Project Title: Fraud Transaction Detection with Random Forest Classifier

Objective:

The main objective of this project is to detect fraudulent transactions using a Random Forest Classifier. The implementation involves data preprocessing, feature selection, model training with hyperparameter tuning, and performance evaluation using metrics like ROC curve analysis and confusion matrix. .

Working Procedure

The project involves the following key steps:

1. Data Loading and Exploration

- ➢ The dataset is loaded from a CSV file.
- ➢ The first few rows and the shape of the dataset are displayed.
- ➢ The distribution of classes in the dataset is visualized to check for any imbalance.

2. Data Cleaning

- ➢ Missing values are handled by removing rows with NaN values.
- ➢ Infinite values are detected and counted for further handling.

3. Data Preprocessing

- ➢ Features (input data) and the target label (fraudulent or non-fraudulent) are separated.
- ➢ The dataset is split into training and testing sets, with 70% used for training and 30% for testing.
- ➢ Data balancing is applied using SMOTE (Synthetic Minority Over-sampling Technique) or RandomUnderSampler, depending on the choice.
- ➢ The data is then standardized using either StandardScaler or MinMaxScaler to improve model performance.

4. Feature Selection

- ➢ The SelectKBest method with the f_classif scoring function is used to evaluate feature importance.
- ➢ Features with scores above a defined threshold are selected for the model.

5. Model Training and Hyperparameter Tuning

- ➢ The training data is further split into validation and test sets.
- ➢ A Random Forest classifier is trained with hyperparameter tuning using GridSearchCV. Parameters like the number of estimators, depth, and samples per split are optimized.
- ➢ The model with the best parameters is selected based on its AUC-ROC score.

6. Model Evaluation

- ➢ The model's performance is evaluated using the ROC curve, AUC, and optimal threshold determination using Youden's J statistic.
- ➢ The final test set is used for prediction, and performance metrics like accuracy, precision, recall, F1-score, and confusion matrix are computed and visualized.

Achievements

The model effectively detected fraudulent transactions by utilizing optimized parameters. A complete machine learning pipeline, including data preprocessing, model training, and evaluation, was successfully implemented. To address the class imbalance issue, data balancing techniques such as SMOTE were applied. Feature selection was performed to enhance the efficiency of the model. The model's performance was thoroughly evaluated using metrics such as the ROC curve, AUC, and confusion matrix.

Dependences to Run the Code

To run the code, the following Python libraries are required: numpy, pandas, scikit-learn, matplotlib, seaborn, and imblearn. The project can be executed in any Python environment that supports Jupyter Notebooks or .ipynb files.