Submission 2 Regression Model

Ariana Bucio, Jadyn Gonzalez, Matt Mead, Bryce Viorst

11/20/2019

We are going to run a regression model against price and log_price in our dataset using a decision tree. First we need to clean the dataset and do our log transformations.

```
setwd("~/Documents/MGSC310-Project-master")
nycAB = read.csv("AB_NYC_2019.csv")
nycAB = nycAB[ , !(names(nycAB) %in% c("name","id","host_name", "latitude", "longitude", "last_review",
nycAB[is.na(nycAB)] = 0
nycAB$host_id = as.factor(nycAB$host_id)

nycAB$log_price = log1p(nycAB$price)
nycAB = nycAB[!(nycAB$log_price < 3),]
nycAB = nycAB[!(nycAB$log_price > 7),]

nycAB$log_num_reviews = log1p(nycAB$number_of_reviews)
nycAB$reviews_per_month_log = log1p(nycAB$reviews_per_month)
nycAB$minimum_nights_log = log1p(nycAB$minimum_nights)
```

Now we'll split the data into training and test sets for modeling.

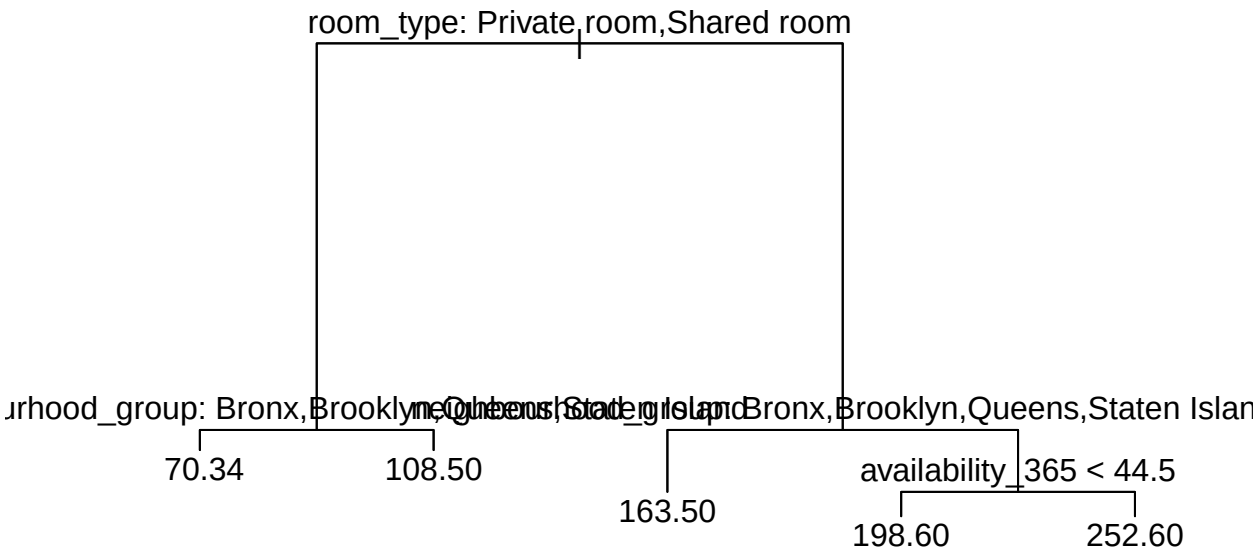
```
set.seed(310)
trainidx = sample(1:nrow(nycAB),size=0.75*nrow(nycAB))
train = nycAB[trainidx,]
test = nycAB[-trainidx,]
```

We assume that the model using the log transformed price will perform the best but we'll run two; one using the regular price and one using the log of price and compare their performance.

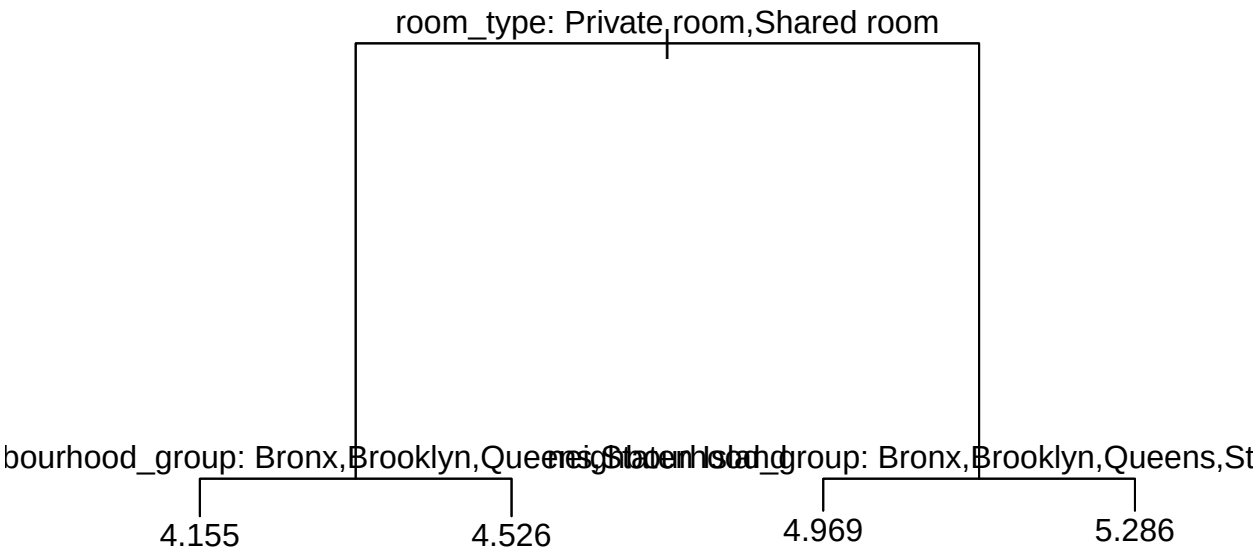
```
library(tree)
regMod = tree(price ~ neighbourhood_group + room_type
              + minimum_nights + number_of_reviews + reviews_per_month_log
              + availability_365 + log_num_reviews + reviews_per_month_log
              + minimum_nights_log,
              data = train)

logMod = tree(log_price ~ neighbourhood_group + room_type
              + minimum_nights + number_of_reviews + reviews_per_month_log
              + availability_365 + log_num_reviews + reviews_per_month_log
              + minimum_nights_log,
              data = train)
```

The plots of both are as follows.

```
# We'll plot both
plot(regMod)
text(regMod, pretty=0)
```




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



In both trees we can see that the variable that affects price the most is room type, more specifically if the room type is the whole house or not. The next most important node for both trees is neighbourhood_group, if it is Manhattan or not. The model using log_price only gives these nodes. The model using the given price variable includes a node if the room is the whole house and is in manhattan it looks at availability.

Now we need to use cross validation to find the best tree size for pruning.

```
cvTreeR = cv.tree(regMod)
bestIdx = which.min(cvTreeR$dev)
cvTreeR$size[bestIdx]
```

```
## [1] 5
```

```
# Best size is 5

cvTreeL = cv.tree(logMod)
bestIdx = which.min(cvTreeL$dev)
cvTreeL$size[bestIdx]
```

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

```
# Best size is 4
```

Since we have our best tree sizes, we can prune each tree and generate predictions.

```
prunedTreeR = prune.tree(regMod, best = 5)
predsTrainR = predict(prunedTreeR)
predsTestR = predict(prunedTreeR, newdata = test)

prunedTreeL = prune.tree(logMod, best = 4)
predsTrainL = predict(prunedTreeL)
predsTestL = predict(prunedTreeL, newdata = test)
```

We'll output the MSE for each model.

```
MSE = function(p,t){
  mean((t-p)^2)
}

MSE(predsTrainR, train$price) #9975.12
```

```
## [1] 9990.492
```

```
MSE(predsTestR,test$price) #9570.264
```

```
## [1] 9516.046
```

```
MSE(predsTrainL, train$log_price) #0.231
```

```
## [1] 0.2312964
```

```
MSE(predsTestL, test$log_price) #0.228
```

```
## [1] 0.2258194
```

The MSE is much lower when we use the log transformation. We do get a lower MSE in the test set which may be an indication that the model is overfitting the data.