# CREDIT CARD FRAUD DETECTION
# Using LOGISTIC REGRESSION ALGORITHM

**A PROJECT REPORT**

**Submitted in Partial Fulfilment for the degree of Bachelor of Technology in ELECTRICAL ENGINEERING**

**MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY,WEST BENGAL**


*Submitted by*

**TAPAS DEBNATH (10301618025)**
**SURJAYAN DUTTA (10301618026)**
**SAPTARSHI JANA (10301618050)**
**PRITAM MAITY (10301618072)**
**KUMOD KUMAR YADAV (10301618097)**

*Under the guidance of*

**MR. SOURAV KUMAR DAS**
**Department Of Electrical Engineering**


**HALDIA INSTITUTE OF TECHNOLOGY**

**Haldia, West Bengal,P.O- HIT,HIT College Rd**
**Khudiram Nagar, Haldia,West Bengal ,721657**

**DEPARTMENT OF ELECTRICAL ENGINEERING**
**Haldia Institute of Technology Affiliated to**

**MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY**

**THIS CERTIFIES THAT**

The project report entitled "CREDIT CARD FRAUD DETECTION USING LOGISTIC REGRESSION ALGORITHM" submitted by TAPAS DEBNATH (University roll no. 10301618025) SURJAYAN DUTTA (University roll no:10301618026); SAPTARSHI JANA (University roll no:10301618026) ; PRITAM MAITY (University roll no: 10301618072); KUMUD KUMAR YADAV (University roll no: 10301618097)" for 8th semester examination have been prepared following the guidelines of Bachelor of Technology degree in Electrical Engineering, awarded by the Maulana Abul Kalam Azad University of Technology. They have carried out the project work under my supervision.

**Prof.(Dr.) DILIP DEY**
Head of Electrical Engineering Department

**Mr. SOURAV KUMAR DAS**
Assistant Professor, Project Supervisor

# ACKNWOLEDGEMENT

With a deep sense of gratitude, I wish to express my sincere thanks to my guide, Mr. Sourav Kumar Das, Assistant Professor, Electrical Engineering Department for giving us the opportunity to work under him on this thesis. I truly appreciate and value his esteemed guidance and encouragement from the beginning to the end of this thesis. We are extremely grateful to him. His knowledge and company at the time of crisis would be remembered lifelong. We want to thank all my teachers for providing a solid background for my studies and research thereafter. They have been great sources of inspiration to us and we thank them from the bottom of my heart. We will be failing in our duty if we do not mention the laboratory staff and administrative staff of this department for their timely help. We also want to thank our parents, who taught us the value of hard work by their own example.

We would like to share this moment of happiness with our parents. They rendered us enormous support during the whole tenure of our stay in Haldia Institute of Technology. Finally, we would like to thank all whose direct and indirect support helped us complete our thesis in time.

We would like to thank our department for giving us the opportunity and platform to make our effort a successful one. We would like to give a special thanks to our head of the Department Prof.(Dr.) Dilip Dey for helping us out with our project.

I would like to express my special thanks to our teachers who gave us the golden opportunity to do this wonderful project on the topic which also helped me in doing a lot of Research and I came to know about so many new things. I am really thankful to them.

Lastly we would also like to thank our team who helped us a lot in finalizing this project within the limited time frame.

# Table of Content

# LIST OF FIGURES

# ABSTRACT

Due to the increase in the number of customers and the increase in the number of companies using credit cards to terminate the transaction, the number of fraud cases has increased dramatically. Dealing with noisy and unequal data, as well as outsiders, has highlighted this problem. In this work, it is proposed to detect fraud using artificial intelligence. The proposed system uses retrofitting to create a divider to prevent fraud in credit card fraud. Managing the contaminated data and ensuring a high level of detection accuracy, a pre-processing step is used. The pre-processing step uses two main novel approaches to refine the data: a descriptive based method and a composite-based method. Compared to the two known categories, the supporting vector machine separator and the voting phase, the proposed separator shows better results in terms of accuracy, sensitivity, and error rate.

Keywords — Classifier; recession; accuracy; smoothness; artificial intelligence; opposite confirmation.

# INTRODUCTION

By the definition of fraud, the purpose of fraud is to gain personal or financial benefit through fraud. Based on this, fraud detection and prevention are two important ways to avoid fraud loss. Fraud deterrence is a preventative method for avoiding fraudulent activities, and fraud detection is a method for detecting fraudulent transactions by fraudsters. Today, various payment cards such as credit cards, charge cards, debit cards and prepaid cards are widely used. These are the most popular payment methods in some countries. In fact, advances in digital technology have paved the way for us to process money, especially payments, from physical activity to digital activity using electronic means. This has revolutionized the outlook for monetary policy, including the business strategies and operations of both large and small businesses. Credit card fraud is the fraudulent use of credit card information to purchase products and services. These transactions can be performed physically or digitally. In a physical transaction, the credit card is physically present. Digital transactions, on the other hand, are done over the internet or by phone. Cardholders typically provide their card number, card verification number, and expiration date through a website or phone. With the rapid increase in e-commerce in recent years, the use of credit cards has increased significantly. Approximately 317 million credit card transactions were processed in Malaysia in 2011, and this number increased to 447 million in 2018. In 2015, global credit card fraud reached a record $ 2184 billion reported by. As the use of credit cards increases, so does the number of fraud cases. Although various verification methods have been implemented, the number of credit card frauds has not decreased significantly. The potential for significant cash profits, combined with the ever-changing nature of financial services, creates many opportunities for fraudsters. Payment card fraud funds are often used for criminal activities that are difficult to prevent. B. To support terrorist attacks. The Internet is the preferred place for scammers because they can disguise their location and identity. The recent surge in credit card fraud has hit the financial sector directly. Losses from credit card fraud primarily hurt merchants as they bear all costs, including card issuer fees, administration fees and other costs. All losses are borne by the trader, which increases the price of the commodity and reduces the discount. Therefore, it is very important to reduce this loss. To minimize fraud, you need an effective fraud detection system.

# LITERATURE SURVEY

Fraud is illegal or criminal deception aimed at bringing financial or personal gain. This is a deliberate act aimed at gaining unauthorized financial gain, contrary to law, regulation or policy. Numerous references related to the detection of anomalies and frauds in this area have already been published and are open to the public. A comprehensive study conducted by Clifton Phua and his collaborators revealed that the techniques used in this area include data mining applications, automated fraud detection, and adversary detection. rice field. In another article, Suman of Research Scholar, GJUS & T of Hisar HCE, introduced techniques such as unsupervised learning and unsupervised learning to detect credit card fraud. Although these methods and algorithms have had unexpected success in some areas, they have failed to provide a durable and consistent fraud detection solution.

A similar area of research was presented by WenFang YU and NaWang to accurately predict fraudulent transactions using outlier mining, outlier detection mining, and distance summing algorithms in a credit card transaction record emulation experiment for particular commercial bank. Did. Outlier mining is an area of data mining used primarily in the area of money and the internet. It deals with the detection of objects that are detached from the main system. Non-genuine transaction. They took the customer behavior attribute and calculated the distance between that attribute's observations and its predictions based on the values of those attributes. Non-traditional such as hybrid data mining / complex network classification algorithms that can detect rogue instances in real card transaction records based on network reconstruction algorithms that allow the creation of representations of instance deviations from reference groups. Techniques have usually proven to be efficient media. Size online transaction. There was also an effort to move forward from a whole new dimension. Attempts have been made to improve the interaction of alert feedback in the event of fraudulent transactions.

In the event of a fraudulent transaction, the approved system will be alerted and a response will be sent to reject the transaction in progress.

# OBJECTIVE AND SCOPE

The use of credit cards to perform financial transactions at banks or other institutions is a common action in light of the currently available technology. Online payments (or any other online transactions) bring benefits to companies and individuals in terms of the convenience, velocity, and flexibility of performing daily duties. The work in  presented a statistical analysis related to the usage of credit cards over five years (from 2006 to 2010). This reflected the huge dependency on credit cards by both people and organizations. To take advantage of advanced technologies, companies try to use advanced techniques to provide highquality services to customers. Automation can be seen as the best solution for attracting more customers and consequently collecting more financial gain. The process of converting a manual system to a fully automatic on, as found in smart cities, is not without risk.
Many surveys have shown that the increase in the dependence on credit cards to perform financial transactions is accompanied by an increasing rate of fraud, as seen in . The increasing capabilities of the attackers or the hackers have accentuated the problem since these people can exploit security gaps to obtain sensitive information about users or their credit information to perform malicious activities, such as fraud . To define this problem accurately, Fig. 1 shows the general scenario of performing credit card fraud.


Fig 1: General Scenario of Online Fraud

As shown in Fig. 1, the attacker can carry out malicious sports on many facets of the net procedure. To clear up this trouble, a fraud detection device is needed. synthetic intelligence (AI) is defined because the research field that pursuits at appearing device learning to reap an wise device which can carry out responsibilities on behalf of the person. this could be carried out via  fundamental steps: training and checking out. AI is hired to build systems for fraud detection, inclusive of classification based systems , clustering-based totally systems , neural network-based totally structures, and support vector machine-based totally structures . although AI-based totally systems can perform properly, they suffer from a few important issues. First, the time period “imbalanced data” refers to unbalanced statistics used for schooling, where one class of the information is ruled with the aid of the alternative (i.e., the majority of information belong to one class and the relaxation belong to the other). This negatively impacts the accuracy of detection . 2nd, the time period “noisy records” refers back to the existence of outliers within the records employed for education. Outliers can be visible out of doors of the regular context of the facts. This problem also leads to poor detection accuracy . 1/3, the concept of drift way that the behaviour of the patron modifications, resulting in adjustments within the facts circulation whilst coping with on line information detection in real time.

# Research Questions

On the basis of the empirical evidence, the following research questions are developed to guide this study and meet its objectives.

- How can a fraud detection system be built using AI that can deal with imbalanced data effectively?
- How can we smooth (or clean) the data before using it for training the machine to ensure high detection accuracy?
- How can the system detect fraud by adapting to the behaviour of the user?

# Contributions

The contributions of this work can be summarized as follows:

- An AI-based system for fraud detection is proposed. The system uses logistic regression to build a classifier called the LogR classifier. The LogR classifier has the ability to deal with imbalanced data and adapt to the behaviour of the user by employing the cross-validation technique.
- To ensure high accuracy detection, two main methods are used to clean the data. The mean-based method deals with missing values, and the clustering-based method deals with outliers.
- Extensive experiments are conducted to train and test the proposed classifier using a standard database.

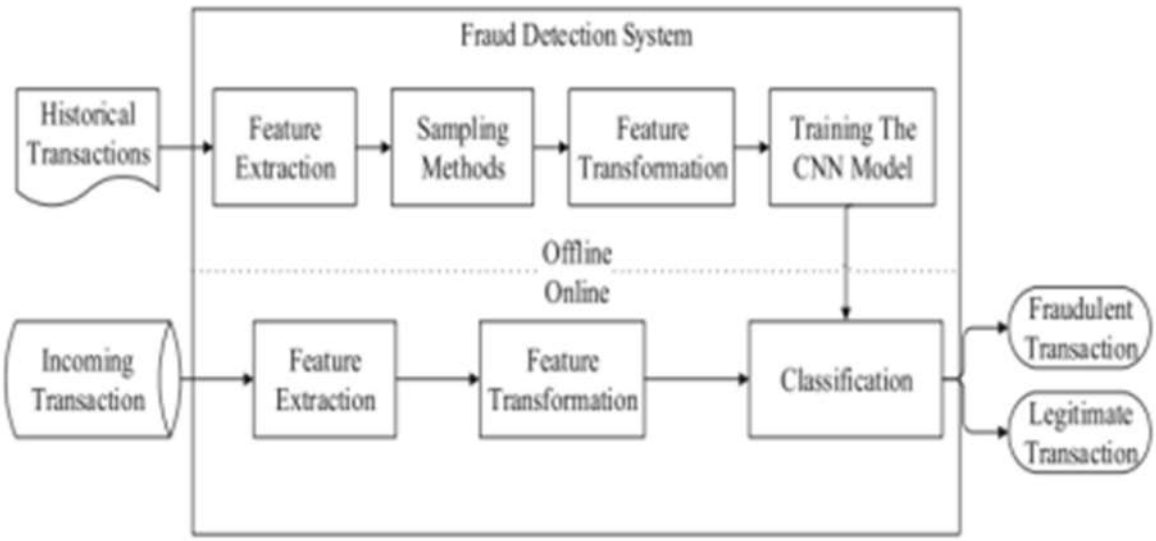



Fig 2: General Scenario of the Fraud Detection System Proposed in the Work

# METHODOLOGY

## Background

The background refers to the credit card research field in terms of the intersection of multiple research sectors. This field can be viewed as the intersection of four main domains, as illustrated in Fig. 3. The definitions of the domains and terms that are applied in this study are listed below.

**Artificial Intelligence (AI):** It can be defined as the science that addresses the methods used for training machines to mimic the brains of humans. In other words, machines can be used to make decisions on behalf of human users. In this context, data mining tasks, such as classification, clustering, applying association rules, and using neural networks, are employed.

**Financial Systems:** These can be defined as the systems that are used to convert manual transactions into digital transactions. In this context, the term "transaction" denotes any financial activity that may be performed by a user based on a specific system .

**Chip Industry:** This term refers to the manufacturing of chips to store critical information on the card of the user. The information acts as a key to trigger any transaction. However, the chip is programmed to match some passwords to allow access to financial interfaces.

**Internet of Things**: It can be defined as a collection of devices connected via a network. The devices vary from small devices with low processing power (such as watches) to large devices high processing power, such as mobile devices. Using IoT devices to perform financial transactions is vital, especially in light of the goal of shifting toward smart cities .

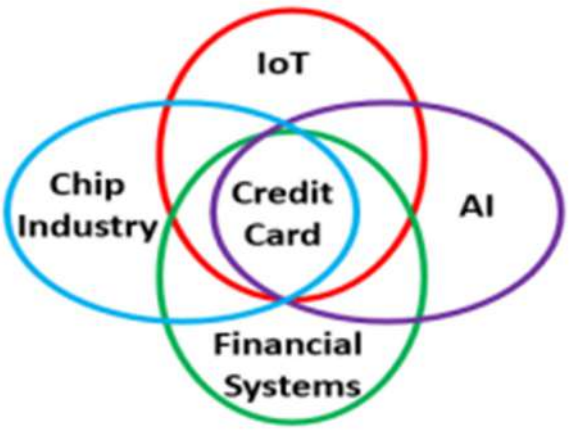



Fig 3: The Intersection of Credit Card Research and other Research Fields

# Groups of AI-based Techniques

Artificial intelligence (AI) is defined as enabling machines to make decisions on behalf of human users. In this context, data mining tasks, such as classification, clustering, applying association rules, and using neural networks, are employed . In addition, AI is employed to build systems for fraud detection, such as classification-based systems , clustering-based systems , neural network-based systems and support vector machine-based systems .

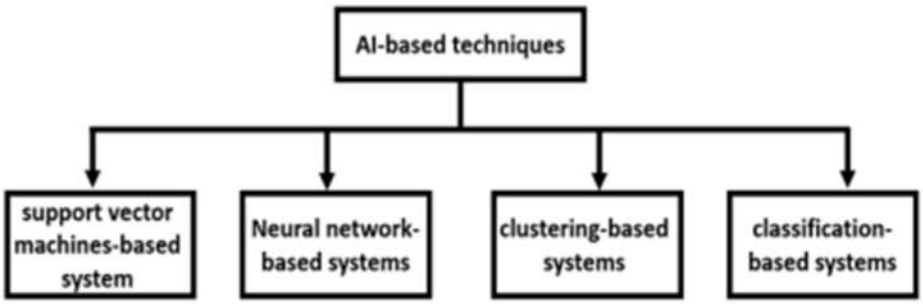



Fig. 4: Categories of AI-based Techniques for Fraud Detection.

# PROCESS and STEPS

There are nine steps, starting with the selection of the database and ending with the use of the classifier in real-life situations. The reason behind selecting logistic regression to build the classifier is related to its efficiency of detecting frauds based on its ability to isolate the data that belong to different binary classes.

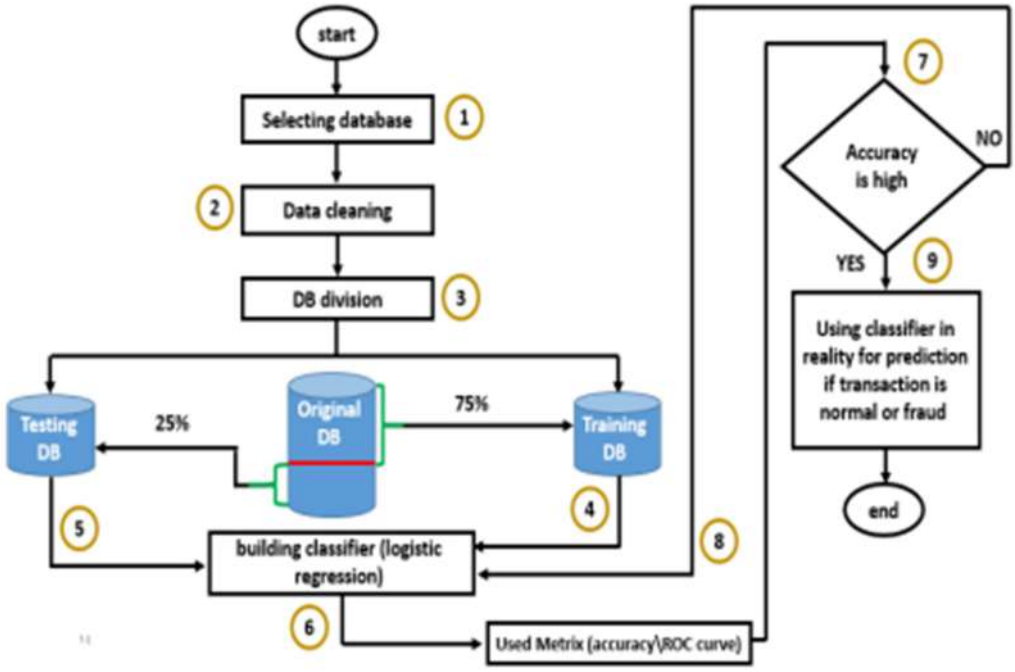



Fig. 5: Flow Chart of the Proposed Approach.

# Selecting the Database:

This work uses a standard dataset that is available on the internet . The dataset contains transactions made using credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred over two days, where we have 492 fraudulent cases out of 284,807 transactions. The dataset is highly unbalanced, and the positive class (fraudulent cases) accounts for 0.172% of all transactions.

# Data Cleaning

The goal of this step is to clean the data and prepare it for the training phase of the classifier. In general, data in reality are noisy. Therefore, a cleaning step is necessary. In the context of the data cleaning process, the procedure is as follows:
1) Fill in the missing values. A missing value means that a cell of a given record is empty due to an mistake during entry.
2) Solve any inconsistencies. This means that if there is a collision in the data, this collision must be resolved.
3) Remove any outliers. Outliers refer to abnormal values (i.e., very high values or very low values).

Fortunately, most of the data used in the data set are cleaned except for some missing values and outliers. The mechanism that is used for handling the missing values depends on the mean (mathematical operation) since the data are numbers
For the handling of outliers, a clustering-based method is employed in this work. The key idea is to create three clusters (one for the normal data, a second one for high values, and a third for low values). After grouping the data into the clusters, the last two clusters (i.e., those that contain outlier

# Database Division

In this step, the database is divided into training and testing databases. The goal of the training database is to construct the classifier (model), while the goal of the testing database is to test (evaluate) the built classifier. In this work, the cross-validation method is used to divide the database, which is divided into 10 parts. The database is divided into 10 parts (i.e., the value of $k = 10$ in the cross-validation method). In the first iteration ($k = 1$), the first nine parts are considered a training set, while the last part of the database is considered a testing set. In the second iteration ($k = 2$), both the first eight parts and the tenth part are considered as a training set, while the ninth part of the database is considered a testing set. This process continues until the last iteration ($k = 10$), where the first part is the testing set and the last nine parts are the training set.

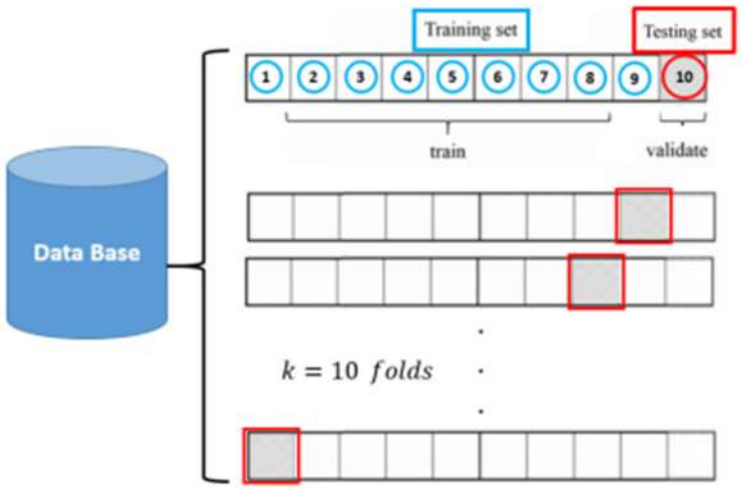



Fig.6: Division of the Database based on Cross Validation.

# Building the Classifier

In the context of building the classifier, logistic regression is employed. Logistic regression is more advanced than linear regression. The reason for this is that linear regression cannot classify data that are widely distributed in a given space, as shown in Fig.

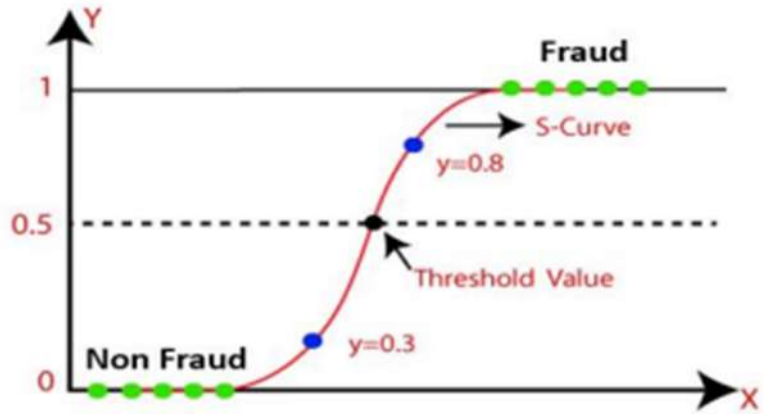



Fig.7 : The Concept of Logistic Regression Classification

Logistic regression is a classification algorithm. It is used to predict a binary outcome based on a set of independent variables.

Logistic regression has the following advantages:
1) Logistic regression is easier to implement than linear regression and is very efficient to train.
2) It makes no assumptions about the distributions of classes in the feature space.
3) It can easily be extended to multiple classes (multinomial regression).
4) It is very efficient for classifying unknown records.

The logistic regression equation can be obtained from the linear regression equation. The mathematical steps to obtain logistic regression equations are given below:

The equation of the straight line can be written as:
$y = a0 + a1 \times x1 + a2 \times x2 + \cdots ak \times xk \ldots$ (1)

In logistic regression, y can be between 0 and 1 only, so we divide the above equation by $(1 - y)$: $y\ 1{-}y\ |0\ for\ y = 0\ and\ \infty\ for\ y = 1$ …(2)

As a result, the logistic regression equation is defined as:
$\log [\ y\ 1{-}y\ ] = a0 + a1 \times x1 + a2 \times x2 + \cdots ak \times xk$ …(3)

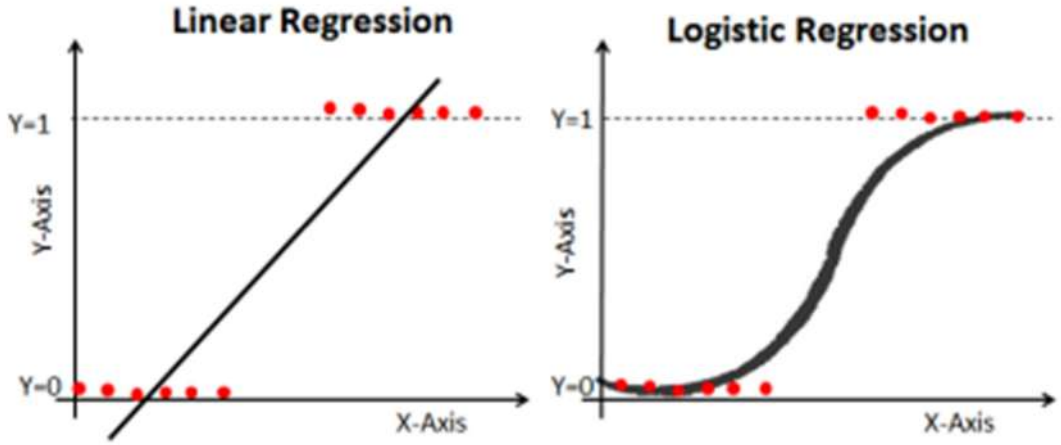



Fig.8: A Visual Comparison between Linear and Logistic Regression

In other words, the fraud class takes the value "1", while the non-fraud class takes the value "0". A threshold of 0.5 is used to differentiate between the two classes.

# Testing the Classifier

Since the cross-validation method divides the database into 10 parts, there are 10 testing data sets. Each testing data set is used to test one classifier (there are 10 classifiers). This in turn gives the model an advantage by allowing it to use the whole database for testing as well as for training. The testing process is tightly coupled with the accuracy of the model. Calculating the final accuracy involves calculating the accuracy of each classifier. Formally, let $Acck\ C$ denote the accuracy of a given trained classifier. Then, the final accuracy of the final classifier ($ACC$ ) is obtained based on the "average" mathematical operation.

$$ACC_F^C = \frac{\sum_{k=1}^{10} Acc_k^C}{k}$$

# Evaluating the Classifier

In general, a confusion matrix is an effective benchmark for analysing how well a classifier can recognize records of different classes. The confusion matrix is formed based on the following terms:
1) True positives (TP): positive records that are correctly labelled by the classifier.
2) True negatives (TN): negative records that are correctly labelled by the classifier.
3) False positives (FP): negative records that are incorrectly labelled positive.
4) False negatives (FN): positive records that are mislabelled negative. Table shows the confusion matrix in terms of the TP, FN, FP, and TN values. Relying on the confusion matrix, the accuracy, sensitivity, and error rate metrics are derived. For a given classifier, the accuracy can be calculated by considering the recognition rate, which is the percentage of records in the test set that are correctly classified (fraudulent or non-fraudulent). The accuracy is defined as:

$$Accuracy = \frac{(TP+TN)}{number\ of\ all\ records\ in\ the\ testing\ set}$$

| **Actual class (Predicted class)** | **Confusion matrix** | | |
|---|---|---|---|
| | **C1** | **¬ C1** | **Total** |
| C1 | True positives (TP) | False negatives (FN) | TP + FN = P |
| ¬ C1 | False positives (FP) | True negatives (TN) | FP + TN = N |

.

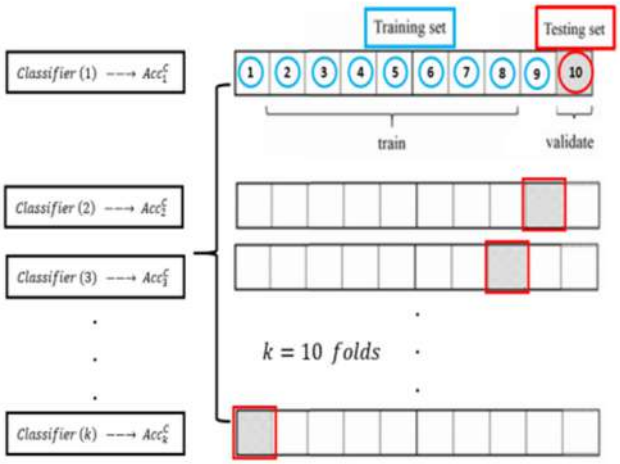



*Fig 9:* Classifiers with Corresponding Accuracies.

# Examining the Value of the Accuracy

In this step, the final calculated accuracy is examined. If it is accepted, then the classifier can be used in real-life situations. Otherwise, the process of building the classifier has a problem, and then retraining the classifier is required.

# RESULT AND DISCUSSION

This section is structured so that the specifications of the machine used to implement the proposed classifier are introduced. Then, the classifiers that are compared with the proposed classifier are described. The system is performed on a machine that has the specifications summarized. The programming language used for the implementation of the classifier is Python.

Since the cross-validation method is used to divide the database, we obtain ten sub-classifiers as mentioned previously. The process of calculating the final values of the AI-based metrics depends on the “average” mathematical operation.

LOGR classifier achieves the best values in terms of accuracy, sensitivity, and error rate. The reason behind this is related to the efficient preprocessing technique used to remove outliers and manipulate the missing values. In addition, cross validation ensures that the entire database is employed as both the training and testing data sets, and this in turn enhances the three metrics.

The detection of credit card fraud is a vital research field. This is because of the increasing number of fraud cases in financial institutions. This issue opens the door for employing artificial intelligence to build systems that can detect fraud. Building an AI-based system to detect fraud requires a database to train the system (or classifier). The data in reality are dirty and have missing values, noisy data, and outliers. Such issues negatively affect the accuracy rate of the system. To overcome these problems, a logistic regression-based classifier is proposed. The data are first cleaned using two methods: the mean-based method and clustering-based method. Second, the classifier is trained based on the crossvalidation technique (folds=10), which ensures that the whole database is used as both the training data set and testing data set. Finally, the proposed classifier is evaluated based on the accuracy, sensitivity, and error rate metrics. The logistic regression-based classifier generates the best results.

# SOURCE CODE with proper explanation:

Credit card frauds are easy and friendly targets. E-commerce and many other online sites have increased the online payment modes, increasing the risk for online frauds. Increase in fraud rates, researchers started using different machine learning methods to detect and analyse frauds in online transactions. The main aim of the paper is to design and develop a novel fraud detection method for Streaming Transaction Data, with an objective, to analyse the past transaction details of the customers and extract the behavioural patterns. Where cardholders are clustered into different groups based on their transaction amount.

For many banks, retaining high profitable customers is the number one business goal. Banking fraud, however, poses a significant threat to this goal for different banks. In terms of substantial financial losses, trust and credibility, this is a concerning issue to both banks and customers alike.

In the banking industry, credit card fraud detection using machine learning is not only a trend but a necessity for them to put proactive monitoring and fraud prevention mechanisms in place. Machine learning is helping these institutions to reduce time-consuming manual reviews, costly chargebacks and fees as well as denials of legitimate transactions.

In this project we will detect fraudulent credit card transactions with the help of Machine learning models. We will analyse customer-level data that has been collected and analysed during a research collaboration of Worldline and the Machine Learning Group.

## What is the difference between ML Credit Card Fraud Detection and Conventional Fraud Detection?

**Machine Learning-based Fraud Detection:**

1.Detecting fraud automatically

2.Real-time streaming

3.Less time needed for verification methods

4.Identifying hidden correlations in data

**Conventional Fraud Detection:**

1.The rules of making a decision on determining schemes should be set manually.

2.Takes an enormous amount of time

3.Multiple verification methods are needed; thus, inconvenient for the user

4.Finds only obvious fraud activities

***Credit Card Fraud Detection with Machine Learning*** is a process of data investigation by a Data Science team and the development of a model that will provide the best results in revealing and preventing fraudulent transactions. This is achieved through bringing together all meaningful

features of card users' transactions, such as Date, User Zone, Product Category, Amount, Provider, Client's Behavioral Patterns, etc. The information is then run through a subtly trained model that finds patterns and rules so that it can classify whether a transaction is fraudulent or is legitimate. All big banks like Chase use fraud monitoring and detection systems.

## How Does Credit Card Fraud Happen?

Credit card fraud is usually caused either by card owner's negligence with his data or by a breach in a website's security. Here are some examples:

1.A consumer reveals his credit card number to unfamiliar individuals.

2.A card is lost or stolen and someone else uses it.

3.Mail is stolen from the intended recipient and used by criminals.

4.Business employees copy cards or card numbers of its owner.

5.Making a counterfeit credit card.

**Credit Card Fraud Detection Systems:**

1.Off-the-shelf fraud risk scores pulled from third parties (e.g. LexisNexis or MicroBilt).

2.Predictive machine learning models that learn from prior data and estimate the probability of a fraudulent credit card transaction.

3.Business rules that set conditions that the transaction must pass to be approved (e.g. no OFAC alert, SSN matches, below deposit/withdrawal limit, etc.)

Among these fraud analytics techniques, predictive Machine Learning models belong to smart Internet security solutions.

**Artificial Intellegence Fraud Detection System Implementation Steps:**

***1.Data Mining***. Implies classifying, grouping, and segmenting of data to search millions of transactions to find patterns and detect fraud.

***2.Pattern Recognition***. Implies detecting the classes, clusters, and patterns of suspicious behavior. Machine Learning here represents the choice of a model/set of models that best fit a certain business problem. For example, the neural networks approach helps automatically identify the characteristics most often found in fraudulent transactions; this method is most effective if you have a lot of transaction samples.

Once the Machine Learning-driven Fraud Protection module is integrated into the E-commerce platform, it starts tracking the transactions. Whenever a user requests a transaction, it is processed for some time. Depending on the level of predicted fraud probability, there are three possible outcomes:

1.If the probability is less than 10%, the transaction is allowed.

2.If the probability is between 10% and 80%, an additional authentication factor (e.g. a one-time SMS code, a fingerprint, or a Secret Question) should be applied.

3.If the probability is more than 80%, the transaction is frozen, so it should be processed manually.

## Requirements for Fraud Detection with AI-based Methods

To run an AI-driven strategy for Credit Card Fraud Analytics, a number of critical requirements should be met. These will ensure that the model reaches its best detection score.

**Amount of data:** Training high-quality Machine Learning models requires significant internal historical data. That means if you do not have enough previous fraudulent and normal transactions, it would be hard to run a Machine Learning model on it because the quality of its training process depends on the quality of the inputs. Because it is rarely the case that a training set contains an equal amount of data samples in two classes, dimensionality reduction or data augmentation techniques are used for that.

**Quality of data:** Models may be subject to bias based on the nature and quality of historical data. This statement means that if the platform maintainers did not collect and sort the data neatly and properly or even mixed the information of fraudulent transactions with the information of normal ones, that is likely to cause a major bias in the model's results.

**The integrity of factors:** If you have enough data that is well-structured and unbiased, and if your business logic is paired nicely with the Machine Learning model, the chances are very high that fraud detection will work well for your customers and your business.

## Data Understanding :

The data set includes credit card transactions made by European cardholders over a period of two days in September 2013. Out of a total of 2,84,807 transactions, 492 were fraudulent. This data set is highly unbalanced, with the positive class (frauds) accounting for 0.172% of the total transactions.

#### IMPORTING DEPENDENCIES

It is a good practice to import all the necessary libraries in one place — so that we can modify them quickly.

1.***NumPy***(Numerical Python) is the fundamental package for numerical computation in Python; it contains a powerful N-dimensional array object. It has around 18,000 comments on GitHub and an active community of 700 contributors. It's a general-purpose array-processing package that provides high-performance multidimensional objects called arrays and tools for working with them.

2.***Pandas*** (Python data analysis) is a must in the data science life cycle. It is the most popular and widely used Python library for data science, along with NumPy in matplotlib. With around 17,00 comments on GitHub and an active community of 1,200 contributors, it is heavily used for data analysis and cleaning. Pandas provides fast, flexible data structures, such as data frame CDs, which are designed to work with structured data very easily and intuitively.

3.***Matplotlib*** has powerful yet beautiful visualizations. It's a plotting library for Python with around 26,000 comments on GitHub and a very vibrant community of about 700 contributors. Because of the graphs and plots that it produces, it's extensively used for data visualization. It also provides an object-oriented API, which can be used to embed those plots into applications.

4.Next in the list of the top python libraries for data science comes **Scikit-learn**, a machine learning library that provides almost all the machine learning algorithms you might need. Scikit-learn is designed to be interpolated into NumPy and SciPy.

```
# Importing the libraries


import numpy as np
import pandas as pd
import time

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler

import sklearn
from sklearn import metrics


# To ignore warnings

import warnings
warnings.filterwarnings("ignore")
```

Data scientists are expected to build high-performing machine learning models, but the starting point is getting the data into the Python environment. Only after importing the data can the data scientist clean, wrangle, visualize, and build predictive models on it.

**Importing the dataset** is pretty much simple. You can use pandas module in python to import it.

```
# Mounting the google drive
from google.colab import drive
```

```
drive.mount('/content/gdrive')

# Loading the data
credit_card_data= pd.read_csv('gdrive/MyDrive/Colab Notebooks/creditcard.csv')
# df = pd.read_csv('./data/creditcard.csv')
credit_card_data.head()
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.m
```

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.0 | -1.359807 | -0.072781 | 2.536347 | 1.378155 | -0.338321 | 0.462388 | 0.239599 | 0.0986 |
| **1** | 0.0 | 1.191857 | 0.266151 | 0.166480 | 0.448154 | 0.060018 | -0.082361 | -0.078803 | 0.0851 |
| **2** | 1.0 | -1.358354 | -1.340163 | 1.773209 | 0.379780 | -0.503198 | 1.800499 | 0.791461 | 0.2476 |
| **3** | 1.0 | -0.966272 | -0.185226 | 1.792993 | -0.863291 | -0.010309 | 1.247203 | 0.237609 | 0.3774 |
| **4** | 2.0 | -1.158233 | 0.877737 | 1.548718 | 0.403034 | -0.407193 | 0.095921 | 0.592941 | -0.2705 |

5 rows × 31 columns

```
credit_card_data.shape
```

```
(284807, 31)
```

```
credit_card_data.tail()
```

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | |
|---|---|---|---|---|---|---|---|---|
| **284802** | 172786.0 | -11.881118 | 10.071785 | -9.834783 | -2.066656 | -5.364473 | -2.606837 | -4.9182 |
| **284803** | 172787.0 | -0.732789 | -0.055080 | 2.035030 | -0.738589 | 0.868229 | 1.058415 | 0.0243 |
| **284804** | 172788.0 | 1.919565 | -0.301254 | -3.249640 | -0.557828 | 2.630515 | 3.031260 | -0.2968 |
| **284805** | 172788.0 | -0.240440 | 0.530483 | 0.702510 | 0.689799 | -0.377961 | 0.623708 | -0.6861 |
| **284806** | 172792.0 | -0.533413 | -0.189733 | 0.703337 | -0.506271 | -0.012546 | -0.649617 | 1.577C |

5 rows × 31 columns

```
credit_card_data.info()  #dataset information
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
```

```
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

As per the count per column, we have no null values. Also, feature selection is not the case for this use case. Anyway, you can try applying feature selection mechanisms to check if the results are optimised.

In the field of data-related research, it is very important to handle missing data either by deleting or imputation(handling the missing values with some estimation).

You may end up building a biased machine learning model which will lead to incorrect results if the missing values are not handled properly. Missing data can lead to a lack of precision in the statistical analysis.

```
# checking the number of missing values in each column
credit_card_data.isnull().sum()
```

```
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
```

```
# distribution of legit transactions & fraudulent transactions
credit_card_data['Class'].value_counts()
```

```
0    284315
1       492
Name: Class, dtype: int64
```

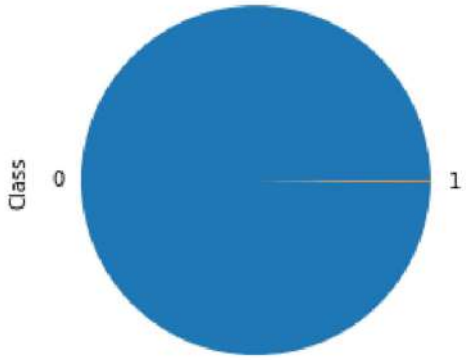
```
# Checking distribution of numerical values in the dataset
credit_card_data.describe()
```

| | Time | V1 | V2 | V3 | V4 | |
|---|---|---|---|---|---|---|
| **count** | 284807.000000 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.848070e+05 | 2.84807 |
| **mean** | 94813.859575 | 3.918649e-15 | 5.682686e-16 | -8.761736e-15 | 2.811118e-15 | -1.5521( |
| **std** | 47488.145955 | 1.958696e+00 | 1.651309e+00 | 1.516255e+00 | 1.415869e+00 | 1.38024 |
| **min** | 0.000000 | -5.640751e+01 | -7.271573e+01 | -4.832559e+01 | -5.683171e+00 | -1.13743 |
| **25%** | 54201.500000 | -9.203734e-01 | -5.985499e-01 | -8.903648e-01 | -8.486401e-01 | -6.9159 |
| **50%** | 84692.000000 | 1.810880e-02 | 6.548556e-02 | 1.798463e-01 | -1.984653e-02 | -5.4335 |
| **75%** | 139320.500000 | 1.315642e+00 | 8.037239e-01 | 1.027196e+00 | 7.433413e-01 | 6.1192 |
| **max** | 172792.000000 | 2.454930e+00 | 2.205773e+01 | 9.382558e+00 | 1.687534e+01 | 3.48016 |

8 rows × 31 columns

```
# Checking the class distribution of the target variable in percentage
print((credit_card_data.groupby('Class')['Class'].count()/credit_card_data['Class'].count())
((credit_card_data.groupby('Class')['Class'].count()/credit_card_data['Class'].count()) *100
```

```
Class
0    99.827251
1     0.172749
Name: Class, dtype: float64
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa846f3790>
```




```
# Checking the correlation
corr = credit_card_data.corr()
corr
```

| | Time | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|---|
| **Time** | 1.000000 | 1.173963e-01 | -1.059333e-02 | -4.196182e-01 | -1.052602e-01 | 1.730721e-01 |
| **V1** | 0.117396 | 1.000000e+00 | 4.135835e-16 | -1.227819e-15 | -9.215150e-16 | 1.812612e-17 |
| **V2** | -0.010593 | 4.135835e-16 | 1.000000e+00 | 3.243764e-16 | -1.121065e-15 | 5.157519e-16 |
| **V3** | -0.419618 | -1.227819e-15 | 3.243764e-16 | 1.000000e+00 | 4.711293e-16 | -6.539009e-17 |
| **V4** | -0.105260 | -9.215150e-16 | -1.121065e-15 | 4.711293e-16 | 1.000000e+00 | -1.719944e-15 |
| **V5** | 0.173072 | 1.812612e-17 | 5.157519e-16 | -6.539009e-17 | -1.719944e-15 | 1.000000e+00 |
| **V6** | -0.063016 | -6.506567e-16 | 2.787346e-16 | 1.627627e-15 | -7.491959e-16 | 2.408382e-16 |
| **V7** | 0.084714 | -1.005191e-15 | 2.055934e-16 | 4.895305e-16 | -4.104503e-16 | 2.715541e-16 |
| **V8** | -0.036949 | -2.433822e-16 | -5.377041e-17 | -1.268779e-15 | 5.697192e-16 | 7.437229e-16 |
| **V9** | -0.008660 | -1.513678e-16 | 1.978488e-17 | 5.568367e-16 | 6.923247e-16 | 7.391702e-16 |
| **V10** | 0.030617 | 7.388135e-17 | -3.991394e-16 | 1.156587e-15 | 2.232685e-16 | -5.202306e-16 |
| **V11** | -0.247689 | 2.125498e-16 | 1.975426e-16 | 1.576830e-15 | 3.459380e-16 | 7.203963e-16 |
| **V12** | 0.124348 | 2.053457e-16 | -9.568710e-17 | 6.310231e-16 | -5.625518e-16 | 7.412552e-16 |
| **V13** | -0.065902 | -2.425603e-17 | 6.295388e-16 | 2.807652e-16 | 1.303306e-16 | 5.886991e-16 |
| **V14** | -0.098757 | -5.020280e-16 | -1.730566e-16 | 4.739859e-16 | 2.282280e-16 | 6.565143e-16 |
| **V15** | -0.183453 | 3.547782e-16 | -4.995814e-17 | 9.068793e-16 | 1.377649e-16 | -8.720275e-16 |
| **V16** | 0.011903 | 7.212815e-17 | 1.177316e-17 | 8.299445e-16 | -9.614528e-16 | 2.246261e-15 |
| **V17** | -0.073297 | -3.879840e-16 | -2.685296e-16 | 7.614712e-16 | -2.699612e-16 | 1.281914e-16 |
| **V18** | 0.090438 | 3.230206e-17 | 3.284605e-16 | 1.509897e-16 | -5.103644e-16 | 5.308590e-16 |
| **V19** | 0.028975 | 1.502024e-16 | -7.118719e-18 | 3.463522e-16 | -3.980557e-16 | -1.450421e-16 |

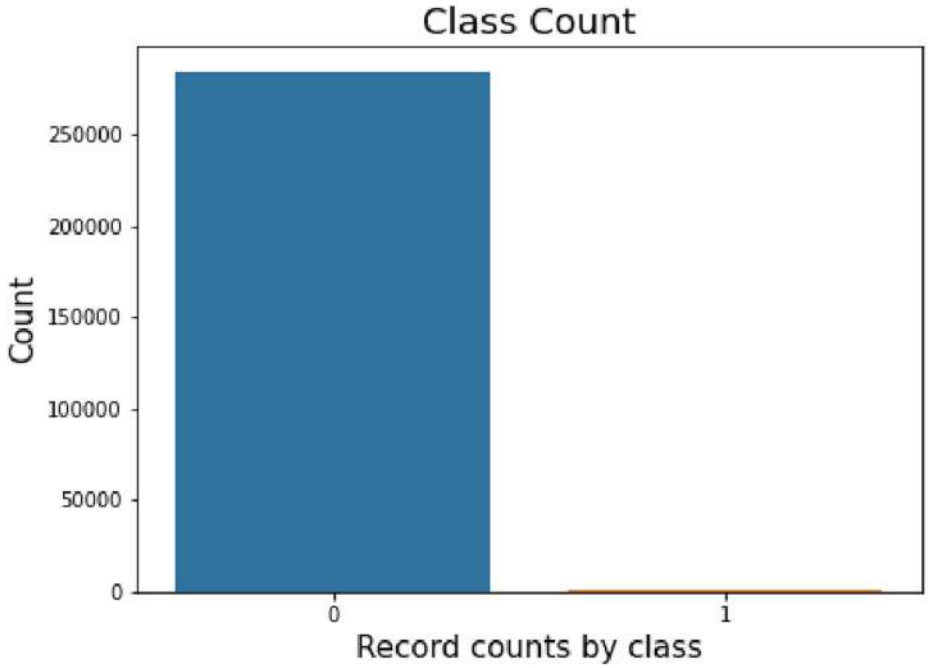
```
# Checking the % distribution of normal vs fraud
classes=credit_card_data['Class'].value_counts()
```

```
normal_share=classes[0]/credit_card_data['Class'].count()*100
fraud_share=classes[1]/credit_card_data['Class'].count()*100

print(normal_share)
print(fraud_share)
```

```
99.82725143693798
0.1727485630620034
```

```
# Create a bar plot for the number and percentage of fraudulent vs non-fraudulent transcatio
plt.figure(figsize=(7,5))
sns.countplot(credit_card_data['Class'])
plt.title("Class Count", fontsize=18)
plt.xlabel("Record counts by class", fontsize=15)
plt.ylabel("Count", fontsize=15)
plt.show()
```




**This Dataset is highly imbalanced**

0 --> Normal Transaction

1 --> fraudulent transaction

The one main thing you will notice about this data is that — the dataset is imbalanced towards a feature. Which seems pretty valid for such kind of data. Because today many banks have adopted different security mechanisms — so it is harder for hackers to make such moves.

Still, sometimes when there is some vulnerability in the system — the chance of such activities can increase.

That's why we can see the majority of transactions belongs to our datasets are normal and only a few percentages of transactions are fraudulent.

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

print(legit.shape)
print(fraud.shape)
```

```
(284315, 31)
(492, 31)
```

Statistical knowledge helps you use the proper methods to collect the data, employ the correct analyses, and effectively present the results. Statistics is a crucial process behind how we make discoveries in science, make decisions based on data, and make predictions.

The mean, median, mode, percentiles, range, variance, and standard deviation are the most commonly used numerical measures for quantitative data. The mean, often called the average, is computed by adding all the data values for a variable and dividing the sum by the number of data values.

```
# statistical measures of the data
legit.Amount.describe()
```

```
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
```

```
fraud.Amount.describe()
```

```
count     492.000000
mean      122.211321
std       256.683288
min         0.000000
25%         1.000000
50%         9.250000
75%       105.890000
max      2125.870000
Name: Amount, dtype: float64
```

```
# compare the values for both transactions
credit_card_data.groupby('Class').mean()
```

| Class | Time | V1 | V2 | V3 | V4 | V5 | V6 | |
|---|---|---|---|---|---|---|---|---|
| 0 | 94838.202258 | 0.008258 | -0.006271 | 0.012171 | -0.007860 | 0.005453 | 0.002419 | 0.0( |

# Under-Sampling

Undersampling is appropriate when there is plenty of data for an accurate analysis. The data scientist uses all of the rare events but reduces the number of abundant events to create two equally sized classes. Typically, scientists randomly delete events in the majority class to end up with the same number of events as the minority class.

The main advantage of undersampling is that data scientists can correct imbalanced data to reduce the risk of their analysis or machine learning algorithm skewing toward the majority. Without resampling, scientists might run a classification model with 90% accuracy.

Build a sample dataset containing similar distribution of normal transactions and Fraudulent Transactions

Number of Fraudulent Transactions --> 492

```
legit_sample = legit.sample(n=492)
```

**Concatenating two Dataframes**

When we concatenate DataFrames, we need to specify the axis. axis=0 tells pandas to stack the second DataFrame UNDER the first one. It will automatically detect whether the column names are the same and will stack accordingly. axis=1 will stack the columns in the second DataFrame to the RIGHT of the first DataFrame.

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

```
new_dataset.head()
```

```
new_dataset.tail()
```

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 |
|---|---|---|---|---|---|---|---|---|
| **279863** | 169142.0 | -1.927883 | 1.125653 | -4.518331 | 1.749293 | -1.566487 | -2.010494 | -0.882850 |
| **280143** | 169347.0 | 1.378559 | 1.289381 | -5.004247 | 1.411850 | 0.442581 | -1.326536 | -1.413170 |
| **280149** | 169351.0 | -0.676143 | 1.126366 | -2.213700 | 0.468308 | -1.120541 | -0.003346 | -2.234739 |
| **281144** | 169966.0 | -3.113832 | 0.585864 | -5.399730 | 1.817092 | -0.840618 | -2.943548 | -2.208002 |
| **281674** | 170348.0 | 1.991976 | 0.158476 | -2.583441 | 0.408670 | 1.151147 | -0.096695 | 0.223050 |

5 rows × 31 columns

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

```
0    492
1    492
Name: Class, dtype: int64
```

```
new_dataset.groupby('Class').mean()
```

| Class | Time | V1 | V2 | V3 | V4 | V5 | V6 | |
|---|---|---|---|---|---|---|---|---|
| **0** | 92478.081301 | -0.035670 | -0.020415 | 0.051259 | -0.075519 | -0.007804 | 0.039392 | 0.0 |
| **1** | 80746.806911 | -4.771948 | 3.623778 | -7.033281 | 4.542029 | -3.151225 | -1.397737 | -5.5 |

2 rows × 30 columns

### Splitting the data into Features & Targets

Data splitting is an important aspect of data science, particularly for creating models based on data. This technique helps ensure the creation of data models and processes that use data models -- such as machine learning -- are accurate.

```
X = new_dataset.drop(columns='Class', axis=1)
Y = new_dataset['Class']
```

```
print(X)
```

```
            Time         V1         V2         V3         V4         V5         V6  \
```

```
       Time        V1        V2        V3        V4        V5        V6  \
37792   39109.0  0.213740 -1.689744 -1.803154  0.516163  1.646675  3.583521
140        87.0 -5.101877  1.897022 -3.458034 -1.277543 -5.517758  2.098366
25455   33618.0 -1.347600  0.328717  1.057139  1.114991  0.492984 -0.221932
34366   37635.0 -9.021973  4.980738 -4.214983 -3.168076 -0.240775  4.656195
69345   53376.0 -0.260219 -2.887931 -0.074508  0.491652 -1.295961  1.167223
...         ...       ...       ...       ...       ...       ...       ...
279863  169142.0 -1.927883  1.125653 -4.518331  1.749293 -1.566487 -2.010494
280143  169347.0  1.378559  1.289381 -5.004247  1.411850  0.442581 -1.326536
280149  169351.0 -0.676143  1.126366 -2.213700  0.468308 -1.120541 -0.003346
281144  169966.0 -3.113832  0.585864 -5.399730  1.817092 -0.840618 -2.943548
281674  170348.0  1.991976  0.158476 -2.583441  0.408670  1.151147 -0.096695

              V7        V8        V9  ...       V20       V21       V22  \
37792   0.289630  0.654983 -0.212519  ...  1.116905  0.354504 -0.344911
140     3.329603  1.250966  0.271501  ... -1.270478 -0.871744 -0.678879
25455  -0.510278  0.739361 -0.668685  ...  0.327888  0.093284 -0.100460
34366  -3.733502 -6.553873  3.490315  ...  2.377352 -0.241141 -0.167430
69345   0.121996  0.217820  0.957714  ...  1.435635  0.238762 -0.764484
...          ...       ...       ...  ...       ...       ...       ...
279863 -0.882850  0.697211 -2.064945  ...  1.252967  0.778584 -0.319189
280143 -1.413170  0.248525 -1.127396  ...  0.226138  0.370612  0.028234
280149 -2.234739  1.210158 -0.652250  ...  0.247968  0.751826  0.834108
281144 -2.208002  1.058733 -1.632333  ...  0.306271  0.583276 -0.269209
281674  0.223050 -0.068384  0.577829  ... -0.017652 -0.164350 -0.295135

              V23       V24       V25       V26       V27       V28  Amount
37792  -0.635714  1.036153  0.629070 -0.324805 -0.084553  0.112238  561.10
140    -0.555900 -0.761660  0.066611  0.767227  0.731634 -0.860310  919.60
25455  -0.150275 -0.535798 -0.221442 -0.349727  0.299235 -0.011044    3.60
34366   0.517135  0.964355  0.847759  0.869946 -0.059640 -0.233492   50.00
69345  -0.665838 -0.729451 -0.135814  0.929224 -0.165579  0.121778  757.21
...          ...       ...       ...       ...       ...       ...     ...
279863  0.639419 -0.294885  0.537503  0.788395  0.292680  0.147968  390.00
280143 -0.145640 -0.081049  0.521875  0.739467  0.389152  0.186637    0.76
280149  0.190944  0.032070 -0.739695  0.471111  0.385107  0.194361   77.89
281144 -0.456108 -0.183659 -0.328168  0.606116  0.884876 -0.253700  245.00
281674 -0.072173 -0.450261  0.313267 -0.289617  0.002988 -0.015309   42.53

[984 rows x 30 columns]
```

```
print(Y)
```

```
37792     0
140       0
25455     0
34366     0
69345     0
         ..
279863    1
280143    1
280149    1
281144    1
281674    1
Name: Class, Length: 984, dtype: int64
```

**Split the data into Training data & Testing Data**

Before splitting train & test — we need to define dependent and independent variables. The dependent variable is also known as X and the independent variable is known as y.

The main idea of splitting the dataset into a validation set is to prevent our model from overfitting i.e., the model becomes really good at classifying the samples in the training set but cannot generalize and make accurate classifications on the data it has not seen before.

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_
```

```
print(X.shape, X_train.shape, X_test.shape)
```

```
(984, 30) (787, 30) (197, 30)
```

# Model Training

Defining models are much easier. A single line of code can define our model. And, in the same way, a single line of code can fit the model on our data.

We can also tune these models by selecting different optimized parameters. But, if the accuracy is better even with less parameter tuning then — no need to make it complex.

**Logistic Regression:** Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems. In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

```
model = LogisticRegression()
```

```
# training the Logistic Regression Model with Training Data
model.fit(X_train, Y_train)
```

```
LogisticRegression()
```

# Model Evaluation

Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses. Model evaluation is important to assess the efficacy of a model during initial research phases, and it also plays a role in model monitoring. Training data is used in Model Evaluation.

**Accuracy Score**

Accuracy score is used to measure the model performance in terms of measuring the ratio of sum of true positive and true negatives out of all the predictions made.But in our opinion, anything greater than 70% is a great model performance. In fact, an accuracy measure of anything between 70%-90% is not only ideal, it's realistic. This is also consistent with industry standards.

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

```
print('Accuracy on Training data : ', training_data_accuracy)
```

```
Accuracy on Training data :  0.940279542566709
```

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

```
print('Accuracy score on Test Data : ', test_data_accuracy)
```

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

# CONCLUSION

Well, congratulation! We just received 93.4% accuracy in our credit card fraud detection. This number should not be surprising as our data was balanced towards one class. The good thing that we have noticed our model is not overfitted. As the results show Logistic Regression provided best results. The result obtained in this project was superior to those achieved by existing methods. Moreover, we implemented our proposed framework on a synthetic credit card fraud dataset to validate the result that were obtained on the credit card fraud dataset. In future works, we intend to use more datasets to validate our framework.

Fraud is a major problem for the whole credit card industry that grows bigger with the increasing popularity of electronic money transfers. To effectively prevent the criminal actions that lead to the leakage of bank account information leak, skimming, counterfeit credit cards, the theft of billions of dollars annually, and the loss of reputation and customer loyalty, credit card issuers should consider the implementation of advanced Credit Card Fraud Prevention and Fraud Detection methods Machine Learning-based methods can continuously improve the accuracy of fraud prevention based on information about each cardholder's behavior.

# REFERENCES


[1] Campus K. Credit card fraud detection using machine learning models and collating machine learning models. Int J Pure Appl Math.

[2] Paschen, Jeannette, Jan Kietzmann, and Tim Christian Kietzmann. "Artificial intelligence (AI) and its implications for market knowledge in B2B marketing." Journal of Business & Industrial Marketing

[3] Abdallah, Aisha, Mohd Aizaini Maarof, and Anazida Zainal. "Fraud detection system: A survey." Journal of Network and Computer Applications.

[4] Somasundaram, Akila, and Srinivasulu Reddy. "Parallel and incremental credit card fraud detection model to handle concept drift and data imbalance." Neural Computing and Applications

[5] Jiang, Changjun, et al. "Credit card fraud detection: A novel approach using aggregation strategy and feedback mechanism." IEEE Internet of Things Journal

[6] P.Sandhya Krishna,Sk.Reshmi Khadherbhi,V.Pavani, Unsupervised or Supervised Feature Finding For Study of Products Sentiment ,International Journal of Advanced Science and Technology

[7] R S M Lakshmi Patibandla and N. Veeranjaneyulu, (2018), "Explanatory & Complex Analysis of Structured Data to Enrich Data in Analytical Appliance", International Journal for Modern Trends in Science and Technology

[8] Burkov A. The hundred-page machine learning book

[9] Yousefi, Niloofar, Marie Alaghband, and Ivan Garibay. "A Comprehensive Survey on Machine Learning Techniques and User Authentication Approaches for Credit Card Fraud Detection."

[10] The Credit card fraud [Online]. https://www.kaggle.com/mlg-ulb/creditcardfraud

[11] Google Colab [Online]. Available: https://colab.research.google.com/

[12] Scikit-learn : machine learning in Python [Online]. https://scikit-learn.org/stable/