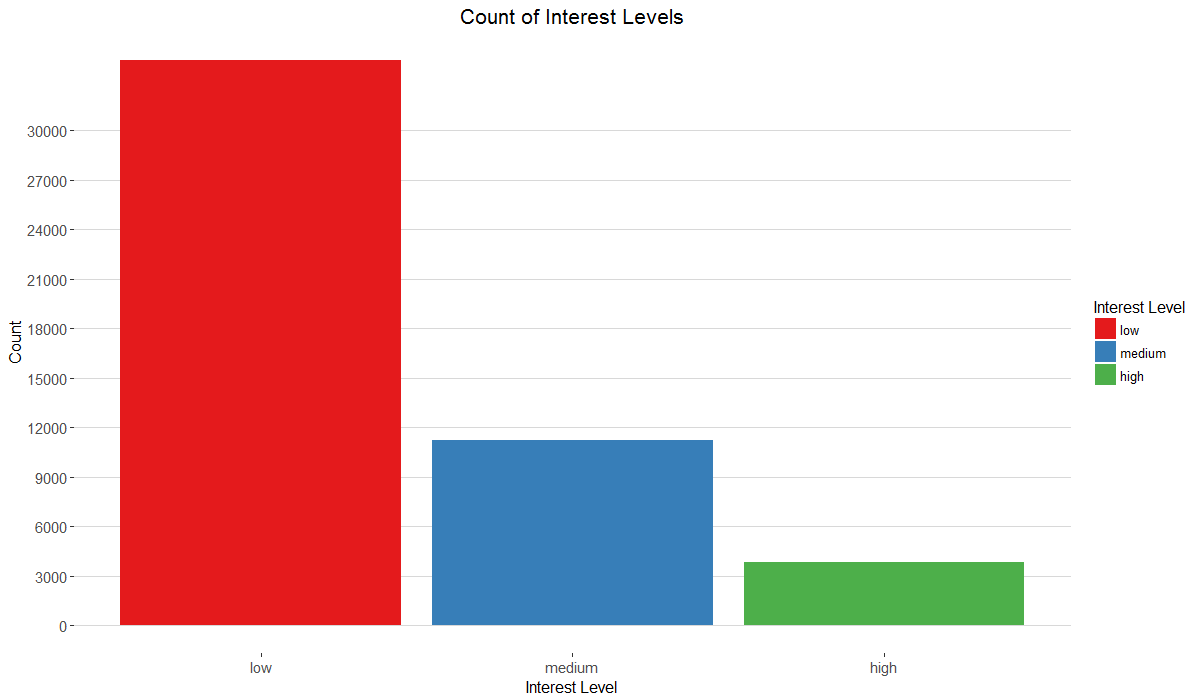
Understanding Class Imbalance and Ensemble Modeling in the Six-Sigma Connect: Rental Listing Inquiries

By Emil Parikh, Glen Ferguson, and Jason Chen

Introduction

The Six-Sigma connect challenge was to predict interest-level—high, medium, or low—of RentHop apartment listings in the New York City area. This is a classification problem that could ideally be solved using supervised learning. To understand the data, we plotted the interest levels to understand the relative number of each interest level. As can be seen in the plot below, the data has a class imbalance with a significantly higher number of low-interest listings than medium- and high-interest listings combined.



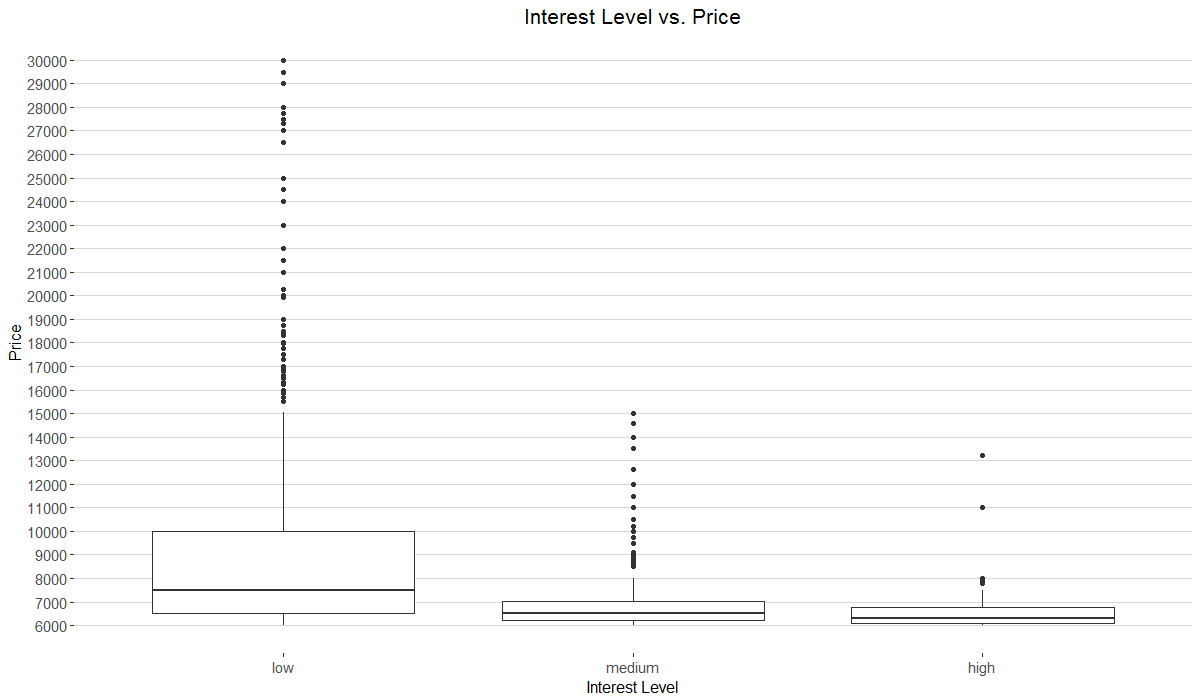
An apt example of this imbalance is the ratio of the height of the tallest mountain in the world (Mt. Everest) to the height of the tallest building in the world (the Burj Khalifa). The Burj Khalifa, while extraordinarily tall, is dwarfed in comparison to Mt. Everest, as can be seen in the image below (the Burj Khalifa is in the orange box!). Thus, the crux of the project is how to classify the high- and medium-interest properties while still capturing the mountain of low-interest properties.



Exploratory Data Analysis

# Simple Features

One of the most interesting facets of the data was the effect of price on interest level. As can be seen in the chart below, the higher priced properties were much lower interest and had a significantly larger variation. This difference indicates that the price is likely to be a significant variable in the data.



Along with price, there were other “simple” features that could be examined immediately such as number of bedrooms and bathrooms; these did not have as large of an impact as price.

# Involved Features

There were also involved features that, while data rich, could not be examined immediately; we needed to conduct feature engineering before extracting value: photos, features, descriptions, and manager id. While the number of photos, number of features, and word count of the description could be immediately calculated, the human interpretation of photo content, feature importance, and description content required the application of advanced techniques in image processing and natural language processing.

## Photos

To transform the photos into data, we first classified images using the Inception model in the Tensorflow neural network. While we were successful at classifying items in the images, the process was slow. To classify all ~300,000 images would have required more time than was possible to complete the project. In a second attempt, we used luminance, but this failed to differentiate among the photos. Finally, we used the mean and standard deviation of the red, green, and blue colors in the photos.

## Features

The first step in extracting feature importance was standardizing names. E.g “hi rise” and “highrise” needed to be grouped together as a single term. This standardized list was then transformed to a count using term-frequency-inverse document frequency (tf-idf). The result of this was 400 columns (for 400 terms) of tf-idf values. However, we had to reduce these 400 columns because, with our limited resources, our models would not run with so much predictors. To reduce these 400 columns in the final model, we used logistic regression with the 400 columns as predictors of interest level to form three columns that were predicted probabilities of high, medium, and low interest. These 400 columns were then combined with the remaining predictors to fit models by.

## Description

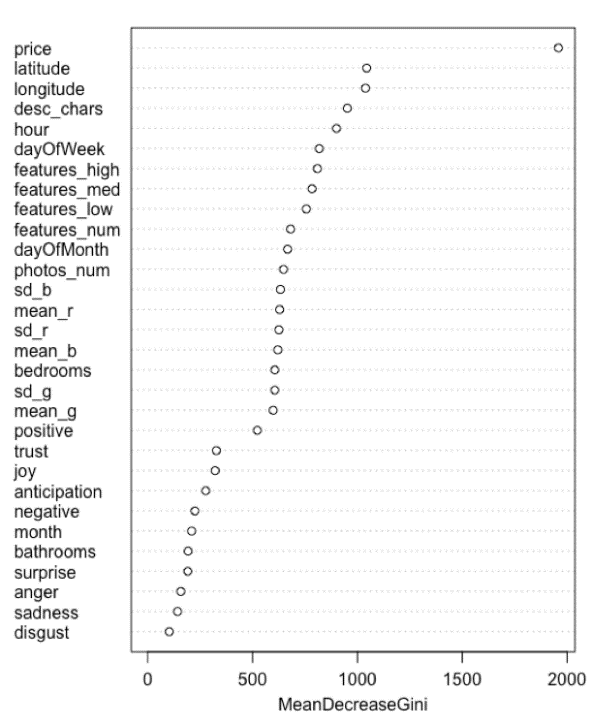
The description column was transformed using two methods. First was to separate common terms in the description by n-gram (a sequence of “n” contiguous words). The NRC library used to determine the n-grams. The most popular features included stainless steel appliances. The values were transformed using tf-idf and categorized using an SVM. This model failed to differentiate the interest levels and was not used in other calculations. The second method used to transform the description column was sentiment analysis, also using the NRC library and the tidytext R package. The sentiments used were positive, negative, anticipation, fear, anger, trust, surprise, sadness, disgust, joy, and anger. For each description, the number of words corresponding to each sentiment were counted and added to each sentiment column. These columns were then used in future models.

## Manager ID

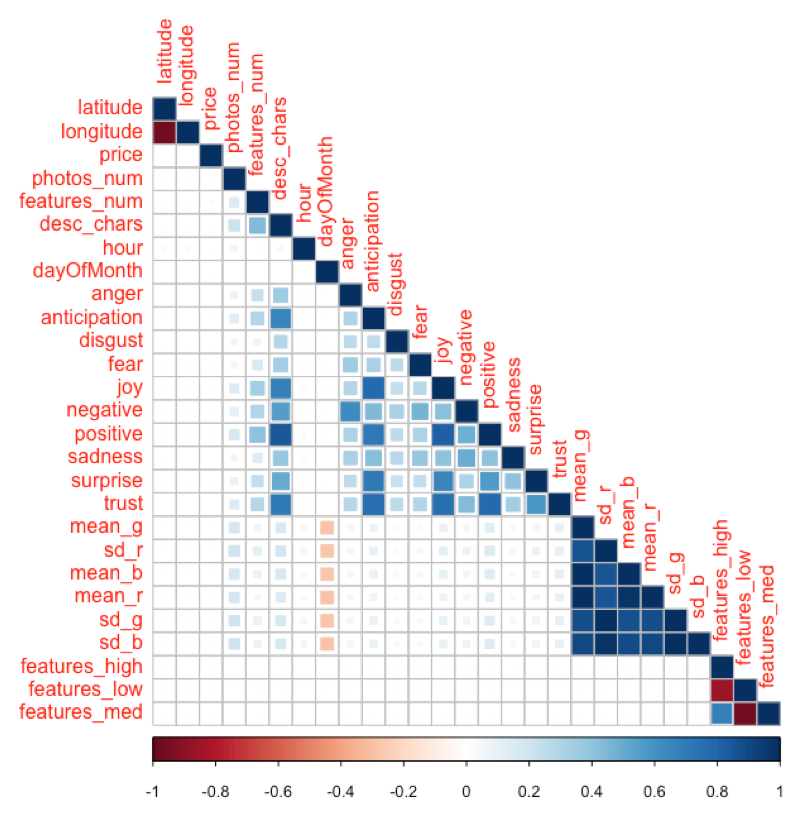
Another column that could have held significant value was the manager id column. To use this column, we followed a Kaggle Kernel that classified the managers based on interest level. These values significantly improved our predications in the validation portion of the training set but significantly increased the logloss (reduced accuracy) for the test set. We believe the inclusion of the output, interest level, in the determining the values for the predictor resulted in leakage. The resulting models were over fit to the data. These features were removed in subsequent models.

Models

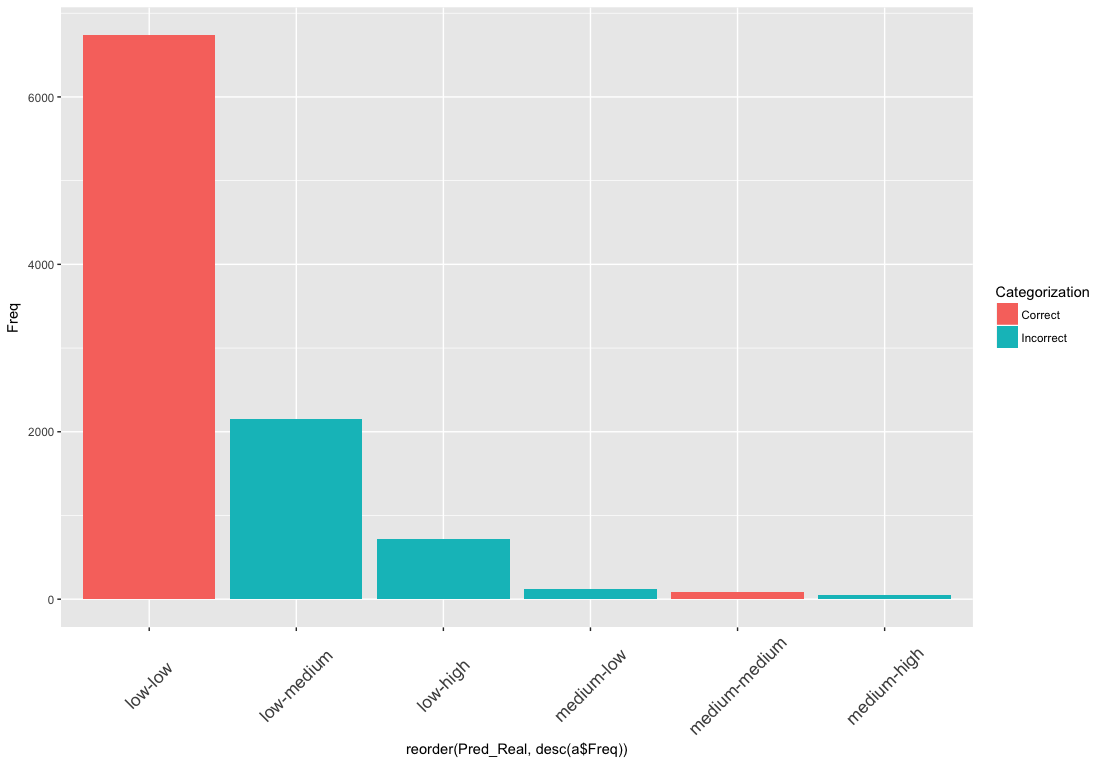
Using a random forest model, we calculated the relative importance of each variable (contribution to decrease in the Gini). The price variable, as expected from the EDA, was the most important variable, while the location (latitude and longitude) followed. The number of characters in the description was also important and the hour and day the listing was created was also important. The sentiment features tended to be the least important.



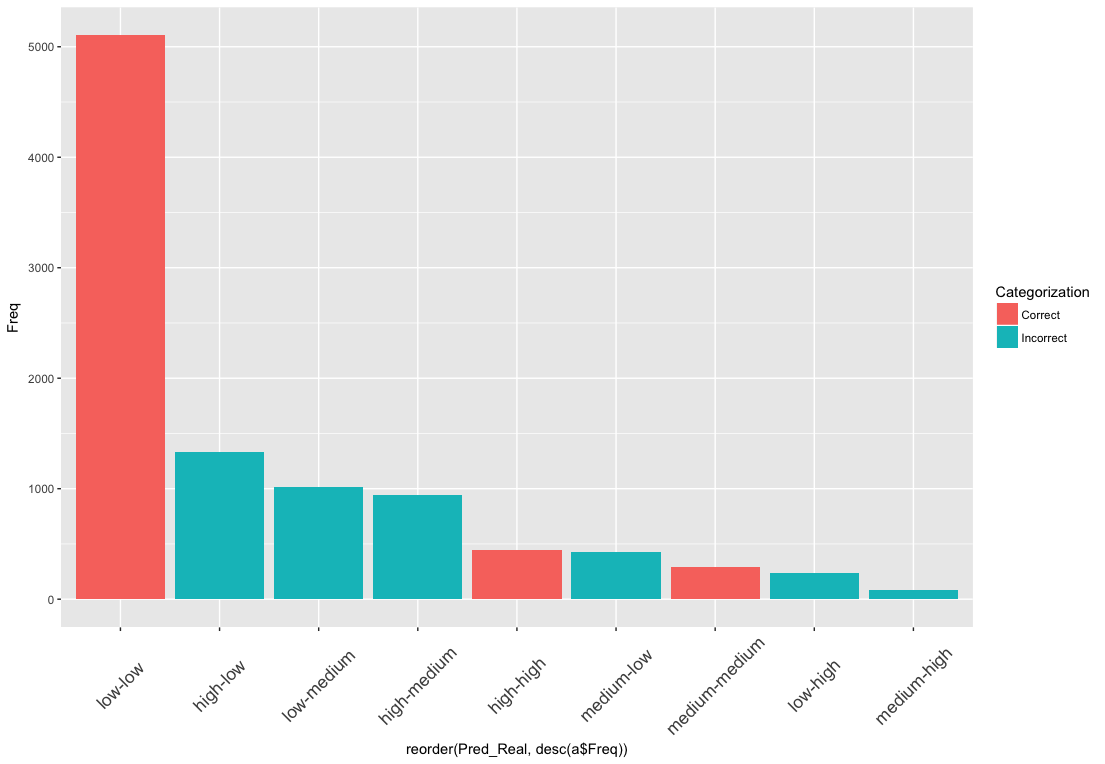
To check for correlation between the variables, we performed a Pearson correlation plot for the numerical variables. As can be seen in the chart below (strong blue is higher positive correlation and stronger red is more negative correlation), the photo variables have significant correlation.



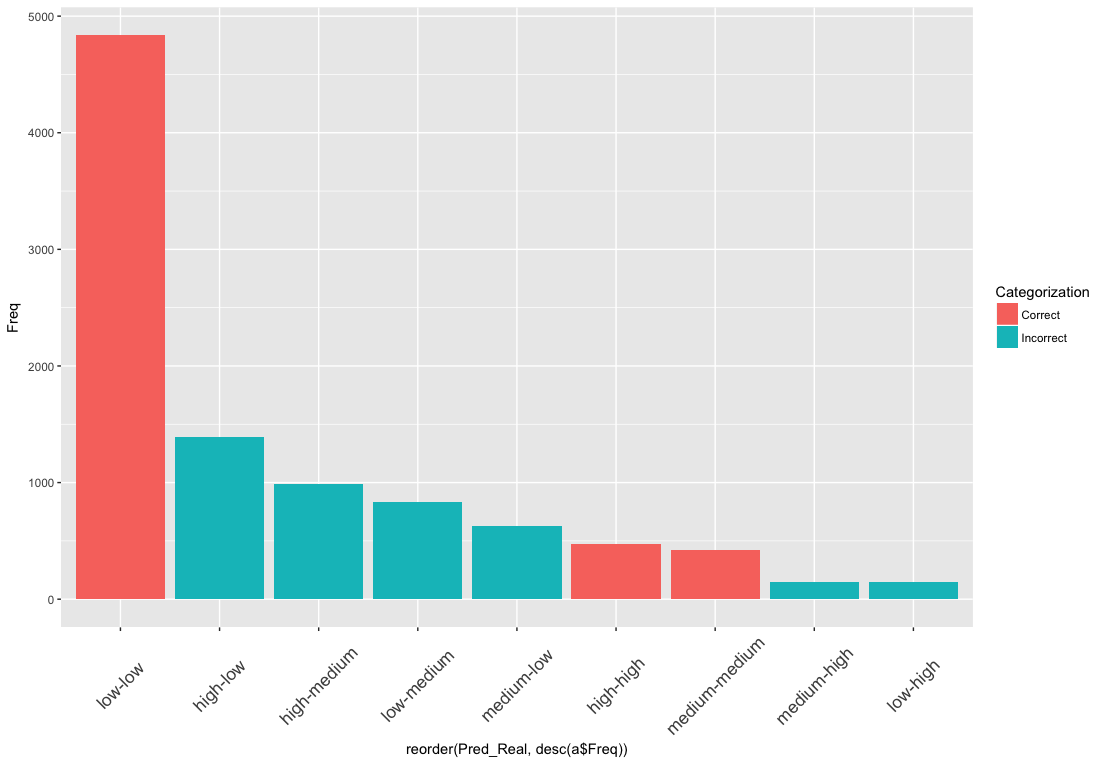
The models chosen for analysis included logistic regression, random forest, gradient boosting, and extreme gradient boosting. The use of increasing complexity was used to determine if simple models could predict the results, and if increasingly complex models would capture different aspects of the data that could be combined. The logistic regression model used regularization including, ridge and elastic net. As can be seen in the bar plot of the confusion matrix below, logistic regression with elastic net failed to capture any of the high interest values but performed well for the low interest properties.



We used the Ranger implementation of the random forest algorithm due to its improved speed. This model, using up to seven predictors for each decision tree, also did not perform well.



The gradient boosting method as implemented in the H2O R-package was also used. This model required some parameter tuning, but changes in these values did not result in a significant increase in accuracy. The values are shown below. While more high and medium values are predicted correctly, a significant number of low values were incorrectly predicted.



XGBoost was implemented to due to its popularity and successful track record with previous Kaggle competitions. Although the model made decent predictions with its default settings. Tuning the model proved to be difficult, due to their large number. To save time, we experimented by manually tuning the learning rate and depth of trees until we minimized the error between the training and validation set. Similar to the other models, XGBoost was very accurate in predicting listings with low interest levels, but struggled to correctly predict high interest listings.



To improve these models by directly addressing the class imbalance we used up-sampling and down-sampling. These methods respectively increase the number of observations for the minority classes and decrease the number of majority classes by random sampling. The number of observations is equalized in the resulting datasets prior to model training and could result in better predictions. The predictions from up- and down-sampled data did not significantly improve most models. For logistic regression, far more high interest properties were correctly predicted but a larger number of low interest properties were incorrectly predicted. Final models only used the XGBoost method.

The final prediction was created from an ensemble of weighted XGBoost models. A base model, an up-sampled model, and a down-sampled model. The base model categorized most of the low interest models correctly but predicted few of the high interest models correctly. The up-sampled was the reverse, and the down-sampled was intermediate. A weighted average somewhat improved the accuracy overall but was comparable on logloss to the base model.

Conclusion

The lesson learned is that class imbalance and a lack of differentiating features resulted in difficulties in categorization. The use of up-sampling tended to improve the correct prediction of high interest properties but tended to be poor at predicting low and medium interest properties. While a significant amount of time was spent on feature engineering, more time finding features that clearly differentiated between low, medium, and high interest properties was critical to making the most accurate predictions. Model tuning, stacking, and ensembling did little to increase the accuracy of the predictions. For this competition, it would have been wise to spend significantly more time on feature engineering and less time tuning models.