CREDIT CARD FRAUD DETECTION

ADS_PHASE 4.

ABSTRACT

This paper is Phase 4 of Himanshu Kumar's "Credit Card Fraud Detection" project, which he presented as a third-year Computer Science And Engineering student. Phase 4 is dubbed "Development Part 2," and it consists of Feature engineering, Model Training, Evaluation.

Himanshu Kumar

Applied Data Science

❖ Table of Contents:

Of course, let's delve deeper into each phase of developing a project to identify credit card fraud:

❖ Feature Engineering:

- **Feature Selection:**
  - To begin, use exploratory data analysis (EDA) to ascertain the attributes' qualities and distribution. Decide which attributes are relevant for detecting fraud.
  - To determine which aspects are most crucial, apply methods such as mutual information, correlation analysis, or feature importance from tree-based models.
  - Get rid of features that add noise to the model or are not informative.
- **Feature Scaling:**
  - Common techniques for feature scaling include standardization (Z-score normalization) and min-max scaling. Features that undergo standardization have a mean of 0 and a standard deviation of 1.
  - By ensuring that features with varying sizes contribute to the model evenly, scaling helps to avoid some characteristics from controlling the learning process.
- **Time-Based Features:**

  Extrapolate significant time-related attributes:
  - Day of the week: This helps identify fraud trends based on day of the week.
  - Time of day: Fraud rates may fluctuate from one hour to the next.
  - The amount of time since the last transaction: This can be used to identify the temporal components of fraud trends.

- **Anomaly Detection Features:**

    - Make use of unsupervised anomaly detection methods such as One-Class SVM, Local Outlier Factor (LOF), and Isolation Forest to generate features that represent each transaction's level of oddity.

    - These anomaly scores can function as characteristics that offer further details on the probability that a transaction is fraudulent.

- **Cross-Feature Interactions:**

  Examine how different elements combine to depict intricate relationships:

    - Provide features that allow for interaction, such a product or the division of two or more features (e.g., merchant score × amount).

    - Features of polynomials: Incorporate words that are squared or cubed for certain attributes to identify non-linear correlations.

❖ Model Training:

- **Data Split:**

    - Divide your data into sets for testing, validation, and training. Utilizing 60–70% for training, 15-20% for validation, and the remaining 15-20% for testing is standard procedure.

- **Baseline Models:**

    - Start with basic models that are easy to understand and can be used as a baseline, such logistic regression.

    - Utilizing the training data, train the model, then assess its performance with the proper assessment metrics (accuracy, precision, recall, and F1-score).

- **Hyperparameter Tuning:**

- Hyperparameter tweaking is necessary for increasingly intricate models such as random forests, decision trees, and deep learning models. To determine the ideal model configuration, this entails experimenting with various combinations of hyperparameters.

- Methods such as random and grid search can assist in automating this procedure.

- **Ensemble Methods:**

  - To aggregate the predictions of several models, take into consideration ensemble techniques like gradient boosting and random forests.

  - These techniques can increase prediction accuracy by utilizing the variety of different models.

- **Deep Learning:**

  - Create designs for deep learning models that use recurrent neural networks (RNNs) or convolutional neural networks (CNNs) to identify complex patterns in the data.

  - Adjust the architecture of the neural network, taking into account the quantity of layers, neurons, and activation functions.

❖ Evaluation:

- **Confusion Matrix:**

  Make use of a confusion matrix to determine different metrics:

  Precision: TP / (TP + FP)

  Recall (Sensitivity): TP / (TP + FN)

  F1-Score: 2 * (Precision * Recall) / (Precision + Recall)

  Accuracy: (TP + TN) / (TP + TN + FP + FN)

- **ROC Curve and AUC:**

- Change the model's decision threshold to plot the Receiver Operating Characteristic (ROC) curve.

- To evaluate the model's capacity to distinguish between positive (fraud) and negative (non-fraud) examples, compute the Area Under the Curve (AUC).

- **Cross-Validation:**

    - The model's generalization performance may be assessed by using k-fold cross-validation, such as 5-fold or 10-fold.

    - Assessing the model's performance on new, untested data becomes easier with this.

- **Anomaly Detection Techniques:**

    - To catch unusual events and outliers, combine your classification models with anomaly detection models like One-Class SVM or Isolation Forest.

- **Business Metrics:**

    - Evaluate the practical implications of your model's functionality. Think about KPIs unique to your organization, such as the expense associated with false positives and false negatives.

    - To achieve the ideal balance between detection and operating expenses, optimize the model using these business KPIs.

Always remember to improve your model over time, iterate on these processes, and update it frequently to accommodate changing fraud trends. Sustaining efficacy requires continuous monitoring and enhancement of a strong fraud detection system.