Feature Engineering and Fraud Detection Within The Hut Group

Thomas Pinder, Nicholas Abad, Luke Lorenzi, Omar Khan, Mengnan Sun & Julie Sun

32136426, 32105207, 32204968, 32065328, 32139495, 32083446

SCC460

Lancaster University
January 21, 2018

Contributions

Within the project the jobs were carried out by the people below:

- Feature Engineering - Everyone
- Dimensionality Reduction - Omar
- Preprocessing - Nicholas
- Exploratory Analysis - Julie
- Logistic Regression - Luke
- Random Forest - Thomas
- Model Comparison - Luke & Thomas
- Presentation - Nicholas & Thomas
- Report - Omar, Nicholas, Luke & Thomas
- Merging individual files into Jupyter Notebook - Mengnan, Luke, Omar, Nicholas & Thomas

1 Introduction

It is estimated by Eaves (2017) that fraud costs the UK economy £193 billion per year which unravels to £6000 per second, of which 5% can be attributed to procurement fraud - fraud that happens directly between customer and merchant. Whilst it may be difficult to stop individuals carrying out fraudulent behaviour, it is possible for companies to enhance and improve their fraud detection methods to catch and prevent fraudulent behaviour within their domain.

The Hut Group, founded in 2014, are an e-commerce business consisting of over 100 websites with a particular focus on the health and beauty industry. With websites such as MyProtein and PreLoved forming the Hut Group's conglomerate, 2016 saw record sales of £501m for the year Group (2017). Naturally for a company of this size, reducing the amount of undetected fraud is a huge issue and until recently, a large majority of this was done manually.

In an attempt to assist The Hut Group with this difficult problem of fraud detection, our group received a dataset that consisted of all fraudulent and non-fraudulent transactions made by customers within a three month period from March 2016 to May 2016. With this dataset, our two project aims were to firstly gain valuable insight on the pre-existing variables in order to determine which will be useful in predicting fraudulent activity and secondly, produce a classifier that labels a transaction as fraudulent or not.

Within the literature, there is little up-to-date research surrounding fraud detection techniques as to publish such work would enable fraudulent individuals to gain an advantage when trying to *beat the system.*

In order to achieve these aims, our three objectives were to engineer a set of new variables through our feature engineering process, quantify the importance of each of the original and new features through the use of random forests, and lastly, construct and compare logistic regression and random forest models with an aim to maximise precision and the area under the curve (AUC) of the Receiving Operating Characteristic (ROC) curve of each model. The findings of this feature engineering and modelling resulted in 12 new features being engineered and random forests being deemed to be the optimal model, of those considered. A random forest, with synthetically re-sampled data, ran with an accuracy, precision and recall scores of 98.9%, 71.2% and 59.2%, respectively.

Unfortunately, from a modelling standpoint, fraud analysis is a typical case of a class imbalance (Section 2.2) problem whereby the result of interest, fraudulent transactions, is significantly outnumbered by the number of non-fraudulent transactions. This imbalance opens the door to a large number of modelling problems, with a common analogy being that the classifier is having to find the needle in the haystack.

2 Methodology

2.1 Preprocessing and Feature Engineering

The dataset consisted of three main tables and seven auxiliary lookup tables. Within the three main tables, the first detailed customers' account details, the second a full list of transactions between March and May, and the final table specifically detailed fraudulent transaction data. In each of these main tables, several new variables were created from the pre-existing variables in order to extract as much information as possible from what was originally given. Some of these new variables included the total number of orders placed for one customer, the proportion of cancelled items over the total number of ordered items, an international or domestic shipping boolean, a priority or standard delivery boolean, and a check as to whether billing and shipping postcodes matched. Specifically, in two variables that represented item categories in the dataset, one-hot encoding was also used in order to represent each possible item category as its own individual variable. By splitting a column within our dataset into multiple binary columns representing a specific item category, each item category has their own weights causing individual categories to be better represented.

In addition to engineering new features out of the dataset, missing data was still prevalent, with values being labelled as *NA*. Due to the aforementioned class imbalance, should an incomplete observation be labelled as non-fraudulent, then it was dropped from the dataset; fortunately this was the case for all 266 observations with missing data.

Conversely to this feature engineering, some features were dropped from the final model in order to order to obtain the most parsimonious model. This insignificance was deemed through feature importance in a random forest and stepwise selection with logistic regression. This is all discussed in greater detail in Sections 2.3 and 2.4.

2.2 Re-Sampling

Within the dataset, just 0.3% of observations were labelled as fraudulent, corresponding to 1212 fraudulent transactions of 418242 in total. This imbalance can be seen at a greater level of granularity in Table 2.1, however the take home message is clear; there are significantly fewer fraudulent transactions to work with when compared to non-fraudulent transactions.

An imbalance such as this becomes a nuisance when performing classification modelling as it makes the job of classifier somewhat analogous to picking a needle out of a haystack. There are too few fraudulent observations for the classifier to effectively *learn* what fraud looks like. To combat this, re-sampling can be carried out to synthetically engineer new observations. Numerous methods for re-sampling exist, with some of the simpler methods replicating observations in the minority class or removing observations in the majority class. Tomek links approach the problem from a slightly more intelligent angle, calculating the distance between any one minority class and its nearest majority class observation. The majority classed observation with the smallest distance is then removed, thus removing potential noise around the minority classes. Another re-sampling method, Synthetic Minority Over-Sampling Technique (SMOTE), involves the random selection of an observation of minority class and drawing a *line* through the feature space to a nearby minority class. Somewhere on this line, a new observation is created of minority class with the process being repeated until the desired proportion of minority-to-majority classes is achieved Weiss (2013).

Table 2.1: Class Imbalances By Site

Site	Proportions
121	0.25%
11	1.64%
15	0.28%
153	0.32%
120	0.06%

It was shown by Elhassan et al. (2016) that whilst Tomek links alone do a good job of increasing a model's accuracy and specificity, this comes at a huge cost to sensitivity. Tomek links with SMOTE then applied significantly increase sensitivity, along with all other performance metrics, giving a better performing model across the board and it is for this reason that this will be the imbalance metric used in the forthcoming analysis to achieve a 50-50 balance of classes.

2.3 Logistic Regression

Logistic regression was a particularly appropriate method to model fraud because the outcome was binary; a transaction was either fraudulent or it wasn't. As such, the *logit* link function was suitable Walsh (1987). Furthermore, logistic regression was common knowledge among all group members and easy to interpret which enabled more productive discussions when trying to solve problems related to model building and analysis.

Once the data had been stratified by site this left 6 smaller, site specific datasets, each with a site-specific set of one-hot encoded product category features, since different sites generally sold different products. Reducing the number of one-hot encoded variables within each site, based upon what that site sold made modelling considerably more efficient. In addition to this, some sites were much larger than others, the smallest having 8746 transactions (Site 11) and the largest having 285973 transactions (Site 121). Since the process of building a GLM was a similar process for all the sites, we will focus on site 11 for demonstrative purposes, however the methodology extends to all other sites for which a GLM has been constructed. Site 119 had no fraud and Site 120 had such a large class imbalance, as seen in Table 2.1, meaning that models couldn't be build for either site.

While one-hot encoding was a useful tool to use as it allowed the individual factor levels to be better represented, it did present problems when trying to build the GLM. As the n levels of the original variable were split into n columns there was a danger of creating multicollinearity within the one-hot encoded columns (https://stats.stackexchange.com/users/74500/matthew drury); a 1 in one column would almost definitely imply a zero in the other $n - 1$ columns. For this reason it was decided to remove them from the GLM.

Through the use of the Akaike Information Criterion (AIC), insignificant variables were identified by first fitting the full additive model and then removing them in a backwards elimination style until all terms were significant. AIC was a particularly useful statistic in a model with such a large amount of features as it penalises models with an excessive amount of features used and by removing several insignificant features it meant that building the model was computationally inexpensive, allowing for many combinations of variables to be tested for the best possible final result. Cross-validation was used to assess the model, as covered in Section 2.5.

2.4 Random Forests

Within the world of machine learning algorithms, random forests can be used for both classification and regression, with only the former being necessary for this instance. The crux of the random forest lies within a decision tree; a set of decision nodes, branches, leaves and a single root node that aim to best partition the data. The exact feature used to partition the data at a decision node is found via the Gini index function, Equation 2.1, whereby p is the probability of an item with label/midpoint i being selected for all d classes. With all Gini indexes calculated, the feature that results in the smallest Gini value is selected.

$$Gini(M) = 1 - \sum_{i=1}^{d} p_i^2 \tag{2.1}$$

The leaves of a decision tree are the final node in the tree's structure and are usually labelled with the class predicted by following that specific path. Conversely, a root node is the original starting point of the tree whilst branches simply link up all the nodes within the tree.

With the notion of a decision tree defined, this concept can be extended to a random forest by utilising bootstrap aggregation, bagging. Bagging is one of the features of a random forest that reduced the possibility of overfitting as, unlike in a decision tree, not all of the original data is used to decide upon the tree's structure; instead, a bootstrapped sample is made of the initial data and then used to build a tree. The second differing feature between a decision tree and a random forest is that in a random forest, not every feature is considered when calculating the splitting feature at each node. Instead in a classification setting, $\sqrt{d}$ features are considered, resulting in a reduced amount of bias being present in the final model. This process is repeated many times with an average being taken of all the resulting trees. This bagging process reduces the variation component of error in the final model, meaning that whilst a decision tree may obtain a higher accuracy on training data than a random forest, a random forest will almost always perform better on unseen test data, something that is very desirable in the case of fraud detection.

Random forests were utilised in two ways, firstly to obtain variable importances and secondly to attempt to model fraud in both the original data and a synthetically re-sampled dataset. Variable importance was employed early on in the analysis pipeline as a way of gauging which variables played a significant role in predicting fraud and allowed for feature engineering to be done in a more intelligent manner by paying particular attention to those variables deemed significant by the random forest. The modelling implementation of a random forest came after all pre-processing and feature engineering had been carried out and the first step was to determine the number of trees needed. This value was determined by fitting multiple random forests to the data and observing where the plateau in accuracy increases occurred.

With the number of trees calculated, the model could be formally fitted, using only variables deemed important (importance value > 0) modelled. Once a model had been built, evaluation techniques discussed further in Section 2.5 were carried out on both the original dataset and the re-sampled dataset to obtain the optimum model.

2.5 Model Assessment

When assessing a model, k-fold cross-validation was used to determine model metrics with k selected as 10 due to its ability to best balance variance and bias Steyerberg et al. (2001). Accuracy is the first metric used due to its ability to give a high-level overview of a model's performance by assessing the proportion of cases that a model predicted correctly, compared to the number of observations. Within fraud detection, a false negative is hugely problematic, however, a false positive is almost equally so due to the inconvenience caused to a customer having their order paused whilst its fraud is checked. For this reason, precision, recall and f-score have been used as they allow for false positives and false negatives to be quantitatively measured in relation to the amount of true positives and negatives. Finally, AUC is used as it measures the probability that a randomly drawn observation is deemed to be classed as fraudulent instead of negative. This becomes useful when assessing a threshold probability to use when classifying observations as fraudulent or not.

3 Results

Using techniques described in Section 2.2, SMOTE with Tomek links were applied to the training data, resulting in Figure 3.1. This shifted the proportion of fraudulent transactions in the training data from 0.3% to 50%, enabling the classifier to more effectively identify fraudulent transactions and learn what comprises of a *typical* fraudulent transaction.

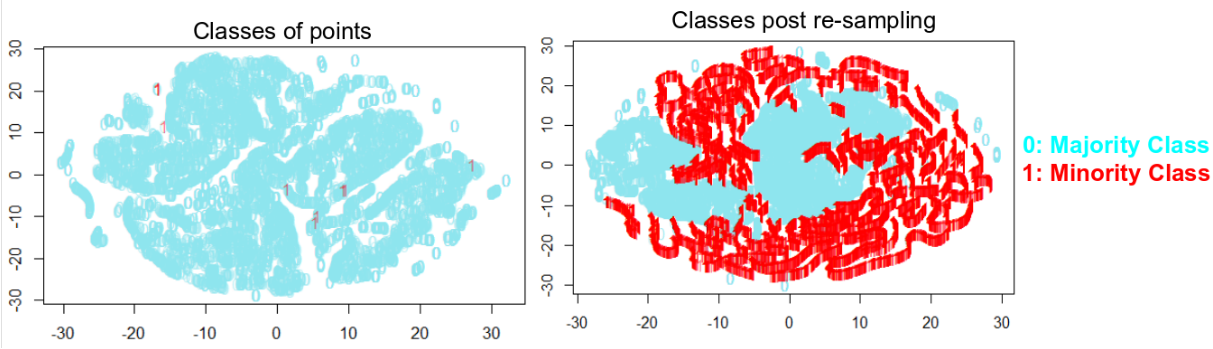

Figure 3.1: Clusters of points pre and post-sampling. t-SNE used to aid visualisation.

Re-sampling, in the form of SMOTE and Tomek links, was only carried out on the training split of the data as re-sampling prior to splitting. Had the entire dataset been re-sampled then split out, there would have been a very high probability that identical artificial observations would be present in both the training and testing dataset. The follow through effect of this is that the classifier would be *cheated* during the testing fit as it would have already seen a number of observation whilst being trained, thus warping the overall model's performance.

With features engineered and data re-sampled, the model training phase could commence. Within the logistic regression setting this stage consisted of attempting to find the most parsimonious model through backwards elimination. This resulted in the following variables being modelled: customer reliability, whether the billing and shipping address match, site, product price, whether the occupation of the customer is known, whether the items purchased were shipped domestically or internationally, the payment provider key, and the locale key. From here the logistic model was fitted to both the original training data and the re-sampled training data and respective testing sets were then passed through to generate performance metrics, as shown in Table 3.1.

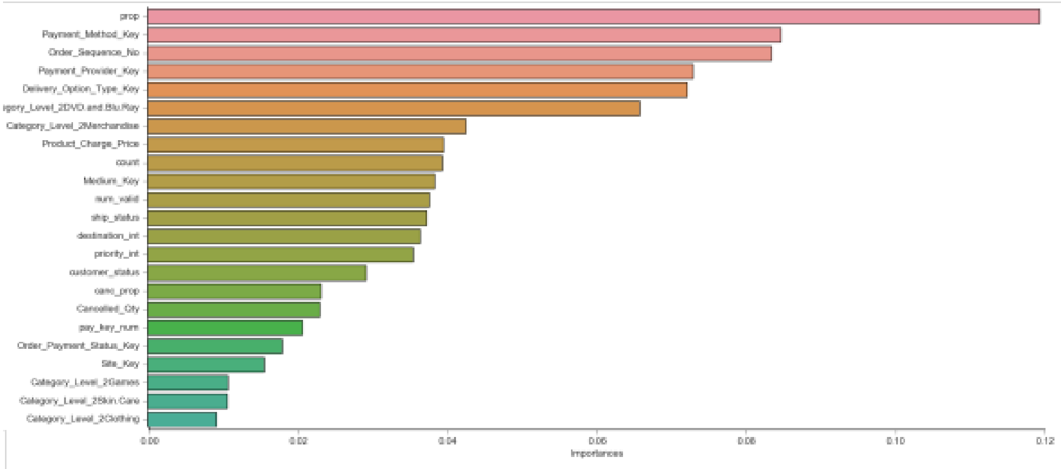

Figure 3.2: Variable Importances for the entire dataset with re-sampling applied. Only variables with an importance > 0 are shown here due to size restrictions.

For the random forest the number of trees used was the first concern. To assess this, numerous random forest models were fitted with the number of trees ranging from 5 to 500. It was found that the random forest quickly reached a 99.8% accuracy within 50 trees, however the result of fitting an additional 30 reduced standard error, resulting in 80 trees being the decided number. The training data was then passed through an 80-tree deep random forest to assess variable importance. As can be seen in Figure 3.2, a large number of variables had little to no importance within the model and they were therefore not considered in further model fitting. Any variable with an importance greater than 0 was deemed to be important and was consequently modelled. The results of this 80-tree deep random forest with a subset of important variables can again be seen in Table 3.1.

Table 3.1: Comparison of Model Results For Entire Dataset

Model	Accuracy	Recall	Precision	F-Score	AUC
Logistic Regression	99.7	0	0	0	78.9
Random Forest	**99.8**	39.0	**70.4**	49.5	**90.0**
Logistic Regression (with re-sampling)	92.6	**57.4**	2.4	4.6	88.4
Random Forest (with re-sampling)	**99.8**	38.9	70.1	**50.0**	**92.0**

As can be seen in Table 3.1, the random forest without re-sampling provides the best all around model at a global level, from the four models tested. Unfortunately, this notion does not extend at a local level when attempting to model fraud at a site specific level with precision, recall and f-score all being significantly reduced. This works antagonistically when addressing our initial aim of trying to reduce the possibility of fraudulent transactions being missed. When comparing metrics between the individual sites using the random forest with and without re-sampling, those aforementioned metrics are significantly improved by up to 9% for precision. For this reason, the random forest with re-sampled minority class points is used to produce models at the site level, the results of which can be seen in Table 3.2.

Table 3.2: Comparison of Model Results

Model	Accuracy	Recall	Precision	F-Score	AUC
Site 121	99.8	33.8	65.8	44.2	89.4
Site 11	98.9	54.1	71.0	60.1	**97.8**
Site 15	98.8	42.1	88.6	55.6	88.5
Site 153	**99.9**	**56.3**	**93.0**	**68.5**	95.9

Table 3.2 shows that a random forest performs best on sites 153 and 11 with strong scores across the board. Interestingly, it can be seen from Table 2.1 that there is no correlation between the proportion of fraud present in the site and the model's ability to predict fraud. Moreover, Site 11 has the highest amount of fraud at 1.64%, whilst Site 153 has just 0.32%. One possible explanation to this is that, of the modelled sites, Sites 11 and 153 have some of the lowest total transaction counts at just 8536 and 30102. Consequently it is plausible that whilst re-sampling enables the classifier to better *learn* what a fraudulent transaction looks like, there is a turning point in the amount of re-sampling that is needed and that in larger sites, such as 121 with 271197 observations, re-sampling does not work as well, as shown by Site 121's poorer metrics in Table 3.2. One of the fundamental aims of this project was to identify a set of global variables that were strongly indicative of fraud across all sites. From inspecting the variable importances, the following variables were important globally and in at least two individual sites: Proportion of a customer's previously cancelled orders*, the order sequence number, payment provider, delivery option, the product's charge price, the customer's trustworthiness status* and if the product was a DVD*. Additionally, the following the variables were not common globally, however they were ranked in the top 10 most important variables in at least one site: if the product was categorised as health and beauty*, games* or merchandise*, if delivery was dispatched with priority* and if the delivery was internationally shipped*.

4 Addressing Validity and Potential Biases

Throughout the project, class imbalances and measurement error were considered to be the two sources of bias that had the potential to significantly affect our conclusions and results if not dealt with properly. Due to only 0.3% of our observations being classified as fraudulent, failing to address this heavy class imbalance problem would lead to a model with a high predictive accuracy rate, but would produce a classifier that would categorize all or nearly all observations as being non-fraudulent, which will not help in fraud detection in the future. In order to alleviate this concern, SMOTE was used to oversample the fraudulent observations and Tomek links were also used to under-sample the non-fraudulent observations. Details regarding these methods have been discussed previously in Section 2.2. Additionally, there is a possibility that measurement errors occurred in the original dataset, which would be categorized as when a fraudulent transaction would "slip through the cracks" and be labelled as non-

* dictates a custom engineered feature

fraudulent. Although this potential bias has been brought to our attention, there was unfortunately nothing that could have been done to verify that all observations were labelled correctly at this time.

Furthermore, due to only being given three months worth of data spanning from March to May of 2016, we encountered an external validity concern in which we were not be able to further generalise our results and findings. With only three months worth of data, special events such as Christmas, Black Friday, Cyber Monday, etc. were unaccounted for and it could be assumed that fraud is heavily prevalent during these days. With a dataset spanning a short period of time, seasonality trends were also not addressed, which might have brought forth issues that we were not able to address in the analysis of our dataset.

5 Conclusions and Further Work

One of the initial aims was to engineer a set of variables that can be used to predict fraud at both a global level and at an individual site level. We have shown through a random forest that variables such as the proportion of orders cancelled previously by a customer, along with the charge price, payment and delivery method are all strong indicators of fraud at a global and site level. Additionally, the sale of products categorised as a DVD, game, health & beauty and merchandise should be treated with caution.

By using a combination of both the original and new features, our second aim was then to produce a classifier that labels an observation as being fraudulent, which was accomplished through the use of a random forest and logistic regression. At a global level, it was shown that the random forest without re-sampling provides the best all around model for our dataset but this result was not consistent when stratifying our dataset into individual sites. When stratified, the random forest with re-sampling improved metrics significantly and with an aim to maximise precision, individual sites improved in this metric by up to 9%.

The project presented in this report has shown in detail how well various implementations of logistic regression and random forest models can be used as classifiers for fraudulent activity, unfortunately the time constraint of the project was a limiting factor. Further work into this project could see additional classifiers such as Naive Bayes, Support vector machines and Neural networks to be constructed to allow for further comparison of classifiers. An extension of this and an idea that was tested initially would be to implement an ensemble classifier that used multiple classifiers to potentially enhance the detection of fraud within the dataset.

Had this project spanned the full 10 weeks of this term, the aim would have been to implement a random forest from scratch in Python, using custom cost functions to assess the classifier's performance. One example of a custom cost function, is the case that false negatives should be penalised more severely dependent upon the erroneous classification's corresponding charge price. The intuition behind this is that the classifier should be more conservative when predicting fraud in higher priced items due to more grievous implications that come with a incorrect classification compared to a mistake in a lower priced item.

Furthermore, being able to train and test the models on a larger dataset would be be beneficial to see how well they classify observations at different periods of the year. Access to a full year of observations would allow the classifiers to take into account busy periods such as Black Friday and Christmas and see if there are general trends across time.

Bibliography

Eaves, D. (2017), 'Fraud costs uk economy £193 billion a year - equating to more than £6,000 lost per second every day - latest thinking blog'.
URL: *https://goo.gl/y1Aq52*

Elhassan, T., Aljurf, M., Al-Mohanna, F. & Shoukri, M. (2016), 'Classification of imbalance data using tomek link (t-link) combined with random under-sampling (rus) as a data reduction method', *Journal of Informatics and Data Mining* .

Group, T. H. (2017), *THG Annual Report*.
URL: *https://goo.gl/L4dyCp*

(https://stats.stackexchange.com/users/74500/matthew drury), M. D. (n.d.), 'One-hot vs dummy encoding in scikit-learn', Cross Validated. URL:https://stats.stackexchange.com/q/224055 (version: 2016-07-19).
URL: *https://stats.stackexchange.com/q/224055*

Steyerberg, E. W., Harrell, F. E., Borsboom, G. J., Eijkemans, M., Vergouwe, Y. & Habbema, J. D. F. (2001), 'Internal validation of predictive models: efficiency of some procedures for logistic regression analysis', *Journal of clinical epidemiology* **54**(8), 774–781.

Walsh, A. (1987), 'Teaching understanding and interpretation of logit regression', *Teaching Sociology* pp. 178–183.

Weiss, G. M. (2013), 'Foundations of imbalanced learning', *Imbalanced Learning: Foundations, Algorithms, and Applications* p. 73.