Fraud in Electricity and Gas Consumption:

The Logistic Regression, KNN and XGB Approach

Introduction

Electricity fraud reduces energy companies' profits and disrupts power grids. However, fraud detection in electricity and gas consumption is not an easy task. Therefore, it would be helpful to leverage AI/ML techniques to help identify potential fraudulent clients/users. Some of the techniques that have been used are eXtreme Gradient Boosting (XGB), Random Forest (RF) or Light Gradient Boosting (LGB) (Oprea and Bâra 2022).

Dataset

We match a client in "client.csv" to his corresponding invoice using the "id". We have 135493 clients but only 31603 (~23%) of them have corresponding invoices. To predict fraudulent manipulations of meters, we have to analyse the customer's consumption behaviour. Therefore, we will exclude clients without any invoice data from our model, and use 31603 clients to train our model

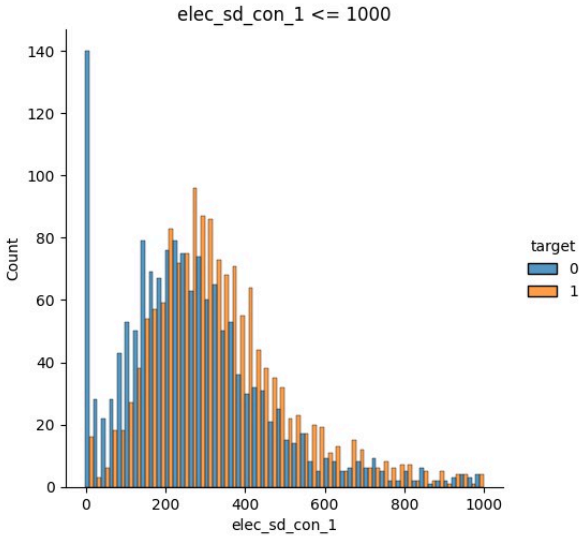
Out of 31603 clients, only 6% are fraud. Therefore, the dataset is highly imbalanced. This means that we cannot simply look at accuracy as the only performance matrix for our model. For example, a model that always predicts non-fraud would have an accuracy of 94%.

To capture the behaviour of the client's consumption, we calculated summary statistics for each client's invoices. We also separated the client's consumption of electricity and gas, by creating new features. After grouping the invoices by client's id, we also grouped the invoices by `counter_type` (ELEC, GAZ), as electricity and gas bills should not be compared directly, due to factors such as different scales of magnitude. For categorical data such as `tarif_type` and `counter_statue` we calculated their mode. For numerical data such as `reading_remarque`, and `consommation_level_1`, we calculated their mean and standard deviation. The approach was inspired by this paper (Coma-Puig et el. 2016)

The visualization below suggests that the summary statistics do separate fraud from not fraud.

We sampled 3000 data points, of which 1500 are fraud clients and 1500 are not fraud clients. We see that the standard deviation of Electricity Consumption Level 1 differentiates the non-fraud clients (blue) from the fraud clients (red).

- Notice that there is a concentration of non-fraud clients around 0.
- Notice that there is a concentration of fraud clients around 300.



Methods

In order to keep our study focused, we needed a performance measure that we could refer to. We crafted a set of performance metrics obtained from the various models we tested. These consisted of:

- ROC Curve and Area Under the Curve (AUC)
- Confusion Matrix
- Accuracy

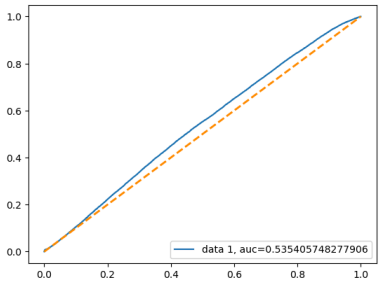
The methods that we tried are namely:

- Logistic Regression
- Logistic Regression with resampling
- Logistic Regression with feature scaling
- Logistic Regression with resampling and feature scaling
- K-Nearest Neighbors (KNN)
- Weighted KNN
- XGB

Logistic Regression:

To get a baseline result to refer to, we simply joined the client and invoice datasets into one dataframe. Running logistic regression on this dataframe achieved the following results

- Accuracy: 0.92
- TP, TN, FP, FN[1]: 0, 193740, 0, 15975
- ROC curve:

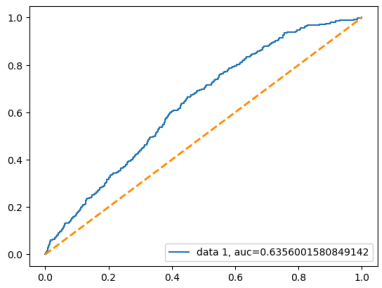


From the results, the accuracy of the model (0.92) seems high, however, by looking at the confusion matrix, we realise that the model simply predicts non-fraud (false) for all invoices. Therefore, the model is not useful, although accuracy seems high.

Another metric we used is the area under the ROC curve. Here we see that the area under the ROC curve is 0.535. This is close to 0.5, which indicates that the model makes seemingly random predictions. Therefore, we need to perform some feature selection, feature scaling or resampling, to improve the performance of our model (Google 2022).

After feature selection, we consider summary statistics of each client, rather than individual invoices. By running logistic regression on these new features, we achieved the following results

- Accuracy: 0.94
- TP, TN, FP, FN: 1, 5946, 6, 368
- ROC curve:



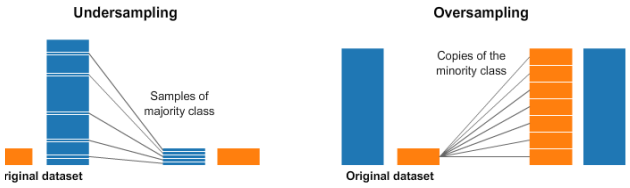
This model which uses summary statistics of a client as features, shows a better performance than the previous model, which simply considered all invoices.

We noticed that the model started to predict fraudulent outcomes. The precision in predicting fraudulent cases increased from 0 to 0.14. Additionally, the area under the ROC curve increased from 0.535 to 0.698, meaning that the model now has a 69.8% chance of distinguishing between a fraud and a non-fraud client.

However, the data was still highly imbalanced since there was a disproportionately low number of fraudulent clients in the dataset. Therefore, we decided to perform resampling methods, so that there will be an equal proportion of fraud and non-fraud clients for training of the model. (Dang et el. 2021)

Logistic Regression with resampling:

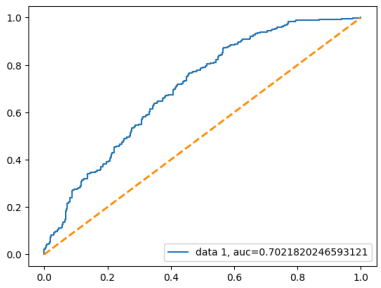
We used two main sampling methods, undersampling and oversampling. Undersampling balances the dataset by taking samples of the majority class such that the number of samples of the majority class now equals to the minority class (Nour 2021).



When running logistic regression on the randomly undersampled dataset, we notice the following results

[1] True Positive (TP), True Negative (TN), False Positive (FP), False Negative(FN)

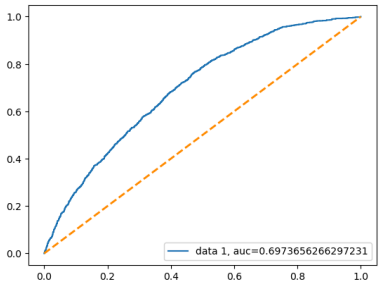
- Accuracy: 0.65
- TP, TN, FP, FN: 224, 236, 132, 111
- ROC curve:



These results showed improvements, most notably in terms of the confusion matrix, yielding much more balanced results. Although there was a drop in accuracy, we can be confident this is a more reliable result as it carries less bias.

However, with undersampling comes the risk of accidentally removing valuable information when sampling from the majority class. Hence, we decided to try oversampling the data as well. Oversampling follows a similar concept of balancing the data, except now we duplicate records in the minority class until the minority class matches that of the majority class. Running logistic regression on the oversampled dataset, we achieved the following results

- Accuracy: 0.65
- TP, TN, FP, FN: 3932, 3871, 2105, 2031
- ROC curve:



While the results showed little change in accuracy, the size of the sample produces a slightly higher AUC. Hence, we favor oversampling going forward.

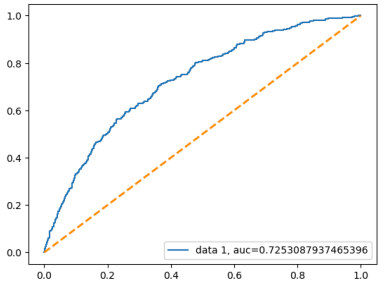
Logistic Regression with feature scaling and one-hot encoding:

While looking through the features in the dataset, we noticed that the magnitude of these features differs greatly. Therefore, the logistic regression model may be biased to certain features. (Zeng 2023)

We perform standardization on the numerical features to scale them and improve the convergence of our logistic regression model. We perform one-hot encoding on categorical features, such as region, district, and category, to ensure that these categorical features can be handled by the logistic regression model (Filho 2023).

By running the Logistic Regression model with feature scaling and one-hot encoding, we get the following results:

- Accuracy: 0.94
- TP, TN, FP, FN: 0, 5951, 1, 369
- ROC curve:



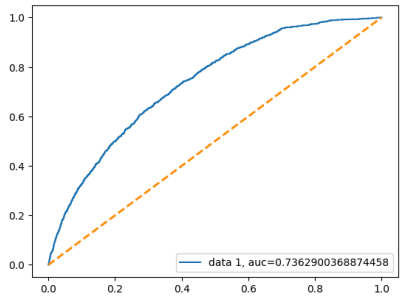
Here we see that with feature scaling and one-hot encoding of categorical variables, the performance of the logistic regression model improved slightly. The area under the ROC curve increases from 0.698 (without resampling) to 0.725.

However, due to the highly imbalanced data, these results may not be representative. Therefore, we implemented a logistic regression model that uses resampling, feature scaling, and one-hot encoding.

Logistic Regression with resampling, feature scaling, and one-hot encoding:

As before, we solve the problem of imbalanced data with resampling. As determined, oversampling is the more effective method of sampling, so we implemented it immediately, yielding our best result

- Accuracy: 0.67
- TP, TN, FP, FN: 4111, 3914, 2062, 1852
- ROC curve:

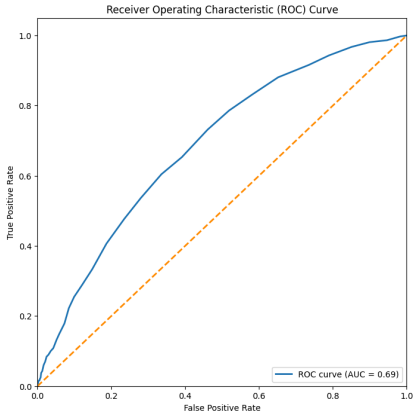


K-Nearest Neighbors (KNN):

Theoretically, KNN is versatile and effective in handling classification tasks. In addition, KNN is a simple algorithm to understand, is easy to implement, and does not require training.

Our KNN model achieved the following results:

- TP, TN, FP, FN = 0, 5952, 0, 369
- Accuracy = 0.94



Due to the highly imbalanced dataset, KNN was unable to make reliable predictions as seen in the ROC curve (in the notebook), where AUC = 0.69.

Weighted KNN:

Due to the highly imbalanced dataset, we tried weighted KNN, which gives higher weights to the points that are closer to the points representing the fraud cases instead of treating all neighbours equally. However, we did not see any improvements.

eXtreme Gradient Boosting (XGB):

XGBoost, which stands for eXtreme Gradient Boosting. It is a predictive model created by using several other models, which is a kind of ensemble learning family.

For our model, we split the entire training dataset into 17 sets, with each set containing the entire fraud training data and 1/17 of the non-fraud training data, so that each set as equal number of fraud and not fraud data. Then we randomize the order of data in each data set to further improve our training. The results are:

- TP, TN, FP, FN = 243, 3781, 2189, 108
- Accuracy = 0.64

Final Results and Discussion

In conclusion, we found the logistic regression model with feature selection, resampling using oversampling and one-hot encoding to be the most effective model. We evaluated the performance of our models using Accuracy, the Confusion Matrix, and the area under the ROC curve. We achieved an accuracy of 0.67, a Precision of 0.67 for predicting fraud, and an area under the ROC curve of 0.741, which shows that our model has a 74.1% chance to distinguish between fraud and non-fraud clients.

We explored using other models like KNN, but it performed poorly since the dataset is very large and highly imbalanced. This makes it hard to pick a suitable value of k. Furthermore, running KNN for such a large dataset (over 30000 clients) is computationally expensive and very slow.

Lastly, for XGB, we can see that the model predicts ‘Fraud’ very often, that is because we have supplied a lot of fraud training data. Unfortunately, most of the predictions are incorrect. We have tried to overcome this by reducing the proportion of fraud training data in each set, but then the model predicts almost all of the clients to be not fraud during testing which is not ideal. We decided in the end for a model that detects fraud, it would be more important for it to have a higher true positive rate, even though the accuracy might drop. Therefore we did not reduce the proportion of not fraud training data.

References

Oprea, SV. ,Bâra, A 2022. Feature engineering solution with structured query language analytic functions in detecting electricity frauds using machine learning. Sci Rep 12, 3257. https://doi.org/10.1038/s41598-022-07337-7

B. Coma-Puig, J. Carmona, R. Gavaldà, S. Alcoverro and V. Martin 2016. Fraud Detection in Energy Consumption: A Supervised Approach, 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Montreal, QC, Canada,, pp. 120-129, doi: 10.1109/DSAA.2016.19.

Google 2022. Classification: ROC Curve and AUC https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

Dang, Tran Khanh, Thanh Cong Tran, Luc Minh Tuan, and Mai Viet Tiep. 2021. Machine Learning Based on Resampling Approaches and Deep Reinforcement Learning for Credit Card Fraud Detection Systems. Applied Sciences 11, no. 21: 10004. https://doi.org/10.3390/app112110004

Nour Al-Rahman Al-Serw 2021. Undersampling and oversampling: An old and a new approach. Medium. Undersampling and oversampling: An old and a new approach | by Nour Al-Rahman Al-Serw | Analytics Vidhya | Medium

Guoping Zeng 2023. On the analytical properties of category encodings in logistic regression, Communications in Statistics - Theory and Methods, 52:6,1870-1887,
DOI: 10.1080/03610926.2021.1939382

Mario Filho 2023. Does Logistic Regression Require Feature Scaling? Forecastegy Does Logistic Regression Require Feature Scaling? | Forecastegy