Name: Jonathan Lawrence
Date: 1/26/2020
Assignment: 7.3 – Hotel Recommendation Algorithms

Approach

The instructions said to predict 'hotel_cluster' which is a categorical string variable. I knew this meant that I needed to select algorithms that would not assume numerical, linear correlations. With that in mind, I chose **classification** algorithms.

Naïve Bayes (NB)

I chose Naïve Bayes because it is known for its simplicity and speed. I wanted something that could run fast in order to quickly tell me if I made a mistake in my data. Furthermore, one of the conditions for NB to work is that its features must be independent (or 'naïve') of one another. The correlation test I performed showed that the variables had no real dependencies on one another, so I felt that it was a good first choice. However, there was a very weak dependency, so the accuracy of the model was a bit low at **0.1034.**

K-Nearest Neighbors Classifier (KNN)

K-Nearest Neighbors is another simple algorithm that I've used on a few occasions for past assignments, so I was relatively familiar with it and wanted to use it fairly early in this assignment. While it was a coincidence that my features were independent in order to use NB, that was the only thing I had to worry about with that algorithm. But for KNN, I knew it didn't make any assumptions about the data structure, so I was able to feed it my data without even having to consider the structure. The trick with KNN though, was knowing how many nearest neighbors to pick. I tried using 3, 5, 7, and 9 nearest neighbors to see which one would give me the best accuracy. I didn't want to use too many neighbors because I have heard that it is generally good practice to stick with 1-10 neighbors. The best was 9 neighbors with an accuracy of **0.2751**.

Random Forest Classifier (RF)

I was only vaguely familiar with Random Forest at this point, but I did know that it was an extension of a decision tree which is exactly the goal of this project; building a system to predict 'hotel_cluster' (or leaf). I learned that a RF takes a random subset of features for each split of the decision trees in order to reduce the chance of having multiple paths with the same structure. The result of similar paths would cause an unrealistic, highly correlated result that would not be as useful. Regardless, it's still possible that multiple paths will end up at the same hotel_cluster, or that some paths might overfit or suggest a different cluster. Therefore, taking the average of all results will help to identify a singular 'hotel_cluster' to recommend. The accuracy of my model was **0.2486**.

Support Vector Machine Classifier (SVM)

I had never heard of this classifier before, but my online research suggested that I try using it, primarily due to the fact that it is effective when working with a large number of features. I had already declared variables for the features (X) and labels (y) in order to run the other algorithms, so there were no issues there. My hotel dataset had 24 columns, but I broke some into additional columns in order to get more specific data. SVM actually resulted in the highest accuracy of all the algorithms I used: **0.3228**.