Credit Card Fraud Detection - Explanation in Points

1. Importing Libraries

- Various libraries are imported to perform data manipulation, visualization, preprocessing, model training, and evaluation.
- **Key Libraries Used:**
    - pandas, numpy: For data handling and numerical computations.
    - matplotlib.pyplot, seaborn: For visualization.
    - sklearn.model_selection: For splitting data and hyperparameter tuning.
    - sklearn.preprocessing: For scaling numerical features.
    - sklearn.ensemble, sklearn.linear_model: Machine learning models (Random Forest, Logistic Regression).
    - sklearn.metrics: For performance evaluation.
    - imblearn.over_sampling: To handle class imbalance using SMOTE.

2. Loading the Dataset

- The dataset is loaded using pd.read_csv().
- Basic dataset information is displayed:
    - shape of the dataset.
    - First 5 rows using data.head().
    - General dataset info (data.info()).
    - Class distribution (data['Class'].value_counts()) to check fraud vs. non-fraud transactions.

3. Data Preprocessing

- **Checking for Missing Values:**
    - data.isnull().sum() is used to detect missing values in the dataset.
- **Feature Scaling:**
    - The Amount feature is normalized using StandardScaler to standardize its values.
- **Dropping Unnecessary Features:**
    - The Time feature is removed as it is not relevant for fraud detection.
- **Separating Features & Target Variable:**
    - X contains all features except the target column (Class).
    - y contains the target labels (Class), which indicate fraud (1) or non-fraud (0).

4. Handling Class Imbalance

- The dataset is highly imbalanced (fewer fraud cases than non-fraud).
- **Steps Taken to Address Imbalance:**
    - **Handling Missing Values:**
        - SimpleImputer(strategy='mean') replaces NaN values with the mean.
    - **Applying SMOTE (Synthetic Minority Over-sampling Technique):**
        - Generates synthetic samples for the minority class (fraud cases) to balance the dataset.

5. Splitting the Dataset

- The resampled data (X_resampled, y_resampled) is split into training and testing sets.
- **Train-Test Split:**
    - train_test_split(X_resampled, y_resampled, test_size=0.3, random_state=42) is used to:
        - Allocate **70% of data for training**.
        - Allocate **30% for testing**.

Conclusion

- This project aims to detect fraudulent transactions in credit card data.
- It involves data preprocessing, handling imbalance using SMOTE, and splitting data for training/testing.
- Further steps would involve training models (e.g., Logistic Regression, Random Forest) and evaluating their performance.