Charlie Denhart
DS4400
11/20/18

Project Checkpoint

Data Collection

The first step I took in this process was to collect my data. All data was retrieved from InsideAirbnb.com, which provides scraped Airbnb data from dozens of cities and over several years. Although I plan on utilizing data from many, if not all, of the cities available, I have just pulled from Boston for now for simplicity. When I use data from multiple cities, I will pull data via HTTP requests from the site in order to ease disk storage load on my computer, but for now I have simply downloaded the Boston data onto my hard drive. Since the scrapes used took place over several periods of time, each belonging to a separate CSV file, I aggregated all of the data collected into a single Pandas data frame.

Data Cleaning

Once collection was completed, I moved on to viewing and cleaning the data. There were several minor actions that were necessary. For all price values (nightly rates, weekly rates, monthly rates, security deposit, etc.) I removed dollar signs and converted the values to integer type. I conducted similar cleaning on all percentages (host acceptance rate and host response rate). For dates present, I simplified the format to simply containing year and month as that is all that is needed to uniquely identify the instance of a unit in time.

Regarding missing values, I simply dropped variables with significant numbers of missing values and then removed all rows with missing values for now. I will likely do some imputation later if more observations are needed. Finally, I needed to deal with data stored as

strings. For the purposes of this check-in, I will simply be running some regressions and attempting to classify the neighborhood of an Airbnb unit. For the regressions, therefore, I converted all categorical variables to N – 1 dummy variables per categorical variable. For the classification, I did the same, but for the neighborhood feature, I utilized Scikit-Learn's Label Encoder function to convert it to a continuous variable capable of being classified by machine learning algorithms.

Feature Engineering

I wanted to make use of some features that could not be used in their current format, so I did some feature engineering. I generated a feature for the number of amenities in a unit by counting the length of the list of amenities provided in the data. I also calculated the number of days as a host by subtracting a given host's start from the current date.

In addition to providing files with detailed features per unit, the site I collected data from also provided corresponding files listing the daily price and availability per unit. I used this information to derive features for occupancy rate and the number of price changes per unit.

Feature Selection

Despite narrowing down features in previous work, I was still left with an abundance of variables, many of which are likely irrelevant. Therefore, I decided to do some feature selection using PCA. I cut down the number of features to both 15 and 50 from the original 126 in order to test different impacts on model performance.

Data Exploration

In order to get a better sense of the data being used, I also did some data exploration. In order to get a sense of how data was correlated, I conducted a Variance Inflation Factor (VIF) test on my independent variables. I found that several variables, including different review scores and the bed type of a unit, exhibited high multicollinearity, so I dropped them from the model. I also wanted to see how the numbers of Airbnb units laid out in different regions across the city, so I calculated density values for each zip code. Then, via the Mapbox GL JavaScript API, I plotted the zip code regions on a map and filled them in appropriately with their density values using a GeoJSON file I constructed.

Modeling

I wanted to see how well I could predict the performance of different units, so I regressed on a few different metrics including occupancy rate, number of reviews, and number of price changes. The regression for occupancy rate was successful, with an adjusted r-squared value of around 0.7 - 0.8 for several specifications. The number of reviews and number of price change regressions were slightly less encouraging, however, with adjusted r-squared values hovering around 0.3 - 0.5.

I also was curious about how well I could determine the neighborhood a unit resided in, so I used a K-Nearest-Neighbors classifier to predict the neighborhood for a unit. Using the reduced features from the PCA, I achieved 93 percent accuracy on a separate training set.