1. Introduction

The following project looks at various predictions tasks on AirBnB data from listings in the Greater Seattle Area. The report will be separated into two sections based on the prediction of the two variables of interest: **`price`** and **`review score rating`**.

2. Exploration

2.1 Dependent Variables

First, we focus on the dependent variables: **`price`** and **`review score rating`**.

2.1.1 PRICE

The price of the Seattle airbnbs in the data is distributed as Figure 1 below.

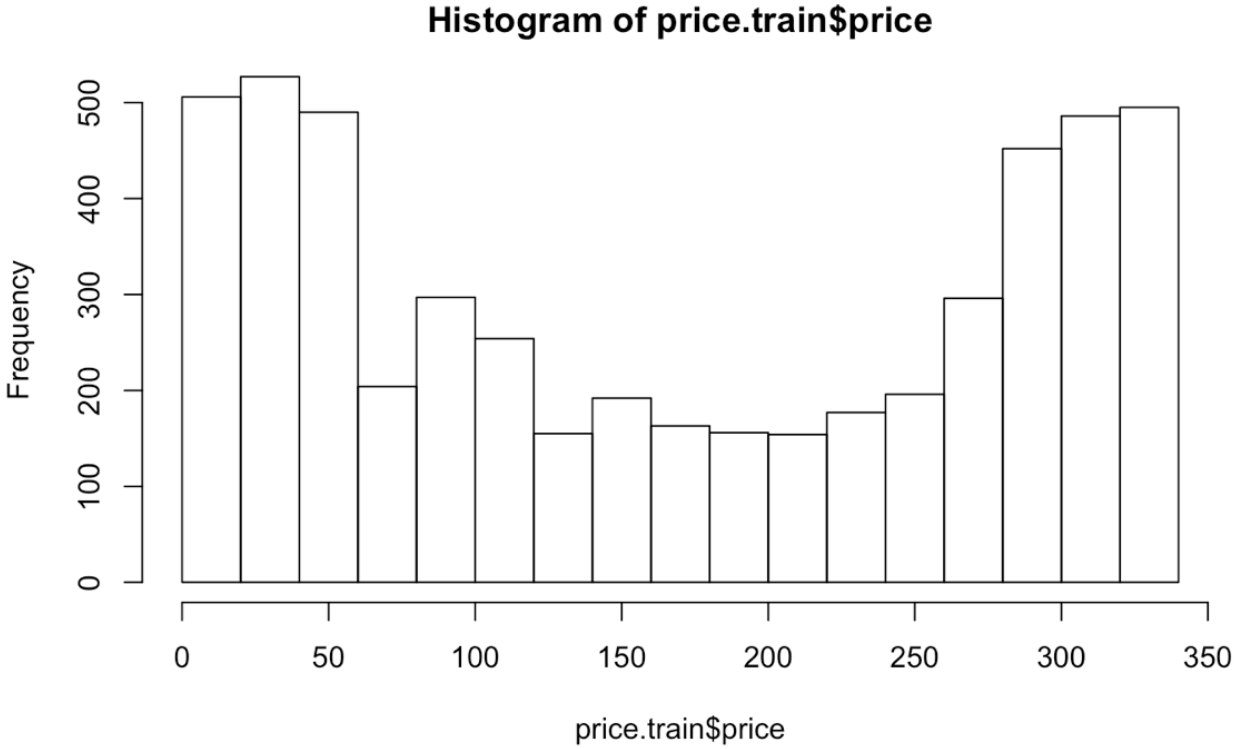

Figure 1: Histogram of Seattle Airbnb price

It seems that there are more Airbnbs in both the higher and lower ranges of the Airbnbs price distribution. There are less Airbnbs with rent prices in the middle price range, which is about $100 to $250 per night.

The key statistics of the Airbnb prices are:

- Min: 1.0
- 1st Quartile: 57.0

- Median: 158.0
- Mean: 168.1
- 3rd Quartile: 287.0
- Max: 337.0

Since the Airbnb prices are quite evenly distributed without long tails on either end, it does not seem necessary to transform the price variable.

2.1.2 REVIEW SCORE

The review score has the following count and proportion:

Undesirable/Desirable	**0**	**1**
Count	580	3463
Proportion	0.1435	0.8565

Table 1: Review Score Distribution

2.2 Exploratory Data Analysis

This section looks at the predictors and their relationships with the 2 response variables. Only select variables have been discussed in the report.

Some of the categorical variables that show skewness in terms of how common the listings are include:

- Most common **`room_type`** is the entire home
- Most likely **`host_response_time`** is within an hour
- Most hosts only have one listing
- Most popular **`bed_type`** is a real bed
- Most common **`property_type`** is house and apartment

Based on the EDA, we decided to categorize the most common variables into one group, while grouping the remaining ones into another group. More details as to the encoding process of the data set is explained in the next section.

We also included the entire data set when building the initial model. We later performed unsupervised learning via **`PCAmix`** to determine if feature selection would improve the model, which will be discussed in the following sections.

2.3 Data Preprocessing

Some of the variables in the original datasets had to be transformed before being used for model fitting.

1. The **latitude**, **longitude**, and **neighborhood_group_cleansed** variables were transformed and encoded to provide more meaning with the following steps:
    - We found the coordinates of one popular landmark situated in each neighborhood.
    - Then we calculated the latitude and longitude point differences of the location of each observation to the corresponding landmark of the neighborhood that the observation is in.
    - We found that each latitude and longitude point converts to a distance of 69 miles.
    - The latitude and longitude point differences of each observation was converted to numeric mileage distances.
    - We encoded a difference of 1 mile as "Center", a difference of 2 miles as "Close", and a difference of more than 2 miles as "Far".
    - Due to ambiguity of the neighborhood that the *other neighborhoods* group are located, those observations were compared to the coordinates of the Space Needle, the most popular landmark in Seattle.

2. Variables having a huge range of values were converted from numeric to categorical:
    - **host_listings_count** was encoded into 2 groups: 1) "Single", corresponds to hosts who have a single listing 2) "Multiple", corresponds to hosts who have more than one listing on AirBnB.

3. Some variables which were already categorical in nature had some of their groups combined:
    - **property_type** was encoded into 2 groups: 1) "House/Apartment", corresponding to the most common property type 2) "Other", corresponding to all other property types.
    - **bed_type** was similarly encoded into 2 groups: 1) "Real Bed" 2) "Other".
    - **cancellation_policy** was encoded into 4 groups: 1) "Flexible" for the flexible policy, 2) "Moderate" for the moderate policy 3) "Strict" for the two strict policies, "Strict 14 with grace period" and "Strict" 4) "Super Strict" for the two super strict policies, "Super strict 30" and "Super strict 60".
    - **amenities** were summed for each observation and listed as a new numeric feature, **amenities_count**, in the data set.

4. The **X**, **id**, **amenities**, **longitude**, and **latitude** variables were removed from the data because they do not contribute to the fitting of the model or are redundant.

3. Supervised Analysis - PRICE

The price dataset was split into 80% training set and 20% validation set. Two linear models and numerous ridge and lasso regression models were fitted.

Model	**Train Set**	**Validation Set**
Linear Model (with all variables)	11169.93	12438.98
Linear Model (with omitted variables)	11239.02	12387.86
Ridge Regression	11226.46	12400.7
Lasso Regression	11170.06	12410.10

Table 2: Mean Squared Error of Fitted Price Models

The models were fitted with the following process:

1. Linear Model (with all variables): All the potential predictor variables provided in the dataset were included to fit the linear model. Numerous variables did not seem to impact the fit significantly at a 90% confidence interval.
2. Linear Model (with omitted variables): From the summary statistics of the full linear model, the predictor variables that significantly impacted the linear model fit were identified. Only these variables are included in the second linear model. The variables included were: **`host_response_rate, host_listings_count, neighbourhood_group_cleansed, room_type, bathrooms, bedrooms, cleaning_fee, distance`**
3. Ridge Regression: Ridge regression models were fit with all the predictor variables included. Numerous models were fit with various tuning parameters. Lambdas that were fit range from about 3 to 36000. The regression fit with the smallest training set mean squared error is the one with lambda being 19.57.
4. Lasso Regression: Lasso regression models were also fit with all the predictor variables included. The 81 variations of tuning parameters fit for the lasso models range from 0.02 to 36.67. The model with the lowest mean squared error is the one with lambda being 0.0215.

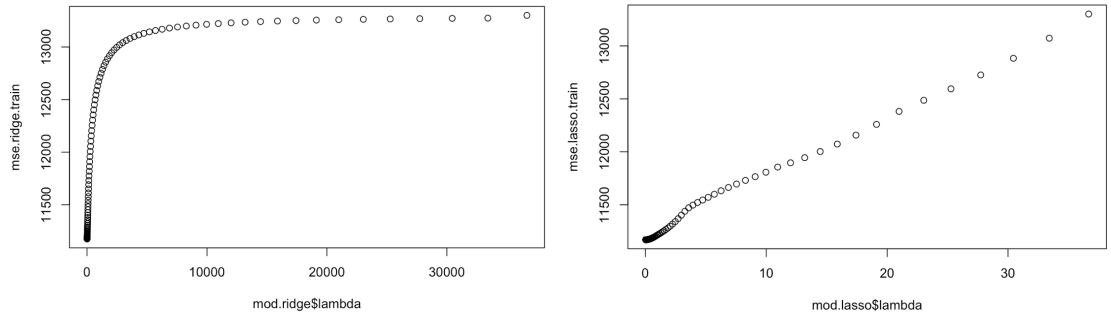

Figure 2: Ridge regression mse (left) and lasso regression mse (mse) of training set with varying tuning parameter

The mse for each model's training set and validation set are all quite similar, and differ within a small range. While the optimal ridge regression model has the second lowest validation set mse, it has a lower training set mse than the linear model with omitted variables. Comparing both the mse values for each model, it seems that the optimal ridge regression fit is the best model that predicts price out of the four types of models fitted. The optimal ridge regression model is as follows:

```
mod.ridge.opt = glmnet(x = model.matrix(~., data = df.train), y = y.train,
                alpha = 0, lambda = 19.57)
```

3. Supervised Analysis - REVIEW SCORES

3.1 Model Fitting

The data was split into 70% train set and 30% validation set. Various models were fit to find the best model with the lowest possible test misclassification rate. The best model was then fit on a separate test file to generate score predictions.

A total of 6 models were experimented. The criterion for choosing the supervised learning techniques were as follows:

1. Algorithms focusing on binary outcome were implemented. This narrowed down the supervised learning techniques to logistic regression, k-NN, LDA/QDA, classification trees, SVM, random forest/bagging, and boosting.
2. Since prediction is the goal, interpretability of the model is not of interest; accurate predictions are more desirable. This further narrowed down the relevant techniques to SVM, random forest/bagging, boosting.

One point to note is the categorical features were one-hot encoded in order to process the k-NN, SVM, random forest, and boosting algorithms. The variables are as follows: **`cancellation_policy`**, **`distance`**, **`bed_type`**, **`host_is_superhost`**, **`host_identity_verified`**, **`instant_bookable`**, **`host_response_time`**, **`room_type`**, **`proprty_type`**, and **`host_listings_count`**

To ensure the breadth of our analysis, we implemented logistic regression, k-NN, classification tree, SVM, random forest/bagging, and boosting. An important note to make is that, although LDA/QDA are similar to logistic regression, but they are more commonly used in response variables with more than two classes, which is not appropriate in this setting.

Model	Train Set	Validation Set
Logistic Regression	20.95%	21.12%
k-NN	12.81%	13.56%
Classification Tree	15.66%	17.45%
SVM	12.76%	13.81%
Random Forest	12.44%	13.15%
Boosting	11.35%	14.04%

Table 3: Misclassification Rate of Baseline Models

The best model for predicting the **`review_scores_rating`** variable utilizes the random forest algorithm.

A benefit of using random forest is that the algorithm can generate new features and is flexible to create a good predictive model. The baseline model used parameters that were by default in the **`randomForest`** function: **`mtry`** = 4, **`ntree`** = 500, **`nodesize`** = 1.

Based on Table 3, the misclassification rates of the training and validation sets seem to be similar to each other, implying that there may be a problem of bias.

To further improve the baseline model and to eliminate the problem of bias, we tuned the parameters and altered **`ntree`**, **`mtry`**, and **`nodesize`** to find the optimal combination. (Figure 3 below shows the error rate in relation to the number of trees split). We took particular care in determining the **`nodesize`** because deeper trees introduce more variance, whereas shallower trees introduce more bias. Likewise, the **`mtry`** parameter was also chosen carefully, because smaller values lead to overfitting.

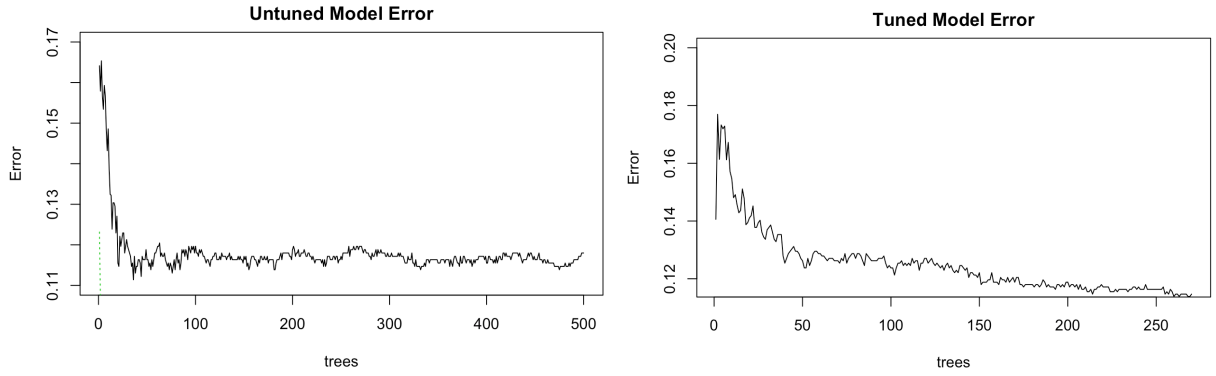

Figure 3: Untuned Model vs. Tuned Model Errors

We performed a larger grid search across the parameters of interest, then looped through each parameter combination and evaluated the model. The final algorithm we implemented was as follows:

```
new.bag =   randomForest(review_scores_rating~., data=review.train,
ntree=270,
            mtry=20, nodesize=9)
```

Model	Validation Set
Random Forest (Baseline Model)	13.15%
Random Forest (Tuned Model)	12.16%

Table 4: Comparison of Misclassification Rate of Random Forest Baseline Model and Random Forest Tuned Model

3.2 Feature Reduction

We looked to unsupervised learning techniques for feature reduction. Since a majority of the data set's variables are categorical, many of the unsupervised learning techniques were not applicable to this project. However, we were able to apply **`PCAmix`** on a mixed data set of continuous and categorical variables such as this.

The output told us that 11 components should be retained for minimal variance in the model. (See Figure 4.) But after running the suggested model, the misclassification rate of the validation test did not decrease.

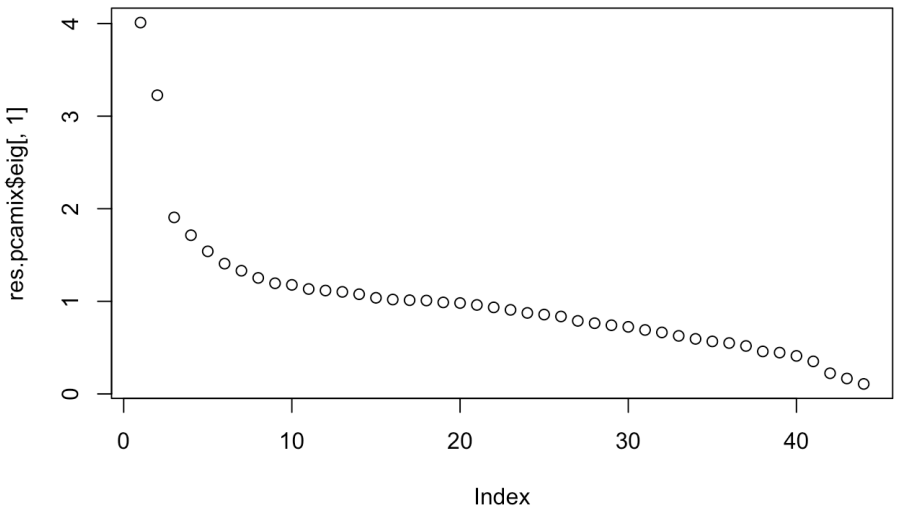

Figure 4: **`PCAmix`** output graph

4. Analysis of Results - PRICE

While the optimal Ridge regression model may predict Seattle Airbnb prices, the model itself is mainly interpretable. A disadvantage to this is that it may be hard for stakeholders to understand which attributes of an Airbnb listing add the most values to the listing. Some information that could be used to educate Airbnb hosts and guests are missed in the Ridge regression model.

Furthermore, while the final model chosen has relatively lower mse among the models fitted, the validation set mse value of 12400.7 suggests that the predictions from the model fit is influenced by noticeable bias and variance in the prediction points. Further study could be done on testing other ways of transforming variables or other regression models to achieve a lower mse.

4. Analysis of Results - REVIEW SCORES

There are naturally constraints on classifiers in general: the prediction and evaluation time is limited and can incur high computational costs, especially as the number of parameters, features, and models tested increase. This would affect the real-world application of the model, since AirBnB would run this algorithm on several tens of thousands of data values.

A big part of building the more computationally advanced models involve standardizing the data. Rare categories may have either too large or too small an effect on the response variable(s). This calls for designing a similarity matrix that reflects the accurate influence of each feature on the response.