**Credit Card Fraud Detection**

**Table of Contents**

**1. Setup Environment**

**2. Data Overview**

**3. Optimize Memory Used by Data**

**4. Basic Data Stats**

**5. Data Preprocessing for EDA**

**6. Exploratory Data Analysis**

**7. Statistical Significance test**

**8. Dimensionality Reduction - PCA**

**9. Feature Encoding**

**10. Data Preprocessing for Model Building**

**11. Model Building**

**12. XGBoost Classifier**

**13. Evaluation Metrics**

**14. Capture Rates and Calibration Curve**

**15. LightGBM**

**16. Random Forest Classifier**

**17. Handling Class Imbalance**

**18. Cost Sensitive Learning with Class weights**

**19. Model Calibration**

**20. Model Tuning**

**21. Feature Importance**

**22. Partial Dependence and Individual Conditional Expectations ICE**

**23. SHAP Values**

**24. Final Words**

**Problem Statement**

The goal of this project is to predict the probability of an online credit card transaction being fraudulent. This is a classic example of a binary classification problem where the target variable is `isFraud`.

**Explanation of Steps**

**1. Setup Environment**

**What:** This section imports all the necessary libraries for data manipulation pandas, numpy, data visualization seaborn, matplotlib, machine learning scikit-learn, xgboost, lightgbm, and other utilities.

**Why:** To ensure all required tools are available for the analysis and modeling process.

**2. Data Overview**

**What:** The transaction and identity datasets are loaded into pandas DataFrames. A sample of each dataset is displayed.

**Why:** To get a first look at the data, understand its structure, and identify the features available in each dataset.

**3. Optimize Memory Used by Data**

**What:** A function `reduce\_mem\_usage` is defined and applied to both dataframes. This function iterates through numeric columns and downcasts their data types to the smallest possible type that can hold the data without loss of information.

**Why:** To reduce the memory footprint of the dataframes, which can significantly speed up subsequent computations, especially with large datasets.

**4. Basic Data Stats**

**What:** This section examines the dimensions shape of the dataframes, checks how many transactions have corresponding identity information, and uses `pandas-summary` to generate detailed statistics for the numeric columns.

**Why:** To understand the scale of the data, the extent of missing values, and the distribution of each feature. This information is crucial for data cleaning and feature engineering.

**5. Data Preprocessing for EDA**

**What:** The transaction and identity dataframes are merged. New columns are created to flag missing values. Columns with a high percentage of missing values over 90% and columns with no variance only one unique value are identified and dropped. Date features weekday, hour, day are extracted from the `TransactionDT` column.

**Why:** To create a unified and cleaner dataset for exploratory data analysis. Feature engineering and cleaning at this stage help in uncovering meaningful patterns.

**6. Exploratory Data Analysis**

**What:** This section visualizes the distribution of the target variable `isFraud`, transaction amounts, and various categorical features like `ProductCD`, `card4`, `card6`, `P\_emaildomain`, and `R\_emaildomain`. It also explores the relationship between these features and the target variable.

**Why:** To gain insights into the data, understand the characteristics of fraudulent transactions, and identify potentially predictive features.

**7. Statistical Significance test**

**What:** Chi-square tests are performed on categorical variables and ANOVA F-tests are performed on numerical variables to determine their statistical significance with respect to the target variable `isFraud`.

**Why:** To statistically validate the relationships observed during EDA and to select features that are most likely to be predictive of fraud.

**8. Dimensionality Reduction - PCA**

**What:** Principal Component Analysis PCA is applied to the 'V' columns V1-V339. These columns are first imputed and scaled, and then PCA is used to reduce them to a smaller set of 30 principal components.

**Why:** To reduce the dimensionality of the feature space, which can help in reducing model complexity, preventing overfitting, and improving model training time.

**9. Feature Encoding**

**What:** Categorical features with a large number of unique values are encoded using frequency encoding. The remaining categorical features are encoded using label encoding.

**Why:** To convert categorical features into a numerical format that can be used by machine learning models.

**10. Data Preprocessing for Model Building**

**What:** The dataframe is further cleaned by dropping unnecessary columns like `TransactionID` and `Date`. The data is then split into features X and the target variable y, and further divided into training and testing sets.

**Why:** To prepare the data for the final model building phase.

**11. Model Building**

**What:** This section introduces the model building process.

Why: To set the stage for training and evaluating machine learning models.

**12. XGBoost Classifier**

**What:** An XGBoost classifier is trained on the training data. Predictions are made on the test data.

**Why:** XGBoost is a powerful and popular gradient boosting algorithm known for its high performance in classification tasks.

**13. Evaluation Metrics**

**What:** A function `compute\_evaluation\_metric` is defined to calculate and display various evaluation metrics like accuracy, AUC score, confusion matrix, classification report, and ROC/PR curves. This function is then used to evaluate the XGBoost model.

**Why:** To systematically assess the performance of the model and understand its strengths and weaknesses.

**14. Capture Rates and Calibration Curve**

**What:** The predicted probabilities are binned, and the capture rate of fraudulent transactions in each bin is analyzed. A calibration curve is also plotted.

**Why:** To assess how well the model's predicted probabilities are calibrated. A well-calibrated model's probabilities can be directly interpreted as a confidence level.

**15. LightGBM**

**What:** A LightGBM classifier is trained and evaluated using the same process as the XGBoost model.

**Why:** LightGBM is another efficient gradient boosting framework that is often faster than XGBoost and can provide comparable or better performance.

**16. Random Forest Classifier**

**What:** A Random Forest classifier is trained and evaluated.

**Why:** To explore another powerful ensemble method and compare its performance with the gradient boosting models.

**17. Handling Class Imbalance**

**What**: The `RandomOverSampler` from the `imblearn` library is used to oversample the minority class fraudulent transactions in the training data. A LightGBM model is then trained on this balanced dataset.

**Why**: To address the class imbalance problem, which can cause models to be biased towards the majority class. Oversampling helps the model learn the patterns in the minority class more effectively.

**18. Cost Sensitive Learning with Class weights**

**What**: A LightGBM model is trained with the `class\_weight='balanced'` parameter.

**Why**: This is another technique to handle class imbalance, where the model assigns a higher penalty to misclassifying the minority class.

**19. Model Calibration**

**What**: The `CalibratedClassifierCV` is used to calibrate the probabilities of the LightGBM model.

**Why**: To improve the reliability of the predicted probabilities, making them more aligned with the true likelihood of an event.

**20. Model Tuning**

**What**: `GridSearchCV` is used to perform hyperparameter tuning on the LightGBM classifier to find the optimal combination of parameters.

**Why**: To improve the model's performance by finding the hyperparameters that result in the best generalization to unseen data.

**21. Feature Importance**

**What**: The feature importances from the tuned LightGBM model are extracted and visualized.

**Why**: To understand which features are most influential in the model's predictions, which can provide valuable insights for business stakeholders.

**22. Partial Dependence and Individual Conditional Expectations ICE**

**What**: Partial Dependence Plots PDP and Individual Conditional Expectation ICE plots are generated for some of the important features.

**Why**: To visualize the marginal effect of a feature on the predicted outcome of a machine learning model, providing a deeper understanding of the model's behavior.

**23. SHAP Values**

**What**: SHAP SHapley Additive exPlanations values are calculated and visualized to explain individual predictions.

**Why**: To provide a detailed explanation of how each feature contributes to a specific prediction, making the model more interpretable.

**24. Final Words**

**What**: This section provides a summary of the entire process and concluding remarks.

**Why**: To wrap up the project and highlight the key takeaways.

**Results and Conclusion**

The project successfully built and evaluated several machine learning models to detect credit card fraud. The LightGBM classifier, after hyperparameter tuning and calibration, proved to be the most effective model. The final model achieved a good balance between precision and recall, and the feature importance analysis provided valuable insights into the key drivers of fraudulent transactions.

Was the problem statement solved?

Yes, the problem statement was solved. We successfully developed a machine learning model that can predict the probability of a credit card transaction being fraudulent with a high degree of accuracy and interpretability. The model can be used to flag suspicious transactions for further investigation, helping to mitigate financial losses due to fraud.