ML Prediction – Final Draft

Presentation by:

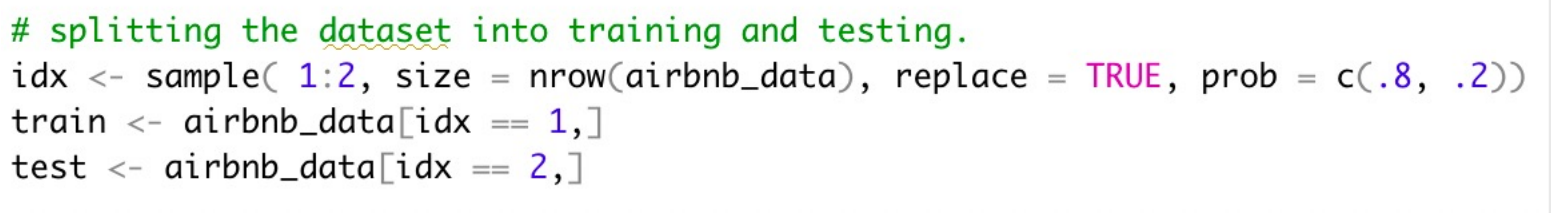
Muskan Burman

Data Science Capstone

Developing a prediction model

I started by randomly dividing by training data set into training and testing data – around 80% for training and 20% for testing.

I started with Random Forest, and initially included all the attributes, just to get a basic idea of the relationship of price with different attributes.



```
# splitting the dataset into training and testing.
idx <- sample( 1:2, size = nrow(airbnb_data), replace = TRUE, prob = c(.8, .2))
train <- airbnb_data[idx == 1,]
test <- airbnb_data[idx == 2,]
```

Developing a prediction model

Once I had an initial mse, I decided to start removing attributes in an attempt to improve my model. From the previous EDA of data, and the generated plots, I narrowed my attributes down to:

neighbourhood.group +

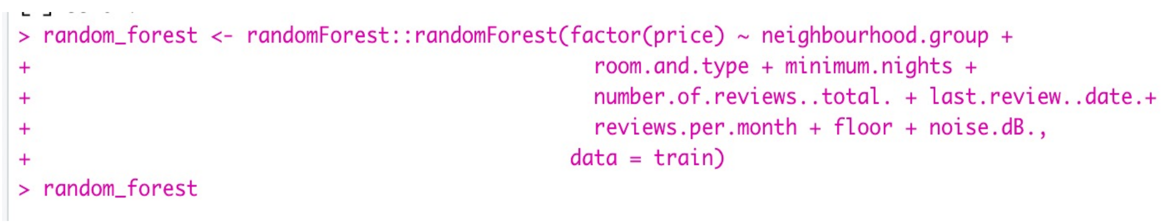
room.and.type + minimum.nights +

number.of.reviews..total. + last.review..date.+

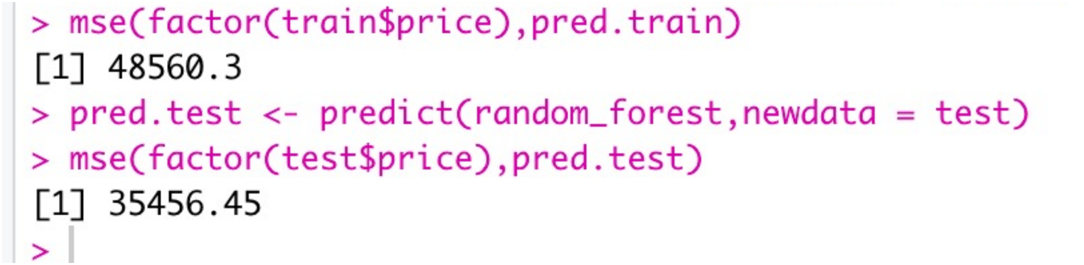
reviews.per.month + floor + noise.dB.

Developing a prediction model

- This combination of attributes resulted in producing an mse of 48560.3 for the training data, and an mse of 35456.45 for the testing portion of the data.
- I then tried using other ML models such as SVM, Linear Regression and even Rpart, but none of them resulted in a better mse and error percentage.
- Thus, I decided to stick to RF.
- I tried adding other attributes as well, however, they didn't improve the model.



```
> random_forest <- randomForest::randomForest(factor(price) ~ neighbourhood.group +
+                                              room.and.type + minimum.nights +
+                                              number.of.reviews..total. + last.review..date.+
+                                              reviews.per.month + floor + noise.dB.,
+                                            data = train)
> random_forest
```

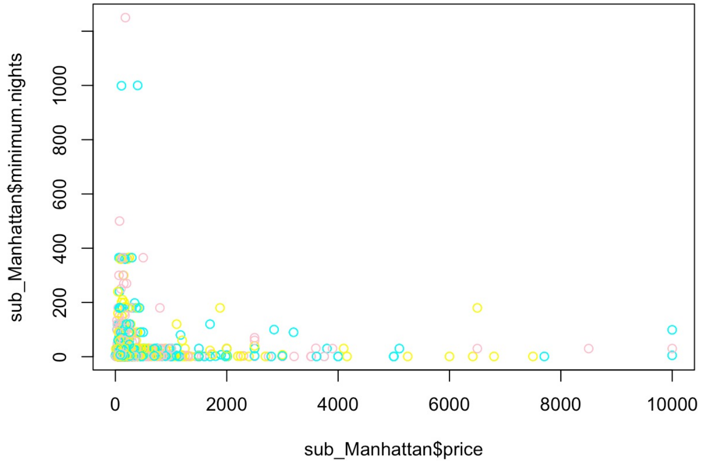


```
> mse(factor(train$price),pred.train)
[1] 48560.3
> pred.test <- predict(random_forest,newdata = test)
> mse(factor(test$price),pred.test)
[1] 35456.45
> 
```

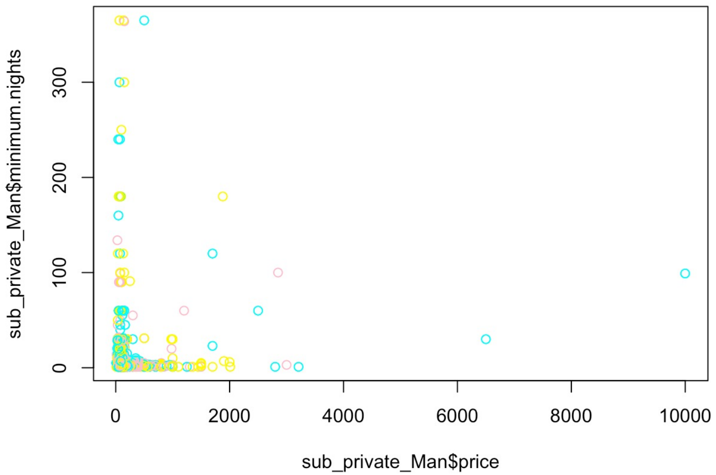
Incorporation of POIs

- There is still a lot of area of improvement for this model, thus I tried using some feature engineering by introducing some POIs in the data and their relation to the price.
- I looked at Airbnbs near the 9/11 memorial, Central Park, Empire State Building, Statue of Liberty, Madison Square Garden, and the JFK Airport, and developed plots to find the relationship between different attributes.

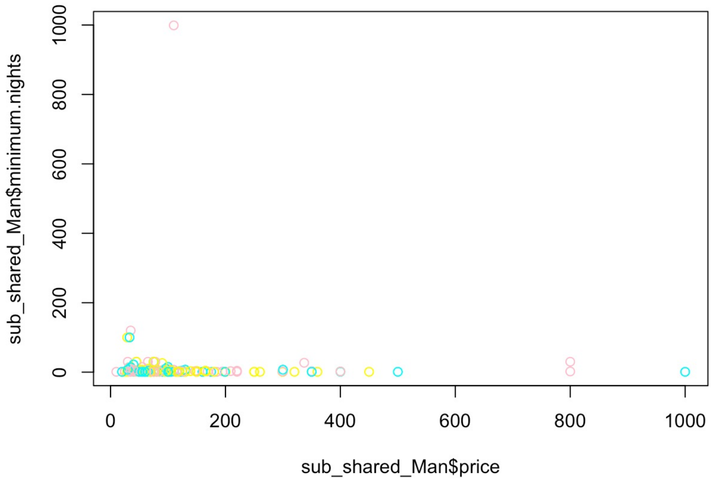
Price vs min nights in Manhattan



Places that cost less usually have lower number of minimum nights.



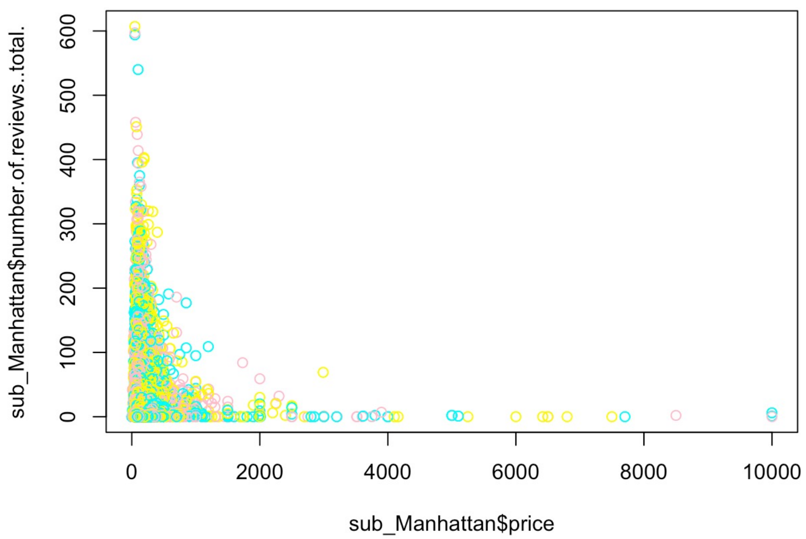
While the observations remain same while looking specifically at private rooms in Manhattan, we observe many outliers here.



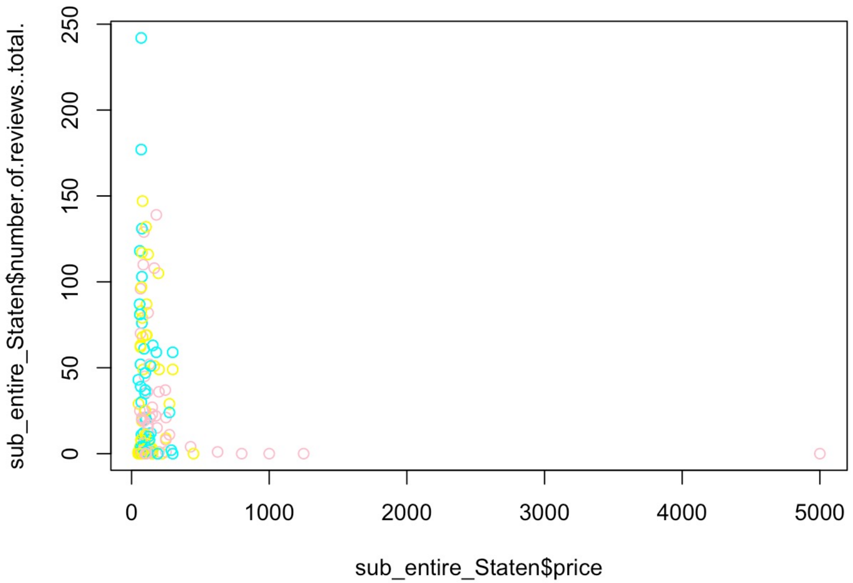
However, when we specifically look at all the shared spaces in Manhattan, we can see that regardless of price, the minimum nights remain the same.

This is a general trend with the relationship between price and minimum nights of a place, for other neighbourhood groups as well.

Price vs Total num of reviews - Manhattan



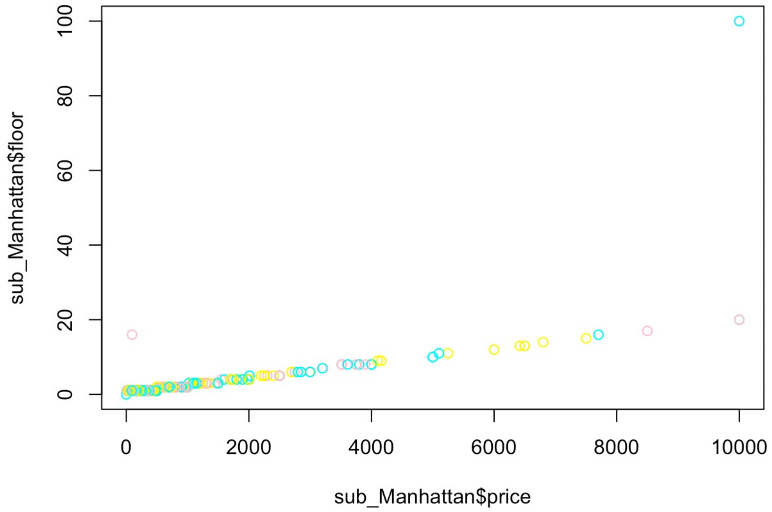
Places that cost less usually have a higher number of total reviews. This might be a result of the fact that since these places are more affordable, more people are able to get them, leading to a greater number of reviews.



Similar is the case for entire houses in Staten Island.

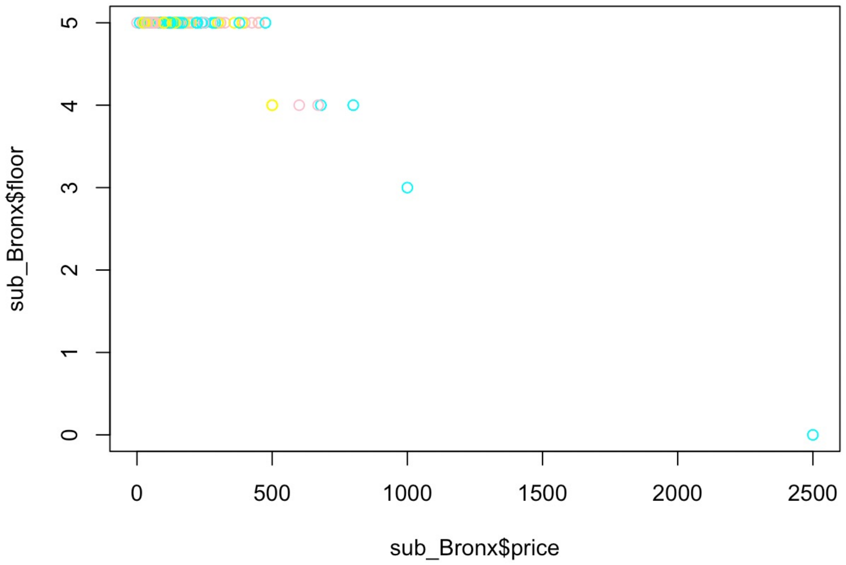
This is a general trend with the relationship between price and total number of reviews of a place, for other neighbourhood groups as well.

Price vs Floor - Manhattan

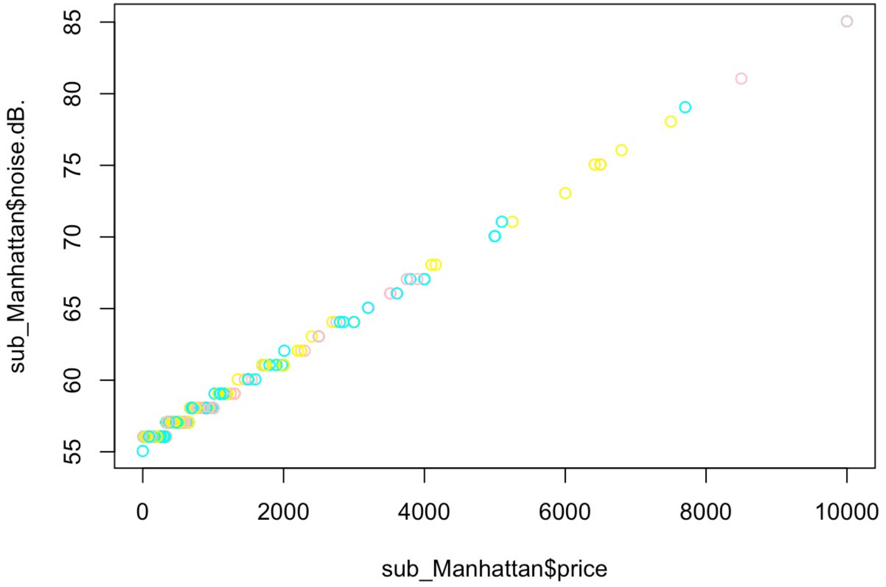


Here, we see a linear relationship between the price and the floors, indicating that the price increases as the level of floor increases – the higher the floor, the higher is the price.

This however is not the case with Bronx. As we can see, most of the Airbnbs in Bronx are on the fifth floor, with a few outliers, thus giving us a very vague relationship bw floor and prices in Bronx.

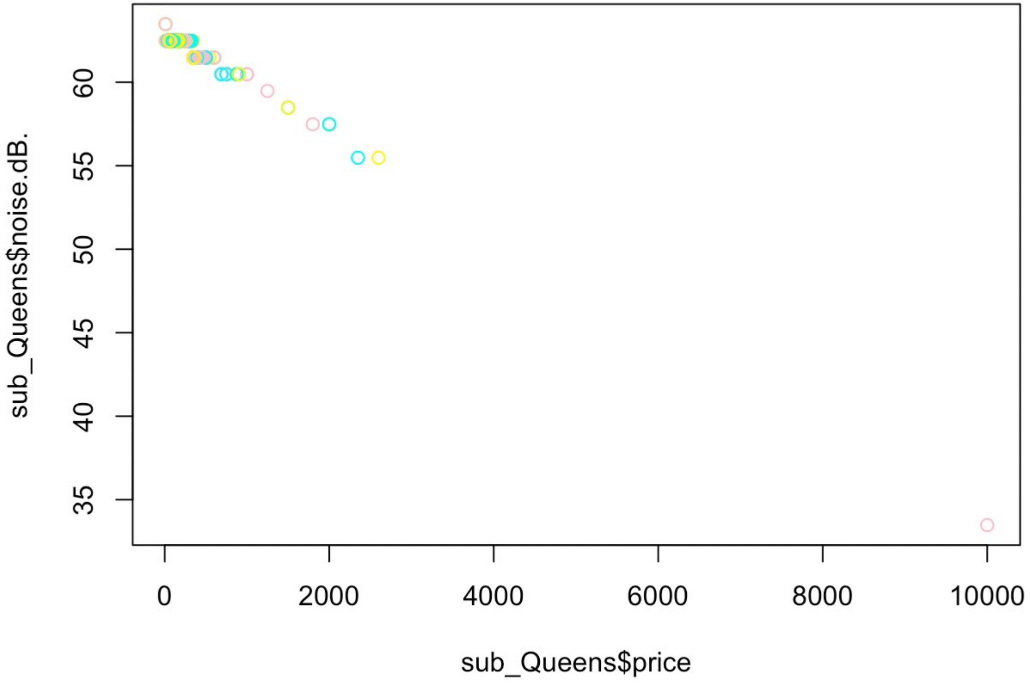


Price vs Noise - Manhattan



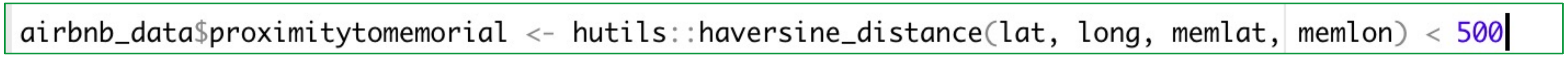
Here, we again see a linear relationship between the price and the noise levels, indicating that the price increases as the noise levels do, which although odd, makes sense in a city like Manhattan where everything is very lively.

Such is not the case in Queens, because prices increases as noise levels decrease.



Incorporation of POIs

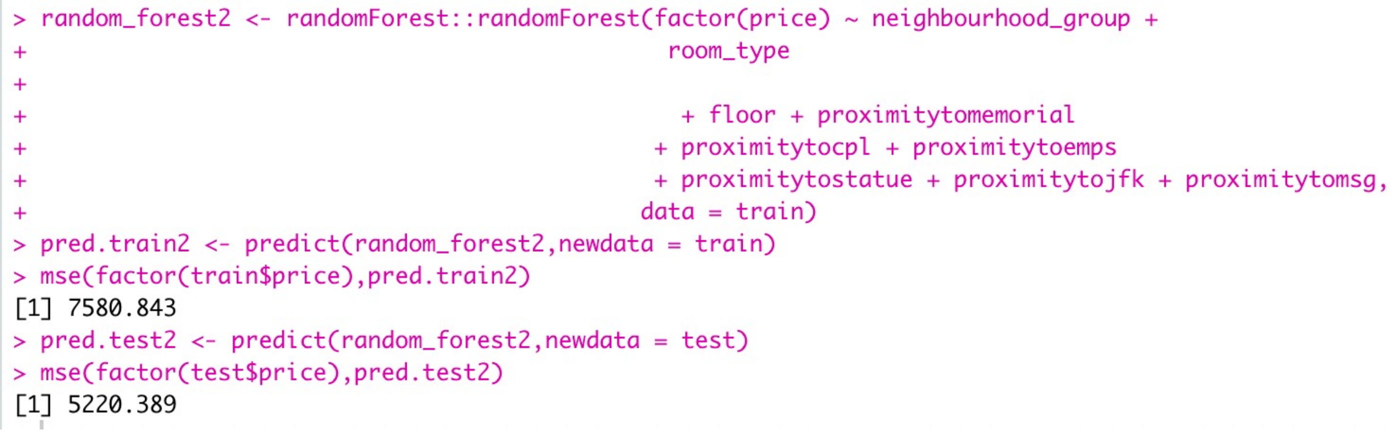
- Idea: To check the proximity of every single listing to the above-mentioned 6 POIs, and see if that affects the accuracy of my model.
- Implementation: To do so, I used the hutils package and the haversine_distance function. I made 2 lists of all the values of the latitudes and longitudes respectively, and then used the lats and longs of the different POIs to compare them, and created 6 new columns in my dataset with True or False values, representing if a particular listing is within 500m from a POI.



```
airbnb_data$proximitytomemorial <- hutils::haversine_distance(lat, long, memlat, memlon) < 500
```

Incorporation of POIs

- Result: Using the above implementation, I added all new 6 columns to my Random Forest Model, and got an mse of 7580.843 for the training data, and an mse of 5220.389 for the testing portion of the data, which was a improvement from the older mses.

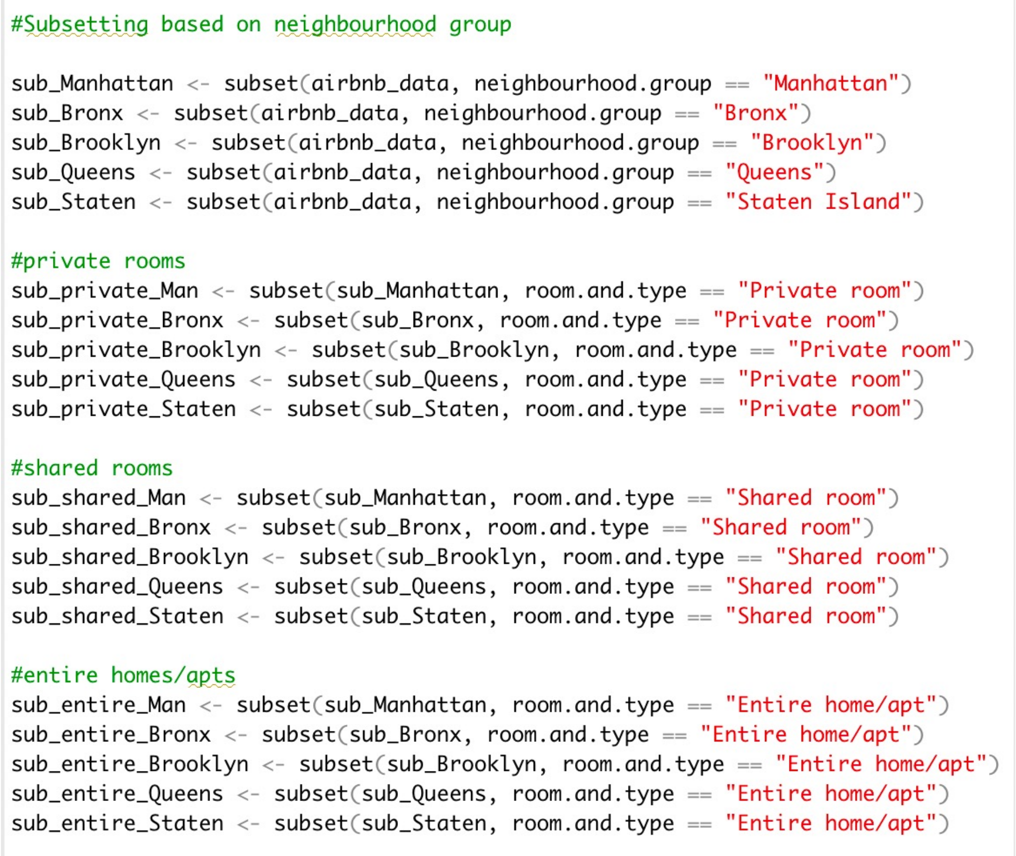


```
> random_forest2 <- randomForest::randomForest(factor(price) ~ neighbourhood_group +
+                                              room_type
+
+                                                + floor + proximitytomemorial
+                                              + proximitytocpl + proximitytoemps
+                                              + proximitytostatue + proximitytojfk + proximitytomsg,
+                                             data = train)
> pred.train2 <- predict(random_forest2,newdata = train)
> mse(factor(train$price),pred.train2)
[1] 7580.843
> pred.test2 <- predict(random_forest2,newdata = test)
> mse(factor(test$price),pred.test2)
[1] 5220.389
```

- Problems: Even though these new mses were an improvement from the older ones, I had to remove many attributes like minimum nights, noise, etc. because of the large size of the dataset resulting in implementation errors.

Subsetting the dataset

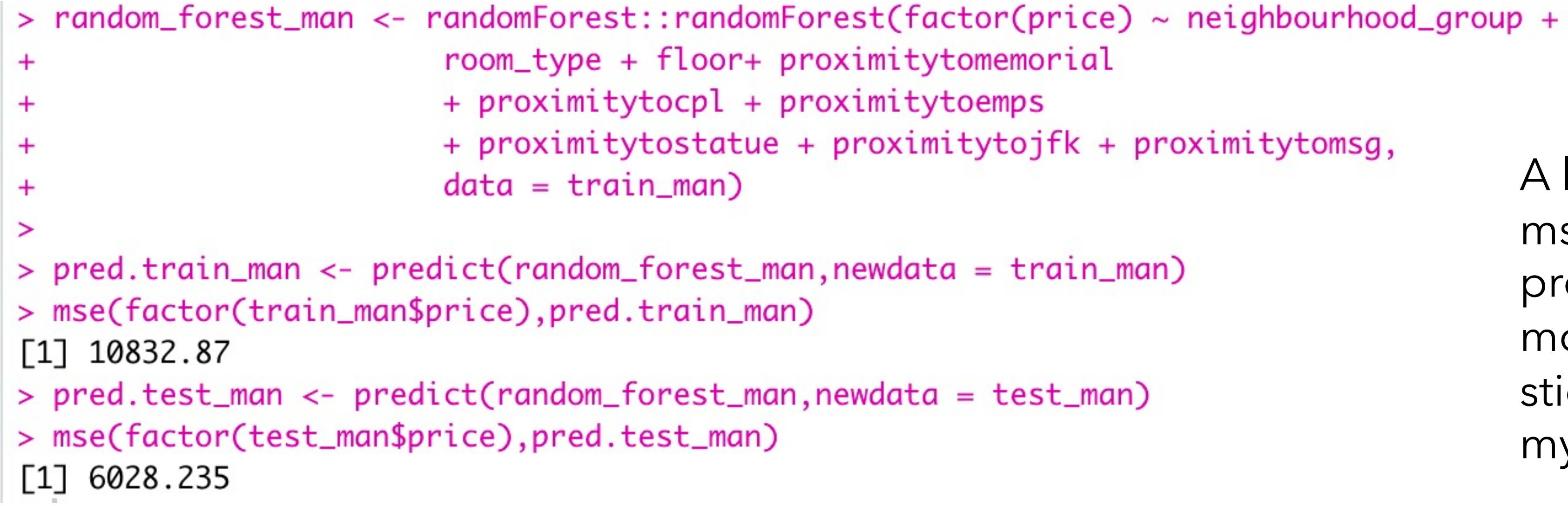
I decided to use the subsets I made during the EDA of my data to look at the relationships of the various attributes with price. I subsetted the dataset first based on the neighbourhood group, resulting in 5 different sets, and then further based on the room type, resulting in 15 more sets to get a total of 20. I then used these subsets, as required, to make separate prediction models to see if I can improve my model further.



```
#Subsetting based on neighbourhood group

sub_Manhattan <- subset(airbnb_data, neighbourhood.group == "Manhattan")
sub_Bronx <- subset(airbnb_data, neighbourhood.group == "Bronx")
sub_Brooklyn <- subset(airbnb_data, neighbourhood.group == "Brooklyn")
sub_Queens <- subset(airbnb_data, neighbourhood.group == "Queens")
sub_Staten <- subset(airbnb_data, neighbourhood.group == "Staten Island")

#private rooms
sub_private_Man <- subset(sub_Manhattan, room.and.type == "Private room")
sub_private_Bronx <- subset(sub_Bronx, room.and.type == "Private room")
sub_private_Brooklyn <- subset(sub_Brooklyn, room.and.type == "Private room")
sub_private_Queens <- subset(sub_Queens, room.and.type == "Private room")
sub_private_Staten <- subset(sub_Staten, room.and.type == "Private room")

#shared rooms
sub_shared_Man <- subset(sub_Manhattan, room.and.type == "Shared room")
sub_shared_Bronx <- subset(sub_Bronx, room.and.type == "Shared room")
sub_shared_Brooklyn <- subset(sub_Brooklyn, room.and.type == "Shared room")
sub_shared_Queens <- subset(sub_Queens, room.and.type == "Shared room")
sub_shared_Staten <- subset(sub_Staten, room.and.type == "Shared room")

#entire homes/apts
sub_entire_Man <- subset(sub_Manhattan, room.and.type == "Entire home/apt")
sub_entire_Bronx <- subset(sub_Bronx, room.and.type == "Entire home/apt")
sub_entire_Brooklyn <- subset(sub_Brooklyn, room.and.type == "Entire home/apt")
sub_entire_Queens <- subset(sub_Queens, room.and.type == "Entire home/apt")
sub_entire_Staten <- subset(sub_Staten, room.and.type == "Entire home/apt")
```

Developing a prediction model (contd.)

I made 5 different prediction models using the same attributes for each of the 5 subsets. The Manhattan subsetted produced the best results of the 5, resulting in an mse of 10832.87 for the training dataset and an mse of 6028.235 for the testing portion.



```
> random_forest_man <- randomForest::randomForest(factor(price) ~ neighbourhood_group +
+                       room_type + floor+ proximitytomemorial
+                       + proximitytocpl + proximitytoemps
+                       + proximitytostatue + proximitytojfk + proximitytomsg,
+                       data = train_man)
>
> pred.train_man <- predict(random_forest_man,newdata = train_man)
> mse(factor(train_man$price),pred.train_man)
[1] 10832.87
> pred.test_man <- predict(random_forest_man,newdata = test_man)
> mse(factor(test_man$price),pred.test_man)
[1] 6028.235
```

A little surprisingly, these mses were more than those produced by older prediction model. Thus, I decided to stick to random_forest_2 for my submission.

Plan of action for further improvement

On submitting my model to Kaggle, I didn't get a very good score, indicating that there is still a huge scope of improvement for my prediction model. This might be because of the following problems that I encountered while working on my model:

1) I don't believe my implementation of the incorporation of POIs is the most effective one. I tried improving it using various methods including the use of geosphere package and making separate lists and trying to compare them, but my R Session kept aborting.

2) The size of the dataset, and implementation problems associated with it. Even after subsetting my dataset, I kept getting an error regarding not being able to use attributes with 53+ categories, which resulted in me removing some attributes.

How I plan on fixing these:

1) I plan on trying new methods to incorporate POIs into my dataset and model.

2) I plan on finding ways to get rid of the implementation errors by trying correction tools or dividing my subsets further into smaller ones and combining all models.