CREDIT CARD FRAUD DETECTION

Problem Statement:

The problem is to develop a machine learning-based system for real-time credit card fraud detection. The goal is to create a solution that can accurately identify fraudulent transaction while minimizing false positives. This project involves data preprocessing, feature engineering, model selection, training and evaluation to create a robust fraud detection system

data sources:

1. Transaction Data :

Information about individual credit card transactions, including the
amount, location, time, and merchant.

2. Customer Data :

Data related to the credit card holder, including their profile, transaction history, and spending patterns.

3. Device Information :

Details about the device used for the transaction, such as the device type, IP address, and geolocation data.

4. Historical Data :

Historical transaction data to establish a baseline for normal behaviour and detect anomalies.

5. Merchant Data :

Information about the merchant, including their reputation and history of fraudulent transactions.

These data sources are typically combined and analysed using machine learning algorithms and anomaly detection techniques to identify potentially fraudulent transactions. It's important to note that the effectiveness of fraud detection systems often depends on the quality and quantity of data available for analysis.

Data preprocessing :

1. Data Cleaning :

Handling Missing Values: Identify and handle missing values in the dataset. You can either remove rows with missing values or impute them using techniques like mean, median, or interpolation.

Outlier Detection: Detect and deal with outliers in the data, which could be indicative of fraudulent activity or data entry errors.

2. Feature Selection :

Identify and select relevant features (attributes) for fraud detection. Not all features may be useful, and selecting the right ones can improve model performance and reduce computational overhead.

3. Data transformation :

Scaling: Normalize or standardize numerical features to bring them to a similar scale. This is important for algorithms sensitive to feature scales, like k-nearest neighbours or support vector machines.

Encoding Categorical Variables : Convert categorical variables into numerical representations, often using techniques like one-hot encoding or label encoding.

4. Data Splitting :

Split the data into training, validation, and test sets. The training set is used to train the model, the validation set is used for tuning hyperparameters, and the test set is used to evaluate model performance.

5. Handling Class Imbalance :

Credit card fraud datasets are often highly imbalanced, with a small percentage of fraudulent transactions. Apply techniques like oversampling (adding more copies of minority class samples) or under sampling (removing some majority class samples) to address class imbalance.

6. Feature Scaling :

Normalize or standardize numerical features to bring them to a similar scale. This helps models like logistic regression and k-nearest neighbours perform better.

7. Dimensionality Reduction :

If you have a high-dimensional dataset, consider applying dimensionality reduction techniques like Principal Component Analysis (PCA) to reduce computational complexity while preserving important information.

Feature engineering :

1. Transaction Amount Normalization :

Calculate the average transaction amount and standard deviation for each cardholder. Then, for each transaction, create a feature that represents how many standard deviations the transaction amount is from the cardholder's average. This can help identify unusual spending patterns.

2. Time-Based Features :

Create features related to time, such as:

Hour of the day: Transactions may exhibit different patterns at different times.

Day of the week: Fraudulent activity may vary by day.

Time since the last transaction: Unusually short time intervals between transactions can be suspicious.

3. Geographical Features :

Incorporate geolocation data if available. Calculate the distance between the transaction location and the cardholder's typical locations. Unusual distances could indicate fraud.

4. Frequency of Transactions :

Calculate the frequency of transactions for each cardholder within specific time windows (e.g., daily, weekly). Sudden spikes or drops in transaction frequency may be indicative of fraud.

5. Velocity Checks :

Compute the velocity of transactions, such as the number of transactions within a short time window. Rapid transactions could signal fraud.

Model Selection :

1. Logistic Regression :

Logistic regression is a simple and interpretable model that can be a good starting point for binary classification tasks like fraud detection. It's particularly useful when interpretability is essential.

2. Decision Trees and Random Forests :

Decision trees and random forests are suitable for handling both categorical and numerical features. They can capture complex relationships in the data and are robust against overfitting.

3. Gradient Boosting Algorithms :

Algorithms like XGBoost, LightGBM, and CatBoost are powerful ensemble methods that often perform well on imbalanced datasets. They can handle a mixture of feature types and automatically select important features.

4. Support Vector Machines (SVM) :

SVMs can be effective when dealing with high-dimensional data and can handle non-linear decision boundaries. However, they may require careful tuning.

5. Neural Networks :

Deep learning techniques, such as feedforward neural networks or convolutional neural networks (CNNs), can be applied for credit card fraud detection. They are capable of capturing complex patterns but may require a large amount of data to perform well.

Evaluation :

1. Confusion Matrix :

A confusion matrix provides a detailed breakdown of model performance, showing the number of true positives (correctly detected fraud), true negatives (correctly identified non-fraud), false positives (legitimate transactions incorrectly flagged as fraud), and false negatives (fraudulent transactions missed by the model).

2. Accuracy :

Accuracy is the ratio of correct predictions (both true positives and true negatives) to the total number of transactions. While it's a common metric, it can be misleading in highly imbalanced datasets, where most transactions are legitimate.

3. Precision (Positive Predictive Value) :

Precision measures the proportion of predicted fraud cases that are truly fraudulent. It's calculated as TP / (TP + FP). A high precision indicates that when the model flags a transaction as fraud, it's likely to be correct.

4. Recall (Sensitivity, True Positive Rate) :

Recall measures the proportion of actual fraud cases that are correctly detected by the model. It's calculated as TP / (TP + FN). High recall means the model is effective at identifying most fraudulent transactions.

5. F1-Score :

The F1-Score is the harmonic mean of precision and recall, giving a balance between the two metrics. It's calculated as 2 * (Precision * Recall) / (Precision + Recall)