**ABSTRACT**

Financial fraud is a growing problem with far consequences in the financial industry and while many techniques have been discovered. The finance and banking are very important sectors in our present day generation, almost everyone has to deal with it either physically or online. Nowadays, most of the transactions are done through online. Billions of dollars of loss are caused every year by credit card transactions. In this, we proposed that fraud detection is a critical problem affecting large financial companies that have increased due to the growth in credit card transactions. This view presents detection of frauds in credit card transactions, using data mining technique of Predictive modeling. The data set contains credit card transaction of cardholders. Some of the algorithms like Decision Tree, Random Forest, SVM and logistic regression are applied on the raw and preprocessed data.The design of this efficient fraud analysis algorithm is key for reducing the losses and more algorithms rely on advanced machine learning techniques to assist fraud investigators.

## INDEX

|  |  |  |
| --- | --- | --- |
| **S.NO** | **CONTENTS** | **PAGE NO** |
| **1** | **INTRODUCTION** | **1-8** |
|  | 1.1 Introduction to Project | 1 |
|  | 1.2 Purpose of the Project | 1 |
|  | 1.3 Python | 2-5 |
|  | 1.3.1 History of Python | 3 |
|  | 1.3.2 Python Features | 3-4 |
|  | 1.3.3 Python Libraries | 4-5 |
|  | 1.4 Machine Learning | 5-7 |
|  | 1.5 Anaconda Navigator | 7-8 |
| **2** | **LITERATURE SURVEY** | **9-10** |
| **3** | **SYSTEM ANALYSIS** | **11-21** |
|  | 3.1 Existing System | 11 |
|  | 3.2 Proposed System | 11-12 |
|  | 3.3 Algorithms Used | 12-17 |
|  | 3.3.1 Logistic Regression Algorithm | 12 |
|  | 3.3.2 Support Vector Machine | 13 |
|  | 3.3.3 Local Outlier Detection Method | 13-14 |
|  | 3.3.4 Decision Tree | 14-15 |
|  | 3.3.5 K-Nearest Neighbors | 15-17 |

|  |  |  |
| --- | --- | --- |
|  | 3.4 Feasibility Study | 17-19 |
|  | 3.4.1 Economical Feasibility | 18 |
|  | 3.4.2 Technical Feasibility | 18 |
|  | 3.4.3 Operational Feasibility | 18-19 |
|  | 3.5 Requirement Specifications | 19-21 |
|  | 3.5.1 Functional Requirements | 19-20 |
|  | 3.5.2 Non-Functional Requirements | 20 |
|  | 3.5.3 Performance Requirements | 20-21 |
|  | 3.5.4 Hardware Requirements | 21 |
|  | 3.5.5 Software Requirements | 21 |
| **4** | **SYSTEM DESIGN** | **22-32** |
|  | 4.1 System Architecture | 22-23 |
|  | 4.2 Input and Output Design | 23 |
|  | 4.2.1 Input Design | 23 |
|  | 4.2.2 Output Design | 23 |
|  | 4.3 UML Diagrams | 24-32 |
|  | 4.3.1 Use Case Diagram | 25 |
|  | 4.3.2 Class Diagram | 26 |
|  | 4.3.3 Sequence Diagram | 27-28 |
|  | 4.3.4 Collaboration Diagram | 29 |
|  | 4.3.5 Component Diagram | 30 |
|  | 4.3.6 Activity Diagram | 31 |
|  | 4.3.7 Deployment Diagram | 32 |
| **5** | **IMPLEMENTATION** | **33-34** |
|  | 5.1 Modules | 33-34 |

|  |  |  |
| --- | --- | --- |
| **6** | **SCREENSHOTS** | **35-48** |
| **7** | **TESTING** | **49-51** |
|  | 7.1 Types of Testing | 49-51 |
|  | 7.1.1 Unit Testing | 49 |
|  | 7.1.2 Integration Testing | 49 |
|  | 7.1.3 Functional Testing | 50 |
|  | 7.1.4 System Testing | 50 |
|  | 7.1.5 White Box Testing | 51 |
|  | 7.1.6 Black Box Testing | 51 |
|  | 7.1.7 Acceptance Testing | 51 |
|  | 7.2 Test Cases | 51 |
| **8** | **CONCLUSION** | **52** |
| **9** | **FUTURE ENHANCEMENT** | **53-54** |
| **10** | **BIBILIOGRAPHY** | **55-56** |

### LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **S.NO** | **FIGURE NAME** | **PAGE NO** |
| **1** | 1.1 Anaconda Navigator | 8 |
| **2** | 3.1 Classification diagram | 12 |
| **3** | 3.2 Logistic Regression Algorithm | 13 |
| **4** | 3.3 Support Vector Machine Detection | 13 |
| **5** | 3.4 K-Nearest Neighbors | 15 |
| **6** | 4.1 System Architecture | 22 |
| **7** | 4.3.1 Use Case diagram | 25 |
| **8** | 4.3.2 Class diagram | 26 |
| **9** | 4.3.3 Sequence diagram | 27 |
| **10** | 4.3.4 Collaboration diagram | 29 |
| **11** | 4.3.5 Component diagram | 30 |
| **12** | 4.3.6 Activity diagram | 31 |
| **13** | 4.3.7 Deployment diagram | 32 |
| **14** | 6.1 Display dataset | 35 |
| **15** | 6.2 Description of datasets and display columns | 36 |
| **16** | 6.3 Detection of Fraud and Non-Fraud Subplots | 37 |
| **17** | 6.4 Subplots | 38 |
| **18** | 6.5 Shuffling of data | 39 |

|  |  |  |
| --- | --- | --- |
| **19** | 6.6 Distribution of Classes | 40 |
| **20** | 6.7 Correlation Matrix | 41 |
| **21** | 6.8 Plotting using different factors | 42 |
| **22** | 6.9 Plot using different columns | 43 |
| **23** | 6.10 Fraud and Non-Fraud Transactions | 44 |
| **24** | 6.11 Time Accuracy | 45 |
| **25** | 6.12 Accuracy score for each classifier | 46 |
| **26**  **27** | * 1. K-means clustering   2. Accuracy Predictions | 47  48 |

**CHAPTER 1 INTRODUCTION**

**1.INTRODUCTION**

#### INTRODUCTION TO PROJECT:

Today, all around the world data is available very easily, from small to big organizations are storing information that has high volume, variety, speed and worth. Early analysis of big data was centered primarily on data volume, for example, general public database, biometrics, financial analysis.

For frauds, the credit card is an easy and friendly target because without any risk a significant amount of money is obtained within a short period. To commit credit card fraud, fraudsters try to steal sensitive information such as credit card number, bank account and social security number. Fraudsters try to make every fraudulent transaction legitimate which makes fraud detection a challenging problem. Increased credit card transactions show that approximately 70% of the people in the US can fall into the trap of these fraudsters.

Credit Card dataset is highly imbalanced dataset because it carries more legitimate transactions as compared to the fraudulent one. That means prediction will get very high accuracy score without detecting a fraud transaction. To handle this kind of problem one better way is to class distribution, i.e., sampling minority classes. In sampling minority, class training example can be increased in proportion to the majority class to raise the chance of correct prediction by the algorithm.

All machine learning algorithms are evaluated using a real world credit card transaction to identify fraud or non-fraud transaction. The main motive of this project is to apply supervised learning method on the real-world dataset.

#### PURPOSE OF THE PROJECT

Previously, in existing system NIDS and HIDS are implemented for identifying existing intrusions based on the id. If there is any new dynamic intrusion occurred in NIDS and HIDS, they didn’t detect the intrusions. So, this is the major drawback in existing system.

#### PYTHON

Machine learning is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of computer programs that can change when exposed to new data.

In this article, we’ll see basics of Machine Learning, and implementation of a simple machine learning algorithm using python. Python community has developed many modules to help programmers implement machine learning. Machine learning involves computer to get trained using a given data set, and use this training to predict the properties of a given new data.

For example, we can train computer by feeding it 1000 images of cats and 1000 more images which are not of a cat, and tell each time to computer whether a picture is cat or not. Then if we show the computer a new image, then from the above training, computer should be able to tell whether this new image is cat or not.

Process of training and prediction involves use of specialized algorithms. We feed the training data to an algorithm, and the algorithm uses this training data to give predictions on a new test data. One such algorithm is K-Nearest Neighbor classification (KNN classification). It takes a test data, and finds k nearest data values to this data from test data set. Then it selects the neighbor of maximum frequency and gives its properties as the prediction result.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

* **Python is Interpreted:** Python is processed at runtime by the interpreter. You need not to compile your program before executing it. This is similar to PERL and PHP.
  + **Python is Interactive:** You can actually sit at a Python prompt and interact

with the interpreter directly to write your programs.

* + **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
  + **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

#### History of Python:

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

#### Python Features

Python's features include :

* **Easy-to-learn :** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read :** Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain :** Python's source code is fairly easy-to-maintain.
* **A broad standard library :** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode :** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable :** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable :** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases :** Python provides interfaces to all major commercial databases.
* **GUI Programming :** Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable :** Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* It supports automatic garbage collection.
* It can be easily integrated with C, C++, ActiveX, CORBA, and Java.

### Python Libraries:

1. **pandas**: pandas package is the most important tool at the disposal of Data Scientists and Analysts working in Python today. The powerful machine learning and glamorous visualization tools may get all the attention, but pandas is the backbone of most data projects.
2. **matplotlib:** matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.
3. **seaborn:** seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
4. **numPy:** numPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements.
5. **sklearn:** Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python and has caused many scientists to switch from ruby to python.
6. **tensorflow:** tensorflow is a free and open-source library and differentiable programming across a range of tasks.It is asymbolic math library, and is also used for machine learning applications such as neural networks.

#### MACHINE LEARNING:

Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. It is closely related to computational statistics, which focuses on making predictions using computers. Machine learning algorithms build a mathematical model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to perform the task across business problems, machine learning is also referred to as predictive analytics.

#### History of ML:

Arthur Samuel, is an American pioneer in the field of computer gaming and artificial intelligence, coined the term "Machine Learning" in 1959 while at IBM. A representative book of the machine learning research during the 1960s was the Nilsson's book on Learning Machines, dealing mostly with machine learning for pattern classification. The interest of machine learning related to pattern recognition continued during the 1970s, as described in the book of Duda and Hart in 1973. In 1981 a report was given on using teaching strategies so that a neural network learns to recognize 40 characters (26 letters, 10 digits, and 4 special symbols) from a computer terminal.

#### Types of ML Algorithms:

Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop a conventional algorithm for effectively performing the task.

#### Supervised learning:

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task. In weakly supervised learning, the training labels are noisy, limited, or imprecise, however, these labels are often cheaper to obtain, resulting in larger effective training sets.

#### Unsupervised learning:

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, though unsupervised learning encompasses other domains involving summarizing and explaining data features.

#### Semi-supervised learning:

Semi-supervised learning falls between unsupervised learning(without any labeled training data) and supervised learning(with completely labeled training data). Many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

#### Reinforcement learning:

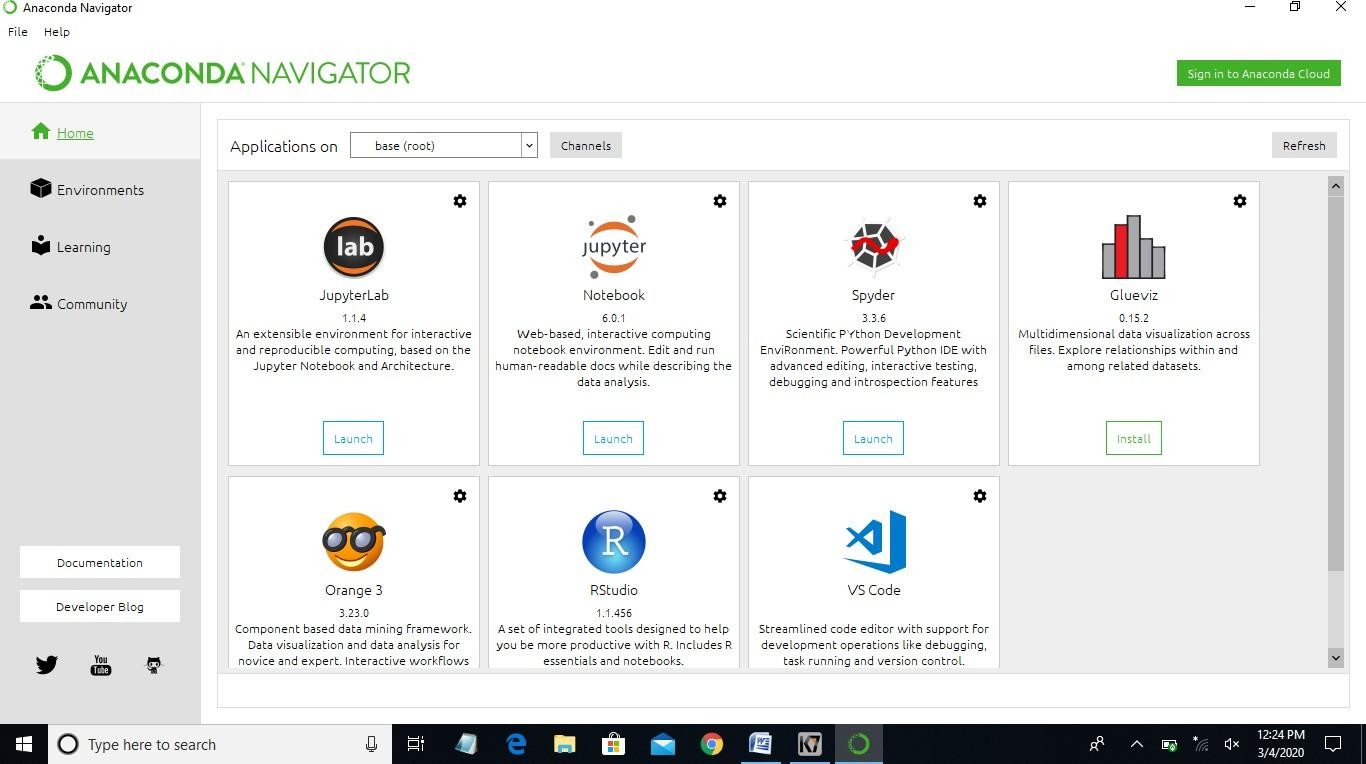
Reinforcement learning can be thought of as a hit and trial method of learning. The machine gets a Reward or Penalty point for each action it performs. If the option is correct, the machine gains the reward point or gets a penalty point in case of a wrong response. In machine learning, the environment is typically represented as the Markov Decision Process (MDP). Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible. Reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against a human opponent.Many reinforcement learning algorithm use dynamic programming techniques.

#### ANACONDA NAVIGATOR:

Anaconda Navigator is a desktop Graphical User Interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage Anaconda packages, environments and channels without using command- line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository.

In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages, and use multiple environments to separate these different versions. The command line program anaconda is both a package manager and an environment manager, to help data scientists ensure that each version of each package has all the dependencies it requires and works correctly.

Anaconda Navigator is an easy, point-and-click way to work with packages and environments without required to type anaconda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages and update them, all inside Navigator.



**Fig 1.1:** Anaconda Navigator

# CHAPTER 2 LITERATURE SURVEY

### LITERATURE SURVEY

In [1] this paper represents an research about a case study involving credit card fraud detection, where data normalization is applied before Cluster Analysis and with results obtained from the use of Cluster Analysis and Artificial Neural Networks on fraud detection has shown that by clustering attributes neuronal inputs can be minimized. And promising results can be obtained by using normalized data and data should be MLP trained. This research was based on unsupervised learning. Significance of this paper was to find new methods for fraud detection and to increase the accuracy of results.

In [2] this paper a new collative comparison measure that reasonably represents the gains and losses due to fraud detection is proposed. A cost sensitive method which is based on Bayes minimum risk is presented using the proposed cost measure. Improvements up to 23% is obtained when this method and other state of art algorithms are compared. The data set for this paper is based on real life transactional data by a large European company and personal details in data is kept confidential, accuracy of an algorithm is around 50%. Significance of this paper was to find an algorithm and to reduce the cost measure. The result obtained was by 23% and the algorithm they find was Bayes minimum risk.

Various modern techniques based on Sequence Alignment, Machine learning, Artificial Intelligence, Genetic Programming, Data mining etc., has been evolved and is still evolving to detect fraudulent transactions in credit card. A sound and clear understanding on all these approaches is needed that will certainly lead to an efficient credit card fraud detection system. Survey of various techniques used in credit card fraud detection mechanisms has been shown in this paper along with evaluation of each methodology based on certain design criteria. Analysis on Credit Card Fraud Detection Methods has been done. The survey in this paper was purely based to detect the efficiency and transparency of each method. Significance of this paper was conduct a survey to compare different credit card fraud detection algorithm to find the most suitable algorithm to solve the problem.

A comparison has been made between models based on artificial intelligence along with general description of the developed fraud detection system are given in this paper such as the Naive Bayesian Classiﬁer and the model based on Bayesian Networks, the clustering model. And in the end conclusions about results of models’ evaluative testing are made. Number of legal truncations was determined greater or equal to 0.65 that is their accuracy was 65% using Bayesian Network. Significance of this paper is to compare between models based on artificial intelligence along with general description of the developed system and to state the accuracy of each model along with the recommendation to make the better model.

Nutan and Suman on review on credit card fraud detection they have supported the theory of what is credit card fraud, types of fraud like telecommunication, bankruptcy fraud etc. and how to detect it, in addition to it they have explained numerous algorithms and methods on how to detect fraud using Bayesian, networks, Hidden Markova model, Decision Tree and more. They have explained in detail about each algorithm and how this algorithm works along with mathematical explanation. Types of machine learning along with classifications has been studied. Pros and cons of each method is listed.

The main aim of the research is to detect the credit card fraud in the dataset obtained from ULB by applying Logistic regression, Decision tree, SVM,K- Nearest Neighbors,and to evaluate their accuracy, precision using different models and compare and collate them to state the best possible model to solve the credit card fraud detection problem

# CHAPTER 3 SYSTEM ANALYSIS

### SYSTEM ANALYSIS

#### EXISTING SYSTEM:

Network Intrusion Detection System (NIDS) and Host Intrusion Detection System (HIDS). Network Intrusion Detection system is a mechanism that is used within the network to identify the malicious event. The network traffic is monitored in the network that is in the sub- net. If an attack is observed it matches the traffic with the known attack list. Then an alert is passed to the administrator. NIDS is installed in router to identify the passage of network traffic.

HIDS runs on an individual system. The functions of two IDSs are the same. HIDS also monitors the unauthorized activity. It takes a short review of the existing files in the system. Then it matches it with the old system files. If it finds an intrusion or changes in the system, then an alert is passed to the administrator. The intrusion can be detected as if a file is modified or deleted, that it means malicious activity is reported.

#### Disadvantages in Existing System:

* + - NIDS can only alarm if the traffic matches the predefined rules.
    - NIDS cannot determine if the attack was successful.
    - NIDS cannot examine the traffic that is encrypted.
    - Switched networks require special configurations so that the NIDS can monitor all the traffic.

#### PROPOSED SYSTEM:

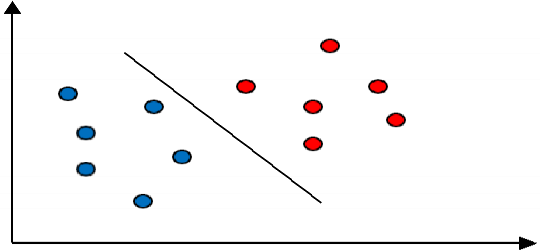
Previously, in existing system NIDS and HIDS are implemented for identifying existing intrusions based on the id. If there is any new dynamic intrusion occurs NIDS and HIDS they didn’t detect the intrusions. So this is the major drawback in existing system.

There are many advantages compared to the existing system such as:

#### ADVANTAGES IN PROPOSED SYSTEM:

* + - Logistic Regression is a technique that has been emerged for the analysis of data for the classification process.
    - The Logistic Classifier uses a set of training data where each one has been labeled into one of two categories.
    - The training data set builds a model and the new unknown data would be categorized into the proper group.

With the help of this line the data can be easily separated with more accuracy. In this technique, two categories are available, we can either classify the data to one class or to the other class depending upon the behavior of the new data. In addition to linear classification, they can perform non-linear classification.



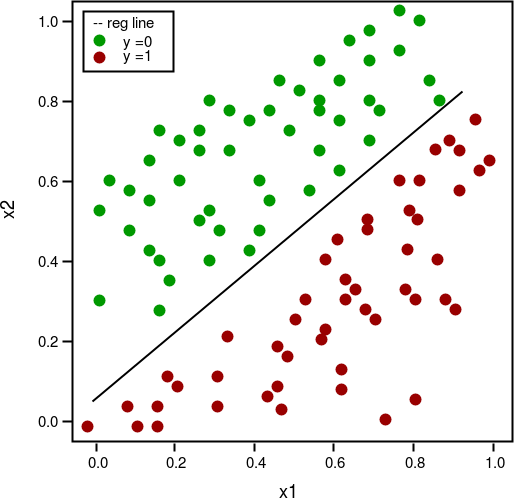
#### Fig 3.1

Also, in non-linear classification, data is not labeled so supervised learning is not possible. In such a context, unsupervised learning approach is implemented, which attempts to cluster the data with similar behavior.

#### ALGORITHMS USED:

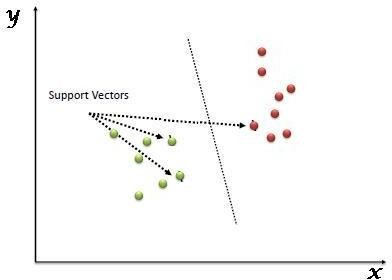
* + 1. **LOGISTIC REGRESSION ALGORITHM:**

Logistic regression becomes a classification technique only when a decision threshold is brought into the picture. The setting of the threshold value is a very important aspect of Logistic regression and is dependent on the classification problem itself. The decision for the value of the threshold value is majorly affected by the values of [precision and recall.](https://www.geeksforgeeks.org/confusion-matrix-machine-learning/)



#### Fig 3.2

* + 1. **SUPPORT VECTOR MACHINE DETECTION:**

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot).

#### Fig 3.3

Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane line).

#### LOCAL OUTLIER FACTOR DETECTION:

Outlier detection (also known as anomaly detection) is the process of finding data objects with behaviour that are very different from expectation. Such objects are called outliers or anomalies.

The most interesting objects are those, that deviates significantly from the normal object. Outliers are not being generated by the same mechanism as rest of the data. Outlier detection is important in many applications, such as:

* + - * Intrusions in communication networks
      * Fraud in financial data
      * Fake news and misinformation
      * Healthcare analysis
      * Security and surveillance

Outlier detection and clustering analysis are two highly related tasks. Clustering finds the majority of patterns in a data set and organizes the data accordingly, whereas outlier detection tries to capture those exceptional cases that deviate substantially from the majority of patterns.

#### DECISION TREE:

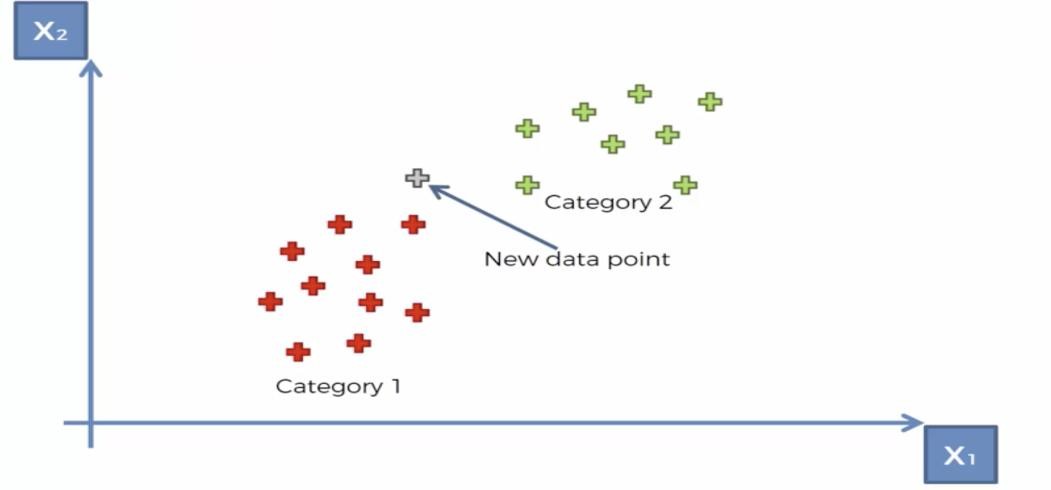
A decision tree is a decision support tool that uses a treelike graph or model of decisions and their possible outcomes, comprising chance event outcomes, resource costs, and utility. It is one way to display an algorithm. Decision trees are generally used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach an objective, but are also a popular tool for machine learning. A decision tree is a flowchart-like structure in which each internal node depicts a "test" on an attribute, each branch represents the outcome of the test and each leaf node accounts for a class label. The path from the root to leaf depicts classification rules. The basic algorithm for decision tree induction is a greedy algorithm that builds decision trees in a top-down recursive divide-and-conquer manner. The algorithm starts with the entire set of rows in the Training set, selects the best attribute that yields maximum information for classification, and originates a test node for this attribute. Then, top-down induction of decision trees divides the current set of tuples according to their values of the current test attribute. Classifier generation stops, if all tuples in a subset pertain to the same class, or if it is not worth to proceed with an additional separation into further subsets.

Decision tree helps to determine the worst, best and expected values for different scenarios, simplified to understand and interpret and allows addition of new possible scenarios.

#### Process for constructing a decision tree are:

Firstly to calculate the entropy of every attribute using the dataset in problem then dataset is divided into subsets using the attribute for which gain is maximum or entropy is minimum after that to make a decision tree node containing that attribute and lastly recursion is performed on subsets using remaining attributes to create a decision tree.

#### K-Nearest Neighbors:



**Fig 3.4**

K-Nearest Neighbors (KNN) is one of the simplest algorithms used in Machine Learning for regression and classification problem. KNN algorithms use data and classify new data points based on similarity measures (e.g. distance function). Classification is done by a majority vote to its neighbors. The data is assigned to the class which has the nearest neighbors. As you increase the number of nearest neighbors, the value of k, accuracy might increase.

#### Making Predictions with KNN:

KNN makes predictions using the training dataset directly.

* + - * Predictions are made for a new instance (x) by searching through the entire training set for the K most similar instances (the neighbors) and summarizing the output variable for those K instances. For regression this might be the mean output variable, in classification this might be the mode (or most common) class value.
      * To determine which of the K instances in the training dataset are most similar to a new input a distance measure is used. For real-valued input variables, the most popular distance measure is Euclidean distance.
      * Euclidean distance is calculated as the square root of the sum of the squared differences between a new point (x) and an existing point (xi) across all input attributes j.

Euclidean Distance(x, xi) = sqrt( sum( (xj – xij)^2 ) ) Other popular distance measures include:

* **Hamming Distance:** Calculate the distance between binary vectors (more).
* **Manhattan Distance:** Calculate the distance between real vectors using the sum of their absolute difference. Also called City Block Distance (more).
* **Minkowski Distance**: Generalization of Euclidean and Manhattan distance (more).

There are many other distance measures that can be used, such as Tanimoto, Jaccard, Mahalanobis and cosine distance. You can choose the best distance metric based on the properties of your data. If you are unsure, you can experiment with different distance metrics and different values of K together and see which mix results in the most accurate models.

#### KNN for Regression

When KNN is used for regression problems the prediction is based on the mean or the median of the K-most similar instances.

#### KNN for Classification

* + When KNN is used for classification, the output can be calculated as the class with the highest frequency from the K-most similar instances. Each instance in essence votes for their class and the class with the most votes is taken as the prediction.
  + Class probabilities can be calculated as the normalized frequency of samples that belong to each class in the set of K most similar instances for a new data instance. For example, in a binary classification problem (class is 0 or 1):

p(class=0) = count(class=0) / (count(class=0)+count(class=1))

If you are using K and you have an even number of classes (e.g. 2) it is a good idea to choose a K value with an odd number to avoid a tie. And the inverse, use an even number for K when you have an odd number of classes.

#### FEASIBILITY STUDY

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system.

All systems are feasible, if they are unlimited resources and infinite time. There are aspects in feasibility study portion of the preliminary investigation.

#### Three key considerations involved in the feasibility analysis are:

* + - Economical feasibility
    - Technical feasibility
    - Operational feasibility

#### ECONOMICAL FEASIBILITY

It refers to the benefits or Outcomes we are deriving from the product as compared to the total cost we are spending for developing the product. If the benefits are more or less the same as the older system, then it is not feasible to develop the product.

In the present system, the development of new product greatly enhances the accuracy of the system and cuts short the delay in the processing this application. The errors can be greatly reduced and at the same time providing a great level of security.

No need for spending money on client for maintenance because the database used is web enabled database.

#### TECHNICAL FEASIBILITY

In the feasibility study, first step is that the organization or company has to decide that what technologies are suitable to develop by considering the existing system. Here in this application we have used the technology is Anaconda(Jupyter Notebook).This is the free software that would be downloaded from web. Anaconda(Jupyter Notebook) it is a tool.

#### OPERATIONAL FEASIBILITY

An application not only requires economic and technical sense,but also needs operational sense.

To operate this application, the user doesn’t require thorough knowledge about Anaconda in developing this project.As this application provide a rich user interface,it is easy to operate in a flexible manner.

|  |  |
| --- | --- |
| **Operation Issues** | **Support Issues** |
| What tools are needed to support operations?  What skills will operators need to be trained in?  What processes need to be created and/or updated?  What documentation does operations need? | What documentation will users be given?  What training will users be given? How will change requests be managed? |

Very often you will need to improve the existing operations, maintenance, and support infrastructure to support the operation of the new application that you intend to develop. To determine what the impact will be you will need to have to understand both the current operations and support infrastructure of your organization and the operations and support characteristics of your new application.

#### DEVELOPERS RESPONSIBILITIES OVERVIEW:

The developer is responsible for:

* + - * Developing the system, which meets the SRS and solving all the requirements of the system training the system and installing the system at client's location after the acceptance testing is successful.
      * Submitting the required user manual describing the system interfaces to work on it and also the documents of the system.
      * Conducting any user training that might be needed for using the system.
      * Maintaining the system for a period of one year after installation.

#### REQUIREMENT SPECIFICATIONS

* + 1. **FUNCTIONAL REQUIREMENTS:**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the

results for later consultation. The various types of outputs in general are:

* + - External Outputs, whose destination is outside the organization.
    - Internal Outputs, whose destination is within the organization and they are the user’s main interface with the computer.
    - Operational outputs, whose use is purely within the computer department.
    - Understanding user’s preferences, expertise level and his business requirements through a friendly questionnaire.
    - Input data can be in four different forms - Relational DB, text files, .xls and xml files.For testing and demo you can choose data from any domain.User-B can provide business data as input.

#### NON-FUNCTIONAL REQUIREMENTS:

* + - * Secure access of confidential data (user’s details). SSL can be used.
      * Better component design to get better performance at peak time.
      * Flexible service based architecture will be highly desirable for future extension.

#### PERFORMANCE REQUIREMENTS:

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specifications for any system can be broadly stated as given below:

* + - The system should be able to interface with the existing system.
    - The system should be accurate.
    - The system be better than the existing system.
    - The existing system is completely dependent on the user to perform all the duties.

#### HARDWARE REQUIREMENTS:

* + - * **Hard Disk :** 500GB minimum
      * **Processor :** i3 or higher
      * **RAM :** 4GB or higher

#### SOFTWARE REQUIREMENTS:

* + - * **Operating System :** Windows
      * **Technology :** Python
      * **Browser :** IE or Chrome
      * **Web Application :** Jupyter Notebook

Setup tools and pip to be installed for python 3.6.x and above.

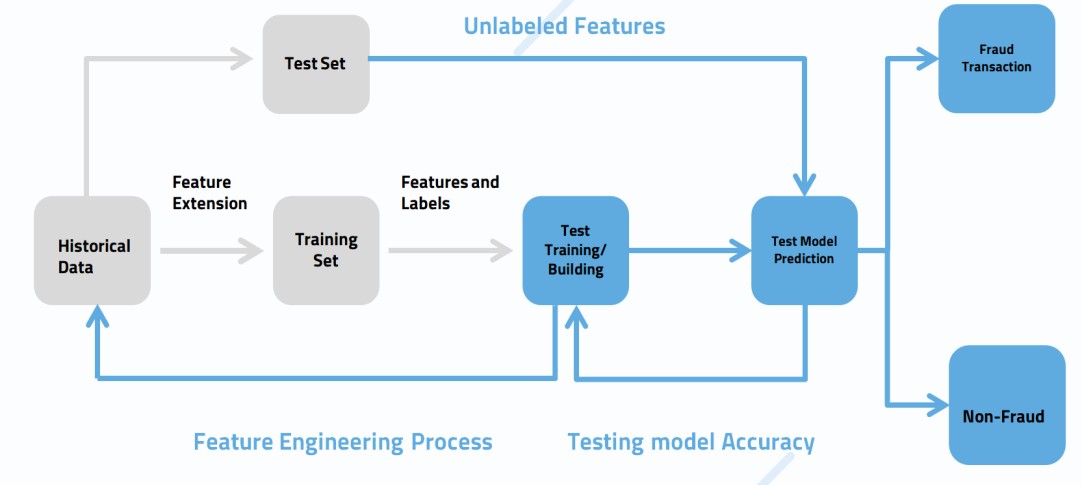
# CHAPTER 4 SYSTEM DESIGN

### SYSTEM DESIGN

System design is the process of defining elements of a system like modules, architecture, components and their interface and data for a system based on the specified requirements.

It is a separable component, frequently one that is interchangeable with others, for assembly into units of differing size, complexity or function.

#### SYSTEM ARCHITECTURE:



**Fig 4.1: System Architecture**

Credit card fraud analysis requires historical data to analyse whether the transaction is fraud or non fraud. At first the collection of large database with historical data is required. Next step is to send this data to the training set for the feature extension. Another test set includes in this feature engineering process.This test set data along with unlabeled feature is sent to the test model prediction. Another training set with the labels sent to test training and building based on the historical data.

After completion of test training and test building considering the test model accuracy.It will sent to the test model prediction.Here the keyrole prediction will be done based on the data that occurred from the previous classifications after undergoing with test sets.Here the data accurately divided into two sectors in prediction phase.That is fraud transaction and another next fraud transaction we get accurate result with this architecture,which made the credit card analysis more powerful.

#### INPUT AND OUTPUT DESIGN:

* + 1. **INPUT DESIGN:**

Input design is a part of overall system design. The main objective during the input design is as given below:

* + - * To produce a cost-effective method of input.
      * To achieve the highest possible level of accuracy
      * To ensure that the input is acceptable and understood by the user.

#### OUTPUT DESIGN:

Outputs from computer systems are required primarily to communicate the results of processing to users.

* + - * They are also used to provide a permanent copy of the results for later consultation.
      * The outputs were needed to be generated as a hard copy and as well as queries to be viewed on the screen.
      * Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing.
      * The standard printer is to be used as output media for hard copies.

#### UML DIAGRAMS:

UML stands for Unified Modeling Language.UML is a standardised 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 the 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 object oriented software and the software development process.The UML uses mostly graphical notations to express the design of software projects.

#### Goals of UML:

The goals of UML are as follows:

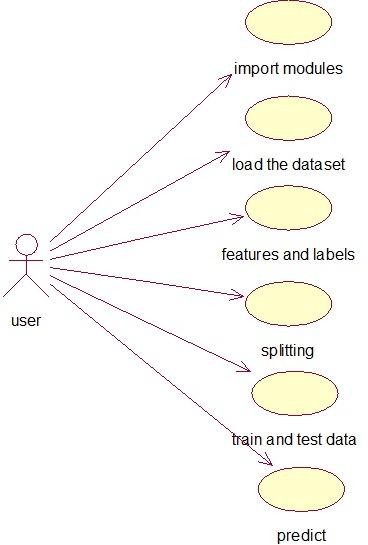
* Be independent of particular programming languages and development processes.
* Provide a formal basis for understanding the modeling language.
* Encourage the growth of the object oriented tools market.
* Support some of the main higher-level development concepts such as collaborations,frameworks,patterns and components.
* Integrate best practices.

#### USE CASE DIAGRAM:

Use Case Diagram describes the functionality of a system and users of the system. Use cases are used during requirements elicitation and analysis to represent the functionality of the system.

These diagrams contain the following elements:

* Actors, which represent users of a system, including human uses and other systems.
* Use cases, which represent functionality or services provided by a system to user. Use case diagrams address the static use case view of a system. These Diagrams are especially important in organizing and modeling the behavior of a system.

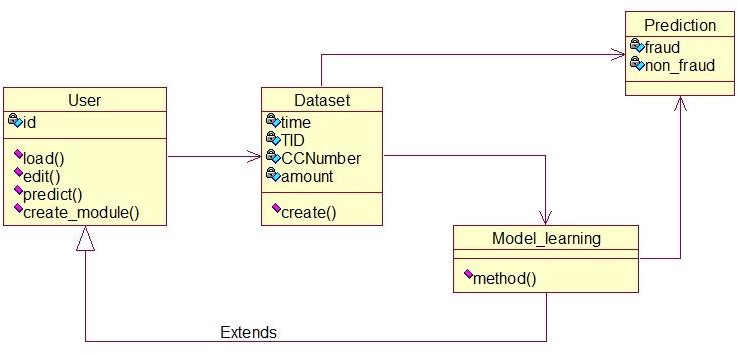


#### Fig 4.3.1: Use case diagram

* + 1. **CLASS DIAGRAM:**

Class Diagram describes the static structure of a system, or how it is structured rather than how it behaves. These diagrams contain the following elements:

* + - * Classes, which represents entities with common characteristics or features. These features include attributes, operation and associations.
      * Associations, which represent relationship that relate two or more other classes where the relationships have common characteristics or features, attributes and operations.



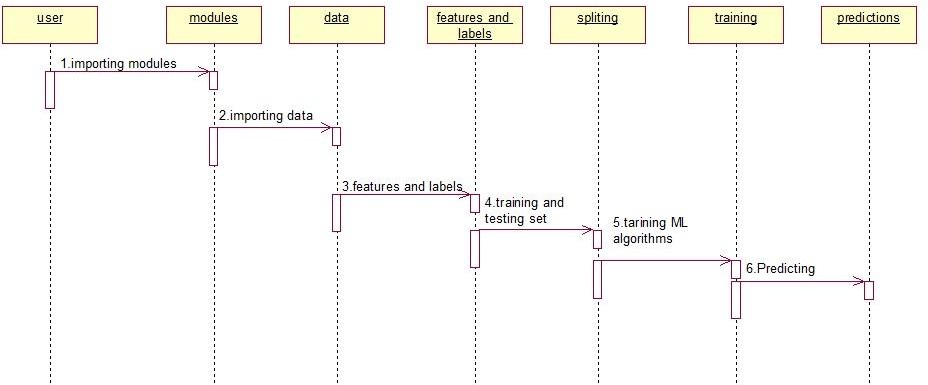
#### Fig 4.3.2: Class Diagram

Classes are represented with boxes that contain three compartments:

* The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
* The middle compartment contains the attributes of the class. They are left- aligned and the first letter is lowercase.
* The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

#### SEQUENCE DIAGRAM:

An interaction diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. Interaction diagrams address the dynamic view of the system. A sequence diagram is an interaction diagram that emphasizes the time ordering of messages.



#### Fig 4.3.3: Sequence diagram

**Elements in sequence diagram:**

The elements used in the sequence diagram are as follows:

* + - * **Actors:** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram.

We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.

* + - * **Lifelines:** A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram. The standard in UML for naming a lifeline follows the following format – Instance Name : Class Name

We display a lifeline in a rectangle called head with its name and type. The head is located on top of a vertical dashed line (referred to as the stem) as shown above. If we want to model an unnamed instance, we follow the same pattern except now the portion of lifeline’s name is left blank.

* + - * **Messages:** Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

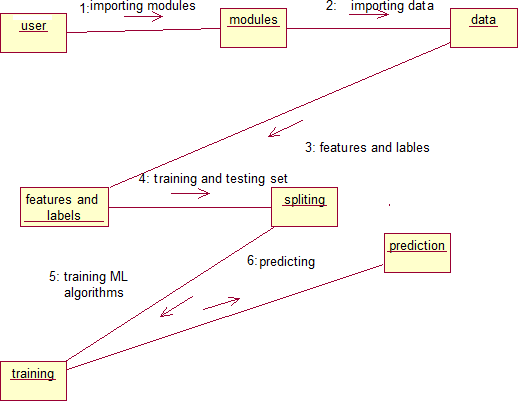
#### Uses of sequence diagrams:

Some of the uses of sequence diagram are:

* + - * Used to model and visualise the logic behind a sophisticated function, operation or procedure.
      * They are also used to show details of UML use case diagrams.
      * Used to understand the detailed functionality of current or future systems.
      * Visualise how messages and tasks move between objects or components in a system.

#### COLLABORATION DIAGRAM:

A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language. These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object.



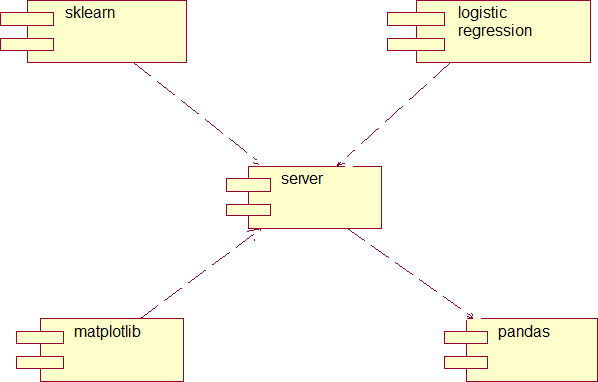
#### Fig 4.3.4 :Collaboration diagram

1. **Objects:** Objects are shown as rectangles with naming labels inside. The naming label follows the convention of object name.Class name. If an object has a property or state that specifically influences the collaboration, this should also be noted.
2. **Actors:** Actors are instances that invoke the interaction in the diagram. Each actor has a name and a role, with one actor initiating the entire use case.
3. **Links:** Links connect objects with actors and are depicted using a solid line between two elements. Each link is an instance where messages can be sent.
4. **Messages:** Messages between objects are shown as a labeled arrow placed near a link. These messages are communications between objects that convey information about the activity and can include the sequence number.

#### COMPONENT DIAGRAM:

A component provides the set of required interfaces that a component realizes or implements. These are the static diagrams of the unified modeling language. Component diagrams are used to represent the working and behavior of various components of a system.

A component diagram displays the structural relationship of components of a software system. These are mostly used when working with complex systems with many components. Components communicate with each other using interfaces. The interfaces are linked using connectors.



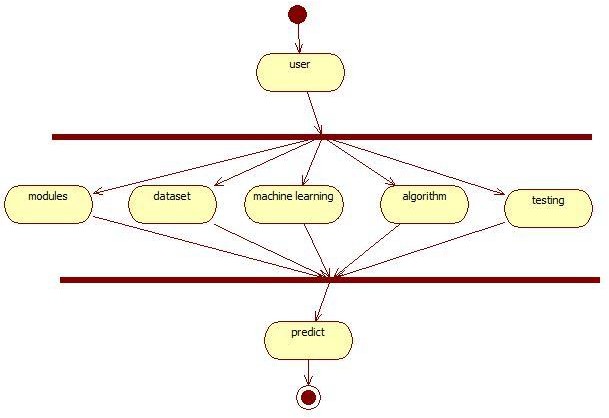
#### Fig 4.3.5 :Component diagram

**Purpose:**

The purpose of component diagrams are:

* Visualize the components of a system.
* Construct executables by using forward and reverse engineering.
* Describe the organization and relationships of the components.

#### ACTIVITY DIAGRAM:

Activity Diagram displays a special state diagram where most of the states are action states and most of the transitions are triggered by completion of the actions in the source states. This diagram focuses on flows driven by internal processing.

#### Fig 4.3.6 :Activity diagram

**NOTATIONS:**

The notations used in the activity diagram are:

**Initial State:** The starting state before an activity takes place is depicted using the initial state.

**Action Flow or Control flows:** Action flows or Control flows are also referred to as paths and edges. They are used to show the transition from one activity state to another.

**Fork:** Fork nodes are used to support concurrent activities.

**Join:** Join nodes are used to support concurrent activities converging into one. For join notations we have two or more incoming edges and one outgoing edge.

**Final State or End State:** The state which the system reaches when a particular process or activity ends is known as a Final State or End State. We use a filled circle within a circle notation to represent the final state in a state machine diagram. A system or a process can have multiple final states.

#### DEPLOYMENT DIAGRAM:

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

**import module**

**features&labels**

**train&test**

**predict**

**system**

**classification**

**prepaire data sets**

#### Fig 4.3.7 :Deployment diagram

**Notations:**

**Node:** A node, represented as a cube, is a physical entity that executes one or more components, subsystems or executables. A node could be a hardware or software element.

**Association:** This is represented by a solid line between two nodes. It shows the path of communication between nodes.

#### Uses:

* To model the hardware topology of a system.
* To model the embedded system.
* To model the hardware details for a client/server system.
* To model the hardware details of a distributed application.
* For Forward and Reverse engineering.

# CHAPTER 5 IMPLEMENTATION

### IMPLEMENTATION

#### 5.1 MODULES:

1. Cleaning the Data.
2. Loading the Data.
3. Analysing the Data.
4. Applying Models.

#### Cleaning the Data:

* First we will perform the cleaning operation on the dataset in order to check for missing values.
* After cleaning the data, we are loading the dataset.

#### Loading the Data:

We are loading the Credit Card Fraud Detection Original dataset into Jupyter Notebook in .CSV (Comma Separated Values) format.

#### Analysing the Data:

* The dataset has been obtained from the “UCI ML’ repo, it has 569 instances and 32 attributes.
* The output is either Benign (357 observations) or malignant (212 observations).
* The most influential variables are Diagnosis, Radius\_mean, texture\_mean, perimeter\_mean, area\_mean etc., the positive class is used for benign cases and the negative class is used for malignant cases.

#### Applying Models:

* After analysing the data, we are applying machine learning algorithms on the given dataset and calculating the performance metrics for each algorithm.
* Here, we are applying the LR,DT, SVM & KNN algorithms.
* Then we are choosing the best algorithm which has high accuracy for credit card fraud detection.

# CHAPTER 6 SCREENSHOTS

### SCREENSHOTS

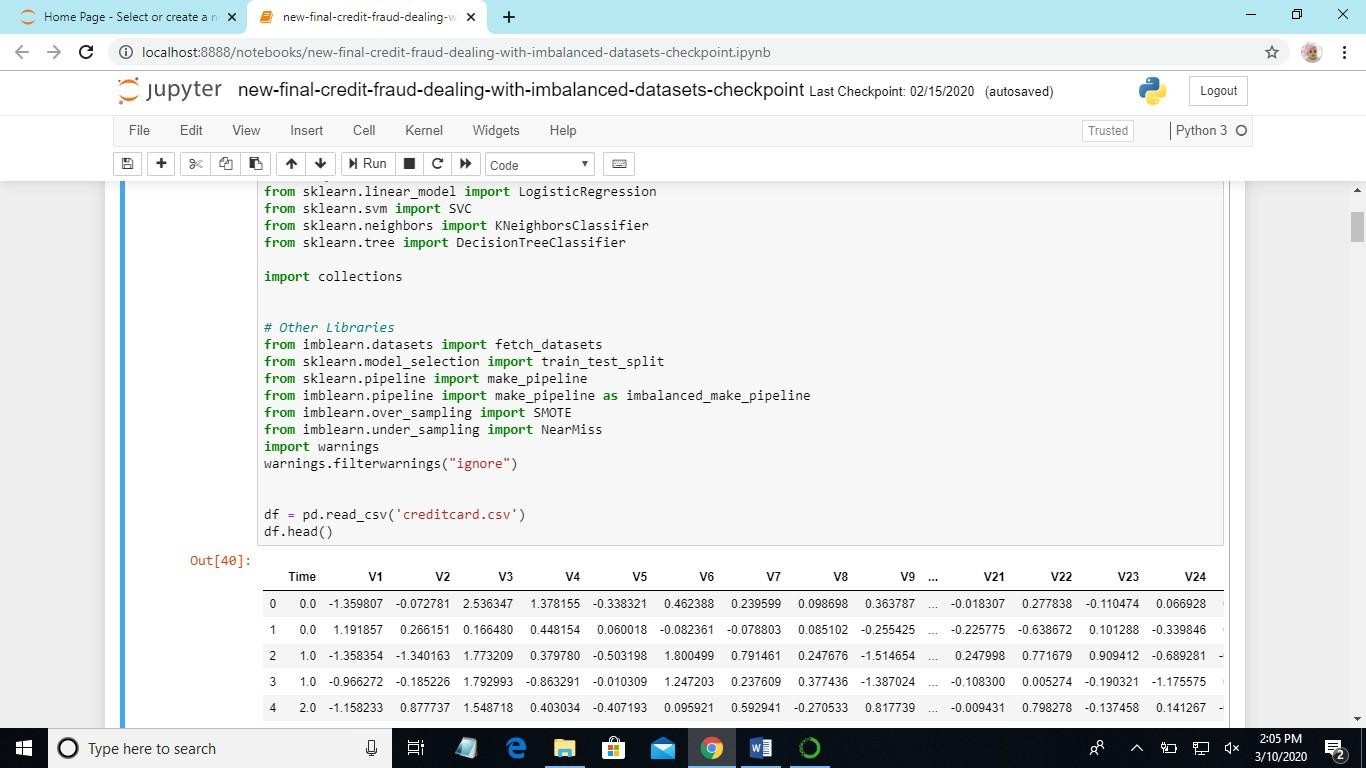


Fig 6.1 :Display dataset

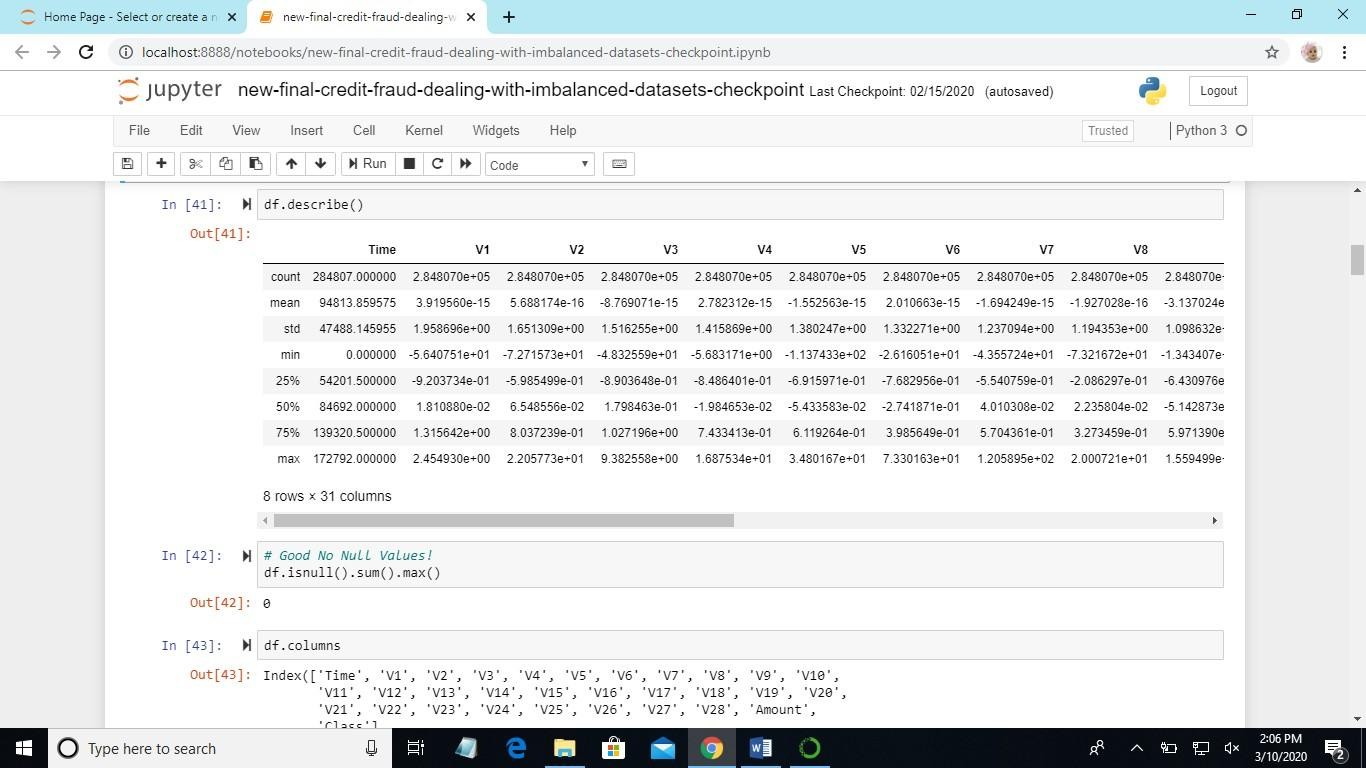


Fig 6.2 :Description of dataset and display columns

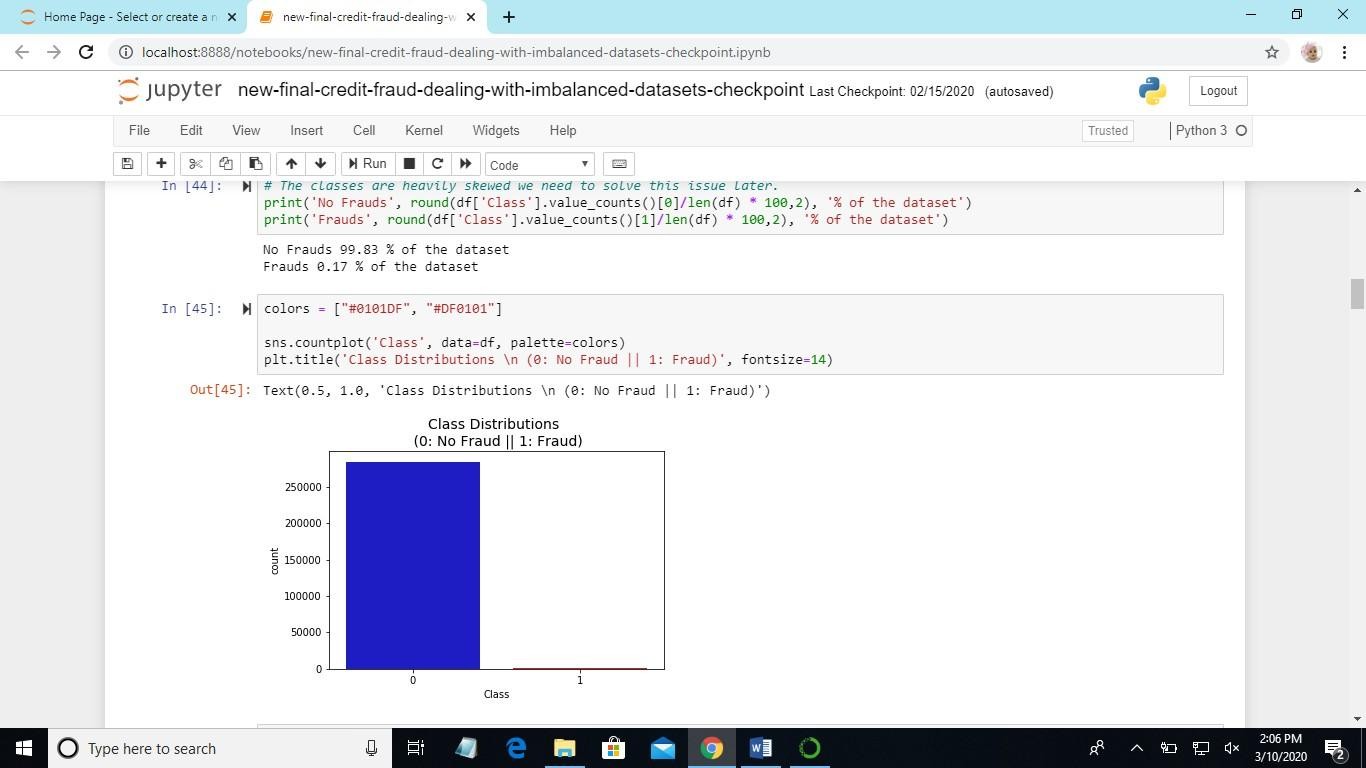


Fig 6.3 :Detection of Fraud and Non-Fraud

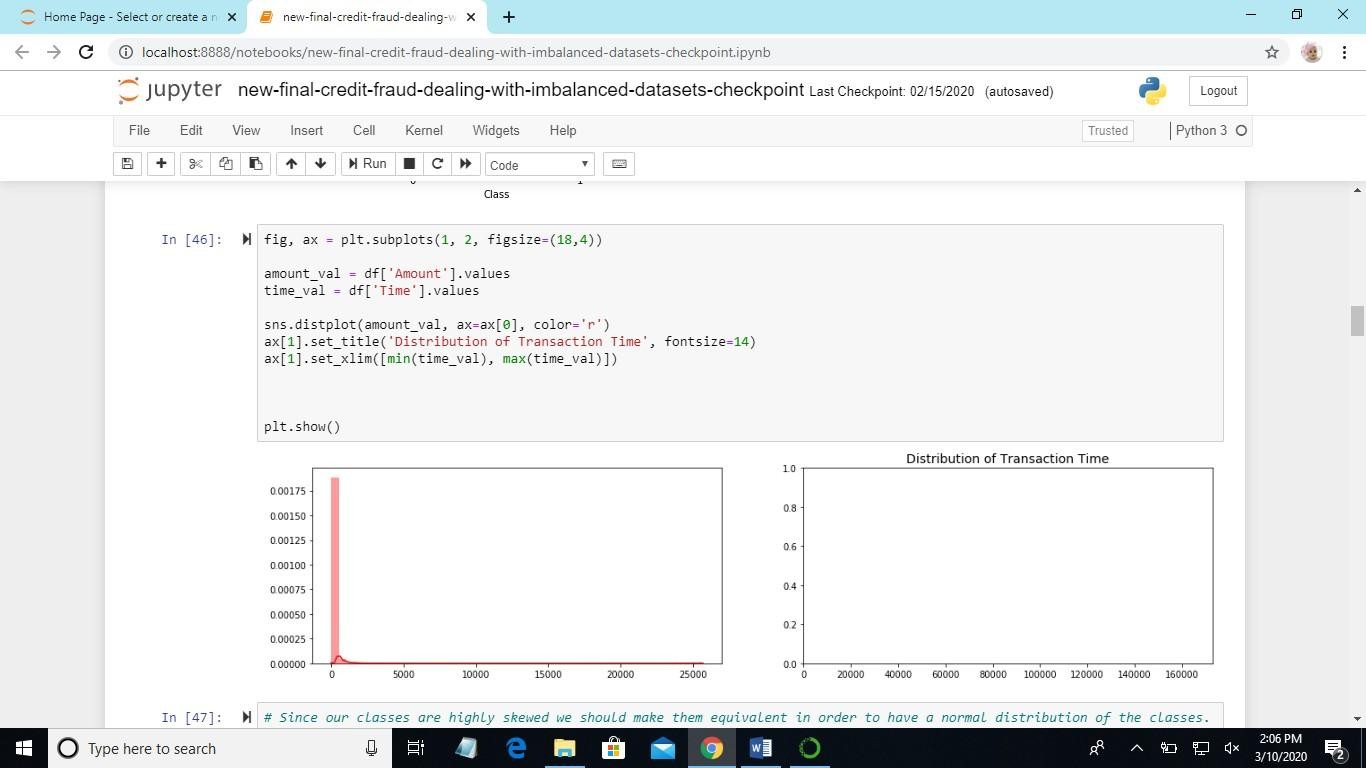


Fig 6.4 : Subplots

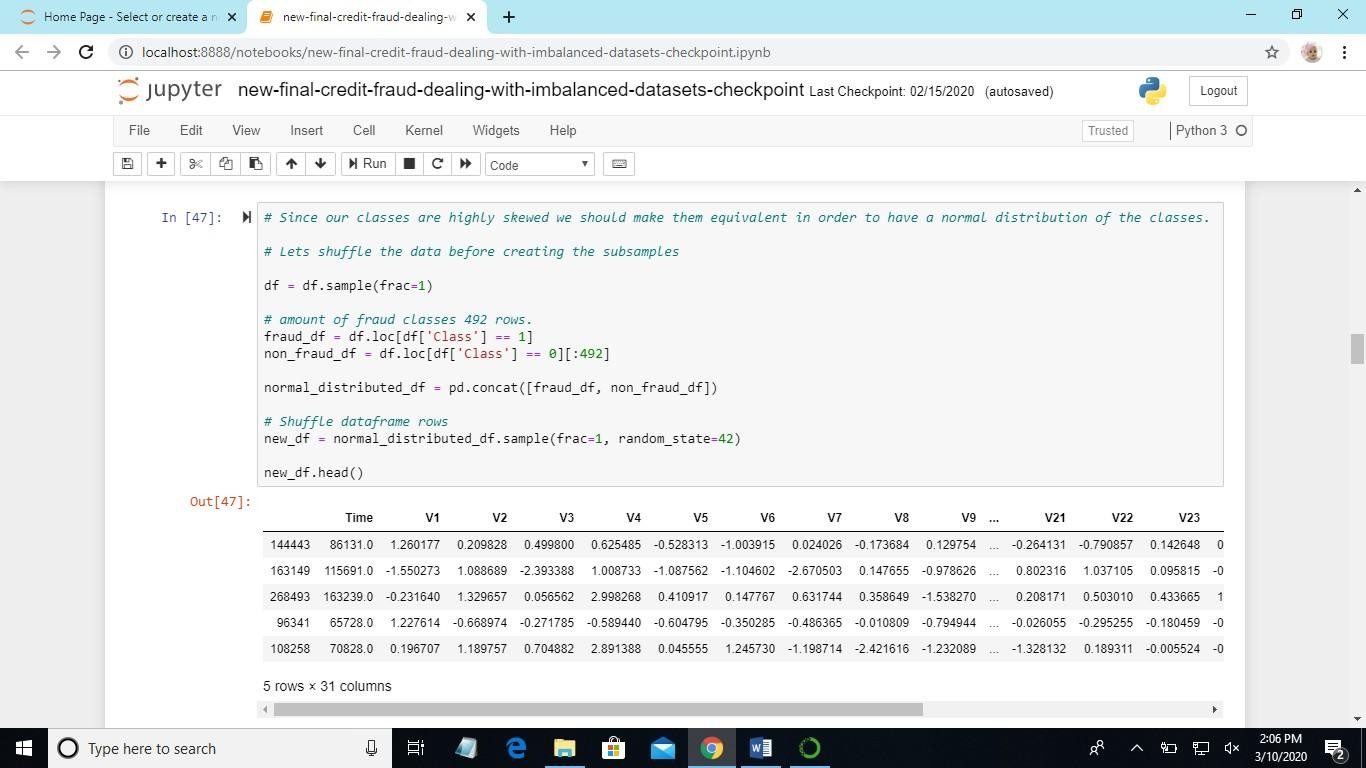


Fig 6.5 : Shuffling of data

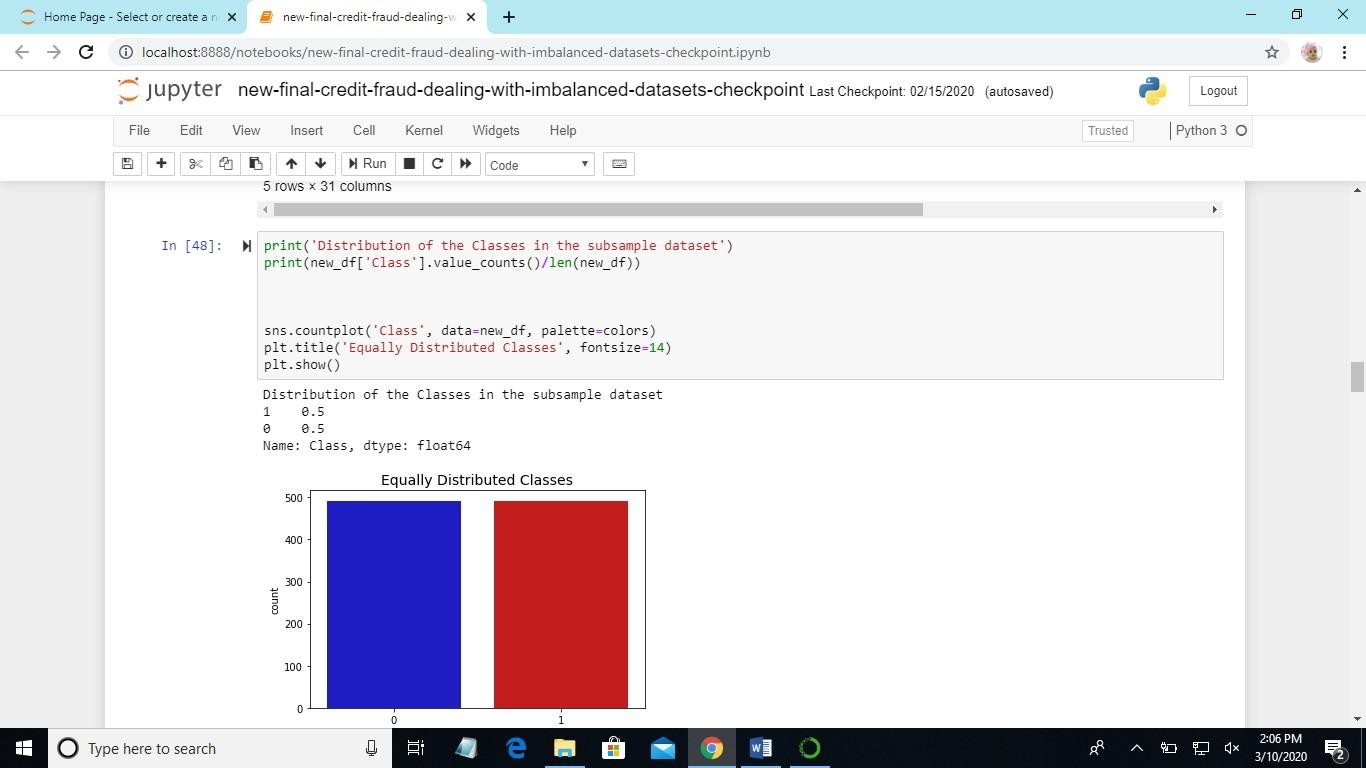


Fig 6.6 : Distribution of classes



Fig 6.7 : Correlation Matrix

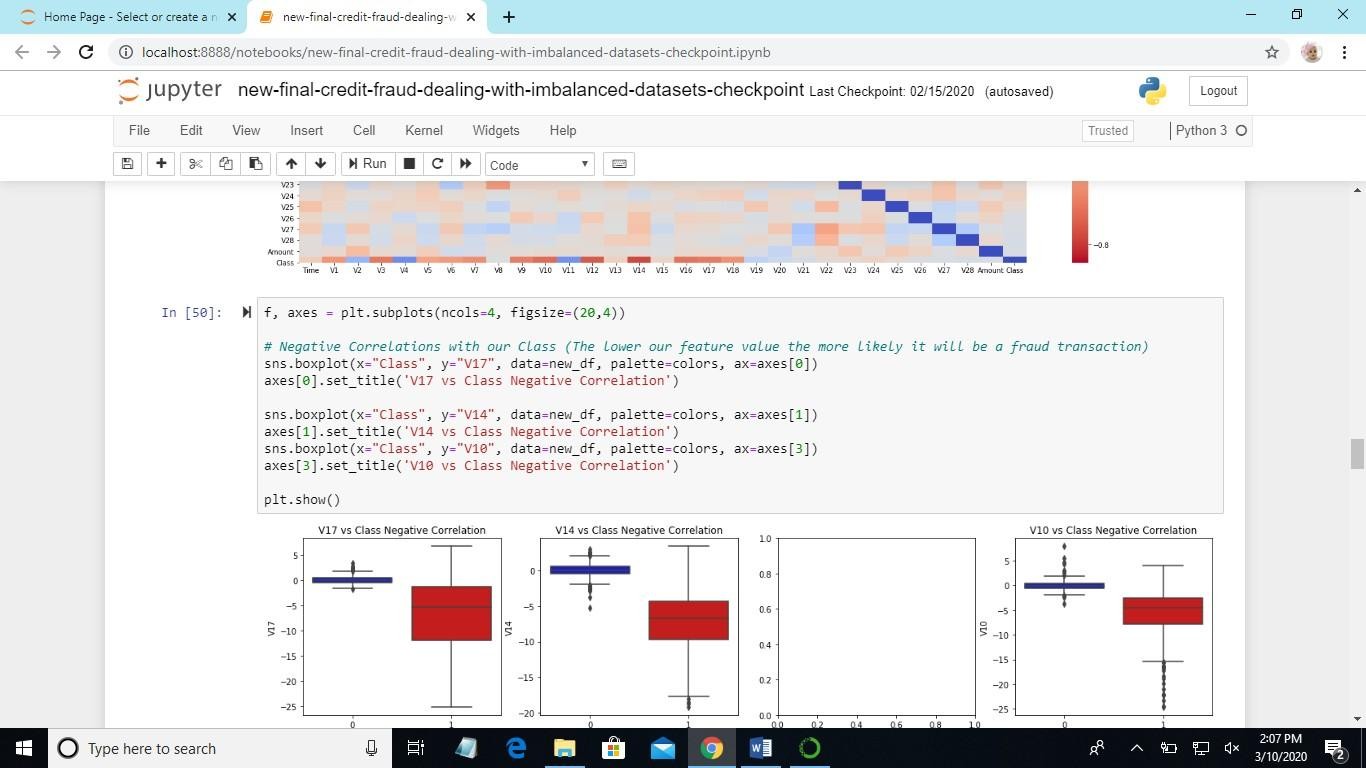


Fig 6.8 : Plotting using different factors

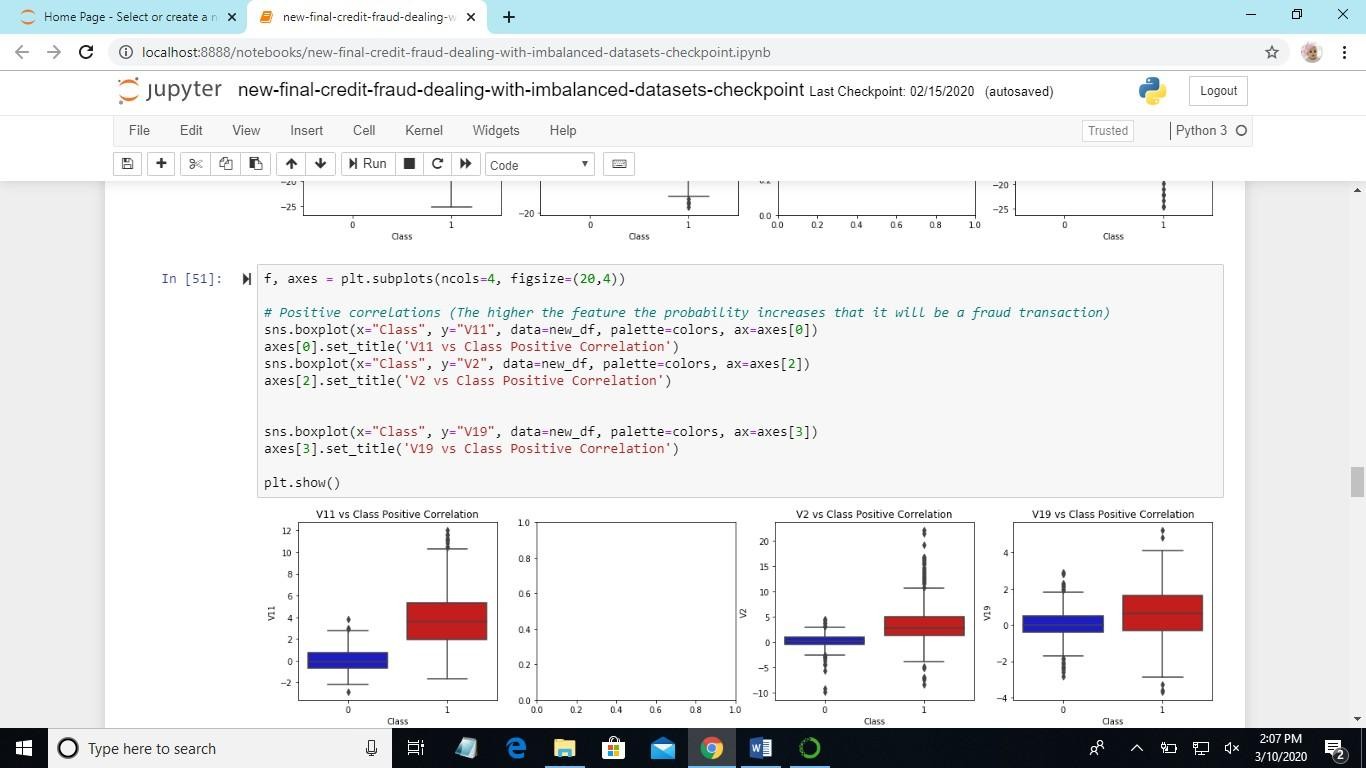


Fig 6.9 : Plot using different columns

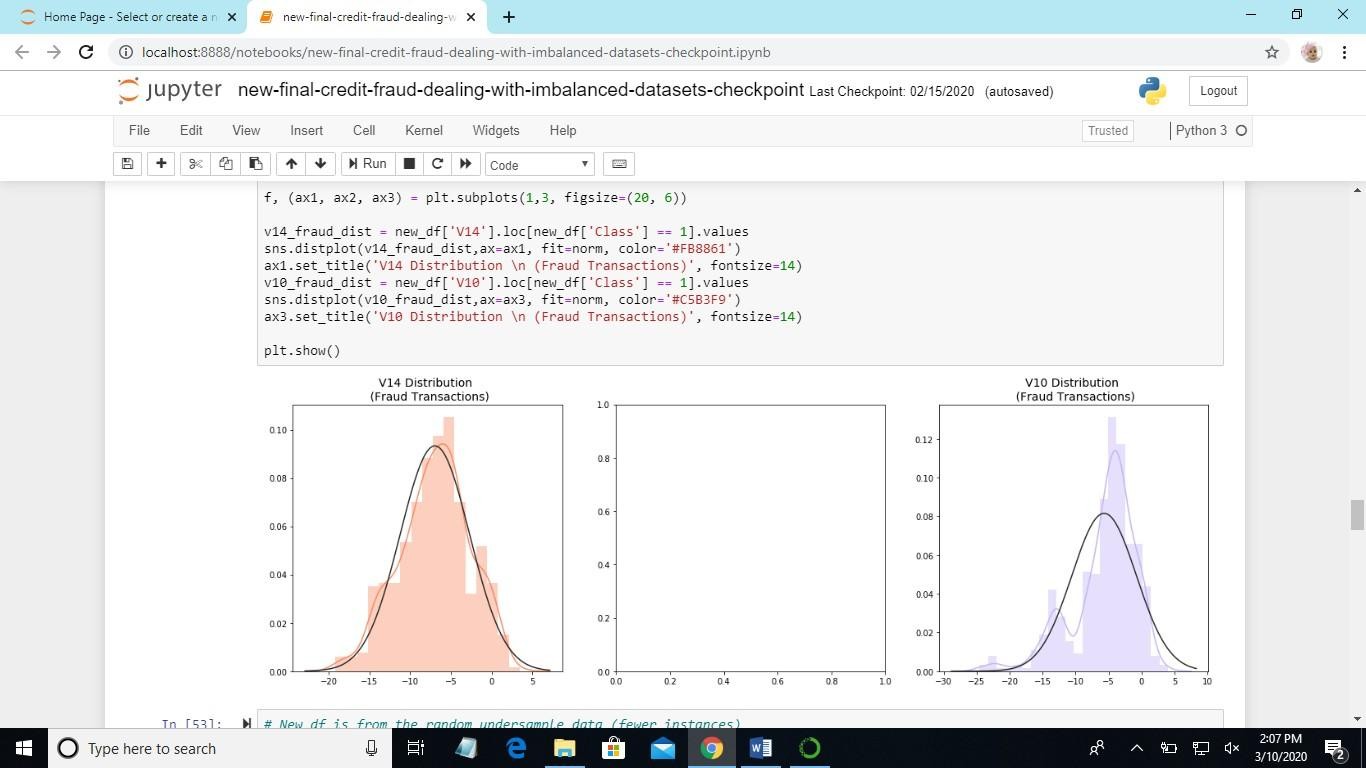


Fig 6.10 :Fraud and Non-Fraud Transactions

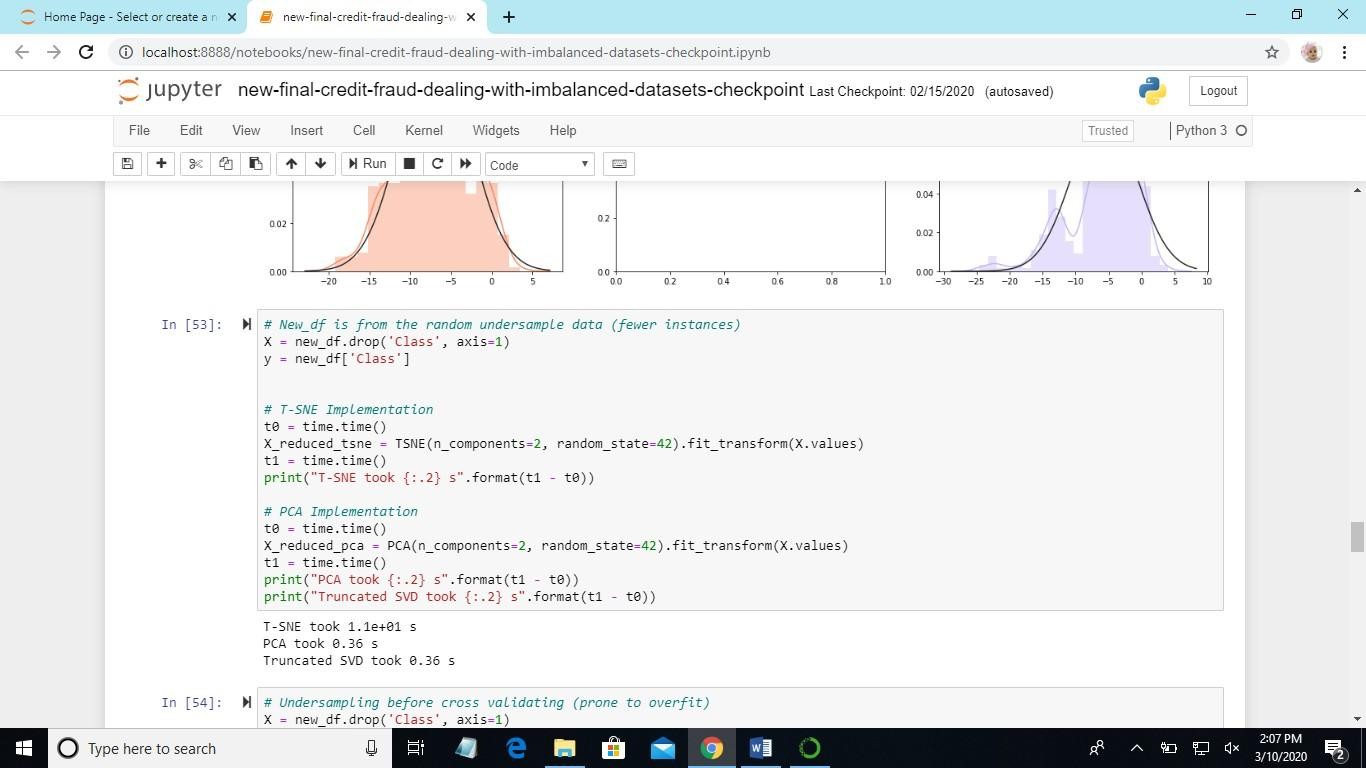


Fig 6.11 :Time Accuracy

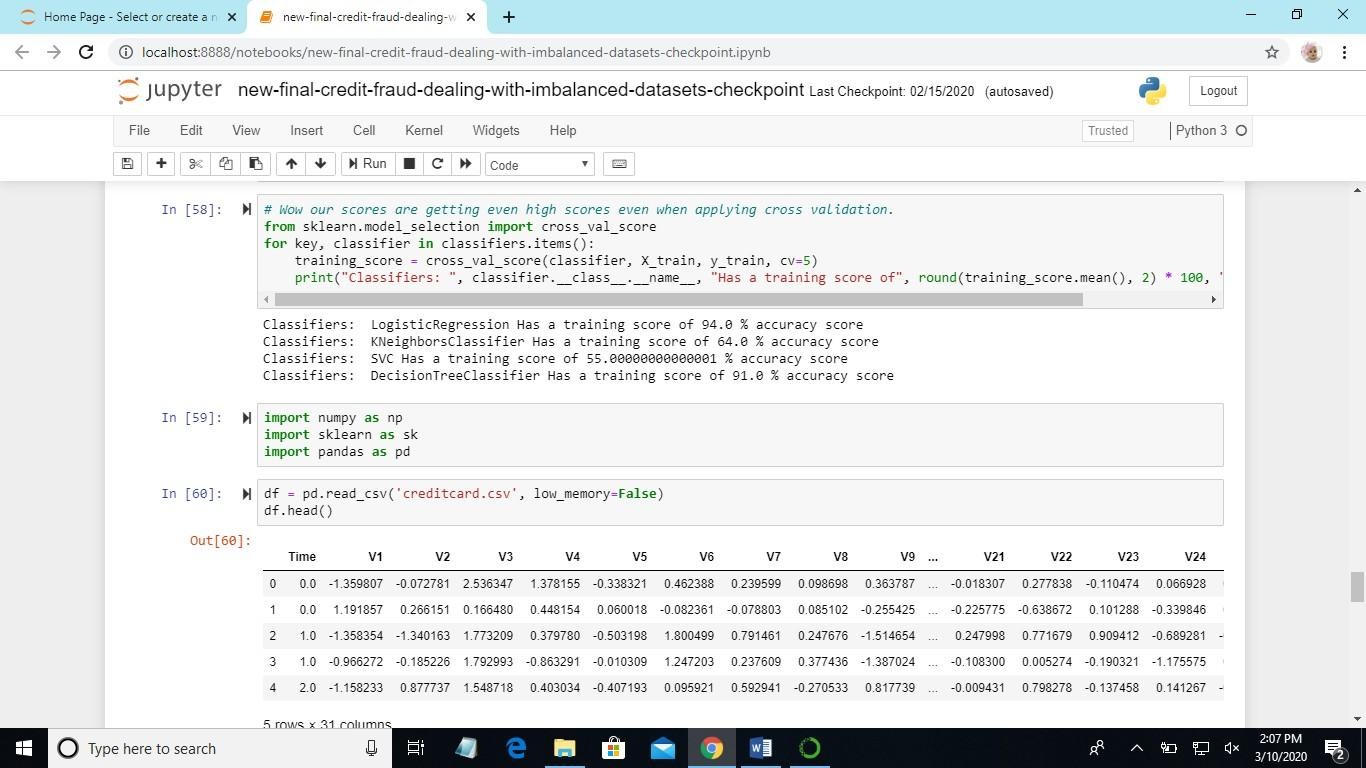


Fig 6.12 :Accuracy score for each classifier

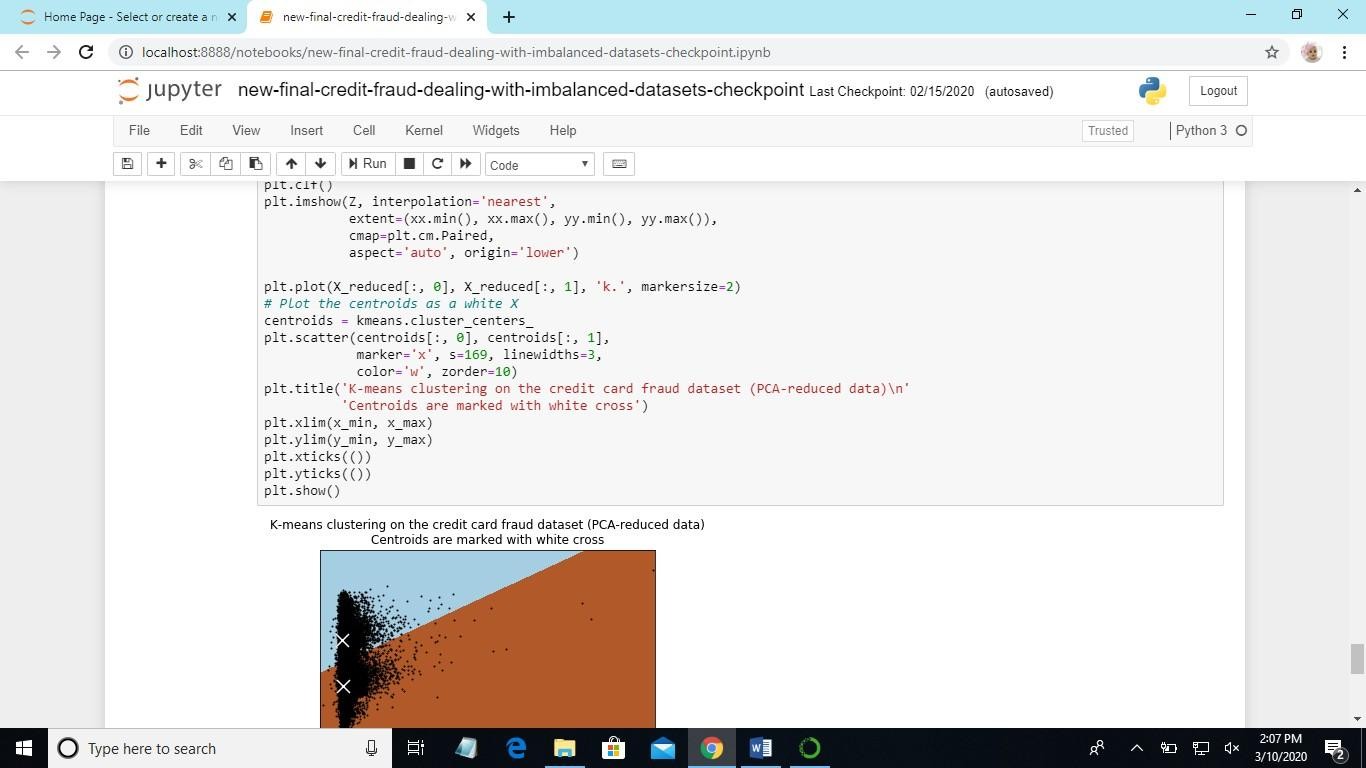


Fig 6.13 :K-means clustering

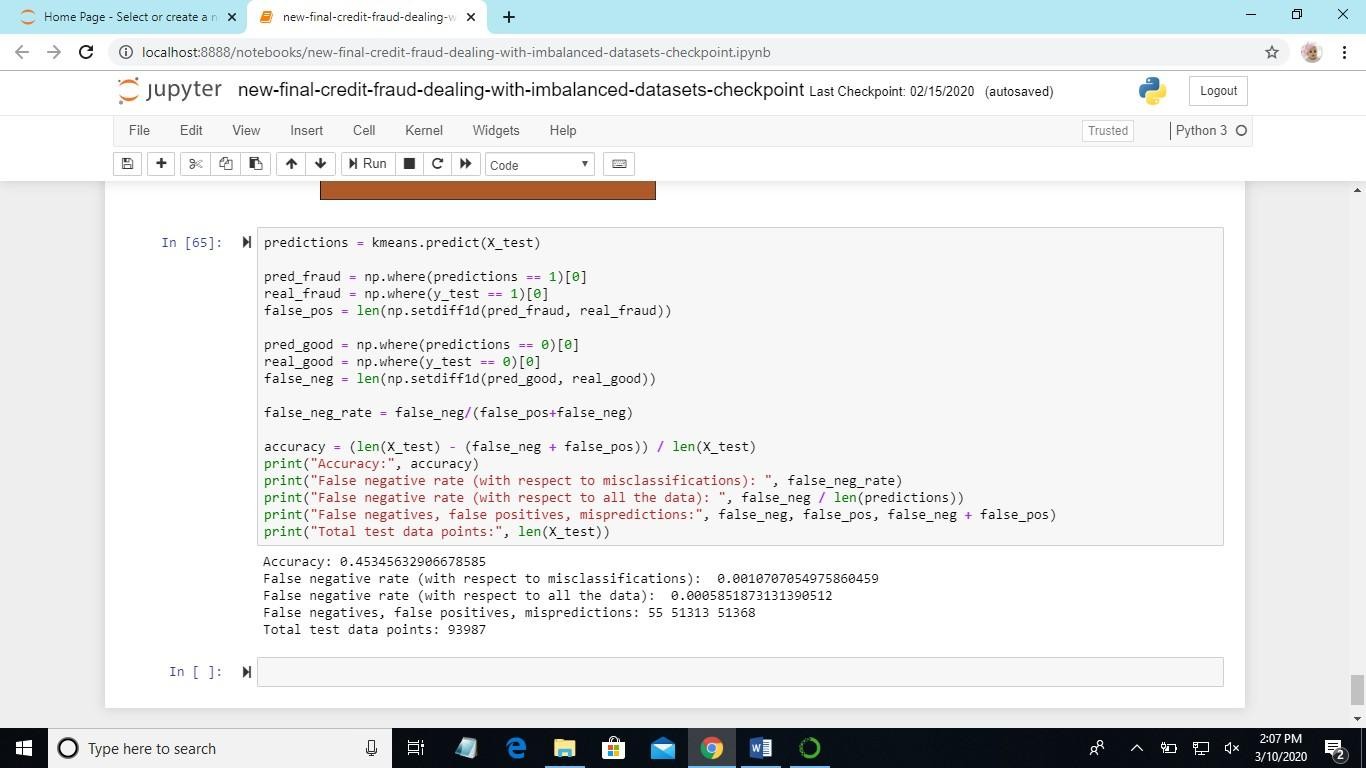


Fig 6.14 :Accuracy Rate Predictions

# CHAPTER 7 TESTING

### TESTING

The purpose of testing is to discover the 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. We have different types of tests. Each test type addresses a specific testing requirement.

#### TYPES OF TESTING:

* + 1. **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 TESTING:

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 TESTING:

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.

#### 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 CASES:

Experiments include an arrangement of steps,conditions and sources information that can be utilized while performing testing undertakings. The principle expectation of this action is to guarantee whether a product passes as far as useful and different perspectives. The way toward creating experiments can likewise help discover issues in the prerequisites or plan an application. Experiment goes about as the beginning stage for the test execution, and in the wake of applying an arrangement of information esteems, the application has a conclusive result and leaves the framework at same end point or otherwise called execution post condition.

# CHAPTER 8 CONCLUSION

### CONCLUSION

In this project, IDS based on SVM and random forest along with random projection is implemented. The performance of IDS is evaluated with the help of NSL – KDD dataset. The IDS is trained and tested by means of data set. The proposed work used random projection technique one of the efficient feature selection technique to choose the best attributes from the data set. They grouped the features to generate a good yield. The classification techniques SVM; random forest along with random projection is used. The detection rate of random projection along with random forest is approximately 100% and the detection rate of random projection along with SVM is 98.5% which is a good detection rate. But, one factor is time management, as we can see, the time taken for the random projection with random forest to obtain the result is very less than the second classifier i.e. SVM with random projection. From the comparison table, we can conclude that random forest along with random projection yielded the best output than support vector machine along with random projection. The various attack classes such as DoS, probe, U2R and R2L are listed along with their accuracy value. The accuracy rate for DoS is 100% for random projection with random forest. This is an efficient rate when compared with other proposed algorithm. In SVM, the accuracy rate for U2R attack is very poor. The accuracy for probe attack in random projection along with random forest is 99.5% which is a good rate when compared with SVM. Finally, we can conclude that random forest is the best classifiers amidst the other classifiers. As a future work, the proposed IDS can be implemented on network for protecting it from unlawful activity. In addition, it can be implemented for http services, ftp services for the detection of unauthorized work.

# CHAPTER 9 FUTURE ENHANCEMENT

### FUTURE ENHANCEMENT

We have presented Exact Match in One Memory Access (EMOMA), a scheme that implements exact match with only one access to external memory, targeted towards hardware implementations of high availability network processing devices.

* EMOMA uses a counting block Bloom filter to select the position that needs to be accessed in an external memory cuckoo hash table to find an element. By sharing one hash function between the cuckoo hash table and the counting block Bloom filter, we enable fast identification of the elements that can create false positives, allowing those elements to be moved in the hash table to avoid the false positives.
* This requires a few additional memory accesses for some insertion operations and a slightly more complex insertion procedure. Our evaluation shows that EMOMA can achieve around 95% utilization of the external memory when using only slightly more than 4 bits of on-chip memory for each element stored in the table.
* This compares quite favorably with previous schemes such as Fast Hash Table [19], and is also simpler for implementation.

Previously, in existing system NIDS and HIDS are implemented for identifying existing intrusions based on the id. If there is any new dynamic intrusion occurs NIDS and HIDS they didn’t detect the intrusions. So this is the major drawback in existing system.

Support vector machine is a technique that has been emerged for the analysis of a data for the classification process. The support vector machine is also known as support vector networks. The SVM uses a set of training data where each one has been labeled into one of two categories. The training data set builds a model and the new unknown data would be categorized into the proper group. There will be a

linear separation between the data that has to be classified. With the help of this line the data can be easily separated with more accuracy. In this technique, two categories are available; we can either classify the data to one class or to the other class depending upon the behavior of the new data. In addition to linear classification, they can perform non linear classification also. In non linear classification, data are not labeled so supervised learning is not possible. In such a context, unsupervised learning approach is implemented, which attempts to cluster the data with similar behaviour.

# CHAPTER 10 BIBLIOGRAPHY

### BIBLIOGRAPHY

1. S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, “Data mining for credit card fraud: A comparative study,” Decis. Support Syst., vol. 50, no. 3, pp.602–613, 2011
2. K. Chaudhary, J. Yadav, and B. Mallick, “A review of Fraud Detection Techniques: Credit Card,” Int. J. Comput. Appl., vol. 45, no. 1, pp. 975–8887, 2012.
3. F. N. Ogwueleka, “Data Mining Application in Credit Card Fraud Detection System,” vol. 6, no. 3, pp. 311–322, 2011.
4. H. Nordberg, K. Bhatia, K. Wang, and Z. Wang, “BioPig: a Hadoop-based analytic toolkit for large-scale sequence data,” Bioinformatics, vol. 29, no. 23, pp. 3014– 3019, Dec. 2013.
5. M. Hegazy, A. Madian, and M. Ragaie, “Enhanced Fraud Miner: Credit Card Fraud Detection using Clustering Data Mining Techniques,” Egypt. Comput. Sci., no. 03, pp. 72–81, 2016.
6. M. Zareapoor and P. Shamsolmoali, “Application of credit card fraud detection: Based on bagging ensemble classifier,” Procedia Comput. Sci., vol. 48, no. C, pp. 679–686, 2015.
7. O. S. Yee, S. Sagadevan, N. Hashimah, and A. Hassain, “Credit Card Fraud Detection Using Machine Learning As Data Mining Technique,” vol. 10, no. 1, pp.23–27.
8. K. Randhawa, C. K. Loo, M. Seera, C. P. Lim, and A. K. Nandi, “Credit Card Fraud Detection Using AdaBoost and Majority Voting,” IEEE Access, vol. 6, pp.14277– 14284, 2018.
9. N. Mahmoudi and E. Duman, “Detecting credit card fraud by Modified Fisher Discriminant Analysis,” Expert Syst. Appl., vol. 42, no. 5, pp. 2510–2516, 2015.
10. A. Dal Pozzolo, O. Caelen, Y. A. Le Borgne, S.Waterschoot, and G.Bontempi, “Learned lessons in credit card fraud detection from a practitioner perspective,” Expert Syst. Appl., vol. 41, no. 10, pp. 4915–4928, 2014.
11. M. A. Scholar, M. Ali, and P. Fellow, “Investigating the Performance of Smote for Class Imbalanced Learning : A Case Study of Credit Scoring Datasets,”vol. 13, no. 33, pp. 340–353, 2017.
12. H. He, W. Zhang, and S. Zhang, “A novel ensemble method for credit scoring: Adaption of different imbalance ratios,” Expert Syst. Appl., vol. 98, pp. 105–117, May 2018.
13. A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, “Calibrating Probability with Undersampling for Unbalanced Classification.
14. Y. Sahin and E. Duman, “Detecting Credit Card Fraud by Decision Trees and Support Vector Machines,” Int. Multiconference Eng.