# Final Project Report (AirBNB Data)

*Jacob Burke*

*11/12/2019*

## AirBNB Data Analysis

## Introduction

The purpose of this analysis and project is to visualize and draw conclusions of the tendencies that AirBNB listings generally follow, in primarily the eastern side of the USA. This report will summarize EDA and EFA that will be run on the combined data of AirBNB listings in Boston, New York, Philadelphia, Washington, Nashville, Miami, and Houston. In addition, for the scenario in which someone would like to easily query AirBNB listings based on mutliple traits, I will look to experiment with building a RSQLite data base for the combined city's listing data.

The data from each city was first cleaned and combined, and the specific code that was used in this process can be found in the Appendix of this report. A preview of the AirBNB listing's data and variable values:

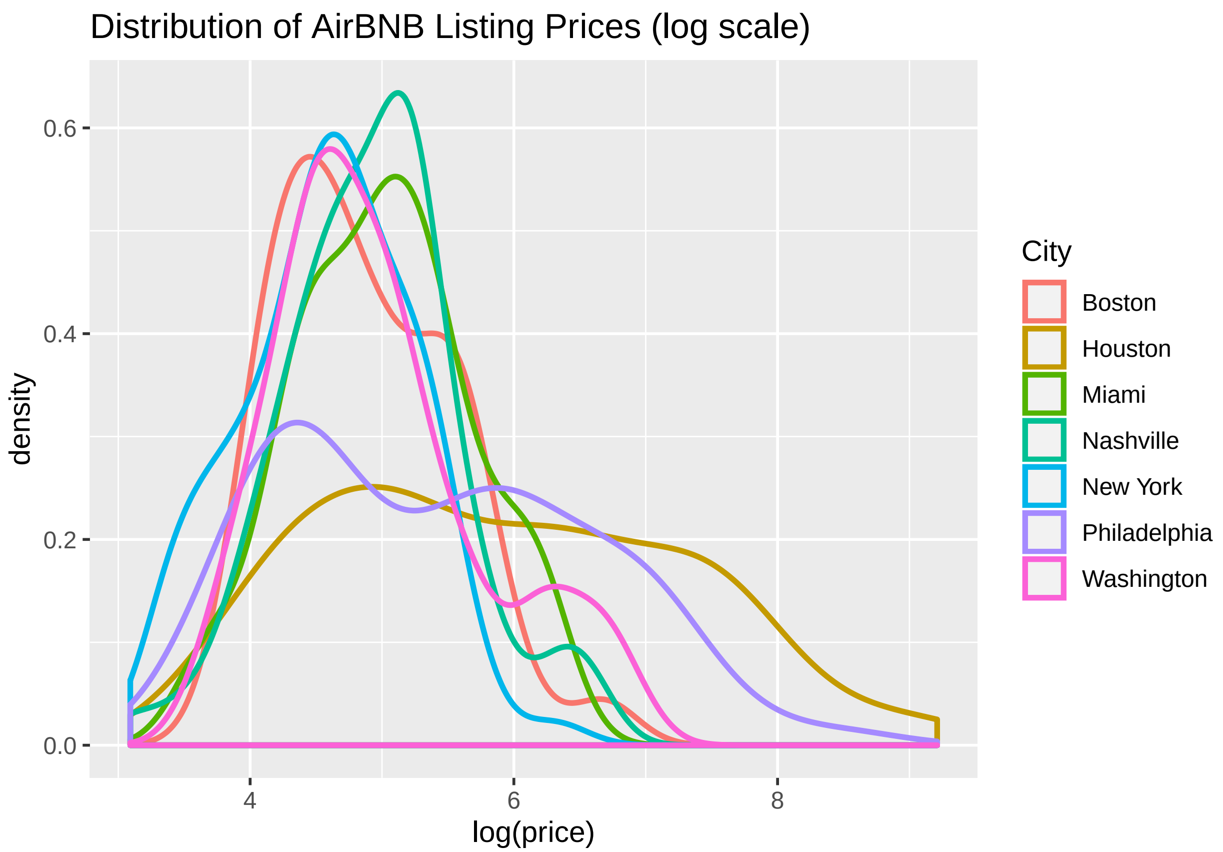
```r
head(BNB)
```

```
##   X  room_id  host_id       room_type reviews accommodates bedrooms price
## 1 1 12784064  3137257 Entire home/apt      28            5        2   200
## 2 2  3755609 19223882    Private room      22            1        1    72
## 3 3   990668  1651480    Private room     112            2        1    93
## 4 4  3968797  4762495    Private room      30            2        1    85
## 5 5  2915643 14891491 Entire home/apt       1            4        1   289
## 6 6 17909550  1118374 Entire home/apt       0            3        0   197
##   latitude longitude              last_modified   City
## 1 42.36467 -71.05576 2017-04-08 15:58:44.097503 Boston
## 2 42.27564 -71.12480 2017-04-08 15:12:31.579145 Boston
## 3 42.34146 -71.07641 2017-04-08 15:19:30.478553 Boston
## 4 42.29111 -71.13134 2017-04-08 15:24:35.994515 Boston
## 5 42.34954 -71.07849 2017-04-08 16:04:58.840221 Boston
## 6 42.35127 -71.12563 2017-04-08 15:58:44.977045 Boston
```

---

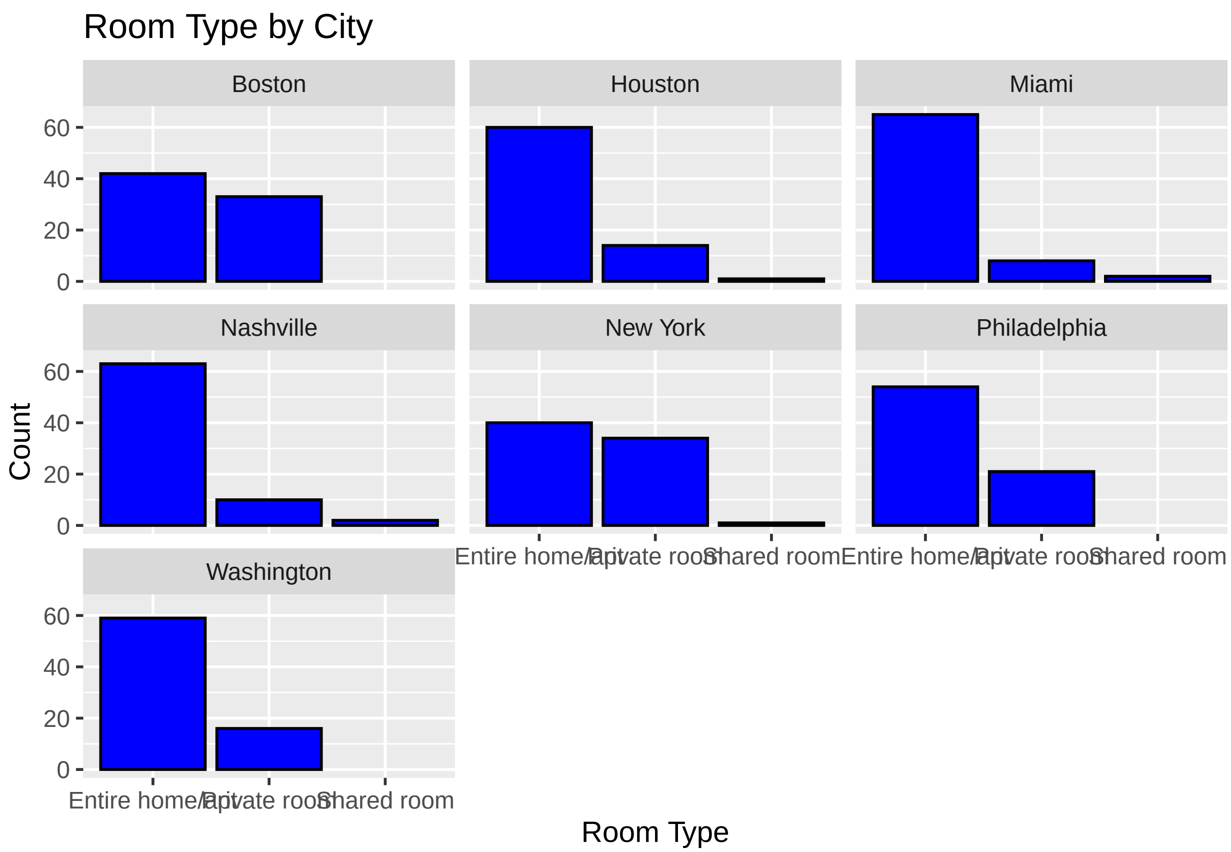
## Initial EDA

For initial exploration of the data, there are a few different ways we should proceed. Firstly, it will be interesting to see how the prices are distributed amongst the different cities.



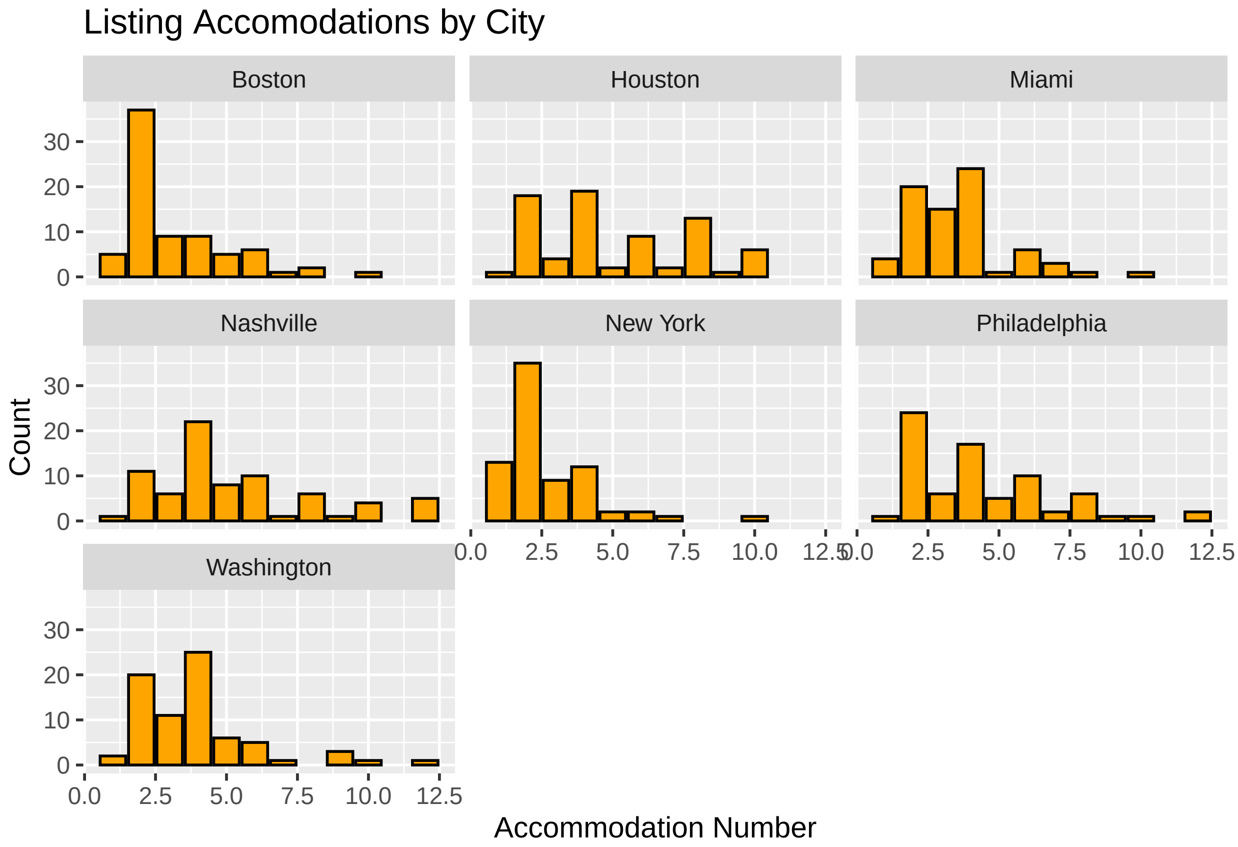
Here we can see that the majority of listing prices amongst all cities lie between *[exp(4) - exp(5.5)]*, ie. within approximately *[55 - 245]* dollars/night. Houston and Philly have the largest variance and spread in prices overall, and the Houston data set contains the most expensive listings out of all cities.

In addition, we should look at the distribution of room types amongst cities.



Overall amongst our entire 7 city data set, we can see that there are more ‘Entire home/apt’ listings than shared and private rooms combined. Boston and New York are the cities with the closest to equal numbers of ‘Entire home/apt’ listings and “Private room” listings.

Finally, lets take a look at the distribution of accommodation numbers (number of people that a listing sleeps) across all cities.



Here we can see that Boston and New Yor have the highest concentration of listings below 2.5 people accommodating. This helps explain our previous plot, in how we saw that both cities have a similar amount of 'Entire Homes' listings to 'Private Rooms' listings, ie. suggesting lower numbers of accommodations in the Private rooms.

In addition, we can see here that Houston is the city with the largest relatively equal spread of listing accommodations, ranging from around 2-10 people. This helps further explain the larger variance of listing prices for Houston that we saw from the previous price density plot. It also makes sense that Houston was seen to have some of the most expensive AirBNB listings out of all cities, as it has a significant number of listings available that can accommodate for 10 or more people.

---

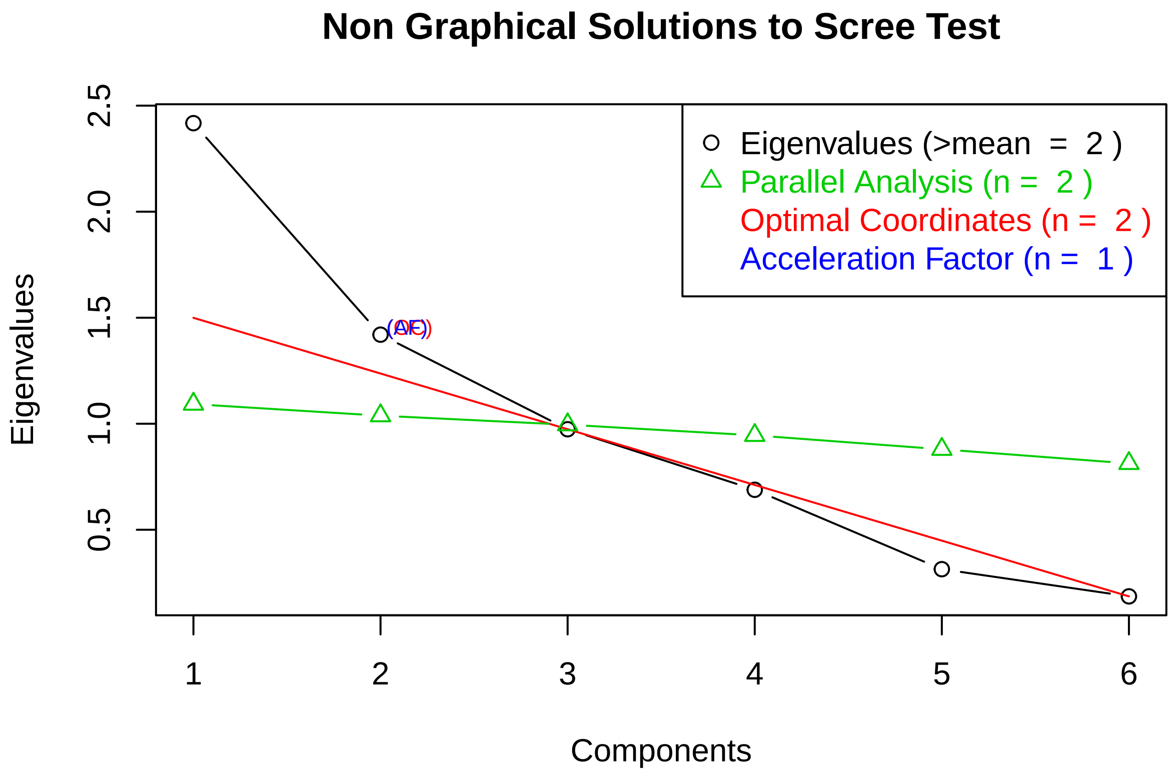
# EFA

For further exploratory analysis, we will run factor analysis on our numerical variable's in our BNB data, to see if we could hypothetically reduce dimensionality. The numerical variables of interest are (accommodates, bedrooms, reviews, price, longitude, latitude).

Since this is 6 variables in total, we will look at the results from combining to 3 components, and look to see how the variables group.

```
##
## Call:
## factanal(x = BNB_factor, factors = 3, rotation = "varimax")
##
## Uniquenesses:
##        price     latitude    longitude accommodates     bedrooms
##         0.71         0.00         0.45         0.00         0.25
##      reviews
##         0.96
##
## Loadings:
##              Factor1 Factor2 Factor3
## accommodates  0.97
## bedrooms      0.73            0.47
## latitude              1.00
## longitude             0.65   -0.31
## price                         0.43
## reviews
##
##                Factor1 Factor2 Factor3
## SS loadings       1.59    1.45    0.58
## Proportion Var    0.26    0.24    0.10
## Cumulative Var    0.26    0.51    0.60
##
## The degrees of freedom for the model is 0 and the fit was 0.0077
```

From this output, we can see that accommodates and bedrooms are fairly correlated. This intuitively makes sense, as both are related to how many people can stay in a listing. If we were in a situation where dimension reduction was required, these could certainly be put into one component. We can also see that lattitude and longitude are related (obvious in the sense of map readings), and price, reviews, and bedrooms have also been placed in the third factor. Now, looking at the scree plot to determine the optimal number of factors to extract from these variables.



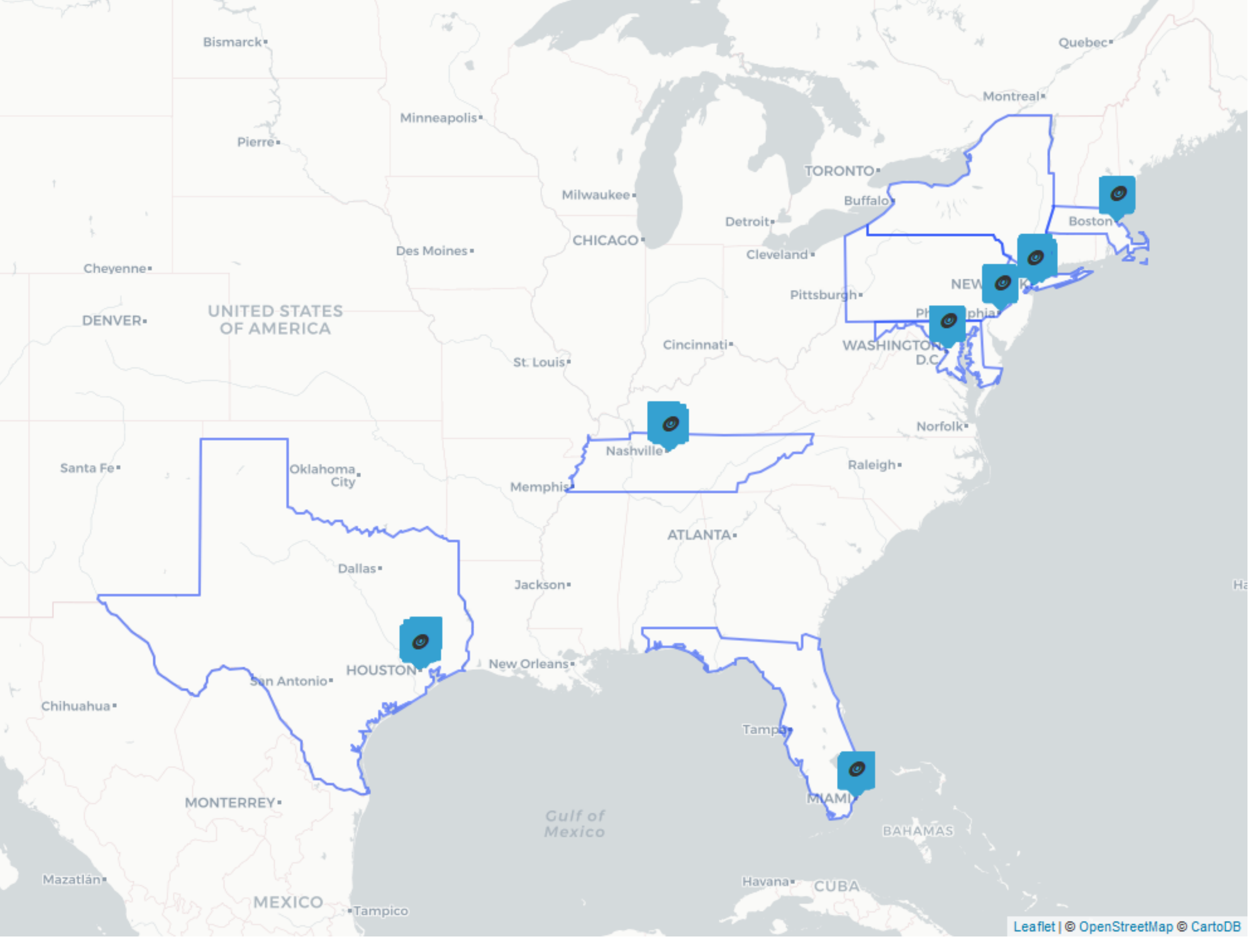
From here we can see that the optimal number of factors for these six variables if we were to reduce dimensions, would in fact be 3.

However, from the 'factanal' output above, we can see that the cumulative variance for these three factors only gets to about 60%, so combining into these factors in this scenario is most likely not optimal, as they will not explain enough of the data.

---

## Mapping

Now, we want to begin to look further into how these listings are arranged in their respective cities proximity-wise.



*As the leaflet package produces html widgets, this is just a screenshot of the map created for output in this pdf file, the interactive leaflet map will also be available in the Shiny app going along with this report.*

---

## SQL Data Base

Now, in the case where we would like to make listings easily accessible through queries, one way we can do this is by building a RSQLite database. For this current BNB data, the database would be relatively simple with only two tables needed. However organizing the data into a database will allow for quick querying, especially for someone not as familiar in R coding, as this allows the querying to then be done in DB browser.

Given that there will be no output in the knitted pdf report for the sqlite database, I'll include the code chunk below to show how the database has been built.

```
library(dplyr)
## Selecting desired tables (listing, and room)

listing <- dplyr::select(BNB, room_id, host_id, latitude, longitude, City)

listing <- distinct(listing)

room <- dplyr::select(BNB, room_id, reviews, accommodates, bedrooms, price, room_type)

room <-  distinct(room)

## Built database, in 'BNB_db.sqlite' file (will be in the Github repo to add to your working directory
## Shiny App usage)

## BNB_db <- dbConnect(SQLite(), "BNB_db.sqlite")

## dbWriteTable(BNB_db, "Listing", listing)
## dbWriteTable(BNB_db, "Room", room)

## dbDisconnect(BNB_db)
```

This data base will be available to run interactive queries within the corresponding Shiny app for this project.

---

# Appendix

Code for cleaning and combining of multiple city AirBNB listing data sets.

```
read <- function(data){

  return(read.csv(data, header = T))
}

## reading in sampled city listing data

Boston <- read("Boston_s.csv")
Houston <- read("Houston_s.csv")
Miami <- read("Miami_s.csv")
Nashville <- read("Nashville_s.csv")
NY <- read("NY_s.csv")
Philly <- read("Philly_s.csv")
Washington <- read("Washington_s.csv")

## adding city attributes, we can then combine all city data for analysis

addCity <- function(df, city){
  return(mutate(df, City = city))
}

Boston <- addCity(Boston, "Boston")
Houston <- addCity(Houston, "Houston")
```

```
Miami <- addCity(Miami, "Miami")
Nashville <- addCity(Nashville, "Nashville")
NY <- addCity(NY, "New York")
Philly <- addCity(Philly, "Philadelphia")
Washington <- addCity(Washington, "Washington")

## dropping columns we won't be using for analysis, so each data frame is of the same value of
## variables, and then we can combine data sets

Boston <- dplyr::select(Boston, -c(minstay, borough, neighborhood, overall_satisfaction))
Houston <- dplyr::select(Houston, -c(survey_id, country, city, borough, minstay, bathrooms, name, locati
                              neighborhood, overall_satisfaction))
Miami <- dplyr::select (Miami, -c(borough, minstay, neighborhood, overall_satisfaction))
Nashville <- dplyr::select(Nashville, -c(survey_id, country, city, borough, bathrooms, minstay, locatio
                                  neighborhood, overall_satisfaction))
NY <- dplyr::select(NY, -c(survey_id, borough, minstay, country, city, name, property_type, location, ba
                    neighborhood, overall_satisfaction))
Philly <- dplyr::select(Philly, -c(borough, minstay, neighborhood, overall_satisfaction))
Washington <- dplyr::select(Washington, -c(survey_id, country, city, borough, bathrooms, minstay, locati
                                    neighborhood, overall_satisfaction))

## Now can combining

BNB <- rbind(Boston, Houston, Miami, Nashville, NY, Philly, Washington)
```