TASK : Credit Card Fraud Detection Project

Objective

The objective of this project is to develop a machine learning pipeline to detect fraudulent credit card transactions. The project focuses on:

1. Data handling and preprocessing.
2. Building supervised and unsupervised models for classification and anomaly detection.
3. Evaluating the models’ performance using metrics and visualizations.
4. Providing explainability for the supervised model using SHAP.

Dataset

Source: Kaggle Credit Card Fraud Detection Dataset
Description: Contains transactions made by European cardholders in September 2013. The dataset is highly imbalanced, with the majority of transactions being non-fraudulent.
Features:

- Time, Amount: Transaction-specific details.
- V1 to V28: Principal Component Analysis (PCA) transformed features.
- Class: Target variable (0 = Non-fraud, 1 = Fraud).

Steps Implemented

1. Data Exploration and Preprocessing

Loading the Dataset:

- Loaded the dataset using Pandas.
- Performed an initial exploration to check for class distribution and missing values.

Data Scaling:
- Scaled the Time and Amount features using StandardScaler.
- Removed the original Time and Amount columns.

Class Imbalance Handling:

- Applied Synthetic Minority Oversampling Technique (SMOTE) to balance the classes during training.

2. Supervised Model Development

Baseline Model:

- Logistic Regression was implemented as a baseline model.

Primary Model:

- XGBoost classifier was trained with hyperparameter tuning.
- Evaluated on metrics such as Precision, Recall, F1-Score, and ROC-AUC.

3. Unsupervised Model Development

Algorithm:

- Isolation Forest was implemented to detect anomalies (fraudulent transactions).
- Configured to identify rare events as anomalies.

Evaluation:

- Compared detected anomalies with the true labels.

4. Model Evaluation

Supervised Models:

- Generated a confusion matrix to analyze predictions.
- Plotted ROC and Precision-Recall curves for performance visualization.

Unsupervised Model:

- Assessed anomaly detection accuracy by checking identified anomalies against known fraud cases.

5. Model Explainability

SHAP:

- Used SHAP (SHapley Additive exPlanations) to provide insights into feature importance and model predictions.
- Generated summary plots to visualize the most impactful features.

Technologies Used

Programming Language: Python
Libraries:

- Data Handling: Pandas, NumPy
- Machine Learning: scikit-learn, XGBoost, imbalanced-learn

- Explainability: SHAP
- Visualization: Matplotlib, Seaborn

Project Files

1. **Jupyter Notebook**: Contains all the code from data preprocessing to model evaluation and explainability.
2. **Dataset**: creditcard.csv (not included in the repository; needs to be added manually).
3. **Visualizations**:
    - Confusion Matrix
    - ROC and Precision-Recall Curves
    - SHAP Summary Plot

Instructions for Reproducing Results

1. Clone the repository:

```
git clone <repository-link>
```

2. Install required libraries:

```
pip install -r requirements.txt
```

3. Place the dataset (creditcard.csv) in the project directory.
4. Run the Jupyter Notebook:
5. jupyter notebook credit_card_fraudlent.ipynb
6. Follow the steps in the notebook to reproduce the results.

Evaluation Metrics

Supervised Models:

- Accuracy, Precision, Recall, F1-Score
- ROC-AUC Curve

Unsupervised Model:

- Detected anomalies matched against known fraudulent transactions.

Results

1. **Supervised Models**:
   - Logistic Regression:

     Precision: *x*
     Recall: *y*
     F1-Score: *z*

   - XGBoost:

     Precision: *a*
     Recall: *b*
     F1-Score: *c*

2. **Unsupervised Model**:
   - Isolation Forest successfully identified *d* anomalies.

3. **Explainability**:
   - Key features impacting fraud detection: V1, V2, scaled_amount, etc.

Future Work

Experiment with other unsupervised models like Autoencoders.
Apply advanced techniques for class imbalance, such as ensemble methods.
Deploy the model using Flask/Django for real-time fraud detection.

Acknowledgments

Kaggle for providing the dataset.
Open-source libraries and frameworks used in this project.