# Analyzing Airbnb Data

Airbnb is a company which runs an online marketplace that lets people rent out their house, apartments, RVs, spare rooms, extra beds, etc. The company was founded in 2008 and has grown considerably since then. According to Airbnb, as of November 2018 there were over 5 million listings in 81,000 cities worldwide.[1]

For this assignment you will look at the following three questions:

- Is there any correlation between the price of a listing and the overall satisfaction?
- Do hosts typically have multiple listings at the same time?
- How do the prices of a rental change over time?

In addition you will write a few paragraphs reflecting on how the code you've written relates to issues in the real world.

## 1 The dataset

To answer these questions we need data. Conveniently, someone has assembled a subset of the immense amounts of data that Airbnb collects and made it available at: `http://tomslee.net/airbnb-data-collection-get-the-data`[2] The website provides many, many csv files from different cities at different times.

You should go to the website, read their description of the data, and download a few datasets to see what these files look like. The starter directory includes the files for Brno, which are what we use in this writeup as an example. *However, be warned that just because your code works on the Brno files does not necessarily mean that it will work on all of the files that we will test your code on! Make sure you read this entire handout and that you understand what assumptions you can and cannot make about the data!* Note that the datasets vary considerably by size (e.g. Linkoping is relatively small; New York is relatively large).

The files all start with a header line followed by a line for each listing. Not every file has the same header line; for example, the file `tomslee_airbnb_aarhus_0627_2016-10-28.csv` does not include columns for `country` and `city`, although most files do. As a result, when you look for the data of relevance for the question that you're answering, you'll want to take a look at the header file to figure out which column contains the relevant data.

For the first three questions, the data of interest is in the columns with the headers:

- `room_id`
- `host_id`
- `room_type`
- `reviews`
- `overall_satisfaction`

[1] https://press.atairbnb.com/fast-facts/, retrieved November 18, 2018.

[2] Airbnb Data Collection: Get the Data by Tom Slee is licensed under CC By-NC 2.5 CA



- `price`

If data is missing *and is relevant for the function you're implementing*, or if a line is irregular for some other reason, you should ignore that line. Examples include:

- An entry (that is relevant to the function you're writing) is empty. For example, even though there is a `country` header in:

  `tomslee_airbnb_brno_1500_2017-07-20.csv`

  there is no data in many of the lines for that column. If you were writing a function that needed to use the value of `country`, then you should ignore those lines.

- In some cases the data in a column will contain a comma. Take a look at

  `tomslee_airbnb_new_york_1438_2017-07-12.csv`

  The first entry has, for the `name` column, the entry:

  ```
  "Room TO SHARE by DAY,week,month"
  ```

  This will cause problems if you try to call `split()` on that line. Even if your function doesn't use that data, your code should identify such lines and ignore them.

  Note/hint: one way to do this would be by counting the number of commas in the line and comparing it against the number that there should be for the given header.

These are just some of the issues that we know you will encounter with the data. There may be others; please check with the professor as you find more edge cases!

# 2 Part I: Correlation between price and satisfaction?

To study whether there is any correlation between the price for a listing and how satisfied occupants are, you should write the following two functions.

## 2.1 Function 1: price_satisfaction(filename)

This function takes a string containing the name of a file. It creates a list of lists (of `float`) where each sublist contains exactly two items: the price and the overall_satisfaction.

One caveat is that some listings may never be rented and others may be rented but never reviewed. It's not perfect (working with data rarely is), but we will handle this situation by only including datapoints where the number of reviews (given by the column with the heading `reviews`) is positive. Note that the number of reviews is not returned in the list of lists; it just determines which price:satisfaction points to include.

For example, given the file `tomslee_airbnb_brno_1500_2017-07-20.csv`, some sublists in the returned list should be:

```
[11.0, 0.0]
[21.0, 4.5]
[19.0, 4.5]
[19.0, 4.5]
[22.0, 0.0]
[14.0, 0.0]
```

### 2.2 Function 2: `correlation(l)`

This function takes a list that is the output of `price_satisfaction` (Function 1) and returns the correlation between the price and the rating. We'll use a Spearman's rank correlation[3] This isn't a stats class, so you don't need to understand anything about how the statistic is computed to implement this function. That said, it's helpful to know that a result of 0 means no relationship, a positive result means that if one variable (e.g. price) increases the other (satisfaction) generally does as well, and a negative result means that if one increases the other generally decreases.

You should import the `scipy` package and include the line:

```
from scipy.stats.stats import spearmanr
```

at the top of your code.

Now, if you are looking for the correlation between two variables, you should create a list consisting of, in this case, all the price data. Then another list consisting of, in this case, all the rating data. The two lists must be in the same order. In other words, `price[i]` and `rating[i]` should contain information for the same listing. Now you can compute the correlation between those two lists using:

```
spearmanr(price,rating)
```

This call to `spearmanr` will return an object of type `SpearmanrResult`. This object has two attributes `correlation` and `pvalue`. Your `correlation` function should return a tuple consisting of `(correlation,pvalue)`

As an example, calling:

```
l = price_satisfaction("tomslee_airbnb_brno_1500_2017-07-20.csv")
print(correlation(l))
```

should give:

[3] `https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient`, retrieved November 19, 2018

```
(0.012991413392543557, 0.7889466867135996)
```

Note/Hint: if `result = spearmanr(price,rating)`, then `correlation = result.correlation` and `pvalue = result.pvalue` will give you the two values that the `price_satisfaction` function should return as a tuple.

# 3 Part II: Multiple listings?

To study whether hosts have multiple listings, you should write the following two functions.

## 3.1 Function 3: host_listings(filename)

This function takes a string containing the name of a file. It creates and returns a dictionary where the keys are `host_id`s (`int`) and the values are a list of `room_id`s (`int`) associated with that host.

For example, given the file `tomslee_airbnb_brno_1500_2017-07-20.csv`, some items in the dictionary should be:

```
41363259: [12036183]
44260343: [16191125, 15831047]
8082029: [12864574, 12079723]
53257320: [10347898]
34115236: [12033092, 10939439, 10980009]
68975698: [12691076]
4504049: [1353870, 1702976, 1775345, 2899786, 860023, 2902973, 2902739]
```

## 3.2 Function 4: num_listings(dict)

This function takes as input a dictionary that is in the format of the dictionary returned by `host_listings` (Function 2). It returns a list `l` where `l[i]` is the number of hosts that have exactly `i` listings.

As an example, calling:

```
d = host_listings("tomslee_airbnb_brno_1500_2017-07-20.csv")
print(num_listings(d))
```

should give:

```
[0, 308, 35, 20, 3, 6, 1, 2, 2, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
```

# 4 Part III: Change over time?

To study how the price on a single listing changes over time you should write the following two functions.

## 4.1 Function 5: room_prices(filename_list, roomtype)

This function takes two parameters. The first is a list of filenames (`str`). The second is a string that is one of the strings:

- "Entire home/apt"
- "Private room"
- "Shared room"

As an example, the following is a valid call to `room_prices`:

```
file_list = ["tomslee_airbnb_brno_1258_2017-05-14.csv", \
             "tomslee_airbnb_brno_1500_2017-07-20.csv", \
             "tomslee_airbnb_brno_1383_2017-06-24.csv"]
output = room_prices(file_list, "Shared room")
```

The function should return a dictionary where the keys are `room_id`s (`int`) and the values are a list of the prices (`float`) for that listing over time, from the oldest data to the most recent. **However, the order of the files in the list passed as a parameter may not be in calendar order!** However, the filename does contain the date information in "yyyy-mm-dd" format. In the example above, the first file in the list has data from May 14, 2017. The second file has data from July 20, 2017. And the third file has data from June 24, 2017. You may assume that the date information will always be contained in the 10 characters immediately preceeding the ".csv" in the filename.

For example, given the call above, some entries in the `output` dictionary that are returned should be:

```
12036183: [42.0, 44.0, 45.0]
6049105: [13.0, 13.0, 14.0]
5938236: [19.0, 18.0, 19.0]
```

```
12691076: [12.0, 13.0, 13.0]
16191125: [10.0, 11.0, 11.0]
17696552: [16.0]
```

Note/Hint: Think carefully about how to get the data in order from earliest to date. One possibility is to sort the files and process them in order, perhaps using a dictionary that maps dates to filenames for this purpose. (But there are also other ways to accomplish this.)

### 4.2 Function 6: `price_change(dict)`

This function takes as input a dictionary in the format returned from `room_prices` (Function 3) and returns a **tuple** with three elements in the following order:

- maximum percentage change (which could be either positive or negative) for the set of properties in the dictionary
- the starting price for that property
- the ending price for that property

The percentage change for any given property is defined as the last price (the one corresponding to the latest date) minus the first price (the one corresponding to the earliest date), divided by the first price. In other words, if the price for a room is initially 40, then rises to 50, then falls to 40 again, the percentage change for that room would be 0%. If it started at 40 then rose to 45 and then to 50, the percentage change would be 25%.

As an example, the following:

```
file_list = ["tomslee_airbnb_brno_1258_2017-05-14.csv", \
             "tomslee_airbnb_brno_1500_2017-07-20.csv", \
             "tomslee_airbnb_brno_1383_2017-06-24.csv"]
output = room_prices(file_list, "Shared room")
print(price_change(output))
```

should print:

```
(18.75, 16.0, 19.0)
```

## 5 Part IV: Reflection

This last part asks you to reflect on the connection between the programming you've just done and "the real world." We consider the ability to make and to analyze these connections a critical skill!

### In the real world

Many cities have struggled with whether and how to regulate Airbnb. Look up a few news articles on this issue (there are many; you can try googling "arguments against airbnb" or "airbnb legal challenges").

### To do

Write a few paragraphs (no more than a page) summarizing some of what you read and **either**:

1. explaining how it relates to the analysis you've done for this assignment. (It may be useful to look for articles on cities where you also have the airbnb data.)

2. describing a data analysis experiment that you would like to conduct using Airbnb data to examine some of the claims in the article.

Make sure your writeup cites the articles that you use!
For this part of the project you should submit a pdf file named `airbnb_writeup.pdf`

## 6 Part V: Optional Extra Credit

You can earn up to 5 points of extra credit by designing your own question and answering it using the Airbnb data. If you choose to do the extra credit, please submit a separate file with the name `airbnb_ec.py`. Make sure the multiline comment at the top clearly describes the question and how your code addresses the question. Make sure that running the `main()` function demonstrates how to run your code.

### 6.1 What to submit

For this assignment you should submit your `airbnb.py` file and your `airbnb_writeup.pdf` files. If you do the extra credit you should also submit a `airbnb_ec.py` file. You do not need to submit any other files.

## 7 Grading

The following is the grading rubric.

| **Execution/Correctness** (45) | |
|---|---|
| `price_satisfaction` | 10 |
| `host_listings` | 10 |
| `room_prices` | 10 |
| `correlation` | 5 |
| `num_listings` | 5 |
| `price_change` | 5 |
| **Implementation** (21) | |
| Multiline comment at top accurate, relevant | 3 |
| Docstrings accurate, relevant | 3 |
| Comments in code accurate, relevant | 3 |
| Good use of conditionals | 3 |
| Good variable names | 3 |
| Good formatting | 3 |
| Other good coding style | 3 |
| **Writeup** (3) | |
| Appropriate article(s) cited | 1 |
| Summary of article(s) | 1 |
| Analysis/description of experiment | 1 |
| **Efficiency** (3) | |
| **Miscellaneous** (3) | |

Table 1: Grading rubric for the Airbnb programming assignment