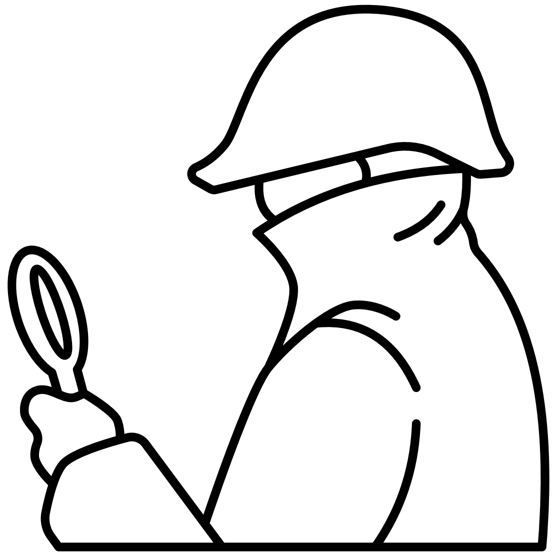
NATIONAL INSTITUTE OF TECHNOLOGY ANDHRA PRADESH

EEE DEPARTMENT

A MINOR PROJECT ON

# CREDIT CARD FRAUD DETECTION



Name -Suryansh Gangwar

Roll no -521248

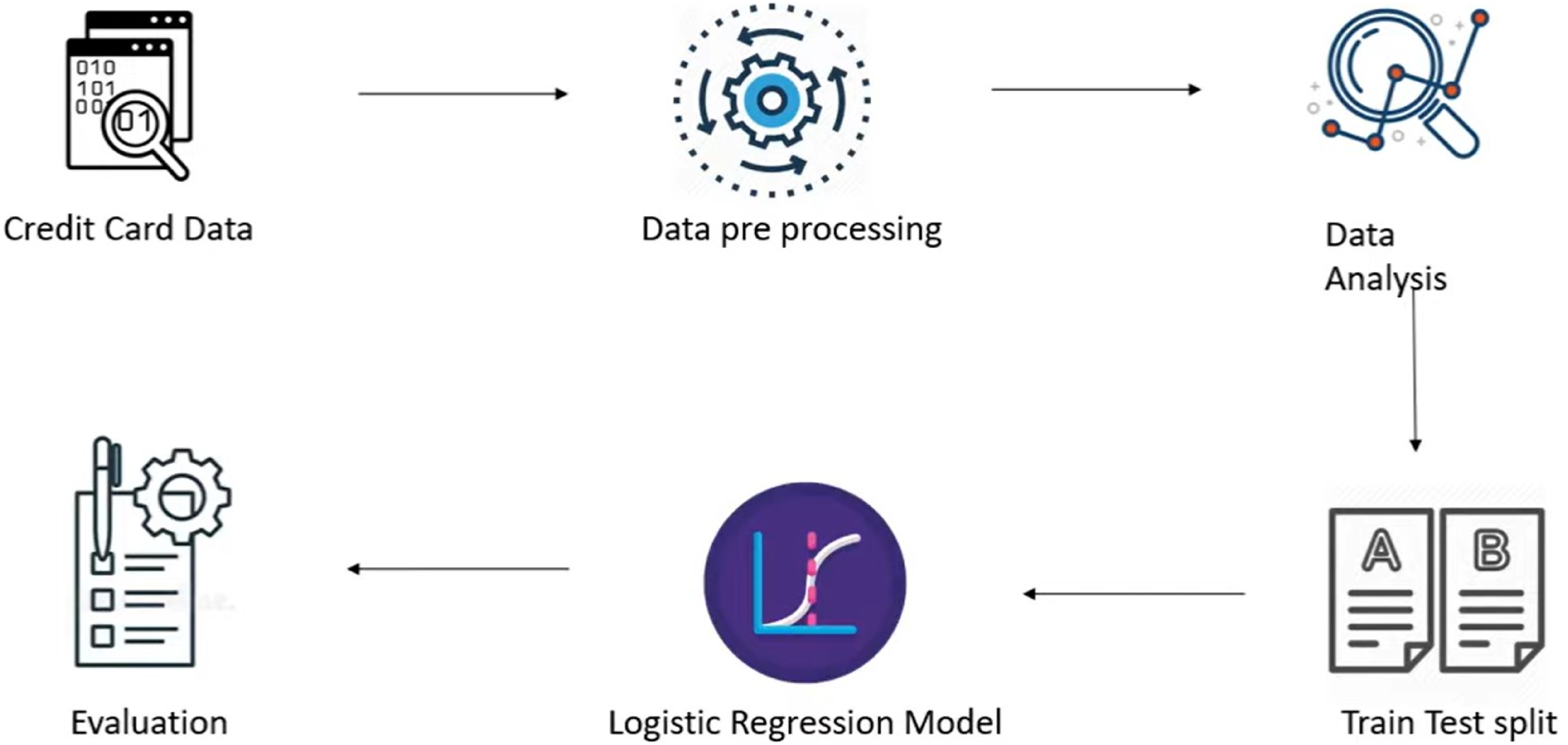
SEC -B

Btech 3yr EEE

# INTRODUCTION

Credit card fraud detection employs machine learning, specifically logistic regression, to analyze transaction data and identify fraudulent activities. Logistic regression, a statistical method, calculates the probability of fraud based on various transaction features like amount, location, and frequency. By training on historical data, the model learns patterns associated with fraud and can predict the likelihood of new transactions being fraudulent. This enables financial institutions to take proactive measures to prevent fraudulent transactions, enhancing security for cardholders. Through continuous learning and adaptation, machine learning algorithms enhance the accuracy and effectiveness of fraud detection systems over time.

# WORK FLOW


Credit Card Data
010
101
00
01
Data pre processing
Data
Analysis
A
B
Train Test split
Logistic Regression Model
Evaluation

# LOGISTIC REGRESSION

Logistic regression is a statistical technique for predicting the probability of binary outcomes.

In credit card fraud detection, it assesses transactional features to predict the likelihood of fraud.

The model is trained on historical data, learning the relationships between features and fraud occurrences.

It assigns probabilities to new transactions, helping classify them as fraudulent or not based on a predetermined threshold.

Evaluation metrics like accuracy, precision, and recall assess its performance in detecting fraudulent transactions.

Logistic regression enhances security measures by identifying and mitigating potential fraud risks in credit card transactions.

# IMPORTING THE DEPENDENCIES

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

These imports are essential for:

**numpy (as np)**: Numerical computations and array manipulation.
**pandas (as pd)**: Data manipulation using DataFrames.
**train_test_split**: Splitting data into training and testing sets.
**LogisticRegression**: Implementing logistic regression for classification.
**accuracy_score**: Evaluating the model's accuracy.

# CODE EXPLAINATION

```
# loading the dataset to a Pandas DataFrame
credit_card_data = pd.read_csv('/content/creditcard.csv')

# first 5 rows of the dataset
credit_card_data.head()

# dataset informations
credit_card_data.info()
```

This code snippet loads a dataset named 'creditcard.csv' into a Pandas DataFrame named 'credit_card_data'. The dataset likely contains credit card transaction data for analysis, often used in tasks such as fraud detection.

This code displays the first 5 rows of the 'credit_card_data' DataFrame, allowing quick inspection of the dataset's structure and content.

This code provides information about the 'credit_card_data' DataFrame, including the data types of each column, the number of non-null values, and memory usage. It helps understand the dataset's characteristics and identify any missing or inconsistent data.

# CODE EXPLAINATION

```
# separating the data for analysis
legit = credit_card_data[credit_card_data.Class == 0]
fraud = credit_card_data[credit_card_data.Class == 1]
```

This code separates the 'credit_card_data' DataFrame into two separate DataFrames: 'legit' containing non-fraudulent transactions (where 'Class' equals 0), and 'fraud' containing fraudulent transactions (where 'Class' equals 1). This separation facilitates separate analysis and modeling for each class of transactions.

```
credit_card_data.isnull().sum()
```

This code calculates the sum of missing values (NaN) for each column in the 'credit_card_data' DataFrame. It helps identify any missing data points, which may require handling before further analysis or model training.

```
credit_card_data['Class'].value_counts()
```

This code counts the occurrences of each unique value in the 'Class' column of the 'credit_card_data' DataFrame. In the context of credit card fraud detection, 'Class' typically represents whether a transaction is fraudulent (1) or not fraudulent (0). This command helps understand the distribution of fraudulent and non-fraudulent transactions in the dataset.

# CODE EXPLAINATION

```
print(legit.shape)          (284315, 31)
print(fraud.shape)          (492, 31)
```

This code prints the dimensions (number of rows and columns) of the 'legit' and 'fraud' DataFrames, providing insight into the number of non-fraudulent and fraudulent transactions, respectively.

```
legit.Amount.describe()
```

This code provides descriptive statistics (such as count, mean, standard deviation, minimum, quartiles, and maximum) for the 'Amount' column in the 'legit' DataFrame. It helps understand the distribution of transaction amounts for non-fraudulent transactions.

```
new_dataset = pd.concat([legit_sample, fraud], axis=0)
```

This code concatenates two DataFrames, 'legit_sample' and 'fraud', along the rows (axis=0) to create a new DataFrame named 'new_dataset'. This operation combines a sample of non-fraudulent transactions with all fraudulent transactions, potentially for further analysis or modeling.

**OUTPUT**

```
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

# CODE EXPLAINATION

```
# accuracy on training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

This code calculates the accuracy of a machine learning model, named 'model', on the training data. It predicts the target variable (Y_train) using the features (X_train), and then compares the predictions with the actual target values to compute the accuracy score.

OUTPUT

```
Accuracy on Training data :  0.9529860228716646
```

```
# accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

This code computes the accuracy of a machine learning model, 'model', on the test data. It generates predictions for the target variable (Y_test) using the features (X_test), and then evaluates the accuracy by comparing these predictions with the actual target values.

OUTPUT

```
Accuracy score on Test Data :  0.934010152284264
```

# OVERVIEW

THE PROJECT INVOLVES LOADING, EXPLORING, PREPARING, AND EVALUATING A DATASET FOR CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING TECHNIQUES, LIKELY LOGISTIC REGRESSION GIVEN THE CONTEXT OF THE CODE SNIPPETS. THE AIM IS TO DEVELOP A MODEL CAPABLE OF ACCURATELY IDENTIFYING FRAUDULENT TRANSACTIONS BASED ON TRANSACTIONAL FEATURES

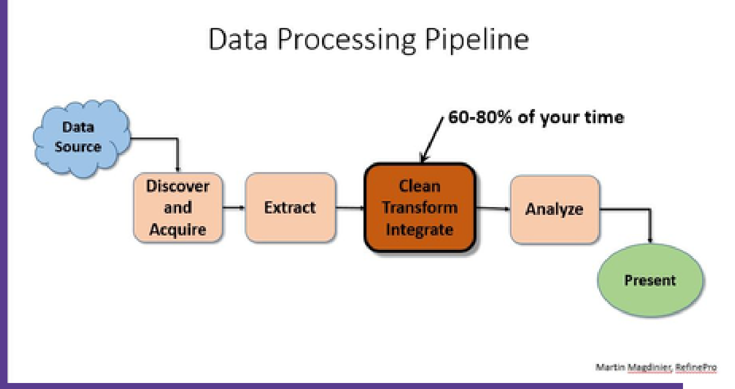
# DATA LOADING AND INSPECTION

- Loaded a dataset named 'creditcard.csv' into a Pandas DataFrame named 'credit_card_data'.
- Inspected the first 5 rows of the dataset to understand its structure and content.
- Obtained information about the dataset to understand its characteristics and data types.
- Checked for missing values in the dataset to ensure data quality.

# DATA EXPLORATION:

- Counted the occurrences of each class (fraudulent and non-fraudulent transactions) in the 'Class' column to understand the distribution of classes.
- Separated the dataset into two DataFrames: 'legit' containing non-fraudulent transactions and 'fraud' containing fraudulent transactions.
- Examined descriptive statistics for transaction amounts in the 'legit' DataFrame to understand the distribution of non-fraudulent transaction amounts.

# DATA PREPARATION



Martin Magdinier, RefinePro

- Concatenated a sample of non-fraudulent transactions ('legit_sample') with all fraudulent transactions ('fraud') along the rows to create a new dataset ('new_dataset'). This likely aimed to balance the classes for modeling purposes.
-

# MODEL EVALUATION:

- Used a machine learning model named 'model' to predict the target variable (fraud or non-fraud) using features (X_train and X_test).
- Calculated the accuracy of the model on the training data ('training_data_accuracy') by comparing the predictions with the actual target values (Y_train).
- Computed the accuracy of the model on the test data ('test_data_accuracy') using the same procedure.
-

# THANK YOU

overview