IEEE Fraud MLMs

Ferris Atassi and Charles Hang

December 2024

1 Introduction

Name: Ferris Atassi + Charles Hang
ESE 527 Practicum
Date: 12/12/2024

Input Space

The initial dataset is represented as:

$$\mathbb{X} = \mathbb{R}^{n \times d}, \quad \mathbb{Y} = \{0, 1\}$$

where n is the number of samples (590540) and d is the number of features (394).

Feature Engineering Morphisms

1. Sorting by TransactionID and TransactionDT

The dataset is first sorted by ‘TransactionID‘ and ‘TransactionDT‘ to ensure temporal ordering.

$$\mathcal{M}_{\text{sort}} : \mathbb{X}_{\text{raw}} \to \mathbb{X}_{\text{sorted}}$$

$$F_{\text{sort}}(\mathbf{x}; \Theta_{\text{sort}}) = \text{sort}(\mathbf{x}, \text{by} = [\text{TransactionID}, \text{TransactionDT}])$$

2. Transaction Density Feature

A new feature is created as the ratio of ‘TransactionAmt‘ to ‘TransactionWeek‘:

$$\mathcal{M}_{\text{density}} : \mathbb{X}_{\text{sorted}} \to \mathbb{X}_{\text{density}}$$

$$F_{\text{density}}(\mathbf{x}; \Theta_{\text{density}}) = \mathbf{x}_{\text{TransactionAmt}} / \mathbf{x}_{\text{TransactionWeek}}$$

3. Day Interaction Feature

A new feature is created as the product of ‘TransactionAmt‘ and ‘TransactionDay‘:

$$\mathcal{M}_{\text{day\_interaction}} : \mathbb{X}_{\text{density}} \to \mathbb{X}_{\text{interaction}}$$

$$F_{\text{day\_interaction}}(\mathbf{x}; \Theta_{\text{day\_interaction}}) = \mathbf{x}_{\text{TransactionAmt}} \cdot \mathbf{x}_{\text{TransactionDay}}$$

4. Device Interaction Feature (if DeviceInfo exists)

If ‘DeviceInfo‘ is present, create a new feature based on a hash of the device information:

$$\mathcal{M}_{\text{device\_interaction}} : \mathbb{X}_{\text{interaction}} \to \mathbb{X}_{\text{device}}$$

$$F_{\text{device\_interaction}}(\mathbf{x}; \Theta_{\text{device\_interaction}}) = \mathbf{x}_{\text{TransactionAmt}} \cdot \text{hash}(\mathbf{x}_{\text{DeviceInfo}}) \% 10$$

5. Binned Transaction Amount

The ‘TransactionAmt‘ feature is discretized into bins:

$$\mathcal{M}_{\text{binned}} : \mathbb{X}_{\text{device}} \to \mathbb{X}_{\text{binned}}$$

$$F_{\text{binned}}(\mathbf{x}; \Theta_{\text{binned}}) = \text{bin}(\mathbf{x}_{\text{TransactionAmt}}, \text{bins} = [-1, 50, 100, 200, 500, 1000, \infty])$$

6. Ratios with Day and Week Features

Two new features are created: 1. ‘Amt$_To_DayOfWeek_Ratio$‘ : $\mathcal{M}_{\text{day\_ratio}} : \mathbb{X}_{\text{binned}} \to \mathbb{X}_{\text{day\_ratio}}$

$$F_{\text{day\_ratio}}(\mathbf{x}; \Theta_{\text{day\_ratio}}) = \mathbf{x}_{\text{TransactionAmt}}/(\mathbf{x}_{\text{TransactionDayOfWeek}} + 1)$$

2. ‘Amt$_To_Week_Ratio$‘ : $\mathcal{M}_{\text{week\_ratio}} : \mathbb{X}_{\text{day\_ratio}} \to \mathbb{X}_{\text{week\_ratio}}$

$$F_{\text{week\_ratio}}(\mathbf{x}; \Theta_{\text{week\_ratio}}) = \mathbf{x}_{\text{TransactionAmt}}/(\mathbf{x}_{\text{TransactionWeek}} + 1)$$

7. Aggregated Card Features

Aggregate ‘TransactionAmt‘ by ‘card1‘ to compute mean, standard deviation, and count:

$$\mathcal{M}_{\text{card\_agg}} : \mathbb{X}_{\text{week\_ratio}} \to \mathbb{X}_{\text{card\_agg}}$$
$$F_{\text{card\_agg}}(\mathbf{x}; \Theta_{\text{card\_agg}}) = \text{groupby}(\mathbf{x}_{\text{card1}}, \text{agg} = [\text{mean}, \text{std}, \text{count}])$$

8. Card4 Frequency Mapping

Map ‘card4‘ to its frequency:

$$\mathcal{M}_{\text{card4\_freq}} : \mathbb{X}_{\text{card\_agg}} \to \mathbb{X}_{\text{freq}}$$
$$F_{\text{card4\_freq}}(\mathbf{x}; \Theta_{\text{card4\_freq}}) = \text{map}(\mathbf{x}_{\text{card4}}, \text{frequency\_dict})$$

Final Feature Engineering Workflow

The feature engineering pipeline can be expressed as:

$$\mathcal{M}_{\text{feature}} = \mathcal{M}_{\text{card4\_freq}} \circ \mathcal{M}_{\text{card\_agg}} \circ \mathcal{M}_{\text{week\_ratio}} \circ \mathcal{M}_{\text{day\_ratio}} \circ \mathcal{M}_{\text{binned}} \circ \mathcal{M}_{\text{device\_interaction}} \circ \mathcal{M}_{\text{day\_interaction}} \circ \mathcal{M}_{\text{density}}$$

where:

$$F_{\text{feature}}(\mathbf{x}; \Theta_{\text{feature}}) = F_{\text{card4\_freq}}(F_{\text{card\_agg}}(F_{\text{week\_ratio}}(F_{\text{day\_ratio}}(F_{\text{binned}}(F_{\text{device\_interaction}}(F_{\text{day\_interaction}}(F_{\text{density}}($$

Output Space After Feature Engineering:

$$\mathbb{X}_{\text{feature\_engineered}} = \mathbb{R}^{n \times d_{\text{engineered}}}$$

Outlier Removal Morphisms

1. Isolation Forest Outlier Removal

The Isolation Forest morphism identifies outliers in the dataset by isolating anomalous data points.

$$\mathcal{M}_{\text{iso}} : \mathbb{X}_{\text{raw}} \to \mathbb{X}_{\text{iso}}$$

The isolation forest morphism is defined as:

$$F_{\text{iso}}(\mathbf{x}; \Theta_{\text{iso}}) = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is normal} \\ -1 & \text{if } \mathbf{x} \text{ is an anomaly} \end{cases}$$

where $\Theta_{\text{iso}} = \{n_{\text{estimators}}, \text{max\_samples}, \text{contamination}, \text{random\_state}\}$.

$$\mathcal{L}_{\text{iso}}(x; \Theta_{\text{iso}}) = \begin{cases} 0 & \text{if } F_{\text{iso}}(x; \Theta_{\text{iso}}) = 1 \\ 1 & \text{if } F_{\text{iso}}(x; \Theta_{\text{iso}}) = -1 \end{cases}$$

Steps: 1. Train the Isolation Forest on $\mathbb{X}_{\text{numeric}}$ to predict anomalies. 2. Filter out rows where $F_{\text{iso}}(\mathbf{x}) = -1$ (anomalies). 3. The output is $\mathbb{X}_{\text{iso}}$, the cleaned dataset.

2. Mahalanobis Distance Outlier Removal

The Mahalanobis Distance morphism identifies outliers based on their multivariate distance from the mean.

$$\mathcal{M}_{\text{maha}} : \mathbb{X}_{\text{numeric}} \to \mathbb{X}_{\text{maha}}$$

The Mahalanobis distance morphism is defined as:

$$F_{\text{maha}}(\mathbf{x}; \Theta_{\text{maha}}) = d_M(\mathbf{x}, \mu, \mathbf{\Sigma}^{-1})$$

where: - d_M is the Mahalanobis distance:

$$d_M(\mathbf{x}, \mu, \mathbf{\Sigma}^{-1}) = \sqrt{(\mathbf{x} - \mu)^\top \mathbf{\Sigma}^{-1} (\mathbf{x} - \mu)}$$

- μ is the mean of the dataset. - $\mathbf{\Sigma}^{-1}$ is the inverse of the covariance matrix.

$$\mathcal{L}_{\text{maha}}(x; \Theta_{\text{maha}}) = \begin{cases} 0 & \text{if } d_M(x, \mu, \Sigma^{-1}) \leq \text{threshold} \\ 1 & \text{if } d_M(x, \mu, \Sigma^{-1}) > \text{threshold} \end{cases}$$

Steps: 1. Compute the covariance matrix $\mathbf{\Sigma}$ and its inverse $\mathbf{\Sigma}^{-1}$. 2. Calculate the Mahalanobis distance for each point. 3. Define a threshold (e.g., 97.5th percentile) to classify points as outliers. 4. Remove points where $F_{\text{maha}}(\mathbf{x}) > \text{threshold}$. 5. The output is $\mathbb{X}_{\text{maha}}$, the cleaned dataset.

Combined Outlier Removal Workflow

The combined outlier removal pipeline is expressed as:

$$\mathcal{M}_{\text{outlier}} = \mathcal{M}_{\text{iso}} \circ \mathcal{M}_{\text{maha}}$$

where:

$$F_{\text{outlier}}(\mathbf{x}; \Theta_{\text{outlier}}) = F_{\text{iso}}(F_{\text{maha}}(\mathbf{x}; \Theta_{\text{maha}}); \Theta_{\text{iso}})$$

$$\mathcal{L}_{\text{outlier}}(x; \Theta_{\text{outlier}}) = \mathcal{L}_{\text{iso}}(x; \Theta_{\text{iso}}) + \mathcal{L}_{\text{maha}}(x; \Theta_{\text{maha}})$$

Output Space After Outlier Removal:

$$\mathbb{X}_{\text{outlier\_cleaned}} = \mathbb{R}^{n' \times d}, \quad n' \leq n$$

where n' is the number of non-outlier samples.

Morphisms for Preprocessing Steps

1. Timestamp Conversion

For columns with `datetime64[ns]` or containing "Timestamp":

$$\mathcal{M}_{\text{time}} : \mathbb{X}_{\text{raw}} \to \mathbb{X}_{\text{time}}$$

$$F_{\text{time}}(\mathbf{x}; \Theta_{\text{time}}) = \text{int64}(\text{to\_datetime}(\mathbf{x}))$$

2. Label Encoding

For categorical columns:

$$\mathcal{M}_{\text{label}} : \mathbb{X}_{\text{time}} \to \mathbb{X}_{\text{label}}$$

$$F_{\text{label}}(\mathbf{x}; \Theta_{\text{label}}) = \text{LabelEncoder}(\mathbf{x})$$

where each categorical column c is transformed into integers using Θ_{label}, the fitted label encoder.

3. Replacement of Infinite Values

For replacing `inf` and `-inf` values:

$$\mathcal{M}_{\text{replace}} : \mathbb{X}_{\text{label}} \to \mathbb{X}_{\text{finite}}$$

$$F_{\text{replace}}(\mathbf{x}; \Theta_{\text{replace}}) = \mathbf{x}[\mathbf{x} \neq \pm\infty]$$

4. Conversion to Numeric

For converting all columns to numeric types:

$$\mathcal{M}_{\text{numeric}} : \mathbb{X}_{\text{finite}} \to \mathbb{X}_{\text{numeric}}$$

$$F_{\text{numeric}}(\mathbf{x}; \Theta_{\text{numeric}}) = \text{to\_numeric}(\mathbf{x})$$

5. Handling Missing Values

Numeric columns:

$$\mathcal{M}_{\text{num\_fill}} : \mathbb{X}_{\text{numeric}} \to \mathbb{X}_{\text{num\_filled}}$$

$$F_{\text{num\_fill}}(\mathbf{x}; \Theta_{\text{num\_fill}}) = \mathbf{x}[\text{fill\_na}(\text{mean}(\mathbf{x}))]$$

Non-numeric columns:

$$\mathcal{M}_{\text{cat\_fill}} : \mathbb{X}_{\text{num\_filled}} \to \mathbb{X}_{\text{filled}}$$

$$F_{\text{cat\_fill}}(\mathbf{x}; \Theta_{\text{cat\_fill}}) = \mathbf{x}[\text{fill\_na}(\text{mode}(\mathbf{x}))]$$

Final Preprocessing Workflow Morphism

The preprocessing pipeline can be expressed as a composition of the individual morphisms:

$$\mathcal{M}_{\text{preprocess}} = \mathcal{M}_{\text{cat\_fill}} \circ \mathcal{M}_{\text{num\_fill}} \circ \mathcal{M}_{\text{numeric}} \circ \mathcal{M}_{\text{replace}} \circ \mathcal{M}_{\text{label}} \circ \mathcal{M}_{\text{time}}$$

where:

$$F_{\text{preprocess}}(\mathbf{x}; \Theta_{\text{preprocess}}) = F_{\text{cat\_fill}}(F_{\text{num\_fill}}(F_{\text{numeric}}(F_{\text{replace}}(F_{\text{label}}(F_{\text{time}}(\mathbf{x}; \Theta_{\text{time}}); \Theta_{\text{label}}); \Theta_{\text{replace}}); \Theta_{\text{numeric}});$$

Output Space

After preprocessing:

$$\mathbb{X}_{\text{processed}} = \mathbb{R}^{n \times d_{\text{processed}}}$$

Final Model Morphism

The final model combines three base learners: XGBoost, Random Forest, and Logistic Regression. Each morphism is defined in terms of its input and output spaces.

Input Space:

$$\mathbb{X}_{\text{model}} = \mathbb{R}^d$$

where d is the number of features.

Output Space:

$$\mathbb{Y}_{\text{model}} = [0, 1]$$

representing the probability of the positive class (isFraud $= 1$).

2 Model Ensembler Section

Overall Morphism

The ensembler combines the predictions of the three base models using weighted voting:

$$\mathcal{M}_{\text{vote}} = \mathcal{M}_{\text{xgb}} \circ_w \mathcal{M}_{\text{rf}} \circ_w \mathcal{M}_{\text{lr}}$$

where:

$$F_{\text{vote}}(\mathbf{x}; \Theta) = w_{\text{xgb}} F_{\text{xgb}}(\mathbf{x}; \Theta_{\text{xgb}}) + w_{\text{rf}} F_{\text{rf}}(\mathbf{x}; \Theta_{\text{rf}}) + w_{\text{lr}} F_{\text{lr}}(\mathbf{x}; \Theta_{\text{lr}})$$

$$\mathcal{L}_{\text{vote}}(y, \hat{y}; w_{\text{xgb}}, w_{\text{rf}}, w_{\text{lr}}) = \mathcal{L}_{\text{xgb}}(y, \hat{y}) \cdot w_{\text{xgb}} + \mathcal{L}_{\text{rf}}(y, \hat{y}) \cdot w_{\text{rf}} + \mathcal{L}_{\text{lr}}(y, \hat{y}) \cdot w_{\text{lr}}$$

- $w_{\text{xgb}}, w_{\text{rf}}, w_{\text{lr}}$: Voting weights assigned to XGBoost, Random Forest, and Logistic Regression, respectively. - $\Theta = \{\Theta_{\text{xgb}}, \Theta_{\text{rf}}, \Theta_{\text{lr}}\}$: The parameter set for all three models.

XGBoost Morphism

XGBoost (Extreme Gradient Boosting) is an ensemble of decision trees trained in a boosting framework:

$$F_{\text{xgb}}(\mathbf{x}; \Theta_{\text{xgb}}) = \arg\max_{k \in \{0,1\}} \left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}; \theta_t)_k \right)$$

$$\mathcal{L}_{\text{xgb}}(y, \hat{y}) = -\frac{1}{n} \sum_{i=1}^{n} \left[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \right] + \Omega(\Theta_{\text{xgb}})$$

where:

- T: The number of boosting rounds (trees).
- $h_t(\mathbf{x}; \theta_t)$: The prediction of the t-th tree for input $\mathbf{x}$, parameterized by θ_t.
- α_t: The weight assigned to the t-th tree, learned during training.
- Θ_{xgb}: Hyperparameters such as learning rate, max depth of trees, and regularization terms.

Logistic Regression Morphism

Logistic Regression is a linear model for binary classification:

$$F_{\text{lr}}(\mathbf{x}; \Theta_{\text{lr}}) = \arg\max_{k \in \{0,1\}} \sigma_k(\mathbf{w}^\top \mathbf{x} + b)$$

$$\mathcal{L}_{\mathrm{lr}}(y, \hat{y}) = -\frac{1}{n} \sum_{i=1}^{n} \left[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \right]$$

where:

- $\mathbf{w}$: Coefficient vector for the features.
- b: Bias term (intercept).
- $\sigma_k(z) = \frac{1}{1+e^{-z}}$: Sigmoid function that outputs probabilities for $k \in \{0, 1\}$.
- Θ_{lr}: Model parameters $(\mathbf{w}, b)$ learned during training.

Random Forest Morphism

Random Forest is an ensemble of decision trees trained independently:

$$F_{\mathrm{rf}}(\mathbf{x}; \Theta_{\mathrm{rf}}) = \arg \max_{k \in \{0,1\}} \left(\frac{1}{T} \sum_{t=1}^{T} h_t(\mathbf{x}; \theta_t)_k \right)$$

$$\mathcal{L}_{\mathrm{rf}}(y, \hat{y}) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{i=1}^{n} \left[y_i \log(h_t(x_i)) + (1 - y_i) \log(1 - h_t(x_i)) \right]$$

where:

- T: The number of decision trees in the forest.
- $h_t(\mathbf{x}; \theta_t)$: The prediction of the t-th tree, parameterized by θ_t.
- Θ_{rf}: Hyperparameters such as number of trees, max depth, and criteria for splitting nodes.

Output of the Ensembler

After combining predictions from all three models using the weighted voting strategy, the ensembler outputs:

$$\mathbb{Y}_{\text{vote}} = [0, 1]$$

where the final prediction is a probability score for the positive class (isFraud = 1).