**Credit card Fraud Detection using Machine learning**

**OBJECTIVE**

'Fraud' in credit card transactions is unauthorized and unwanted usage of an account by someone other than the owner of that account. Fraud transactions in credit card data transaction are increasing each year. The main objective is to develop novel techniques to detect and prevent such frauds in credit card data transaction.

**ABSTRACT:**

Credit card fraud detection is presently the most frequently occurring problem in the present world. We made an attempt for finding the frauds in the credit card business by using the algorithms which adopted machine learning techniques. We are using Decision Tree, Random Forest and Extreme Gradient boosting algorithms. The efficiency of the model can be decided by using some public data as sample. Then, an actual world credit card facts group from a financial institution is examined. Along with this, some clatter is supplemented to the data samples to auxiliary check the sturdiness of the systems. The significance of the methods used in the paper is the first method constructs a tree against the activities performed by the user and using this tree scams will be suspected. In the second method a user activity based forest will have constructed and using this forest an attempt will be made in identifying the suspect. The investigate outcomes absolutely show that the mainstream elective technique attains decent precision degrees in sensing scam circumstances in credit cards.

**Keywords:** Decision Tree, Random Forest and Extreme Gradient boosting algorithms.

**INTRODUCTION**

Falsification of the credit card can be defined as the unapproved use of a customer’s card data to create purchases or to dismiss funds from the cardholder's record. The misconduct extortion starts from the credit card when somebody incorrectly acquires the number printed on card or the essential records for the card to be operated [9,10]. The owner of the card, the agent by whom card is issued and even guarantor of a card might not be informed of the fraud until the record is used to create purchases. As shopping through internet-based applications and paying bills online has been come into practice, there is no longer requirement of a physical card to create purchases.

Fraud detection in online shopping systems is the hottest topic nowadays. Fraud investigators, banking systems, and electronic payment systems such as PayPal must have an efficient and complex fraud detection system to prevent fraud activities that change rapidly. According to a Cyber Source report from 2017, the present fraud loss by order channel, that is, the percentage of fraud loss in their web store was 74 percent and 49 percent in their mobile channels [1]. Based on this information, the lesson is to determine anomalies across patterns of fraud behavior that have undergone change relative to the past

The rising of E-commerce business has resulted in a gentle growth within the usage of credit cards for online transactions and purchases. With the rise in the usage of credit cards, the number of fraud cases has also been doubled. Credit card frauds are those which are done with an intention to gain money in a deceptive manner without the knowledge of the cardholder.

**LITERATURE REVIEW**

**[1]** **L. Bhavya , V. Sasidhar Reddy , U. Anjali Mohan , S. Karishma, 2020, Credit Card Fraud Detection using Classification, Unsupervised, Neural Networks Models, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 09, Issue 04 (April 2020),**

Nowadays online transactions have grown in large quantities. Among them, online credit card transactions hold a huge share. Therefore, there is much need for credit card fraud detection applications in bans and financial business. Credit card fraud purposes may be to obtain goods without paying or to obtain unauthorized funds from an account. With the demand for money credit card fraud events became common. This results in a huge loss in finances to the cardholder.

**[2] Renjith, Shini. (2018). Detection of Fraudulent Sellers in Online Marketplaces using Support Vector Machine Approach. International Journal of Engineering Trends and Technology. 57. 48-53. 10.14445/22315381/IJETT-V57P210.**

The e-commerce share in the global retail spend is showing a steady increase over the years indicating an evident shift of consumer attention from bricks and mortar to clicks in retail sector. In recent years, online marketplaces have become one of the key contributors to this growth. Fraudulent e-commerce buyers and their transactions are being studied in detail and multiple strategies to control and prevent them are discussed. Another area of fraud happening in marketplaces are on the seller side and is called merchant fraud. Goods/services offered and sold at cheap rates, but never shipped is a simple example of this type of fraud. This paper attempts to suggest a framework to detect such fraudulent sellers with the help of machine learning techniques.

**[3] Saputra, Adi & Suharjito, Suharjito. (2019). Fraud Detection using Machine Learning in e-Commerce. 10.14569/IJACSA.2019.0100943.**

The volume of internet users is increasingly causing transactions on e-commerce to increase as well. We observe the quantity of fraud on online transactions is increasing too. Fraud prevention in e-commerce shall be developed using machine learning, this work to analyze the suitable machine learning algorithm, the algorithm to be used is the Decision Tree, Naive Bayes, Random Forest, and Neural Network. Result of evaluation using confusion matrix achieve the highest accuracy of the neural network by 96 percent, random forest is 95 percent, Naïve Bayes is 95 percent, and Decision tree is 91 percent. Synthetic Minority Over-sampling Technique (SMOTE) is able to increase the average of F1-Score from 67.9 percent to 94.5 percent and the average of G-Mean from 73.5 percent to 84.6 percent.

**[4] A. K. Rai and R. K. Dwivedi, "Fraud Detection in Credit Card Data using Unsupervised Machine Learning Based Scheme," 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2020, pp. 421-426, doi: 10.1109/ICESC48915.2020.9155615.**

Development of communication technologies and e-commerce has made the credit card as the most common technique of payment for both online and regular purchases. So, security in this system is highly expected to prevent fraud transactions. Fraud transactions in credit card data transaction are increasing each year. In this direction, researchers are also trying the novel techniques to detect and prevent such frauds. However, there is always a need of some techniques that should precisely and efficiently detect these frauds. This paper proposes a scheme for detecting frauds in credit card data which uses a Neural Network (NN) based unsupervised learning technique. Proposed method outperforms the existing approaches of Auto Encoder (AE), Local Outlier Factor (LOF), Isolation Forest (IF) and K-Means clustering. Proposed NN based fraud detection method performs with 99.87% accuracy whereas existing methods AE, IF, LOF and K Means gives 97%, 98%, 98% and 99.75% accuracy respectively.

**SCOPE:**

* Our Models can be used for detecting the fraudulent transactions.
* It can be helpful for the customers without losing their information.

**EXISTING SYSTEM:**

Much research has been done on studying credit card fraud detection. In past people manually detect fraud transactions. But, the entire problem of credit card fraud detection suffers from a problem of Imbalanced data (a very highly imbalanced data). This problem requires us to heavily process the data before training any machine learning model like Random Forest etc.

**Disadvantages:**

* Low accuracy.
* Time consuming.
* High complexities.

**PROPOSED SYSTEM:**

We propose this system to investigate a problem of whether it is valuable or not to use machine learning techniques of Decision Tree, Random Forest and XGBoost has best accuracy to detect whether the credit card is fraud or not fraud using XGBoost.

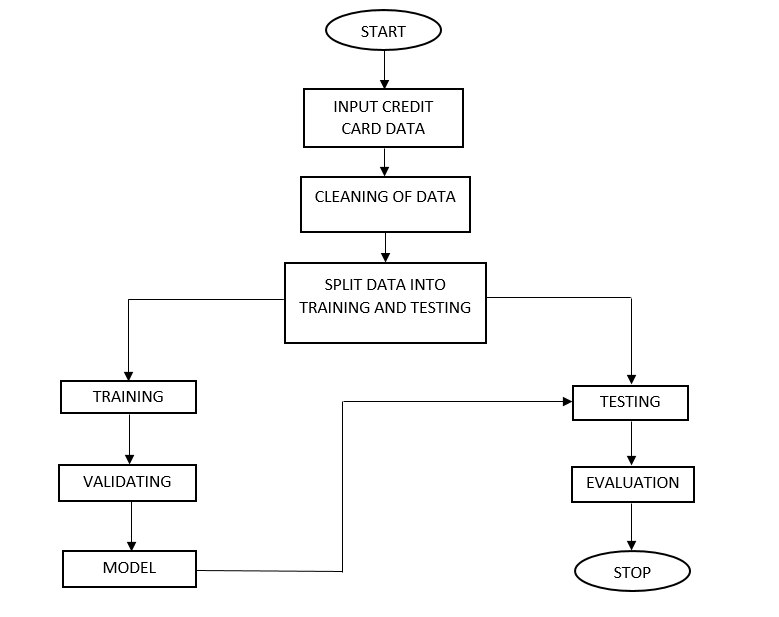
**ADVANTAGES:**

* High accuracy.
* Time Saving.
* Low complexities.
* High reliability.

**APPLICATIONS:**

* Used for preventing credit card frauds by banks.
* Financial industries employees them heavily to prevent frauds.

**Flow Chart:**



**HARDWARE & SOFTWARE REQUIREMENTS**

# HARDWARE CONFIGURATION:

# Processor - I3/Intel Processor

# RAM - 4GB (min)

# Hard Disk - 128 GB

# Key Board - Standard Windows Keyboard

# Mouse - Two or Three Button Mouse

# Monitor - LCD

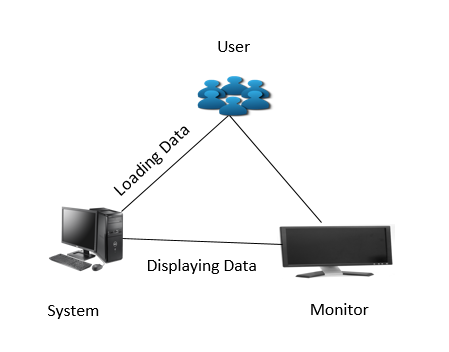
**SOFTWARE CONFIGURATION:**

* Operating System : Windows 7+
* Server side Script : Python 3.6+
* IDE : Jupyter Notebook
* Libraries Used : Pandas, Numpy, scikit-learn, Matplotlib

**LEARNING OUTCOMES:**

* About Classification in machine learning.
* About preprocessing techniques.
* About Random Forest.
* About Decision Tree.
* About XGBoost.
* Knowledge on PyCharm Editor.

**ARCHITECTURE**



**ALGORITHMS**

**K-Means Clustering:**

There is an algorithm that tries to minimize the distance of the points in a cluster with their centroid – the k-means clustering technique.

The main objective of the K-Means algorithm is to minimize the sum of distances between the points and their respective cluster centroid.

A cluster refers to a collection of data points aggregated together because of certain similarities.

You’ll define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster.

Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.

In other words, the K-means algorithm identifies k number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible.

**STEPS:**

1. Choose the number of clusters k.
2. Select k random points from the data as centroids.
3. Assign all the points to the closest cluster centroid.
4. Re-compute the centroids of newly formed clusters.
5. Repeat steps 3 and 4.

**Local Outlier Factor:**

Local outlier factor (LOF) is an algorithm that identifies the outliers present in the dataset.

LRD is inverse of the average reachability distance of A from its neighbors. Intuitively according to LRD formula, more the average reachability distance (i.e., neighbors are far from the point), less density of points are present around a particular point. This tells how far a point is from the nearest cluster of points. Low values of LRD implies that the closest cluster is far from the point.

LRD of each point is used to compare with the average LRD of its K neighbors. LOF is the ratio of the average LRD of the K neighbors of A to the LRD of A.

Intuitively, if the point is not an outlier (inlier), the ratio of average LRD of neighbors is approximately equal to the LRD of a point (because the density of a point and its neighbors are roughly equal). In that case, LOF is nearly equal to 1. On the other hand, if the point is an outlier, the LRD of a point is less than the average LRD of neighbors. Then LOF value will be high.

Generally, if LOF> 1, it is considered as an outlier, but that is not always true. Let’s say we know that we only have one outlier in the data, then we take the maximum LOF value among all the LOF values, and the point corresponding to the maximum LOF value will be considered as an outlier.

**Isolation Factor:**

It is a tree-based algorithm, built around the theory of decision trees and random forests. When presented with a dataset, the algorithm splits the data into two parts based on a random threshold value. This process continues recursively until each data point is isolated. Once the algorithm runs through the whole data, it filters the data points which took fewer steps than others to be isolated. Isolation Forest in sklearn is part of the Ensemble model class, it returns the anomaly score of each instance to measure abnormality.

In most unsupervised methods, “normal” data points are first profiled and anomalies are reported if they do not resemble that profile. Isolation forest, on the other hand, takes a different approach; it isolates anomalous data points explicitly.

It is important to mention that Isolation Forest is an unsupervised machine learning algorithm. Meaning, there is no actual “training” or “learning” involved in the process and there is no pre-determined labeling of “outlier” or “not-outlier” in the dataset. So there is no accuracy test in the conventional machine learning sense.

Isolation Forest Algorithm returns the anomaly score of each sample using the Isolation Forest algorithm.

The Isolation Forest ‘isolates’ observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature.

Since recursive partitioning can be represented by a tree structure, the number of splitting required to isolate a sample is equivalent to the path length from the root node to the terminating node.

Random partitioning produces noticeably shorter paths for anomalies.

Hence, when a forest of random trees collectively produce shorter path lengths for particular samples, they are highly likely to be anomalies.

**Auto Encoders:**

Autoencoders are a specific type of feed forward neural networks where the input is the same as the output. They compress the input into a lower-dimensional code and then reconstruct the output from this representation. The code is a compact “summary” or “compression” of the input, also called the latent-space representation.

An autoencoder consists of 3 components: encoder, code and decoder. The encoder compresses the input and produces the code, the decoder then reconstructs the input only using this code.

Autoencoders are mainly a dimensionality reduction (or compression) algorithm with a couple of important properties:

Data-specific: Autoencoders are only able to meaningfully compress data similar to what they have been trained on. Since they learn features specific for the given training data, they are different than a standard data compression algorithm like gzip. So we can’t expect an autoencoder trained on handwritten digits to compress landscape photos.

Lossy: The output of the autoencoder will not be exactly the same as the input, it will be a close but degraded representation. If you want lossless compression they are not the way to go.

Unsupervised: To train an autoencoder we don’t need to do anything fancy, just throw the raw input data at it. Autoencoders are considered an unsupervised learning technique since they don’t need explicit labels to train on. But to be more precise they are self-supervised because they generate their own labels from the training data.

**Neural Networks:**

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. Neural Networks are used for solving many business problems such as sales forecasting, customer research, data validation, and risk management. It is neurally implemented mathematical model. It contains huge number of interconnected processing elements called neurons to do all operations. Information stored in the neurons are basically the weighted linkage of neurons.

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated.

Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabeled data according to similarities among the example inputs, and they classify data when they have a labeled dataset to train on. (Neural networks can also extract features that are fed to other algorithms for clustering and classification; so you can think of deep neural networks as components of larger machine-learning applications involving algorithms for reinforcement learning, classification and regression.)

**SYSTEM DESIGN**

**UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



**COLLABORATION DIAGRAM:**

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



**DEPLOYMENT DIAGRAM**

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used to deploy the application.



**ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



**COMPONENT DIAGRAM**

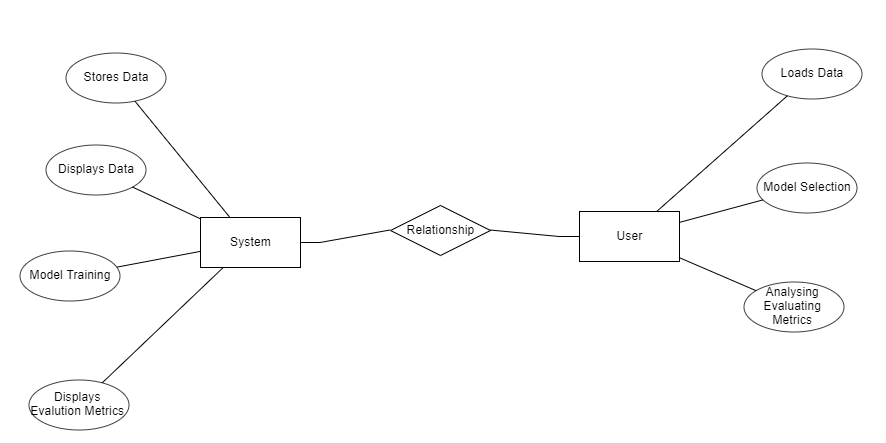
A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.



**ER DIAGRAM:**

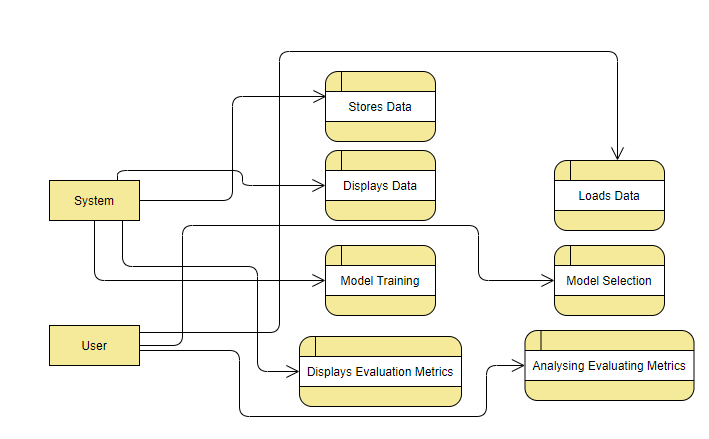
An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.



**DFD DIAGRAM:**

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.



**SYSTEM STUDY**

**FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**SYSTEM TEST**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**6.1 Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# 6.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**6.3 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**CONCLUSION:**

In this application, we have successfully created unsupervised ML models to detect whether the credit card is fraud or not fraud. We noticed that out of Local Outlier Factor, Isolation Forest, K-Means Clustering, Neural Networks and Auto Encoders Neural Networks performs well with accuracy score of 99% and precision and recall scores of 85%.

**FUTURE SCOPE:**

This system can be extended to improve the models precision and recall scores by applying imbalanced data treatment techniques to improve precision and recall scores of our Machine Learning models.

**References:**

* L. Bhavya , V. Sasidhar Reddy , U. Anjali Mohan , S. Karishma, 2020, Credit Card Fraud Detection using Classification, Unsupervised, Neural Networks Models, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 09, Issue 04 (April 2020).
* <https://towardsdatascience.com/credit-card-fraud-detection-using-machine-learning-python-5b098d4a8edc>
* <https://www.ijert.org/credit-card-fraud-detection-using-machine-learning-algorithms>
* A. K. Rai and R. K. Dwivedi, "Fraud Detection in Credit Card Data using Unsupervised Machine Learning Based Scheme," 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2020, pp. 421-426, doi: 10.1109/ICESC48915.2020.9155615.
* https://www.kaggle.com/code/gpreda/credit-card-fraud-detection-predictive-models