Fraud Detection Model Development

Intent

The goal is to build a predictive model to identify fraudulent transactions from a dataset of credit card transactions. I aim to use a robust classification algorithm to achieve high accuracy and reliability in distinguishing between fraudulent and non-fraudulent transactions.

Methods

Data Loading and Preprocessing

- Loaded the dataset from a JSON file.
- Checked for empty strings and NaN values and handled them by replacing empty strings with NaNs and dropping columns with more than 50% NaNs, followed by rows with any NaNs.
- Categorical columns were label encoded.
- Date columns were converted to datetime objects and additional features were engineered from these dates.

Handling Imbalanced Data

- Used SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset as fraud cases are significantly fewer compared to non-fraud cases.

Model Training and Evaluation

- Split the data into training and testing sets.
- Trained a RandomForestClassifier, taking into account class imbalance.
- Evaluated the model using metrics such as accuracy, AUC-ROC, classification report, and confusion matrix.
- Identified feature importances to understand which features contribute most to the model's predictions.

Comparison with XGBoost

- Compared the performance of RandomForest with XGBoost, another powerful classification algorithm.
- Chose the better-performing model based on metrics like accuracy and AUC-ROC.

Methods Attempted That Didn't Work

- **Initial Data Preprocessing:** Dropping rows with any NaN values led to a significant reduction in the dataset size, which might not be ideal for performance.
- **Model Training with All Features:** Including all features without considering their importance led to longer training times and potential overfitting.

Ideas for Future Work

- **Cross-Validation:** Perform cross-validation to ensure the model's robustness across different subsets of the data.
- **Hyperparameter Tuning:** Experiment with different hyperparameters to further improve model performance.
- **Feature Engineering:** Explore additional features or transformations that might enhance model performance, such as interaction terms or polynomial features.
- **Model Explainability:** Use SHAP or LIME to understand the model's decision process better and provide more transparency.
- **Pertinent Visualizations:** Visualizations such as feature importance plots and confusion matrices can be added to the script to better understand the model's performance and decision-making process

Conclusion

The RandomForest model outperformed XGBoost in this scenario, with higher accuracy, AUC-ROC, and better performance in the confusion matrix. It was chosen as the final model due to its reliability and fewer errors in predicting fraudulent transactions.