Homework 4 - Stat 495

Cassandra Jin

Due Wednesday, Oct. 11th 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, using those methods.

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 (or I tell you something has to change) 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)
kchouse <- select(kchouse, -id, -date) # remove id, date variables
```

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 in this area of Washington, and has turned to you, a statistical consultant for help. The developer 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 decision 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:

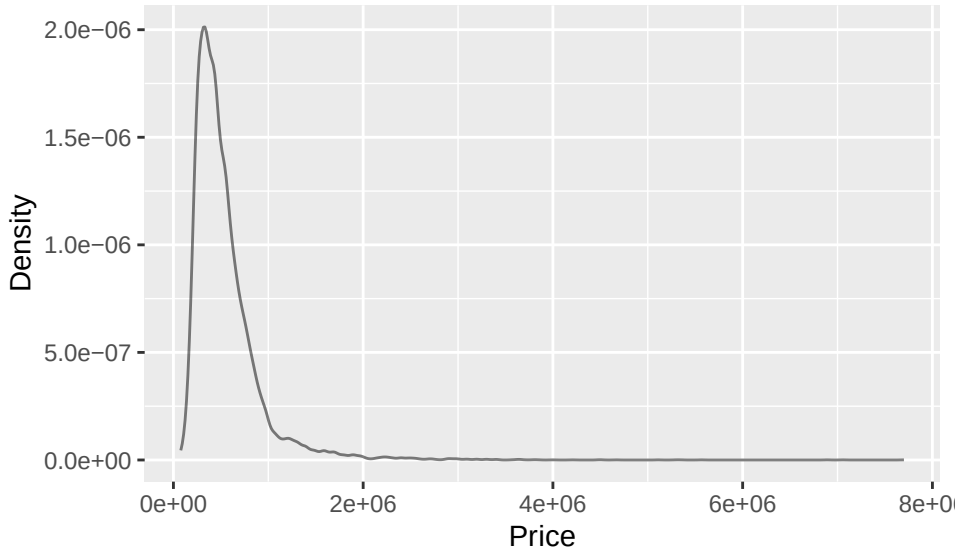
- Exploratory Analysis
- GLM - at least 2 appropriate models fit with output and assessment
- Tree - at least 2 appropriate models fit with output and assessment

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

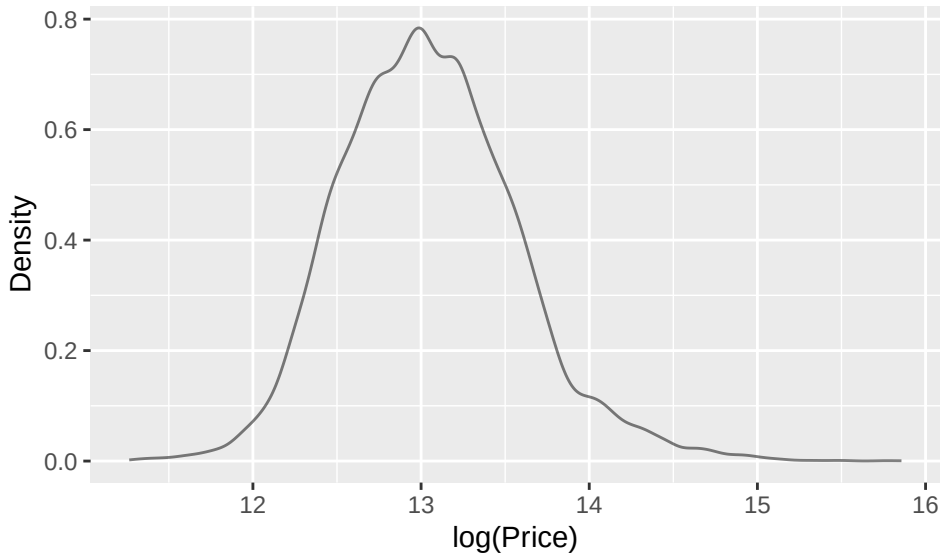
Be sure you understand how to read output for both GLMs and trees, as this is material covered on the midterm.

Exploratory Analysis

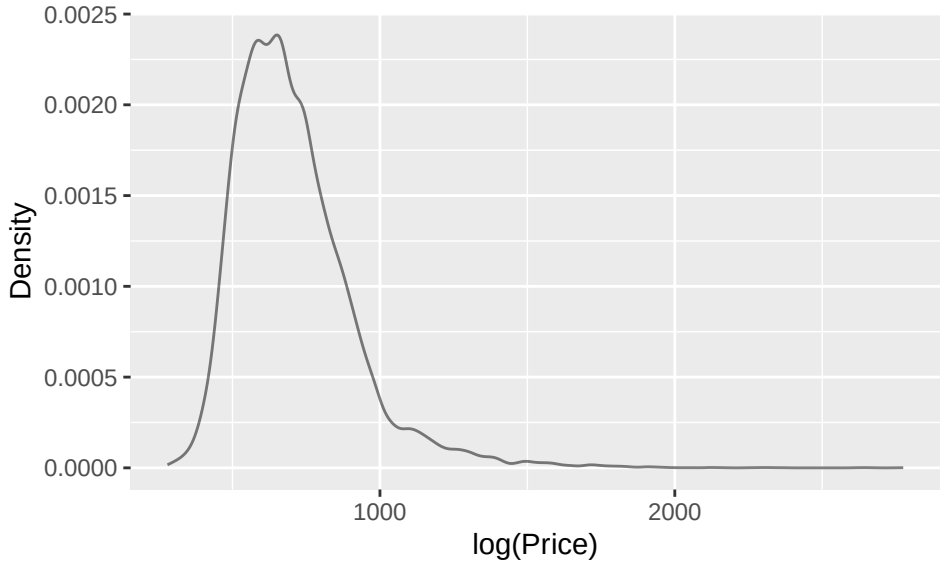
```
# explore the data
# you need to do this before fitting anything!

# submission should demonstrate you explored the data somewhat before
# fitting the models below

# If there is anything else you should do before fitting models,
# do it here

# original density distribution of response, price
gf_dens(~ price, data = kchouse,
              ylab = "Density",
              xlab = "Price")
```




```
kchouse <- mutate(kchouse, log_price = log(price)) # log transform price
kchouse <- mutate(kchouse, sqrt_price = sqrt(price)) # square-root transform price

gf_dens(~ log_price, data = kchouse,
              ylab = "Density",
              xlab = "log(Price)")
```




```
gf_dens(~ sqrt_price, data = kchouse,
              ylab = "Density",
              xlab = "log(Price)")
```



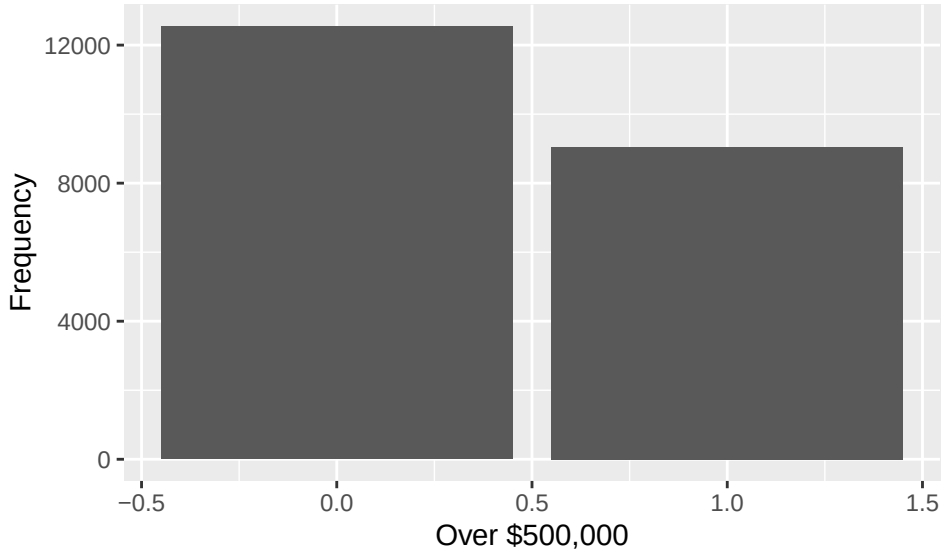

```
favstats(~ log_price, data = kchouse)
```

```
##      min      Q1 median     Q3     max    mean       sd     n missing
##  11.2645 12.6823 13.017 13.377 15.8567 13.0482 0.526555 21597       0
```

Since Price is very right-skewed, we perform a log-transform to achieve a more normally distributed outcome variable. (realized afterwards that a quantitative outcome is not what we're looking for)

```
# make binary variable for Price > $500,000
# make quantitative variables -> binary, since most observations are 0
kchouse <- kchouse %>%
  mutate(overhalfmil = ifelse(price > 500000, 1, 0), # 1=yes, 0=no
         basement = ifelse(sqft_basement > 0, 1, 0),
         renovated = ifelse(yr_renovated > 0, 1, 0),
         log_sqft_living = log(sqft_living),
```

```
         log_sqft_lot = log(sqft_lot),
         log_sqft_above = log(sqft_above),
         log_sqft_living15 = log(sqft_living15),
         log_sqft_lot15 = log(sqft_lot15))

# bar graph and tally for Price > 500000 frequency
gf_bar(~ overhalfmil, data = kchouse,
       ylab = "Frequency",
       xlab = "Over $500,000")
```

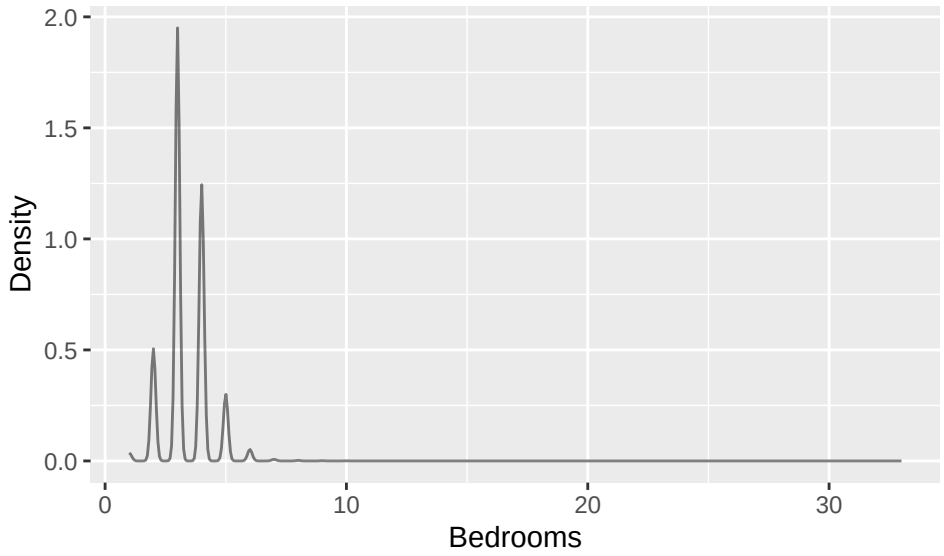



```
tally(~ overhalfmil, format = "percent", data = kchouse)
```

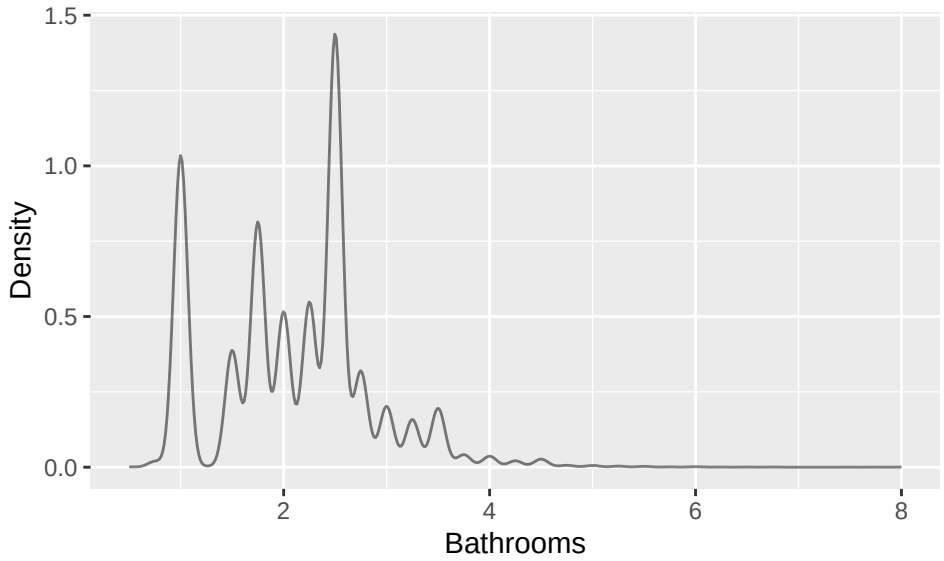
```
## overhalfmil
##       0       1
## 58.0914 41.9086
```

```
kchouse <- select(kchouse, -price, -sqft_living, -sqft_lot, -sqft_above, -sqft_living15, -sqft_lot15, -s
```

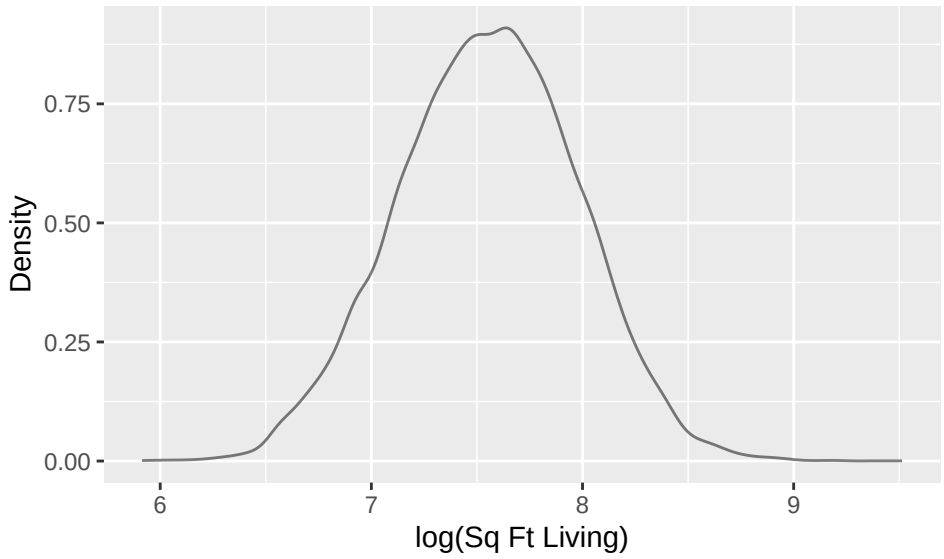
```
# density plots for quantitative variables
gf_dens(~ bedrooms, data = kchouse,
              ylab = "Density",
              xlab = "Bedrooms")
```



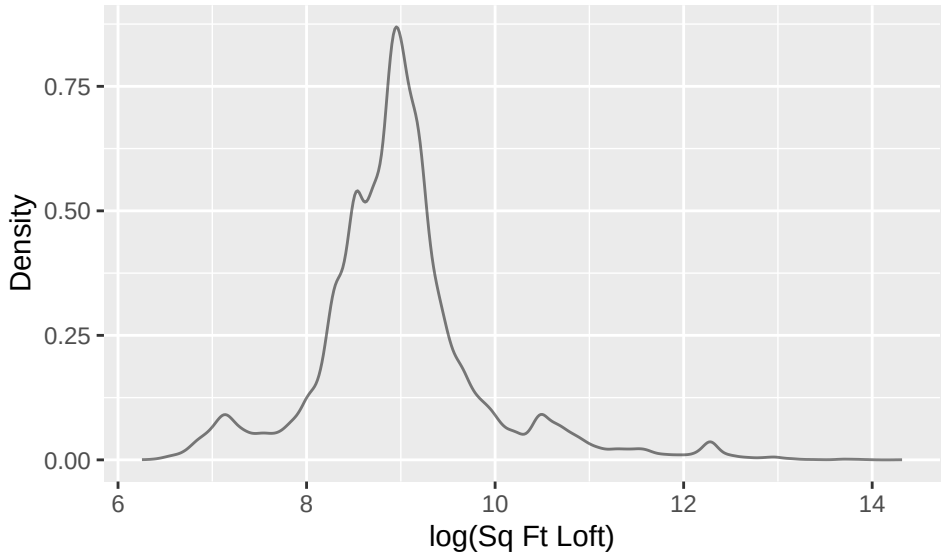

```
gf_dens(~ bathrooms, data = kchouse,
              ylab = "Density",
              xlab = "Bathrooms")
```



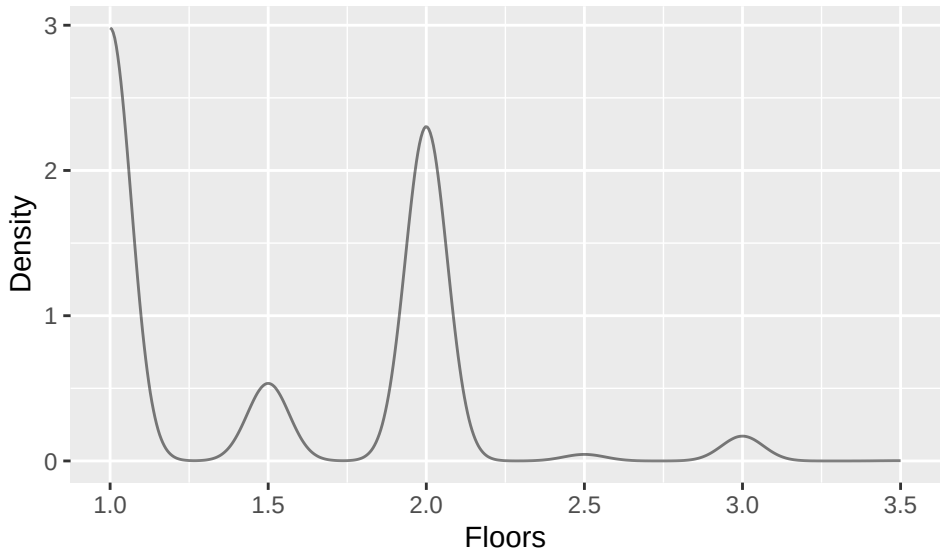

```
gf_dens(~ log_sqft_living, data = kchouse,
              ylab = "Density",
              xlab = "log(Sq Ft Living)")
```



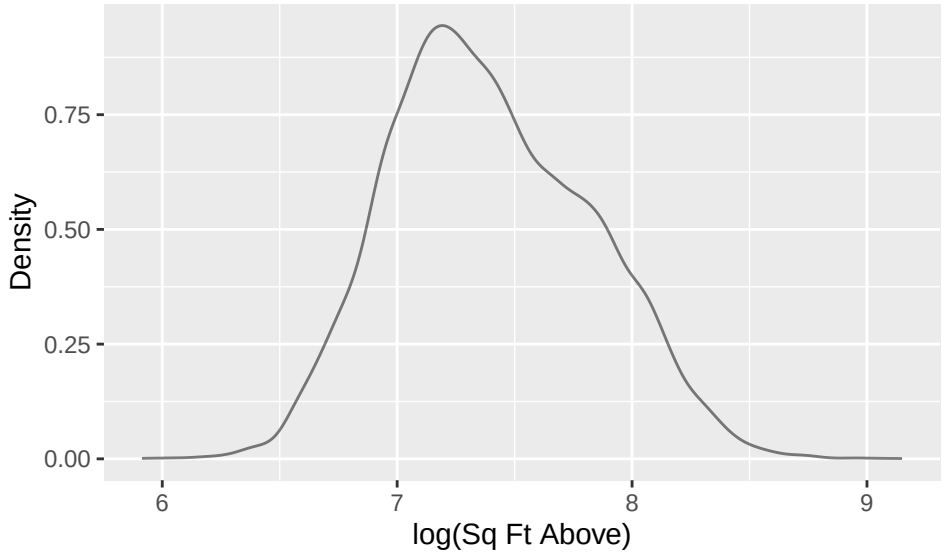

```
gf_dens(~ log_sqft_lot, data = kchouse,
              ylab = "Density",
              xlab = "log(Sq Ft Loft)")
```



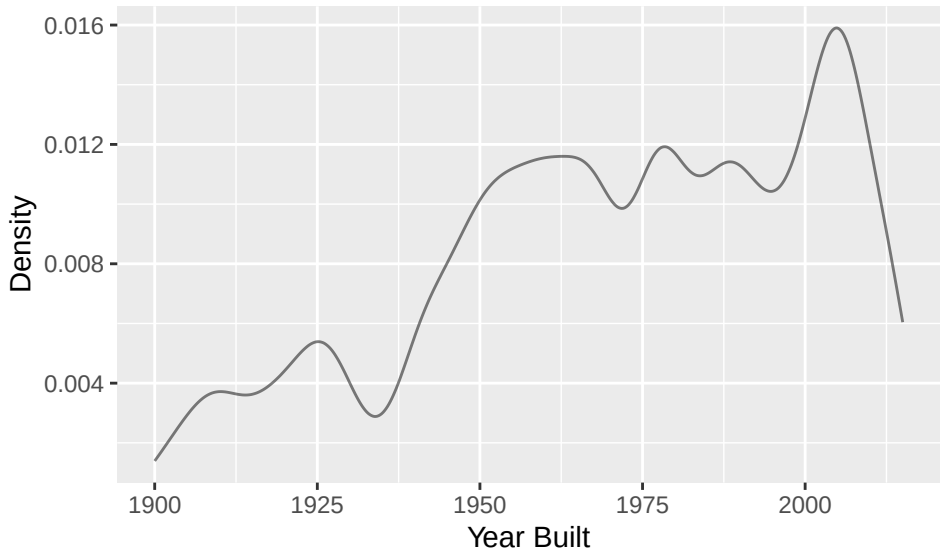

```
gf_dens(~ floors, data = kchouse,
              ylab = "Density",
              xlab = "Floors")
```



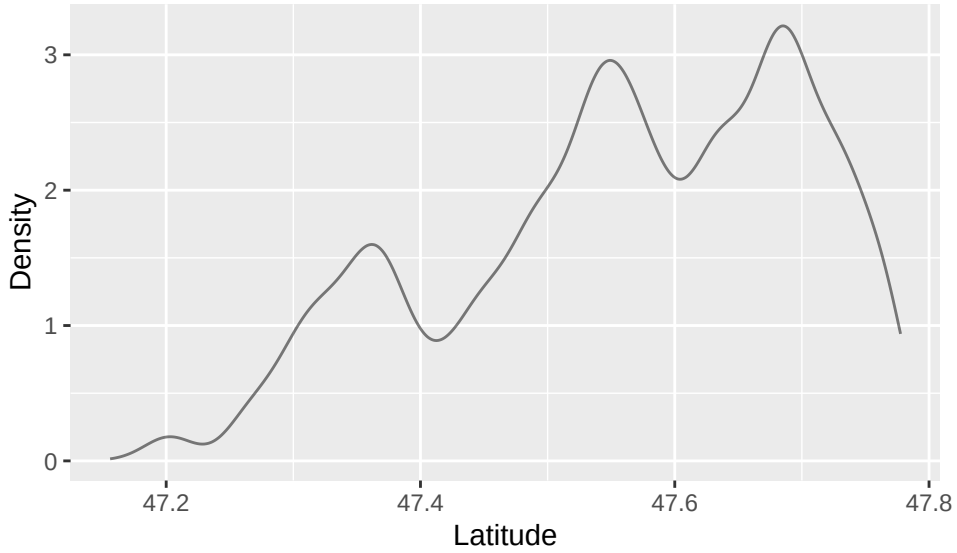

```
gf_dens(~ log_sqft_above, data = kchouse,
              ylab = "Density",
              xlab = "log(Sq Ft Above)")
```



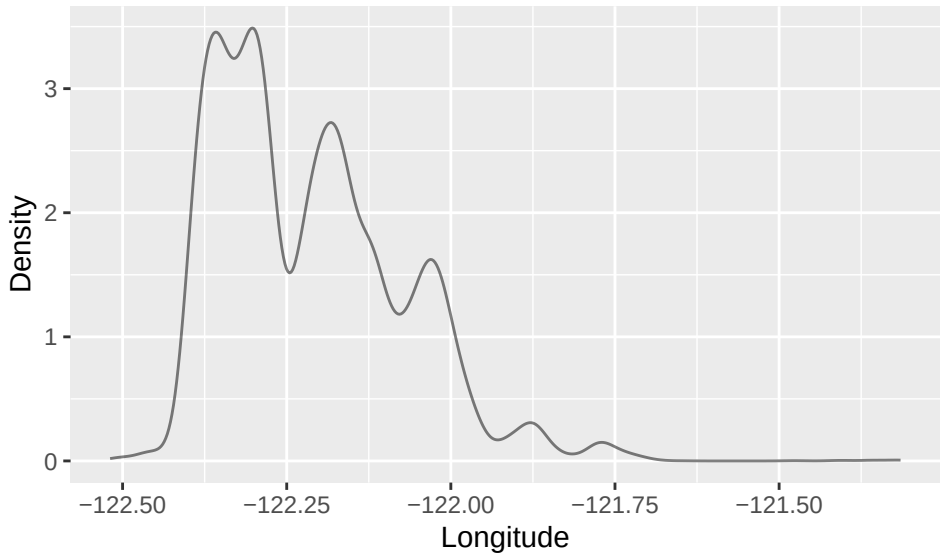

```
gf_dens(~ yr_built, data = kchouse,
              ylab = "Density",
              xlab = "Year Built")
```



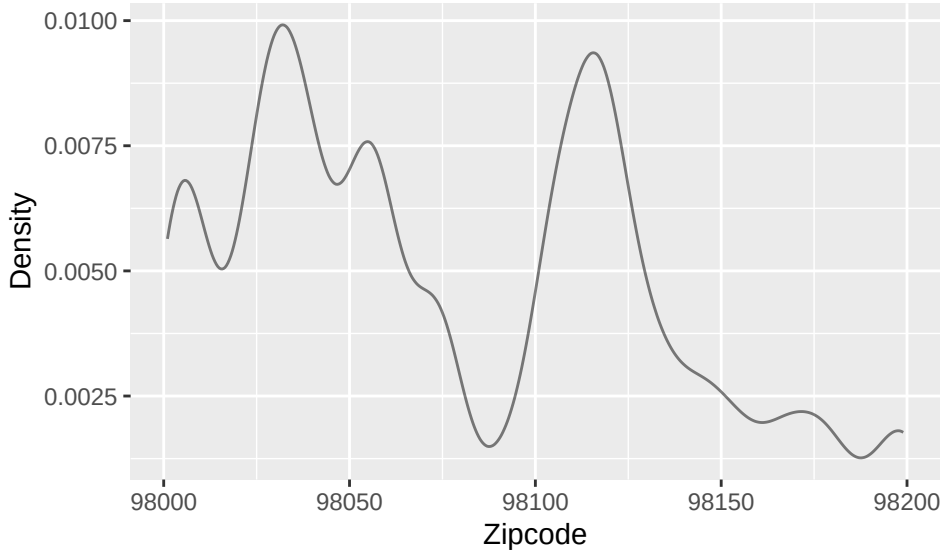

```
gf_dens(~ lat, data = kchouse,
              ylab = "Density",
              xlab = "Latitude")
```



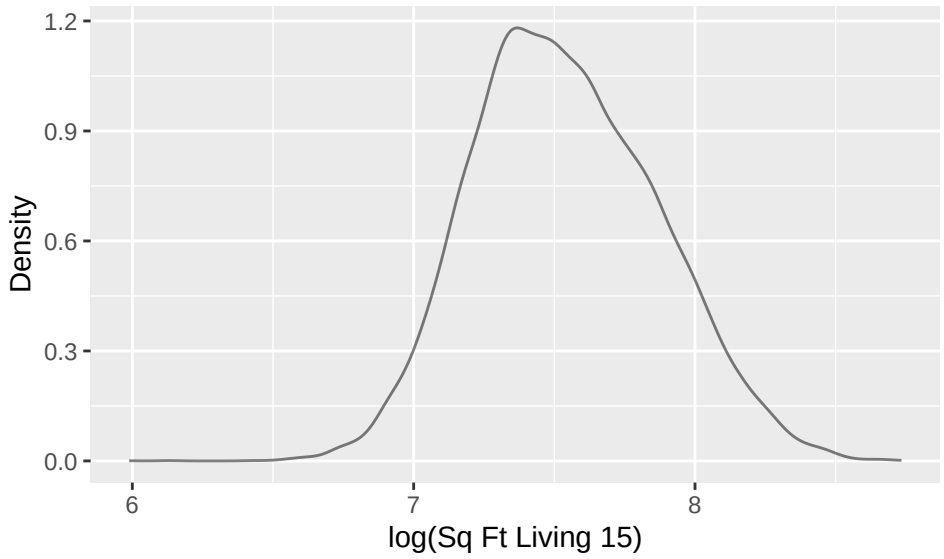

```
gf_dens(~ long, data = kchouse,
              ylab = "Density",
              xlab = "Longitude")
```



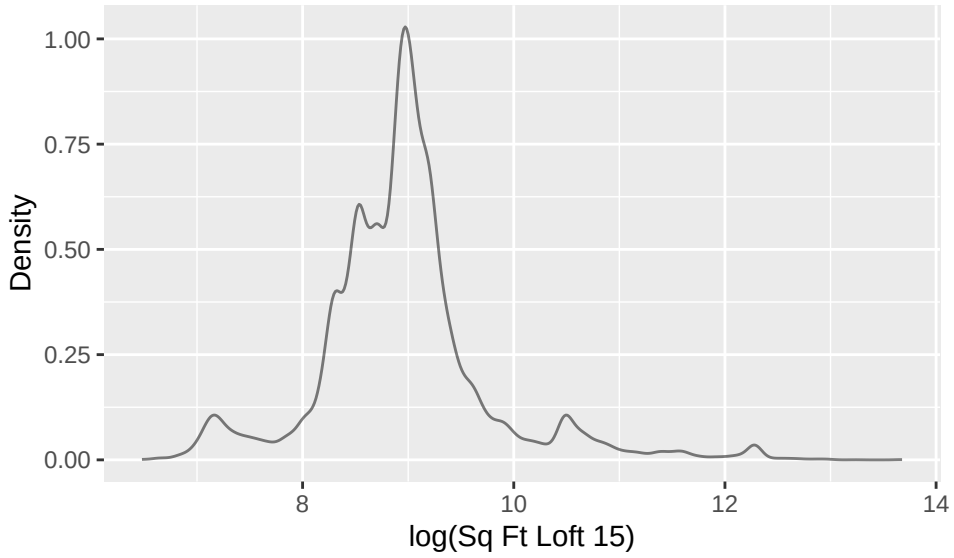

```
gf_dens(~ zipcode, data = kchouse,
              ylab = "Density",
              xlab = "Zipcode")
```



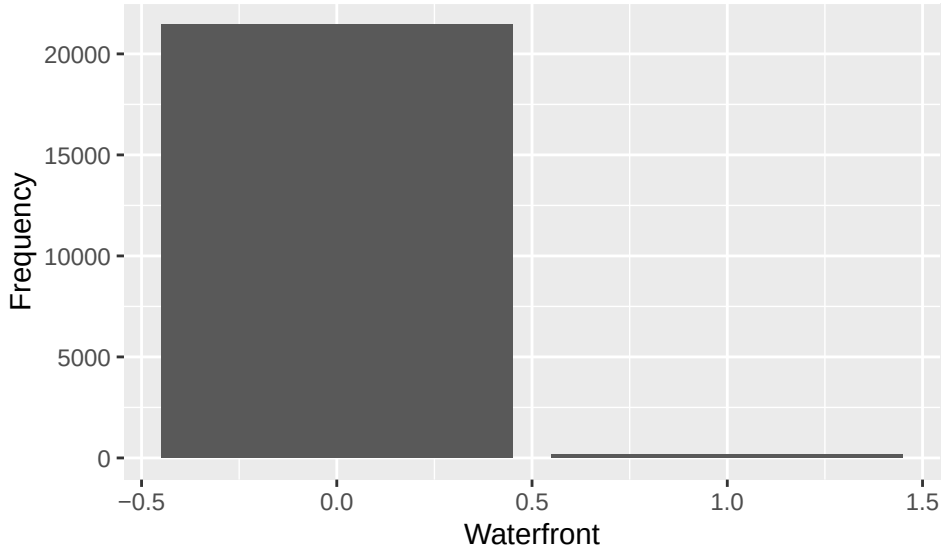

```
gf_dens(~ log_sqft_living15, data = kchouse,
              ylab = "Density",
              xlab = "log(Sq Ft Living 15)")
```



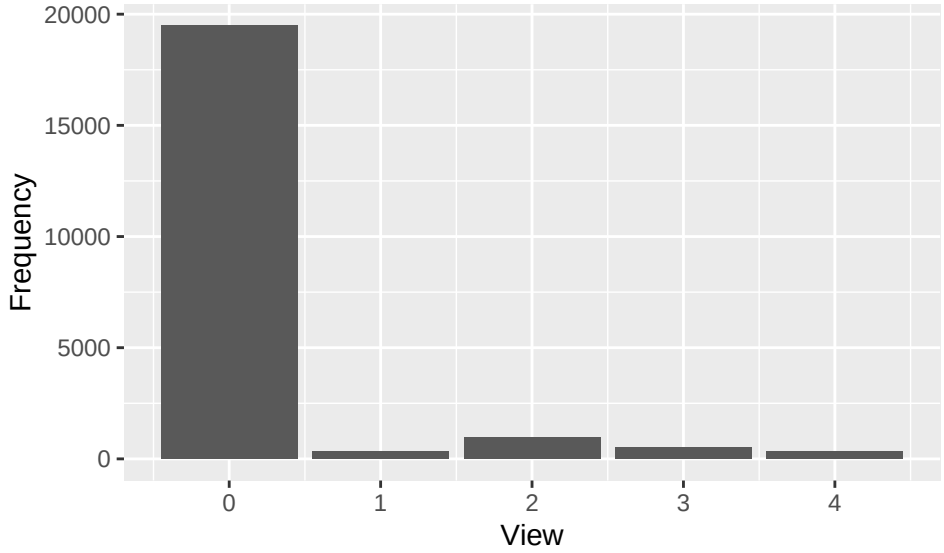

```
gf_dens(~ log_sqft_lot15, data = kchouse,
              ylab = "Density",
              xlab = "log(Sq Ft Loft 15)")
```



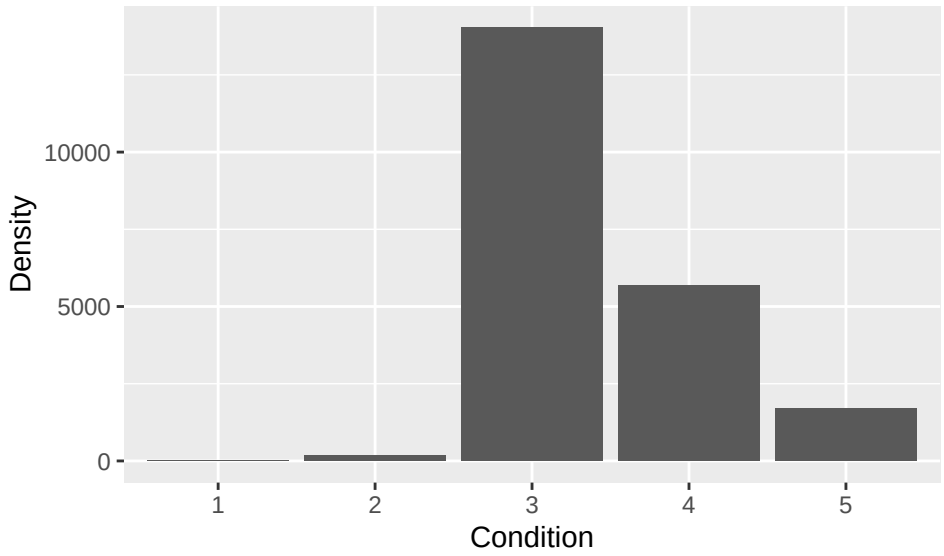

```
# bar graphs for qualitative variables
gf_bar(~ waterfront, data = kchouse,
             ylab = "Frequency",
             xlab = "Waterfront")
```



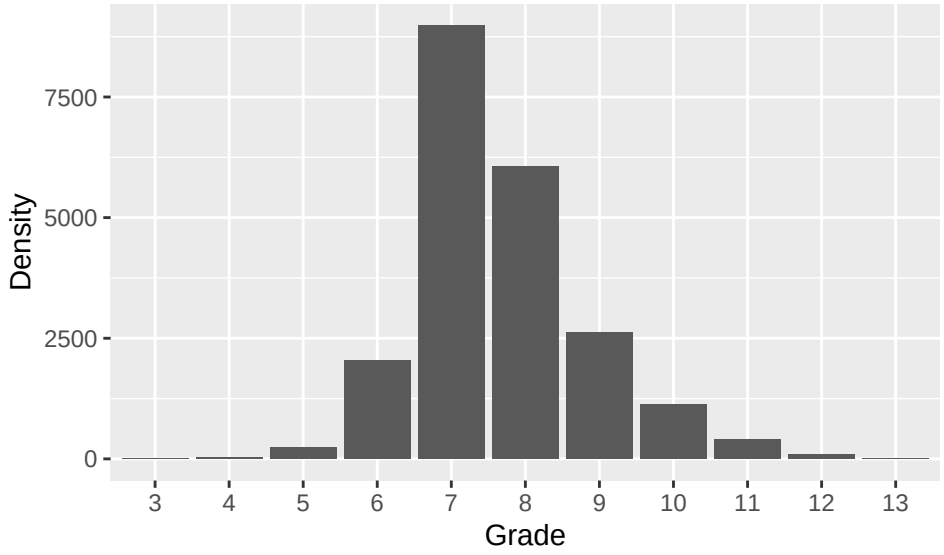

```
gf_bar(~ view, data = kchouse,
             ylab = "Frequency",
             xlab = "View")
```



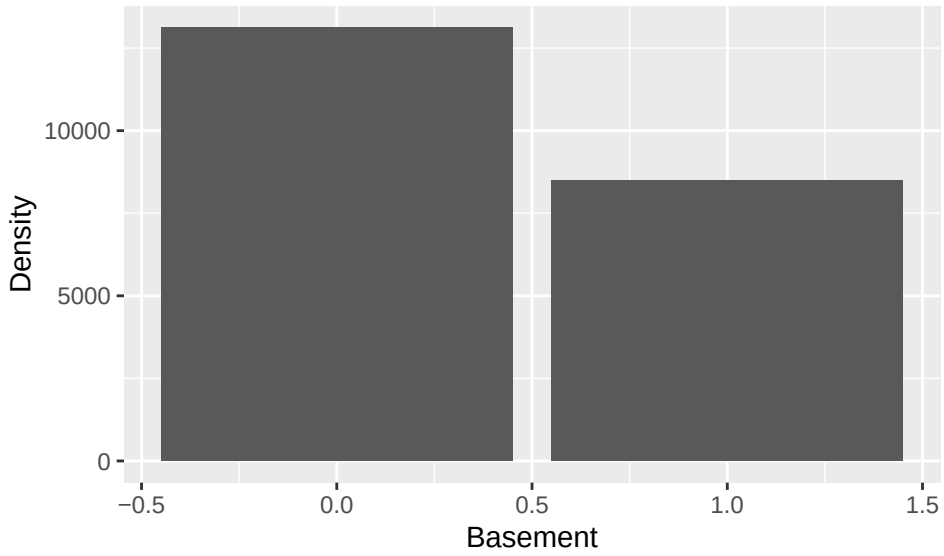

```
gf_bar(~ as.factor(condition), data = kchouse,
              ylab = "Density",
              xlab = "Condition")
```



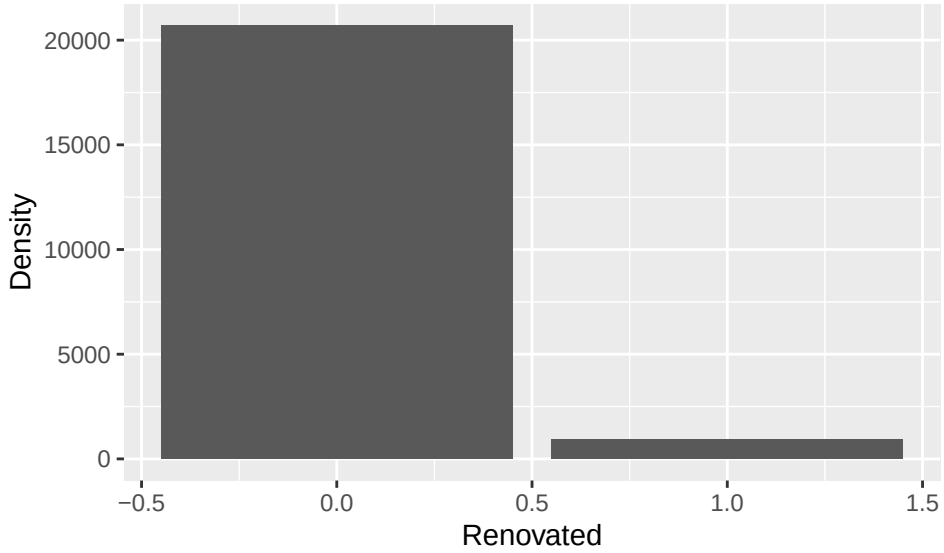

```
gf_bar(~ as.factor(grade), data = kchouse,
               ylab = "Density",
               xlab = "Grade")
```




```
gf_bar(~ basement, data = kchouse,
               ylab = "Density",
               xlab = "Basement")
```




```
gf_bar(~ renovated, data = kchouse,
               ylab = "Density",
               xlab = "Renovated")
```



Applying the log transform to predictor variables does make the model slightly more complicated to interpret, but for the sake of predictability, I'm choosing to push the variables to a normal distribution with more convenient spread. Although the the variable `sqft_basement` is left-skewed, we cannot apply the log transform because the values are 0 for some observations.

For EDA, we consider the categorical variables with the as.factor() function to see how observations are distributed across levels. However, when fitting the model, I will treat these variables (view, condition, grade) as numerical ones, since they are progressive in a way and thus can be viewed quantitatively. I will also keep waterfront quantitative to let the model take in 0 or 1 and have a corresponding coefficient that covers the effect of this variable.

```r
# split data into training and test set
set.seed(495)

n <- nrow(kchouse)
train_index <- sample(1:n, 0.75 * n)
test_index <- setdiff(1:n, train_index)

train <- kchouse[train_index, ]
test <- kchouse[test_index, ]

# construct different structure for LASSO and elastic-net
x_train <- model.matrix(overhalfmil ~ . , train)[, -1]
x_test <- model.matrix(overhalfmil ~ . , test)[, -1]
y_train <- train %>%
  dplyr::select(overhalfmil) %>%
  unlist() %>%
  as.numeric()
y_test <- test %>%
  dplyr::select(overhalfmil) %>%
  unlist() %>%
  as.numeric()
```

Variable Selection

```r
# fit kitchen-sink model, check diagnostics, obtain test error
ksmodel <- lm(overhalfmil ~ bedrooms + bathrooms + log_sqft_living + log_sqft_lot + floors + log_sqft_al
msummary(ksmodel)
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.30e+00   6.12e+00   -0.21   0.8318
## bedrooms         -2.13e-02   4.17e-03   -5.10  3.4e-07 ***
## bathrooms         9.17e-03   6.61e-03    1.39   0.1657
## log_sqft_living   2.17e-01   2.91e-02    7.45  9.4e-14 ***
## log_sqft_lot      3.34e-03   8.27e-03    0.40   0.6868
## floors            6.50e-02   8.06e-03    8.07  7.7e-16 ***
## log_sqft_above    7.26e-02   2.88e-02    2.52   0.0116 *
## basement          3.80e-02   1.21e-02    3.13   0.0017 **
## renovated         1.78e-02   1.51e-02    1.18   0.2381
## yr_built         -3.70e-03   1.47e-04  -25.13  < 2e-16 ***
## zipcode          -4.11e-04   6.74e-05   -6.10  1.1e-09 ***
## log_sqft_living15  2.18e-01   1.45e-02   15.04  < 2e-16 ***
## log_sqft_lot15   -2.27e-02   8.97e-03   -2.53   0.0115 *
## lat               9.94e-01   2.20e-02   45.26  < 2e-16 ***
## long              2.26e-02   2.72e-02    0.83   0.4061
## waterfront        4.69e-02   3.67e-02    1.28   0.2013
## view              3.29e-02   4.29e-03    7.67  1.8e-14 ***
## condition         4.44e-02   4.80e-03    9.25  < 2e-16 ***
## grade             1.22e-01   4.27e-03   28.52  < 2e-16 ***
##
## Residual standard error: 0.355 on 16178 degrees of freedom
## Multiple R-squared:  0.482,  Adjusted R-squared:  0.482
## F-statistic:  838 on 18 and 16178 DF,  p-value: <2e-16
```

```r
# doesn't make sense for logistic
# kspred <- predict(ksmodel, newdata = test)
# mean((kspred - test$overhalfmil)^2)
# mplot(ksmodel, which = 1)
# mplot(ksmodel, which = 2)

houseStep <- MASS::stepAIC(ksmodel, trace = FALSE, direction = "both")
houseStep$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## overhalfmil ~ bedrooms + bathrooms + log_sqft_living + log_sqft_lot +
##     floors + log_sqft_above + basement + renovated + yr_built +
##     zipcode + log_sqft_living15 + log_sqft_lot15 + lat + long +
##     waterfront + view + condition + grade
##
## Final Model:
## overhalfmil ~ bedrooms + bathrooms + log_sqft_living + floors +
##     log_sqft_above + basement + yr_built + zipcode + log_sqft_living15 +
##     log_sqft_lot15 + lat + view + condition + grade
##
##
##             Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                                 16178    2040.07 -33519.6
## 2 - log_sqft_lot  1 0.0205066     16179    2040.09 -33521.5
## 3         - long  1 0.0953797     16180    2040.19 -33522.7
## 4    - renovated  1 0.1800523     16181    2040.37 -33523.3
## 5   - waterfront  1 0.2206671     16182    2040.59 -33523.5
```

At a cutoff of 0.05, the kitchen-sink model output shows that the variables bathrooms, log_sqft_lot, renovated, long, and waterfront, are not significant for predicting if price is over $500,000.

Using the stepAIC() function to perform stepwise selection, we see that 4 variables are recommended to be removed from the kitchen-sink model.

```r
set.seed(495)
cv.outl <- cv.glmnet(x_train, y_train, alpha = 1) # fit lasso model
bestlaml <- cv.outl$lambda.min # select lambda that minimizes MSE bestlaml

lasso_mod <- glmnet(x_train, y_train, alpha = 1)
round(coefficients(lasso_mod, s = bestlaml), 3)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)        0.879
## bedrooms           .
## bathrooms         -0.026
## floors             0.018
## waterfront        -0.166
## view              -0.007
## condition         -0.002
## grade              0.009
## yr_built          -0.001
```

```
## zipcode            0.000
## lat               -0.091
## long               0.068
## log_price          1.102
## sqrt_price        -0.001
## basement          -0.006
## renovated         -0.022
## log_sqft_living    0.021
## log_sqft_lot      -0.006
## log_sqft_above     0.000
## log_sqft_living15  0.040
## log_sqft_lot15     0.007
```

Using LASSO on the model drives the coefficients for bedrooms, zipcode, log_sqft_above to 0.

We try an elastic-net model.

```
set.seed(495)
cv.outl <- cv.glmnet(x_train, y_train, alpha = 0.5) # fit elastic-net model
bestlaml <- cv.outl$lambda.min # select lambda that minimizes MSE bestlaml

lasso_mod <- glmnet(x_train, y_train, alpha = 0.5)
round(coefficients(lasso_mod, s = bestlaml), 3)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)        0.822
## bedrooms           0.000
## bathrooms         -0.026
## floors             0.018
## waterfront        -0.166
## view              -0.007
## condition         -0.002
## grade              0.009
## yr_built          -0.001
## zipcode            0.000
## lat               -0.091
## long               0.069
## log_price          1.100
## sqrt_price        -0.001
## basement          -0.007
## renovated         -0.022
## log_sqft_living    0.024
## log_sqft_lot      -0.007
## log_sqft_above    -0.004
## log_sqft_living15  0.040
## log_sqft_lot15     0.009
```

I prefer the LASSO model, which includes fewer predictors.

GLM Model Fitting

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

```r
# 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!

kslogmodel <- glm(overhalfmil ~ bedrooms + bathrooms + log_sqft_living + log_sqft_lot + floors + log_sq
msummary(kslogmodel)
```

```
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -6.60e+01   5.61e+01   -1.18  0.23927
## bedrooms          -2.21e-01   3.54e-02   -6.25  4.0e-10 ***
## bathrooms          2.89e-01   5.99e-02    4.83  1.4e-06 ***
## log_sqft_living    1.43e+00   2.36e-01    6.07  1.3e-09 ***
## log_sqft_lot       7.15e-02   7.55e-02    0.95  0.34369
## floors             5.48e-01   6.81e-02    8.04  9.0e-16 ***
## log_sqft_above     8.45e-01   2.33e-01    3.63  0.00028 ***
## basement           5.01e-01   1.01e-01    4.98  6.4e-07 ***
## renovated          7.71e-02   1.37e-01    0.56  0.57242
## yr_built          -3.58e-02   1.37e-03  -26.11  < 2e-16 ***
## zipcode           -2.98e-03   6.34e-04   -4.70  2.6e-06 ***
## log_sqft_living15  1.88e+00   1.32e-01   14.22  < 2e-16 ***
## log_sqft_lot15    -3.01e-01   8.20e-02   -3.67  0.00025 ***
## lat                8.75e+00   2.22e-01   39.50  < 2e-16 ***
## long               2.41e-01   2.49e-01    0.97  0.33323
## waterfront         2.94e+00   6.12e-01    4.81  1.5e-06 ***
## view               4.32e-01   4.57e-02    9.46  < 2e-16 ***
## condition          3.52e-01   4.20e-02    8.37  < 2e-16 ***
## grade              1.31e+00   4.42e-02   29.71  < 2e-16 ***
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22019  on 16196  degrees of freedom
## Residual deviance: 10943  on 16178  degrees of freedom
## AIC: 10981
##
## Number of Fisher Scoring iterations: 6
```

```r
# eliminate variables based on lm, kitchen-sink, and LASSO output
logmod <- glm(overhalfmil ~ bedrooms + log_sqft_living + floors + basement + yr_built + log_sqft_living
msummary(logmod)
```

```
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -2.82e+02   2.95e+01   -9.56  < 2e-16 ***
## bedrooms          -1.77e-01   3.43e-02   -5.15  2.6e-07 ***
## log_sqft_living    2.34e+00   1.30e-01   18.02  < 2e-16 ***
## floors             6.40e-01   6.28e-02   10.20  < 2e-16 ***
## basement           2.36e-01   5.92e-02    3.98  6.9e-05 ***
## yr_built          -3.33e-02   1.22e-03  -27.36  < 2e-16 ***
## log_sqft_living15  1.96e+00   1.30e-01   15.11  < 2e-16 ***
## log_sqft_lot15    -2.31e-01   3.98e-02   -5.81  6.3e-09 ***
## lat                8.62e+00   2.20e-01   39.21  < 2e-16 ***
## long               8.63e-01   2.19e-01    3.93  8.4e-05 ***
## waterfront         3.01e+00   6.09e-01    4.95  7.4e-07 ***
## view               4.10e-01   4.49e-02    9.13  < 2e-16 ***
## condition          3.65e-01   4.08e-02    8.95  < 2e-16 ***
```

```
## grade              1.35e+00   4.37e-02   30.91  < 2e-16 ***
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22019  on 16196  degrees of freedom
## Residual deviance: 11006  on 16183  degrees of freedom
## AIC: 11034
##
## Number of Fisher Scoring iterations: 6
```

```
lmtest::lrtest(logmod)
```

```
## Likelihood ratio test
##
## Model 1: overhalfmil ~ bedrooms + log_sqft_living + floors + basement +
##     yr_built + log_sqft_living15 + log_sqft_lot15 + lat + long +
##     waterfront + view + condition + grade
## Model 2: overhalfmil ~ 1
##   #Df LogLik  Df Chisq Pr(>Chisq)
## 1  14  -5503
## 2   1 -11010 -13 11013     <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
tally(~ overhalfmil, format = "percent", data = train)
```

```
## overhalfmil
##       0       1
## 58.1713 41.8287
```

We could get about 58.0914% right just by saying the prices are all not over $500,000.

If we use a naive cutoff of 0.5 to make predictions, we get:

```
logaug <- logmod %>%
  # predict(newdata = test, type.predict = "response")
  augment(type.predict = "response")
logaug <- mutate(logaug, binprediction = round(.fitted, 0))

tally(~ binprediction, data = logaug)
```

```
## binprediction
##    0    1
## 9744 6453
```

```
6453/(6453+9744)
```

```
## [1] 0.398407
```

It's predicting 39.84 percent of the observations are over $500,000. We make a confusion matrix from this.

```
with(logaug, table(overhalfmil, binprediction))
```

```
##            binprediction
## overhalfmil    0    1
##           0 8315 1107
##           1 1429 5346
```

```
(8315+5346)/(8315+5346+1429+1107)
```

```
## [1] 0.843428
```

Overall, we are getting 84.34% correct. The logistic glm model does alright.

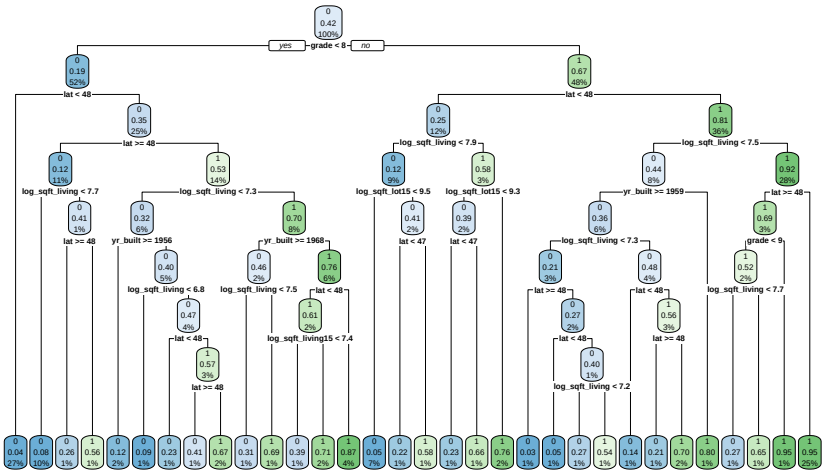
Tree Fitting

Binary response, so we use method = "class."

```
# choose and fit at least 2 different, appropriate 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!

# tree with variables eliminated by LASSO
set.seed(495)
kchouse.rpart <- rpart(overhalfmil ~ bathrooms + log_sqft_living + log_sqft_lot + floors + basement + r
printcp(kchouse.rpart)
```

```
##
## Classification tree:
## rpart(formula = overhalfmil ~ bathrooms + log_sqft_living + log_sqft_lot +
##     floors + basement + renovated + yr_built + log_sqft_living15 +
##     log_sqft_lot15 + lat + long + waterfront + view + condition +
##     grade, data = train, method = "class", control = rpart.control(minbucket = 100,
##     cp = 0, minsplit = 100))
##
## Variables actually used in tree construction:
## [1] grade             lat               log_sqft_living   log_sqft_living15
## [5] log_sqft_lot15    yr_built
##
## Root node error: 6775/16197 = 0.4183
##
## n= 16197
##
##           CP nsplit rel error xerror     xstd
## 1  0.3908487      0    1.0000 1.0000 0.009266
## 2  0.1464207      1    0.6092 0.6092 0.008185
## 3  0.0245018      2    0.4627 0.4632 0.007424
## 4  0.0221402      5    0.3892 0.3919 0.006954
## 5  0.0190406      6    0.3671 0.3811 0.006876
## 6  0.0119557      7    0.3480 0.3511 0.006650
## 7  0.0081181      8    0.3361 0.3446 0.006598
## 8  0.0065437      9    0.3280 0.3410 0.006569
## 9  0.0047232     12    0.3083 0.3190 0.006387
## 10 0.0044280     13    0.3036 0.3170 0.006371
## 11 0.0029028     15    0.2948 0.3169 0.006370
## 12 0.0026076     19    0.2815 0.3051 0.006268
## 13 0.0019926     22    0.2737 0.2954 0.006181
## 14 0.0017712     24    0.2697 0.2911 0.006143
## 15 0.0009594     26    0.2661 0.2889 0.006122
## 16 0.0003936     28    0.2642 0.2887 0.006121
## 17 0.0000000     31    0.2630 0.2887 0.006121
```

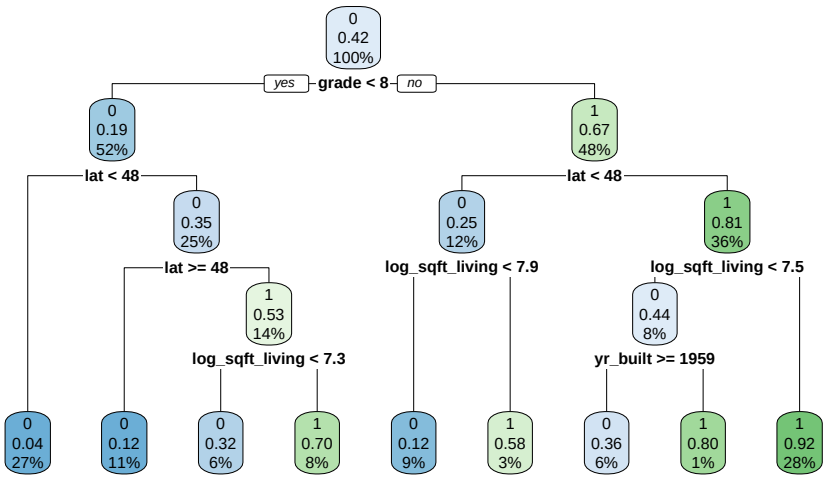
```
rpart.plot(kchouse.rpart)
```




```
# tree with variables eliminated based on lm, kitchen-sink, and LASSO output
set.seed(495)
kchouse.rpart2 <- rpart(overhalfmil ~ bedrooms + log_sqft_living + floors + basement + yr_built + log_s
printcp(kchouse.rpart2)
```

```
##
## Classification tree:
## rpart(formula = overhalfmil ~ bedrooms + log_sqft_living + floors +
##     basement + yr_built + log_sqft_living15 + log_sqft_lot15 +
##     lat + long + waterfront + view + condition + grade, data = train,
##     method = "class", control = rpart.control(minbucket = 100,
##         minsplit = 200))
##
## Variables actually used in tree construction:
## [1] grade           lat             log_sqft_living yr_built
##
## Root node error: 6775/16197 = 0.4183
##
## n= 16197
##
##         CP nsplit rel error xerror     xstd
## 1 0.39085      0    1.0000 1.0000 0.009266
## 2 0.14642      1    0.6092 0.6092 0.008185
## 3 0.02450      2    0.4627 0.4632 0.007424
## 4 0.02214      5    0.3892 0.3919 0.006954
## 5 0.01904      6    0.3671 0.3811 0.006876
## 6 0.01196      7    0.3480 0.3511 0.006650
## 7 0.01000      8    0.3361 0.3475 0.006621
```

```
rpart.plot(kchouse.rpart2)
```




```
train2 <- mutate(train, predprice = predict(kchouse.rpart, type="class"))
tally(predprice ~ overhalfmil, data = train2)
```

```
##          overhalfmil
## predprice    0    1
##         0 8455  815
##         1  967 5960
```

```
train3 <- mutate(train, predprice2 = predict(kchouse.rpart2, type="class"))
tally(predprice2 ~ overhalfmil, data = train3)
```

```
##           overhalfmil
## predprice2    0    1
##          0 8398 1253
##          1 1024 5522
```

```
tally(~ overhalfmil, data = train)
```

```
## overhalfmil
##    0    1
## 9422 6775
```

```
(8455+5960)/(8455+5960+815+967)
```

```
## [1] 0.88998
```

```
(8398+1253)/(8398+1253+1253+1024)
```

```
## [1] 0.809105
```

The bigger model, with variables eliminated by just LASSO, has a higher predictive success rate than the model containing fewer variables. I will play around with the minsplit and minbucket values more, cp too.

Thinking about Homework 5

The eventual Homework 5 submission is your write-up of the data analysis / report to the real estate developer (with minor caveats - all code must be shown). It is expected to include the following sections (along with all code necessary to reproduce your results):

- Introduction and Exploratory Data Analysis - could be two separate sections
- Your final GLM and relevant details
- Your final 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. An activity with the writing associate will help with this as well.

- 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 model 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 (and honestly, it's not a good idea to share all of that anyway - it's too much for a report!).

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 that sentence actually do that? 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. If your tree is too large for a figure, think about how else to describe it.
- 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. You can assume the developer has an intro stats level of background statistical knowledge, so they won't need you to describe basic plots and such, but the new techniques from class would be, well, new.