit = 10, maxdepth = 12, xval = 10)
)

yhat = predict(m3, Xtest)
e = yhat - Ytest
sqrt(sum(e^2)/106)

## [1] 111454.6

m3$cptable

##           CP nsplit rel error    xerror       xstd
## 1  0.42758149      0 1.0000000 1.0033581 0.07572629
## 2  0.12824221      1 0.5724185 0.6290791 0.05820362
## 3  0.09266116      2 0.4441763 0.5672022 0.05401394
```

```
## 4  0.07319492      3 0.3515151 0.4550638 0.04217392
## 5  0.02585520      4 0.2783202 0.3534294 0.03651200
## 6  0.02291037      5 0.2524650 0.3043492 0.02887121
## 7  0.01939333      6 0.2295546 0.3147219 0.02938829
## 8  0.01776872      7 0.2101613 0.3044802 0.02975724
## 9  0.01663924      8 0.1923926 0.3003660 0.02986459
## 10 0.01348814      9 0.1757533 0.2768037 0.02751636
## 11 0.01014447     10 0.1622652 0.2655842 0.02747896
## 12 0.01000000     11 0.1521207 0.2619472 0.02733595
```

```
# function to get optimal cp
get_cp <- function(x) {
  min    <- which.min(x$cptable[, "xerror"])
  cp <- x$cptable[min, "CP"]
}

# function to get minimum error
get_min_error <- function(x) {
  min    <- which.min(x$cptable[, "xerror"])
  xerror <- x$cptable[min, "xerror"]
}

optimal_tree <- rpart(
    formula = Ytrain ~ .,
    data    = Xtrain,
    method  = "anova",
    control = list(minsplit = 11, maxdepth = 8, cp = 0.01)
    )

pred <- predict(optimal_tree, newdata = Xtrain)
RMSE(pred = pred, obs = Ytrain)
```

```
## [1] 68903.54
```

##RANDOM FORESTS

```
m1 <- randomForest(
  formula = Ytrain ~ .,
  data    = Xtrain
)

m1
```

```
##
## Call:
##  randomForest(formula = Ytrain ~ ., data = Xtrain)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 10
##
```

```
##           Mean of squared residuals: 4408749600
##                     % Var explained: 85.87

which.min(m1$mse)

## [1] 159

# RMSE of this optimal random forest
sqrt(m1$mse[which.min(m1$mse)])

## [1] 65665.63

features <- setdiff(names(Xtrain), Ytrain)

set.seed(1989)

m2 <- tuneRF(
  x          = Xtrain,
  y          = Ytrain,
  ntreeTry   = 500,
  mtryStart  = 5,
  stepFactor = 1.5,
  improve    = 0.01,
  trace      = FALSE      # to not show real-time progress
)

## -0.03910721 0.01
## 0.03347455 0.01
## 0.03056411 0.01
## 0.01855087 0.01
```

```
## 0.008980753 0.01
```