Homework 4 - Stat 495

Justin Papagelis

Due Wednesday, Oct. 5th by midnight

Practicing Academic Integrity

If you worked with others or used resources outside of provided course material (anything besides our textbook(s), course materials in Moodle/Git repo, R help menu) to complete this assignment, please acknowledge them below using a bulleted list.

I acknowledge the following individuals with whom I worked on this assignment:

Name(s) and corresponding problem(s)

-

I used the following sources to help complete this assignment:

Source(s) and corresponding problem(s)

-

Homework 4 and 5 Purpose

Homework 4 allows you to practice the two new methods from class recently - GLMs and regression/classification trees. Homework 5 serves as a way to practice our write-up of analyses.

In short, you will be performing some analysis in Homework 4 and then writing it up formally for Homework 5. You will receive some general feedback on Homework 4 as a class, and can use it to refine your models for the write-up in Homework 5. In other words, you may change your models between the two assignments, particularly if you find an issue as you review your work from Homework 4. However, you must submit Homework 4 with potential models for the write-up for both a GLM and tree as discussed below.

Homework 4 - Analysis

For our analysis, we will use the King County, Washington (State) house sales data set, which I am re-hosting from Kaggle. The Kaggle reference is: https://www.kaggle.com/swathiachath/kc-housesales-data/version/1

```
kchouse <- read.csv("https://awagaman.people.amherst.edu/stat495/kc_house_data.csv", header = T)
```

A data dictionary taken from Kaggle is provided for your use (separate file).

Motivation to predict when Price is greater than $500,000

A real estate developer is interested in understanding the features that are predictive of homes selling for more than half a million dollars, and has turned to you, a statistical consultant for help. He wants a model that can be applied to make predictions in this setting and wants to understand how the variables in the model are impacting it.

To practice new techniques from class, in your analysis, you are required to use both an appropriate generalized linear model and tree to address the developer's questions of interest, including a model comparison.

Instructions for your Homework 4 submission

The outline for Homework 4 is next, but I want to include information here for you about what you'll need in Homework 5 so that you can include extra information for yourself in your Homework 4 submission, as desired. Look at the end of the assignment for this, and be sure to read it!

Homework 4 requires the following pieces:

- Exploratory Analysis
- GLM - at least 2 models fit with output and assessment
- Tree - at least 2 models fit with output and assessment

It doesn't matter which you tackle first, the GLMs or the trees.

Be sure you understand how to read output for both GLMs and trees, as this is material covered on the midterm.

Exploratory Analysis

For this analysis, we used the King County, Washington (State) house sales data set (`kchouse`). Reference for data: https://www.kaggle.com/swathiachath/kc-housesales-data/version/1. There is no information on the source of the data. The data set contains the following variables:

- id - notation for a specific house - Numeric
- date - date the house was sold - String
- price - price the house sold for in dollars - Numeric
- bedrooms - number of bedrooms in house - Numeric
- bathrooms- number of bathrooms per bedroom - Numeric
- sqft_living - square footage of the home - Numeric
- sqft_lot - square footage of the lot - Numeric
- floors - total floors (levels) in house - Numeric
- waterfront - house has a waterfront view - Y/N
- view - house has been viewed - Numeric
- condition - How good the condition is (overall) - Numeric
- grade - overall grade given to housing unit (based on King County grading system: 1 poor, 13 excellent) - Numeric
- sqft_above - square footage of house apart from basement - Numeric
- yr_built - the year the house was built - Numeric
- yr_renovated - the year when the house was generated
- zipcode - the zipcode the house is in - Numeric
- lat - Latitude coordinate of house - Numeric
- long - Longitude coordinate of house - Numeric
- sqft_living15 - living room area in 2015 (may or may not have been affected the lot size area) - Numeric
- sqft_lot15 - lot size area in 2015 - Numeric

Before we fit any models, we need to make some adjustments to how the data is being treated by R. For example, we need to change the `date` to a date format and change `waterfront` and `zipcode` to be treated as categorical variables instead of numeric variables. We also changed the missing values from `yr_renovated` and `sqft_basement` to be "na" instead of 0.

```
kchouse <- kchouse %>%
  mutate(date = lubridate::mdy(date),
         waterfront = as.factor(waterfront),
         zipcode = as.numeric(as.factor(zipcode)))
kchouse <- kchouse %>%
  mutate_at(c('yr_renovated', 'sqft_basement'), ~na_if(.,0))
```

After examining `yr_renovated` and `sqft_basement`, it appears that 95.8% and 60.7% of the values, respectively, are missing. From the scatterplots below, there does not seem to be any clear relationship between `yr_renovated` and `price` as well as `sqft_basement` and `price`. Therefore, we will drop these predictors when creating our models.

```
sum(is.na(kchouse$yr_renovated))/nrow(kchouse)
```

```
## [1] 0.957679
```

```
sum(is.na(kchouse$sqft_basement))/nrow(kchouse)
```

```
## [1] 0.607029
```

```
fig1 <- ggplot(data = kchouse, aes(x = yr_renovated, y = price)) +
  geom_point() +
  labs(title = "Scatterplot of Year Renovated and Price", x = "Year Renovated",
       y = "Price")
fig2 <- ggplot(data = kchouse, aes(x = sqft_basement, y = price)) +
  geom_point() +
  labs(title = "Scatterplot of Sq Ft Basement and Price",
       x = "Square Feet Basement", y = "Price")
```

```
grid.arrange(fig1, fig2, ncol = 2)
```

```
## Warning: Removed 20683 rows containing missing values (geom_point).
```

```
## Warning: Removed 13110 rows containing missing values (geom_point).
```

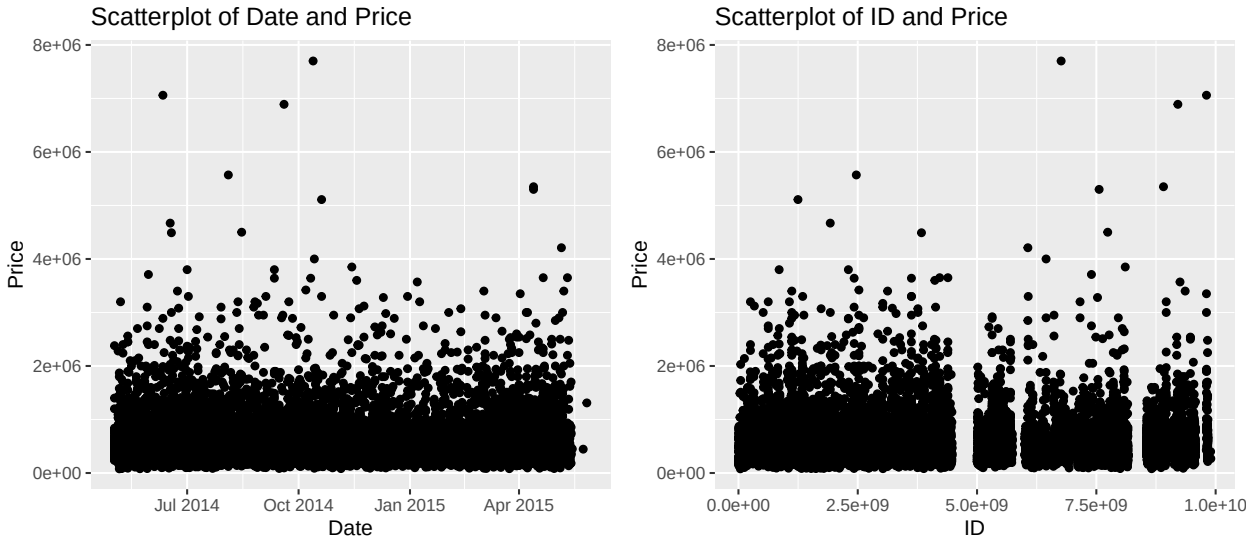
Additionally, from the Scatterplot of Bedrooms and Price, there appears to be an outlier so we will remove that observation.

```
ggplot(data = kchouse, aes(x = bedrooms, y = price)) +
  geom_point() +
  labs(title = "Scatterplot of Bedrooms and Price",
       x = "Bedrooms", y = "Price")
```

Since there are a couple variables such as `date`, `id` that do not have a relationship with `price` as seen in the scatterplots below, we will remove those predictors as well.

```
fig3 <- ggplot(data = kchouse, aes(x = date, y = price)) +
  geom_point() +
  labs(title = "Scatterplot of Date and Price",
       x = "Date", y = "Price")

fig4 <- ggplot(data = kchouse, aes(x = id, y = price)) +
  geom_point() +
  labs(title = "Scatterplot of ID and Price",
       x = "ID", y = "Price")
```

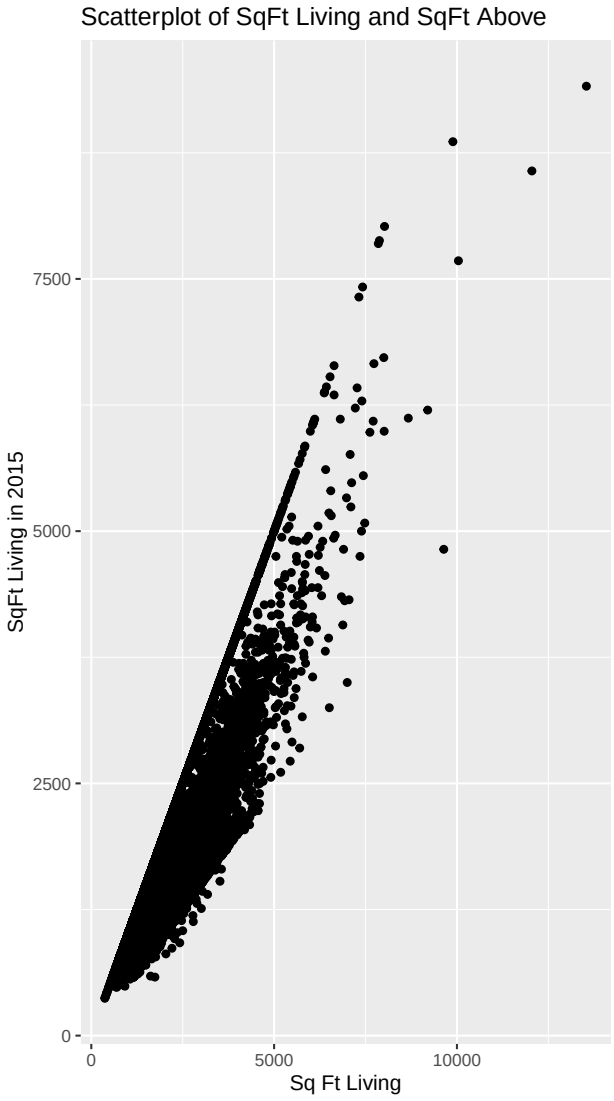
```
grid.arrange(fig3, fig4, ncol = 2)
```

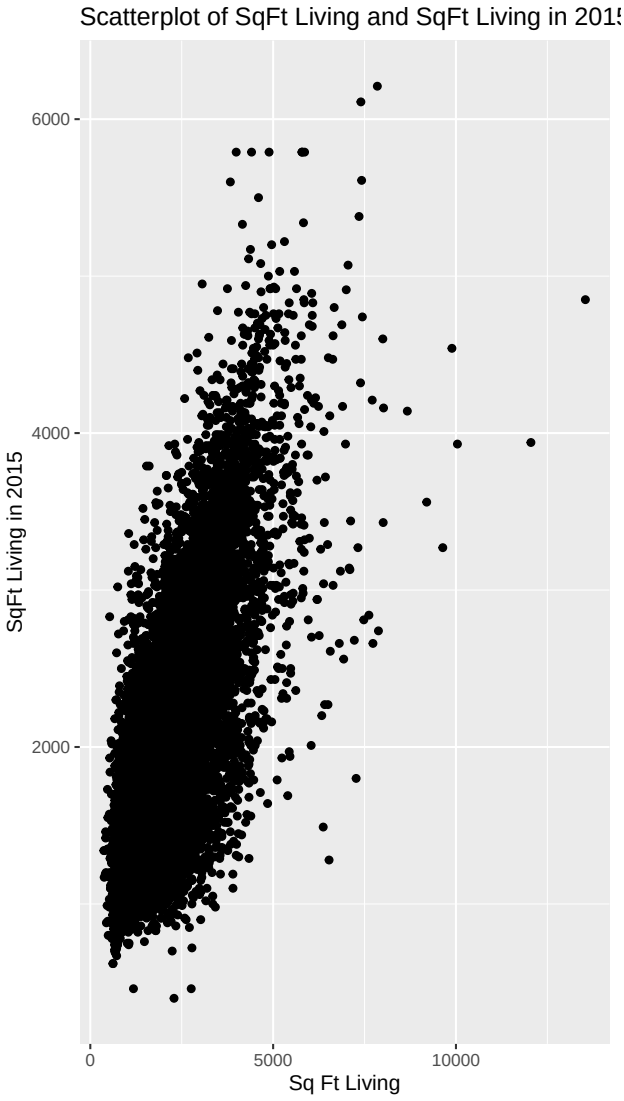


Additionally, it appears that there are some concerns with multicollinearity between `sqft_living` and other variables such as `sqft_living15` and `sqft_above` so we are going to remove `sqft_living` as well.

```
fig5 <- ggplot(data = kchouse, aes(x = sqft_living, y = sqft_above)) +
  geom_point() +
  labs(title = "Scatterplot of SqFt Living and SqFt Above",
       x = "Sq Ft Living", y = "SqFt Living in 2015")

fig6 <- ggplot(data = kchouse, aes(x = sqft_living, y = sqft_living15)) +
  geom_point() +
  labs(title = "Scatterplot of SqFt Living and SqFt Living in 2015",
       x = "Sq Ft Living", y = "SqFt Living in 2015")
```

```
grid.arrange(fig5, fig6, ncol = 2)
```

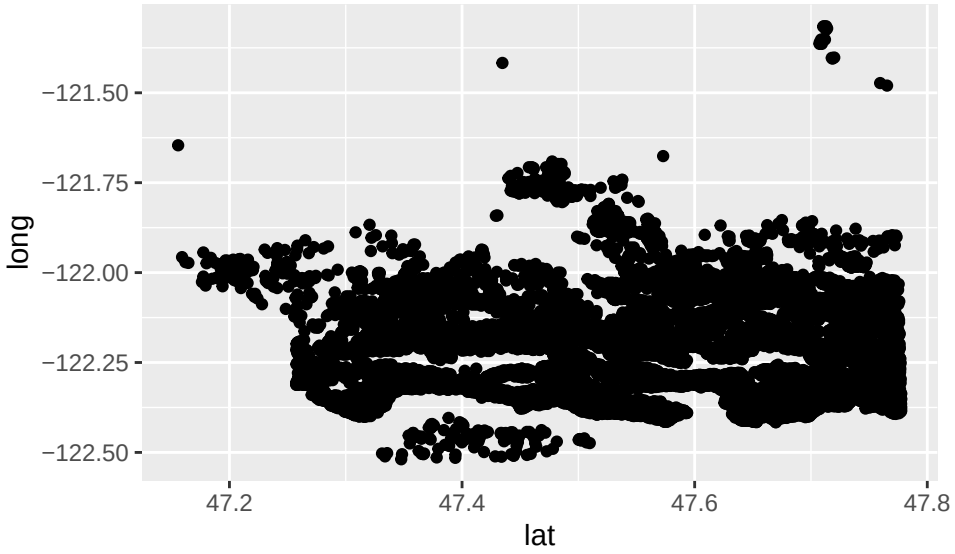





```
kchouse1 <- kchouse %>%
  filter(id != 2402100895) %>%
  select(-c(id,zipcode, yr_renovated, date, sqft_basement, sqft_living))
```

From the scatterplot below of the latitude and the longitude, it seems like the houses are all bunched together in near each other (near Seattle) so the results of this analysis can only be generalized to that region of the state.

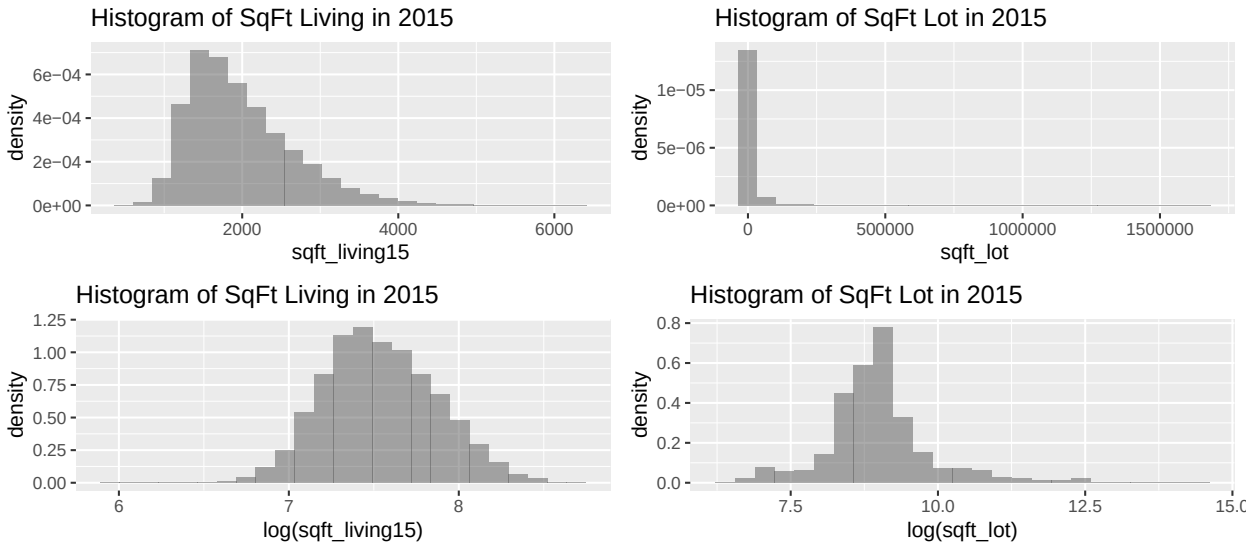
```
ggplot(data = kchouse, aes(x = lat, y = long)) +
  geom_point()
```



It also appears that many of the predictors are right-skewed which means that a log transformation could be helpful for a model. For example, as seen below, `sqft_living15` and `sqft_lot15` before and after a log transformation. This is the same for additional variables as well.

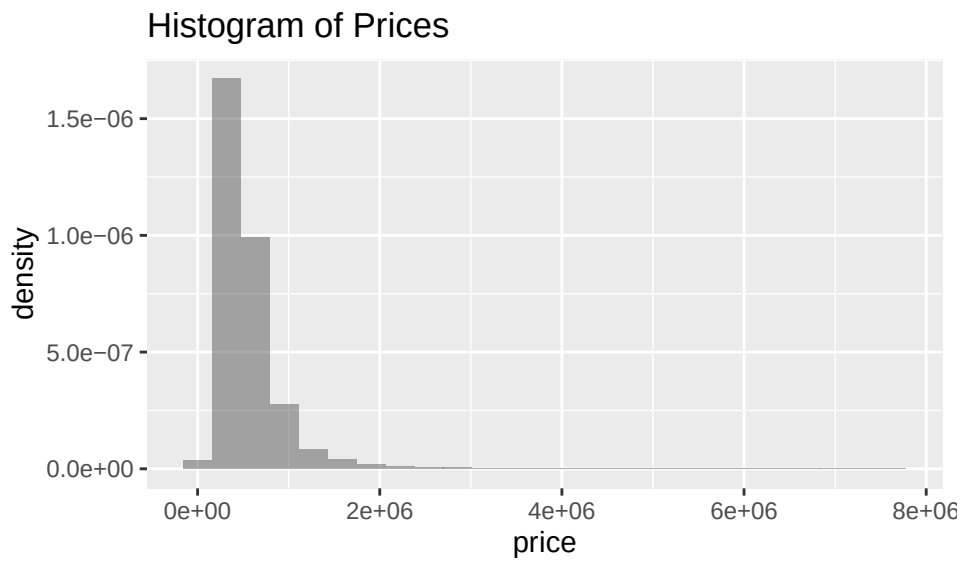
```
fig7 <- gf_dhistogram(~ sqft_living15, data = kchouse) %>%
  gf_labs(title = "Histogram of SqFt Living in 2015")

fig8 <- gf_dhistogram(~ sqft_lot, data = kchouse) %>%
  gf_labs(title = "Histogram of SqFt Lot in 2015")

fig9 <- gf_dhistogram(~ log(sqft_living15), data = kchouse) %>%
  gf_labs(title = "Histogram of SqFt Living in 2015")

fig10 <- gf_dhistogram(~ log(sqft_lot), data = kchouse) %>%
  gf_labs(title = "Histogram of SqFt Lot in 2015")
```

```
grid.arrange(fig7, fig8, fig9, fig10, ncol = 2)
```



From the histogram below, the distribution of `price` is right-skewed with a median of $450,000 and an IQR is $323,000. There appears to be many outliers on the high end for `price`. The minimum of `price` is $78,000 and the maximum is $7,700,000. Since the distribution is right-skewed, if we were predicting price, a log-transformation could be useful.

```
gf_dhistogram(~ price, data = kchouse) %>%
  gf_labs(title = "Histogram of Prices")
```




```
favstats(~ price, data = kchouse)
```

```
##    min     Q1 median     Q3     max   mean     sd     n missing
##  78000 322000 450000 645000 7700000 540297 367368 21597       0
```

Next, since we want to know if the price of the house is greater or less than $500,000, we will transform the price variable so that it is an binary predictor that is `1` if the price of the house is greater than $500,000 and `0` if the price of the house is less than $500,000.

```
kchouse1 <- mutate(kchouse1, price = ifelse(price > 500000, 1, 0))
```

The last thing that we will do before creating the models is create a training and test set. The training set is used to create a model and then the test set is used to see how the model performs in making predictions. We will randomly create a 75/25 split for the training/test set.

```
# create the training and test set for price
set.seed(1)
n <- nrow(kchouse1)
train_index <- sample(1:n, 3/4 * n)
test_index <- setdiff(1:n, train_index)

train <- kchouse1[train_index, ]
test <- kchouse1[test_index, ]
```

GLM Model Fitting

The first models that we are going to be fitting are GLMs (or Generalized Linear Models) which are extensions of ordinary linear regression (fitting a least-squares curve) to cases where the response variables are in an exponential family form. This can be response variables that are binomial, Poisson, gamma, or beta. Since we transformed `price` into a binary variable, it can now be modeled as a binomial distribution.

First, we are going to fit a logistic model using all of the variables that we have not removed from the data set. We use the training set to create the model.

```
logmod <- glm(price ~ ., data = train, family = "binomial")
msummary(logmod)
```

```
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.80e+02   2.89e+01  -13.13  < 2e-16 ***
## bedrooms      -3.58e-02   3.33e-02   -1.08     0.28
## bathrooms      6.93e-01   5.42e-02   12.79  < 2e-16 ***
## sqft_lot       4.62e-06   1.09e-06    4.24  2.2e-05 ***
## floors         3.66e-01   5.80e-02    6.30  2.9e-10 ***
## waterfront1    2.12e+00   5.26e-01    4.04  5.3e-05 ***
## view           3.76e-01   4.41e-02    8.53  < 2e-16 ***
## condition      4.03e-01   4.03e-02   10.01  < 2e-16 ***
## grade          1.44e+00   4.47e-02   32.14  < 2e-16 ***
## sqft_above     5.28e-04   6.29e-05    8.40  < 2e-16 ***
## yr_built      -3.62e-02   1.29e-03  -28.19  < 2e-16 ***
## lat            9.03e+00   2.24e-01   40.28  < 2e-16 ***
## long          -2.96e-02   2.15e-01   -0.14     0.89
## sqft_living15  1.07e-03   6.55e-05   16.34  < 2e-16 ***
## sqft_lot15    -2.20e-07   1.55e-06   -0.14     0.89
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22024  on 16196  degrees of freedom
## Residual deviance: 11044  on 16182  degrees of freedom
## AIC: 11074
##
## Number of Fisher Scoring iterations: 6
```

It appears from the model that we have 9 significant predictors and we can perform a likelihood ratio test to determine if the overall model is significant.

```
lrtest(logmod)
```

```
## Likelihood ratio test
##
## Model 1: price ~ bedrooms + bathrooms + sqft_lot + floors + waterfront +
##     view + condition + grade + sqft_above + yr_built + lat +
##     long + sqft_living15 + sqft_lot15
## Model 2: price ~ 1
##   #Df LogLik  Df Chisq Pr(>Chisq)
## 1  15  -5522
## 2   1 -11012 -14 10980     <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since we have a significant p-value, it appears that the overall model is significant. Now in order to make predictions using this model, we need to bin the fitted values which we will do below.

```
augprice1 <- augment(logmod, type.predict = "response")
augprice1 <- mutate(augprice1, binprediction = round(.fitted, 0))
tally(~ binprediction, data = augprice1)
```

```
## binprediction
##    0    1
## 9826 6371
```

```
9826/(9826+6371)
```

```
## [1] 0.606656
```

After using 0.5 as a default cutoff, the model classifies 60.7% of the prices as below $500,000. This is close to the actual percentage of the training data that is classified as below $500,000 which is 58.1% of the observations. We can try to match this a little better by adjusting the cutoff to which we round the numbers. Finding, the appropriate quantile, we now see that the percentage of the binned fitted values for prices below $500,000 is 58.4% which is much closer than before.

```
# find out the actual percentage in training set
tally(~ price, data = augprice1)
```

```
## price
##    0    1
## 9415 6782
```

```
9415/(9415+6782)
```

```
## [1] 0.58128
```

```
# find the appropriate quantile
with(augprice1, quantile(.fitted, 0.58128))
```

```
##  58.128%
## 0.450849
```

```
# re-bin the values and find out the percentage
augprice1 <- mutate(augprice1, binprediction2 = as.numeric(.fitted > 0.454465))
tally(~ binprediction2, data = augprice1)
```

```
## binprediction2
##    0    1
## 9455 6742
```

```
9455/(9455+6742)
```

```
## [1] 0.58375
```

Next, we can create a confusion matrix and determine how many of the predictions were correct. From the matrix and calculation below, it appears that 84.0% of the predictions for the training set were correct.

```
with(augprice1, table(price, binprediction2))
```

```
##      binprediction2
## price    0    1
##     0 8138 1277
##     1 1317 5465
```

```
(8138+5465)/(8138+5465+1277+1317)
```

```
## [1] 0.839847
```

We will do the same on the test set to see how the model performs. We follow the same process as above to calculate the predicted values and then bin them using our previous cutoff. The model classifies 58.9% of the prices below $500,000.

```
augprice2 <- mutate(test, fitted = predict(logmod, newdata = test, type = "response"))
augprice2 <- mutate(augprice2, binprediction = as.numeric(fitted > 0.454465))
tally(~ binprediction, data = augprice2)
```

```
## binprediction
##    0    1
## 3181 2218
```

```
(3181)/(3181+2218)
```

```
## [1] 0.589183
```

Next, we generate a confusion matrix and it appears that 83.8% of the predictions were correct.

```
with(augprice2, table(price, binprediction))
```

```
##      binprediction
## price    0    1
##     0 2719  412
##     1  462 1806
```

```
(2719+1806)/(2719+1806+462+412)
```

```
## [1] 0.838118
```

```
# dDELETE I THINK
# get MSE of training set
augprice1 <- augment(logmod, type.predict = "response")
mean((augprice1$.fitted - augprice1$price)^2)
```

```
## [1] 0.10984
```

```
#get MSE of test set
test4 <- mutate(test, fitted = predict(logmod, newdata = test, type = "response"))
test4 <-filter(test4, fitted != "NA")
mean((test4$fitted - test4$price)^2)
```

```
## [1] 0.110467
```

Next, we will try creating a smaller model by removing some of the predictors that were not significant in our previous model. We are going to remove `bedrooms`, `long`, and `sqft_lot15`. After creating this model, we then repeated the process and removed other variables that were significant, but did not seem to have much of an impact on the predictions because the other variables were much more influential. The other variables we removed were `sqft_lot`, `floors`, `waterfront`, `view`, `condition`, and `sqft_above`.

```
logmod2 <- glm(price ~ bathrooms + grade + yr_built + lat + sqft_living15,
  data = train, family = "binomial")
msummary(logmod2)
```

```
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.31e+02   1.01e+01   -32.8   <2e-16 ***
## bathrooms      9.04e-01   4.69e-02    19.3   <2e-16 ***
## grade          1.58e+00   4.19e-02    37.8   <2e-16 ***
## yr_built      -3.81e-02   1.08e-03   -35.2   <2e-16 ***
## lat            8.17e+00   2.09e-01    39.0   <2e-16 ***
## sqft_living15  1.30e-03   5.46e-05    23.8   <2e-16 ***
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22024  on 16196  degrees of freedom
## Residual deviance: 11470  on 16191  degrees of freedom
## AIC: 11482
##
## Number of Fisher Scoring iterations: 6
```

```
lrtest(logmod2)
```

```
## Likelihood ratio test
##
## Model 1: price ~ bathrooms + grade + yr_built + lat + sqft_living15
## Model 2: price ~ 1
##   #Df LogLik Df Chisq Pr(>Chisq)
## 1   6  -5735
## 2   1 -11012 -5 10554     <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

All of the variables in our reduced model are significant and the model overall is significant as well. Now, we can create some predictions following the same process as above.

```
augprice3 <- augment(logmod2, type.predict = "response")
augprice3 <- mutate(augprice3, binprediction = round(.fitted, 0))
tally(~ binprediction, data = augprice3)
```

```
## binprediction
##    0    1
## 9829 6368
```

```
9829/(9829+6386)
```

```
## [1] 0.606167
```

After using 0.5 as a default cutoff, the model classifies 60.6% of the prices as below $500,000. After adjusting the quantile, we now see that the percentage of the binned fitted values for prices below $500,000 is 58.1% which is much closer than before.

```
# find the appropriate quantile
with(augprice3, quantile(.fitted, 0.58128))
```

```
##  58.128%
## 0.451563
```

```
# re-bin the values and find out the percentage
augprice3 <- mutate(augprice3, binprediction2 = as.numeric(.fitted > 0.451563))
tally(~ binprediction2, data = augprice3)
```

```
## binprediction2
##    0    1
## 9414 6783
```

```
9414/(9414+6783)
```

```
## [1] 0.581219
```

Next, we can create a confusion matrix and determine how many of the predictions were correct. From the matrix and calculation below, it appears that 84.0% of the predictions for the training set were correct.

```
with(augprice3, table(price, binprediction2))
```

```
##      binprediction2
## price    0    1
##     0 8064 1351
##     1 1350 5432
```

```
(8119+5486)/(8119+5486+1296+1296)
```

```
## [1] 0.83997
```

We will do the same on the test set to see how the model performs. We follow the same process as above to calculate the predicted values and then bin them using our previous cutoff. The model classifies 58.6% of the prices below $500,000.

```
augprice4 <- mutate(test, fitted = predict(logmod2, newdata = test, type = "response"))
augprice4 <- mutate(augprice4, binprediction3 = as.numeric(fitted > 0.451563))
tally(~ binprediction3, data = augprice4)
```

```
## binprediction3
##    0    1
## 3164 2235
```

```
(3164)/(3164+2235)
```

```
## [1] 0.586034
```

Next, we generate a confusion matrix and it appears that 83.1% of the predictions were correct.

```
with(augprice4, table(price, binprediction3))
```

```
##      binprediction3
## price    0    1
##     0 2690  441
##     1  474 1794
```

```
(2690+1794)/(2690+1794+474+441)
```

```
## [1] 0.830524
```

Overall, from the two logistic regression models that were created, they both performed very similar to each other with the second one having a slightly lower correct prediction rate. However, for the second model, we removed many predictors that were not influential to the model and therefore were not helping make the predictions. After those variables were removed, the logistic model became much more simple with less predictors. The real estate developer's questions of interest is to find the variables that are indicative of houses that sell for over $500,000 and by keeping a similar performance of the model, but removing extraneous predictors, we are able to find the predictors that are the most important. The second model contained the predictors: `bathrooms`, `grade`, `yr_built`, `lat`, and `sqft_living15`.

Tree Fitting

```
# choose and fit at least 2 different models for your tree
# include appropriate output and assessment of their performance

# think ahead: what does a reader need to know to follow your model fitting process?
# you may want to jot down notes for yourself that you'll need later!
```

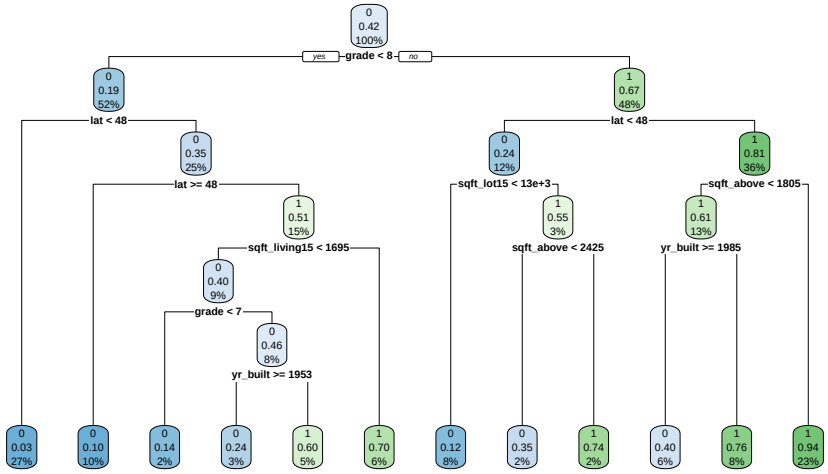
The second method that we are going to be using to fit models is through classification trees. Classification trees are a nonparametric method of prediction and are easy to interpret. Classification trees are used when the response variable is binary so it is appropriate to use them in order to predict if the price is over or under $500,000. They are created through recursive partitioning to create "splits" between observations with similar qualities.

We will first create a tree with the default stopping criteria.

```
price.rpart1 <- rpart(price ~ ., data = train, method = "class")
printcp(price.rpart1)
```

```
##
## Classification tree:
## rpart(formula = price ~ ., data = train, method = "class")
##
## Variables actually used in tree construction:
## [1] grade         lat           sqft_above    sqft_living15 sqft_lot15
## [6] yr_built
##
## Root node error: 6782/16197 = 0.4187
##
## n= 16197
##
##         CP nsplit rel error xerror     xstd
## 1 0.39266      0    1.0000 1.0000 0.009258
## 2 0.14612      1    0.6073 0.6073 0.008172
## 3 0.01863      2    0.4612 0.4636 0.007422
## 4 0.01371      5    0.4053 0.3947 0.006970
## 5 0.01054      7    0.3779 0.3803 0.006866
## 6 0.01003      9    0.3568 0.3747 0.006825
## 7 0.01000     11    0.3368 0.3689 0.006782
```

```
rpart.plot(price.rpart1)
```



The classification tree above uses the predictors, `grade`, `lat`, `sqft_above`, `sqft_living15`, `sqft_lot15`, and `yr_built`. The error from this classification tree is 0.8589 which is calculated from the root node error and the relative error in the output above.

```
1 - 0.3368*0.4187
```

```
## [1] 0.858982
```

Next, we will using the classification tree that we created to see how well it predicts the training set. From the confusion matrix below, it appears that the classification tree predicts correctly for 85.9% of the observations.

```
train1 <- mutate(train, predprice = predict(price.rpart1, type = "class"))
tally(predprice ~ price, data = train1)
```

```
##          price
## predprice    0    1
##         0 8227 1096
##         1 1188 5686
```

```
(8227+5696)/(8227+5696+1188+1096)
```

```
## [1] 0.859073
```

And doing the same thing for the test we, it appears that the model predicts correctly 85.5% of the observations.

```
test1 <- mutate(test, predprice = predict(price.rpart1, type = "class", newdata = test))
tally(predprice ~ price, data = test1)
```

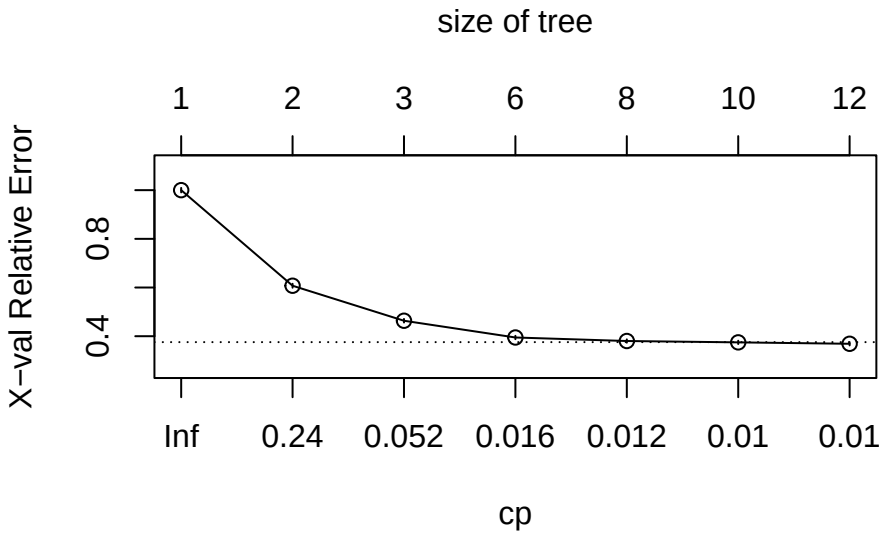
```
##          price
## predprice    0    1
##         0 2748  398
##         1  383 1870
```

```
(2748+1870)/(2748+1870+398+383)
```

```
## [1] 0.855344
```

Before we create the second model, we examine the cross -validation process that was used in creating the previous model. The default cross-validation is a 10-fold cross-validation. Below we can see that as the complexity parameter reaches 0.16, the relative error calculated from the cross-validation decreases only a small amount as the complexity parameter increases. Therefore, we are going to use a complexity parameter of 0.016 for our next tree.

```
plotcp(price.rpart1)
```



We are also going to control `minsplit = 500` which means that there needs to be 500 observations in a node before a split is attempted. This is reasonable for a tree created from our training data with a size of 16197. We also set `minbucket = 300` which means that the minimum number of observations in any terminal node is 300 observations. There controls are to try and attempt to build a smaller tree while keeping the same effectiveness of the first model.

```
price.control <- rpart.control(minsplit = 500, minbucket = 300, cp = 0.016)
price.rpart2 <- rpart(price ~ ., data = train, method = "class", control = price.control)
printcp(price.rpart2)
```

```
##
## Classification tree:
## rpart(formula = price ~ ., data = train, method = "class", control = price.control)
##
## Variables actually used in tree construction:
## [1] grade         lat           sqft_living15
##
## Root node error: 6782/16197 = 0.4187
##
## n= 16197
##
```

```
##        CP nsplit rel error xerror     xstd
## 1 0.39266      0    1.0000 1.0000 0.009258
## 2 0.14612      1    0.6073 0.6073 0.008172
## 3 0.01863      2    0.4612 0.4643 0.007426
## 4 0.01600      5    0.4053 0.4039 0.007034
```

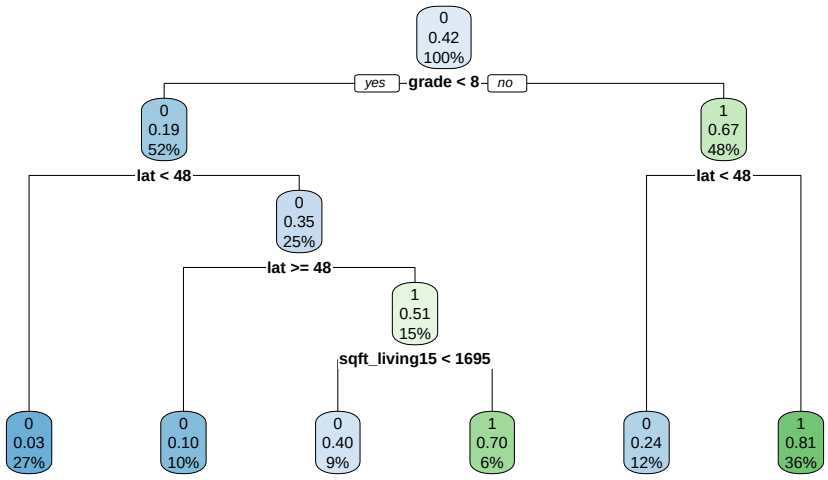
The classification tree above uses the predictors, `grade`, `lat`, and `sqft_living15`. The error from this classification tree is 0.8303 which is calculated from the root node error and the relative error in the output above.

```
1 - 0.4053*0.4187
```

```
## [1] 0.830301
```

From the output below, the classification tree we created is much more simple.

```
rpart.plot(price.rpart2)
```



We will now see how the model performs in classifying observations.

```
train2 <- mutate(train, predprice = predict(price.rpart2, type = "class"))
tally(predprice ~ price, data = train2)
```

```
##          price
## predprice    0    1
##         0 8054 1388
##         1 1361 5394
```

```
(8054+5394)/(8054+5394+1361+1388)
```

```
## [1] 0.830277
```

It appears the the model classifies observations 83.0% correctly in the training set. We will do the same for the test set.

```
test2 <- mutate(test, predprice = predict(price.rpart2, type = "class", newdata = test))
tally(predprice ~ price, data = test2)
```

```
##          price
## predprice    0    1
##         0 2710  496
##         1  421 1772
```

```
(2710+1772)/(2710+1772+421+496)
```

```
## [1] 0.830154
```

It appears the the model classifies observations 83.0% correctly in the test set as well.

From these two models, the first one is more complex and predicts correctly about 2.5-3% better than the second model. However, this means that the first classification tree is less interpretable than the second one. The first tree has 11 splits whereas the second one only has 5. Because of this reason, we would choose the second model over the first one. The slight decrease in correct predictions matters less than the increase in interpretability for the model. Additionally, the error of the second classification tree is slightly less than that of the first tree. Also, since we are determining which of the variables are the best predictors if a house is above $500,000, the second model only uses 3 predictors whereas the first model uses 6. The predictors used in the second model are `grade`, `lat`, and `sqft_living15`.

Thinking about Homework 5

The eventual Homework 5 submission will include the following sections (along with all code necessary to reproduce your results):

- Introduction and Exploratory Data Analysis - could be two separate sections
- Your GLM and relevant details
- Your Tree and relevant details (can be before or after your GLM)
- Your Model Comparison (can be woven in sections above)
- Your Conclusion

Descriptions of the purpose for each section follow. The idea here is to explain why you'd write each section, and you can work out what needs to be in each in order to fulfill that purpose.

- Introduction - The real estate developer has interests, but will you be able to address them? How do you understand the tasks you are presented with? It's important to state what you will be doing in the analysis, so the reader can make sure their understanding lines up with what you will be doing. The reader also needs an introduction to your data, either in this section, or below. They need enough detail to be able to determine if your actions later in the analysis are reasonable.
- Exploratory Data Analysis - You are the analyst here. That means you can use variable re-expressions, subsets of observations, and employ other analytic practices (e.g. training/testing data sets) at your discretion to aid in your analysis, so long as you explain your rationale for them. The reader needs to know what you found and what you decided to do about it, because this has impacts on the rest of your analysis. Remember our lessons from the first classes - be sure you look at the data! Here's your chance to share what you found based on how it impacts your analysis.
- GLM and Tree sections - These are new techniques. By having each in their own section, you can focus on your explanation for them one at a time. The reader doesn't know when to use these methods or what they do. You have options that you need to pick for the different techniques (various tuning parameters) - for example, are you using a classification or a regression tree? What tree stopping options are being used? What input variables are you using and why? What type of GLM are you using? How will you be measuring performance? Be sure you discuss what options you are selecting and what made you choose those values. Helping explain this shows your understanding of the techniques (remember the class example that minbucket of 30 was nonsensical for the iris data).

You also want these sections to show off your ability to build models. It is not appropriate to just fit kitchen sink models (as your only model) or accept all default settings without exploring to see if that is really what is best for the task at hand. Your write-up should make it clear what you explored, but you don't have to show every model you considered.

Finally, remember to check out your "best" model for each technique. Did you check reasonable diagnostics? Does the model make sense? Are variables behaving as you expect? If you focus solely on model performance, that's missing the point. For example, you might find a model has very little error because a variant of the response was accidentally included as a predictor. You are responsible for checking over the model before reporting it. Your write-up should convey that you've done this (the reader can't read your mind to know you did it).

- Model Comparison - There are several ways to compare the models - performance, interpretability, variables involved, etc. Remember you are trying to address the real estate developer's questions of interest, so you probably need to focus on just one model, or maybe you found a way to combine results from a "best" GLM and a "best" tree. The idea for this section is that you need to convey the process used to pick a final model(s) to use to answer those questions, with justification for your choices.

- Conclusion - This section should be a stand alone summary of your analysis. It is where you get to state your final model and a quick summary of the process used to get there. You also need to address the developer's questions, and this is your last place to do that! Remember to flush out the details here. If your final sentence is "Model 5 is the model I choose and it answers your questions", does it? Are you doing your job as the analyst to leave things at that? For that matter, what is Model 5?
- Printing Trees - It is fine to print your trees out to separate .pdfs (just be sure to include them in your submission!). Remember if you want to include the .pdf as an image, you have example code for that. Please do show your code though (so no echo = FALSE), and remember that your work should be reproducible.
- Audience - For purposes of this submission, remember that the audience is the real estate developer, so you'll probably need to include some explanations that you'd leave out of a normal homework. For example, the developer isn't going to know what a GLM is, or what type you are using, or why you'd use it. You should think about where that information should go, and do your best to explain what you need in order to report your findings. Similarly, assume that you found this data to assist the developer and they need basic details about it to understand the variables. They didn't hand it over to you.