**Regression and Classification with Ames Housing data**



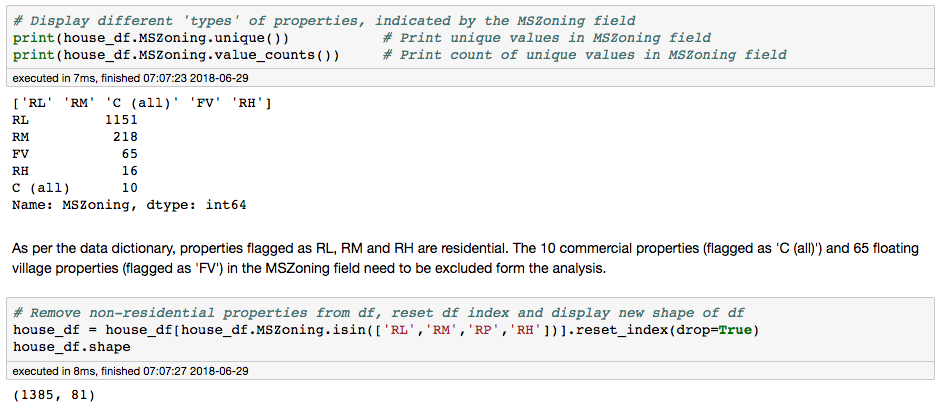
Having recently joined a company in Ames, Iowa that specialises in purchasing existing residential properties, performing cost-effective renovations (if required) and on-selling these properties for profit, I’ve been tasked with supporting the company to optimise investment and maximise return by:

1. Developing an algorithm to estimate the price of residential houses based on fixed features i.e. characteristics that cannot be easily renovated (e.g. location, square feet, number of bedrooms and bathrooms)
2. Identifying characteristics of residential houses that can be cost-effectively renovated and estimating the mean value of these renovations

This project uses the [Ames housing data made available on Kaggle](https://www.kaggle.com/c/house-prices-advanced-regression-techniques), which includes 81 features describing a wide range of characteristics of 1,460 homes in Ames, Iowa sold between 2006 and 2010. Models were required to be trained on houses sold prior to 2010 and evaluated on houses sold in 2010.

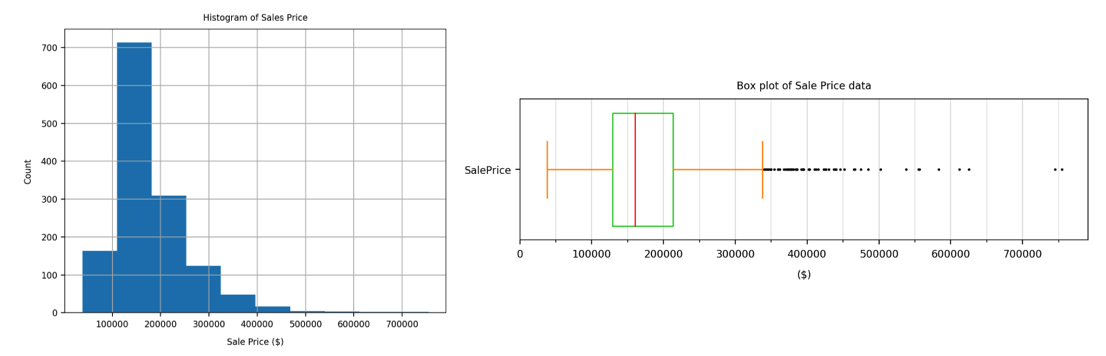
**Step 1: Understand my data**

After considering the problem that was required to be solved, my first step was to understand the available data. Exploratory data analysis (EDA) revealed that not all houses were residential. These non-residential properties needed to be removed from the dataset before continuing EDA.



Initial EDA showed that a number of features had missing values, which would need to be managed appropriately before modelling. Before doing this however, I started by examining the target variable’s (i.e. the SalesPrice) descriptive statistics. The Pandas DataFrameSummary object and the boxplot below, showed:

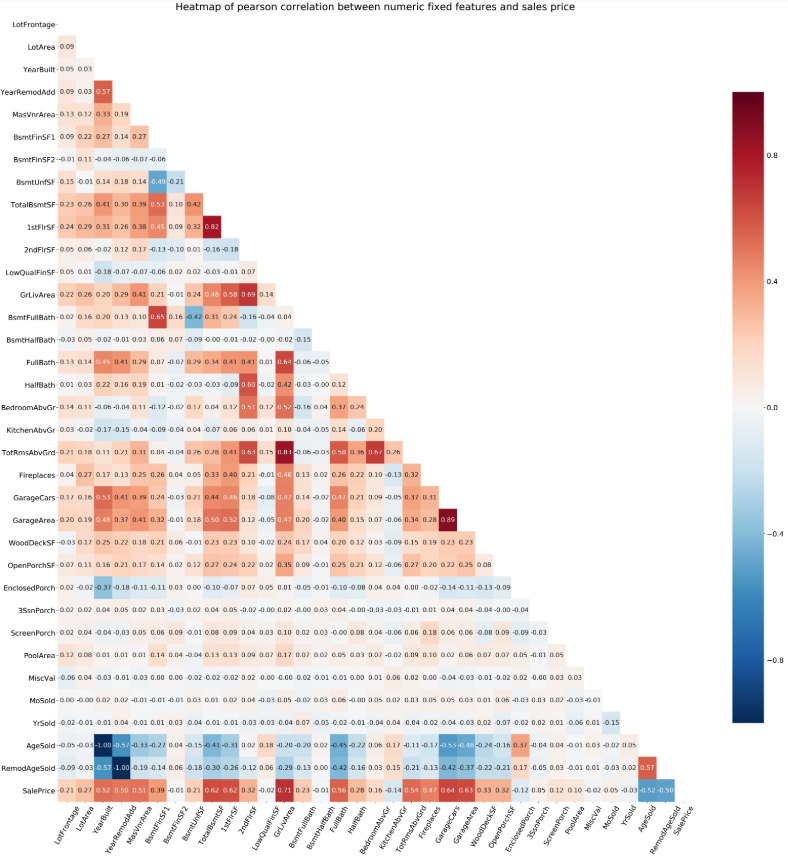
* A mean and median sale price of $180,136 and $160,000 respectively, indicating a positive skew, which was confirmed by a histogram of the Sale Price and positive kurtosis and skewness values
* A sale price range from $37,500 to $755,000, with 61 outliers – defined here as sales price > 75th percentile + 1.5 \* interquartile range. The 95th percentile ($325,925) was also considerably less than the maximum sale price.



**Step 2: Cleaning and EDA of numerical, fixed features to identify features for inclusion in model**

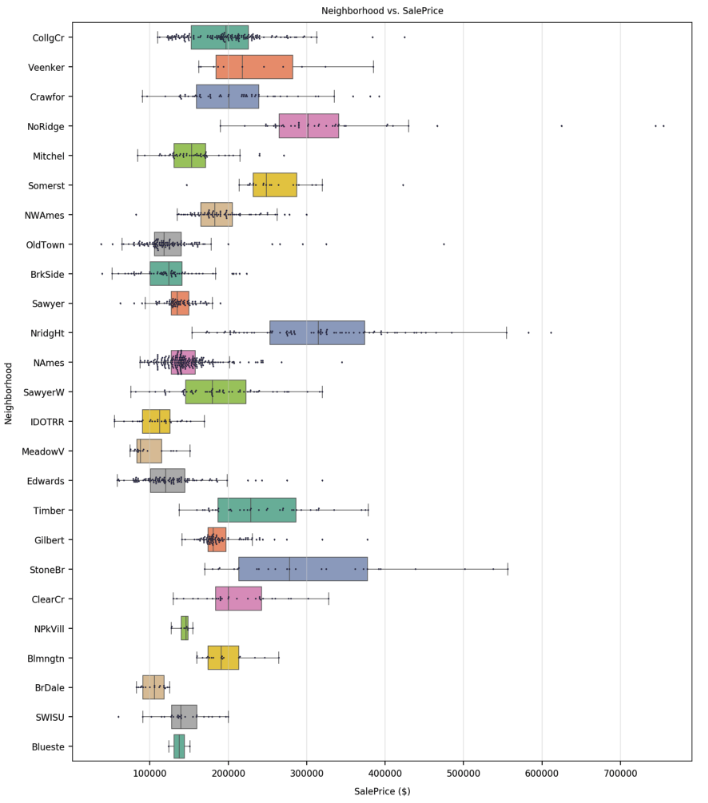
After engineering new features (e.g. the age of the property when sold and the age of the most recent renovation when sold), subsetting the data on relevant features and filling missing values appropriately, I examined the Pearson and Spearman correlations (using a function I developed to create the heatmap displayed below) between the numerical fixed features and the sales price. The objective of this was to identify the most relevant features to include in my regression model.

Using the heatmap below (and additional visualisations), I selected features with a correlation exceeding +/- 0.5. Considering the assumption of independence between predictors for multilinear regression, I then eliminated features with high intercorrelations. I noticed that a lot of my selected features were related to house size, so I then selected some additional predictors for my regression model to reflect specific (non-space related) features that may influence the sale price (e.g. number of fire places, square feet of decking).

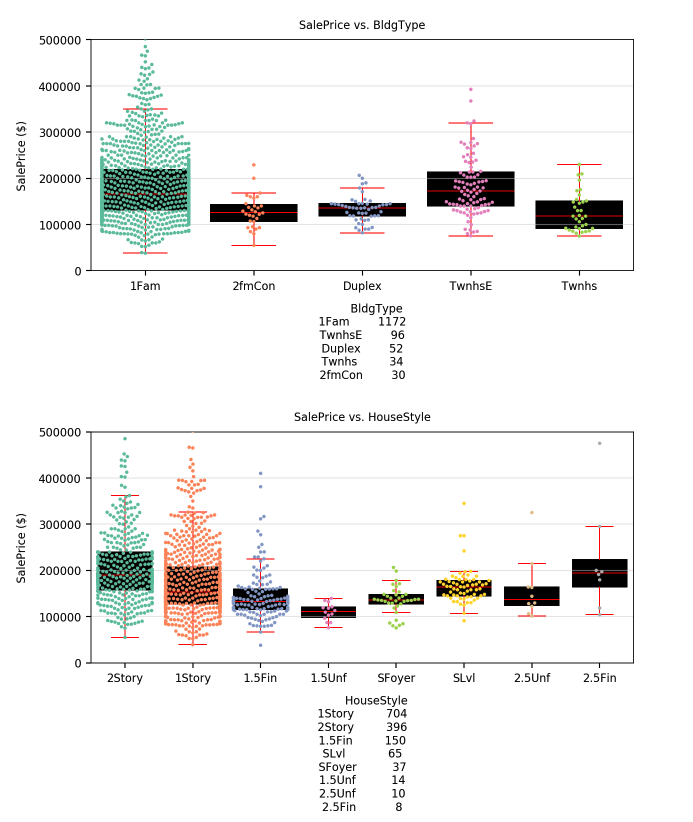
­

**Step 3: Cleaning and EDA of categorical, fixed features to identify features for inclusion in model**

Intuitively, the house’s neighborhood felt like a relevant predictor for Sale Price, so I used a combined boxplot and swarmplot to visualise this relationship. This visualisation illustrated variation in Sale Price and a good distribution in the number of properties across the 25 neighborhoods. I was definitely going to include this fixed feature as a predictor in my model.

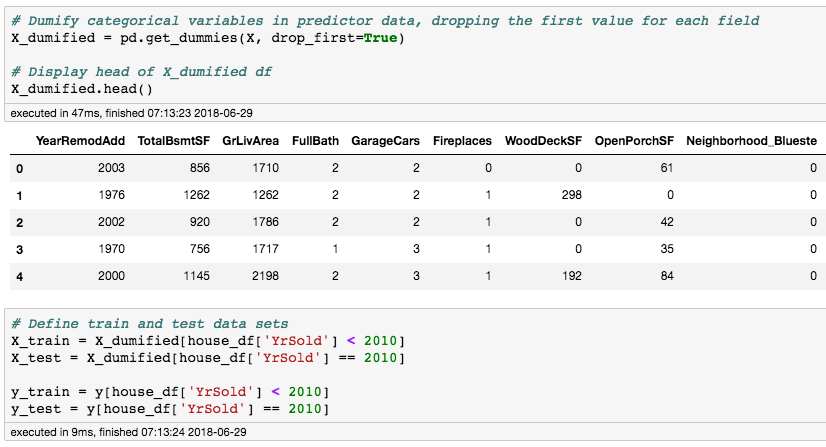


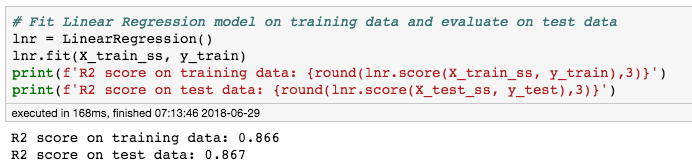
After cleaning the categorical fixed features of interest, I developed a function to generate a combined boxplot and swarmplot to illustrate the relationship between the categorical features and the sales price. The swarmplot – and the inclusion of the value counts in the x-axis (see partial screen shot below) - enabled me to identify the categorical features that exhibited both a variation in sales price and a ‘reasonable’ distribution (in terms of spread of data points) across the different values of each categorical feature, such as the two features displayed here.

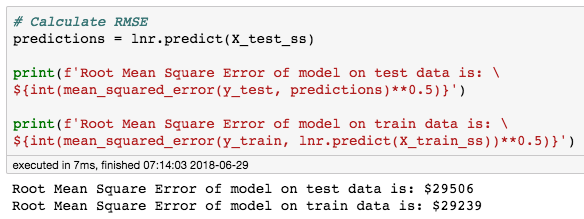


**Step 4: Train and evaluate a linear regression model to predict sales prices using fixed features**

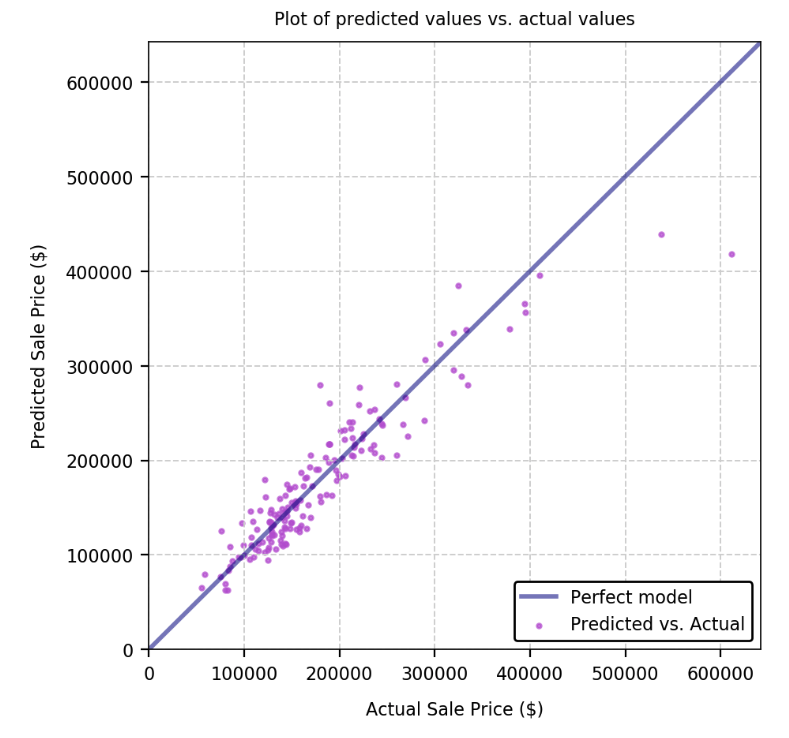
Having identified the numeric and categorical features for inclusion in my model and imputing all missing values for these features, I was then ready to create my regression model to predict the sales price from fixed features only. All categorical features were dumified (resulting in a total of 73 features) and the data was subset into the train (pre-2010 sales) and test (2010 sales) splits.



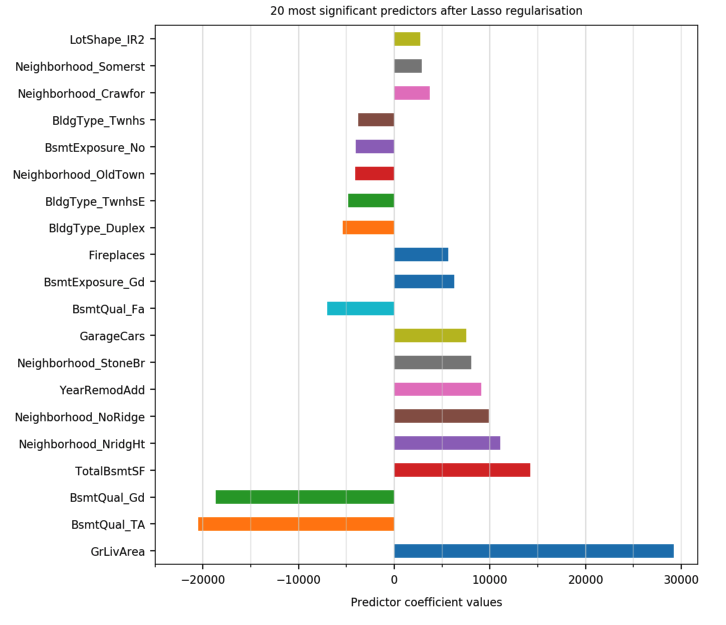
After standardising the train and test data using the mean and standard deviation of the training data only, I started by training and evaluating a linear regression model without any cross-validation or regularisation. This model had an R2 score of 0.867 on the test data, which meant that the predictor variables I’d chosen explained 86.7% of the variance in the target test data not explained by the baseline prediction (i.e. the mean of the sale price). The root mean square error on the test data was $29,506 and the model generalised well from the training data to the test data – a surprisingly good result! 



A scatter plot of the predicted vs. actual values (see below) confirmed that the model was performing well, with just 4 homes reporting residuals exceeding +/- $60,000. The two largest underpredictions were for the 2 highest value properties – easily spotted as the 2 data points with an actual sale price exceeding $500,00 in the chart below.



I tried to improve the model performance by grid searching on the RidgeCV and LassoCV hyperparameters, but was not able to improve on the R2 score above – which I found surprising. Knowing that Lasso regularisation results in a sparser model and deals with multicollinearity between predictors well, I decided to choose the optimum Lasso model yielded by the LassoCV grid search for predicting the sale price of residential houses based on fixed features. This model achieved a very similar R2 score (0.865) to the linear regression model above – but importantly, it zeroed out 22% of my predictors. The 20 largest model coefficients in terms of absolute value, are displayed in the bar plot below.



With a very similar R2 score to the standard linear regression model, the plot of the predicted vs. actual sales prices for the optimum Lasso regularisation model was very similar to the scatter plot above. Since the predictor data had been standardised, the model coefficients above can be interpreted as the monetary value of each feature differenced from its mean value by 1 standard deviation.

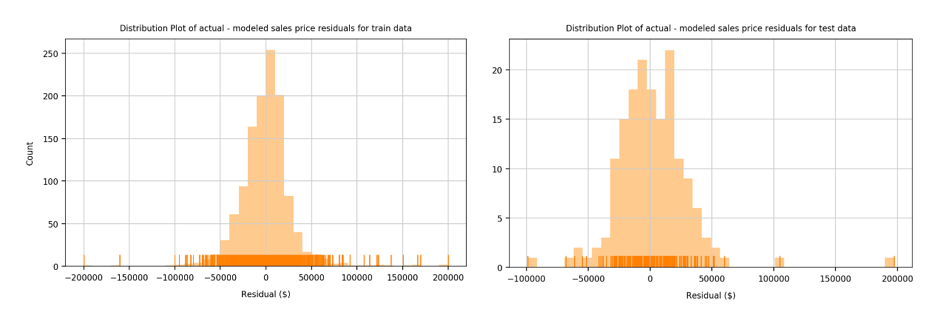
Part 1: Modeling the sales prices of houses based on their fixed features

Part 2: Determining the value of changeable property characteristics unexplained by the fixed features

**Step 5: EDA of residuals from optimum Lasso model for predicting sales price from fixed features.**

Now that I had a model that performed well in predicting the sales price of residential houses based on their fixed features, the next step was to examine how much of the residuals from the optimum Lasso model above could be explained by the changeable features of these properties. This would enable my company to understand the benefits of renovations to quality and condition related features.

I started by reviewing the residuals from the optimum Lasso model above for both the train and test data. Histograms of these residuals revealed approximately normal distributions in these residuals, as expected. There were a few large outliers within the residuals for the training data, which I then truncated at +/- $200,000.



**Part 6: Cleaning and EDA of changeable features to identify predictors for inclusion in the model of sale price residuals**

After sub setting the feature data on the remaining changeable characteristics and imputing missing values based on the data dictionary and insights gained by examining related fields, I used the functions developed for the analysis above to identify 66 predictors (after the dumification of categorical features) for inclusion in this next model. This time I simulated a pure hold-out test set by examining the relationships between the predictors and the target (i.e. sale price residuals) in the training data only.

The linear relationship between the 2 remaining numerical predictors (OverallQual and OverallCond) and the residuals was relatively week – refer to the correlations within the screen shot below. Interestingly, the correlation between the sale price and the residuals is higher (0.41), reinforcing the previous observation that the model doesn’t perform as well for higher value properties.

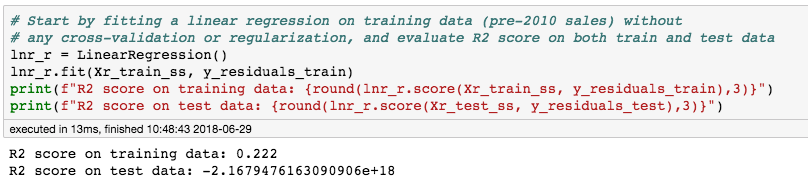


The combined boxplots and swarm plots of the sales price residuals for each categorical feature also showed much less variation across the different values for these predictors than previously observed when identifying predictors for the sale price.

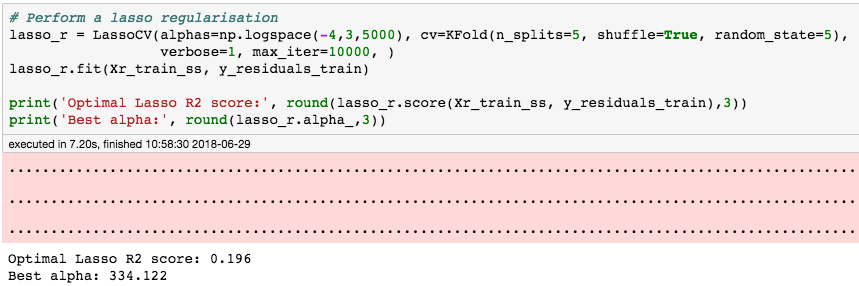
**Step 6: Train and evaluate a linear regression model to predict sales price residuals using changeable features**

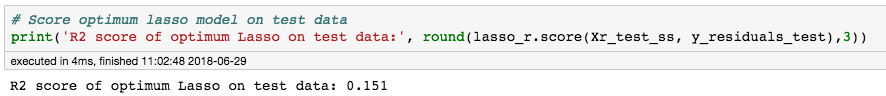
With much weaker relationships between the changeable features and the sales price residuals from the optimum Lasso model identified in Part 1, I wasn’t expecting good model results of the sales price residuals.

Fitting a linear regression model without cross validation or regularisation resulted in a relatively low R2 score on the training data and an extremely large, negative R2 score on the test data – probably indicating overfitting.

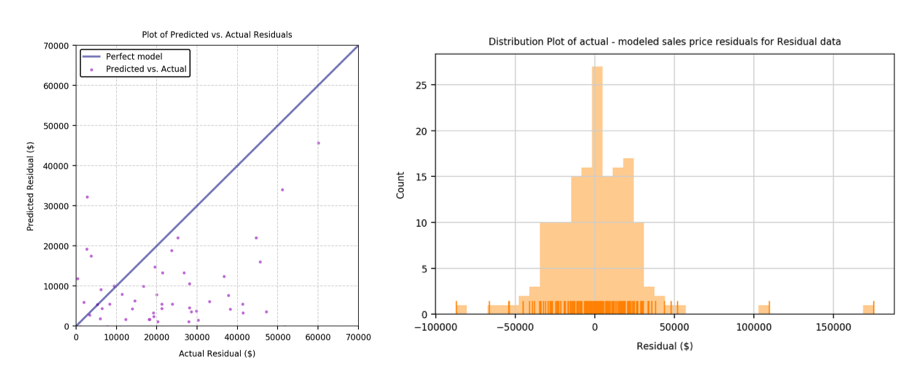


This time, grid searching on the LassoCV hyperparameters yielded a big impreovement in the model performance – but still relatively low R2 scores:

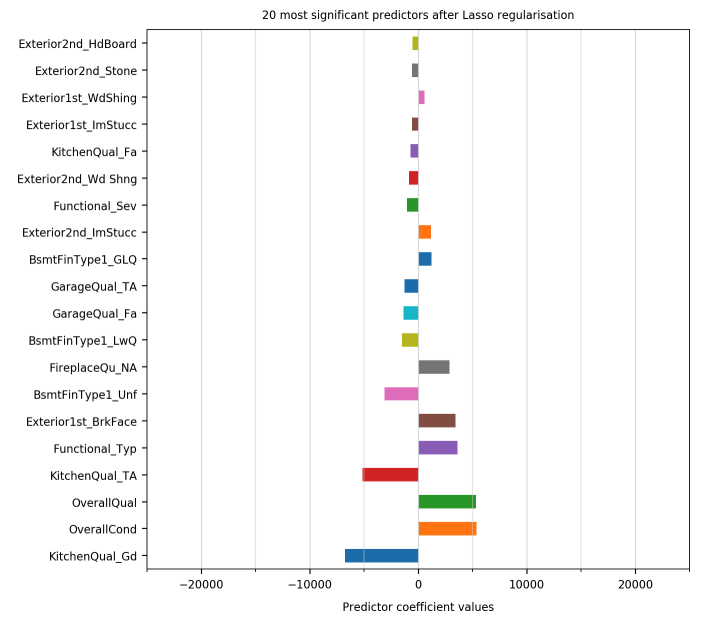




As can be seen above, the R2 score on the training data is slightly lower than the standard Linear Regression, but this is to be expected since the cross validation means that we're not fitting the data to all of the training data. The good news is that this resulted in a model that generalised better to the test data, and an R2 score of 0.151 – still not great, but better than the largen negative R2 score from the standard linear regression model. A plot of the residuals vs the actuals (residuals) – truncated at $70,000 - and a histogram of these differences confirms this ‘less than impressive’ model performance. This histogram below indicates that the R2 score would have been heavily influenced by a small number of large outliers.



The lasso prediction above zeroed out 45% of the model coefficients, reinforcing the view that the unregularized model was overfitted to the training data. The 20 largest predictor coefficients for the LassoCV model are displayed in the bar plot below.



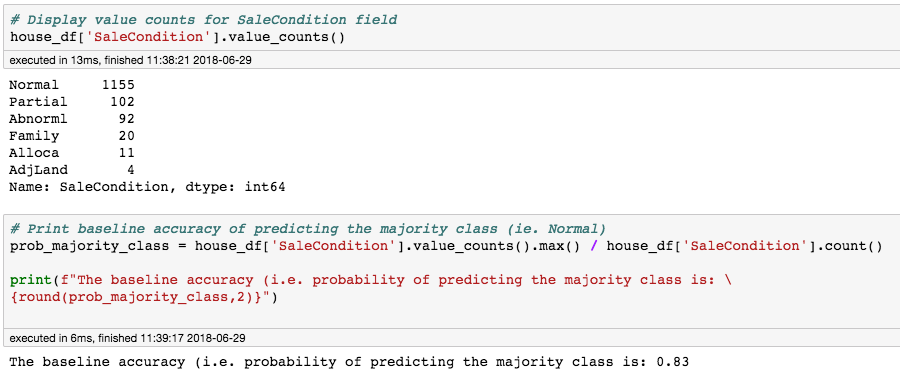
The chart above provides further evidence as to why the model's accuracy in predicting the residuals is relatively poor. 2 of the 3 largest predictors are OverallQual and OverallCond which both had a relatively low correlation with the target (residuals) itself!

**Conclusions**

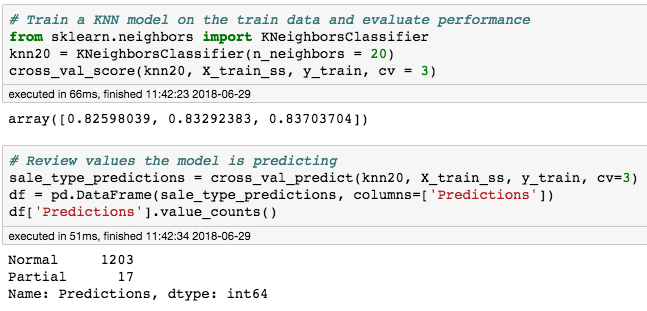
The sales price can be modeled with much greater accuracy using fixed features, than the residuals from this model can be predicted using changeable features!! I would therefore trust my first model as a good indicator of which properties to buy (based on their fixed features), however I wouldn't trust a model's ability to accurately identify which properties to renovate and how.

**Part 3: What Property characteristics predict and abnormal sale?**

As an additional (optional) challenge, I’ve been asked to predict which features can be used to predict the ‘abnormal’ sale of a property. An initial analysis of the ‘SaleCondition’ field reveals a strong class imbalance and a probability of predicting the majority class (i.e. a normal sale) of 0.83.



Using a very simple predictor matrix consisting of sale price and neighbourhood only, and training a KNN model with 20 nearest neighbours resulted in the prediction of the 2 most common sale conditions only – see below.



Training the model with a lower number of neighbours increased the variance of the model and hence resulted in a better distribution of predicted sales conditions across the different types, but the model wasn’t performing much better than the baseline. A combination of bootstrapping and down sampling would be required to achieve a more accurate model. This is something I’ll follow-up after the course.

©2018 Angus Franz

# **Classifying Data Science job adverts on indeed.co.uk as a low or high salary using NLP and Logistic Regression**

April 26, 2018

As a Data Science student at General Assembly London, I have had the chance to do several exciting projects to introduce me to to the world of Data Science. This describes one such project looking at Data Science job adverts on indeed.co.uk. I hope it is as exciting to read as it has been for me to work on.

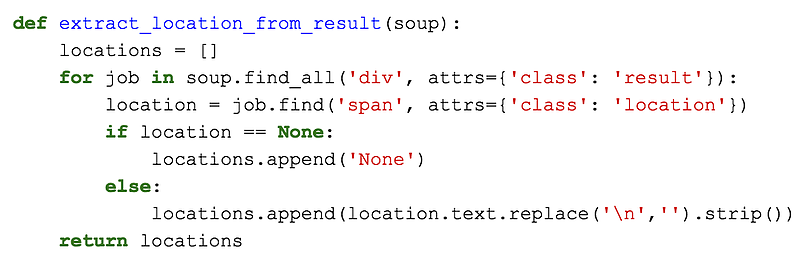
See the code on my GitHub: <https://github.com/angusfranz/classifying_indeed.co.uk_jobs> .

indeed.co.uk is one of the most popular job advertising websites, advertising thousands of jobs. In this project, I looked at the current jobs advertised on indeed.co.uk under the search ‘data science’ and modelled some aspects of the data to pull out what features of a Data Science job suggested a high salary.

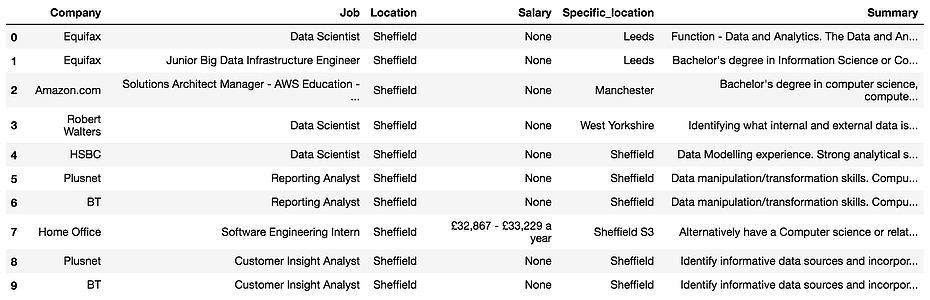
So many different jobs are advertised on indeed and I expected that many non Data Science jobs would also be picked up in the search. Therefore, my first expectation was that pure Data Science jobs are likely to have a higher average salary than those that are not pure Data Science Jobs. My second expectation was that jobs with greater seniority would have a higher salary; and my final expectation was that industry would be a significant factor in predicting if a salary would be high. For example, it would be expected that jobs in education would pay less.

Data was collected off indeed.co.uk by scraping using the package BeautifulSoup. The search term ‘data science’ was used for 14 different cities in the UK: London, Edinburgh, Newcastle, Cambridge, Manchester, Bristol, Liverpool, Leeds, York, Glasgow, Sheffield, Oxford, Cardiff and Belfast.

It was then parsed into a usable form with functions written to find the salient location, company, job title, salary, and summary information within the whole job advert. The example below shows the function written to extract the location data. This function iterated through all the job adverts using the soup.find\_all() function to find all the job adverts, marked in the website by the html tag <div> and class ‘result’. Within the job advert it would find, using the find() method, the location marked by the tag <span> and class ‘location’. If a location was present it would append it to the ‘locations’ list.



Using these five functions resulted in five lists which were put into a data frame for easy analysis. The top few rows are displayed below.



There were many duplicates and job adverts without salaries. All these job adverts were dropped. The salary was cleaned up into float format and if a range was given, the mean of that range was used. A total of 1015 jobs were in the final dataset. Initially a logistic regression model was applied, using just the location data as predictors. In order to put this into the model, the pandas.get\_dummies() method was applied to create dummy variables. The dummy variable for Belfast as the city with a mean salary closest to the median salary across all salaries was dropped. sklearn’s CountVectorizer function was then used to create a matrix of what words were used in each job summary. A logistic regression was run on this and was found to be a poor estimator of high or low salary. This process was then repeated on the job titles. Another logistic regression was then taken, now including industry and job location, improving the model further. Finally, a random forest classifier was used to see if this model would predict better.

Different common words were found in the job titles of adverts with a low and high salary. The word clouds show nicely which words are most frequent in the two classes. The most common words in job titles with high salaries were data, scientist, senior and engineer. Other important words were developer and manager. Many of the words are associated with more senior job roles and so may play a part in predicting low or high salary when modelled later. Similarly, common words seen in low salary job titles were associated with less senior job roles. Words such as graduate, associate, assistant, analyst and junior.

###### **Results and Conclusions**



**Logistic Regression - Location Only**

The logistic regression using the dummy variables of location was a weak to average classifier, giving an accuracy of 0.7 on the test data, not too much higher than the baseline of ~0.5. However, the table of coefficients for each city shows that London was the strongest predictor for a high salary, whilst York was the strongest predictor of a low salary. This suggests that the highest paid jobs are in London. The ROC curve with an AUC of 0.63 shows how the model is not a very strong estimator.

**Logistic Regression and Random Forest- All Variables**

The models that I ran on all the variables created a far better result, with an accuracy 0.82 for the logistic regression and 0.83 for the random forest. It was found that there were a few common strong predictors for both models which are the features that most effect whether a job on indeed.com has a high or low salary.

For the logistic regression, the strongest predictors were the words 'senior' and 'graduate' in the job title and the location of London. The bar chart shows how 'senior' is a strong positive coefficient whilst 'graduate' a strong negative coefficient. This means that job adverts with the word 'senior' in the job title are very likely to be a high salary, whilst those with the word 'graduate' are more likely to have a low salary. The other top coefficients can also be seen below. Interestingly, jobs in the education industry appear to be paid less.

Similarly with the random forest, the words 'senior', 'graduate' and the location of London have the highest feature importances. However it cannot be seen whether they mean lower or higher salary. It can be assumed that they are the same as the logistic regression.

So to conclude, look for words like senior, manager or director for jobs in London that are not in the education industry if you're looking for a well paid data science job!

I hope this has been interesting! I would really appreciate any comments or questions that you have. Have a look at my code at my GitHub: [https://github.com/angusfranz/classifying\_indeed.co.uk\_jobs .](https://github.com/angusfranz/classifying_indeed.co.uk_jobs)







Recent Posts

Classifying Data Science job adverts on indeed.co.uk as a low or high salary using NLP and Logistic Regression

April 26, 2018

1/1

Tags

[Classification](https://www.angusfranz.com/blog/tag/Classification)

[Logistic Regression](https://www.angusfranz.com/blog/tag/Logistic%20Regression)

[NLP](https://www.angusfranz.com/blog/tag/NLP)

[Random Forest](https://www.angusfranz.com/blog/tag/Random%20Forest)

[[GitHub-Mark-120px-plus](https://github.com/angusfranz)](https://github.com/angusfranz" \t "_blank)

[[https://static.wixstatic.com/media/6ea5b4a88f0b4f91945b40499aa0af00.png/v1/fill/w_20,h_20,al_c,usm_0.66_1.00_0.01/6ea5b4a88f0b4f91945b40499aa0af00.png](https://www.linkedin.com/in/angus-franz/)](https://www.linkedin.com/in/angus-franz/" \t "_blank)

## [Angus Franz](https://www.angusfranz.com/)

[Home](https://www.angusfranz.com/" \t "_self)

[Portfolio](https://www.angusfranz.com/portfolio" \t "_self)

[Blog](https://www.angusfranz.com/blog" \t "_self)

Data Scientist

# **Classifying Data Science job adverts on indeed.co.uk as a low or high salary using NLP and Logistic Regression**

April 26, 2018

As a Data Science student at General Assembly London, I have had the chance to do several exciting projects to introduce me to to the world of Data Science. This describes one such project looking at Data Science job adverts on indeed.co.uk. I hope it is as exciting to read as it has been for me to work on.

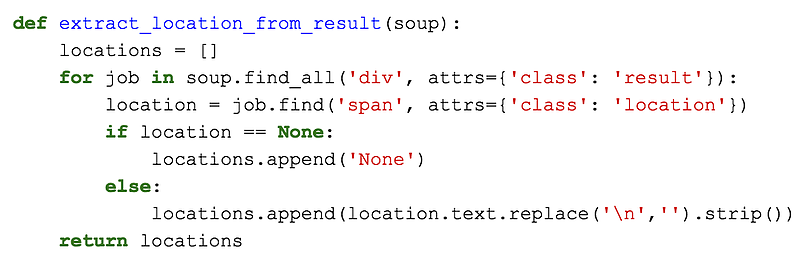
See the code on my GitHub: <https://github.com/angusfranz/classifying_indeed.co.uk_jobs> .

indeed.co.uk is one of the most popular job advertising websites, advertising thousands of jobs. In this project, I looked at the current jobs advertised on indeed.co.uk under the search ‘data science’ and modelled some aspects of the data to pull out what features of a Data Science job suggested a high salary.

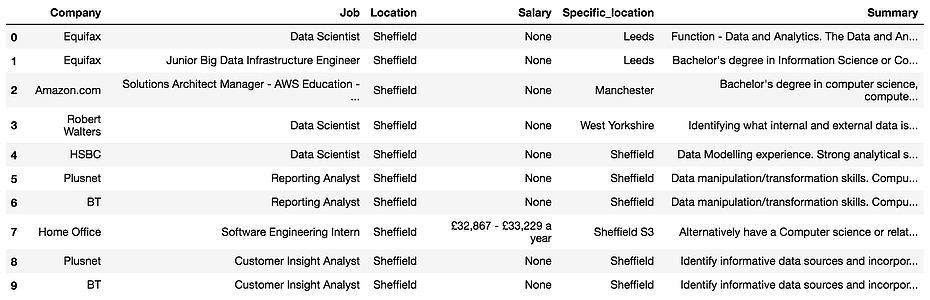
So many different jobs are advertised on indeed and I expected that many non Data Science jobs would also be picked up in the search. Therefore, my first expectation was that pure Data Science jobs are likely to have a higher average salary than those that are not pure Data Science Jobs. My second expectation was that jobs with greater seniority would have a higher salary; and my final expectation was that industry would be a significant factor in predicting if a salary would be high. For example, it would be expected that jobs in education would pay less.

Data was collected off indeed.co.uk by scraping using the package BeautifulSoup. The search term ‘data science’ was used for 14 different cities in the UK: London, Edinburgh, Newcastle, Cambridge, Manchester, Bristol, Liverpool, Leeds, York, Glasgow, Sheffield, Oxford, Cardiff and Belfast.

It was then parsed into a usable form with functions written to find the salient location, company, job title, salary, and summary information within the whole job advert. The example below shows the function written to extract the location data. This function iterated through all the job adverts using the soup.find\_all() function to find all the job adverts, marked in the website by the html tag <div> and class ‘result’. Within the job advert it would find, using the find() method, the location marked by the tag <span> and class ‘location’. If a location was present it would append it to the ‘locations’ list.



Using these five functions resulted in five lists which were put into a data frame for easy analysis. The top few rows are displayed below.



There were many duplicates and job adverts without salaries. All these job adverts were dropped. The salary was cleaned up into float format and if a range was given, the mean of that range was used. A total of 1015 jobs were in the final dataset. Initially a logistic regression model was applied, using just the location data as predictors. In order to put this into the model, the pandas.get\_dummies() method was applied to create dummy variables. The dummy variable for Belfast as the city with a mean salary closest to the median salary across all salaries was dropped. sklearn’s CountVectorizer function was then used to create a matrix of what words were used in each job summary. A logistic regression was run on this and was found to be a poor estimator of high or low salary. This process was then repeated on the job titles. Another logistic regression was then taken, now including industry and job location, improving the model further. Finally, a random forest classifier was used to see if this model would predict better.

Different common words were found in the job titles of adverts with a low and high salary. The word clouds show nicely which words are most frequent in the two classes. The most common words in job titles with high salaries were data, scientist, senior and engineer. Other important words were developer and manager. Many of the words are associated with more senior job roles and so may play a part in predicting low or high salary when modelled later. Similarly, common words seen in low salary job titles were associated with less senior job roles. Words such as graduate, associate, assistant, analyst and junior.

###### **Results and Conclusions**



**Logistic Regression - Location Only**

The logistic regression using the dummy variables of location was a weak to average classifier, giving an accuracy of 0.7 on the test data, not too much higher than the baseline of ~0.5. However, the table of coefficients for each city shows that London was the strongest predictor for a high salary, whilst York was the strongest predictor of a low salary. This suggests that the highest paid jobs are in London. The ROC curve with an AUC of 0.63 shows how the model is not a very strong estimator.

**Logistic Regression and Random Forest- All Variables**

The models that I ran on all the variables created a far better result, with an accuracy 0.82 for the logistic regression and 0.83 for the random forest. It was found that there were a few common strong predictors for both models which are the features that most effect whether a job on indeed.com has a high or low salary.

For the logistic regression, the strongest predictors were the words 'senior' and 'graduate' in the job title and the location of London. The bar chart shows how 'senior' is a strong positive coefficient whilst 'graduate' a strong negative coefficient. This means that job adverts with the word 'senior' in the job title are very likely to be a high salary, whilst those with the word 'graduate' are more likely to have a low salary. The other top coefficients can also be seen below. Interestingly, jobs in the education industry appear to be paid less.

Similarly with the random forest, the words 'senior', 'graduate' and the location of London have the highest feature importances. However it cannot be seen whether they mean lower or higher salary. It can be assumed that they are the same as the logistic regression.

So to conclude, look for words like senior, manager or director for jobs in London that are not in the education industry if you're looking for a well paid data science job!

I hope this has been interesting! I would really appreciate any comments or questions that you have. Have a look at my code at my GitHub: [https://github.com/angusfranz/classifying\_indeed.co.uk\_jobs .](https://github.com/angusfranz/classifying_indeed.co.uk_jobs)