**Credit Card Fraud Detection**

**PHASE3: DEVELOPMENT PART 1**

**TOPIC : Building your Project by Loading and Preprocessing The Dataset.**



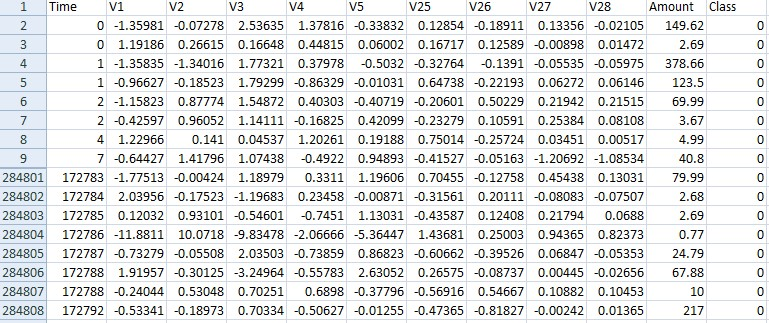
**EXPLANATION OF DATASET:**

* Credit card fraud detection is the collective term for the policies, tools, methodologies, and practices that credit card companies and financial institutions take to combat identity fraud and stop fraudulent transactions.
* This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.
* It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data.
* Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'.
* Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset.
* The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning.
* Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.
* The dataset will contain various features or attributes related to each transaction.
* Transaction amount
* Transaction date and time
* Merchant information
* Customer account details
* Location information
* Card type
* Transaction category

**GIVEN DATASET:**

the given dataset for credit card fraud detection :

<https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>

****

**Import Libraries :**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from imblearn.over\_sampling import SMOTE

from collections import Counter

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import confusion\_matrix, classification\_report

from tabulate import tabulate

# load dataset:

data=pd.read\_csv('/kaggle/input/creditcardfraud/creditcard.csv')

data.head()

# OUTPUT:

| Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V21 | V22 | V23 | V24 | V25 | V26 | V27 | V28 | Amount | Class |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0.0 | -1.359807 | -0.072781 | 2.536347 | 1.378155 | -0.338321 | 0.462388 | 0.239599 | 0.098698 | 0.363787 | ... | -0.018307 | 0.277838 | -0.110474 | 0.066928 | 0.128539 | -0.189115 | 0.133558 | -0.021053 | 149.62 | 0 |
| 1 | 0.0 | 1.191857 | 0.266151 | 0.166480 | 0.448154 | 0.060018 | -0.082361 | -0.078803 | 0.085102 | -0.255425 | ... | -0.225775 | -0.638672 | 0.101288 | -0.339846 | 0.167170 | 0.125895 | -0.008983 | 0.014724 | 2.69 | 0 |
| 2 | 1.0 | -1.358354 | -1.340163 | 1.773209 | 0.379780 | -0.503198 | 1.800499 | 0.791461 | 0.247676 | -1.514654 | ... | 0.247998 | 0.771679 | 0.909412 | -0.689281 | -0.327642 | -0.139097 | -0.055353 | -0.059752 | 378.66 | 0 |
| 3 | 1.0 | -0.966272 | -0.185226 | 1.792993 | -0.863291 | -0.010309 | 1.247203 | 0.237609 | 0.377436 | -1.387024 | ... | -0.108300 | 0.005274 | -0.190321 | -1.175575 | 0.647376 | -0.221929 | 0.062723 | 0.061458 | 123.50 | 0 |
| 4 | 2.0 | -1.158233 | 0.877737 | 1.548718 | 0.403034 | -0.407193 | 0.095921 | 0.592941 | -0.270533 | 0.817739 | ... | -0.009431 | 0.798278 | -0.137458 | 0.141267 | -0.206010 | 0.502292 | 0.219422 | 0.215153 | 69.99 | 0 |

5 rows × 31 columns

# ****DATA PREPROCESSING:****

# Data preprocessing is the concept of changing the raw data into a clean data set. The dataset is preprocessed in order to check missing values, noisy data, and other inconsistencies before executing it to the algorithm.

data.describe()

# OUTPUT:

| Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V21 | V22 | V23 | V24 | V25 | V26 | V27 | V28 | Amount | Class |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| count | 284807.000000 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | ... | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 284807.000000 | 284807.000000 |
| mean | 94813.859575 | 1.168375e-15 | 3.416908e-16 | -1.379537e-15 | 2.074095e-15 | 9.604066e-16 | 1.487313e-15 | -5.556467e-16 | 1.213481e-16 | -2.406331e-15 | ... | 1.654067e-16 | -3.568593e-16 | 2.578648e-16 | 4.473266e-15 | 5.340915e-16 | 1.683437e-15 | -3.660091e-16 | -1.227390e-16 | 88.349619 | 0.001727 |
| std | 47488.145955 | 1.958696e+00 | 1.651309e+00 | 1.516255e+00 | 1.415869e+00 | 1.380247e+00 | 1.332271e+00 | 1.237094e+00 | 1.194353e+00 | 1.098632e+00 | ... | 7.345240e-01 | 7.257016e-01 | 6.244603e-01 | 6.056471e-01 | 5.212781e-01 | 4.822270e-01 | 4.036325e-01 | 3.300833e-01 | 250.120109 | 0.041527 |
| min | 0.000000 | -5.640751e+01 | -7.271573e+01 | -4.832559e+01 | -5.683171e+00 | -1.137433e+02 | -2.616051e+01 | -4.355724e+01 | -7.321672e+01 | -1.343407e+01 | ... | -3.483038e+01 | -1.093314e+01 | -4.480774e+01 | -2.836627e+00 | -1.029540e+01 | -2.604551e+00 | -2.256568e+01 | -1.543008e+01 | 0.000000 | 0.000000 |
| 25% | 54201.500000 | -9.203734e-01 | -5.985499e-01 | -8.903648e-01 | -8.486401e-01 | -6.915971e-01 | -7.682956e-01 | -5.540759e-01 | -2.086297e-01 | -6.430976e-01 | ... | -2.283949e-01 | -5.423504e-01 | -1.618463e-01 | -3.545861e-01 | -3.171451e-01 | -3.269839e-01 | -7.083953e-02 | -5.295979e-02 | 5.600000 | 0.000000 |
| 50% | 84692.000000 | 1.810880e-02 | 6.548556e-02 | 1.798463e-01 | -1.984653e-02 | -5.433583e-02 | -2.741871e-01 | 4.010308e-02 | 2.235804e-02 | -5.142873e-02 | ... | -2.945017e-02 | 6.781943e-03 | -1.119293e-02 | 4.097606e-02 | 1.659350e-02 | -5.213911e-02 | 1.342146e-03 | 1.124383e-02 | 22.000000 | 0.000000 |
| 75% | 139320.500000 | 1.315642e+00 | 8.037239e-01 | 1.027196e+00 | 7.433413e-01 | 6.119264e-01 | 3.985649e-01 | 5.704361e-01 | 3.273459e-01 | 5.971390e-01 | ... | 1.863772e-01 | 5.285536e-01 | 1.476421e-01 | 4.395266e-01 | 3.507156e-01 | 2.409522e-01 | 9.104512e-02 | 7.827995e-02 | 77.165000 | 0.000000 |
| max | 172792.000000 | 2.454930e+00 | 2.205773e+01 | 9.382558e+00 | 1.687534e+01 | 3.480167e+01 | 7.330163e+01 | 1.205895e+02 | 2.000721e+01 | 1.559499e+01 | ... | 2.720284e+01 | 1.050309e+01 | 2.252841e+01 | 4.584549e+00 | 7.519589e+00 | 3.517346e+00 | 3.161220e+01 | 3.384781e+01 | 25691.160000 | 1.000000 |

8 rows × 31 columns

*# checking null values in dataset*

data.isnull().sum()

# OUTPUT:

Time 0

V1 0

V2 0

V3 0

V4 0

V5 0

V6 0

V7 0

V8 0

V9 0

V10 0

V11 0

V12 0

V13 0

V14 0

V15 0

V16 0

V17 0

V18 0

V19 0

V20 0

V21 0

V22 0

V23 0

V24 0

V25 0

V26 0

V27 0

V28 0

Amount 0

Class 0

dtype: int64

data.columns

# OUTPUT:

Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',

'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20',

'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount',

'Class'],

dtype='object')

data.shape

# OUTPUT:

(284807, 31)

*# cheking duplicates in data*

data.duplicated().value\_counts()

# OUTPUT:

False 283726

True 1081

Name: count, dtype: int64

*# droping the duplicates*

data=data.drop\_duplicates()

data.duplicated().any()

**OUTPUT:**

False

data.shape

**OUTPUT:**

(283726, 31)

data.Class.unique()

**OUTPUT:**

array([0, 1])

### data contains legtimate and fraud values:

[legitimate ,fraud]=data.Class.value\_counts()

print(legitimate)

print(fraud)

**OUTPUT:**

283253

473

fraud = data[data['Class']==0]

legitimate = data[data['Class']==1]

print(fraud.shape)

print(legitimate.shape)

# OUTPUT:

(283253, 31)

(473, 31)

# DATA ANALYSIS:

# Data Analysis is the technique of collecting, transforming, and organizing data to make future predictions and informed data-driven decisions. It also helps to find possible solutions for a business problem.

DATA VISUALIZATION:

fraud\_data = data[data['Class'] == 0]

legitimate\_data = data[data['Class'] == 1]

plt.figure(figsize=(8, 6))

plt.title("Transaction Amount - Fraud vs. Legitimate")

plt.xlabel("Amount ($)")

plt.ylabel("Frequency")

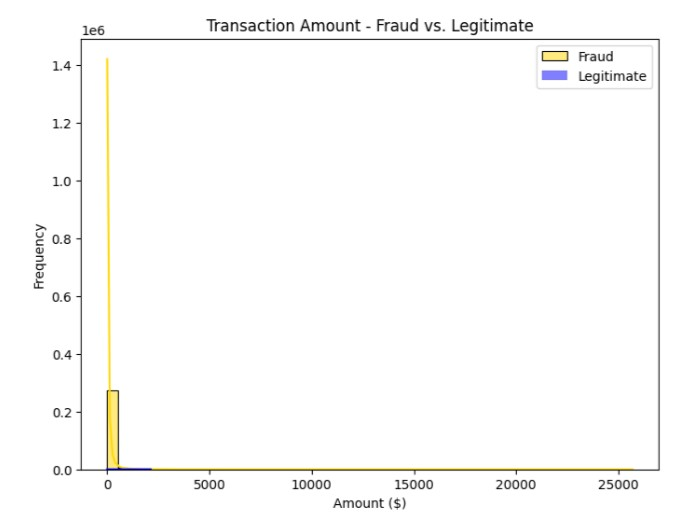
sns.histplot(fraud\_data['Amount'], bins=50, kde=True, color="gold", label="Fraud")

sns.histplot(legitimate\_data['Amount'], bins=50, kde=True, color="blue", label="Legitimate")

plt.legend()

plt.show()

**OUTPUT:**



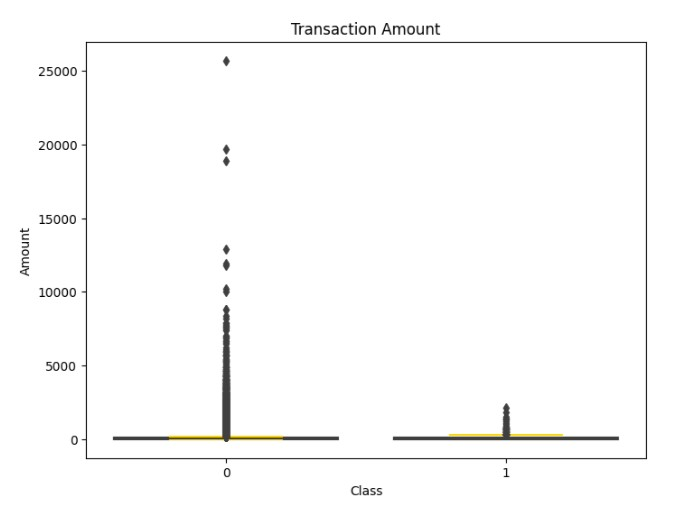
plt.figure(figsize=(8, 6))

plt.title("Transaction Amount")

sns.boxplot(x="Class", y="Amount", data=data, boxprops=dict(facecolor="blue"), whiskerprops=dict(color="gold"), capprops=dict(color="gold"))

plt.show()

**OUTPUT:**

****

corr\_imbalanced = credit\_card.corr()

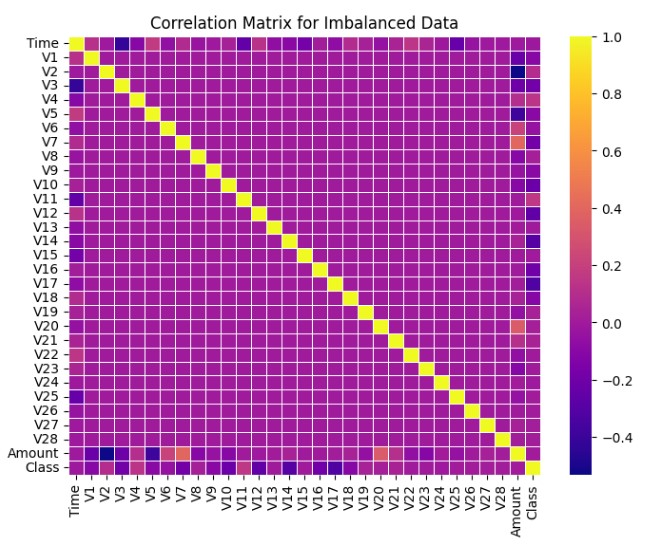
plt.figure(figsize=(8, 6))

sns.heatmap(corr\_imbalanced, annot=False, cmap="plasma", linewidths=0.5)

plt.title("Correlation Matrix for Imbalanced Data")

plt.show()

**OUTPUT:**

****

corr\_balanced = x\_smote.corr()

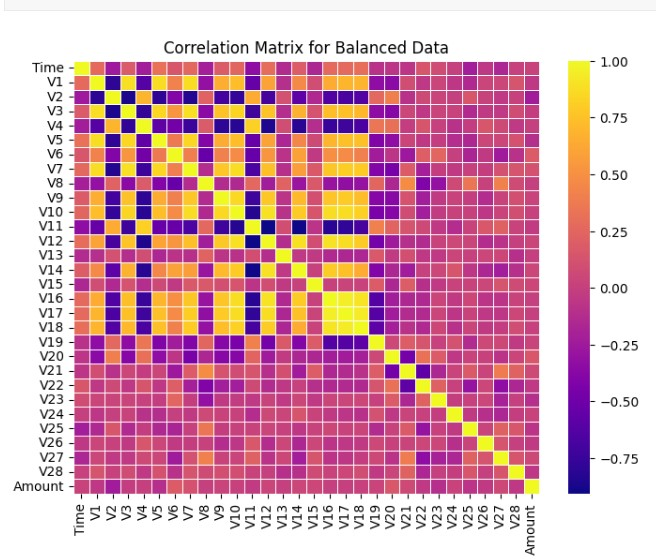
plt.figure(figsize=(8, 6))

sns.heatmap(corr\_balanced, annot=False, cmap="plasma", linewidths=0.5)

plt.title("Correlation Matrix for Balanced Data")

plt.show()

**OUTPUT:**

****

custom\_palette = sns.color\_palette(["blue", "gold"])

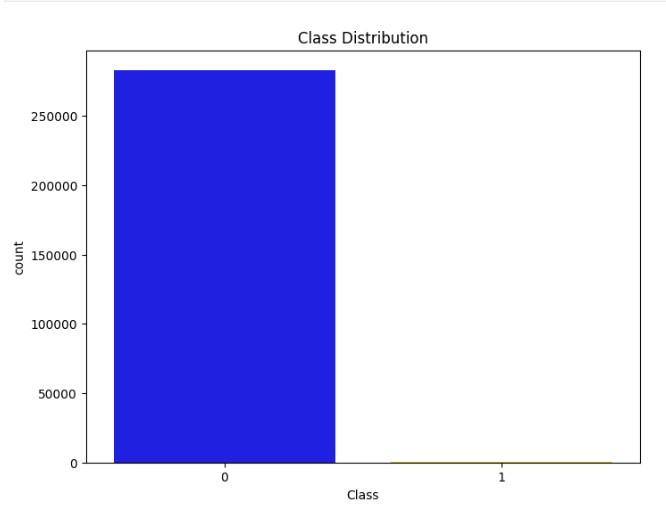
plt.figure(figsize=(8, 6))

sns.countplot(x="Class", data=data, palette=custom\_palette)

plt.title("Class Distribution")

plt.show()

**OUTPUT:**

****

custom\_palette = sns.color\_palette(["blue", "gold"])

plt.figure(figsize=(8, 6))

sns.countplot(x=y\_smote, palette=custom\_palette)

plt.title("Distribution of Classes After Resampling (SMOTE)")

plt.xlabel("Class (0: Fraud, 1: Legitimate)")

plt.ylabel("Count")

plt.show()

**OUTPUT:**

****