Project 2: Predicting King County Real Estate Prices

Cam Nguyen, Kalina Gavrilova, Lauren Louie

2022-11-29

Problem Description

Our client is interested in entering the real estate market, and wants our team to help him gather more information regarding the housing market in the King County. Most importantly, he would like to know which factors determine the price of a home, and to predict how much a new set of King County homes could be sold for based on their characteristics.

Objective

Create two models using available data from homes sold in the King County to establish which variables are most important in determining the price of a home, and use these models to predict the prices of new homes that our client could put on the market.

Data Description

The data set available to us had just over 15,000 observations, and included 20 predictive variables about the homes that might have influenced their final price. Some of these variables were related to the actual specifications of the houses (such as size in square feet, number of bedrooms and bathrooms, etc.), while others revolved more around the sale of the houses (date of sale, number of times it was viewed, etc.), and others considered location factors (zip code, whether or not the house had a waterfront location, etc.).

Model 1 - Regression/Classification Tree

Loading Data and removing variables

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(rpart)
library(rpart.plot)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(car)
```

```
## Loading required package: carData
```

```
library(lmtest)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
house <- read.csv("house_8.csv", header = TRUE)
head(house, 10)
```

```
##     X         id Year Month Day day_of_week   price bedrooms bathrooms
## 1   1 5457801925 2015     4  11           6  885000        4      3.75
## 2   2 6613000750 2014    10   1           3 1600000        4      2.75
## 3   3 4330600301 2014     7  18           5  218450        2      1.00
## 4   4 9285800055 2014    10  14           2  619500        4      2.50
## 5   5  326049111 2014     6  26           4  285000        2      1.00
## 6   6 7527000090 2014     8  14           4  540000        4      1.75
## 7   7 3861400030 2014    11  24           1  950000        4      1.75
## 8   8 3623500408 2015     3  30           1 2600000        3      3.00
## 9   9 3365900041 2015     1  21           3  319000        3      1.50
## 10 10 3621059043 2014     5  27           2  293000        4      2.50
##    sqft_living sqft_lot floors waterfront view condition grade sqft_above
## 1         2400     3520    1.0          0    0         3     7       1370
## 2         3680     5000    2.0          0    3         3     9       2480
## 3          840     7425    1.0          0    0         4     6        840
## 4         2210     5077    1.5          0    0         4     8       1480
## 5         1010     7200    1.0          0    0         3     7       1010
## 6         2260    19500    1.0          0    2         3     8       1450
## 7         2210    19025    1.0          0    0         4     7       1460
## 8         3410    16015    2.0          1    4         4    10       2220
## 9         2010    10100    1.0          0    0         4     7       1110
## 10        3250   235063    1.0          0    2         3     9       3250
##    sqft_basement yr_built yr_renovated zipcode     lat     long
## 1           1030     1924         2005   98109 47.6295 -122.346
## 2           1200     1936            0   98105 47.6599 -122.269
## 3              0     1952            0   98166 47.4749 -122.339
## 4            730     1912            0   98126 47.5719 -122.377
## 5              0     1975            0   98155 47.7651 -122.291
## 6            810     1971            0   98074 47.6555 -122.086
## 7            750     1952            0   98004 47.5927 -122.203
## 8           1190     1973            0   98040 47.5721 -122.239
## 9            900     1964            0   98168 47.4738 -122.266
## 10             0     1973            0   98092 47.2582 -122.113
```

```
str(house)
```

```
## 'data.frame':    15053 obs. of  23 variables:
##  $ X            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ id           : num  5.46e+09 6.61e+09 4.33e+09 9.29e+09 3.26e+08 ...
##  $ Year         : int  2015 2014 2014 2014 2014 2014 2014 2015 2015 2014 ...
##  $ Month        : int  4 10 7 10 6 8 11 3 1 5 ...
##  $ Day          : int  11 1 18 14 26 14 24 30 21 27 ...
##  $ day_of_week  : int  6 3 5 2 4 4 1 1 3 2 ...
##  $ price        : num  885000 1600000 218450 619500 285000 ...
##  $ bedrooms     : int  4 4 2 4 2 4 4 3 3 4 ...
##  $ bathrooms    : num  3.75 2.75 1 2.5 1 1.75 1.75 3 1.5 2.5 ...
##  $ sqft_living  : int  2400 3680 840 2210 1010 2260 2210 3410 2010 3250 ...
##  $ sqft_lot     : int  3520 5000 7425 5077 7200 19500 19025 16015 10100 235063 ...
##  $ floors       : num  1 2 1 1.5 1 1 1 2 1 1 ...
##  $ waterfront   : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ view         : int  0 3 0 0 0 2 0 4 0 2 ...
##  $ condition    : int  3 3 4 4 3 3 4 4 4 3 ...
##  $ grade        : int  7 9 6 8 7 8 7 10 7 9 ...
##  $ sqft_above   : int  1370 2480 840 1480 1010 1450 1460 2220 1110 3250 ...
##  $ sqft_basement: int  1030 1200 0 730 0 810 750 1190 900 0 ...
##  $ yr_built     : int  1924 1936 1952 1912 1975 1971 1952 1973 1964 1973 ...
##  $ yr_renovated : int  2005 0 0 0 0 0 0 0 0 0 ...
##  $ zipcode      : int  98109 98105 98166 98126 98155 98074 98004 98040 98168 98092
...
##  $ lat          : num  47.6 47.7 47.5 47.6 47.8 ...
##  $ long         : num  -122 -122 -122 -122 -122 ...
```

```
names(house)
```

```
##  [1] "X"             "id"            "Year"          "Month"
##  [5] "Day"           "day_of_week"   "price"         "bedrooms"
##  [9] "bathrooms"     "sqft_living"   "sqft_lot"      "floors"
## [13] "waterfront"    "view"          "condition"     "grade"
## [17] "sqft_above"    "sqft_basement" "yr_built"      "yr_renovated"
## [21] "zipcode"       "lat"           "long"
```

```
house <- house[ , c(7, 10, 13, 16, 19)]
names(house)
```

```
## [1] "price"       "sqft_living" "waterfront"  "grade"       "yr_built"
```

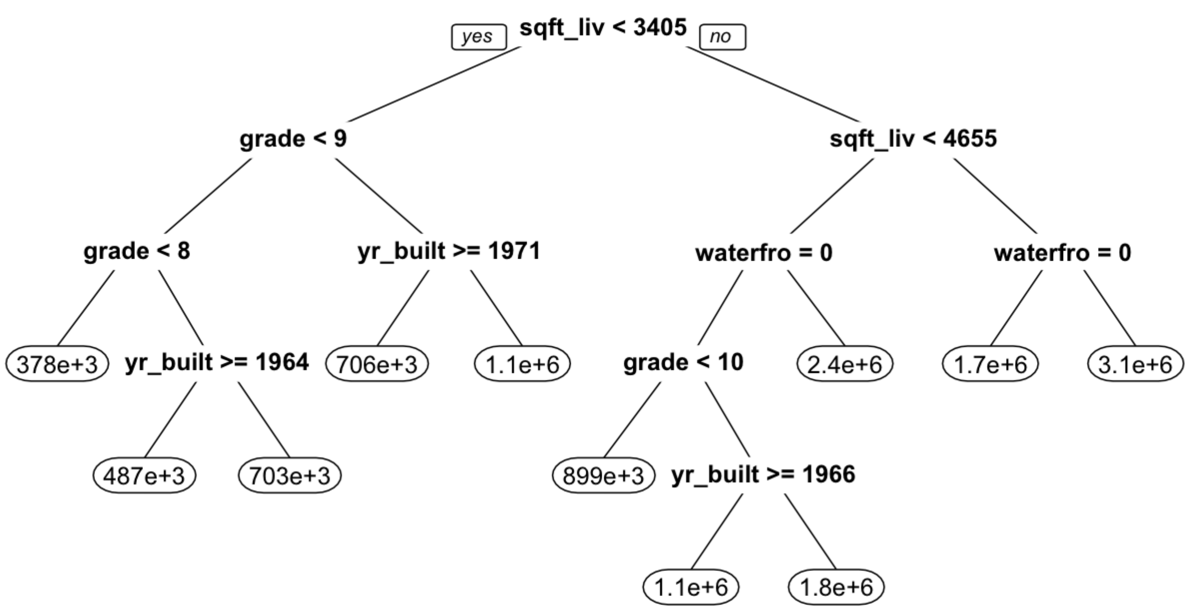
Training Validation Split

```
set.seed(666)


train_index <- sample(1:nrow(house), 0.7 * nrow(house))
valid_index <- setdiff(1:nrow(house), train_index)

train_df <- house[train_index, ]
valid_df <- house[valid_index, ]
nrow(train_df)
```

```
## [1] 10537
```

Regression Tree

```
regress_tr <- rpart(price ~ sqft_living + waterfront + grade + yr_built,
                    data = train_df, method = "anova", maxdepth = 20)
prp(regress_tr)
```




```
predict_train <- predict(regress_tr, train_df)
accuracy(predict_train, train_df$price)
```

```
##                       ME     RMSE      MAE       MPE     MAPE
## Test set 6.515312e-12 230609.5 154969.9 -14.60543 33.45082
```

```
predict_valid <- predict(regress_tr, valid_df)
accuracy(predict_valid, valid_df$price)
```

```
##                   ME     RMSE      MAE      MPE     MAPE
## Test set 1249.893 249148.4 154187.5 -14.2688 33.25046
```

Predict new record

```
new_houses <- read.csv("house_test_8.csv", header = TRUE)
names(new_houses)
```

```
##  [1] "X"              "id"             "Year"           "Month"
##  [5] "Day"            "day_of_week"    "bedrooms"       "bathrooms"
##  [9] "sqft_living"    "sqft_lot"       "floors"         "waterfront"
## [13] "view"           "condition"      "grade"          "sqft_above"
## [17] "sqft_basement" "yr_built"       "yr_renovated"   "zipcode"
## [21] "lat"            "long"
```

```
new_houses <- new_houses[ , c(9, 12, 15, 18)]
names(house)
```

```
## [1] "price"       "sqft_living" "waterfront"  "grade"       "yr_built"
```

```
regress_tr_pred <- predict(regress_tr, newdata = new_houses)
regress_tr_pred
```

```
##         1         2         3         4         5         6         7         8
##  486559.9  378269.5  378269.5  378269.5 1133001.5  378269.5  486559.9  378269.5
##         9        10        11        12        13        14        15        16
##  378269.5 1133001.5  486559.9  378269.5 1081603.9  378269.5 1081603.9  378269.5
##        17        18        19        20
##  378269.5  378269.5  486559.9  378269.5
```

Classification Tree to test accuracy of model

```
house$cat_price <- ifelse(house$price <= mean(house$price, na.rm = TRUE), "0", "1")
table(house$cat_price)
```

```
##
##    0    1
## 9491 5562
```

```
# mean(house$price)
# median(house$price)

house$cat_price <- as.factor(house$cat_price)



# Remove the numerical Price variable to avoid
# confusion (optional, but advisable)
house_cat <- house[,- c(1)]
names(house_cat)
```

```
## [1] "sqft_living" "waterfront"  "grade"       "yr_built"    "cat_price"
```

Training validation split

```
set.seed(666)


train_cat_index <- sample(1:nrow(house_cat), 0.7 * nrow(house_cat))
valid_cat_index <- setdiff(1:nrow(house_cat), train_cat_index)

train_cat_df <- house_cat[train_cat_index, ]
valid_cat_df <- house_cat[valid_cat_index, ]

# check

nrow(train_cat_df)
```

```
## [1] 10537
```

```
nrow(valid_cat_df)
```

```
## [1] 4516
```

```
head(train_cat_df)
```

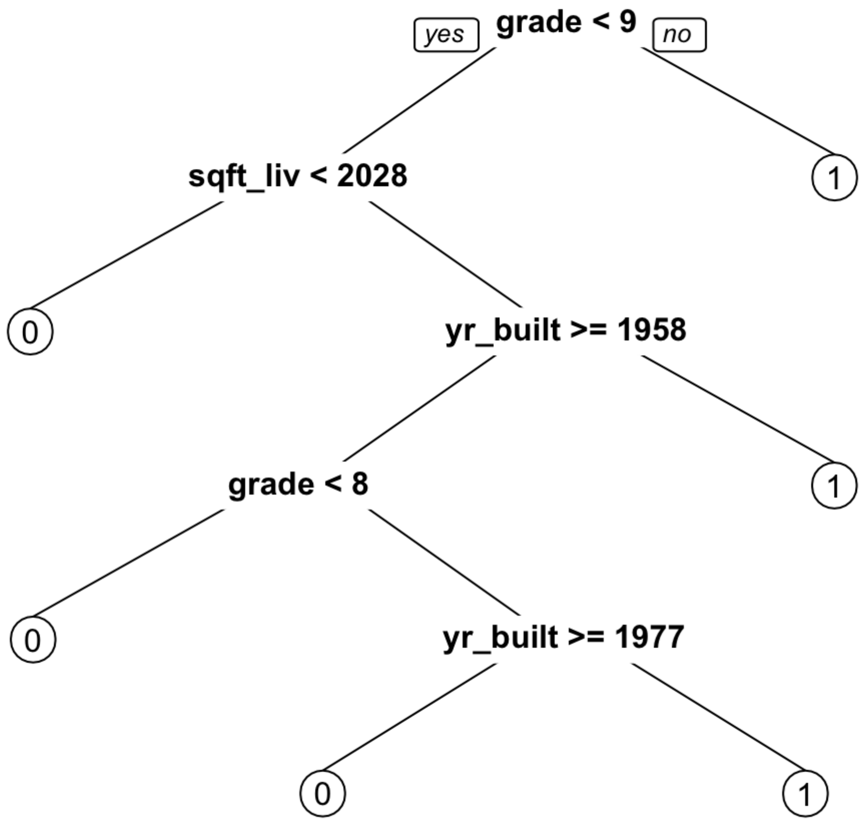
```
##       sqft_living waterfront grade yr_built cat_price
## 1598         1010          0     7     1924         0
## 12926        2510          0     8     1984         0
## 14944        2490          0     8     2000         0
## 13195        1680          0     8     1989         0
## 7291         1560          0     7     1946         1
## 14990        1320          0     8     2014         0
```

```
head(valid_cat_df)
```

```
##    sqft_living waterfront grade yr_built cat_price
## 3          840          0     6     1952         0
## 6         2260          0     8     1971         0
## 9         2010          0     7     1964         0
## 10        3250          0     9     1973         0
## 12        3310          0     9     1992         1
## 15        1480          0     7     1968         1
```

Classification tree

```
class_tr <- rpart(cat_price ~ sqft_living + waterfront + grade + yr_built, data = train_
cat_df, method = "class", maxdepth = 20)

prp(class_tr)
```




```
# The confusion matrices

# training set

class_tr_train_predict <- predict(class_tr, train_cat_df,
                                  type = "class")

t(t(head(class_tr_train_predict,3)))
```

```
##       [,1]
## 1598  0
## 12926 0
## 14944 0
## Levels: 0 1
```

```
confusionMatrix(class_tr_train_predict, train_cat_df$cat_price, positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 6000 1337
##          1  656 2544
##
##                Accuracy : 0.8109
##                  95% CI : (0.8032, 0.8183)
##     No Information Rate : 0.6317
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5781
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.6555
##             Specificity : 0.9014
##          Pos Pred Value : 0.7950
##          Neg Pred Value : 0.8178
##              Prevalence : 0.3683
##          Detection Rate : 0.2414
##    Detection Prevalence : 0.3037
##       Balanced Accuracy : 0.7785
##
##        'Positive' Class : 1
##
```

```
# validation set
class_tr_valid_predict <- predict(class_tr, valid_cat_df,
                                  type = "class")
t(t(head(class_tr_valid_predict,3)))
```

```
##   [,1]
## 3 0
## 6 1
## 9 0
## Levels: 0 1
```

```
confusionMatrix(class_tr_valid_predict, valid_cat_df$cat_price, positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2580  588
##          1  255 1093
##
##                Accuracy : 0.8133
##                  95% CI : (0.8017, 0.8246)
##     No Information Rate : 0.6278
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5838
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.6502
##             Specificity : 0.9101
##          Pos Pred Value : 0.8108
##          Neg Pred Value : 0.8144
##              Prevalence : 0.3722
##          Detection Rate : 0.2420
##    Detection Prevalence : 0.2985
##       Balanced Accuracy : 0.7801
##
##        'Positive' Class : 1
##
```

```
# The probabilities
class_tr_valid_predict_prob <- predict(class_tr, valid_cat_df,
                                  type = "prob")

head(class_tr_valid_predict_prob)
```

```
##            0         1
## 3  0.8555791 0.1444209
## 6  0.3921053 0.6078947
## 9  0.8555791 0.1444209
## 10 0.1398964 0.8601036
## 12 0.1398964 0.8601036
## 15 0.8555791 0.1444209
```

```
# How do the accuracies compare?
```

First try had more variables for the model but after running regression tree took out the variables and still had same accuracy.

Model Process and Analysis

For this model, we chose to build both a regression tree which would predict the numerical price of new records, as well as a classification tree, which can predict the price of new records as a categorical value (high or low price). The variables we considered were: square footage of the house, whether the house was a waterfront property, the overall 'grade' assigned to the house by the county based on its quality, and the age of the house. We felt that these variables would have the biggest impact on the price of a home in this area, and would have the least correlation to each other relative to the other variables available to us- for example, homes with a larger square footage likely also have more bedrooms and bathrooms than homes with a smaller square footage, so by only using the square footage as a variable, we are mitigating the impacts of correlated independent variables on our models.

Our regression tree indicates that the most important factor regarding a house's price is its size- however, all of the variables we selected were represented in the tree. Overall, the error of this tree was relatively low, with an RMSE of 249,148 in the validation data. This measure of error puts the magnitude of the error on the same scale as the value we are predicting, meaning that in this scenario, the model had an average error of $249,148 in its predictions of the home prices.

To create a classification tree from the same variables, we created a new categorical price variable, in which the homes that were below the average price within the data set would be classified as low (0), and those above the average price would be classified as high (1). Interestingly, in this tree the most important factor in determining a house's price was it's grade, and its waterfront location was not a factor that was represented in the tree. This tree had great accuracy, however, with around 81% in both the training and validation sets. The drawback of this tree, however, is that despite its accuracy in sorting homes as high or low value, it does not provide a numerical prediction for the housing prices like the regression tree does.

Model 2 - Linear Regression

Training validation split

```
set.seed(666)


train_index_lr <- sample(1:nrow(house), 0.7 * nrow(house))
valid_index_lr <- setdiff(1:nrow(house), train_cat_index)

train_df_lr <- house[train_index_lr, ]
valid_df_lr <- house[valid_index_lr, ]

# check

nrow(train_df_lr)
```

```
## [1] 10537
```

```
nrow(valid_df_lr)
```

```
## [1] 4516
```

```
head(train_df_lr)
```

```
##        price sqft_living waterfront grade yr_built cat_price
## 1598  501000        1010          0     7     1924         0
## 12926 450000        2510          0     8     1984         0
## 14944 344200        2490          0     8     2000         0
## 13195 265000        1680          0     8     1989         0
## 7291  600000        1560          0     7     1946         1
## 14990 499950        1320          0     8     2014         0
```

```
head(valid_df_lr)
```

```
##     price sqft_living waterfront grade yr_built cat_price
## 3  218450         840          0     6     1952         0
## 6  540000        2260          0     8     1971         0
## 9  319000        2010          0     7     1964         0
## 10 293000        3250          0     9     1973         0
## 12 695000        3310          0     9     1992         1
## 15 547000        1480          0     7     1968         1
```

Training the model

```
price_model <- lm(price ~ sqft_living + waterfront + grade + yr_built,
                     data = train_df_lr)
summary(price_model)
```

```
##
## Call:
## lm(formula = price ~ sqft_living + waterfront + grade + yr_built,
##     data = train_df_lr)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1215416  -117029   -10295    91155  2727724
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.848e+06  1.539e+05   37.99   <2e-16 ***
## sqft_living  1.605e+02  3.624e+00   44.30   <2e-16 ***
## waterfront   7.811e+05  2.430e+04   32.15   <2e-16 ***
## grade        1.460e+05  2.995e+03   48.74   <2e-16 ***
## yr_built    -3.433e+03  8.146e+01  -42.14   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 219900 on 10532 degrees of freedom
## Multiple R-squared:  0.6377, Adjusted R-squared:  0.6376
## F-statistic:  4635 on 4 and 10532 DF,  p-value: < 2.2e-16
```

Model Evaluation

```
price_model_pred_train <- predict(price_model,
                                  train_df_lr)

accuracy(price_model_pred_train, train_df_lr$price)
```

```
##                    ME     RMSE      MAE       MPE     MAPE
## Test set 8.011167e-09 219823.9 144901.8 -8.594001 30.43025
```

```
price_model_pred_valid <- predict(price_model,
                                  valid_df_lr)

accuracy(price_model_pred_valid, valid_df_lr$price)
```

```
##                ME     RMSE      MAE       MPE     MAPE
## Test set 893.0132 228258.5 143086.4 -8.322517 30.08709
```

```
summary(price_model)
```

```
##
## Call:
## lm(formula = price ~ sqft_living + waterfront + grade + yr_built,
##     data = train_df_lr)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1215416  -117029   -10295    91155  2727724
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.848e+06  1.539e+05   37.99   <2e-16 ***
## sqft_living  1.605e+02  3.624e+00   44.30   <2e-16 ***
## waterfront   7.811e+05  2.430e+04   32.15   <2e-16 ***
## grade        1.460e+05  2.995e+03   48.74   <2e-16 ***
## yr_built    -3.433e+03  8.146e+01  -42.14   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 219900 on 10532 degrees of freedom
## Multiple R-squared:  0.6377, Adjusted R-squared:  0.6376
## F-statistic:  4635 on 4 and 10532 DF,  p-value: < 2.2e-16
```

```
# summary(regress_tr)

vif(price_model)
```

```
## sqft_living  waterfront       grade    yr_built
##    2.415102    1.017402    2.726334    1.263326
```

```
bptest(price_model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  price_model
## BP = 1526.1, df = 4, p-value < 2.2e-16
```

Predicting

```
price_model_pred_new <- predict(price_model,
                                newdata = new_houses, interval = "confidence")
price_model_pred_new
```

```
##          fit       lwr       upr
## 1   617744.2  612949.9  622538.6
## 2   672603.9  662019.4  683188.5
## 3   363620.6  358449.6  368791.5
## 4   406866.4  401028.3  412704.5
## 5  1017877.8 1008483.1 1027272.5
## 6   354402.8  347783.4  361022.2
## 7   382680.9  374344.2  391017.5
## 8   386268.6  380784.9  391752.3
## 9   616293.7  607650.7  624936.8
## 10 1157145.7 1143425.0 1170866.4
## 11  579266.3  574288.0  584244.6
## 12  380022.4  373344.8  386699.9
## 13 1653153.1 1606040.0 1700266.3
## 14  226949.4  219637.0  234261.8
## 15  963921.9  954920.0  972923.7
## 16  380213.5  375055.2  385371.8
## 17  346089.9  338646.0  353533.8
## 18  477211.6  463875.3  490547.9
## 19  650071.5  644729.8  655413.1
## 20  343515.5  338368.1  348662.9
```

Model Process and Analysis

The statistical results of our model indicate that all of the variables are statistically significant. This means that the results of our model are highly unlikely to be based on random chance alone, and that the variables selected are playing a role in determining the price of the homes overall. The square footage variable had a coefficient of 160.5- this means that for each 1 sq. ft. increase in size, the home's price is estimated to increase by $160.50. Applying the same principal to the rest of the variables, waterfront location increased a home's price by $781,100, a 1-point increase in a home's grade increased its price by $146,000, and a 1-year increase in a home's age decreased its price by $3,433. These relationships all make sense- homes that are larger, are on the water, and have higher grades assigned to them regarding their quality are likely going to be worth more money, while homes that are older are likely to be worth less money.

Overall, this model resulted in a similar but slightly smaller error as our regression tree, with an average error of $228,259 compared to $249,148 in our previous regression tree model. The accuracy of this model was somewhat good, with an adjusted R-squared value of 0.6376. This value suggests that 63.76% of variation in the housing prices in this data can be attributed to the variables considered by the model.

Model Selection

Based on these results, we would select model 1 because of the visual learning benefits of the tree. Our client is able to see how records are evaluated and houses are priced. The first model is more useful for our client to learn about the housing market. The two models have very close RMSE values and MAPE values. While the error in this model was slightly larger than that of model 2, the difference was relatively small ($20,889) compared to the value of the actual homes. The classification tree is also useful for the client to see what values and variables make a home higher than the median price or lower.

New Homes Prediction

```
regress_tr_pred
```

```
##         1         2         3         4         5         6         7         8 
##  486559.9  378269.5  378269.5  378269.5 1133001.5  378269.5  486559.9  378269.5 
##         9        10        11        12        13        14        15        16 
##  378269.5 1133001.5  486559.9  378269.5 1081603.9  378269.5 1081603.9  378269.5 
##        17        18        19        20 
##  378269.5  378269.5  486559.9  378269.5
```

Using the regression tree in model 1, these are the predicted prices of the 20 new homes based on their characteristics.