IT1244 Project Report: Fraud in Electricity and Gas Consumption Dataset

Team 17
Ang Wei Cheng
Chan Pei Yu Sarah
Liew Zhi Song, Elias
Nyan Lin

Introduction

Electricity theft leads to substantial financial losses for utility companies, amounting to billions of dollars worldwide. This undermines their ability to invest in infrastructure upgrades and improve services for customers. Additionally, unauthorised connections and tampered metres can strain local power grids, creating a risk of outages and instability that affects all consumers, not just those engaged in fraudulent activities. (Din, J, 2024)

Therefore, we aim to tackle this problem by optimising the process of detecting fraud cases through testing a range of Machine Learning techniques such as **KNN, Logistic Regression, Random Forest and Feedforward Neural Network**.

Through research, we found out that there were 2 limitations current works face:

- In our case, the primary concern is the high cost of misclassifying fraudulent cases as non-fraudulent, as failing to detect fraud could lead to significant financial losses relying on the F1 Score, which balances precision and recall equally. However, in scenarios like ours and potentially in theirs, using F1 Score can be suboptimal. Models that generate False Negatives – such as undetected fraud – pose a higher risk than False Positives. While F1 Score offers a balanced view, the trade-off can result in under-detecting critical issues like fraud or faults. (MB Soultanzadeh et al., 2024)
- Imbalance in the datasets was a central challenge; a few cases of fraud were available, and that seemed to be hard for the models to train and detect fraudulent patterns. This issue finds further emphasis in the work of ZS Rubaidi et al.(Zainab S, R. n.d.), where, in fraud detection problems, a few fraudulent cases are causing high bias toward the majority class with high false negatives and misleading accuracy metrics. This imbalance influences the model's generalisation, especially for minority cases, which influences its effectiveness when new data are sufficiently different from the training dataset. To handle this, SMOTE was used to create synthetic samples of the minority class. Then again, this technique has its drawbacks: SMOTE can easily result in overfitting by merely replicating noise in the data, boundary overlap by placing synthetic samples near the class borders, and high-dimensional data may reduce the model's performance.

Our project strives to tackle these limitations and enhance overall model performance by:

- Using more suitable performance metrics that emphasise more on detecting actual fraud cases. We will adopt the **F2 Score** over the F1 Score, as it places greater emphasis on **Recall**, helping us minimise **False Negatives** by ensuring that more fraudulent cases are correctly identified. A shift toward a recall-focused metric like **F2** would have better aligned their evaluation to the operational reality of reducing the risk of missed detections, even if it increases the number of false positives.
- Instead of just using **SMOTE**, **TOMEK** or **ENN** alone in data balancing, we applied a hybrid data balancing technique that combines **SMOTE with Tomek links** and **SMOTE with ENN**.

Dataset

Handling Variable-Length Transaction Data through Aggregation and Analysis

Limitation 1: Variable Number of Invoices per client. Clients in the dataset have 3 to 50 invoices, introducing significant variability in transaction data length. This variable-length data makes it challenging to use in machine learning models that typically require fixed-length inputs, requiring aggregation or summarization to ensure consistency.

Solution 1: We addressed this by merging datasets based on the client's ID. Then, we aggregated the transaction data to represent each client as a single entry, capturing essential patterns across all invoices, which will be further elaborated below.

After merging the datasets, we performed exploratory data analysis to assess the relationship between feature variables and the target variable (fraud). This revealed key limitations in the dataset:

Features engineering, aggregation and transformation

Limitation 2a: Low correlation and significance with the target variable (fraud) across all feature variables. We assessed the relationships between features and the target variable (fraud) using several statistical methods. For numerical and ordinal features, we applied the Point-biserial correlation along with p-values to evaluate their correlations and significance with the fraud target. For categorical features, we used the Chi-square test to examine their association with the target. Additionally, we utilised a decision tree to capture non-linear relationships with the target variable, allowing us to determine feature importance effectively. *(Refer to Correlation of X and Y variables code folder)* The low correlation will limit our model's ability to leverage these features effectively for fraud detection. Low-correlation features also add noise and may complicate model training, reducing model accuracy and interpretability.

Limitation 2b: Limited features to explain client consumption patterns. The combined 17 features may not capture sufficient variability or complexity in client behaviour to effectively identify fraud. This can restrict the model's predictive power.

Solution 2: To improve the feature space, we aggregated key variables at the client level, dropped low-correlation and low-significance features (e.g. date, id and counter_coefficient), and engineered new variables to enhance the representation of each client's consumption behaviour. Through exploratory data analysis, we explored the linear correlations and p-values of features to target variable (fraud), by performing point-biserial correlation on numerical and ordinal features, and chi-square test on categorical features. We also performed a decision tree to obtain the feature importance and to find out the non-linear correlation of features to the target. Key newly engineered features included:

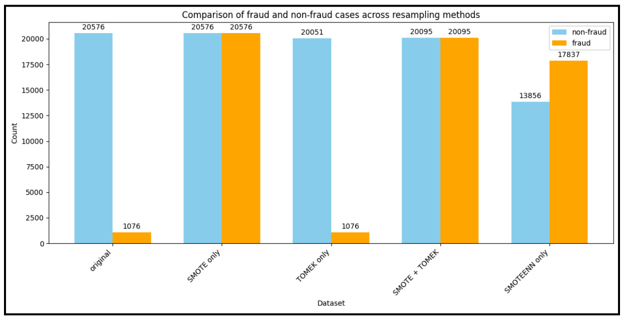
- **Consumption Levels (1-4)**: Aggregated as sum, average, standard deviation, maximum, and minimum values to capture distributional insights.
- **Reading Remarks**: Averaged to reflect ordinal trends.
- **Counter statue:** Averaged to reflect ordinal trends.
- **Tarif Type**: Mode type (categorical) and count of unique values (represent the variability of client's tarif type).
- **Counter Type**: Mode type (categorical) and unique counts (represent the variability of the client's countertype).
- **Months Number**: Mode month (categorical) and unique counts (represent how many different months the client had invoices).
- **Transaction Patterns**: Total transactions, average monthly transactions, and transaction period length based on first and last transactions.

Then, we transformed our dataset. Handling categorical variables through one-hot encoding (e.g. 'dis', 'catg', 'region', 'months_number_mode', 'counter_type_mode'). Standardisation of numerical variables to ensure consistent scale across all numerical inputs. Finally, the data was split into training and testing sets (80:20 ratio).

Handling imbalanced dataset

Limitation 3: Highly imbalanced set. Fraud: 1076, Non-fraud: 20576, resulting in a class imbalance ratio of 1:19. This imbalance presents challenges, as models trained on such data tend to favour the majority class (non-fraud), which could lead to high overall accuracy but poor detection of fraud cases. Without addressing this imbalance, the model may achieve low recall and precision for fraud cases, thus failing to effectively identify fraudulent activity.

Solution 3: To mitigate the imbalance issue, we implemented both oversampling (SMOTE) and undersampling techniques (Tomek Links, Edited Nearest Neighbours (ENN)). SMOTE helps to synthetically increase the minority class (fraud), while Tomek Links and ENN remove noisy or borderline majority samples. This dual approach of oversampling and strategic undersampling helps achieve a dataset that is both balanced and less noisy, ultimately leading to improved classifier performance in identifying fraudulent transactions.



We employed both combinations of balancing strategies in our model (SMOTE+Tomek & SMOTE+ENN), to find out which combination will produce better performance metrics across all our models (KNN, Logistic regression, Random Forest and Feed Forward Neural Network).

Method

In this project, each training dataset is balanced using resampling methods (SMOTE+ENN and SMOTE+Tomek) to address the class imbalance, ensuring the model learns effectively without bias. However, cross-validation is performed on the original imbalance data to better simulate real-world conditions, where fraud cases remain rare. By doing so, we aimed to optimise model performance that mirrors actual deployment scenarios, ensuring that the model's metrics are realistic and reflect its behaviour on truly imbalanced data. Although performance is not ideal, this approach helps to prevent models from overfitting to balanced synthetic training data and allows us to better tune model parameters to an imbalanced dataset.

When tuning model parameters, we used a tailored parameter grid for each model. RandomizedSearchCV, which explores five random parameter combinations from the grid, was used to identify the best combination of hyperparameters from a wide search space, optimising the model efficiently. We employed stratified 3-fold cross-validation to validate model robustness and minimise overfitting, ensuring consistent performance across various data subsets.

For each fold of cross-validation, we also performed threshold tuning to optimise the threshold for the classification of fraud cases and to balance precision and recall more effectively.

Throughout the process, we prioritise the F2 score to find the best hyperparameter, as reducing false negatives is crucial for effective fraud detection.

KNN

K-Nearest Neighbors (KNN) is a non-parametric, instance-based learning algorithm that classifies data points by assessing their similarity (distance) to the nearest data points in the feature space. In fraud detection, KNN can be effective at capturing localised patterns in data, such as isolated clusters of fraud cases or anomalies. Because KNN makes no assumptions about the underlying data distribution, it can adapt well to non-linear and complex relationships, making it suitable for detecting fraud cases that do not follow typical patterns.

Optimization

- **n_neighbors**: [21, 148, 2] 21 to 147 in steps of 2 - Controls the number of neighbours considered in classification.
- **weights**: ['uniform', 'distance'] - Distance-based weighting can improve classification by giving more influence to closer neighbours.
- **metric**: ['euclidean', 'manhattan'] - Distance metrics for measuring similarity between data points.
- **algorithm**: ['auto', 'kd_tree'] - Algorithms to speed up nearest-neighbour searches, with kd_tree often faster in higher dimensions.
- **leaf_size**: [10, 30] - Impacts the speed of tree-based searches (KD-tree, Ball-tree) for neighbour search efficiency.

Logistic Regression (LR)

Logistic regression is a widely used statistical method for binary classification problems, where the goal is to predict the probability that a given instance belongs to one of two classes. It extends linear regression to classification tasks by using a logistic (sigmoid) function, ensuring the output is bounded between 0 and 1, which can be interpreted as a probability.

Optimisation:

- solver: ['liblinear', 'saga'] – 'liblinear' works well with small datasets and supports l1 and l2 penalties, while 'saga' is suitable for larger datasets and supports elasticnet.
- **penalty:** ['l1', 'l2', 'elasticnet'] – 'l1' applies Lasso regularisation, driving some coefficients to zero, improving feature selection. 'l2' (Ridge) keeps coefficients small to prevent overfitting. 'elasticnet' combines both, balancing sparsity and regularisation.
- **C:** [0.01, 0.1, 1, 10] – Controls the inverse strength of regularisation. A smaller value increases regularisation to prevent overfitting but may reduce model flexibility.
- **class_weight:** ['balanced'] – Adjusts weights inversely proportional to class frequencies, ensuring the model focuses on minority classes such as fraud.
- **max_iter:** [100, 500] – Sets the maximum number of iterations allowed for the solver to converge, ensuring the model reaches optimal coefficients.
- **fit_intercept:** [True] – Adds an intercept term to the model, improving performance when the data isn't centred.
- **tol:** [1e-4, 1e-3] – Sets the convergence tolerance, controlling the threshold at which training stops to prevent excessive iterations.
- **l1_ratio:** [0.1, 0.5, 0.9] (for 'elasticnet') – Defines the balance between L1 and L2 penalties, controlling how sparse the model should be.

Random Forest (RF)

Random Forest is an ensemble learning method that constructs multiple decision trees during training. Each tree is trained on random subsets of data and features, improving robustness and reducing overfitting. RF is especially useful in fraud detection due to its ability to capture complex patterns in imbalanced data and handle a large number of features efficiently.

Optimization:

- **n_estimators:** [50, 100] - controls the number of decision trees in the forest.
- **max_depth:** [10, 15] - limit how deep each tree can grow to prevent overfitting.
- **min_samples_split:** [5, 10] - minimum number of samples required to split a node. Higher values prevent overfitting by making the splits less frequent.
- **min_samples_leaf:** [2, 4] - sets the minimum number of samples required to form a leaf. Higher values smooth out predictions.
- **max_features:** ['sqrt', 'log2'] - determines the number of features considered for splitting at each node which helps to control model variance.
- **class_weight:** ['balanced'] - adjust the weight inversely proportional to class frequencies, addressing class imbalance critical in fraud.
- **bootstrap:** ['True'] - enables sampling with replacement for each tree, ensuring variety across trees and improving generalisation.
- **criterion:** ['gini'] - to measure node quality by impurity, which determines the best split. It helps the model decide which feature is most informative at each step.

Feedforward Neural Network (FNN)

A Feedforward Neural Network (FNN) is a deep learning model with multiple layers, where data flows in one direction from input to output. While FNNs can learn complex patterns, they are more sensitive to data quality, making them harder to train effectively on imbalanced datasets like fraud detection.

Optimization:

- **optimizers**: ['adam', 'rmsprop', 'nadam', 'adamax', 'sgd'] - control the gradient descent algorithm to minimise loss.
- **learning_rate:** [0.001, 0.01, 0.1] - determines how quickly the model updates weights during training.
- **dropout_rate:** [0.2, 0.3, 0.5, 0.7] - add regularisation to prevent overfitting by randomly dropping neurons during training.
- **epochs:** [10, 20, 30, 50] - number of times the model trains over the entire dataset.
- **batch_size:** [8, 16, 32, 64] - number of samples used per gradient update, affecting convergence speed.

Results and Discussions

The better resampling method

	Mean performance scores across all models		
Sampling method	F2 score	Precision	Recall
SMOTE+ENN	0.3025	0.0975	0.6700
SMOTE+Tomek	0.3025	0.1000	0.6200

Comparing sampling methods (SMOTE+ENN and SMOTE+Tomek), SMOTE+ENN performs slightly better as it generally enhances recall without reducing much precision.

Comparing across models

Our criterion for evaluation would be placing heavier emphasis on **recall** and **F2 scores** because to businesses, missing actual positive cases (false negatives) is more costly than incorrectly predicting positive cases.

Mean performance metrics summary of models (SMOTE + ENN):

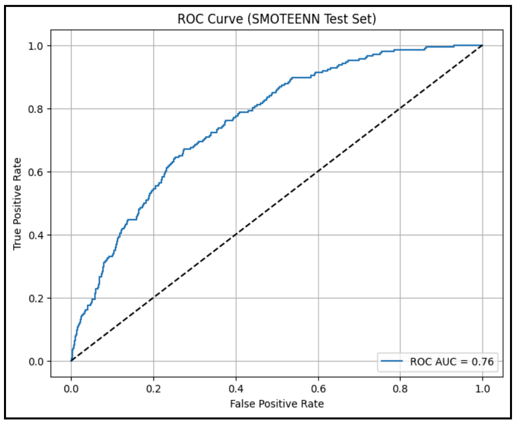
Performance metrics	**KNN**	**LR**	**RF**	**FNN**
F1	0.16	0.18	0.17	0.16
F2	0.29	0.31	0.32	0.29
Precision	0.09	0.11	0.10	0.09
Recall	0.67	0.62	0.71	0.68
PR AUC	0.12	0.12	0.12	0.10

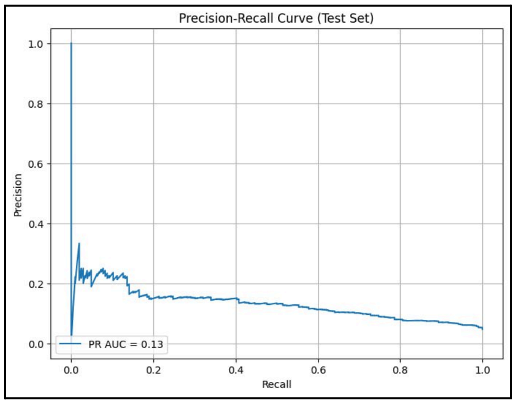
From our results, our Random Forest model is the best performing. It has the highest F2 (0.32) and recall (0.71) scores, and comparable PR AUC (0.12) compared to the other models. This higher recall suggests RF is more effective at identifying fraud cases, which is critical based on our criterion. However, it still has relatively low precision (0.10), which means that although they successfully capture ~71% of actual fraud cases, the majority of cases flagged as fraud are false positives. High false positives have limitations as it can lead to operational inefficiency, increased operational costs, alert fatigue, negative customer experience and loss of customers' trust.

Final Model and Evaluation

After training, the best Random Forest parameters are: {'n_estimators': 100, 'min_samples_split': 10, 'min_samples_leaf': 4, 'max_features': 'log2', 'max_depth': 10, 'criterion': 'gini', 'class_weight': 'balanced', 'bootstrap': True}.

The strengths of Random Forest (IBM, n.d.) are its effectiveness with Imbalanced Data; using class weighting and training with SMOTE+ENN, RF ensures that both fraudulent and non-fraudulent patterns are captured effectively. It prioritises recall through the F2 score, minimising the risk of missing fraudulent cases. Another important strength is its ability to perform feature importance selection, which helps identify key indicators of fraud. It can handle complex Data Relationships, capturing non-linear interactions and dependencies within the data, suitable for subtle fraud detection patterns. Finally, it is also resistant to Noise and Outliers. The ensemble nature reduces the impact of noisy data, crucial for detecting fraud amidst inconsistent records. For these reasons, it is no surprise that RF performs the best among the other models.





Test set	Random Forest
Accuracy	0.74
F1 score	0.19
F2 score	0.33
Precision	0.11
Recall	0.65
PR AUC	0.14
ROC AUC	0.76

The Random Forest model demonstrates mixed results in fraud detection. The low F1 Score (0.19) and low PR AUC (0.14) reflects our model's difficulty in balancing precision and recall. The higher F2 Score (0.33) shows a slight improvement in capturing fraudulent cases, though improvements are needed. Although precision is low (0.11), indicating many false positives, recall is relatively high (0.65), effectively identifying many fraud cases. The ROC AUC (0.76) suggests the model generally distinguishes well between fraud and non-fraud cases.

Our final performance metric scores on the test set, although slightly better than the training set which shows reduced overfitting, indicate that our RF model still does not perform well. There are a few reasons for their sub-performance:

1) **Imbalanced Dataset**: The extreme imbalance in fraud cases (minority class) presents inherent challenges. While resampling techniques like SMOTE+ENN aimed to balance the classes, they may have introduced noise, especially if the synthetic minority samples do not accurately represent real fraud patterns. Furthermore, the limited representation of minority cases in the original dataset may prevent the model from learning the decision boundaries necessary for effective fraud classification.
2) **Low Feature-Target Correlation**: Despite feature engineering and aggregation of invoice data to better capture clients' transaction patterns, the correlation between these engineered features and the target variable remains low. *(Refer to Correlation of X and Y variables code folder, cell 5.)* This low correlation suggests that the features may lack sufficient information to differentiate fraudulent from non-fraudulent behaviour. Without strong associations to the target, the model struggles to effectively distinguish between classes, impacting overall predictive performance.

To address these limitations, we propose collaborating with **domain experts** to develop domain-specific features. These features, informed by industry knowledge, could better highlight patterns or behaviours indicative of fraud, potentially providing the model with more relevant information and improving its classification accuracy.

References

Din, J., Su, H., Ali, S., & Salman, M. (2024). Research on Blockchain-Enabled Smart Grid for Anti-Theft Electricity Securing Peer-to-Peer Transactions in Modern Grids. *Sensors, 24*(5), 1668.

MB Soultanzadeh et al. (2024), in "Fault Detection and Diagnosis in Light Commercial Buildings' HVAC Systems"

Zainab Saad Rubaidi. (n.d.). Fraud detection using large-scale imbalance dataset

IBM. (n.d.). Random forest. IBM.