Peer Assessment II DBaines

1 Background

As a statistical consultant working for a real estate investment firm, your task is to develop a model to predict the selling price of a given home in Ames, Iowa. Your employer hopes to use this information to help assess whether the asking price of a house is higher or lower than the true value of the house. If the home is undervalued, it may be a good investment for the firm.

2 Training Data and relevant packages

In order to better assess the quality of the model you will produce, the data have been randomly divided into three separate pieces: a training data set, a testing data set, and a validation data set. For now we will load the training data set, the others will be loaded and used later.

```
load("ames_train.Rdata")
```

Use the code block below to load any necessary packages

```
library(statsr)
library(dplyr)
library(ggplot2)
library(BAS)
```

```
## Warning: package 'BAS' was built under R version 3.4.2
```

```
library(MASS)
```

2.1 Part 1 - Exploratory Data Analysis (EDA)

When you first get your data, it's very tempting to immediately begin fitting models and assessing how they perform. However, before you begin modeling, it's absolutely essential to explore the structure of the data and the relationships between the variables in the data set.

Do a detailed EDA of the ames_train data set, to learn about the structure of the data and the relationships between the variables in the data set (refer to Introduction to Probability and Data, Week 2, for a reminder about EDA if needed). Your EDA should involve creating and reviewing many plots/graphs and considering the patterns and relationships you see.

After you have explored completely, submit the three graphs/plots that you found most informative during your EDA process, and briefly explain what you learned from each (why you found each informative).

The first thing to be done was to restructure the `MS.SubClass` variable from the data set into a factor from a integer. With that complete, I could continue with the EDA. The next step in my process was to only analyze and regress from the normal sales in the data set so all of my models and the diagnostics of those models we're based on normal sales and not partial, abnormal, etc sales. From there, I picked a host of variables that I thought would correlate to the price of a home (Chunk 1) and noted those correlations as they may make good covariates in the model. Lastly, I graphed the most promising covariates in Figures 2-4 which represent the relationship between the log price of the home and the log area, the overall quality, and the year the home was built.

Chunk 1

```
ames_train$MS.SubClass <- as.factor(ames_train$MS.SubClass)

# build a new df with only normal sales so abnormal or partial sales don't corrupt the E
DA
normal.sales <- ames_train %>%
  filter(Sale.Condition == 'Normal')

# find correlations for variables that should affect the price of the home
# log transform non-normal variables

cor <- data_frame(
  var = c('log.area','log.lot.area','overall.quality','overall.condition','year.built',
'year.sold','bedrm','rooms','full.bath'),
  cor = c(
    cor(log(normal.sales$area),log(normal.sales$price)),
    cor(log(normal.sales$Lot.Area),log(normal.sales$price)),
    cor(normal.sales$Overall.Qual,log(normal.sales$price)),
    cor(normal.sales$Overall.Cond,log(normal.sales$price)),
    cor(normal.sales$Year.Built,log(normal.sales$price)),
    cor(normal.sales$Yr.Sold,log(normal.sales$price)),
    cor(normal.sales$Bedroom.AbvGr,log(normal.sales$price)),
    cor(normal.sales$TotRms.AbvGrd,log(normal.sales$price)),
    cor(normal.sales$Full.Bath,log(normal.sales$price))
  )
)

cor
```

```
## # A tibble: 9 x 2
##                 var         cor
##               <chr>       <dbl>
## 1          log.area  0.75795218
## 2      log.lot.area  0.39562223
## 3   overall.quality  0.82061753
## 4 overall.condition -0.07574616
## 5        year.built  0.60597666
## 6         year.sold  0.01187609
## 7             bedrm  0.27207018
## 8             rooms  0.54218041
## 9         full.bath  0.59047853
```

Figure 2

```
ggplot(normal.sales, aes(x = log(area), y = log(price))) + geom_point() + xlab('Log of A
rea of Home') + ylab('Log of Home Price')
```

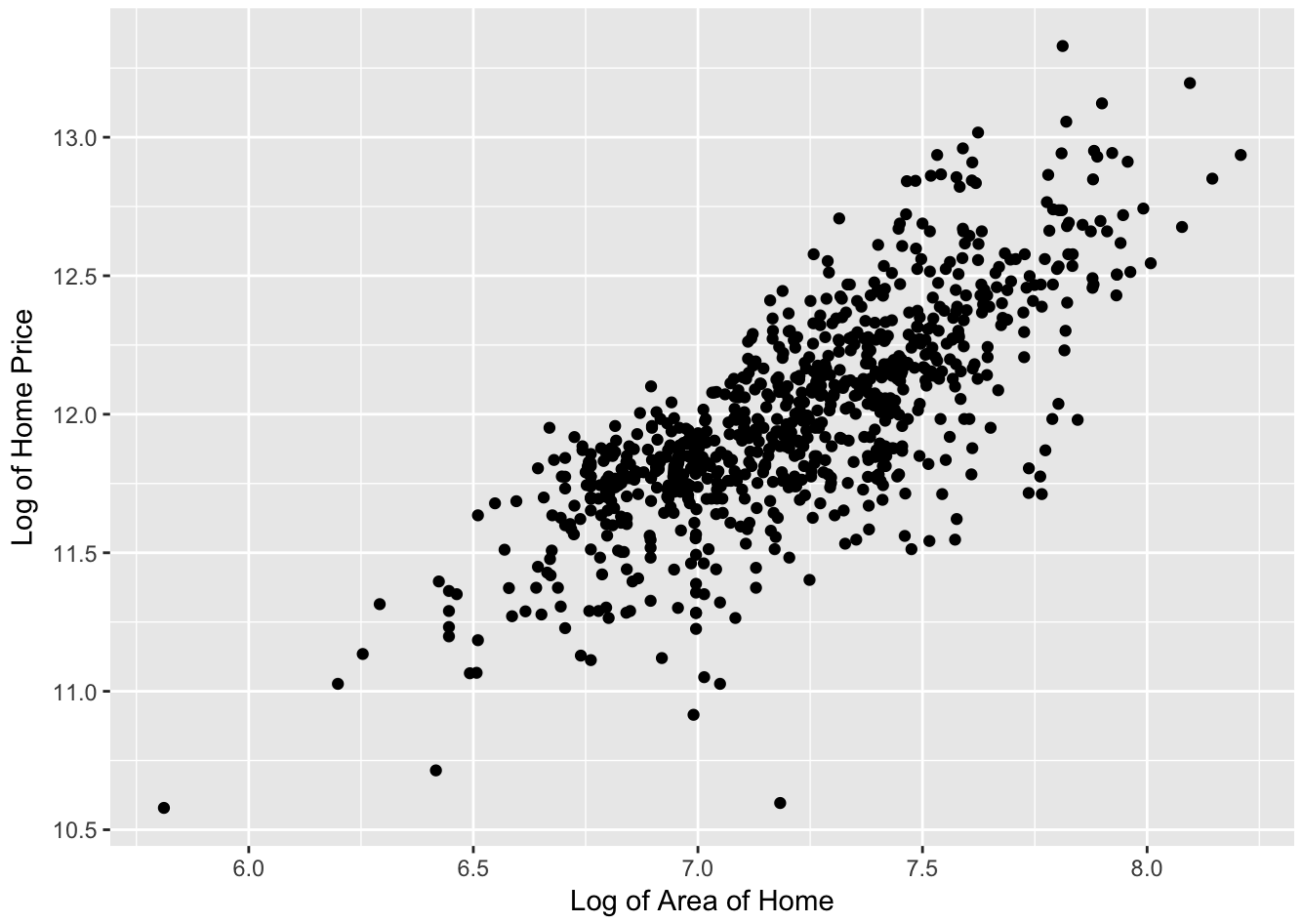


Figure 3

```
ggplot(normal.sales, aes(x = Overall.Qual, y = log(price))) + geom_point() + xlab('Overa
ll Quality') + ylab('Log of Home Price')
```

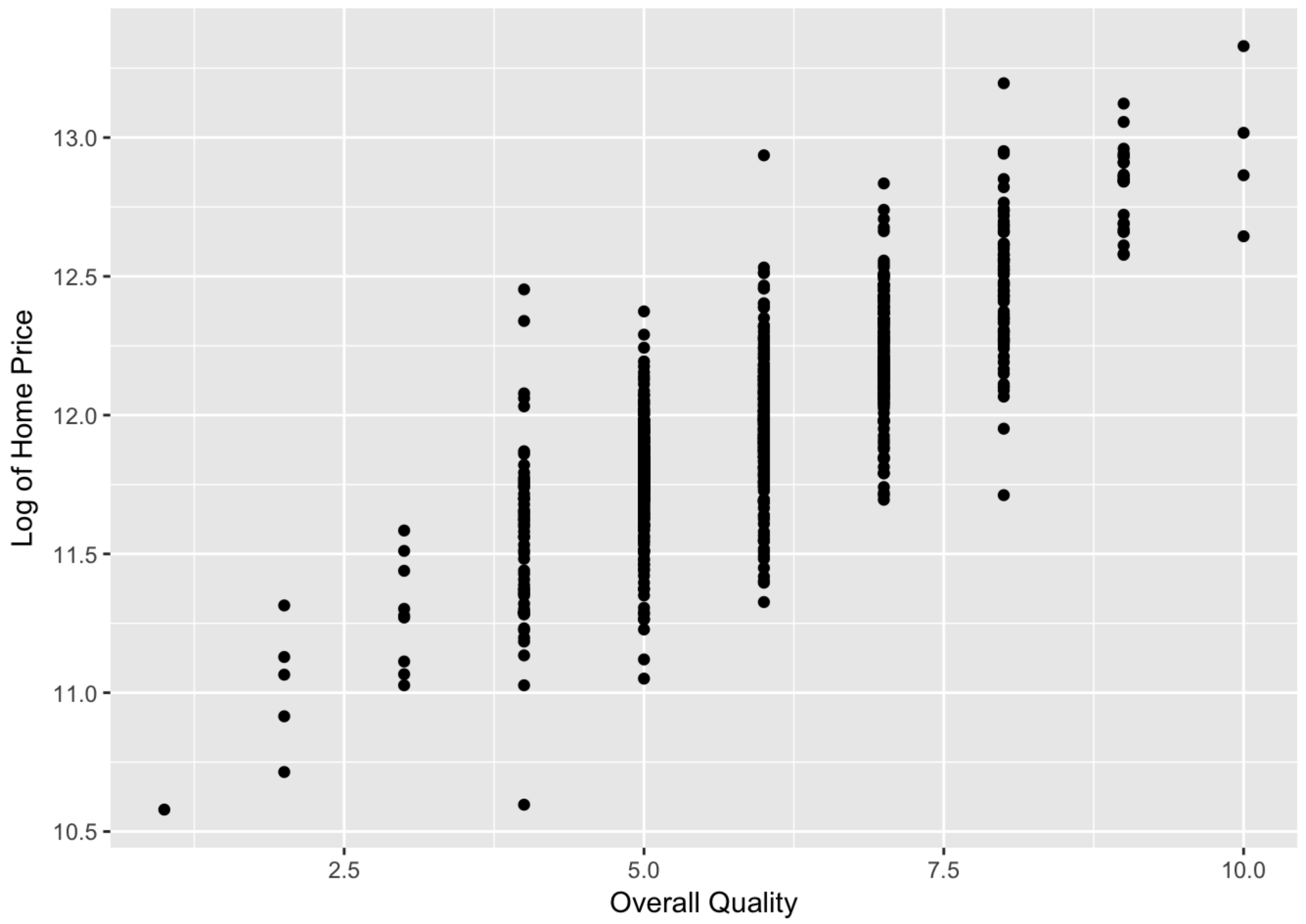
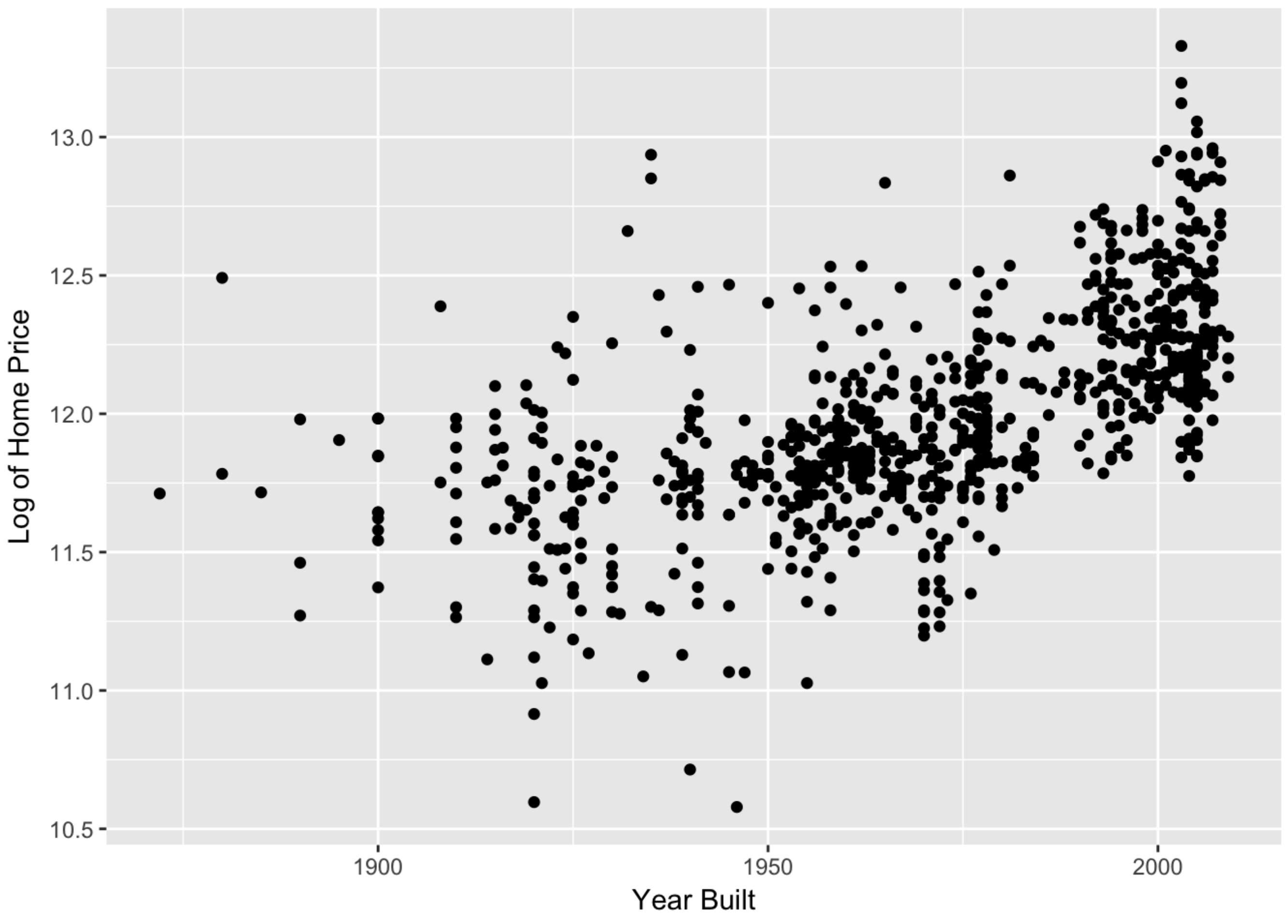


Figure 4

```
ggplot(normal.sales, aes(x = Year.Built, y = log(price))) + geom_point() + xlab('Year Bu
ilt') + ylab('Log of Home Price')
```



2.2 Part 2 - Development and assessment of an initial model, following a semi-guided process of analysis

2.2.1 Section 2.1 An Initial Model

In building a model, it is often useful to start by creating a simple, intuitive initial model based on the results of the exploratory data analysis. (Note: The goal at this stage is **not** to identify the "best" possible model but rather to choose a reasonable and understandable starting point. Later you will expand and revise this model to create your final model.

Based on your EDA, select *at most* 10 predictor variables from "ames_train" and create a linear model for `price` (or a transformed version of price) using those variables. Provide the *R code* and the *summary output table* for your model, a *brief justification* for the variables you have chosen, and a *brief discussion* of the model results in context (focused on the variables that appear to be important predictors and how they relate to sales price).

I built the below intial model using the variables that had the highest correlation to `log(price)` as well as variables that I thought might make sense when predicting home price. From the summary output of the initial model, the adjusted r-squared of 86.14% means that 86.14% of the variation in the log price of the homes in the training data set are related to the variables in this initial model (with penalties for the number of variables). That's not a bad starting point (anything over 70% is good) but I'll drill deeper into the makeup of the model in the following sections.

Chunk 5

```
model.full <- lm(log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qual + Be
droom.AbvGr + TotRms.AbvGrd + Full.Bath,normal.sales)
summary(model.full)
```

```
##
## Call:
## lm(formula = log(price) ~ log(area) + log(Lot.Area) + Year.Built +
##     Overall.Qual + Bedroom.AbvGr + TotRms.AbvGrd + Full.Bath,
##     data = normal.sales)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97210 -0.08370  0.00655  0.09320  0.58597
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.7495250  0.4857374  -1.543   0.1232
## log(area)      0.4964457  0.0321243  15.454  < 2e-16 ***
## log(Lot.Area)  0.1589831  0.0101776  15.621  < 2e-16 ***
## Year.Built     0.0036502  0.0002236  16.325  < 2e-16 ***
## Overall.Qual   0.1149933  0.0057970  19.837  < 2e-16 ***
## Bedroom.AbvGr -0.0391574  0.0088802  -4.410 1.17e-05 ***
## TotRms.AbvGrd -0.0005214  0.0061426  -0.085   0.9324
## Full.Bath     -0.0340287  0.0135634  -2.509   0.0123 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1422 on 826 degrees of freedom
## Multiple R-squared:  0.8626, Adjusted R-squared:  0.8614
## F-statistic: 740.7 on 7 and 826 DF,  p-value: < 2.2e-16
```

2.2.2 Section 2.2 Model Selection

Now either using `BAS` another stepwise selection procedure choose the "best" model you can, using your initial model as your starting point. Try at least two different model selection methods and compare their results. Do they both arrive at the same model or do they disagree? What do you think this means?

I decided to use three different model selection strategies to determine the "best" model based on the intitial model that I created in the above section. By taking a look at all three of these techniques - BAS analysis, BIC, and AIC - I decided on a model that had a slightly lower adjusted r-squared than my initial model, but whose variables made more sense given the question being asked.

First, by constructing and then analyzing the Bayesian regression model in Chunk 6 and Figure 7, I thought I should remove the `Full.Bath` and `TotRms.AbvGrd` variables since the probabilty those variables affect the log price of that home were 46% and 3.4%, respectively.

Secondly, according to the AIC method in Chunk 8, the final model should also exclude the `Full.Bath` and `TotRms.AbvGrd` variables since their inclusion results in a higher AIC for the model.

Lastly, Chunks 9-12 use the BIC method of analyzing the efficiency of the model. By adding and removing variables in a trial and error fashion, the best model according to that process also remove the `Full.Bath` and `TotRms.AbvGrd` variables to achieve the lowest possible BIC. With both the BIC and AIC methods, having a higher value indicates that there are variables in the model that can be removed or other added to make the model's predictive power more accurate.

For this intial assessment of my model, Chunk 13 holds the summary data for these final initial model to be tested.

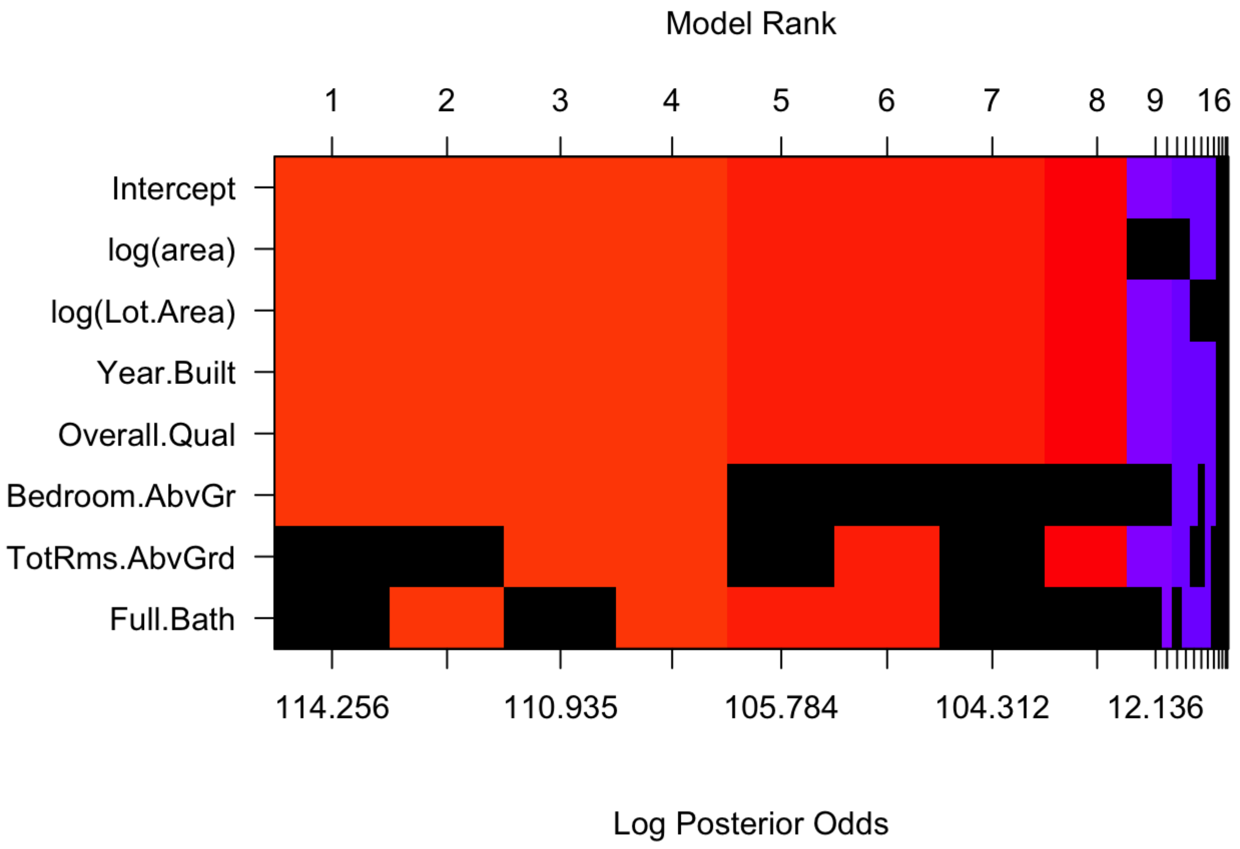
Chunk 6

```
# build a bayesian regression model that we can double check with both AIC and BIC
model.full.bas <- bas.lm(log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Q
ual + Bedroom.AbvGr + TotRms.AbvGrd + Full.Bath, normal.sales, prior = 'BIC', modelprior
= uniform())
summary(model.full.bas)
```

```
##               P(B != 0 | Y)    model 1       model 2       model 3
## Intercept        1.00000000     1.0000     1.0000000  1.000000e+00
## log(area)        1.00000000     1.0000     1.0000000  1.000000e+00
## log(Lot.Area)    1.00000000     1.0000     1.0000000  1.000000e+00
## Year.Built       1.00000000     1.0000     1.0000000  1.000000e+00
## Overall.Qual     1.00000000     1.0000     1.0000000  1.000000e+00
## Bedroom.AbvGr    0.99982511     1.0000     1.0000000  1.000000e+00
## TotRms.AbvGrd    0.03429629     0.0000     0.0000000  1.000000e+00
## Full.Bath        0.45990901     0.0000     1.0000000  0.000000e+00
## BF                       NA     1.0000     0.8524512  3.609375e-02
## PostProbs                NA     0.5212     0.4443000  1.880000e-02
## R2                       NA     0.8615     0.8626000  8.615000e-01
## dim                      NA     6.0000     7.0000000  7.000000e+00
## logmarg                  NA -1197.2834 -1197.4430477 -1.200605e+03
##                     model 4       model 5
## Intercept      1.000000e+00  1.000000e+00
## log(area)      1.000000e+00  1.000000e+00
## log(Lot.Area)  1.000000e+00  1.000000e+00
## Year.Built     1.000000e+00  1.000000e+00
## Overall.Qual   1.000000e+00  1.000000e+00
## Bedroom.AbvGr  1.000000e+00  0.000000e+00
## TotRms.AbvGrd  1.000000e+00  0.000000e+00
## Full.Bath      1.000000e+00  1.000000e+00
## BF             2.962554e-02  2.091542e-04
## PostProbs      1.540000e-02  1.000000e-04
## R2             8.626000e-01  8.587000e-01
## dim            8.000000e+00  6.000000e+00
## logmarg       -1.200803e+03 -1.205756e+03
```

Figure 7

```
image(model.full.bas,rotate = FALSE)
```



Chunk 8

```
stepAIC(k = log(nrow(normal.sales)),model.full)
```

```
## Start:  AIC=-3208.07
## log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qual +
##     Bedroom.AbvGr + TotRms.AbvGrd + Full.Bath
##
##                  Df Sum of Sq    RSS     AIC
## - TotRms.AbvGrd  1    0.0001 16.695 -3214.8
## - Full.Bath      1    0.1272 16.822 -3208.5
## <none>                       16.695 -3208.1
## - Bedroom.AbvGr  1    0.3930 17.088 -3195.4
## - log(area)      1    4.8270 21.522 -3003.0
## - log(Lot.Area)  1    4.9319 21.627 -2998.9
## - Year.Built     1    5.3864 22.081 -2981.6
## - Overall.Qual   1    7.9532 24.648 -2889.9
##
## Step:  AIC=-3214.79
## log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qual +
##     Bedroom.AbvGr + Full.Bath
##
##                  Df Sum of Sq    RSS     AIC
## - Full.Bath      1    0.1287 16.824 -3215.1
## <none>                       16.695 -3214.8
## - Bedroom.AbvGr  1    0.4741 17.169 -3198.2
## - log(Lot.Area)  1    5.0154 21.710 -3002.4
## - Year.Built     1    5.4832 22.178 -2984.7
## - log(area)      1    6.3878 23.083 -2951.3
## - Overall.Qual   1    7.9561 24.651 -2896.5
##
## Step:  AIC=-3215.11
## log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qual +
##     Bedroom.AbvGr
##
##                  Df Sum of Sq    RSS     AIC
## <none>                       16.824 -3215.1
## - Bedroom.AbvGr  1    0.5455 17.369 -3195.2
## - log(Lot.Area)  1    5.1689 21.993 -2998.4
## - Year.Built     1    5.5914 22.415 -2982.5
## - log(area)      1    6.8081 23.632 -2938.4
## - Overall.Qual   1    7.9095 24.733 -2900.5
```

```
##
## Call:
## lm(formula = log(price) ~ log(area) + log(Lot.Area) + Year.Built +
##     Overall.Qual + Bedroom.AbvGr, data = normal.sales)
##
## Coefficients:
##   (Intercept)      log(area)  log(Lot.Area)     Year.Built   Overall.Qual
##     -0.213468       0.466309       0.160810       0.003458       0.114606
## Bedroom.AbvGr
##     -0.042002
```

Chunk 9

```
BIC(model.full)
```

```
## [1] -834.5579
```

Chunk 10

```
# remove the 'Bedrooms.AbvGr' variable since that has the highest p-value in the lm and
 the lowest probablity of affecting the response variable in the bas.lm
model.nobdrms <- lm(log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qual +
TotRms.AbvGrd,normal.sales)

BIC(model.nobdrms)
```

```
## [1] -820.4852
```

Chunk 11

```
# remove the 'Bedrooms.AbvGr' variable since that has the highest p-value in the lm and
 the lowest probablity of affecting the response variable in the bas.lm
model.norms <- lm(log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qual + B
edroom.AbvGr,normal.sales)

BIC(model.norms)
```

```
## [1] -841.5961
```

Chunk 12

```
# remove the 'Bedrooms.AbvGr' variable since that has the highest p-value in the lm and
 the lowest probablity of affecting the response variable in the bas.lm
model.normsatall <- lm(log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qua
l,normal.sales)

BIC(model.normsatall)
```

```
## [1] -821.7085
```

Chunk 13

```
# summarize the final model from the BAS, BIC, and AIC processes
final.model <- lm(log(price) ~ log(area) + log(Lot.Area) + Year.Built + Overall.Qual + B
edroom.AbvGr,normal.sales)
summary(final.model)
```

```
##
## Call:
## lm(formula = log(price) ~ log(area) + log(Lot.Area) + Year.Built +
##     Overall.Qual + Bedroom.AbvGr, data = normal.sales)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.96581 -0.08579  0.00802  0.09369  0.56239
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.2134679  0.4363425  -0.489    0.625
## log(area)      0.4663086  0.0254746  18.305  < 2e-16 ***
## log(Lot.Area)  0.1608100  0.0100823  15.950  < 2e-16 ***
## Year.Built     0.0034578  0.0002084  16.589  < 2e-16 ***
## Overall.Qual   0.1146059  0.0058087  19.730  < 2e-16 ***
## Bedroom.AbvGr -0.0420016  0.0081060  -5.182 2.77e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1425 on 828 degrees of freedom
## Multiple R-squared:  0.8615, Adjusted R-squared:  0.8607
## F-statistic:  1030 on 5 and 828 DF,  p-value: < 2.2e-16
```

2.2.3 Section 2.3 Initial Model Residuals

One way to assess the performance of a model is to examine the model's residuals. In the space below, create a residual plot for your preferred model from above and use it to assess whether your model appears to fit the data well. Comment on any interesting structure in the residual plot (trend, outliers, etc.) and briefly discuss potential implications it may have for your model and inference / prediction you might produce.

I used two plots to check on the residuals of my model on the training set: Figure 14 and 15. The plots show that as the price increases, the model becomes less accurate. I also created the histogram of the log of price in Figure 16 to see where the median log price was (11.95) and to see how the model performed at that level since that's where the highest concentration of obversations exist. There are also two outliers at the high end of the price spectrum where the model was almost $200,000 off. These would affect the Residual Mean Squared Error for this training data set but potentially not other data sets so I'll keep this fact in mind but not change the model based on 2 of 834 observations in the data set on which the model was created.

Figure 14

```
normal.sales$predicted.price <- exp(predict(final.model))
normal.sales$residual <- sqrt((normal.sales$predicted.price - normal.sales$price)^2)
ggplot(normal.sales, aes(residual, log(price))) + geom_point() + ylab('Log of Price') +
xlab('Residual')
```

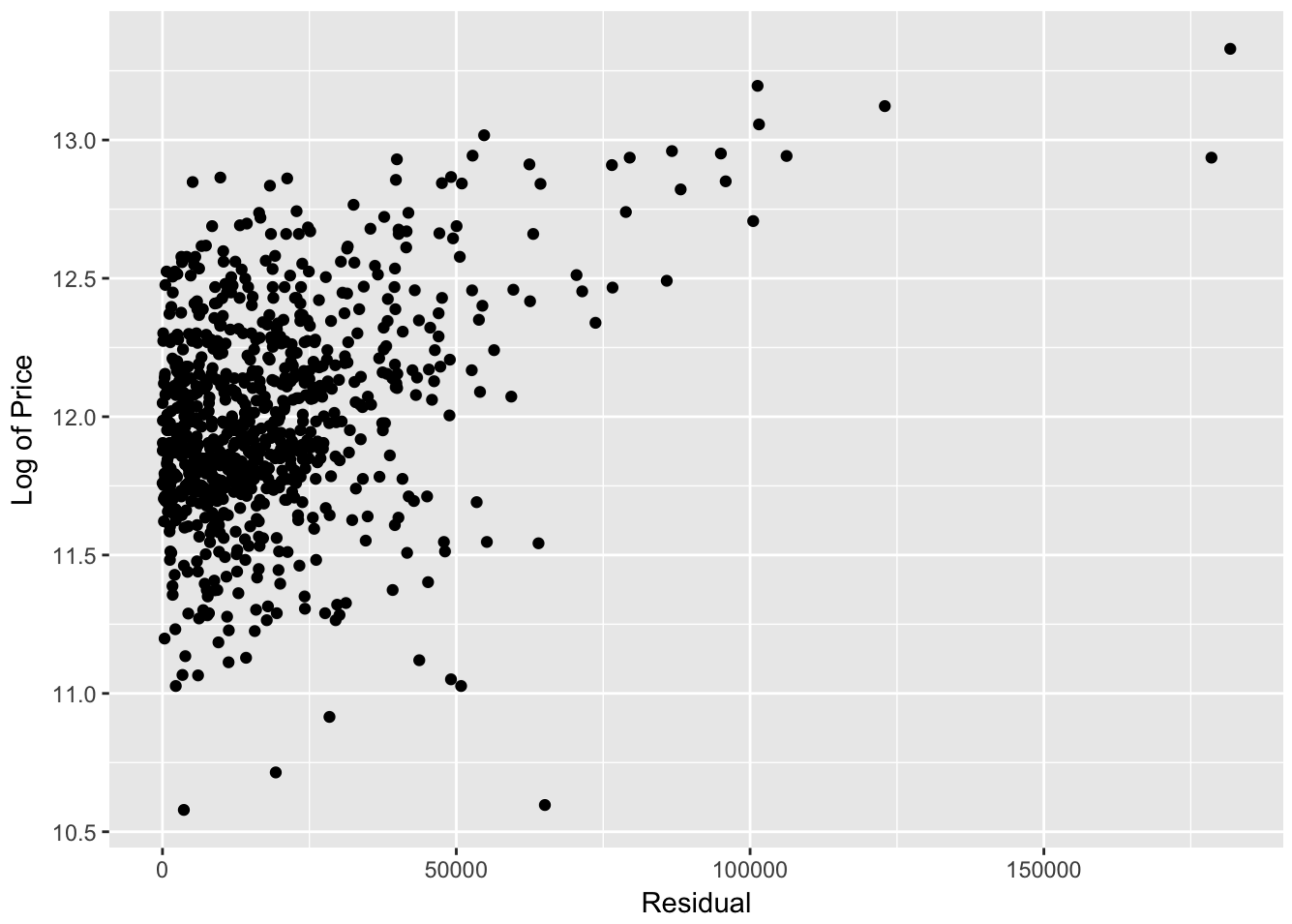


Figure 15

```
qqplot(normal.sales$residual,log(normal.sales$price))
```

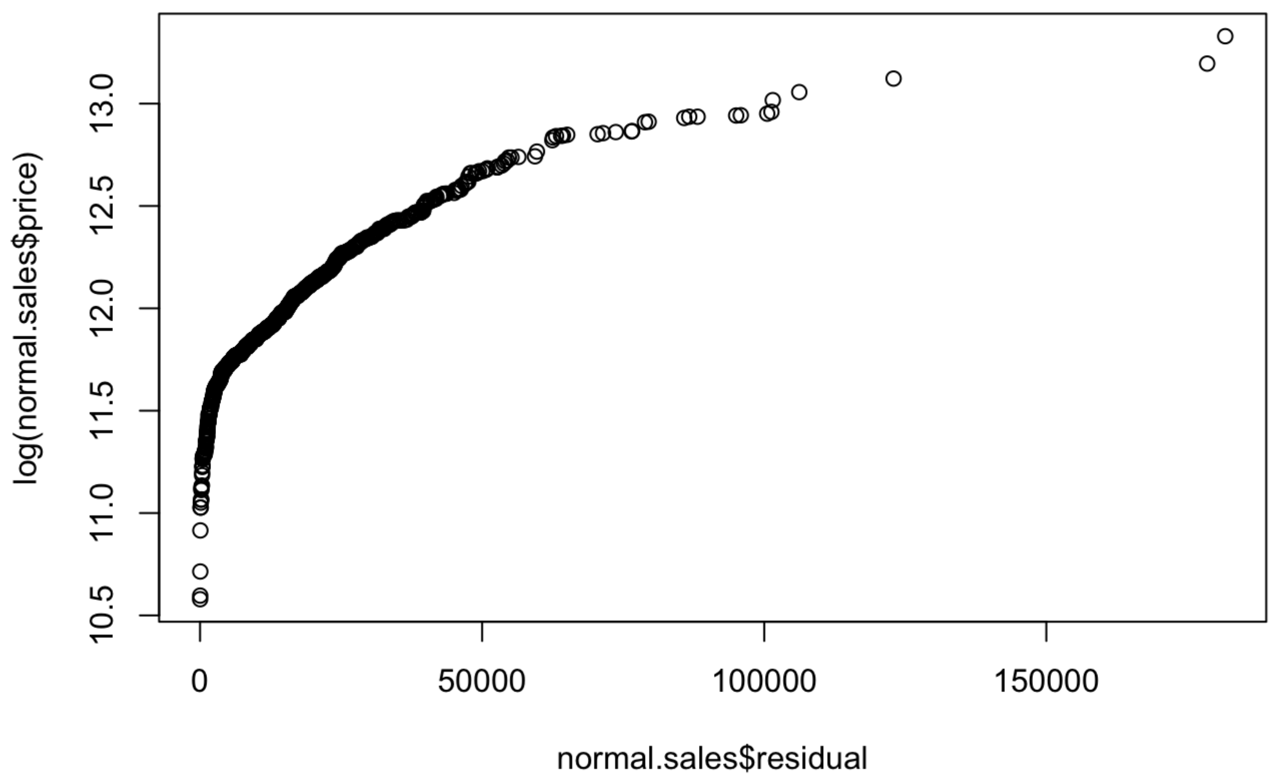
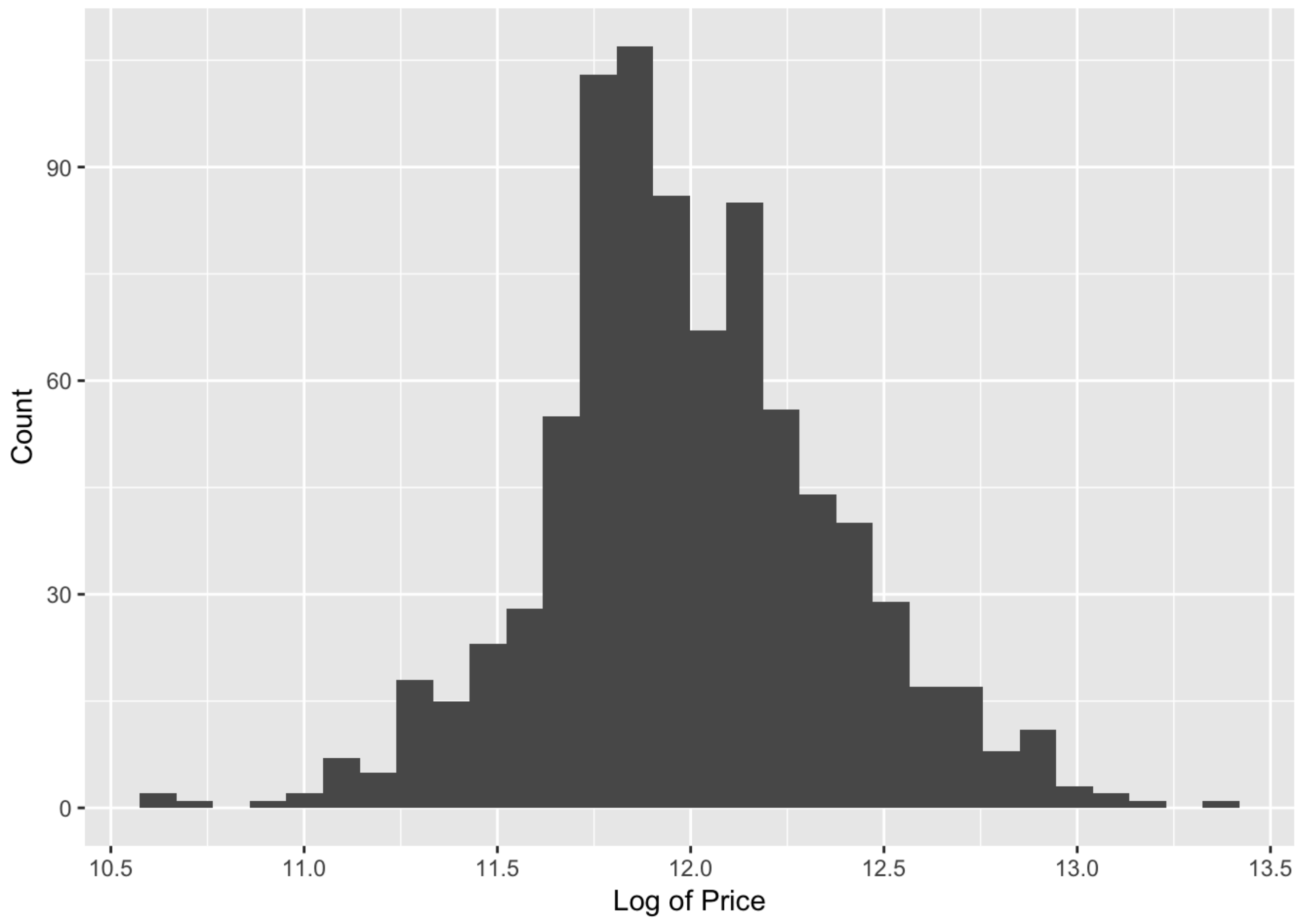


Figure 16

```
ggplot(normal.sales, aes(log(price))) + geom_histogram() + xlab('Log of Price') + ylab(
'Count')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



2.2.4 Section 2.4 Initial Model RMSE

You can calculate it directly based on the model output. Be specific about the units of your RMSE (depending on whether you transformed your response variable). The value you report will be more meaningful if it is in the original units (dollars).

The RMSE for the training set based on the above model is $18,475.21. This means that, with this model and the training data set, my model's predicted price is off by $18,475 of the real price of the home on average.

Chunk 17

```
mean(normal.sales$residual)
```

```
## [1] 18475.21
```

2.2.5 Section 2.5 Overfitting

The process of building a model generally involves starting with an initial model (as you have done above), identifying its shortcomings, and adapting the model accordingly. This process may be repeated several times until the model fits the data reasonably well. However, the model may do well on training data but perform poorly out-of-sample (meaning, on a dataset other than the original training data) because the model is overly-tuned to specifically fit the training data. This is called “overfitting.” To determine whether overfitting is occurring on a model, compare the performance of a model on both in-sample and out-of-sample data sets. To look at performance of your initial model on out-of-sample data, you will use the data set `ames_test`.

```
load("ames_test.Rdata")
```

Use your model from above to generate predictions for the housing prices in the test data set. Are the predictions significantly more accurate (compared to the actual sales prices) for the training data than the test data? Why or why not? Briefly explain how you determined that (what steps or processes did you use)?

I used the model above on the test data set (with the restructured `MS.SubClass` variable and all but normal sales excluded) and arrived at a RMSE of $19,708 (Chunk 18). While the out-of-sample RMSE from the test data set is a bit higher than the RMSE from the train data set, Chunk 19 and 20 show that the model has better “coverage” on the test data set than the train data set. In other words, with this model, there are more predictions that fall in the 95% interval of the real prices of the homes in the test data set than in the train data set. This could be because are fewer outliers in the test training set, but I’d have to do more analysis to confirm that.

Chunk 18

```
# factor ms.subclass
ames_test$MS.SubClass <- as.factor(ames_test$MS.SubClass)

normal.sales.test <- ames_test %>%
  filter(Sale.Condition == 'Normal')

normal.sales.test$predicted.price <- exp(predict(final.model,ames_test))
normal.sales.test$residual <- sqrt((normal.sales.test$predicted.price - normal.sales.tes
t$price)^2)

mean(normal.sales.test$residual)
```

```
## [1] 19708.74
```

Chunk 19

```
predict.full.train <- exp(predict(final.model, normal.sales, interval = 'prediction'))
coverage.full.train <-
  mean(normal.sales$price > predict.full.train[,'lwr'] &
  normal.sales$price < predict.full.train[,'upr'])
coverage.full.train
```

```
## [1] 0.9496403
```

Chunk 20

```
predict.full.test <- exp(predict(final.model, normal.sales.test, interval = 'prediction'
))
coverage.full.test <-
  mean(normal.sales.test$price > predict.full.test[,'lwr'] &
  normal.sales.test$price < predict.full.test[,'upr'])
coverage.full.test
```

```
## [1] 0.9522644
```

Note to the learner: If in real-life practice this out-of-sample analysis shows evidence that the training data fits your model a lot better than the test data, it is probably a good idea to go back and revise the model (usually by simplifying the model) to reduce this overfitting. For simplicity, we do not ask you to do this on the assignment, however.

2.3 Part 3 Development of a Final Model

Now that you have developed an initial model to use as a baseline, create a final model with *at most* 20 variables to predict housing prices in Ames, IA, selecting from the full array of variables in the dataset and using any of the tools that we introduced in this specialization.

Carefully document the process that you used to come up with your final model, so that you can answer the questions below.

2.3.1 Section 3.1 Final Model

Provide the summary table for your model.

For the development of this final model, I used the variables from the initial model that were tested for validity as well as a few other variables that I thought would affect the price of a home. To start the diagnostic process, I reviewed the adjusted r-squared which, at 90.65%, was already higher than that of the initial model so I knew I was onto something (Chunk 21). I then used the AIC method (with the stepAIC() function) to develop the “best” fit model (chunk 22 and 23). Finally, I calculated the RMSE and the coverage for the model to verify that the model worked well on the training and test data (Chunk 24-27).

Chunk 21

```
model.full <- lm(log(price) ~
                   log(area) +
                   log(Lot.Area) +
                   Bedroom.AbvGr +
                   TotRms.AbvGrd +
                   Year.Built +
                   Overall.Qual +
                   Overall.Cond +
                   Neighborhood +
                   Bldg.Type +
                   Full.Bath +
                   Half.Bath +
                   MS.SubClass +
                   Yr.Sold,
                 normal.sales
                 )

summary(model.full)
```

```
##
## Call:
## lm(formula = log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr +
##     TotRms.AbvGrd + Year.Built + Overall.Qual + Overall.Cond +
##     Neighborhood + Bldg.Type + Full.Bath + Half.Bath + MS.SubClass +
##     Yr.Sold, data = normal.sales)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.78566 -0.06238  0.00022  0.06869  0.42370
##
## Coefficients: (2 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -6.4951518  6.3297580  -1.026  0.30515
## log(area)            0.6017838  0.0311620  19.311  < 2e-16 ***
## log(Lot.Area)        0.1183081  0.0134384   8.804  < 2e-16 ***
## Bedroom.AbvGr       -0.0218353  0.0081376  -2.683  0.00745 **
## TotRms.AbvGrd       -0.0048081  0.0054061  -0.889  0.37408
## Year.Built           0.0045970  0.0004084  11.255  < 2e-16 ***
## Overall.Qual         0.0703568  0.0055628  12.648  < 2e-16 ***
## Overall.Cond         0.0598587  0.0041149  14.547  < 2e-16 ***
## NeighborhoodBlueste  0.0746141  0.0851058   0.877  0.38091
## NeighborhoodBrDale  -0.0344332  0.0708584  -0.486  0.62714
## NeighborhoodBrkSide -0.0222329  0.0565039  -0.393  0.69408
## NeighborhoodClearCr -0.0068904  0.0624877  -0.110  0.91222
## NeighborhoodCollgCr -0.0862213  0.0503745  -1.712  0.08737 .
## NeighborhoodCrawfor  0.0454411  0.0560864   0.810  0.41807
## NeighborhoodEdwards -0.1406954  0.0526452  -2.673  0.00769 **
## NeighborhoodGilbert -0.1058611  0.0529197  -2.000  0.04580 *
## NeighborhoodGreens   0.1532072  0.0733433   2.089  0.03704 *
## NeighborhoodGrnHill  0.2902253  0.0917344   3.164  0.00162 **
## NeighborhoodIDOTRR  -0.1478141  0.0577751  -2.558  0.01070 *
## NeighborhoodLandmrk  0.0215575  0.1254060   0.172  0.86356
## NeighborhoodMeadowV -0.0909477  0.0614154  -1.481  0.13905
## NeighborhoodMitchel -0.0423549  0.0522641  -0.810  0.41796
## NeighborhoodNAmes   -0.0640849  0.0519446  -1.234  0.21768
## NeighborhoodNoRidge  0.0413080  0.0533595   0.774  0.43908
## NeighborhoodNPkVill  0.0405005  0.0752477   0.538  0.59057
## NeighborhoodNridgHt  0.0953640  0.0499103   1.911  0.05641 .
## NeighborhoodNWAmes  -0.0987106  0.0528338  -1.868  0.06209 .
## NeighborhoodOldTown -0.1099336  0.0559454  -1.965  0.04977 *
## NeighborhoodSawyer  -0.0784386  0.0531297  -1.476  0.14025
## NeighborhoodSawyerW -0.1339670  0.0516054  -2.596  0.00961 **
## NeighborhoodSomerst  0.0352446  0.0500980   0.704  0.48195
## NeighborhoodStoneBr  0.0659945  0.0553522   1.192  0.23352
## NeighborhoodSWISU   -0.0850053  0.0651554  -1.305  0.19240
## NeighborhoodTimber  -0.0195377  0.0560398  -0.349  0.72745
## NeighborhoodVeenker -0.0092952  0.0619480  -0.150  0.88077
## Bldg.Type2fmCon      0.0524471  0.0828858   0.633  0.52707
## Bldg.TypeDuplex     -0.1368489  0.0259480  -5.274 1.73e-07 ***
## Bldg.TypeTwnhs      -0.0071722  0.0607425  -0.118  0.90604
## Bldg.TypeTwnhsE      0.0260159  0.0571837   0.455  0.64927
## Full.Bath            0.0013642  0.0131909   0.103  0.91765
## Half.Bath            0.0139754  0.0121189   1.153  0.24918    
## MS.SubClass30        0.0147046  0.0260907   0.564  0.57319    
## MS.SubClass40       -0.3568756  0.1158460  -3.081  0.00214 ** 
## MS.SubClass45       -0.0185476  0.0473448  -0.392  0.69535    
## MS.SubClass50       -0.0678194  0.0215036  -3.154  0.00167 ** 
## MS.SubClass60       -0.1093579  0.0170946  -6.397 2.73e-10 ***
## MS.SubClass70       -0.0630965  0.0316040  -1.996  0.04623 *  
## MS.SubClass75       -0.0062772  0.0504836  -0.124  0.90108    
## MS.SubClass80       -0.0413157  0.0205564  -2.010  0.04479 *  
## MS.SubClass85        0.0583225  0.0268200   2.175  0.02996 *  
## MS.SubClass90               NA         NA      NA       NA    
## MS.SubClass120      -0.0685116  0.0563003  -1.217  0.22401    
## MS.SubClass160      -0.2000093  0.0546623  -3.659  0.00027 ***
## MS.SubClass180              NA         NA      NA       NA    
## MS.SubClass190      -0.0681692  0.0860968  -0.792  0.42873    
## Yr.Sold              0.0017137  0.0031060   0.552  0.58129    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1124 on 780 degrees of freedom
## Multiple R-squared:  0.9189, Adjusted R-squared:  0.9134 
## F-statistic: 166.7 on 53 and 780 DF,  p-value: < 2.2e-16
```

Chunk 22

```
#step aic the model
model.full.aic <- stepAIC(model.full)
```

```
## Start:  AIC=-3593.5
## log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr + TotRms.AbvGrd +
##     Year.Built + Overall.Qual + Overall.Cond + Neighborhood +
##     Bldg.Type + Full.Bath + Half.Bath + MS.SubClass + Yr.Sold
##
##                 Df Sum of Sq     RSS     AIC
## - Bldg.Type      2    0.0208  9.8758 -3595.7
## - Full.Bath      1    0.0001  9.8552 -3595.5
## - Yr.Sold        1    0.0038  9.8589 -3595.2
## - TotRms.AbvGrd  1    0.0100  9.8650 -3594.7
## - Half.Bath      1    0.0168  9.8718 -3594.1
## <none>                        9.8550 -3593.5
## - Bedroom.AbvGr  1    0.0910  9.9460 -3587.8
## - MS.SubClass   12    0.9451 10.8001 -3541.1
## - log(Lot.Area)  1    0.9793 10.8343 -3516.5
## - Year.Built     1    1.6005 11.4556 -3470.0
## - Neighborhood  27    2.3772 12.2323 -3467.3
## - Overall.Qual   1    2.0211 11.8761 -3439.9
## - Overall.Cond   1    2.6736 12.5286 -3395.3
## - log(area)      1    4.7119 14.5669 -3269.6
##
## Step:  AIC=-3595.74
## log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr + TotRms.AbvGrd +
##     Year.Built + Overall.Qual + Overall.Cond + Neighborhood +
##     Full.Bath + Half.Bath + MS.SubClass + Yr.Sold
##
##                 Df Sum of Sq     RSS     AIC
## - Full.Bath      1    0.0000  9.8759 -3597.7
## - Yr.Sold        1    0.0040  9.8799 -3597.4
## - TotRms.AbvGrd  1    0.0125  9.8884 -3596.7
## - Half.Bath      1    0.0161  9.8919 -3596.4
## <none>                        9.8758 -3595.7
## - Bedroom.AbvGr  1    0.0838  9.9597 -3590.7
## - MS.SubClass   14    1.2415 11.1174 -3525.0
## - log(Lot.Area)  1    1.0389 10.9148 -3514.3
## - Year.Built     1    1.5997 11.4756 -3472.5
## - Neighborhood  27    2.3614 12.2373 -3470.9
## - Overall.Qual   1    2.0450 11.9208 -3440.8
## - Overall.Cond   1    2.6608 12.5366 -3398.8
## - log(area)      1    4.7180 14.5939 -3272.1
##
## Step:  AIC=-3597.74
## log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr + TotRms.AbvGrd +
##     Year.Built + Overall.Qual + Overall.Cond + Neighborhood +
##     Half.Bath + MS.SubClass + Yr.Sold
##
##                 Df Sum of Sq     RSS     AIC
## - Yr.Sold        1    0.0041  9.8799 -3599.4
## - TotRms.AbvGrd  1    0.0125  9.8884 -3598.7
## - Half.Bath      1    0.0178  9.8936 -3598.2
## <none>                        9.8759 -3597.7
## - Bedroom.AbvGr  1    0.0859  9.9617 -3592.5
## - MS.SubClass   14    1.2696 11.1455 -3524.9
## - log(Lot.Area)  1    1.0389 10.9148 -3516.3
## - Neighborhood  27    2.3878 12.2636 -3471.1
## - Year.Built     1    1.6467 11.5226 -3471.1
## - Overall.Qual   1    2.0452 11.9210 -3442.8
## - Overall.Cond   1    2.6611 12.5369 -3400.8
## - log(area)      1    5.3585 15.2344 -3238.2
##
## Step:  AIC=-3599.4
## log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr + TotRms.AbvGrd +
##     Year.Built + Overall.Qual + Overall.Cond + Neighborhood +
##     Half.Bath + MS.SubClass
##
##                 Df Sum of Sq     RSS     AIC
## - TotRms.AbvGrd  1    0.0126  9.8926 -3600.3
## - Half.Bath      1    0.0195  9.8994 -3599.8
## <none>                        9.8799 -3599.4
## - Bedroom.AbvGr  1    0.0841  9.9640 -3594.3
## - MS.SubClass   14    1.2836 11.1635 -3525.5
## - log(Lot.Area)  1    1.0350 10.9150 -3518.3
## - Year.Built     1    1.6434 11.5233 -3473.1
## - Neighborhood  27    2.3849 12.2648 -3473.1
## - Overall.Qual   1    2.0423 11.9222 -3444.7
## - Overall.Cond   1    2.6574 12.5374 -3402.7
## - log(area)      1    5.3594 15.2393 -3240.0
##
## Step:  AIC=-3600.33
## log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr + Year.Built +
##     Overall.Qual + Overall.Cond + Neighborhood + Half.Bath +
##     MS.SubClass
##
##                 Df Sum of Sq     RSS     AIC
## - Half.Bath      1    0.0222  9.9147 -3600.5
## <none>                        9.8926 -3600.3
## - Bedroom.AbvGr  1    0.1242 10.0168 -3591.9
## - MS.SubClass   14    1.2925 11.1851 -3525.9
## - log(Lot.Area)  1    1.0326 10.9252 -3519.5
## - Neighborhood  27    2.3753 12.2678 -3474.9
## - Year.Built     1    1.6615 11.5540 -3472.9
## - Overall.Qual   1    2.0393 11.9319 -3446.0
## - Overall.Cond   1    2.6522 12.5447 -3404.2
## - log(area)      1    6.7796 16.6721 -3167.0
##
## Step:  AIC=-3600.46
## log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr + Year.Built +
##     Overall.Qual + Overall.Cond + Neighborhood + MS.SubClass
##
##                 Df Sum of Sq     RSS     AIC
## <none>                        9.9147 -3600.5
## - Bedroom.AbvGr  1    0.1274 10.0421 -3591.8
## - MS.SubClass   14    1.3238 11.2385 -3523.9
## - log(Lot.Area)  1    1.0195 10.9342 -3520.8
## - Neighborhood  27    2.3614 12.2762 -3476.3
## - Year.Built     1    1.7030 11.6177 -3470.3
## - Overall.Qual   1    2.0510 11.9658 -3445.6
## - Overall.Cond   1    2.6540 12.5687 -3404.6
## - log(area)      1    6.9942 16.9089 -3157.3
```

Chunk 23

```
summary(model.full.aic)
```

```
##
## Call:
## lm(formula = log(price) ~ log(area) + log(Lot.Area) + Bedroom.AbvGr +
##     Year.Built + Overall.Qual + Overall.Cond + Neighborhood +
##     MS.SubClass, data = normal.sales)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.78815 -0.06308  0.00051  0.06851  0.43625
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -3.114707   0.827157  -3.766 0.000179 ***
## log(area)            0.591912   0.025137  23.547  < 2e-16 ***
## log(Lot.Area)        0.118954   0.013232   8.990  < 2e-16 ***
## Bedroom.AbvGr       -0.023640   0.007438  -3.178 0.001540 **
## Year.Built           0.004647   0.000400  11.619  < 2e-16 ***
## Overall.Qual         0.070730   0.005547  12.751  < 2e-16 ***
## Overall.Cond         0.059564   0.004106  14.505  < 2e-16 ***
## NeighborhoodBlueste  0.064318   0.084266   0.763 0.445530
## NeighborhoodBrDale  -0.044239   0.069019  -0.641 0.521731
## NeighborhoodBrkSide -0.017578   0.056059  -0.314 0.753941
## NeighborhoodClearCr -0.003054   0.061887  -0.049 0.960652
## NeighborhoodCollgCr -0.081142   0.049666  -1.634 0.102714
## NeighborhoodCrawfor  0.052538   0.055518   0.946 0.344279
## NeighborhoodEdwards -0.132857   0.051885  -2.561 0.010635 *
## NeighborhoodGilbert -0.097357   0.052038  -1.871 0.061731 .
## NeighborhoodGreens   0.152986   0.071973   2.126 0.033848 *
## NeighborhoodGrnHill  0.299657   0.091164   3.287 0.001058 **
## NeighborhoodIDOTRR  -0.141877   0.057234  -2.479 0.013387 *
## NeighborhoodLandmrk  0.033751   0.124669   0.271 0.786673
## NeighborhoodMeadowV -0.086824   0.060921  -1.425 0.154496
## NeighborhoodMitchel -0.036457   0.051398  -0.709 0.478346
## NeighborhoodNAmes   -0.056270   0.050968  -1.104 0.269926
## NeighborhoodNoRidge  0.046025   0.052679   0.874 0.382554
## NeighborhoodNPkVill  0.032766   0.072955   0.449 0.653459
## NeighborhoodNridgHt  0.092558   0.049314   1.877 0.060901 .
## NeighborhoodNWAmes  -0.092809   0.052225  -1.777 0.075936 .
## NeighborhoodOldTown -0.102731   0.055414  -1.854 0.064132 .
## NeighborhoodSawyer  -0.071763   0.052203  -1.375 0.169619
## NeighborhoodSawyerW -0.125015   0.050840  -2.459 0.014147 *
## NeighborhoodSomerst  0.040474   0.049592   0.816 0.414664
## NeighborhoodStoneBr  0.069520   0.055153   1.260 0.207866
## NeighborhoodSWISU   -0.079741   0.064521  -1.236 0.216868
## NeighborhoodTimber  -0.015957   0.055526  -0.287 0.773897
## NeighborhoodVeenker  0.002597   0.060758   0.043 0.965920
## MS.SubClass30        0.015524   0.025667   0.605 0.545491
## MS.SubClass40       -0.350342   0.115514  -3.033 0.002502 **
## MS.SubClass45       -0.018867   0.047209  -0.400 0.689530
## MS.SubClass50       -0.064153   0.020972  -3.059 0.002296 **
## MS.SubClass60       -0.100969   0.014838  -6.805 2.01e-11 ***
## MS.SubClass70       -0.059460   0.030987  -1.919 0.055361 .
## MS.SubClass75       -0.005810   0.050042  -0.116 0.907601
## MS.SubClass80       -0.041027   0.020529  -1.999 0.046002 *  
## MS.SubClass85        0.055647   0.026721   2.083 0.037618 *  
## MS.SubClass90       -0.143180   0.024772  -5.780 1.08e-08 ***
## MS.SubClass120      -0.041541   0.024113  -1.723 0.085324 .  
## MS.SubClass160      -0.174992   0.032790  -5.337 1.24e-07 ***
## MS.SubClass180       0.017711   0.056470   0.314 0.753884    
## MS.SubClass190      -0.020120   0.033200  -0.606 0.544670    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1123 on 786 degrees of freedom
## Multiple R-squared:  0.9184, Adjusted R-squared:  0.9135 
## F-statistic: 188.2 on 47 and 786 DF,  p-value: < 2.2e-16
```

Chunk 24

```
# predict prices and record the residuals
normal.sales$final.price.prediction <- exp(predict(model.full.aic))
normal.sales$final.residuals <- sqrt((normal.sales$final.price.prediction - normal.sales
$price)^2)
final.rmse.train <- mean(normal.sales$final.residuals)
final.rmse.train
```

```
## [1] 14236.25
```

Chunk 25

```
normal.sales.test$final.price.prediction <- exp(predict(model.full.aic,normal.sales.test
))
normal.sales.test$final.residuals <- sqrt((normal.sales.test$final.price.prediction - no
rmal.sales.test$price)^2)
final.rmse.test <- mean(normal.sales.test$final.residuals)
final.rmse.test
```

```
## [1] 15886.83
```

Chunk 26

```
predict.full.aic.train <- exp(predict(model.full.aic, normal.sales, interval = 'predicti
on'))
coverage.full.aic.train <-
  mean(normal.sales$price > predict.full.aic.train[,'lwr'] &
  normal.sales$price < predict.full.aic.train[,'upr'])
coverage.full.aic.train
```

```
## [1] 0.9556355
```

Chunk 27

```
predict.full.aic.test <- exp(predict(model.full.aic, normal.sales.test, interval = 'pred
iction'))
coverage.full.aic.test <-
  mean(normal.sales.test$price > predict.full.aic.test[,'lwr'] &
  normal.sales.test$price < predict.full.aic.test[,'upr'])
coverage.full.aic.test
```

```
## [1] 0.9510404
```

2.3.2 Section 3.2 Transformation

Did you decide to transform any variables? Why or why not? Explain in a few sentences.

I only tranfosformed for variables for this model: log transformed the `price` , `Lot.Area` , and `area` to make them more normal and transformed `MS.SubClass` into a factor from a integer.

2.3.3 Section 3.3 Variable Interaction

Did you decide to include any variable interactions? Why or why not? Explain in a few sentences.

I did not include any variable interactions. The reason is that, through the EDA process, I couldn't find any variables (typically categorical) that had any affect on price that needed to be combined with other variables to improve their affect. If I did, I would've experimented with other combinations and interactions of variables to see if it would increase the adjusted r-squared of the model.

2.3.4 Section 3.4 Variable Selection

What method did you use to select the variables you included? Why did you select the method you used? Explain in a few sentences.

I used the AIC method using the `stepAIC()` function in the MASS package to select the "best" model be recursively removing vairables until the AIC was the lowest. The AIC method only checks the model against like models so it doesn't prove absolute model quality, but with my initial adjusted r-squared of 91% I felt confident that the AIC method would effectively improve the variable combination in the model.

2.3.5 Section 3.5 Model Testing

How did testing the model on out-of-sample data affect whether or how you changed your model? Explain in a few sentences.

I ran the model on both the training and testing data set and compared the RMSE for both to see how well the model fit an out-of-sample data set. The model's predictions yielded a RMSE of $14,236.25 on the training set (Chunk 24) and $15,846.48 on the testing set (Chunk 25) which is a tighter margin of error than the initial model RMSE of $18,475.21 (Chunk 17) so I didn't change the final model at all. I then compared the coverage of the model on both the testing and training data set and that analysis yielded expected results at 95.6% and 95.1%, respectively, as detailed in Chunk 26 and 27.

2.4 Part 4 Final Model Assessment

2.4.1 Section 4.1 Final Model Residual

For your final model, create and briefly interpret an informative plot of the residuals.

Figure 28

```
qqplot(normal.sales$final.residuals, log(normal.sales$price))
```

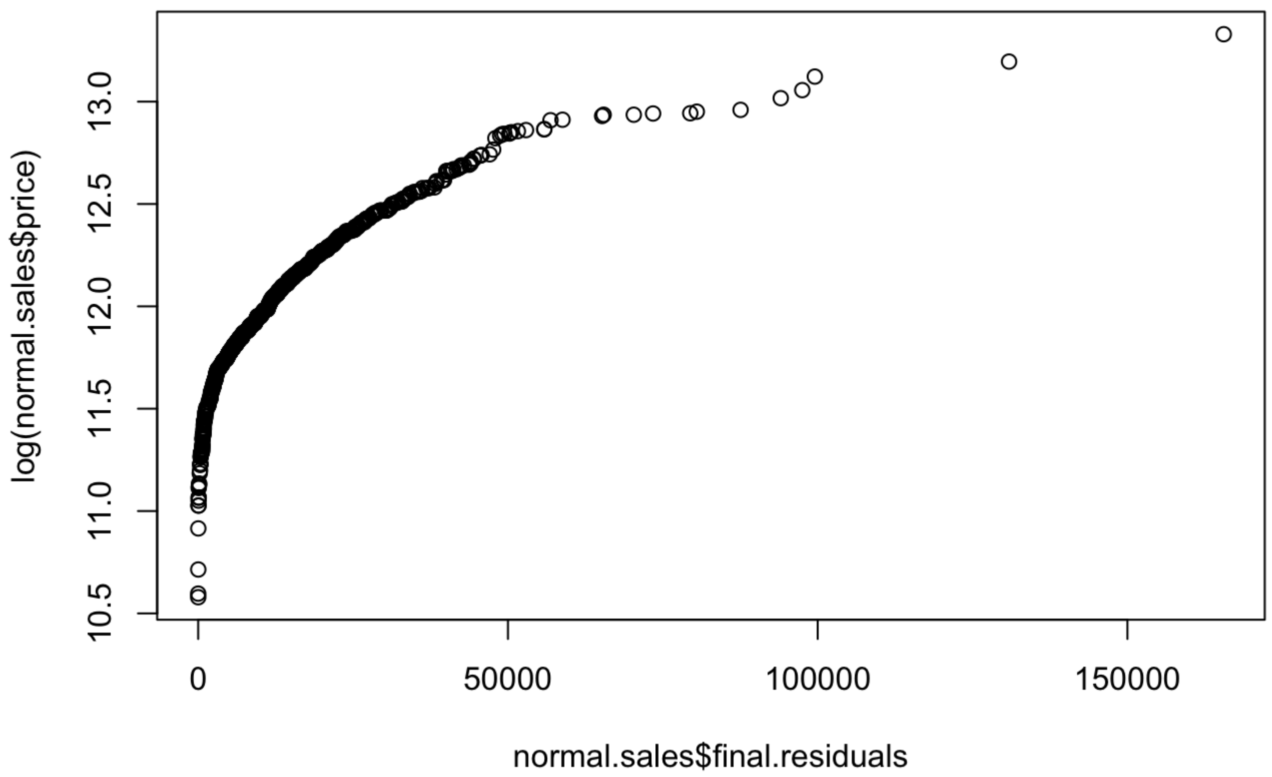


Figure 29

```
qqplot(normal.sales.test$final.residuals, log(normal.sales.test$price))
```

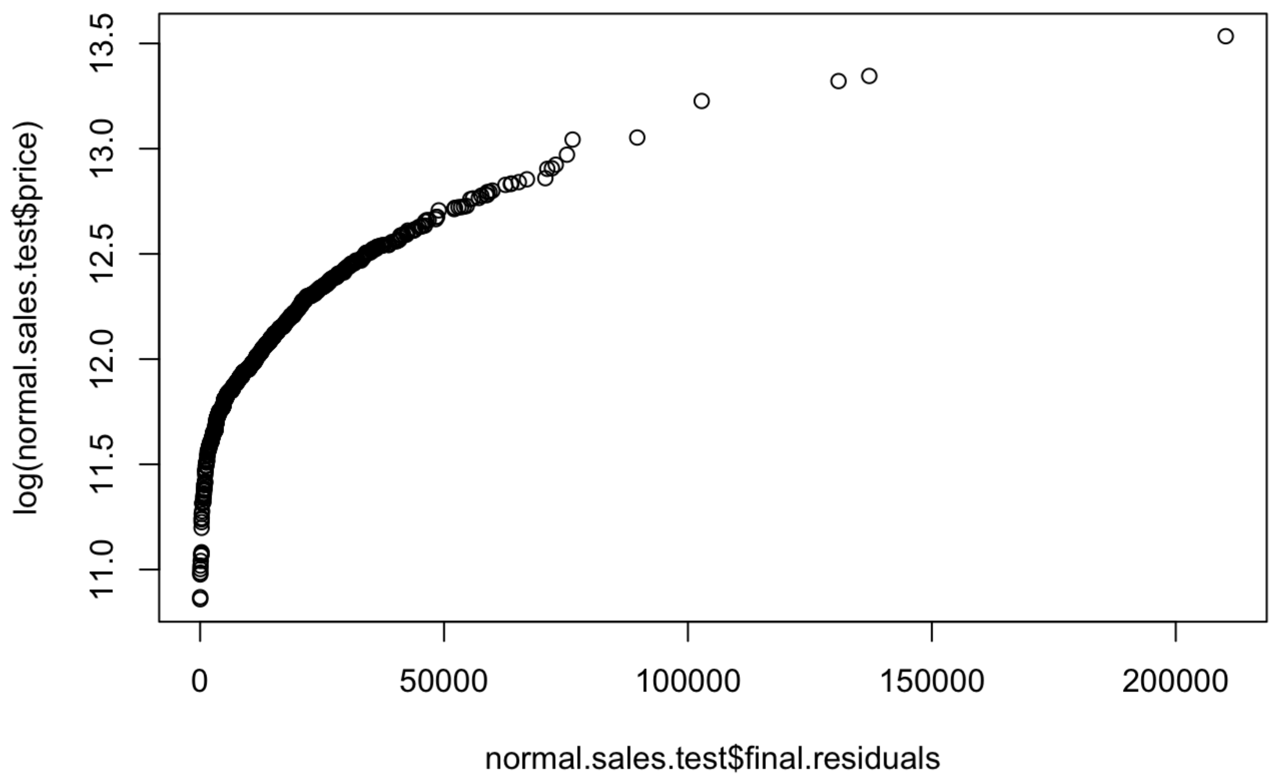
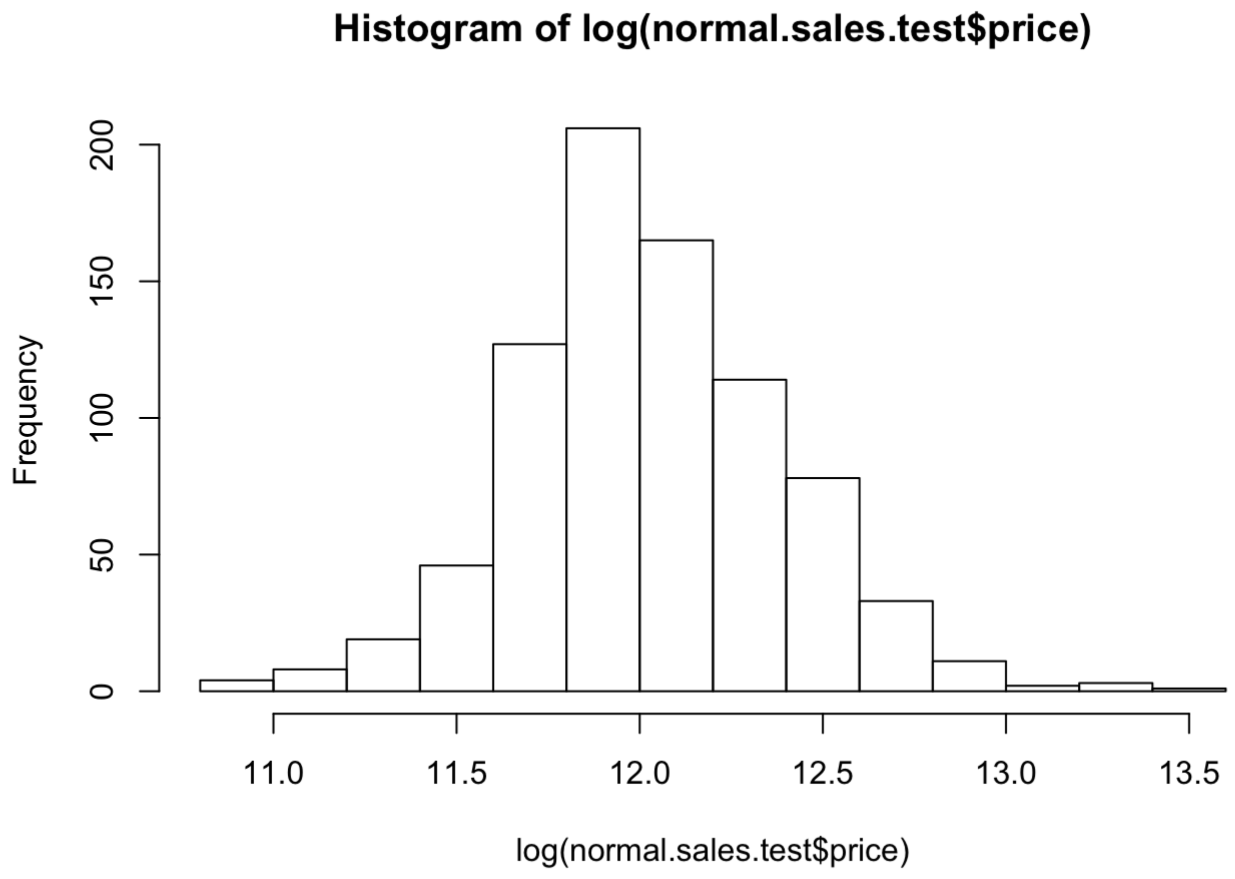


Figure 29a

```
hist(log(normal.sales.test$price))
```



Figures 28 and 29 show the results of the residuals against the log of the prices and it appears that the model's accuracy decreases as the price inceases. This would make sense since the price of a home could be inflated for a multitude of reasons that aren't correlative to other aspects of the home. That said, since the majority of the log prices are in the 11.5 to 12.5 range as depicted by Figure 29a, it appears that the model works well with the majority of the observations in both the training and testing data sets.

2.4.2 Section 4.2 Final Model RMSE

For your final model, calculate and briefly comment on the RMSE.

As mentioned in Section 3.5, the final model's predictions yielded a RMSE of $14,236.25 on the training set (Chunk 24) and $15,846.48 on the testing set (Chunk 25) which is a tighter margin of error than the initial model RMSE of $18,475.21 (Chunk 17).

2.4.3 Section 4.3 Final Model Evaluation

What are some strengths and weaknesses of your model?

Two strengths of my model are its accuracy for predicting home prices for the majority of homes in the population (with the limited scope of Ames, IA) and its composition of variables that can be found in most homes. This means that the model is simple and parsimonious and can be applied accurately to the majority of homes in Ames. A weakness of my model is that, as the price increases, the model becomes less accurate and can yield some predicted prices that are much lower than the actual price. This shouldn't be too much of an issue for this project however since homes that would warrant prices that are higher in the price histogram wouldn't make good investments for the firm.

2.4.4 Section 4.4 Final Model Validation

Testing your final model on a separate, validation data set is a great way to determine how your model will perform in real-life practice.

You will use the "ames_validation" dataset to do some additional assessment of your final model. Discuss your findings, be sure to mention: * What is the RMSE of your final model when applied to the validation data?
* How does this value compare to that of the training data and/or testing data? * What percentage of the 95% predictive confidence (or credible) intervals contain the true price of the house in the validation data set?
* From this result, does your final model properly reflect uncertainty?

When using the `ames_validation` data set, the model yielded an RMSE of $14,811.53 (Chunk 31). This compares to an RMSE of $14,236.25 on the training set (Chunk 24) and $15,846.48 on the testing set (Chunk 25) which means that the model does a good job of predicting home prices across multiple data sets. When comparing coverage and "uncertainty" with this model across the data sets, the model yields a coverage of 95.7% (95.7% of predictions fall within the 95% confidence interval of the data set) (Chunk 32) which is a bit higher than either of the training or testing data set but it still means that the model has little significant uncertainty. In other words, the model properly relects uncertainty for any given data set.

Chunk 30

```
load("ames_validation.Rdata")
ames_validation$MS.SubClass <- factor(ames_validation$MS.SubClass)
normal.sales.validation <- ames_validation %>%
  filter(Sale.Condition == 'Normal')
```

Chunk 31

```
normal.sales.validation$final.price.prediction <- exp(predict(model.full.aic,normal.sale
s.validation))
normal.sales.validation$final.residuals <- sqrt((normal.sales.validation$final.price.pre
diction - normal.sales.validation$price)^2)
final.rmse.validate <- mean(normal.sales.validation$final.residuals)
final.rmse.validate
```

```
## [1] 14811.53
```

Chunk 32

```
predict.full.aic.validation <- exp(predict(model.full.aic, normal.sales.validation, inte
rval = 'prediction'))
coverage.full.aic.validation <-
  mean(normal.sales.validation$price > predict.full.aic.validation[,'lwr'] &
  normal.sales.validation$price < predict.full.aic.validation[,'upr'])
coverage.full.aic.validation
```

```
## [1] 0.9567497
```

2.5 Part 5 Conclusion

Provide a brief summary of your results, and a brief discussion of what you have learned about the data and your model.

It was a fun process to start with an EDA, find some variables that one would think would affect a home’s price, build a model and use diagnostics to evalaute it, and finally arrive at a “best fit” model and use out-of-sample data sets to determine how well the model performs. I’m happy that the model performed well in the end (by analyzing the results of the predictions on the `ames_validation` data set) but I learned mostly about how different distributions and data types affect a model’s predictive behavior and how to objectively diagnose and evaluate a model to make it better or to, at least, speak articulately to why a model isn’t performing well.
