Term Project 390.4- 2019

## R Markdown

pacman::p\_load(dplyr, tidyr, ggplot2, magrittr, stringr, mlr)  
housing\_data = read.csv("housing\_data\_2016\_2017.csv")

##Delete features that are irrelevant to sale price

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$dogs\_allowed, 1, 3) == "yes", 1, 0)) %>%  
 mutate(cats\_allowed = ifelse(substr(housing\_data$cats\_allowed, 1, 3) == "yes", 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 ; else = 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"))  
  
## convert garage\_exists feature to binary  
  
d %<>%  
 mutate(garage\_exists = sjmisc::rec(garage\_exists, rec = "NA = 0 ; else = copy")) ##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))  
  
  
str(d)

## 'data.frame': 2230 obs. of 24 variables:  
## $ approx\_year\_built : int 1955 1955 2004 2002 1949 1938 1950 1960 1960 2005 ...  
## $ community\_district\_num : int 25 25 24 25 26 28 29 28 25 30 ...  
## $ coop\_condo : Factor w/ 2 levels "co-op","condo": 1 1 2 2 1 1 1 1 1 2 ...  
## $ dining\_room\_type : Factor w/ 5 levels "combo","dining area",..: 1 3 1 1 1 1 1 NA NA 5 ...  
## $ full\_address\_or\_zip\_code : Factor w/ 1176 levels " Bayside NY, 11360",..: 1158 562 24 223 497 121 391 941 415 586 ...  
## $ garage\_exists : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ kitchen\_type : Factor w/ 4 levels "combo","eat in",..: 2 2 3 2 2 2 3 3 2 2 ...  
## $ num\_bedrooms : int 2 1 1 3 2 2 1 0 1 1 ...  
## $ num\_floors\_in\_building : int 6 7 1 NA 2 6 NA 2 NA 4 ...  
## $ num\_full\_bathrooms : int 1 1 1 2 1 1 1 1 1 1 ...  
## $ num\_half\_bathrooms : int NA NA NA NA NA NA NA NA NA NA ...  
## $ num\_total\_rooms : int 5 4 3 5 4 4 3 2 4 3 ...  
## $ parking\_charges : Factor w/ 90 levels " NA ","100","105",..: 1 1 1 1 1 1 1 1 41 1 ...  
## $ pct\_tax\_deductibl : int NA NA NA NA 39 NA NA NA NA NA ...  
## $ sale\_price : Factor w/ 316 levels " NA ","100000",..: 107 113 33 252 119 126 38 8 94 250 ...  
## $ sq\_footage : int NA 890 550 NA 675 1000 NA 375 NA 681 ...  
## $ total\_taxes : Factor w/ 294 levels " NA ","100","1024",..: 1 1 255 68 1 1 1 1 1 19 ...  
## $ walk\_score : int 82 89 90 94 71 90 72 93 70 98 ...  
## $ listing\_price\_to\_nearest\_1000: int NA NA NA NA NA NA NA NA NA NA ...  
## $ lat : num 40.7 40.8 40.7 40.8 40.7 ...  
## $ lon : num -73.8 -73.8 -73.9 -73.8 -73.7 ...  
## $ zip\_code : chr "11355" "11354" "11368" "11354" ...  
## $ pets\_allowed : num 0 0 0 0 1 1 0 0 0 0 ...  
## $ monthly\_cost : num 767 604 167 275 660 932 660 514 781 NA ...

##Change variable types

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)$lat, lon = #geocode(full\_address\_or\_zip\_code)$lon )  
#geocoordinates for relevant LIRR stations  
  
lirr\_coord = read.csv("coord.csv")  
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)) )  
#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)

#Let’s first create a matrix with columns that represents missingness

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:

M %<>% select\_if(function(x){sum(x) > 0})

#Now let’s impute missing data using the package. we cannot fit RF models to the entire dataset (it’s 26,000! observations) so we will sample 5 for X1 and for each of the trees and then average. That will be good enough.

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 want  
## 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 want  
## 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 want  
## 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 want  
## to do regression?

## done!  
## missForest iteration 5 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 want  
## to do regression?

## done!  
## missForest iteration 6 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 want  
## to do regression?

## done!  
## missForest iteration 7 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 want  
## 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   
## -305524 -36972 -1025 36580 377157   
##   
## Coefficients: (3 not defined because of singularities)  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -5.197e+07 9.499e+06 -5.471 7.16e-08  
## approx\_year\_built 8.039e+01 2.648e+02 0.304 0.761568  
## community\_district\_num 3.529e+03 1.200e+03 2.941 0.003422  
## coop\_condocondo 1.942e+05 1.598e+04 12.152 < 2e-16  
## dining\_room\_typedining area 2.474e+04 5.484e+04 0.451 0.652057  
## dining\_room\_typeformal 2.882e+04 8.836e+03 3.261 0.001186  
## dining\_room\_typeother 1.703e+04 1.174e+04 1.451 0.147313  
## garage\_exists 1.118e+04 9.344e+03 1.196 0.232155  
## kitchen\_typeeat in -5.655e+03 1.037e+04 -0.545 0.585824  
## kitchen\_typeefficiency -2.087e+04 1.025e+04 -2.035 0.042346  
## num\_bedrooms 4.812e+04 8.211e+03 5.861 8.47e-09  
## num\_floors\_in\_building 3.040e+03 7.311e+02 4.158 3.78e-05  
## num\_full\_bathrooms 2.067e+04 5.467e+04 0.378 0.705519  
## num\_half\_bathrooms -2.748e+04 2.949e+04 -0.932 0.351777  
## num\_total\_rooms 1.550e+04 5.304e+03 2.923 0.003629  
## parking\_charges 4.024e+02 1.040e+02 3.868 0.000124  
## pct\_tax\_deductibl 1.179e+02 9.625e+02 0.122 0.902592  
## sq\_footage 2.776e+01 1.327e+01 2.092 0.036956  
## total\_taxes -3.097e+00 6.056e+00 -0.511 0.609278  
## walk\_score -6.809e+02 3.998e+02 -1.703 0.089198  
## lat 8.402e+05 1.418e+05 5.924 5.92e-09  
## lon -2.350e+05 8.819e+04 -2.664 0.007968  
## pets\_allowed 1.232e+04 7.261e+03 1.697 0.090287  
## monthly\_cost 1.457e+02 1.480e+01 9.841 < 2e-16  
## price\_persqft 3.416e+05 6.792e+04 5.029 6.91e-07  
## shortest\_dist -8.183e+01 6.320e+03 -0.013 0.989675  
## is\_missing\_approx\_year\_built -8.621e+03 3.500e+04 -0.246 0.805523  
## is\_missing\_community\_district\_num 2.270e+03 7.635e+04 0.030 0.976291  
## is\_missing\_dining\_room\_type -4.918e+03 8.076e+03 -0.609 0.542840  
## is\_missing\_kitchen\_type -3.731e+04 2.981e+04 -1.252 0.211322  
## is\_missing\_num\_bedrooms NA NA NA NA  
## is\_missing\_num\_floors\_in\_building -7.114e+03 8.635e+03 -0.824 0.410428  
## is\_missing\_num\_half\_bathrooms 3.035e+03 1.474e+04 0.206 0.836975  
## is\_missing\_num\_total\_rooms NA NA NA NA  
## is\_missing\_parking\_charges -7.907e+03 7.965e+03 -0.993 0.321362  
## is\_missing\_pct\_tax\_deductibl 4.201e+03 8.954e+03 0.469 0.639141  
## is\_missing\_sq\_footage -9.088e+03 6.996e+03 -1.299 0.194523  
## is\_missing\_total\_taxes -1.476e+04 9.449e+03 -1.562 0.118840  
## is\_missing\_monthly\_cost -9.227e+03 2.068e+04 -0.446 0.655623  
## 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 \*\*\*  
## shortest\_dist   
## 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: 74870 on 491 degrees of freedom  
## (1702 observations deleted due to missingness)  
## Multiple R-squared: 0.838, Adjusted R-squared: 0.8261   
## F-statistic: 70.54 on 36 and 491 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\_price\_persqft))

Linear Regression

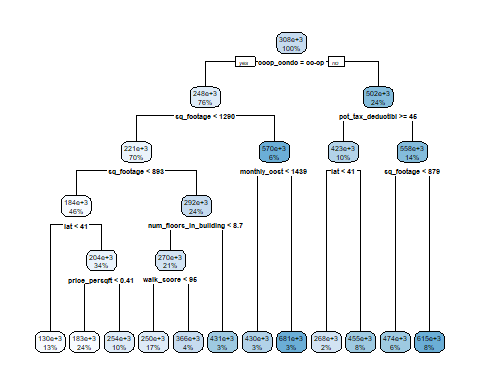
linear = lm(Ytrain ~ ., data = Xtrain)## simple linear model  
summary(linear)

##   
## Call:  
## lm(formula = Ytrain ~ ., data = Xtrain)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -306620 -33030 -29 33533 361122   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -5.419e+07 1.057e+07 -5.128 4.64e-07  
## approx\_year\_built 9.192e+01 2.842e+02 0.323 0.746543  
## community\_district\_num 3.217e+03 1.269e+03 2.536 0.011612  
## coop\_condocondo 2.200e+05 1.803e+04 12.201 < 2e-16  
## dining\_room\_typedining area 2.896e+04 5.358e+04 0.540 0.589214  
## dining\_room\_typeformal 2.488e+04 9.640e+03 2.581 0.010230  
## dining\_room\_typeother 1.586e+04 1.281e+04 1.238 0.216304  
## garage\_exists 1.343e+04 1.053e+04 1.276 0.202749  
## kitchen\_typeeat in -2.688e+03 1.149e+04 -0.234 0.815162  
## kitchen\_typeefficiency -2.544e+04 1.136e+04 -2.239 0.025710  
## num\_bedrooms 3.726e+04 9.065e+03 4.110 4.83e-05  
## num\_floors\_in\_building 2.865e+03 8.196e+02 3.496 0.000528  
## num\_full\_bathrooms 2.659e+04 5.349e+04 0.497 0.619459  
## num\_half\_bathrooms -2.110e+04 3.105e+04 -0.679 0.497243  
## num\_total\_rooms 1.946e+04 5.830e+03 3.338 0.000925  
## parking\_charges 4.414e+02 1.091e+02 4.047 6.27e-05  
## pct\_tax\_deductibl 1.296e+03 1.291e+03 1.004 0.315891  
## sq\_footage 2.092e+01 1.374e+01 1.522 0.128824  
## total\_taxes -2.808e+00 7.034e+00 -0.399 0.689991  
## walk\_score -4.850e+02 4.361e+02 -1.112 0.266771  
## lat 8.550e+05 1.521e+05 5.619 3.68e-08  
## lon -2.559e+05 9.886e+04 -2.588 0.010014  
## pets\_allowed 7.921e+03 7.881e+03 1.005 0.315488  
## monthly\_cost 1.804e+02 1.909e+01 9.446 < 2e-16  
## price\_persqft 2.651e+05 7.418e+04 3.574 0.000397  
## shortest\_dist 4.248e+03 6.965e+03 0.610 0.542276  
## is\_missing\_approx\_year\_built 1.630e+04 4.519e+04 0.361 0.718432  
## is\_missing\_community\_district\_num -8.432e+03 7.433e+04 -0.113 0.909737  
## is\_missing\_dining\_room\_type -1.095e+04 8.741e+03 -1.253 0.210878  
## is\_missing\_kitchen\_type -3.657e+04 3.205e+04 -1.141 0.254462  
## is\_missing\_num\_floors\_in\_building -3.048e+03 9.623e+03 -0.317 0.751590  
## is\_missing\_num\_half\_bathrooms 1.004e+04 1.842e+04 0.545 0.585996  
## is\_missing\_parking\_charges -5.871e+03 8.490e+03 -0.691 0.489688  
## is\_missing\_pct\_tax\_deductibl -2.979e+03 9.604e+03 -0.310 0.756575  
## is\_missing\_sq\_footage -1.556e+04 7.558e+03 -2.059 0.040195  
## is\_missing\_total\_taxes -2.322e+04 1.060e+04 -2.191 0.029037  
## is\_missing\_monthly\_cost -2.361e+03 2.274e+04 -0.104 0.917382  
##   
## (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 \*\*\*  
## shortest\_dist   
## 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: 72570 on 385 degrees of freedom  
## Multiple R-squared: 0.8461, Adjusted R-squared: 0.8317   
## F-statistic: 58.78 on 36 and 385 DF, p-value: < 2.2e-16

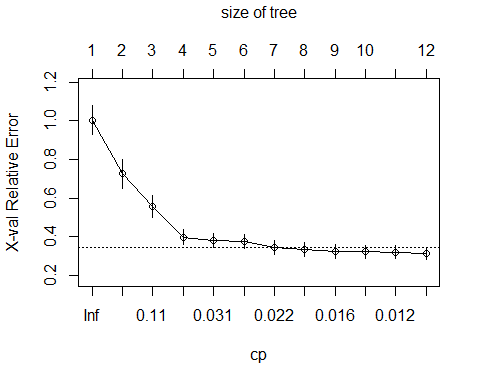
yhat = predict(linear, Xtest)  
  
e = yhat - Ytest  
  
sqrt(sum(e^2) / nrow(Xtest))

## [1] 89226.62

#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)



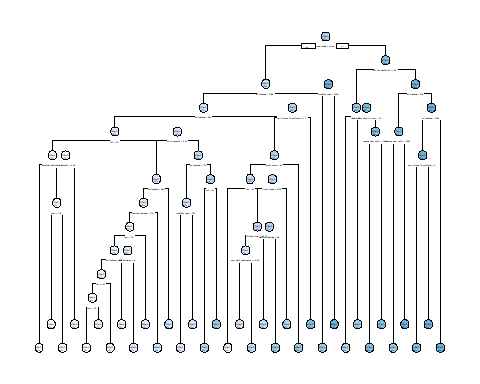
summary(m1)

## Call:  
## rpart(formula = Ytrain ~ ., data = Xtrain, method = "anova")  
## n= 422   
##   
## CP nsplit rel error xerror xstd  
## 1 0.37546175 0 1.0000000 1.0034411 0.07564411  
## 2 0.21400392 1 0.6245383 0.7261829 0.07556486  
## 3 0.05914965 2 0.4105343 0.5569354 0.05709584  
## 4 0.03323586 3 0.3513847 0.3974251 0.03719573  
## 5 0.02939286 4 0.3181488 0.3796016 0.03626881  
## 6 0.02383162 5 0.2887560 0.3746506 0.03814914  
## 7 0.02121911 6 0.2649243 0.3451752 0.03529107  
## 8 0.01615883 7 0.2437052 0.3338604 0.03548722  
## 9 0.01549767 8 0.2275464 0.3243375 0.03539680  
## 10 0.01274120 9 0.2120487 0.3212148 0.03381850  
## 11 0.01146318 10 0.1993075 0.3194338 0.03345413  
## 12 0.01000000 11 0.1878443 0.3147390 0.03133233  
##   
## Variable importance  
## monthly\_cost coop\_condo approx\_year\_built   
## 17 17 14   
## sq\_footage total\_taxes price\_persqft   
## 14 10 9   
## num\_total\_rooms num\_floors\_in\_building num\_bedrooms   
## 4 3 3   
## num\_half\_bathrooms lat pct\_tax\_deductibl   
## 2 2 2   
## lon parking\_charges shortest\_dist   
## 1 1 1   
## walk\_score   
## 1   
##   
## Node number 1: 422 observations, complexity param=0.3754617  
## mean=308191.7, MSE=3.121006e+10   
## left son=2 (322 obs) right son=3 (100 obs)  
## Primary splits:  
## coop\_condo splits as LR, improve=0.3754617, (0 missing)  
## price\_persqft < 0.4447451 to the left, improve=0.3753338, (0 missing)  
## approx\_year\_built < 1970.5 to the left, improve=0.3463094, (0 missing)  
## total\_taxes < 4123.255 to the left, improve=0.3067335, (0 missing)  
## num\_total\_rooms < 4.5 to the left, improve=0.2781774, (0 missing)  
## Surrogate splits:  
## approx\_year\_built < 1976.5 to the left, agree=0.945, adj=0.77, (0 split)  
## monthly\_cost < 430.5 to the right, agree=0.893, adj=0.55, (0 split)  
## price\_persqft < 0.5799058 to the left, agree=0.863, adj=0.42, (0 split)  
## total\_taxes < 4729.05 to the left, agree=0.813, adj=0.21, (0 split)  
## lon < -73.93462 to the right, agree=0.775, adj=0.05, (0 split)  
##   
## Node number 2: 322 observations, complexity param=0.2140039  
## mean=247866, MSE=1.936501e+10   
## left son=4 (297 obs) right son=5 (25 obs)  
## Primary splits:  
## sq\_footage < 1289.735 to the left, improve=0.4520174, (0 missing)  
## monthly\_cost < 1048 to the left, improve=0.4489013, (0 missing)  
## num\_total\_rooms < 4.5 to the left, improve=0.3092985, (0 missing)  
## total\_taxes < 4123.255 to the left, improve=0.2923872, (0 missing)  
## num\_bedrooms < 1.5 to the left, improve=0.2636009, (0 missing)  
## Surrogate splits:  
## monthly\_cost < 1461.5 to the left, agree=0.966, adj=0.56, (0 split)  
## total\_taxes < 4384.158 to the left, agree=0.950, adj=0.36, (0 split)  
## num\_total\_rooms < 6.5 to the left, agree=0.938, adj=0.20, (0 split)  
## num\_bedrooms < 2.5 to the left, agree=0.929, adj=0.08, (0 split)  
## num\_floors\_in\_building < 33.5 to the left, agree=0.925, adj=0.04, (0 split)  
##   
## Node number 3: 100 observations, complexity param=0.03323586  
## mean=502440.4, MSE=1.990038e+10   
## left son=6 (41 obs) right son=7 (59 obs)  
## Primary splits:  
## pct\_tax\_deductibl < 44.51 to the right, improve=0.2199645, (0 missing)  
## lat < 40.70138 to the left, improve=0.2086435, (0 missing)  
## total\_taxes < 3766.725 to the left, improve=0.1944179, (0 missing)  
## num\_bedrooms < 1.5 to the left, improve=0.1805530, (0 missing)  
## sq\_footage < 1276.95 to the left, improve=0.1793721, (0 missing)  
## Surrogate splits:  
## total\_taxes < 3114 to the left, agree=0.97, adj=0.927, (0 split)  
## lat < 40.74632 to the left, agree=0.72, adj=0.317, (0 split)  
## shortest\_dist < 1.857622 to the right, agree=0.70, adj=0.268, (0 split)  
## num\_floors\_in\_building < 6.595 to the left, agree=0.69, adj=0.244, (0 split)  
## parking\_charges < 93.34625 to the left, agree=0.69, adj=0.244, (0 split)  
##   
## Node number 4: 297 observations, complexity param=0.05914965  
## mean=220721.7, MSE=8.985055e+09   
## left son=8 (196 obs) right son=9 (101 obs)  
## Primary splits:  
## sq\_footage < 893.19 to the left, improve=0.2919323, (0 missing)  
## monthly\_cost < 854.5 to the left, improve=0.2916923, (0 missing)  
## num\_bedrooms < 1.5 to the left, improve=0.2343048, (0 missing)  
## price\_persqft < 0.4133491 to the left, improve=0.2264688, (0 missing)  
## total\_taxes < 2451.14 to the left, improve=0.1898755, (0 missing)  
## Surrogate splits:  
## num\_bedrooms < 1.5 to the left, agree=0.882, adj=0.653, (0 split)  
## num\_total\_rooms < 4.5 to the left, agree=0.842, adj=0.535, (0 split)  
## monthly\_cost < 761.5 to the left, agree=0.805, adj=0.426, (0 split)  
## num\_half\_bathrooms < 0.9883333 to the left, agree=0.764, adj=0.307, (0 split)  
## total\_taxes < 3772.21 to the left, agree=0.747, adj=0.257, (0 split)  
##   
## Node number 5: 25 observations, complexity param=0.02939286  
## mean=570340, MSE=2.993613e+10   
## left son=10 (11 obs) right son=11 (14 obs)  
## Primary splits:  
## monthly\_cost < 1439 to the left, improve=0.5172651, (0 missing)  
## num\_floors\_in\_building < 14.5 to the left, improve=0.4783154, (0 missing)  
## price\_persqft < 0.4648582 to the left, improve=0.4288003, (0 missing)  
## community\_district\_num < 25.5 to the left, improve=0.4022045, (0 missing)  
## parking\_charges < 148.69 to the left, improve=0.3577354, (0 missing)  
## Surrogate splits:  
## approx\_year\_built < 1961.5 to the left, agree=0.84, adj=0.636, (0 split)  
## num\_floors\_in\_building < 11.715 to the left, agree=0.84, adj=0.636, (0 split)  
## sq\_footage < 1309.895 to the left, agree=0.80, adj=0.545, (0 split)  
## num\_half\_bathrooms < 1.02 to the right, agree=0.72, adj=0.364, (0 split)  
## price\_persqft < 0.3752277 to the left, agree=0.72, adj=0.364, (0 split)  
##   
## Node number 6: 41 observations, complexity param=0.01549767  
## mean=423073.2, MSE=1.354759e+10   
## left son=12 (7 obs) right son=13 (34 obs)  
## Primary splits:  
## lat < 40.70268 to the left, improve=0.3674748, (0 missing)  
## approx\_year\_built < 1993.5 to the left, improve=0.2555522, (0 missing)  
## num\_half\_bathrooms < 1.043333 to the right, improve=0.2227680, (0 missing)  
## price\_persqft < 0.4993325 to the left, improve=0.2161214, (0 missing)  
## lon < -73.83487 to the left, improve=0.1708481, (0 missing)  
## Surrogate splits:  
## parking\_charges < 80.01643 to the left, agree=0.951, adj=0.714, (0 split)  
## shortest\_dist < 2.114041 to the right, agree=0.951, adj=0.714, (0 split)  
## price\_persqft < 0.4540916 to the left, agree=0.927, adj=0.571, (0 split)  
## num\_half\_bathrooms < 1.01 to the right, agree=0.902, adj=0.429, (0 split)  
## total\_taxes < 3390 to the right, agree=0.878, adj=0.286, (0 split)  
##   
## Node number 7: 59 observations, complexity param=0.02121911  
## mean=557593.9, MSE=1.689575e+10   
## left son=14 (24 obs) right son=15 (35 obs)  
## Primary splits:  
## sq\_footage < 878.5 to the left, improve=0.2803528, (0 missing)  
## monthly\_cost < 699.5 to the left, improve=0.2486606, (0 missing)  
## num\_bedrooms < 1.5 to the left, improve=0.1986494, (0 missing)  
## num\_total\_rooms < 4.5 to the left, improve=0.1740567, (0 missing)  
## num\_half\_bathrooms < 0.865 to the left, improve=0.1285185, (0 missing)  
## Surrogate splits:  
## num\_total\_rooms < 4.5 to the left, agree=0.864, adj=0.667, (0 split)  
## num\_bedrooms < 1.5 to the left, agree=0.797, adj=0.500, (0 split)  
## price\_persqft < 0.7865664 to the right, agree=0.746, adj=0.375, (0 split)  
## num\_half\_bathrooms < 0.83 to the left, agree=0.695, adj=0.250, (0 split)  
## pct\_tax\_deductibl < 40.365 to the left, agree=0.695, adj=0.250, (0 split)  
##   
## Node number 8: 196 observations, complexity param=0.01615883  
## mean=183956.8, MSE=4.364622e+09   
## left son=16 (53 obs) right son=17 (143 obs)  
## Primary splits:  
## lat < 40.71924 to the left, improve=0.2487793, (0 missing)  
## price\_persqft < 0.4133491 to the left, improve=0.2287369, (0 missing)  
## monthly\_cost < 857 to the left, improve=0.1863259, (0 missing)  
## parking\_charges < 132.945 to the left, improve=0.1743516, (0 missing)  
## total\_taxes < 2249.4 to the left, improve=0.1738126, (0 missing)  
## Surrogate splits:  
## total\_taxes < 2019.318 to the left, agree=0.832, adj=0.377, (0 split)  
## price\_persqft < 0.3264372 to the left, agree=0.827, adj=0.358, (0 split)  
## num\_half\_bathrooms < 0.9983333 to the right, agree=0.765, adj=0.132, (0 split)  
## community\_district\_num < 26.5 to the right, agree=0.745, adj=0.057, (0 split)  
## parking\_charges < 43.5475 to the left, agree=0.740, adj=0.038, (0 split)  
##   
## Node number 9: 101 observations, complexity param=0.02383162  
## mean=292067.6, MSE=1.023818e+10   
## left son=18 (87 obs) right son=19 (14 obs)  
## Primary splits:  
## num\_floors\_in\_building < 8.735 to the left, improve=0.3035404, (0 missing)  
## parking\_charges < 112.665 to the left, improve=0.2682958, (0 missing)  
## price\_persqft < 0.4949068 to the left, improve=0.2560616, (0 missing)  
## walk\_score < 91.5 to the left, improve=0.1889427, (0 missing)  
## lat < 40.70825 to the left, improve=0.1826157, (0 missing)  
## Surrogate splits:  
## approx\_year\_built < 1964.5 to the left, agree=0.881, adj=0.143, (0 split)  
## total\_taxes < 4512.72 to the left, agree=0.881, adj=0.143, (0 split)  
## is\_missing\_num\_half\_bathrooms < 0.5 to the right, agree=0.871, adj=0.071, (0 split)  
##   
## Node number 10: 11 observations  
## mean=429954.5, MSE=5.376839e+09   
##   
## Node number 11: 14 observations  
## mean=680642.9, MSE=2.158109e+10   
##   
## Node number 12: 7 observations  
## mean=267571.4, MSE=8.949388e+09   
##   
## Node number 13: 34 observations  
## mean=455088.2, MSE=8.490919e+09   
##   
## Node number 14: 24 observations  
## mean=474480.8, MSE=1.670047e+10   
##   
## Node number 15: 35 observations  
## mean=614585.7, MSE=9.044807e+09   
##   
## Node number 16: 53 observations  
## mean=129830.2, MSE=8.785938e+08   
##   
## Node number 17: 143 observations, complexity param=0.01146318  
## mean=204017.7, MSE=4.168379e+09   
## left son=34 (100 obs) right son=35 (43 obs)  
## Primary splits:  
## price\_persqft < 0.4128651 to the left, improve=0.2532848, (0 missing)  
## num\_floors\_in\_building < 7.045 to the left, improve=0.2387858, (0 missing)  
## parking\_charges < 132.945 to the left, improve=0.2262675, (0 missing)  
## monthly\_cost < 857 to the left, improve=0.2245285, (0 missing)  
## sq\_footage < 782.865 to the left, improve=0.1932615, (0 missing)  
## Surrogate splits:  
## parking\_charges < 128.6812 to the left, agree=0.916, adj=0.721, (0 split)  
## walk\_score < 94.5 to the left, agree=0.888, adj=0.628, (0 split)  
## num\_floors\_in\_building < 7.045 to the left, agree=0.874, adj=0.581, (0 split)  
## num\_half\_bathrooms < 0.895 to the left, agree=0.853, adj=0.512, (0 split)  
## lon < -73.87859 to the right, agree=0.839, adj=0.465, (0 split)  
##   
## Node number 18: 87 observations, complexity param=0.0127412  
## mean=269704.9, MSE=7.245552e+09   
## left son=36 (72 obs) right son=37 (15 obs)  
## Primary splits:  
## walk\_score < 95 to the left, improve=0.2662115, (0 missing)  
## parking\_charges < 112.665 to the left, improve=0.2519489, (0 missing)  
## price\_persqft < 0.4366634 to the left, improve=0.2215021, (0 missing)  
## lat < 40.69952 to the left, improve=0.1946319, (0 missing)  
## lon < -73.881 to the right, improve=0.1943707, (0 missing)  
## Surrogate splits:  
## lon < -73.8632 to the right, agree=0.908, adj=0.467, (0 split)  
## price\_persqft < 0.4742005 to the left, agree=0.885, adj=0.333, (0 split)  
## approx\_year\_built < 1924.5 to the right, agree=0.862, adj=0.200, (0 split)  
## community\_district\_num < 29 to the left, agree=0.862, adj=0.200, (0 split)  
## num\_floors\_in\_building < 6.295 to the left, agree=0.851, adj=0.133, (0 split)  
##   
## Node number 19: 14 observations  
## mean=431035.7, MSE=6.415374e+09   
##   
## Node number 34: 100 observations  
## mean=182710.7, MSE=2.279301e+09   
##   
## Node number 35: 43 observations  
## mean=253568.8, MSE=5.050479e+09   
##   
## Node number 36: 72 observations  
## mean=249658.9, MSE=4.714457e+09   
##   
## Node number 37: 15 observations  
## mean=365925.9, MSE=8.207485e+09

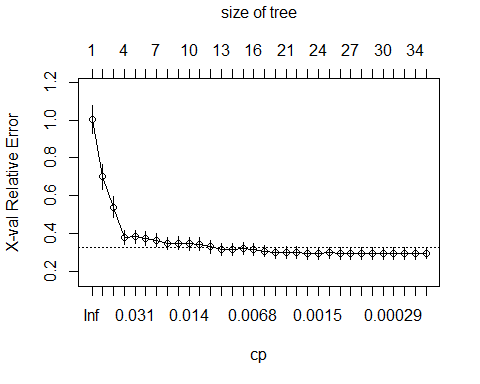
yhat = predict(m1, Xtest)  
e = yhat - Ytest  
sqrt(sum(e^2)/106)

## [1] 116050.1

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] 116341.9

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] 116050.1

m3$cptable

## CP nsplit rel error xerror xstd  
## 1 0.37546175 0 1.0000000 1.0083110 0.07608757  
## 2 0.21400392 1 0.6245383 0.6779121 0.07132898  
## 3 0.05914965 2 0.4105343 0.5066375 0.04874261  
## 4 0.03323586 3 0.3513847 0.3967907 0.03591619  
## 5 0.02939286 4 0.3181488 0.3674374 0.03379264  
## 6 0.02383162 5 0.2887560 0.3512867 0.03285244  
## 7 0.02121911 6 0.2649243 0.3352912 0.03260968  
## 8 0.01615883 7 0.2437052 0.3183722 0.03170800  
## 9 0.01549767 8 0.2275464 0.3110734 0.03014351  
## 10 0.01274120 9 0.2120487 0.3039764 0.02931314  
## 11 0.01146318 10 0.1993075 0.2995381 0.02764982  
## 12 0.01000000 11 0.1878443 0.2975064 0.02848047

# 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] 76567.84

##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: 11  
##   
## Mean of squared residuals: 5266405826  
## % Var explained: 83.13

which.min(m1$mse)

## [1] 500

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

## [1] 72570.01

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   
)

## -0.02710464 0.01   
## 0.04875721 0.01   
## 0.05662543 0.01   
## -0.02580426 0.01

