

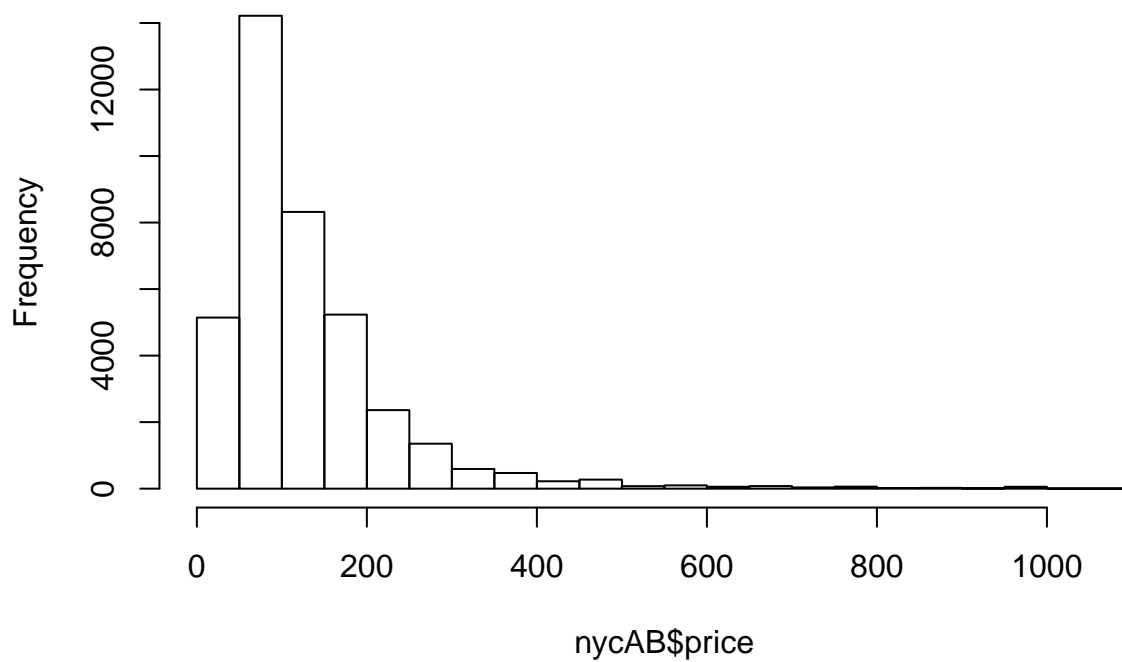
Code Output

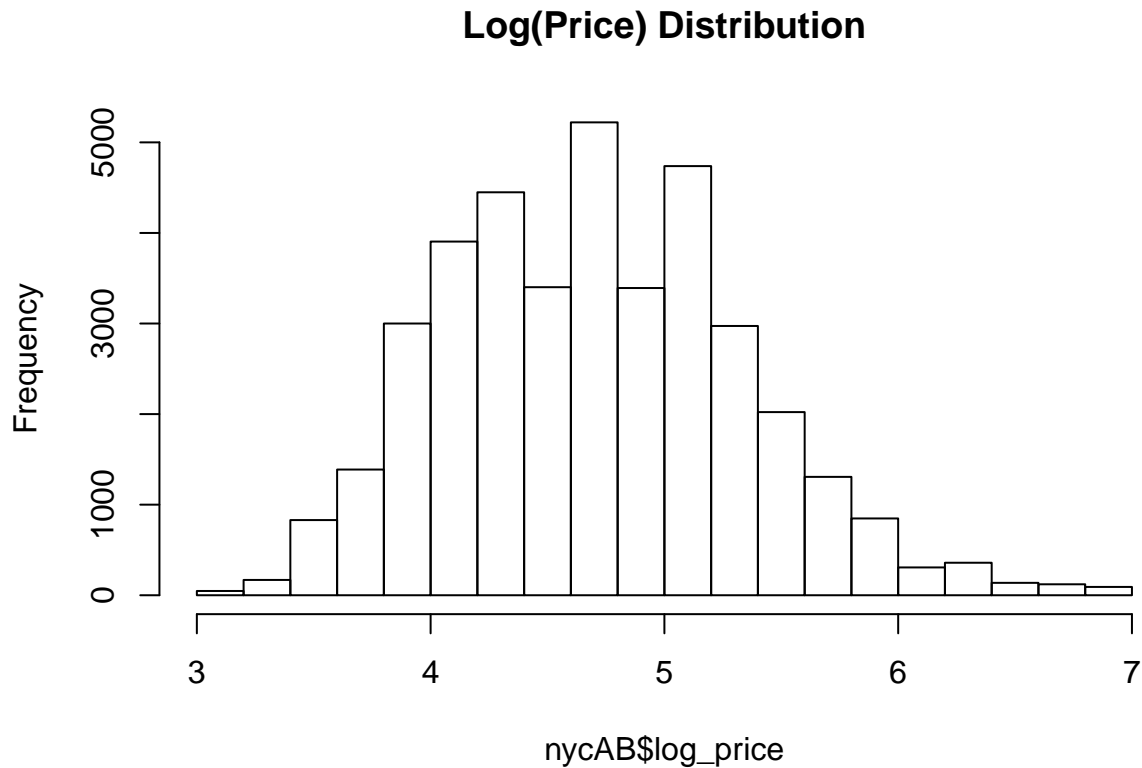
Jadyn Gonzalez, Ariana Bucio, Matt Mead, Bryce Viorst

12/10/2019

We are trying to predict price. Upon inspecting the original 'price' variable, we found a very skewed distribution. We used a log transformation to get a closer to normal distribution of price.

Price Distribution





Our first model we used was a lasso model to attempt to find the best variables to select in other models. We ran the model and found the best value for lambda.

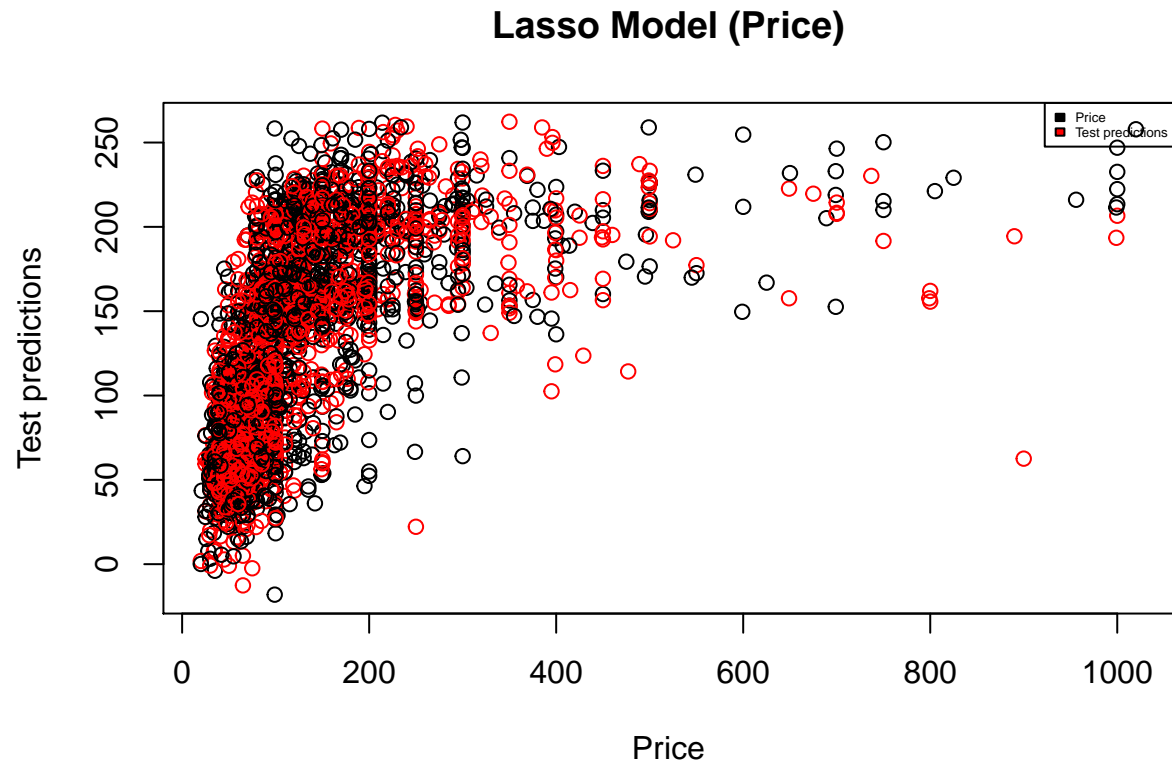
```
## [1] 0.05948865
```

Once we found lambda, we re-trained the model then found the RMSE and generated our plots.

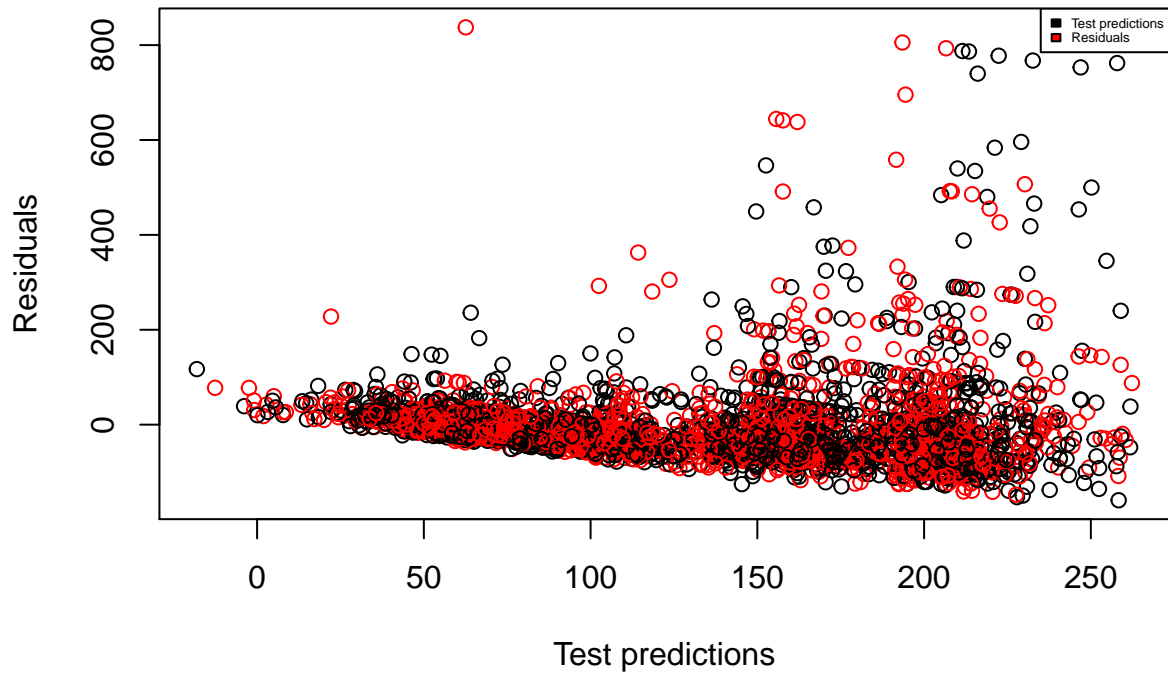
```
## RMSE_train_LASS01 RMSE_test_LASS01
## 1 92.16207 95.21181

## 16 x 1 sparse Matrix of class "dgCMatrix"
## s0
## (Intercept) 81.45209427
## neighbourhood_group1 -28.15757481
## neighbourhood_group2 .
## neighbourhood_group3 45.88488602
## neighbourhood_group4 -18.84415477
## neighbourhood_group5 -39.23124813
## room_type1 101.38941322
## room_type2 .
## room_type3 -33.95420207
## minimum_nights -0.06708542
## number_of_reviews -0.07109291
## reviews_per_month -3.71106781
## availability_365 0.13984070
```

```
## log_num_reviews      -6.20032411
## reviews_per_month_log 13.98975966
## minimum_nights_log    -11.87091041
```



Lasso Model (Price)



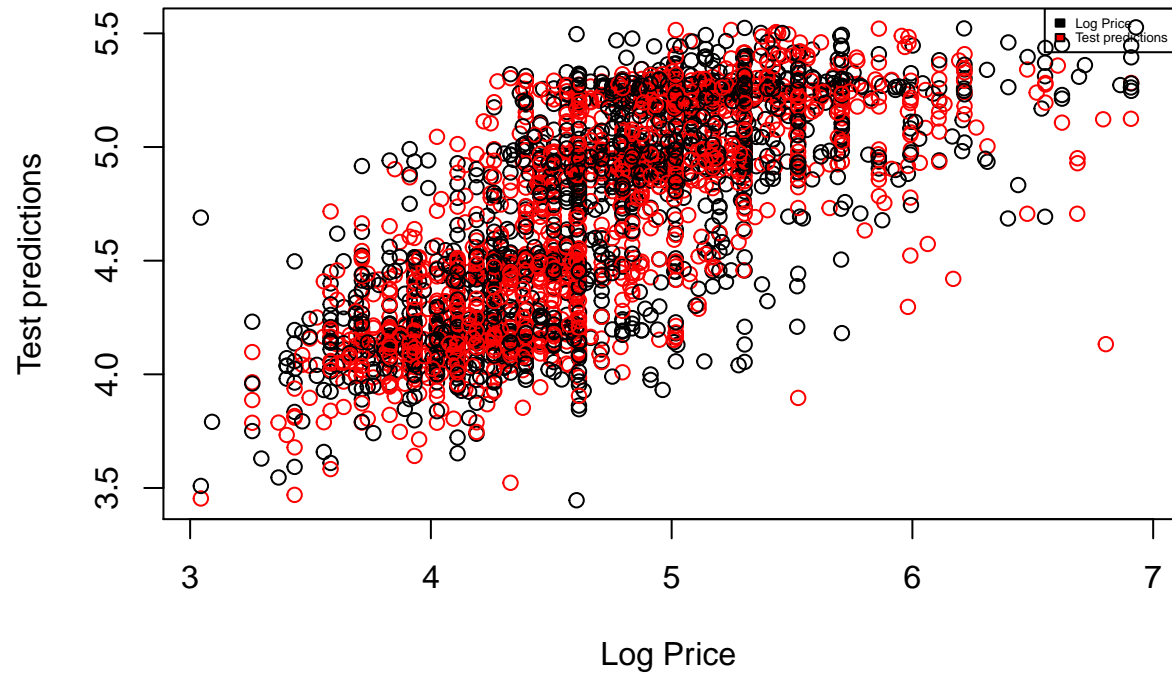
Now we did the same steps using the log of price as the output variable.

```
## [1] 0.0003843989

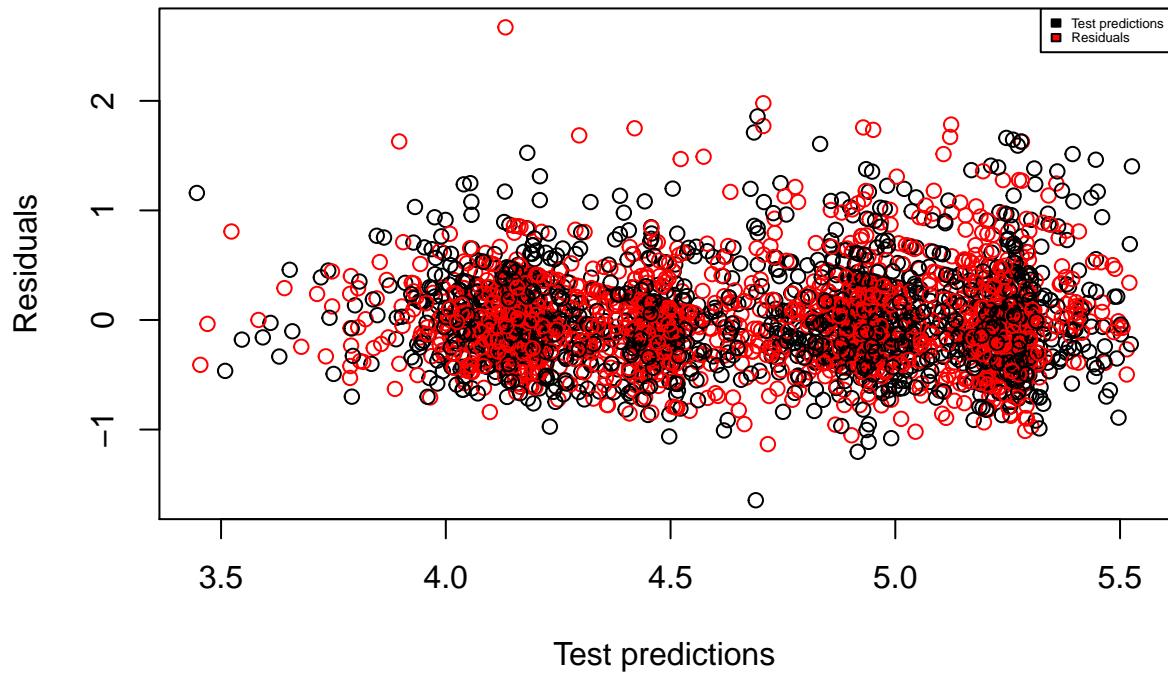
## RMSE_train_LASSO2 RMSE_test_LASSO2
## 1 1.556107 1.550764

## 16 x 1 sparse Matrix of class "dgCMatrix"
## s0
## (Intercept) 4.2804399581
## neighbourhood_group1 -0.2708510635
## neighbourhood_group2 .
## neighbourhood_group3 0.3072219089
## neighbourhood_group4 -0.1609740474
## neighbourhood_group5 -0.3083413599
## room_type1 0.7829308254
## room_type2 .
## room_type3 -0.4114584451
## minimum_nights .
## number_of_reviews -0.0004145433
## reviews_per_month -0.0193414123
## availability_365 0.0006935239
## log_num_reviews -0.0161582488
## reviews_per_month_log 0.0566029825
## minimum_nights_log -0.0893523178
```

Lasso Model (Log Price)



Lasso Model (Log Price)

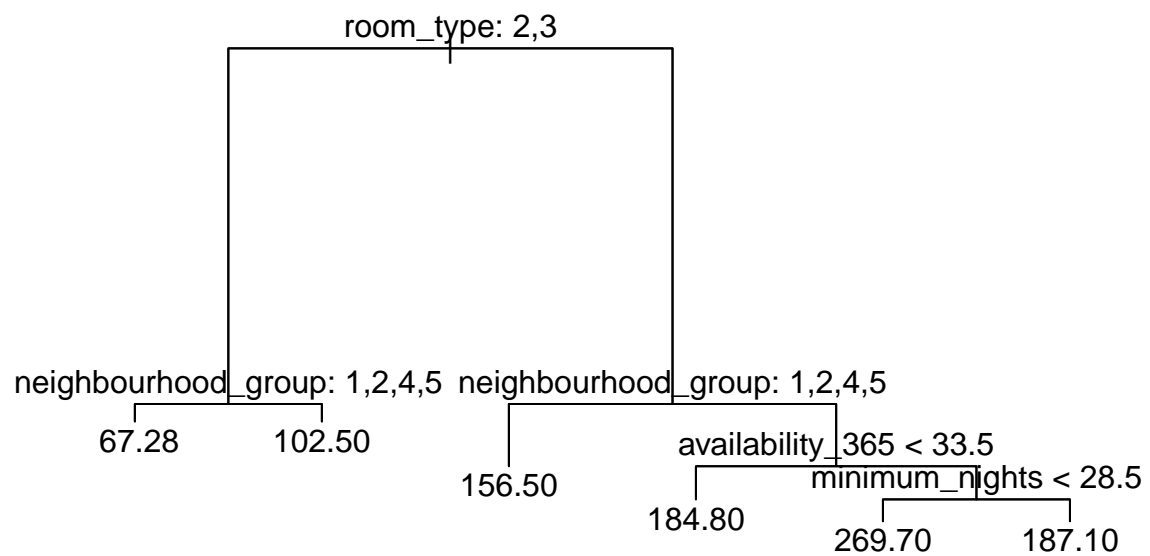


```
##           X1 LogPrice
## 22034 4.495465 4.330733
## 34496 4.504848 3.713572
## 37086 4.442489 3.931826
## 23454 4.044292 4.110874
## 17881 4.117792 4.060443
## 36337 4.529334 4.394449
```

Our second model choice was a decision tree.

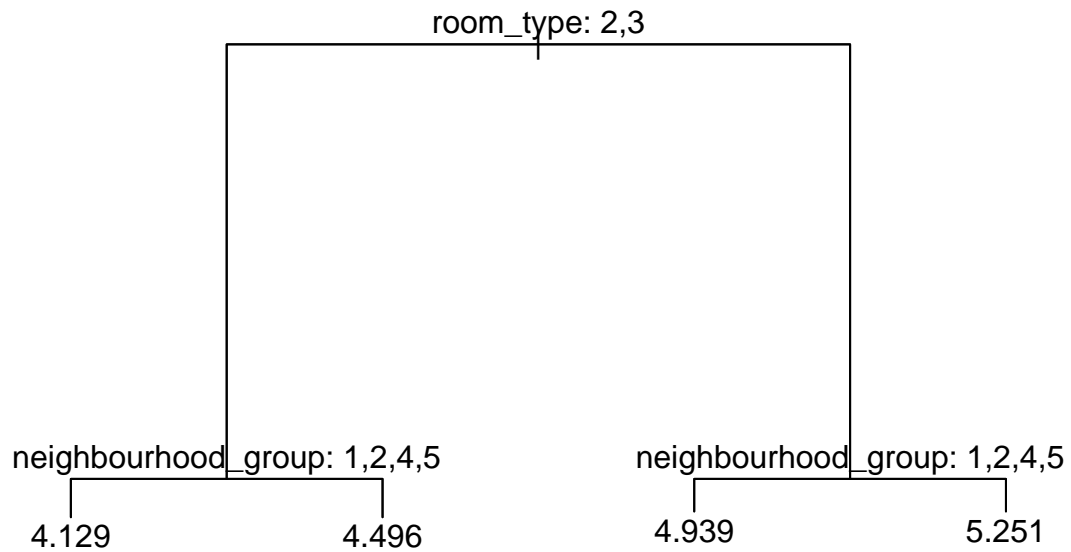
The tree output for 'price'.

```
plot(regMod)
text(regMod, pretty=0)
```



The tree output for 'log_price'.

```
plot(logMod)
text(logMod, pretty = 0)
```



We found the best tree size for 'price' and 'log_price' respectively.

```
## [1] 6
```

```
## [1] 4
```

Then we compared RMSE. Since the model appeared to be overfit, we continued to the random forest model.

```
## [1] "RMSE train v test of 'price'."
```

```
## [1] 92.59612
```

```
## [1] 96.96846
```

```
## [1] "RMSE train v test of 'log_price'."
```

```
## [1] 1.583465
```

```
## [1] 1.586467
```

The output for the random forest model.


```
##
## Call:
##  randomForest(formula = log_price ~ neighbourhood_group + room_type +      availability_365 + log_nu
##              Type of random forest: regression
##              Number of trees: 500
## No. of variables tried at each split: 5
##
##              Mean of squared residuals: 0.1875766
##              % Var explained: 54.12
```

We compared RMSE and found that random forest showed the best results.

```
## [1] "RMSE test"
```

```
## [1] 1.533759
```

```
## [1] "RMSE train"
```

```
## [1] 1.493944
```