Jack Bonacci
Capstone Project- Data Wrangling
30 July 2018

The dataset on which I am working is from a project listed on Kaggle called "AirBnb Listings in Major U.S. Cities". The data comes partitioned into testing and training datasets, but the testing dataset does not have include dependent variable observations. In order to be able to test for out-of-sample performance of my model, I decided to take the training dataset and partition it. The training dataset started as 74,100 observations and 29 variables.

First, I imported pandas and read the dataset into a dataframe called orig_data in my jupyter notebook using pd.read_csv(). I then called orig_data.describe(include = 'all') to get a feel for the data. The jupyter notebook display could not display the output from the .describe() call in its entirety, so I exported the dataframe that resulted from orig_data.describe() to a CSV file using orig_data.to_csv(). I then opened the exported file in excel from my local drive to see the results.

I saw some variables with fewer than 74100 observations, indicating NA values. I also scanned for outliers, and found that 'number_of_reviews' has a maximum of 605, a mean of about 20, and a standard deviation of about 38. I dropped this variable because of the outlier. As far as the variables with NA values, I dropped 'neighbourhood' because it not necessary as the dataset has latitude and longitude variables as well as a zip code variable. I dropped 'id', 'first_review', 'last_review', and 'thumbnail_url' because they had NA values and do not seem like they would explain any variation in the price of an Air Bnb. This new dataset was stored in a dataframe called trimmed_data. The columns were dropped using orig_data.drop(columns=[list]).

Next, I called trimmed_data.dropna() to drop the remaining observations with NA values. Most of these were present in 'review_scores_rating', but I feel that customer reviews of an Air Bnb are important in describing variation of prices charged. The resulting dataframe, called 'complete' has 47,249 observations and 23 variables.

To be sure that it was worthwhile to first trim the dataset before dropping NA values, I called orig_data.dropna(), and stored the result in the dataframe named "check" in order to check if I saved any observations through the trimming process. There were 38,502 observations and 29 variables in the resulting dataframe. Therefore, performing the trimming process before the process of dropping NA values saved nearly 10,000 observations.

Next, I partitioned the cleaned dataset into a training set and a testing set. To do this, I imported 'test_train_split' from 'sklearn.cross_validation'. I then sliced 'complete', the cleaned dataset, into 'y' and 'x' to separate the dependent variable

'log_price' and the 22 independent variables, respectively. I then called "xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.25)". This separated the 'complete' dataset into four dataframes:

xtrain: 35,436 observations, 22 variables
xtest: 11,813 observations, 22 variables
ytrain: 35,436 observations, 1 variable
ytest: 11,813 observations, 1 variable