## **Phase-2**

## **Data Analytics**

**Student Name:** Joshika.S  
**Register Number:** 613023243021  
**Institution:** Vivekanandha College Of Technology For women  
**Department:** B.Tech-Artificial Intelligence & Data Science  
**Date of Submission:** 04-05-2025  
**GitHub Repository Link:**

### **1.Problem Statement**

The rapid expansion of digital transactions has led to an increased risk of fraudulent activities, especially in credit card payments. Fraudulent transactions can cause significant financial losses for both consumers and financial institutions. Traditional fraud detection methods, such as rule-based systems, are often insufficient to handle the complexity and scale of modern financial ecosystems. This problem can be addressed through AI-powered credit card fraud detection, which leverages advanced data analytics to identify anomalous patterns, predict potential fraud, and prevent unauthorized transactions in real time.

### **2. Project Objectives**

### **1. Transaction Data Analysis:**

\* Analyze historical credit card transaction data, including transaction amount, merchant information, geographical location, time of purchase, user behavior, and more.

\* Identify patterns and anomalies that indicate fraudulent activity.

**2. Model Development:**

\* Develop a machine learning (ML) model (e.g., decision trees, neural networks, random forests, etc.) to classify transactions as legitimate or potentially fraudulent.

\* Use supervised learning with labeled data (fraudulent vs. legitimate transactions) to train the model and improve its predictive accuracy.

\* Apply unsupervised learning (e.g., clustering techniques) to detect unknown fraud patterns without labeled data.

**3. Feature Engineering and Selection**:

\* Identify key features that contribute to fraud detection (e.g., transaction frequency, geographic distance between transactions, spending behavior).

\* Develop methods to extract new features, including transaction velocity, merchant type, and user behavior analysis (e.g., spending patterns).

**4. Real-Time Detection:**

\* Design a system that operates in real-time, capable of processing transactions as they occur.

\* Implement strategies for immediate detection and prevention of fraud (e.g., transaction blocks, customer alerts, or additional authentication5. Reducing False Positives:

\* Minimize false positives by improving the precision of fraud detection models. False positives (legitimate transactions flagged as fraud) lead to user frustration and potential lost business.

**5. Continuous Learning:**

\* Implement a feedback loop where the model is continuously updated with new data to adapt to evolving fraud tactics.

. \*Gained practical experience with **data visualization tools** like heatmaps, boxplots, and correlation matrices to support decision-making.

\*Enhanced our use of **descriptive and inferential statistics** to validate assumptions and test hypotheses.

\*Improved data wrangling skills, including grouping, filtering, and aggregating data for deeper exploration.

**3.Flowchart for the project work flow:**

### **4. Data Description**

**Dataset Overview**

* **Name:** Student Performance Data Set
* **Source:** [UCI Machine Learning Repository](https://archive.ics.uci.edu/ml/datasets/student+performance)
* **Type:** Structured, tabular data
* **Nature:** Static dataset (no temporal updates or streaming)

**Dataset Composition**

* **Total Records:** 395 students
* **Total Features:** 33 variables
  + Mix of **numerical** (e.g., grades, absences) and **categorical** (e.g., school, internet access) data
* **Target Variable:** G3 — Final grade (numerical, range 0–20)

**Main Attribute Categories**

1. **Demographic Information**
   * Age, gender, address type (urban/rural)
   * Parental education and job
2. **Academic Performance**
   * First (G1) and second (G2) period grades
   * Study time, course failure history
3. **Behavioral Patterns**
   * Daily and weekend alcohol consumption
   * Number of school absences
   * Access to internet, participation in extracurriculars

### **5.Data Preprocessing**

1. **Data Integrity Check**

* Verified that the dataset contained **no missing or null values**, ensuring consistency across all records.
* Confirmed the **absence of duplicate rows**, maintaining data quality and preventing biased learning.

1. **Feature Selection & Cleaning**

* Removed **low-variance features** that provided little to no predictive value (e.g., features like school if all values were the same).
* This step reduced noise and improved model efficiency.

1. **Categorical Encoding**

* Transformed categorical variables using **One-Hot Encoding**, converting text-based features into binary indicator variables.
* Ensured all encoded features were compatible with machine learning algorithms.

1. **Feature Scaling**

* Applied **StandardScaler** to numerical features to standardize them (mean = 0, standard deviation = 1).
* This helped improve convergence in algorithms sensitive to feature scales, such as gradient descent-based models.

1. **Outlier Detection**

* Used **boxplots** and **Z-score analysis** to identify potential outliers.

Investigated extreme outliers manually to assess whether they were valid or errors, helping maintain model robustness.

### **6. Exploratory Data Analysis (EDA)**

**1. Understand the Dataset**

Most fraud datasets (like the famous Kaggle one) include:

* **Transaction features**: anonymized or raw transaction details (e.g., Amount, Time, Location, Merchant, etc.)
* **Target variable**: A binary flag, often named is\_fraud or Class (0 = normal, 1 = fraud).

**2. Data Cleaning**

* Check for **missing values**
* Handle **duplicate transactions**
* Normalize timestamp if needed (e.g., convert UNIX to human-readable)
* Investigate outliers in Amount, Time, etc.

**3. Target Variable Distribution**

import seaborn as sns

sns.countplot(data=df, x='Class')

This step reveals **class imbalance** – often <1% of data is fraud.**4. Feature Distributions**

Use histograms or KDE plots to compare fraudulent vs. non-fraudulent transactions:

* **Transaction Amount**: Are frauds typically small or large?
* **Transaction Time**: Do frauds spike at odd hours?
* **Location / Merchant / Type**: Any patterns?

**5. Correlation Analysis**

Use heatmaps or correlation matrices:

import seaborn as sns

sns.heatmap(df.corr(), annot=True)

**6. Dimensionality Reduction (Optional)**

For high-dimensional data (e.g., PCA-anonymized features), use:

* **PCA or t-SNE** to visualize fraud vs. non-fraud clusters

from sklearn.decomposition import PCA

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(df.drop('Class', axis=1))

**7. Behavioral Patterns**

Analyze patterns like:

* **Multiple small transactions in short time (test frauds)**
* **Unusual geolocation jumps (impossible travel time)**

**8. Aggregations by User or Card ID**

If available:Total amount per user/day

* Number of distinct locations
* Sudden behavior shifts

**7.Feature Engineering**

**1. Interaction Features**

**What you did:**

total\_alcohol = Dalc + Walc  
This is a good example of a **domain-informed interaction feature**. In a fraud context, interaction features might look like:

* transaction\_speed = transaction\_amount / time\_since\_last\_txn
* avg\_amount\_per\_merchant = total\_spent / number\_of\_merchants

**2. Binary Features from Ordinals or Categories**

higher\_edu = (yes/no) from parents' education  
This is a clean way to reduce dimensionality when detailed levels aren’t necessary.

**3. Removing Correlated/Redundant Features**

Reduces **multicollinearity** and noise, especially helpful for linear models and PCA.

**4. Label Encoding for Binary Features**

You encoded features like internet, nursery as 0/1.

Be cautious not to use **label encoding** for nominal categories with >2 values (use One-Hot or embeddings instead).

**5. Scaling Numeric Features**

Used StandardScaler – this is crucial for:

* Gradient descent-based models (e.g., Logistic Regression, Neural Networks)
* Distance-based models (e.g., KNN, SVM)

**8.Model Building**

**1. Interaction Features**

* **Transaction speed** = transaction amount ÷ time since last transaction
* **Spending pattern ratio** = amount spent at high-risk merchants ÷ total spend
* **Average amount per merchant** = total spent ÷ number of distinct merchants

**2. Binary Features from Ordinal or Categorical Variables**

* **Is weekend transaction**: yes if transaction occurred on Saturday or Sunday
* **Is night transaction**: yes if time is between 10 p.m. and 6 a.m.
* **Is new device**: yes if device has not been seen before for that user

**3. Removing Highly Correlated or Redundant Features**

Redundant features add noise and can bias models.

Examples:

* Drop one of two time features that track the same thing
* Remove total\_spent if it can be directly inferred from avg\_amount × txn\_count

**4. Encoding Binary & Categorical Features**

Transform non-numeric data into numerical values:

* For **binary features** (e.g., "has internet access"), assign values like 0 and 1
* For **multi-class categorical variables** (e.g., merchant type, region), use dummy variables or learned embeddings

**5. Scaling Numerical Features**

Scale features to a standard range or distribution:

* **Standard scaling**: centers data with mean = 0 and std = 1 (good for most models)
* **Robust scaling**: resists the impact of outliers (useful for skewed transaction amounts)
* **Log transformation**: reduces skew in long-tailed distributions

**6. Temporal Features**

Extract features from time-related data to capture fraud patterns:

* **Hour of day**: frauds may peak at night or early morning
* **Day of week**: weekend behavior can differ from weekdays
* **Time since last transaction**: short gaps may indicate account takeover or fraud bursts

### **9.Visualization of results & model Insights**

**1. Feature Importance**

* **What was done:**  
  A **bar plot** was used to visualize feature importance scores from a trained **Random Forest model**.
* **Key insight:**  
  Variables **G1** (first period grade) and **G2** (second period grade) were the most predictive features for the final outcome (G3).  
  Other influential features included **study time** and **failures**, indicating academic engagement and past difficulties also affect performance.

**2. Model Comparison**

* **What was done:**  
  Plotted **Mean Absolute Error (MAE)**, **Root Mean Squared Error (RMSE)**, and **R² score** for both **Linear Regression** and **Random Forest** models.
* **Key insight:**  
  **Random Forest** significantly outperformed **Linear Regression**, particularly in terms of **RMSE**, suggesting better handling of non-linearity and interactions.

**3. Residual Plots**

* **What was done:**  
  Created residual plots by comparing predicted vs. actual grades to identify systematic prediction errors or bias.
* **Key insight:**  
  Residuals were evenly distributed around zero, indicating **no major bias** or heteroscedasticity. This confirms the model’s stability across different ranges of the target variable.

**4. User Testing via Gradio**

* **What was done:**  
  Deployed the model in an **interactive interface using Gradio**, allowing users to input features like G1, G2, study time, etc., and get predicted final grades.
* **Key insight:**  
  Enabled real-time, **user-friendly validation** of the model with varied test cases and edge scenarios.

### **7. Tools and Technologies Used**

**1.Relational Databases (SQL) MySQL / PostgreSQL:**

Common relational databases used to store transactional data such as user details, transaction records, merchant information, etc.

**NoSQL Databases:**

MongoDB / Cassandra: Used for storing large amounts of unstructured data, like transaction logs and user behavior data, which can scale horizontally.

**Cloud Storage:**

Amazon S3 / Google Cloud Storage: Used to store large datasets, logs, and files needed for training and monitoring fraud detection models.

**2. Data Preprocessing and Exploratory Data Analysis (EDA)**

**Python:**

Pandas: For data manipulation and preprocessing. Helps in cleaning, transforming, and analyzing data.

**NumPy:**

Used for numerical computing and handling large multi-dimensional arrays and matrices.

Matplotlib /Seaborn: For data visualization to explore the distribution of features, detect anomalies, and analyze relationships between features.

**3. Machine Learning Algorithms and Model Development**

**Scikit-learn:**

Logistic Regression: For binary classification (fraud vs legitimate).

Random Forest: Ensemble model that provides a robust and interpretable solution for fraud detection.

**8.Team Members and Contributions**

| **Team Member** | **Responsibilities** |
| --- | --- |
| **Indhumathi.E** | **- Data Cleaning:** verified data integrity, handled duplicates, standardized formatting**.** |
| **Joshika.S** | **- Performed Exploratory Data Analysis (EDA):** visualized trends, distributions, and correlations. |
| **Janani.S** | **- Handled Feature Engineering:** created derived features, encoded variables, and scaled data. |
| **Jaya.S** | **- Responsible for Model Development:** trained and evaluated Linear Regression and Random Forest models. |
| **Swetha.C** | **- In charge of Documentation and Reporting**: compiled results, wrote final report, and designed presentation slides**.** |