Data Scientist Test

Henrique Lispector

September 30th, 2018

Loading the data

```
house_sales <- read.csv("~/Desktop/data_scientist_test/house_sales.csv")
```

Data Pre-Processing

First, we will look at the first six rows of the data to see how the dataset looks like:

```
head(house_sales)
```

```
##     price num_bed num_bath size_house size_lot num_floors is_waterfront
## 1  221900       3     1.00       1180     5650          1             0
## 2  538000       3     2.25       2570     7242          2             0
## 3  180000       2     1.00        770    10000          1             0
## 4  604000       4     3.00       1960     5000          1             0
## 5  510000       3     2.00       1680     8080          1             0
## 6 1225000       4     4.50       5420   101930          1             0
##   condition size_basement year_built renovation_date   zip latitude
## 1         3             0       1955               0 98178 47.51123
## 2         3           400       1951            1991 98125 47.72102
## 3         3             0       1933               0 98028 47.73793
## 4         5           910       1965               0 98136 47.52082
## 5         3             0       1987               0 98074 47.61681
## 6         3          1530       2001               0 98053 47.65612
##   longitude avg_size_neighbor_houses avg_size_neighbor_lot
## 1 -122.2568                     1340                  5650
## 2 -122.3189                     1690                  7639
## 3 -122.2332                     2720                  8062
## 4 -122.3932                     1360                  5000
## 5 -122.0449                     1800                  7503
## 6 -122.0053                     4760                101930
```

The data looks ok.

Now, we need to check if the data types of the variables are coded correctly. First, we check what are their current data type as follows:

```
lapply(house_sales, class)
```

```
## $price
## [1] "integer"
##
## $num_bed
## [1] "integer"
##
## $num_bath
```

```
## [1] "numeric"
##
## $size_house
## [1] "integer"
##
## $size_lot
## [1] "integer"
##
## $num_floors
## [1] "numeric"
##
## $is_waterfront
## [1] "integer"
##
## $condition
## [1] "integer"
##
## $size_basement
## [1] "integer"
##
## $year_built
## [1] "integer"
##
## $renovation_date
## [1] "integer"
##
## $zip
## [1] "integer"
##
## $latitude
## [1] "numeric"
##
## $longitude
## [1] "numeric"
##
## $avg_size_neighbor_houses
## [1] "integer"
##
## $avg_size_neighbor_lot
## [1] "integer"
```

I will change the data type of some variables, since some are categorical and some can be continuous numbers as follows:

```
house_sales1 <- with(house_sales, data.frame(price = as.numeric(price),
                                             num_bed = as.numeric(num_bed),
                                             num_bath = as.numeric(num_bath),
                                             size_house = as.numeric(size_house),
                                             size_lot = as.numeric(size_lot),
                                             num_floors = as.factor(num_floors),
                                             is_waterfront = as.factor(is_waterfront),
                                             condition = as.factor(condition),
                                             size_basement = as.numeric(size_basement),
                                             year_built = as.numeric(year_built),
                                             renovation_date = as.factor(renovation_date),
```

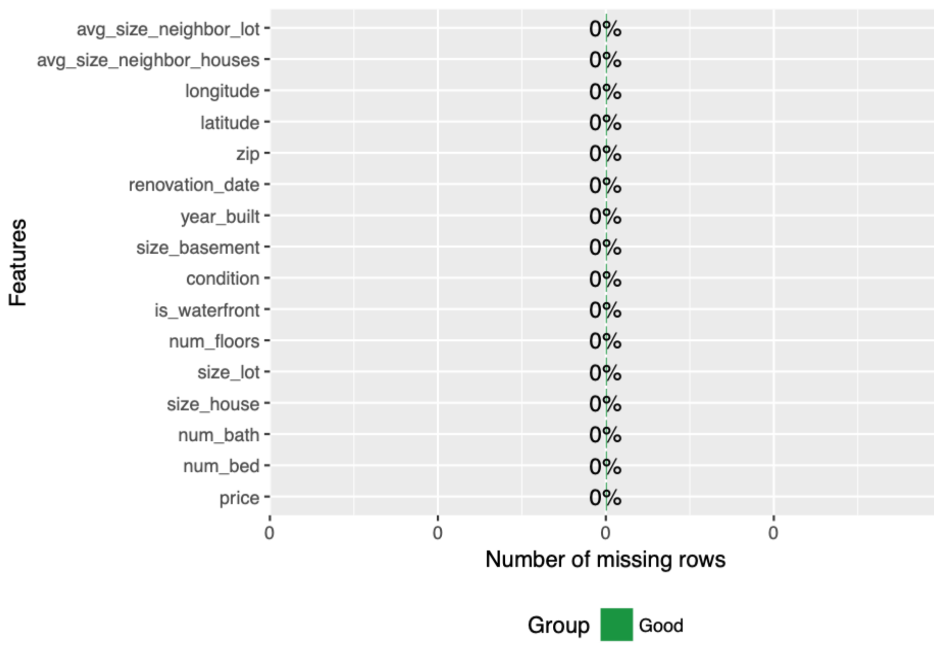
```
                                zip = as.factor(zip),
                                latitude = latitude,
                                longitude = longitude,
                                avg_size_neighbor_houses = as.numeric(avg_size_neighbor_hou
                                avg_size_neighbor_lot = as.numeric(avg_size_neighbor_lot)))
```

Now we need to check if there are any NA's in the dataset, which can cause problems to our analysis. So we load the package "DataExplorer" and make a plot to see if there are any NA's. We could also use the is.na(function), but for simplicity, we will use the following command.

```
library(DataExplorer)
```

```
## Warning: package 'DataExplorer' was built under R version 3.4.4
```

```
plot_missing(house_sales1)
```



Since there are no NA's, we can move on to the next step of our analysis, which is the Exploratory Data Analysis.

Exploratory Data Analysis

In order to get the big picture of the dataset, we can use the summary function below:

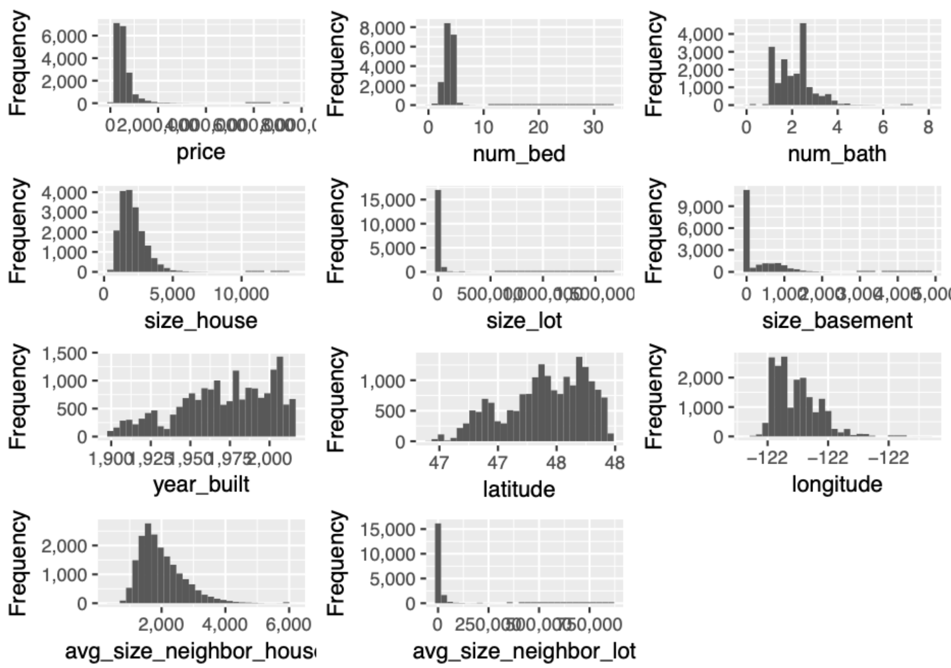
```
summary(house_sales1)
```

```
##      price             num_bed          num_bath       size_house
##  Min.   :  78000   Min.   : 0.000   Min.   :0.000   Min.   :  290
##  1st Qu.: 321838   1st Qu.: 3.000   1st Qu.:1.750   1st Qu.: 1430
```

```
##  Median : 450000   Median : 3.000   Median :2.250   Median : 1920
##  Mean   : 542362   Mean   : 3.373   Mean   :2.119   Mean   : 2084
##  3rd Qu.: 648000   3rd Qu.: 4.000   3rd Qu.:2.500   3rd Qu.: 2560
##  Max.   :7700000   Max.   :33.000   Max.   :8.000   Max.   :13540
##
##     size_lot       num_floors  is_waterfront condition size_basement
##  Min.   :    520   1  :9124   0:18307       1:   26   Min.   :   0.0
##  1st Qu.:   5050   1.5:1617   1:  141       2:  150   1st Qu.:   0.0
##  Median :   7600   2  :7030                 3:11941   Median :   0.0
##  Mean   :  15036   2.5: 144                 4: 4865   Mean   : 293.6
##  3rd Qu.:  10625   3  : 525                 5: 1466   3rd Qu.: 570.0
##  Max.   :1651359   3.5:   8                           Max.   :4820.0
##
##    year_built   renovation_date      zip           latitude
##  Min.   :1900   0      :17661   98103  :  512   Min.   :47.16
##  1st Qu.:1952   2014   :   77   98038  :  504   1st Qu.:47.47
##  Median :1975   2003   :   34   98115  :  495   Median :47.57
##  Mean   :1971   2013   :   33   98117  :  478   Mean   :47.56
##  3rd Qu.:1997   2007   :   30   98034  :  477   3rd Qu.:47.68
##  Max.   :2015   2000   :   29   98052  :  475   Max.   :47.78
##                 (Other):  584   (Other):15507
##    longitude      avg_size_neighbor_houses avg_size_neighbor_lot
##  Min.   :-122.5   Min.   : 399             Min.   :   651
##  1st Qu.:-122.3   1st Qu.:1490             1st Qu.:  5100
##  Median :-122.2   Median :1840             Median :  7611
##  Mean   :-122.2   Mean   :1988             Mean   : 12572
##  3rd Qu.:-122.1   3rd Qu.:2370             3rd Qu.: 10050
##  Max.   :-121.3   Max.   :6110             Max.   :858132
##
```

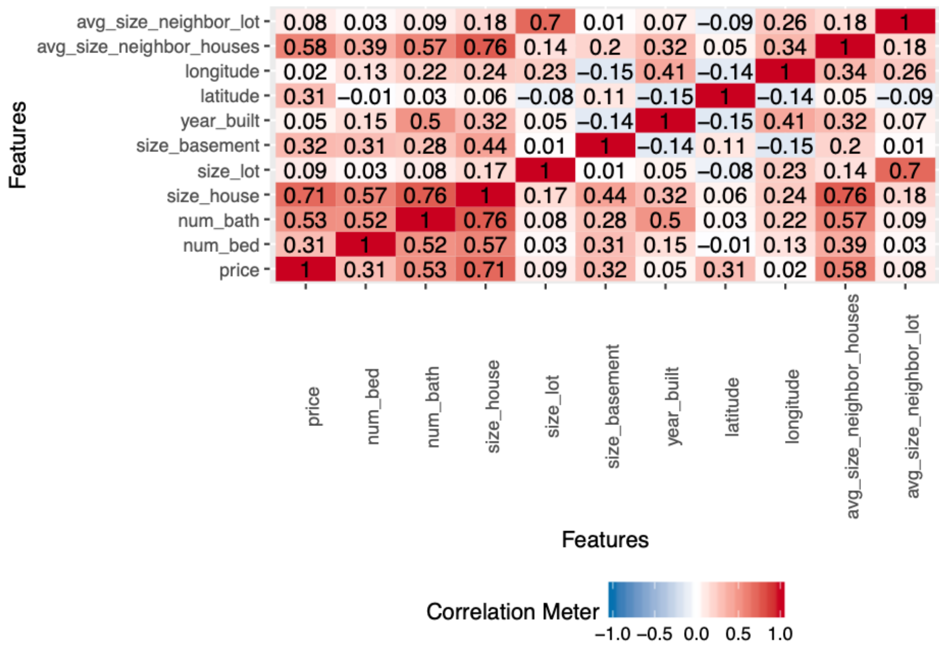
For a more visual appealing exploration, we can graph histograms of the numerical variables:

```
plot_histogram(house_sales1)
```



And do a correlation plot:

```
plot_correlation(house_sales1, type = 'continuous')
```



We can see that "size_house" is the variable with the strongest correlation with "price". Others such as "avg_size_neighbor_houses" and "num_bath" also appear to have some correlation with "price", which might suggest that they will be present in a future regression model.

We could also use box plots to look fore outliers in our data, but due to the short time available, I will skip this analysis.

Partitioning the data

Before we apply a regression algorithm and start making predictions, it is wise to partition our data into two subsets: a training data subset, and a test data subset. We will apply the algorithm to the training data, test it on the test data, and check the performance of predictions. The "set.seed" function will assign a random constant value to the runs, so the analysis is simplified. In addition, this training data will contain a random sample of 70% from the "house_sales1" dataset, while the other 30% will be assigned to the test data subset.

```
set.seed(5)
index_training <- sample(1:nrow(house_sales1), round(0.7*nrow(house_sales1)))
training_data <- house_sales1[index_training,]
test_data <- house_sales1[-index_training,]
```

Fitting the Models and Making and Evaluating Predictions

Since we saw that "size_house" was the variable with the highest correlation with "price", let's start with that variable to run our first model:

```r
test_model1 <- lm(price ~ size_house, training_data)
summary(test_model1)
```

```
##
## Call:
## lm(formula = price ~ size_house, data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1501326  -148374   -24821   106038  4271091
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -48239.724   5754.029  -8.384   <2e-16 ***
## size_house     282.834      2.526 111.973   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 264600 on 12912 degrees of freedom
## Multiple R-squared:  0.4927, Adjusted R-squared:  0.4926
## F-statistic: 1.254e+04 on 1 and 12912 DF,  p-value: < 2.2e-16
```

From the output of the summary of model1, we can conclude that yes, "size_house" affects "price". Now, let's make the predictions and get our errors. We will use two measurements of error that are common: RMSE (Root Mean Squared Error) and MAPE (Mean Absolute Percentage Error)

```r
pred1 <- predict(test_model1, test_data)

difference_1 <- abs(pred1 - test_data$price)

rmse1 <- sqrt(mean(difference_1^2))
rmse1
```

```
## [1] 261412.4
```

```r
mape1 <- mean(difference_1/test_data$price)
mape1
```

```
## [1] 0.3565869
```

We have our first results, but can we improve it? I believe so. Let's try to put "avg_size_neighbor_houses" into our model, since it had a somewhat significant correlation with "price".

```r
test_model2 <- lm(price ~ size_house + avg_size_neighbor_houses , training_data)
summary(test_model2)
```

```
##
## Call:
## lm(formula = price ~ size_house + avg_size_neighbor_houses, data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1265788  -146652   -23112   107799  4437624
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               -1.033e+05  7.119e+03  -14.51   <2e-16 ***
```

```
## size_house                2.455e+02  3.817e+00   64.32   <2e-16 ***
## avg_size_neighbor_houses  6.698e+01  5.162e+00   12.98   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 262900 on 12911 degrees of freedom
## Multiple R-squared:  0.4992, Adjusted R-squared:  0.4991
## F-statistic:  6434 on 2 and 12911 DF,  p-value: < 2.2e-16
```

```r
pred2 <- predict(test_model2, test_data)

difference_2 <- abs(pred2 - test_data$price)
rmse2 <- sqrt(mean(difference_2^2))
rmse2
```

```
## [1] 260017.8
```

```r
mape2 <- mean(difference_2/test_data$price)
mape2
```

```
## [1] 0.3542602
```

We see that both the RMSE and the MAPE were slightly reduced. Let's try to insert the "num_bath" variable into the model.

```r
test_model3 <- lm(price ~ size_house + avg_size_neighbor_houses + num_bath , training_data)
summary(test_model3)
```

```
##
## Call:
## lm(formula = price ~ size_house + avg_size_neighbor_houses +
##     num_bath, data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1269802  -146433   -23070   107226  4430027
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -1.011e+05  7.988e+03  -12.65   <2e-16 ***
## size_house                2.473e+02  4.798e+00   51.55   <2e-16 ***
## avg_size_neighbor_houses  6.698e+01  5.162e+00   12.98   <2e-16 ***
## num_bath                 -2.823e+03  4.557e+03   -0.62    0.536
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 262900 on 12910 degrees of freedom
## Multiple R-squared:  0.4992, Adjusted R-squared:  0.4991
## F-statistic:  4290 on 3 and 12910 DF,  p-value: < 2.2e-16
```

```r
pred3 <- predict(test_model3, test_data)

difference_3 <- abs(pred3 - test_data$price)
rmse3 <- sqrt(mean(difference_3^2))
rmse3
```

```
## [1] 259998.3
```

```r
mape3 <- mean(difference_3/test_data$price)
mape3
```

```
## [1] 0.3542405
```

Here we see something strnage. Although we see that the "num_bath" variable is not significant to the regression model, meaning, the null hypothesis that "num_bath" is not significant is not rejected, it does improve our model by a little bit. Let's try to add now "size_basement" into our model.

```r
test_model4 <- lm(price ~ size_house + avg_size_neighbor_houses + num_bath + size_basement , training_da
summary(test_model4)
```

```
##
## Call:
## lm(formula = price ~ size_house + avg_size_neighbor_houses +
##     num_bath + size_basement, data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1296626  -145787   -24160   107267  4440290
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -1.033e+05  7.980e+03 -12.939  < 2e-16 ***
## size_house                2.329e+02  5.237e+00  44.460  < 2e-16 ***
## avg_size_neighbor_houses  7.460e+01  5.272e+00  14.149  < 2e-16 ***
## num_bath                 -3.076e+02  4.564e+03  -0.067    0.946
## size_basement             4.061e+01  5.953e+00   6.822 9.39e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 262400 on 12909 degrees of freedom
## Multiple R-squared:  0.501,  Adjusted R-squared:  0.5008
## F-statistic:  3240 on 4 and 12909 DF,  p-value: < 2.2e-16
```

```r
pred4 <- predict(test_model4, test_data)

difference_4 <- abs(pred4 - test_data$price)
rmse4 <- sqrt(mean(difference_4^2))
rmse4
```

```
## [1] 259927.9
```

```r
mape4 <- mean(difference_4/test_data$price)
mape4
```

```
## [1] 0.3536577
```

Again we see a small improvement, as the errors got lower. Now, what if we took out the "num_bath" variable from the model, since it is not significant to the regression. Do we get better results?

```r
test_model5 <- lm(price ~ size_house + avg_size_neighbor_houses + size_basement , training_data)
summary(test_model5)
```

```
##
## Call:
## lm(formula = price ~ size_house + avg_size_neighbor_houses +
##     size_basement, data = training_data)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1296213  -145800   -24155   107251  4441121
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -1.035e+05  7.107e+03  -14.56  < 2e-16 ***
## size_house                2.326e+02  4.248e+00   54.77  < 2e-16 ***
## avg_size_neighbor_houses  7.460e+01  5.271e+00   14.15  < 2e-16 ***
## size_basement             4.064e+01  5.933e+00    6.85 7.72e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 262400 on 12910 degrees of freedom
## Multiple R-squared:  0.501,  Adjusted R-squared:  0.5009
## F-statistic:  4321 on 3 and 12910 DF,  p-value: < 2.2e-16
```

```
pred5 <- predict(test_model5, test_data)

difference_5 <- abs(pred5 - test_data$price)
rmse5 <- sqrt(mean(difference_5^2))
rmse5
```

```
## [1] 259930
```

```
mape5 <- mean(difference_5/test_data$price)
mape5
```

```
## [1] 0.3536582
```

The errors get higher by a little bit. So let's put "num_bath" back into our model and now try to run the model with a few more variables to see if we get a significant improvement.

```
test_model6 <- lm(price ~ num_bed + num_bath + size_house + size_lot + num_floors + is_waterfront + con
summary(test_model6)
```

```
##
## Call:
## lm(formula = price ~ num_bed + num_bath + size_house + size_lot +
##     num_floors + is_waterfront + condition + size_basement +
##     year_built, data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1797522  -123176   -14356    97530  3876399
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    6.153e+06  2.034e+05  30.254  < 2e-16 ***
## num_bed       -5.613e+04  2.779e+03 -20.195  < 2e-16 ***
## num_bath       6.855e+04  4.875e+03  14.062  < 2e-16 ***
## size_house     3.027e+02  4.142e+00  73.077  < 2e-16 ***
## size_lot      -2.402e-01  4.991e-02  -4.814 1.50e-06 ***
## num_floors1.5 -1.309e+04  8.226e+03  -1.592 0.111516
## num_floors2    1.351e+04  6.563e+03   2.059 0.039511 *
## num_floors2.5  9.513e+04  2.371e+04   4.012 6.06e-05 ***
```

```
## num_floors3     1.892e+05  1.395e+04  13.560  < 2e-16 ***
## num_floors3.5   3.380e+05  9.696e+04   3.486 0.000492 ***
## is_waterfront1  8.055e+05  2.402e+04  33.527  < 2e-16 ***
## condition2     -5.601e+04  5.665e+04  -0.989 0.322778
## condition3     -4.631e+03  5.196e+04  -0.089 0.928971
## condition4      7.789e+03  5.198e+04   0.150 0.880884
## condition5      3.200e+04  5.235e+04   0.611 0.541108
## size_basement  -3.628e+01  6.409e+00  -5.661 1.54e-08 ***
## year_built     -3.147e+03  1.023e+02 -30.749  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 237000 on 12897 degrees of freedom
## Multiple R-squared:  0.5933, Adjusted R-squared:  0.5928
## F-statistic:  1176 on 16 and 12897 DF,  p-value: < 2.2e-16
```

```r
pred6 <- predict(test_model6, test_data)

difference_6 <- abs(pred6 - test_data$price)
rmse6 <- sqrt(mean(difference_6^2))
rmse6
```

```
## [1] 235024.5
```

```r
mape6 <- mean(difference_6/test_data$price)
mape6
```

```
## [1] 0.3215144
```

Ok, now we got a significant improvement. If we add two more variables, do we get an improvement?

```r
test_model7 <- lm(price ~ num_bed + num_bath + size_house + size_lot + num_floors + is_waterfront + cond
summary(test_model7)
```

```
##
## Call:
## lm(formula = price ~ num_bed + num_bath + size_house + size_lot +
##     num_floors + is_waterfront + condition + size_basement +
##     year_built + avg_size_neighbor_houses + avg_size_neighbor_lot,
##     data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1476124  -124157   -13751    97262  4070260
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)               6.233e+06  2.005e+05  31.087  < 2e-16 ***
## num_bed                  -5.353e+04  2.743e+03 -19.514  < 2e-16 ***
## num_bath                  6.905e+04  4.797e+03  14.393  < 2e-16 ***
## size_house                2.473e+02  5.096e+00  48.532  < 2e-16 ***
## size_lot                  1.640e-01  6.921e-02   2.370   0.0178 *
## num_floors1.5            -2.543e+03  8.110e+03  -0.314   0.7539
## num_floors2               1.522e+04  6.473e+03   2.352   0.0187 *
## num_floors2.5             1.199e+05  2.337e+04   5.133 2.90e-07 ***
## num_floors3               2.158e+05  1.386e+04  15.574  < 2e-16 ***
## num_floors3.5             3.743e+05  9.543e+04   3.922 8.82e-05 ***
```

```
## is_waterfront1           8.002e+05  2.364e+04  33.848  < 2e-16 ***
## condition2              -2.588e+04  5.576e+04  -0.464   0.6426
## condition3               3.017e+04  5.115e+04   0.590   0.5553
## condition4               4.080e+04  5.116e+04   0.797   0.4252
## condition5               7.030e+04  5.154e+04   1.364   0.1726
## size_basement           -1.650e+01  6.420e+00  -2.570   0.0102 *
## year_built              -3.247e+03  1.011e+02 -32.119  < 2e-16 ***
## avg_size_neighbor_houses  9.293e+01  4.777e+00  19.455  < 2e-16 ***
## avg_size_neighbor_lot    -8.863e-01  1.128e-01  -7.855 4.32e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 233200 on 12895 degrees of freedom
## Multiple R-squared:  0.6064, Adjusted R-squared:  0.6058
## F-statistic:  1104 on 18 and 12895 DF,  p-value: < 2.2e-16
```

```
pred7 <- predict(test_model7, test_data)

difference_7 <- abs(pred7 - test_data$price)
rmse7 <- sqrt(mean(difference_7^2))
rmse7
```

```
## [1] 231947.2
```

```
mape7 <- mean(difference_7/test_data$price)
mape7
```

```
## [1] 0.316373
```

Yes, we got an improvement. What if we add "latitude" and "longitude". Can we improve?

```
test_model8 <- lm(price ~ num_bed + num_bath + size_house + size_lot + num_floors + is_waterfront + cond
summary(test_model8)
```

```
##
## Call:
## lm(formula = price ~ num_bed + num_bath + size_house + size_lot +
##     num_floors + is_waterfront + condition + size_basement +
##     year_built + avg_size_neighbor_houses + avg_size_neighbor_lot +
##     latitude + longitude, data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1339683  -110638   -11696    85050  4043239
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -5.340e+07  2.192e+06 -24.359  < 2e-16 ***
## num_bed                  -4.549e+04  2.558e+03 -17.782  < 2e-16 ***
## num_bath                  5.998e+04  4.467e+03  13.427  < 2e-16 ***
## size_house                2.488e+02  4.743e+00  52.449  < 2e-16 ***
## size_lot                  3.167e-01  6.454e-02   4.907 9.35e-07 ***
## num_floors1.5            -1.119e+04  7.546e+03  -1.483  0.13803
## num_floors2              -3.480e+02  6.038e+03  -0.058  0.95405
## num_floors2.5             1.054e+05  2.175e+04   4.845 1.28e-06 ***
## num_floors3               8.818e+04  1.333e+04   6.614 3.88e-11 ***
## num_floors3.5             2.885e+05  8.879e+04   3.249  0.00116 **
```

```
## is_waterfront1           8.313e+05  2.204e+04  37.721  < 2e-16 ***
## condition2              -1.797e+04  5.186e+04  -0.347  0.72891
## condition3               1.176e+04  4.757e+04   0.247  0.80476
## condition4               4.438e+04  4.758e+04   0.933  0.35100
## condition5               7.652e+04  4.794e+04   1.596  0.11045
## size_basement           -5.233e+01  6.088e+00  -8.595  < 2e-16 ***
## year_built              -1.988e+03  1.008e+02 -19.725  < 2e-16 ***
## avg_size_neighbor_houses  8.813e+01  4.516e+00  19.516  < 2e-16 ***
## avg_size_neighbor_lot    -6.195e-01  1.054e-01  -5.878 4.26e-09 ***
## latitude                 6.092e+05  1.448e+04  42.069  < 2e-16 ***
## longitude               -2.307e+05  1.659e+04 -13.911  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 216900 on 12893 degrees of freedom
## Multiple R-squared:  0.6596, Adjusted R-squared:  0.6591
## F-statistic:  1249 on 20 and 12893 DF,  p-value: < 2.2e-16
```

```r
pred8 <- predict(test_model8, test_data)

difference_8 <- abs(pred8 - test_data$price)
rmse8 <- sqrt(mean(difference_8^2))
rmse8
```

```
## [1] 216544.7
```

```r
mape8 <- mean(difference_8/test_data$price)
mape8
```

```
## [1] 0.2745434
```

Yes, we improved it even more by lowering the MAPE by around 4% and lowering the RMSE from 231947.2 to 216544.7 .

We see that "condition"" may not be significant. If we take it out from our model, can it get better? Let's see.

```r
test_model9 <- lm(price ~ num_bed + num_bath + size_house + size_lot + num_floors + is_waterfront + siz
summary(test_model9)
```

```
##
## Call:
## lm(formula = price ~ num_bed + num_bath + size_house + size_lot +
##     num_floors + is_waterfront + size_basement + year_built +
##     avg_size_neighbor_houses + avg_size_neighbor_lot + latitude +
##     longitude, data = training_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1369151  -111478   -11361    85378  4028615
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -5.117e+07  2.187e+06 -23.400  < 2e-16 ***
## num_bed                  -4.429e+04  2.564e+03 -17.274  < 2e-16 ***
## num_bath                  6.263e+04  4.469e+03  14.015  < 2e-16 ***
## size_house                2.478e+02  4.758e+00  52.086  < 2e-16 ***
## size_lot                  3.029e-01  6.467e-02   4.684 2.85e-06 ***
```

```
## num_floors1.5            -1.153e+04  7.566e+03  -1.523  0.12768
## num_floors2              -6.064e+03  6.013e+03  -1.009  0.31318
## num_floors2.5             9.935e+04  2.182e+04   4.554 5.32e-06 ***
## num_floors3               8.479e+04  1.336e+04   6.344 2.31e-10 ***
## num_floors3.5             2.898e+05  8.911e+04   3.252  0.00115 **
## is_waterfront1            8.336e+05  2.212e+04  37.690  < 2e-16 ***
## size_basement            -4.835e+01  6.097e+00  -7.930 2.36e-15 ***
## year_built               -2.249e+03  9.689e+01 -23.216  < 2e-16 ***
## avg_size_neighbor_houses  8.791e+01  4.527e+00  19.419  < 2e-16 ***
## avg_size_neighbor_lot    -6.015e-01  1.058e-01  -5.688 1.32e-08 ***
## latitude                  5.974e+05  1.446e+04  41.305  < 2e-16 ***
## longitude                -2.215e+05  1.662e+04 -13.329  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 217700 on 12897 degrees of freedom
## Multiple R-squared:  0.657,  Adjusted R-squared:  0.6565
## F-statistic:  1544 on 16 and 12897 DF,  p-value: < 2.2e-16
```

```r
pred9 <- predict(test_model9, test_data)

difference_9 <- abs(pred9 - test_data$price)
rmse9 <- sqrt(mean(difference_9^2))
rmse9
```

```
## [1] 217031.1
```

```r
mape9 <- mean(difference_9/test_data$price)
mape9
```

```
## [1] 0.2768687
```

The model performs a little bit worse without condition. So we decide to leave it there and go with model 8. I think that with an outlier analysis the model could get better. Also, I could try a stepwise approach, meaning, trying all different regression combinations and seeing which one give us the best result. Feature engineering could also be tried here. Due to the limited amount of time, I will go with model 8.

Conclusion

The model that performed the best was model 8. The model is able to predict house prices with an accuracy of about 73% and a RMSE of 216544.7.