**PHASE-3 SUBMISSION TEAM-4**

Data Pre-processing For Credit Card Fraud Detection Using Data Science

**Importing needed packages:**

>>>import pandas as pd

>>>import numpy as np

>>>import seaborn as sns

>>>import matplotlib.pyplot as plt

>>>from sklearn.model\_selection import train\_test\_split

>>>from sklearn.preprocessing import StandardScaler

>>>from sklearn.metrics import r2\_score, mean\_absolute\_error,mean\_squared\_error

>>>from sklearn.linear\_model import LinearRegression

>>>from sklearn.linear\_model import Lasso

>>>from sklearn.ensemble import RandomForestRegressor

>>>from sklearn.svm import SVR

>>>import xgboost as xg

**Loading the dataset:**

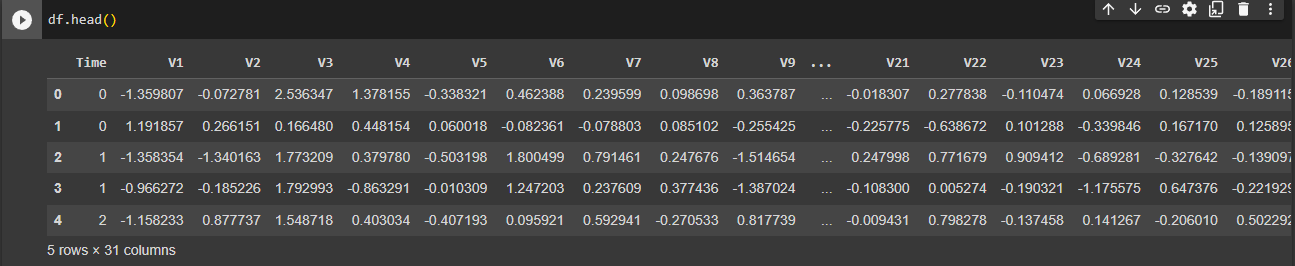
>>>df=pd.read\_csv(‘/content/creditcard.csv’)

**Data preprocessing techniques for Credit card fraud detection:**

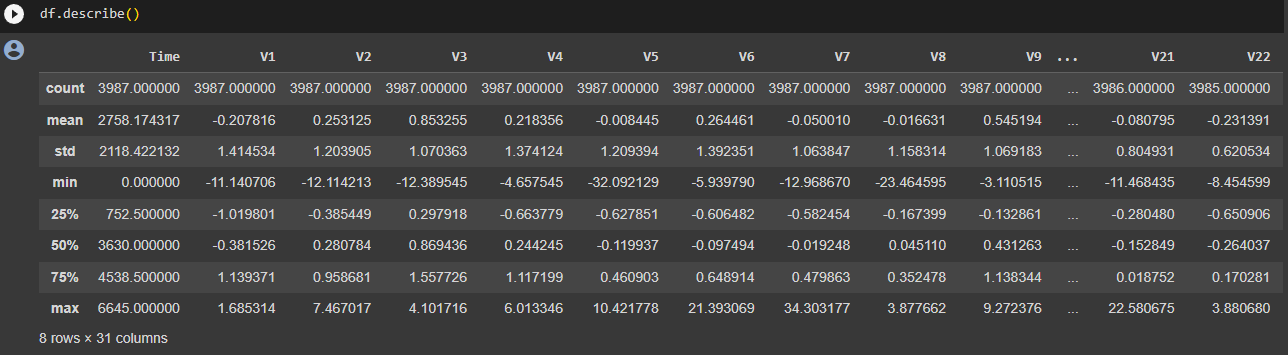
**Data Collection and Understanding:**

* Gather a comprehensive dataset containing both genuine and fraudulent transactions.

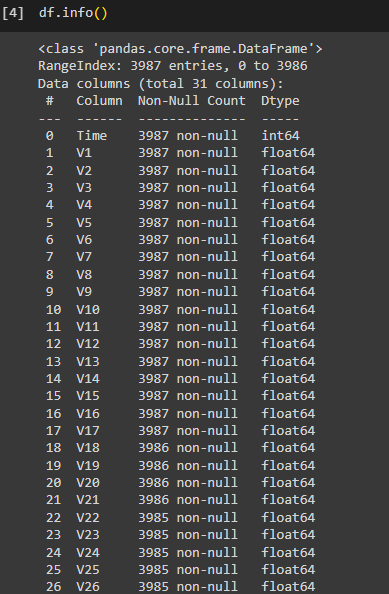
>>>print(df.head())



>>>print(df.describe())



>>>print(df.info())



>>>print(df.tail())



**Feature Scaling:**

* Standardize or normalize numerical features to have a similar scale. This is important for algorithms that are sensitive to the scale of input features.

>>>from sklearn.preprocessing import StandardScaler

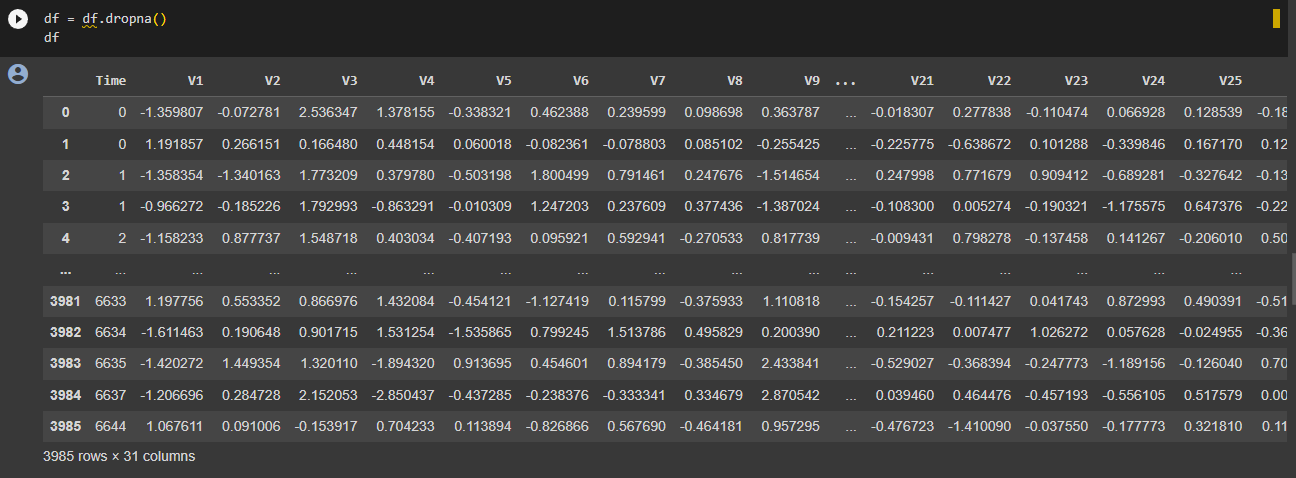
>>>scaler = StandardScaler()

>>>df['amount'] = scaler.fit\_transform(df[['amount']])

**Handling missing values:**

* Deal with missing values by either removing rows with missing values or imputing missing values using methods such as mean imputation.

>>>df = df.dropna()



**Splitting the dataset:**

* Divide the preprocessed data into training and testing sets to evaluate the model's performance.

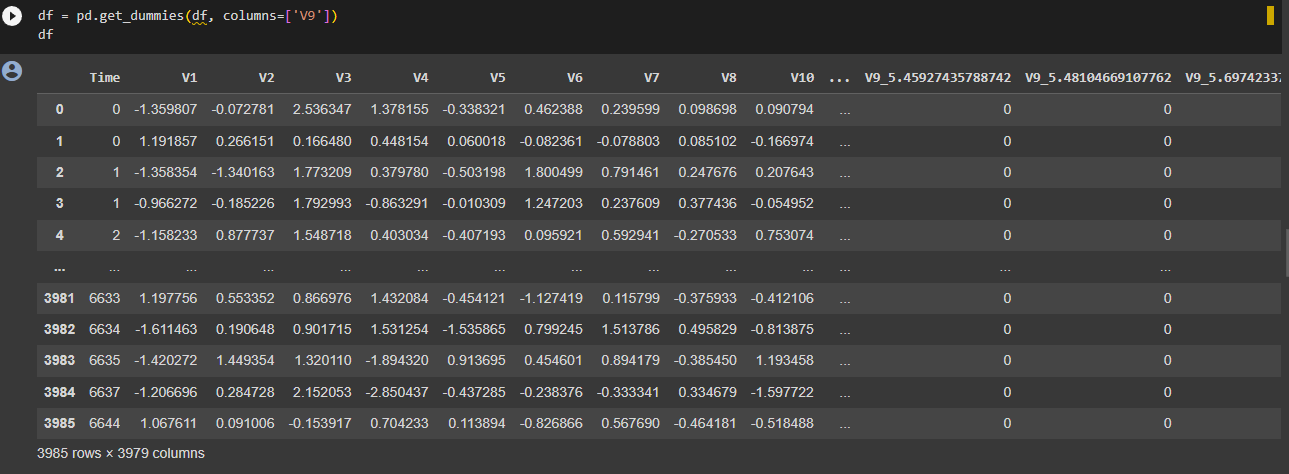
>>>from sklearn.model\_selection import train\_test\_split

>>>X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

**Encoding categorical variables:**

* Convert categorical variables into a format suitable for machine learning algorithms, such as one-hot encoding.

>>>df = pd.get\_dummies(df, columns=['V9'])

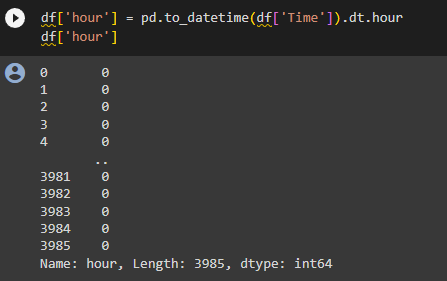


**Time based features:**

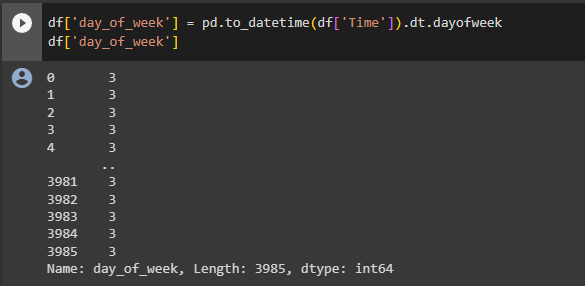
* If your dataset includes timestamps, consider extracting features such as the hour of the day, day of the week, or time since the last transaction. Fraud patterns may vary based on time.

>>>df['hour'] = pd.to\_datetime(df['Time']).dt.hour

>>>df[‘hour’]



>>>df['day\_of\_week'] = pd.to\_datetime(df['timestamp']).dt.dayofweek



**Dimensionality reduction:**

* If your dataset has a large number of features, consider applying dimensionality reduction techniques like Principal Component Analysis (PCA) to reduce the number of features while retaining important information.

>>>from sklearn.decomposition import PCA

>>>pca = PCA(n\_components=2)

>>>reduced\_features = pca.fit\_transform(df.drop(['timestamp', 'y'], axis=1))