DATA ANALYTICS

RentHop Report

MARCH 31, 2017

Queen Mary University of London

HESHAM DAR - 160982589
PETUR EINARSON - 160990447
CHRISTOPHER ADNDRE OTTESEN - 160915509

Contents

List of Figures

List of Tables

1 Introduction

RentHop is a website where users can search for rental listings in major cities in the USA. Two Sigma, the company behind RentHop, was created with the idea that finding a home shouldn't be about looking at every apartment listing but rather looking at the best ones. Quality, not quantity. They aim to achieve this by applying data analysis and quantitative algorithms to their apartment listings. Which is why they want to better understand their customer's preferences and specifically which types of apartments generate the highest levels of interest.

The company spokesperson explains that finding a suitable place to live is a tricky process to automate, as the act of searching involves a great sense of human feeling. When one finds a place to live, a large emotional investment is done in the process as different people have different preferences for what is and isn't important.

Often the sense of reward gotten from finding a good place is directly tied to the effort and time exerted on finding the place. Therefore, when dealing with apartment brokers, the spokesperson mentions that the faster a broker does his job, the fewer people like him. People tend to prefer a broker who spends a large amount of time to find the perfect place, instead of the broker that immediately shows them the perfect place. When a broker immediately shows them a great place, people often feel cheated and find it unfair to pay a great fee for that kind of service. They feel as if the broker was too quick and that he hasn't invested enough time in them. Which is why it can be very difficult for a computer to emulate a good apartment broker. (Lee Lin, 2017)

Although difficult, automation in this field is a desirable effect. It will enable brokers to spend more time on value-adding tasks, as less time is wasted on showing people subpar apartments that don’t fit the user's criteria.

2 Approach

2.1 Getting to know RentHop

The initial step was to get to know RentHop. To review the website and understand how listings are presented. This will give an insight into how the data is collected and what it represents.

The website aims to make it as easy as possible for the user to browse listings and to navigate the ocean of apartments, by showing him the best ones first. They achieve this by using a proprietary scoring system: "HopScore" (RentHop.com, 2017). The HopScore is based on a number of factors, which fall into three broad categories:

- Listing quality
- Agent Performance
- Listing freshness

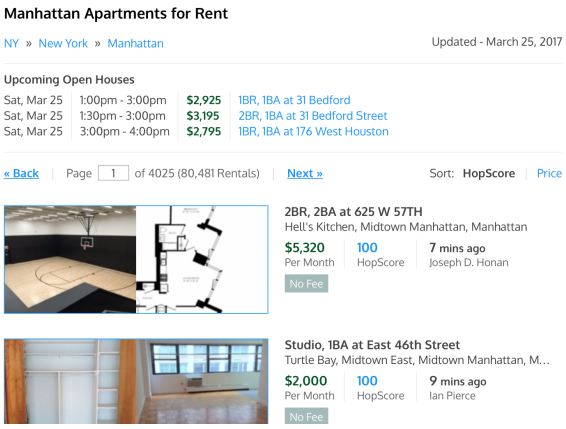

Figure 1 - Example listing from the RentHop website.

The website also offers a filtering function to sort listings based on different variables such as price, location, the number of bathrooms and more. Each individual listing has fields for images, address, price, description and more, presented in an informative and aesthetically pleasing manner. The information contains both qualitative information and quantitative information, which opens up for a wide range of feature selections and analyses.

2.2 Exploring features

RentHop provides a labelled dataset through the Data Science community Kaggle.

The labels are three and indicate the level of interest: "Low", "Medium" and "High". Example features are the price per month in USD, location: address and longitude & latitude, number of bedrooms, description as well as photos.

RentHop does not provide information about how the interest level is calculated. Most likely they use various factors such as a number of times users has clicked on a listing, how many clicks they do within each listing and if they contact the broker of the listing trough the InMail system provided by RentHop.

The advertised goal was to use the provided features to understand why a listing gets a certain interest level. In other words, what makes one renting more appealing than another?

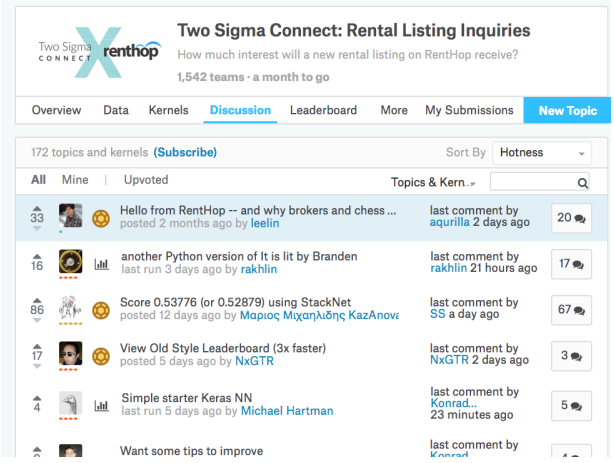

Figure 2 - Information on Kaggle website from the RentHop representative.

3 Business value

3.1 Value for RentHop

The main business value of this project is two-fold. By providing an in-depth analysis of the metrics RentHop is currently collecting on their properties, it may help guide where further insight may be achieved. By producing a model with which user interest can be predicted, RentHop could potentially alter business practices and make actionable decisions to maximise interest in existing properties of focus on particular types of properties that are associated with high levels of interest.

3.2 Value for the Listing Authors

When people look for a place to live, one can assume they look for a place fitting their needs in terms of price, the number of bedrooms and other metrics such as distance from the workplace, city centre or areas of interest. Unfortunately, there are many metrics that are very difficult to quantify which do affect the appeal of an apartment; how people feel about an area, neighbourhood safety and demographics of the area. An assumption can be made that people prefer to stay in areas where they feel like they fit in or have similar people surrounding them. It can also be assumed that people tend to have some prior knowledge about where they want to live and know what their budget is. These personal preferences and feelings are of significant importance when looking for an apartment.

Data analytics and statistics can help understand what features attract visitors to one flat over another. The analysis can shed light on what creates attraction to one listing but puts off people from another, all these human interactions potentially scrutinised by statistical data and compared with the authors gut feeling.

Ways of designing a listing might provoke different feelings in people. The particular wording, picture or price. These are psychological effects the data analysis aims to uncover to understand what attracts people to different types of properties and uncover the main goal of inference: predicting the levels of interest in different types of listings.

3.3 Value for people searching for listings

RentHop users looking for an apartment could benefit from better property searches, shortening the time it takes them to find their ideal apartment without having to input a significant amount of information about their preferences.

4 Workflow

4.1 Getting started

As stated earlier, the first step of the project was getting to know RentHop. The first step was to understand RentHop, read the declaration from the spokesperson and explore the assignment given to get a sense of direction as to where to start. Thereafter, the following workflow was implemented until the end of the project.

4.1.1 Workflow

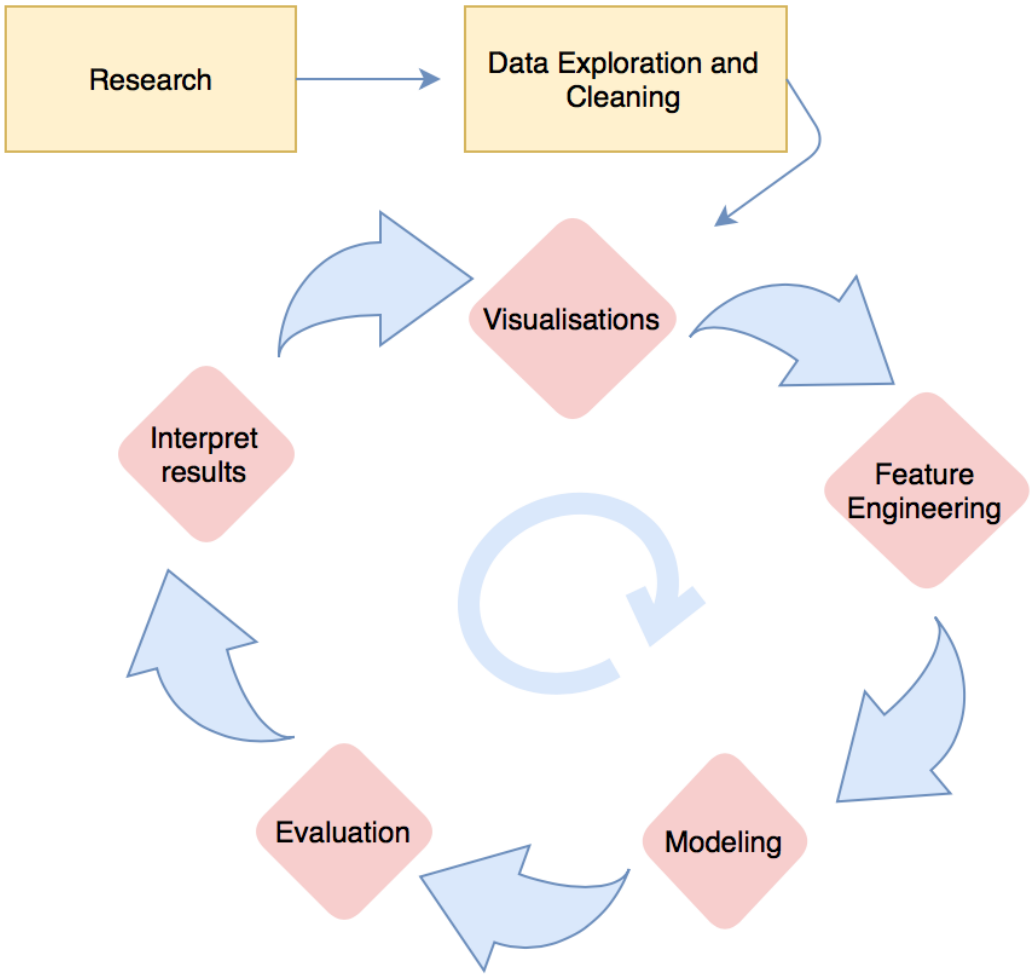

Figure 3 - General workflow following an iterative process of analysis, feature engineering and modelling.

The figure describes how the iterative workflow process was executed. The beginning stage was getting to know the competition as well as general research on rental apartments. Thereafter the data was collected, explored and cleaned. This step will be later explained more thoroughly. The cycle shows how an attempt is made get useful insights from the data, which is then visualised, evaluated and the results interpreted. The cycle was repeated as many times as was necessary to achieve a satisfactory result.

4.1.2 **Tools and techniques**

The project is written in Python, originally by using the Jupyter Notebook. This was later changed to the IDE Spyder to better facilitate version control and backups as Jupyter was found to be problematic in this area. Both Jupyter and Spyder allow for code snippets to be run. This made testing smaller pieces of code easier as the whole script didn't need to run each time a change was made. In particular time intensive scripts such as loading, parsing and cleaning the dataset. Main third-party libraries which were used are SciKit learn and Pandas.

4.1.3 **Version control and backup**

A GitHub repository was created enable coding cooperation and as a standard coding practice. This made it easy for everyone to have the most up to date code and that all code would be backed up. This means that the code can be rolled back if any errors occurred or if an earlier version had code that was needed.

5 Data

5.1 Data Collection

The data is comprised of details about apartments for rent in New York City, USA. The raw data included the following features:

Table 1 - Initial Data made available.

Field	Type
bathrooms: number of bathrooms	Integer
bedrooms: number of bathrooms	Integer
building_id	ID
created	Date
description	Text
display_address	Text
features: a list of features about this apartment	Text (multiple categories)
latitude	Real number
listing_id	ID
longitude	Real number
manager_id	ID
photos	URLs and .jpg files
price: in USD per month	Real number
street_address	Text
interest_level: **target variable** with 3 labels high, medium, low	Label, Text format

5.2 Initial data exploration and cleaning

5.2.1 Interest level

As described earlier, the interest level is the label of the data. It is the goal feature in predicting what makes a listing more appealing to users.

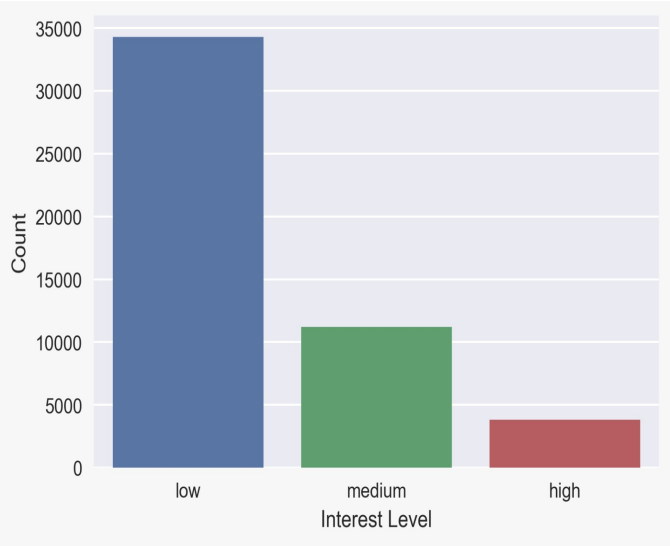

Figure 4 - Distribution of target classes.

Table 2 - Breakdown of target class distribution.

Interest	Count	Percentage
Low	34,284	69%
Medium	11,229	23%
High	3,839	8%
Total	49,352	100%

The vast majority, or 69% of the data, contains low-interest rentals and only 8% is high-interest apartments. This means the dataset is very imbalanced.

5.2.2 Initial Data Summary

Doing a simple summary statistic before any cleaning of the data column showed that outliers were present in location data and price data. This finding is highlighted in red in Table 3.

Table 3 - Summary of numerical data types.

	bathrooms	bedrooms	latitude	listing_id	longitude	price
count	49352.00000	49352.000000	49352.000000	4.935200e+04	49352.000000	4.935200e+04
mean	1.21218	1.541640	40.741545	7.024055e+06	-73.955716	3.830174e+03
std	0.50142	1.115018	0.638535	1.262746e+05	1.177912	2.206687e+04
min	0.00000	0.000000	0.000000	6.811957e+06	-118.271000	4.300000e+01
25%	1.00000	1.000000	40.728300	6.915888e+06	-73.991700	2.500000e+03
50%	1.00000	1.000000	40.751800	7.021070e+06	-73.977900	3.150000e+03
75%	1.00000	2.000000	40.774300	7.128733e+06	-73.954800	4.100000e+03
Max	10.00000	8.000000	44.883500	7.753784e+06	0.000000	4.490000e+06

5.2.3 Positional data: Latitude and Longitude

Some properties were listed with latitude and longitude values of 0 which indicates missing data. Furthermore, the location data for some properties contained values that were unreasonably high or low given a dataset of properties in only New York. Further exploration showed these rentals referred to apartments in other cities such as Las Vegas and Boston.

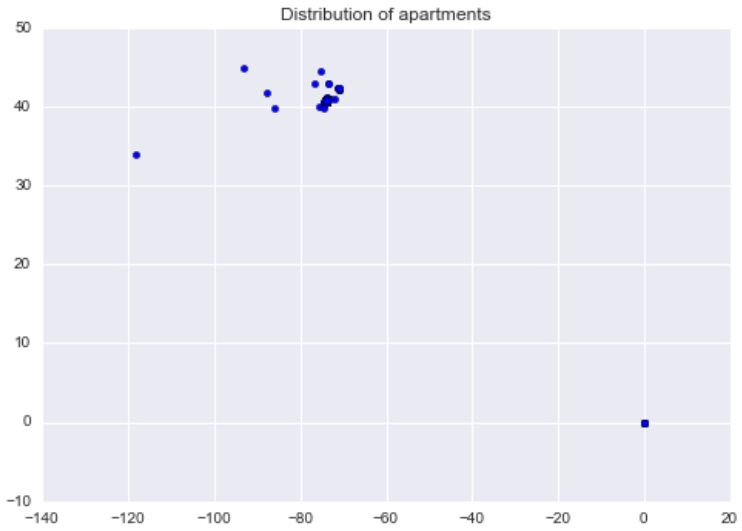

Figure 5 - Map view of apartment locations prior to cleaning.

A common approach to handling data with outliers such as these is removing the top and bottom 1% of the data, which is just under 1000 listings. Although this initial approach removed all outliers, it was found that it removed a rather large amount of valid data. After some further plotting, it was decided to instead find longitude and latitude boundaries of New York (MapDevelopers, 2017) and remove rentals listed outside of it. This method found that only 60 rentals were listed outside of New York.

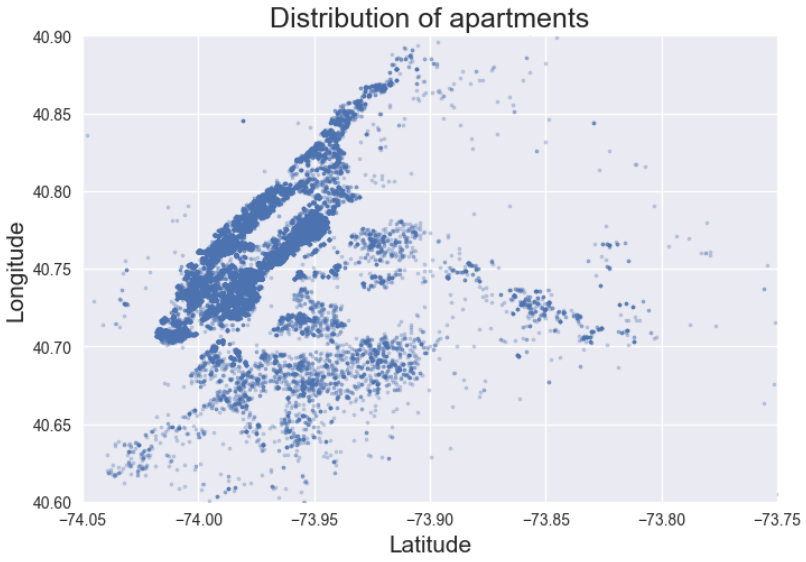

Figure 6 - Map view of apartment locations after cleaning coloured by interest level.

5.2.4 Price

The price data had listings ranging from $40 - $4.5 million per month. This is both unreasonably low and unreasonably high for a rental apartment, even for one of the most expensive cities in the world. Visualising the data clearly showed outliers present. After discarding outliers by removing the top and bottom 1% of the data, what is left is a typical right-skewed distribution.

Figure 7 - Plot illustrating the outliers in the price data.

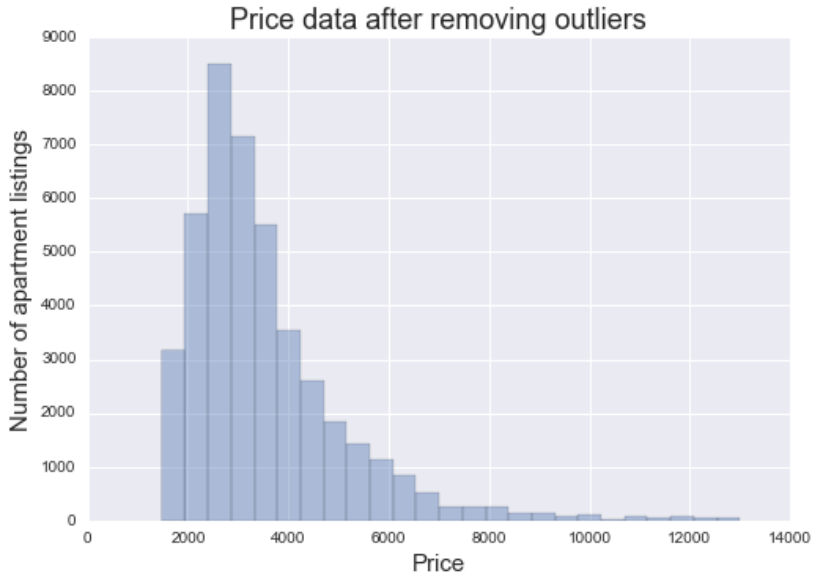

Figure 8 - Plot illustrating the price after outliers are removed.

6 Exploratory Data Analysis

This phase of the project was an extension of the initial data review and QC stage and involved investigating the dataset thoroughly to discover correlations and patterns which would help to guide the feature engineering and modelling processes. Primarily it was important to find if there was a strong relation between potential modelling features and the target variable, as well as the nature of that relationship.

6.1.1 Price per Month

Investigating how the price of apartments varied with the level of interest revealed an interesting trend, where lower price properties were generally associated with higher levels of interest, this is shown in Figure 9. Additionally, the distribution of prices within each interest level can be seen, with the high-interest group having a smaller spread than the medium or high-interest groups. The very high-cost apartments can also clearly be seen as a small peak in the low-interest category.

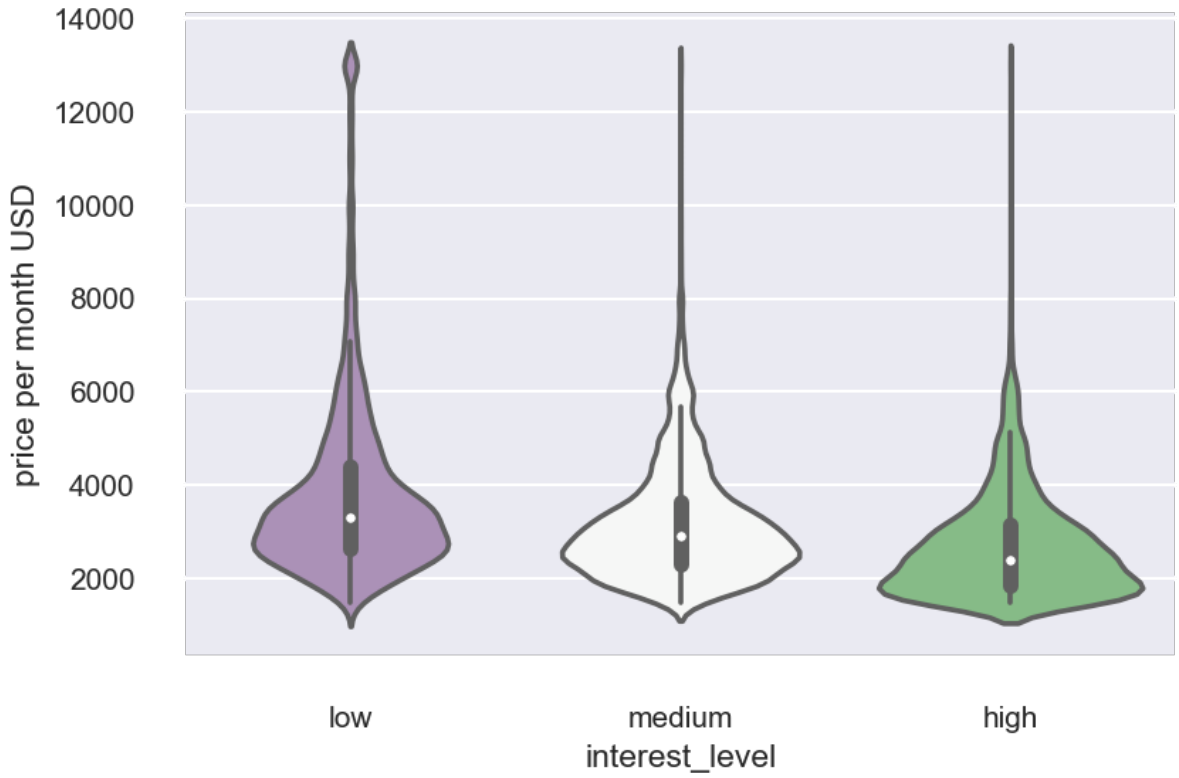

Figure 9 - Violin plot showing the distribution of rental price per month (USD) for the different interest levels.

6.1.2 Bedrooms

Analysing the data on the number of bedrooms revealed that the majority of apartments were single bed shortly followed by 2-bed apartments. These also contained the largest groups of medium and high-interest apartments, potentially indicating that individuals or couples may be the predominant demographic searching for property on the RentHop site. Similarly, the very low drop seen going from 3 to 4 bedrooms may indicate a lack of large families looking for apartments on the website.

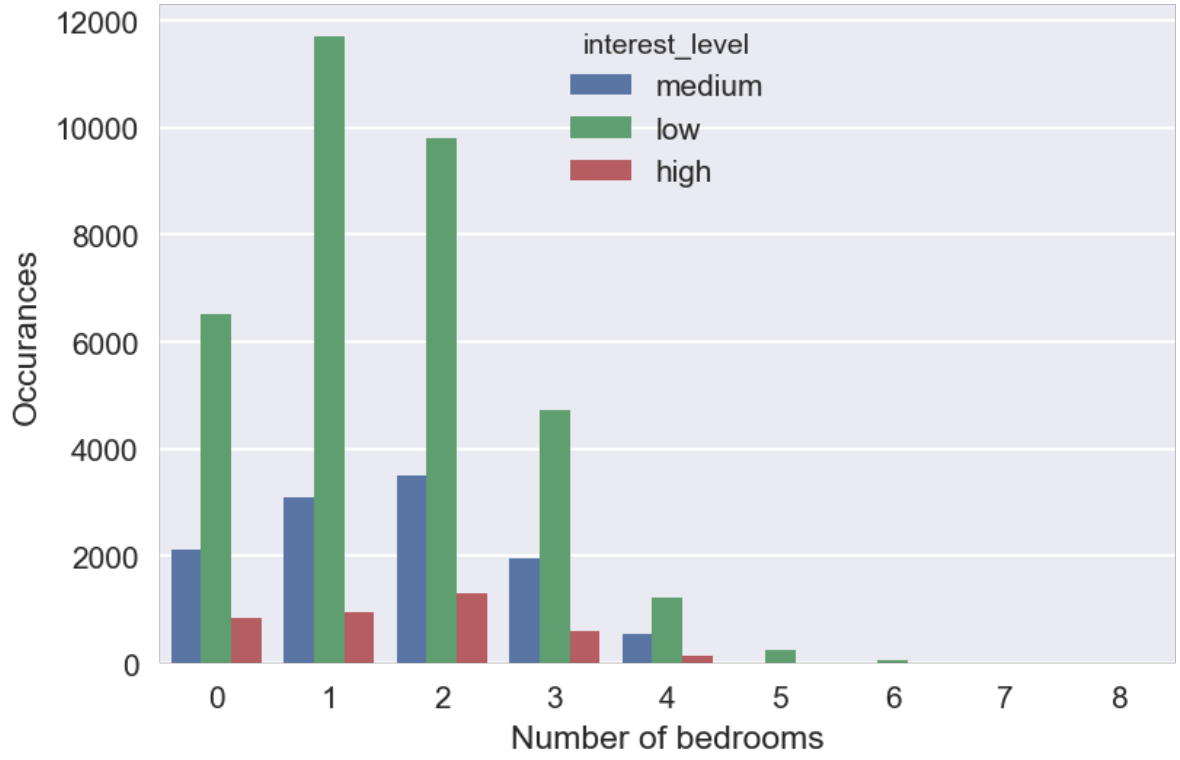

Figure 10 - Plot of occurrences of apartments with different numbers of bedrooms grouped by interest level.

6.1.3 Bathrooms

Compared to the number of bedrooms in apartments, the number of bathrooms appears to be much less informative as a feature. The vast majority of properties contain only 1 bathroom with a smaller but not insignificant amount containing 2 bathrooms. Due to the low spread of data, there is little association seen between the number of bathrooms and the level of interest.

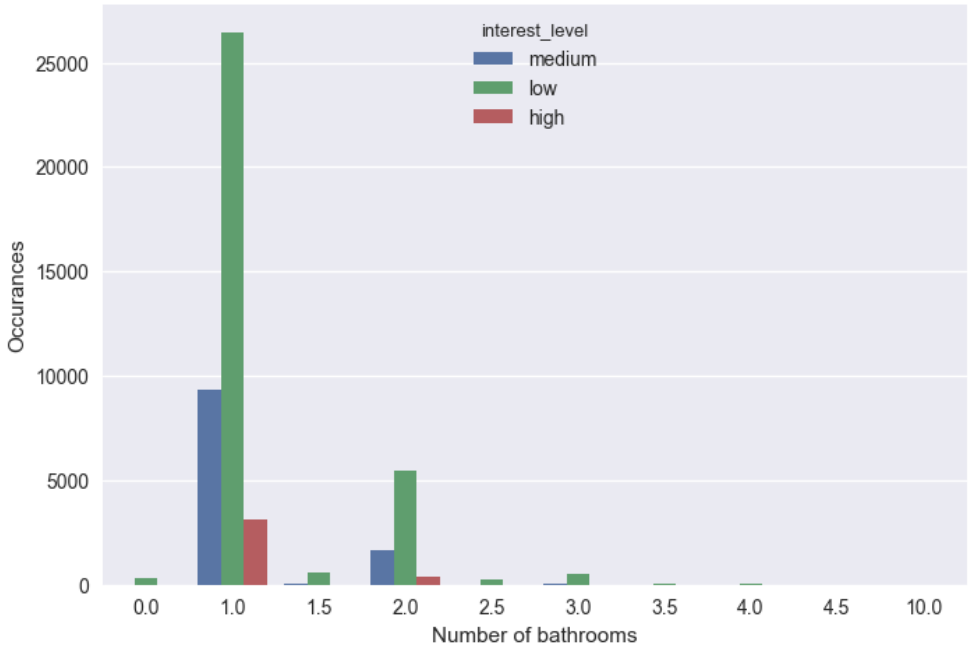

Figure 11 - Interest levels by number of bathrooms

6.1.4 Geospatial

Viewing apartment locations by their interest levels reveals clear spatial distributions. These indicate that there are certain areas that many RentHop users are predominantly interested in, though primarily in Manhattan. This shows that relation to certain boroughs in New York may be a strong predictor of interest levels.

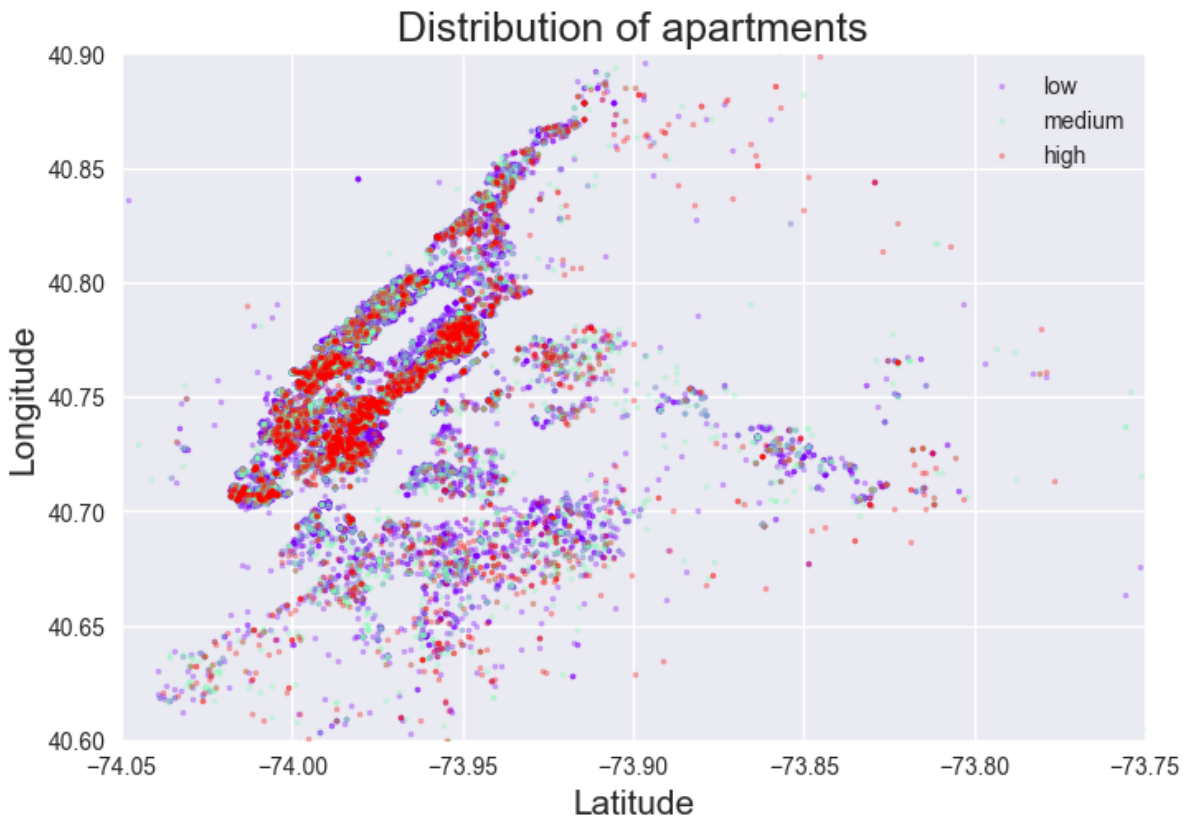

Figure 12 - Distribution of apartments by location

6.1.5 Time data

Given that the time data provided only included when the property listed was made it was expected to show little correlation with the level of interest. While the absolute date and time showed little correlation with the level of interest (given that the dataset was collected over a 2-month period) there were some weak patterns when looking at the day of the week and hour the post was created. Figure 13 and Figure 14 show the number of apartments posted on given days of the week and at given hours. Surprisingly the majority of posts occur during the night, which could potentially be due to the way RentHop creates apartment postings. Plotting the day of the week along with a number of listings and interest levels shows a small peak in the number of listings on Tuesday and Wednesday but with no significantly obvious pattern in the interest levels.

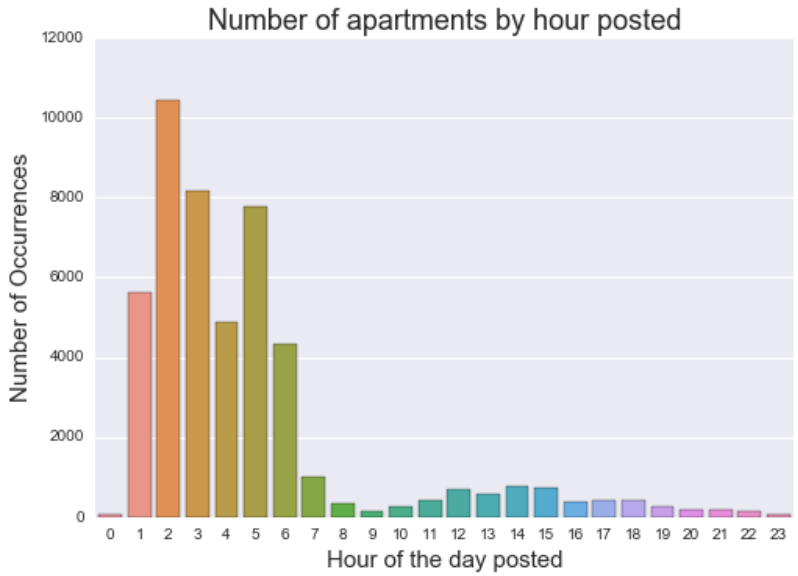

Figure 13 - Number of apartments by hour posted

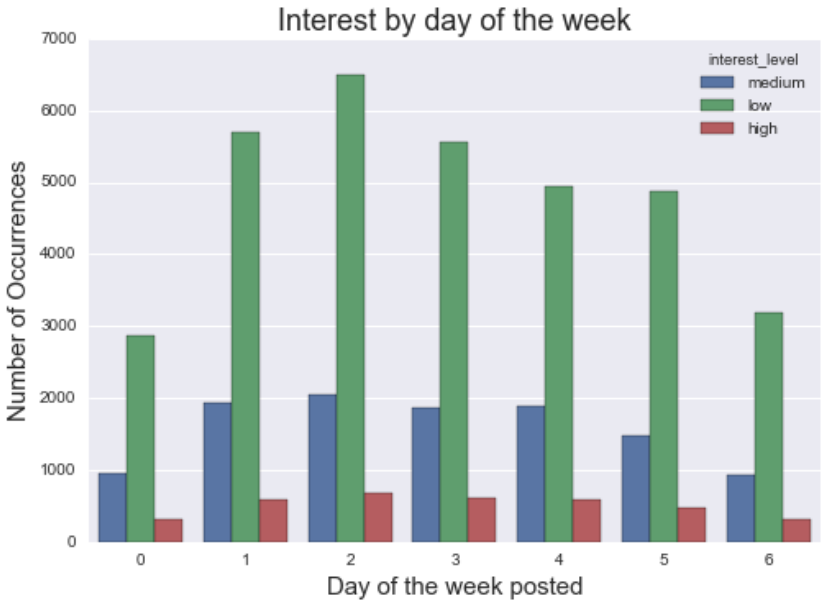

Figure 14 - Interest by day of the week.

6.1.6 Apartment Photos

While the actual image data was not fully explored, the associated metadata (i.e. the number of images) showed that a significant number of properties were listed without any images, with the median being 5 images posted. This also showed that apartments with no photos got very little attention.

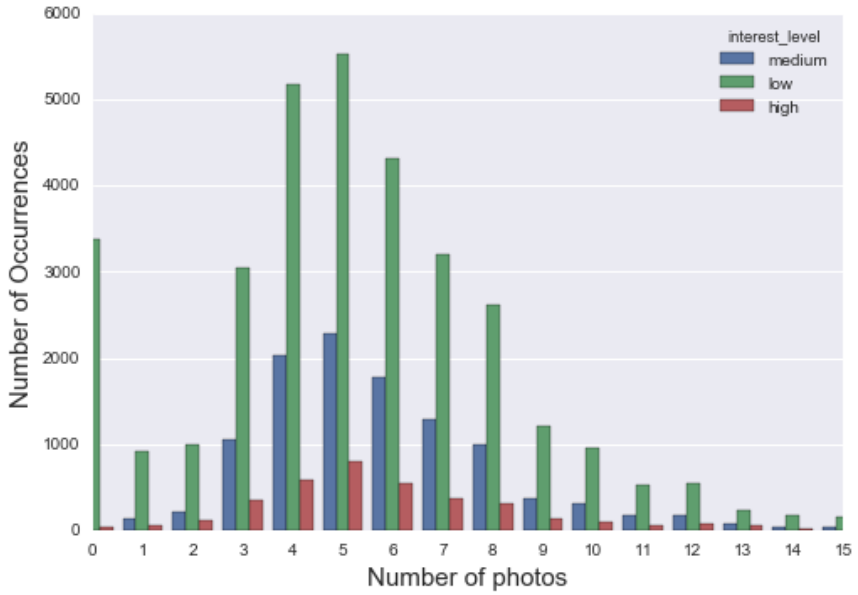

Figure 15 - Number of photos in advert

6.1.7 Apartment features

The number of descriptive features of an apartment is shown to peak at around 3 images. However, there is no strong discernible pattern with the related levels of interest.

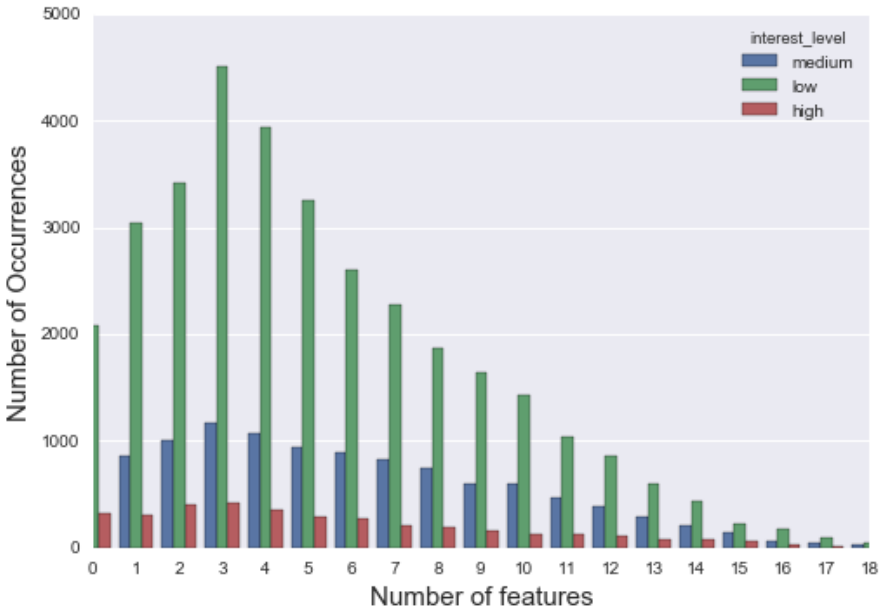

Figure 16 - Interest levels by number of features

7 Features Engineering

Feature engineering involves transforming and combining features in an attempt to produce better representations of the dataset for the purposes of modelling.

7.1 Feature extraction from the description

Figure 17 The most commonly used words in the description

The description potentially contains important features. Unfortunately, these features are qualitative text that is not suited for a machine learning model. Therefore, the qualitative text had to be converted into quantifiable data to feed the algorithm. A simple feature derived was a count of a number of words in the description as the individual features didn't seem to have much say in the prediction. The length of the description could indicate that a broker spent a good amount of time describing the apartment.

7.1.1 Extract individual brokers quality

Some brokers had more than one listing. Therefore, an interesting feature is to assess the quality of the broker by extracting the number of listings the broker has published and then giving an average score for the listings published by the broker.

7.1.2 Extract individual buildings quality

This feature is similar to the individual broker quality feature but instead, gives a score to buildings. For example, if a building complex is popular, it will get a good score in the feature set due to its popularity. Thus, if new listings in the same building appear, it might be able to classify based on the buildings past popularity.

7.1.3 Price feature to get apartment size

The dataset comes monthly price in USD but does not have information about its size. Therefore, by using data about the number of bedrooms and bathrooms derived those features to say something about the size of apartments in relation to price: Price-per-bedroom.

7.1.4 Using pictures as a feature

The dataset contains a large number of photos that can be used to derive insight into the data. Unfortunately, deriving information from the pictures as an enormously large job. Based on findings from other people working on the same task it was decided not to attempt to derive information from the pictures directly but instead count the number of pictures in each listing.

Another approach could have been to try to do *simple* computer vision and derivative features such as colours, brightness, saturation etc. This approach has been tested by some on the project forum at Kaggle, and the result has very little to say for final prediction. The return on time invested for such an analysis is therefore not necessary to get a good prediction.

7.1.5 Deriving features from the 'feature' column

Each listing contains some check boxes the users can select for the listing; these are features such as pets allowed or what is included in the rent or in the building. A number of *features* selected in each listing were used as a feature for the prediction model.

Figure 18 Word cloud extracted from the feature column and shows the most common words in the features.

7.1.6 Location features

New York consists of different neighbourhoods, and these can impact the popularity of a listing. Therefore, each listing was clustered into a neighbourhood, this was done by using the Euclidian distance. The neighbourhood clusters used is The Bronx, Staten Island, Manhattan, Queens and Brooklyn. Additionally, the longitude and latitude itself were left in as features for each listing.

7.1.7 Time features

The dataset contains information about when a listing was published, this information was used to create the features *day_created, day_of_week_created* and *hour_created.* This was done because the time a listing is published has the potential to impact the listings popularity.

8 Modelling and Inference

Following the analysis of the dataset, the primary task was to build a predictive model. As discussed previously, the desired target was user interest which had been supplied in the original dataset (termed interest_level). Interest levels fell under one of 3 classes, low, medium, and high making this task a multi-class classification problem, and as the labels were provided supervised methods could be used.

8.1.1 Data Preparation

Proper procedure for modelling involved splitting the available data set into three subsets; training, validation and test. This is important in the modelling process as there is a significant risk of overtraining to the available data, which would result in the model being less generalised. By having separate validation and testing sets, the model can be evaluated on data that have not yet seen and a truer measure of its performance can be attained.

The given split was 90:10 between the training and test sets, then the training set was subsequently split 90:10, for training and validation respectively. Each split was carried out in a random manner using a preset seed for repeatability.

One of the primary concerns with the dataset was the large imbalance between classes, and therefore the type of split had to be carefully considered to ensure the training, validation and testing set all had the same relative representation of each class.

Additionally, each feature was scaled based on the statistics of the training set to 0 mean and unit variance for numerical variables. This is an essential step for a number of machine learning algorithms to prevent coefficients of smaller magnitude features from shrinking till they have no influence on the end model.

8.1.2 Evaluation Metrics

A suitable evaluation metric was required as the basis for model comparison, the selection of which required consideration for the type of modelling problem as well as the nature of the data. Simple evaluation metrics such as classification can be misleading with imbalanced datasets as a naive approach of labelling all data as the dominant class may result in a very high score but would not be conducive to a useful model.

It was decided that log-loss would be used as a single comparison metric when comparing improvements of the model. Log-loss is particularly useful as it penalises the model to a larger degree when an incorrect classification is made with a greater confidence (higher estimated probability). Additionally, the multiclass form of log loss is the sum of the individual loss of each class (Scikit Learn, 2016), so poor performance in underrepresented classes will not be overlooked completely.

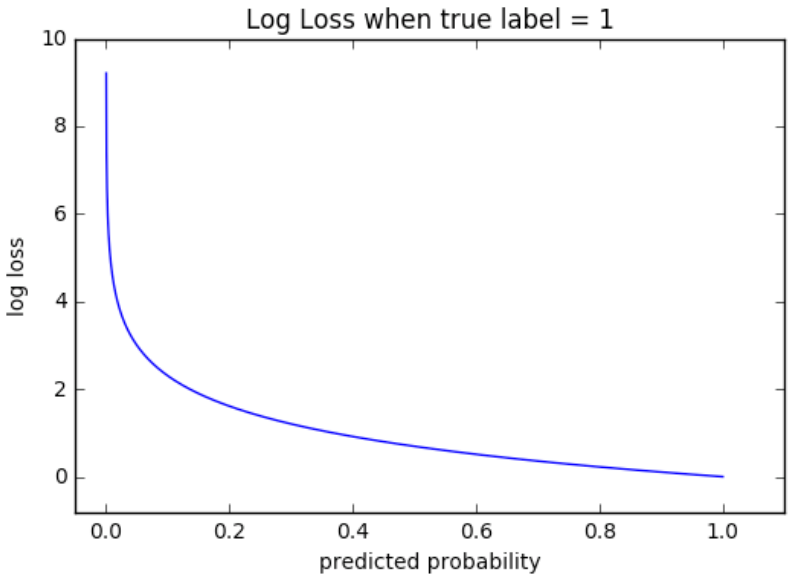

Figure 19 - plot showing the change in log loss as the predicted probability approaches 1 (assuming the true label is 1).

In addition to the log-loss metric the F1 score (as well as the component precision and recall scores), and confusion matrices were also calculated were also used to provide a clearer view of the class wise predictive power of a given model. (Scikit Learn, 2016)

Table 4 - Evaluation metric formula.

Metric	Formula	Details
Log-loss	$logLoss = \frac{-1}{N}\sum_{i=1}^{N}(y_i(logp_i) + (1-y_i)log(1-p_i))$	y = true class label (1 or 0) p = probability N = no of examples
Precision	$P = \frac{T_p}{T_p+F_p}$	Tp = True Positive Fp = False Positive
Recall	$R = \frac{T_p}{T_p+F_n}$	Tp = True Positive Fn = False Negative
F1 Score	$F1 = 2\frac{P \times R}{P+R}$	P = Precision R = Recall

8.1.3 Models

The general approach to modelling involved application of a number of different algorithms with the full complement of engineered features. Models were selected included Logistic Regression, Random Forest (decision tree ensemble), and Neural Networks. This followed the notion that with a rich enough feature representation only a simple model is required (e.g. Logistic Regression), however in lieu of good features, utilising an algorithm such as neural nets can potentially compensate for this.

8.2 Benchmarking

A benchmark was created early on in the process to provide a metric with which further work could be compared to. The benchmark model used was a multi-class (one vs all) Logistic Regression Classifier, which was selected due to its simplicity. The features used for the benchmark only included the numerical features supplied in the raw dataset after initial QC had been carried out. These included bedrooms, bathrooms, latitude, longitude, and price. The primary metric used to determine the model quality was log loss (as discussed in the Evaluation section) however, a number of other metrics were also produced.

Table 5 - Model Information (benchmark).

Features	bedrooms, bathrooms, latitude, longitude, price
Model	Logistic Regression
Log-loss Train	0.823408
Log-loss Test	0.817069

Table 6 - Test set precision/recall and F1 score (benchmark).

	precision	**recall**	**f1-score**	**support**
low	0.78	0.82	0.8	3377
medium	0.39	0.25	0.3	1121
high	0.26	0.43	0.32	383
avg/total	0.65	0.66	0.65	4881

Table 7 - Test set confusion matrix (benchmark).

prediction > true V	**low**	**medium**	**high**
low	**3025**	129	275
medium	813	**90**	220
high	208	33	**143**

While the log-loss metric is not easily interpretable with only the benchmark on a class by class basis, the confusion matrix clearly shows that the majority of medium and high-interest properties are being mislabelled as low-interest properties. This is likely due to the highly imbalanced nature of the data set as discussed previously that the simple feature representation and model cannot account for.

8.3 Model Optimisation

After benchmarking, the modelling process went through several iterations to improve the score. This included further features engineering and selection, parameter tweaking/experimentation as well as exploring alternative models. This was guided by the training and validation dataset, with the test set being used sparingly to minimise selection bias.

8.3.1 Feature Selection

The first improvement to the model and typically most important in any modelling process is the representation of the dataset, i.e. features given or generated from the raw data. Details on features and feature engineering have been discussed in section 6.1.7.

The newly generated features were used as further input to the baseline model and the performance change measured. Additionally, the importance of each feature was measured using a random forest model, this is shown in the figure. The relative importance of each feature was then used to determine its potential usefulness in modelling as some methods are sensitive to the number of features included. This was generated using the Extra-Trees Classifier (SciKit Learn, 2016) which finds the mean importance of each feature over the number of trees used in the model (1000 trees were used in this instance). It was decided any feature below a threshold of 0.04 mean importance would be discarded from the modelling process.

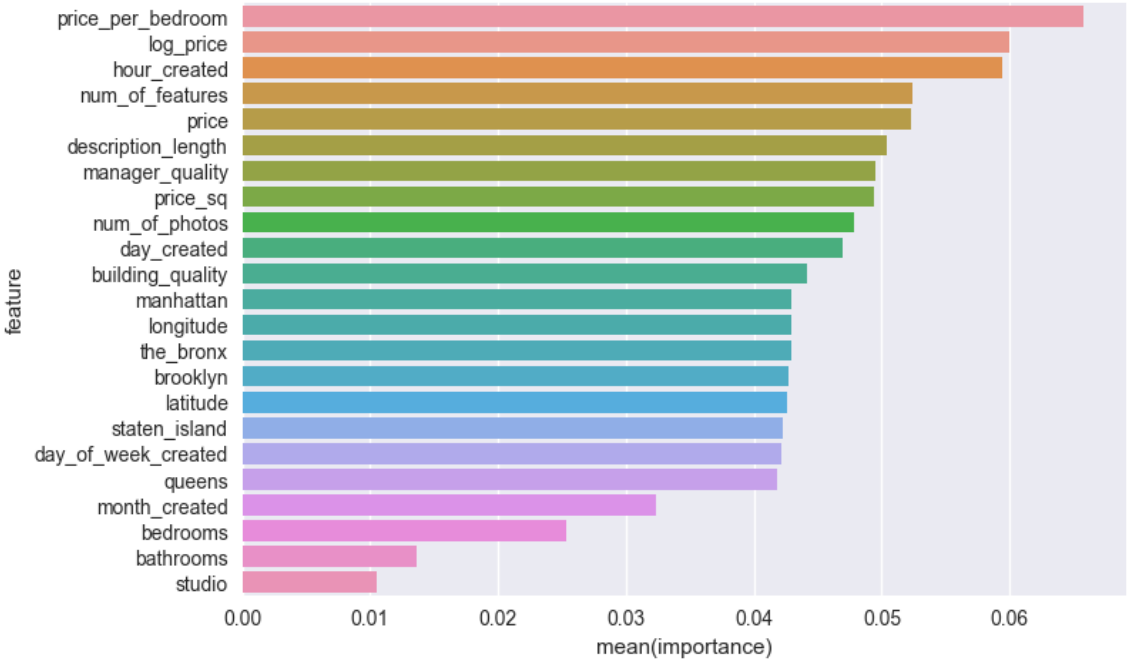

Figure 20 - plot showing the relative importance of each feature.

With the newly created list of features, the benchmark model was tested again to see what performance increase was gained from the feature engineering process.

8.3.2 Benchmark with Engineered and Selected Features

Table 8 - Model Information.

Features	latitude, longitude, price, log_price, price_sq, the_bronx, staten_island, manhattan, queens, brooklyn, num_of_photos, price_per_bedroom, description_length, num_of_features, day_created, manager_quality, building_quality, hour_created, day_of_week_created
Model	Logistic Regression (same as benchmark)
Log-loss Train	0.690451
Log-loss Test	0.815249

Table 9 - Test set precision, recall, and F1 score (benchmark with full features).

	precision	**recall**	**f1-score**	**support**
low	0.78	0.86	0.82	3429
medium	0.41	0.25	0.31	1123
high	0.31	0.37	0.34	384
avg/total	0.66	0.69	0.67	4936

Table 10 - Test set confusion matrix (benchmark with full features).

prediction > true V	**low**	**medium**	**high**
low	**2961**	315	153
medium	687	**280**	156
high	157	86	**141**

Comparing the new result to the benchmark a very small to insignificant improvement (0.002) can be seen in terms of the test log loss, whereas the train log loss reduced by an appreciable amount (0.13 reduction). This is likely due to overfitting to the training set which can lead to a decrease in the performance on the test data.

8.3.3 Model Selection and Parameter Tuning

- Aside from including better features, the other primary method of improving prediction was to compare various types of classification models and optimise their associated parameters. The models that were selected for comparison included:

Random Forest Classifier, Neural Network, Logistic Regression.

In each case, a number of parameters were optimised based on a range using 5-fold cross validation to minimise the risk of overfitting to the training dataset. A range of values for each parameter was given and searched over in combination for each model until the best was found (based on log-loss).

Table 11 - Model parameter information.

Model	**Parameters Tuned**
Logistic Regression	• regularisation
Random Forest	• number of trees • tree depth
Neural Network	• learning rate • network architecture • activation function

After optimising the parameters of each model, the result of the test set was used to determine which performed best. Figure 21 shows the distribution of scores using 5-fold cross validation (with the training set data). It clearly shows that the performance of the neural net and random forest classifier outperform the logistic regression classifier by a significant margin. Additionally, the neural net is shown to perform slightly better than the random forest classifier.

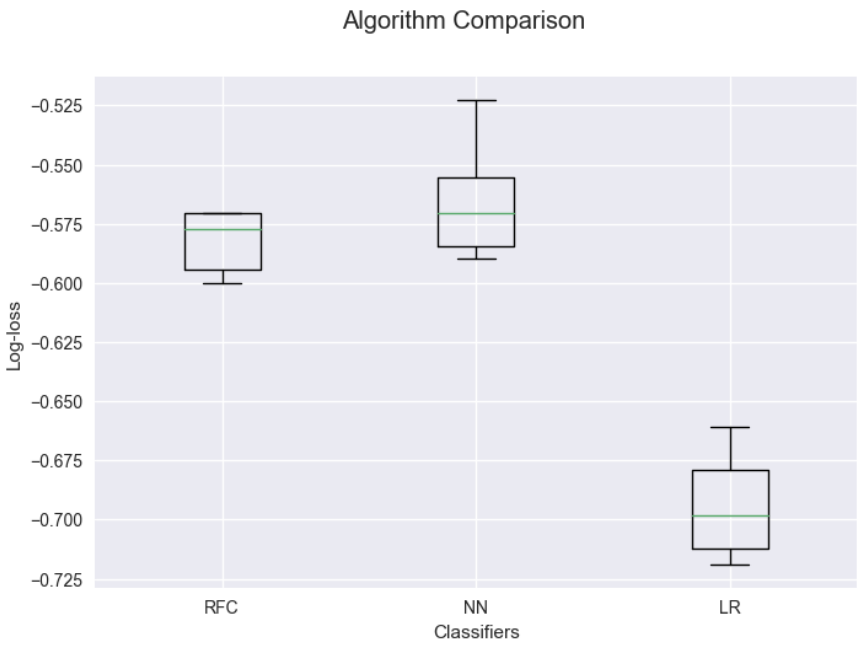

Figure 21 - Comparison of different classifiers over 5-fold cross validation.

The log-loss measure for the test set was also calculated (Table 12). As with the cross-validation on the training set, the neural net is shown to outperform the random forest classifier and Logistic regression classifier.

Table 12 - Log loss for each method on the training and test set.

Model	Train Log-loss	Test Log-loss
Logistic Regression	0.690451	0.815249
Random Forest	0.729006	0.791998
Neural Network	0.548426	0.686575

8.4 Final Model

Based on the performance metrics shown in the model selection section (8.3.3), the neural network model was shown to give the best results. The details of the model are listed below.

Table 13 - Final Model description.

Features	latitude, longitude, price, log_price, price_sq, the_bronx, staten_island, manhattan, queens, brooklyn, num_of_photos, price_per_bedroom, description_length, num_of_features, day_created, manager_quality, building_quality, hour_created, day_of_week_created
Model	Neural Network
Model Parameters	Learning Rate: 1E-6 Network structure: 10-30-5 Activation function: tanh
Log-loss Train	0.548426
Log-loss Test	0.686575

Table 14 - Test set precision/recall and F1 score.

	precision	**recall**	**f1-score**	**support**
low	0.77	0.91	0.84	3429
medium	0.45	0.3	0.36	1123
high	0.47	0.16	0.23	384
avg/total	0.68	0.72	0.68	4936

Table 15 - Confusion matrix for the test set using the neural network model

prediction > true V	**low**	**medium**	**high**
low	**3134**	271	24
medium	742	**337**	44
high	184	140	**60**

8.4.1 Model Assessment and Usage

While the final model provided the best performance in terms of the log loss-metric, it still performed relatively poorly in classifying the under-represented classes (medium and high). This can be seen in the confusion matrix where the medium interest properties are often classified as low-interest properties, and high-interest properties are misclassified as both medium and low.

While the misclassification rate is high resulting from this, the reason for the relatively low log-loss is due to the low confidence associated with the predictions. In practical terms, the per class probability would provide a better assessment of which class the property belongs to over the maximum probability. For example, the output of a property may result in the normalised class probabilities of low: 0.4, medium: 0.3, high: 0.3. A property such as this would be classified as a low-interest property. However, the confidence in that classification would be low.

Example output:

'Low' probability	'Medium' probability	'High' probability
0.630708	0.311329	0.579633
0.876721	0.113089	0.101895
0.939759	0.590345	0.120648

9 Challenges and Success

9.1 Data Leakage

Data leakage is the notion of information about the target variable being included in the features being used for prediction. This is an often overlooked issue with modelling which potentially leads to an overconfidence in the performance of a model. Certain features created in the process of this project were potential avenues for data leakage and appropriate actions were required to ensure the test data was not contaminated.

Figure 22 illustrates this issue of data leakage when precautions are not met, showing a very high importance for the building quality feature. The score in this instance has been inflated as the feature was created using information about the interest level (the target variable). While it is an acceptable feature to include, when creating it for the test set, only information from the training set should be used (R, 2013).

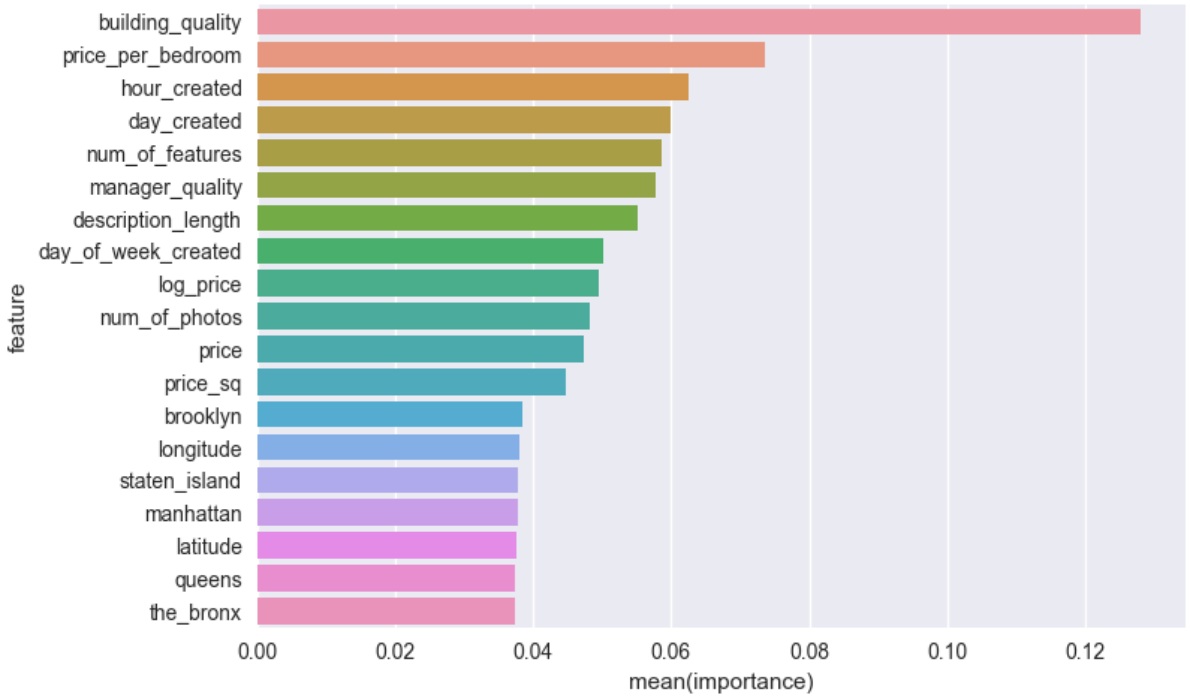

Figure 22 - feature importance plot illustrating the issue of data leakage.

9.2 Class Imbalance

One major challenge with the RentHop data set which has been a running consideration throughout this report has been the imbalance in the target classes. This had a significant impact on the way the data had to be handled in terms of training and testing, the selection of evaluation metric, as well as the choice and overall performance of the final model. A number of approaches could have been considered to compensate for this imbalance including generation of new data or creating a hierarchy of classifiers.

9.3 Collaboration on the code

During the initial phase of the project, it was decided to build the model and do an exploratory data analysis in Jupyter Notebook to make it easier to run cell by cell. Unfortunately, this approach made it difficult to share code and collaborate as the Python Notebook is not supported by Git version control. Therefore, after spending a good amount of time on getting this to work the project was converted into regular python files and continued by using Spyder. This made it possible to utilise GitHub to the fullest and easier collaborate on the code.

10 Conclusions and Further Work

The main deliverable for this project was the predictive model. While in some instances it performed well the primary drawback was the poor classification quality for the high interest apartments. However, the neural network was quite accurate in predicting low interest apartments and the raw probabilities provide better insight than the absolute classification scores.

This could be used by Two Sigma to help the landlords that post to the site, that their apartment listing is showing behaviours of an apartment that receives low interest. It could then highlight the features that had the largest impact on the prediction to make recommendations how the landlord could improve their listing, and drive actionable decisions.

There is still room for improvements however. As stated in the model section, rudimentary computer vision techniques could be utilized to analyse the photos posted to the apartment listings. To analyse elements such as lighting, hue and contrast. This could give an indication as to what types of photos might lead to more interest. Another improvement would be to do sentiment analysis and other natural language processing on the description. As well as a richer features, the model may be further turned by including a larger parameter search space, and data augmentation may help to alleviate issues such as the large class imbalance.

11 Bibliography

Lin, L., 2017. *Hello from RentHop -- and why brokers and chess players are similar.* [Online]
Available at: https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries/discussion/28604
[Accessed 1 3 2017].

MapDevelopers, 2017. *MapDevelopers.* [Online]
Available at: https://www.mapdevelopers.com/geocode_bounding_box.php
[Accessed 31 03 2017].

RentHop.com, 2017. *RentHop.* [Online]
Available at: https://www.renthop.com/agent-guide/the-hopscore
[Accessed 31 03 2017].

R, R., 2013. *Kaggle.* [Online]
Available at: https://www.kaggle.com/wiki/Leakage
[Accessed 31 03 2017].

Scikit Learn, 2016. *Scikit learn.* [Online]
Available at: http://scikit-learn.org/stable/modules/generated/sklearn.metrics.log_loss.html
[Accessed 5 3 2017].

SciKit Learn, 2016. *SciKit Learn.* [Online]
Available at: http://scikit-learn.org/stable/modules/generated/sklearn.tree.ExtraTreeClassifier.html
[Accessed 05 3 2017].

Scikit Learn, 2016. *sklearn.metrics.f1_score¶.* [Online]
Available at: http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html
[Accessed 5 3 2017].

Two Sigma, RentHop, 2017. *Two Sigma Connect: Rental Listing Inquiries.* [Online]
Available at: https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries
[Accessed 1 3 2017].

12 List of Appendixes

12.1 Appendix 1

Preprocess.py – Python file with the source code for the pre-processing part of the code

12.2 Appendix 2

eda.py – Python file with the source code for the exploratory data analysis part of the code

12.3 Appendix 3

The_Rent_mainfile.py – Python file with the source code for the main part of the code – including the modelling.

12.4 Appendix 4

Train.json – Json file with the dataset used. This can be found at https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries/data

12.5 Appendix 5

GitHub repository – https://github.com/chriotte/dacwFlatHopRentalPrices