ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

We avail this opportunity to express our deep sense of gratitude and hearty thanks to management of Vardhaman College of Engineering, for providing congenial atmosphere and encouragement.

Our sincere thanks to Dr Mallikharjuna Babu, Director & CEO, for his encouragement.

We show gratitude to **Dr. S Sai Satyanarayana Reddy**, **Principal** for having provided all the facilities and support.

We would like to thank **Dr. Rajanikanth Aluvalu, Head of the Department, Computer Science and Engineering** for his expert guidance and encouragement at various levels of our Project.

We are thankful to our guide **Mr. Faculty Name, Designation** for his sustained inspiring Guidance and cooperation throughout the process of this project. His wise counsel and suggestions were invaluable.

We express our deep sense of gratitude and thanks to all the **Teaching** and **Non-Teaching Staff** of our college who stood with us during the project and helped us to make it a successful venture.

We place highest regards to our **Parents**, our **Friends** and **Well wishers** who helped a lot in making the report of this project.

16881A05G8	**A. Sri Satya Preetam**
16881A05G6	**P. Nishanth Goud**
16881A05H2	**M . MOUNIKA**

ABSTRACT

A novel supervised machine learning system is developed to classify network traffic whether it is malicious or benign. To find the best model considering detection success rate, a combination of the supervised learning algorithm and feature selection method has been used. Through this study, it is found that Artificial Neural Network (ANN) based machine learning with wrapper feature selection outperforms support vector machine (SVM) technique while classifying network traffic. To evaluate the performance, the NSL-KDD dataset is used to classify network traffic using SVM and ANN supervised machine learning techniques. A comparative study shows that the proposed model is efficient than other existing models with respect to the intrusion detection success rate.

Existing System, While network IDS that works based on the signature, have seen commercial success and widespread adoption by the technology-based organization throughout the globe, anomaly-based network IDS have not gained success on the same scale. Due to that reason in the field of IDS, currently, anomaly-based detection is a major focus area of research and development and before going to any wide-scale deployment of the anomaly-based intrusion detection system, key issues remain to be solved. But the literature today is limited when it comes to comparing how intrusion detection performs when using supervised machine learning techniques.

Proposed System, The promise and the contribution of machine learning did until today are fascinating. There are many real-life applications we are using today offered by machine learning. It seems that machine learning will rule the world in the coming days. Hence we came out into a hypothesis that the challenge of identifying new attacks or zero-day attacks facing by the technology-enabled organizations today can be overcome using machine learning techniques. Here we developed a supervised machine learning model that can classify unseen network traffic based on what is learned from the seen traffic. We used both SVM and ANN learning algorithms to find the best classifier with higher accuracy and success rate.

Table Of Contents

List of tables

Symbols & Abbreviations

SDLC	Software Development Life Cycle
SVM	Support Vector Machine
UML	Unified Modeling Language
IDS	Intrusion Detection System
IDE	Integrated Development Environment
IEEE	International of Electrical and Electronic Engineers

1.INTRODUCTION

With the wide-spreading usages of internet and increases in access to online content, cybercrime is also happening at an increasing rate. Intrusion detection is the first step to prevent security attacks. Hence the security solutions such as Firewall, Intrusion Detection System (IDS), Unified Threat Modeling (UTM) and Intrusion Prevention System (IPS) are getting much attention in studies. IDS detect attacks from a variety of systems and network sources by collecting information and then analyze the information for possible security breaches. The network-based IDS analyzes the data packets that travel over a network and this analysis is carried out in two ways. Till today anomaly-based detection is far behind than the detection that works based on the signature and hence anomaly-based detection still remains a major area for research. The challenges with anomaly-based intrusion detection are that it needs to deal with the novel attack for which there is no prior knowledge to identify the anomaly. Hence the system somehow needs to have the intelligence to segregate which traffic is harmless and which one is malicious or anomalous and for that machine learning techniques are being explored by the researchers over the last few years. IDS, however, is not an answer to all security-related problems. For example, IDS cannot compensate weak identification and authentication mechanisms or if there is a weakness in the network protocols.

1.1 MOTIVATION

While network IDS that works based on signature have seen commercial success and widespread adoption by the technology-based organization throughout the globe, anomaly-based network IDS have not gained success on the same scale. Due to that reason in the field of IDS, currently, anomaly-based detection is a major focus area of research and development and before going to any wide-scale deployment of anomaly-based intrusion detection systems, key issues remain to be solved. But the literature today is limited when it comes to comparing how intrusion detection performs when using supervised machine learning techniques.

1.2 PROBLEM DEFINITION

Currently anomaly based detection is a major focus area of research and development and before going to any wide scale deployment of anomaly based intrusion detection system, key issues remain to be solved. But the literature today is limited when it comes to compare on how intrusion detection performs when using supervised machine learning techniques.

1.3 OBJECTIVE OF PROJECT

We developed a supervised machine learning model that can classify unseen network traffic based on what is learned from the seen traffic. We used both SVM and ANN learning algorithms to find the best classifier with higher accuracy and success rate.

1.4 LIMITATIONS OF PROJECT

The literature today is limited when it comes to comparing how intrusion detection performs when using supervised machine learning techniques.

1.5 ORGANIZATION OF DOCUMENTATION

The most common classifications are network intrusion detection systems (NIDS) and host-based intrusion detection systems (HIDS). A system that monitors important operating system files is an example of an HIDS, while a system that analyzes incoming network traffic is an example of an NIDS.

2.LITERATURE SURVEY

2.1 INTRODUCTION

Koushal Kumar & Jaspreet Singh Batth From the literature survey we come to know that a lot of effort has been done on improving the Naïve Bayes classifier, following two approaches: selecting feature subset and relaxing independence assumptions are mostly used. In this study, the authors have proposed a new algorithm that works on both the above-mentioned approaches. In the present work, an updated version of the Naive Bayes classifier algorithm without assuming conditional independence of different attributes is proposed.

Nivedita S Naganhalli, Dr. Sujata Terdal In 1998, the DARPA Intrusion Detection Assessment Program was prepared and managed by MIT Lincoln Labs. Its purpose was to study and evaluate intrusion detection research. Standard data sets include various simulation intrusions in military network environments. The connection to the dataset includes a sequence of TCP packets beginning and ending at a well-defined time between the source IP address and the destination IP address using a well-defined protocol. Each connection is categorized as a normal or specific type of attack. Data sets are categorized into five sub-sets: denial-of-service attacks, local or remote network attacks, user/root attacks, sample attacks, and generic data. Each record is classified as normal or attack with exactly one type of attack.

2.2 EXISTING SYSTEM

While network IDS that works based on signature have seen commercial success and widespread adoption by the technology-based organization throughout the globe, anomaly-based network IDS have not gained success on the same scale. Due to that reason in the field of IDS, currently, anomaly-based detection is a major focus area of research and development and before going to any wide-scale deployment of the anomaly-based intrusion detection system, key issues remain to be solved. But the literature today is limited when it comes to comparing how intrusion detection performs when using supervised machine learning techniques.

2.3 DISADVANTAGES OF EXISTING SYSTEM

Intrusion detection systems are able to detect behavior that is not normal for average network usage. While it's good to be able to detect abnormal network usage, the disadvantage is that the intrusion software can create a large number of false alarms.

2.4 PROPOSED SYSTEM

The promise and the contribution of machine learning did until today are fascinating. There are many real-life applications we are using today offered by machine learning. It seems that machine learning will rule the world in the coming days. Hence we came out into a hypothesis that the challenge of identifying new attacks or zero-day attacks facing by the technology-enabled organizations today can be overcome using machine learning techniques. Here we developed a supervised machine learning model that can classify unseen network traffic based on what is learned from the seen traffic. We used both SVM and ANN learning algorithms to find the best classifier with higher accuracy and success rate.

3.ANALYSIS

3.1 INTRODUCTION

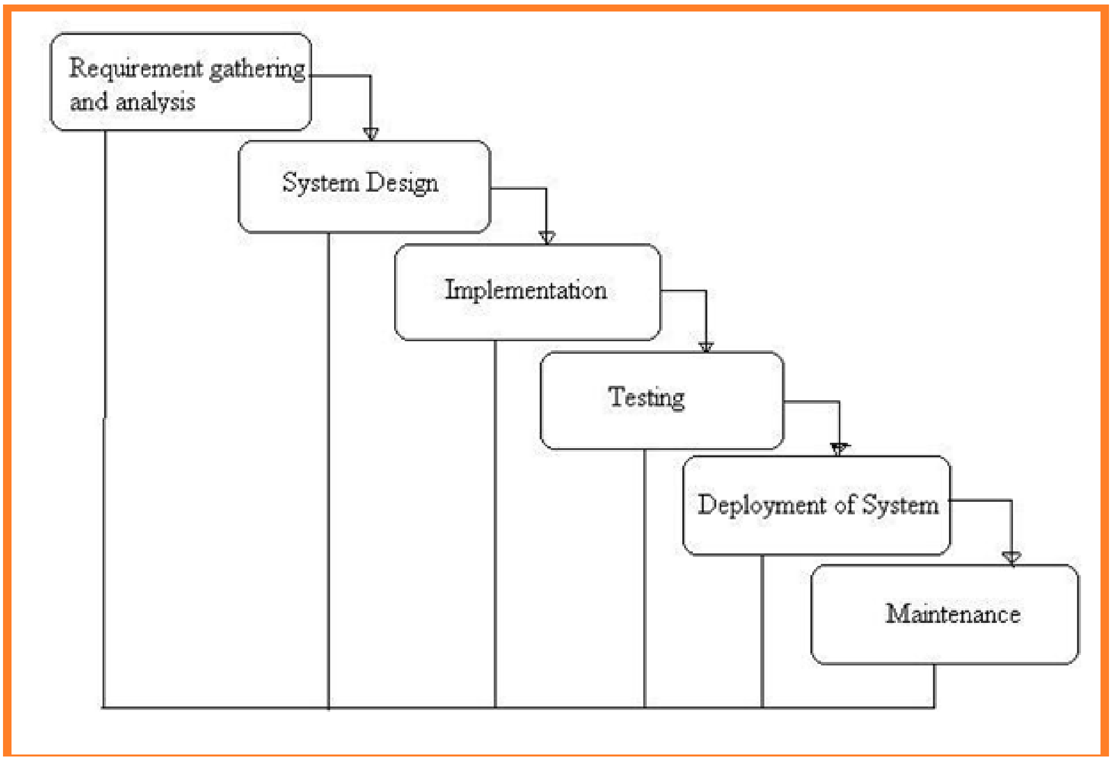


Fig:-1 Project SDLC

- Project Requisites Accumulating and Analysis
- Application System Design
- Practical Implementation
- Manual Testing of My Application
- Application Deployment of System
- Maintenance of the Project

Requisites Accumulating and Analysis

It’s the first and foremost stage of any project as our is an academic leave for requisites amassing we followed of IEEE Journals and Amassed so many IEEE Relegated papers and final culled a Paper designated “Individual web revisitation by setting and substance importance input and for analysis stage we took referees from the paper and did literature survey of some papers and amassed all the Requisites of the project in this stage

System Design

In System Design has divided into three types like GUI Designing, UML Designing with avails in development of the project in a facile way with a different actor and its utilizer case by utilizer case diagram, flow of the project utilizing sequence, the Class diagram gives information about the different class in the project with methods that have to be utilized in the project if comes to our project our UML Will utilizable in this way The third and post-import for the project in system design is Database design where we endeavor to design database predicated on the number of modules in our project

Implementation

The Implementation is Phase where we endeavor to give the practical output of the work done in designing stage and most of the Coding in Business logic lay coms into action in this stage its main and crucial part of the project

Testing

Unit Testing It is done by the developer itself in every stage of the project and fine-tuning the bug and module predicated additionally done by the developer only here we are going to solve all the runtime errors Manual Testing As our Project is academic Leave we can do any automatic testing so we follow manual testing by endeavor and error methods Deployment of System Once the project is total yare we will come to the deployment of a client system in genuine world as its academic leave we did deployment in college lab only with all need Software's with having Windows OS Maintenance The Maintenance of our Project is a one-time process only.

Non-Functional Requisites Expanded System admin security

overseer to eschew the abuse of the application by PC ought to be exceptionally secured and available.

Compactness

The Presentation of this application is facile to utilize so it looks simple for the user client to comprehend and react to identically tantamount.

Unwavering

quality and the functionalities accessible in the application this substructure has a high probability to convey us the required inquiries. 10 Time take for Reaction: The time taken by the application to culminate an undertaking given by the client is very fast.

Multifariousness

Our application can be stretched out to incorporate the vicissitudes done by applications present now to enhance the performance of the item. This is implicatively insinuated for the future works that will be done on the application.

Vigor

The project is to blame tolerance concerning illicit client/beneficiary sources of info. Blunder checking has been worked on the platforms to avert the platform's disappointment.

3.2 SOFTWARE REQUIREMENT SPECIFICATION

Feature Selection:

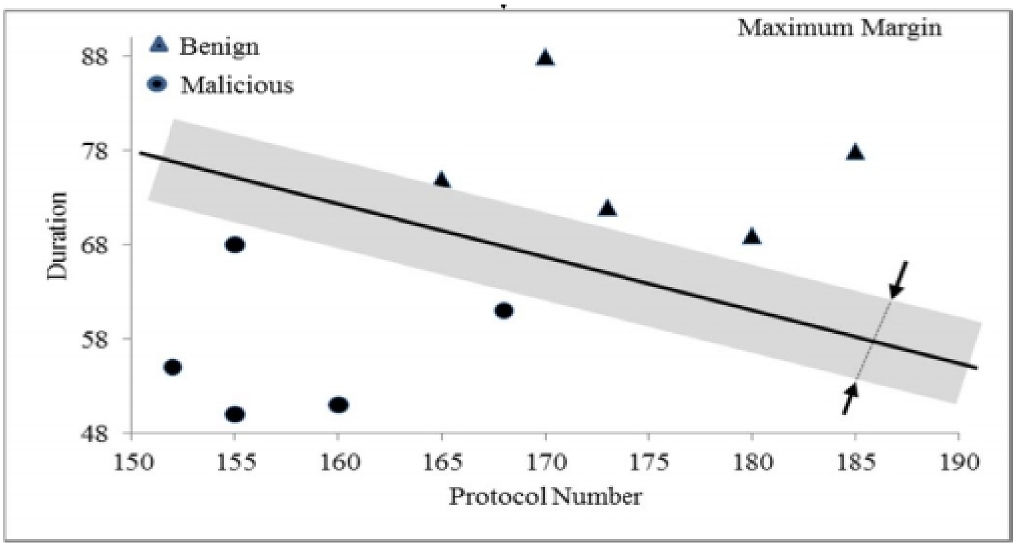
Feature selection is an important part of machine learning to reduce data dimensionality and extensive research carried out for a reliable feature selection method. For feature selection filter method and the wrapper, method have been used. In the filter method, features are selected on the basis of their scores in various statistical tests that measure the relevance of features by their correlation with the dependent variable or outcome variable. The wrapper method finds a subset of features by measuring the usefulness of a subset of the feature with the dependent variable. Hence filter methods are independent of any machine learning algorithm whereas in the wrapper method the best feature subset selected depends on the machine learning algorithm used to train the model.

Building Machine Intelligence:

Based on the best features found in the feature selection process, learning models are developed. To develop the learning model, machine learning algorithm is used. The training dataset is used to train the algorithm with the selected features. In supervised machine learning, each instance in the training dataset has the class it belongs to. The algorithm build the learning model based on which the machine learning algorithm is being used.

Support Vector Machine (SVM):

In SVM a separating hyperplane defines the classifier depending on the type of problem and available datasets. In the case where the dataset is one dimensional, the hyperplane is a point, for two-dimensional data it is a separating line as shown in the below Figure.



Artificial Neural Network (ANN):

Artificial Neural Network is another tool used in machine learning. As its name suggests, ANN is a system inspired by the human brain system and replicates the learning system of the human brain. It consists of input and output layers with one or more hidden layers in most cases as shown in Figure. The ANN uses a technique called backpropagation to adjust the outcome with the expected result or class.

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and the 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 funds 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 the 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.

PROCESS MODEL USED WITH JUSTIFICATION

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by the software industry to develop good software.

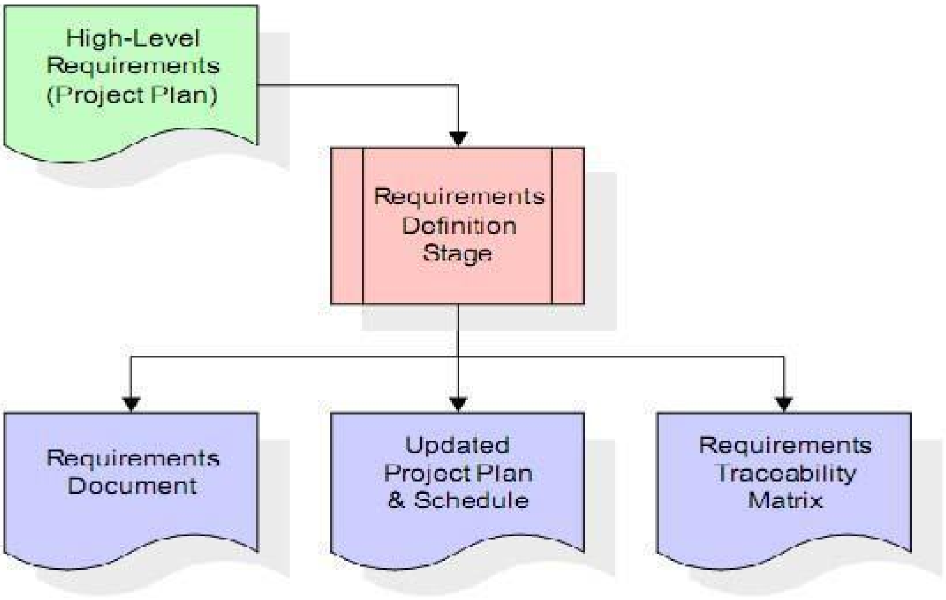
Stages of SDLC:

Requirement Gathering and Analysis

- Designing
- Coding
- Testing
- Deployment

Requirements Definition Stage and Analysis:

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission-critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at a minimum, contain a requirement title and textual description.

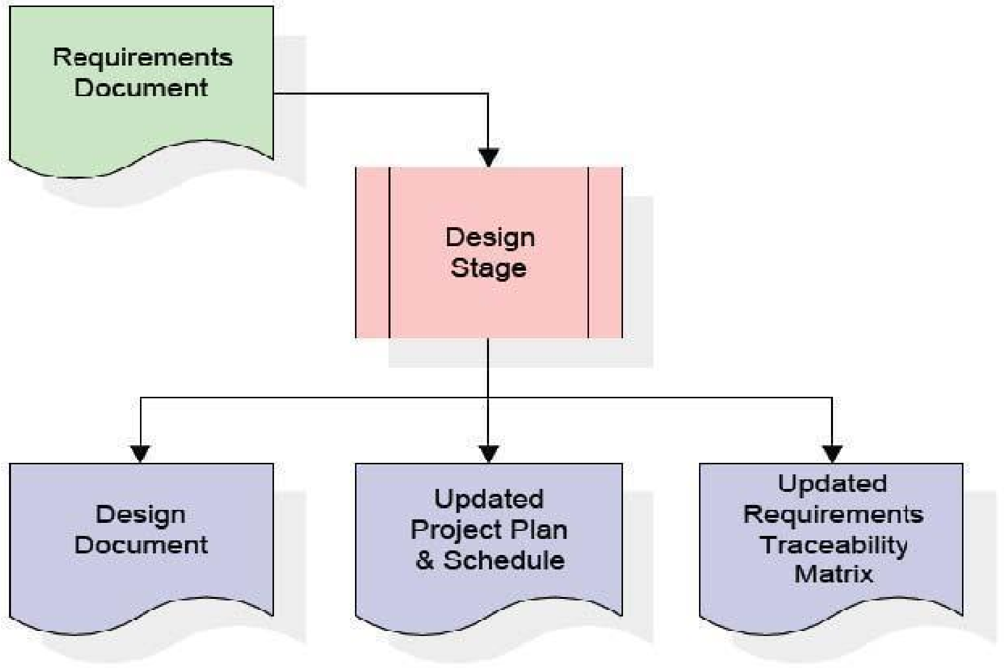

High-Level Requirements (Project Plan)
Requirements Definition Stage
Requirements Document
Updated Project Plan & Schedule
Requirements Traceability Matrix

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). the requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are not included in the requirements document. The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability. The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

Design Stage:

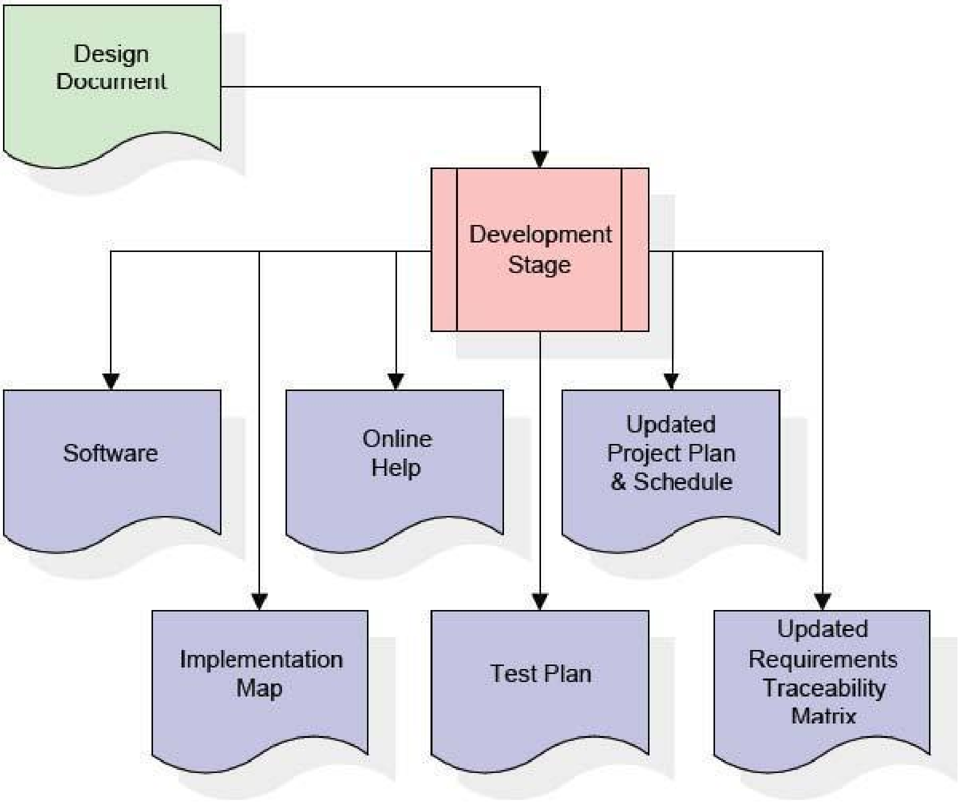
The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.


Requirements Document
Design Stage
Design Document
Updated Project Plan & Schedule
Updated Requirements Traceability Matrix

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

Development Stage:

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

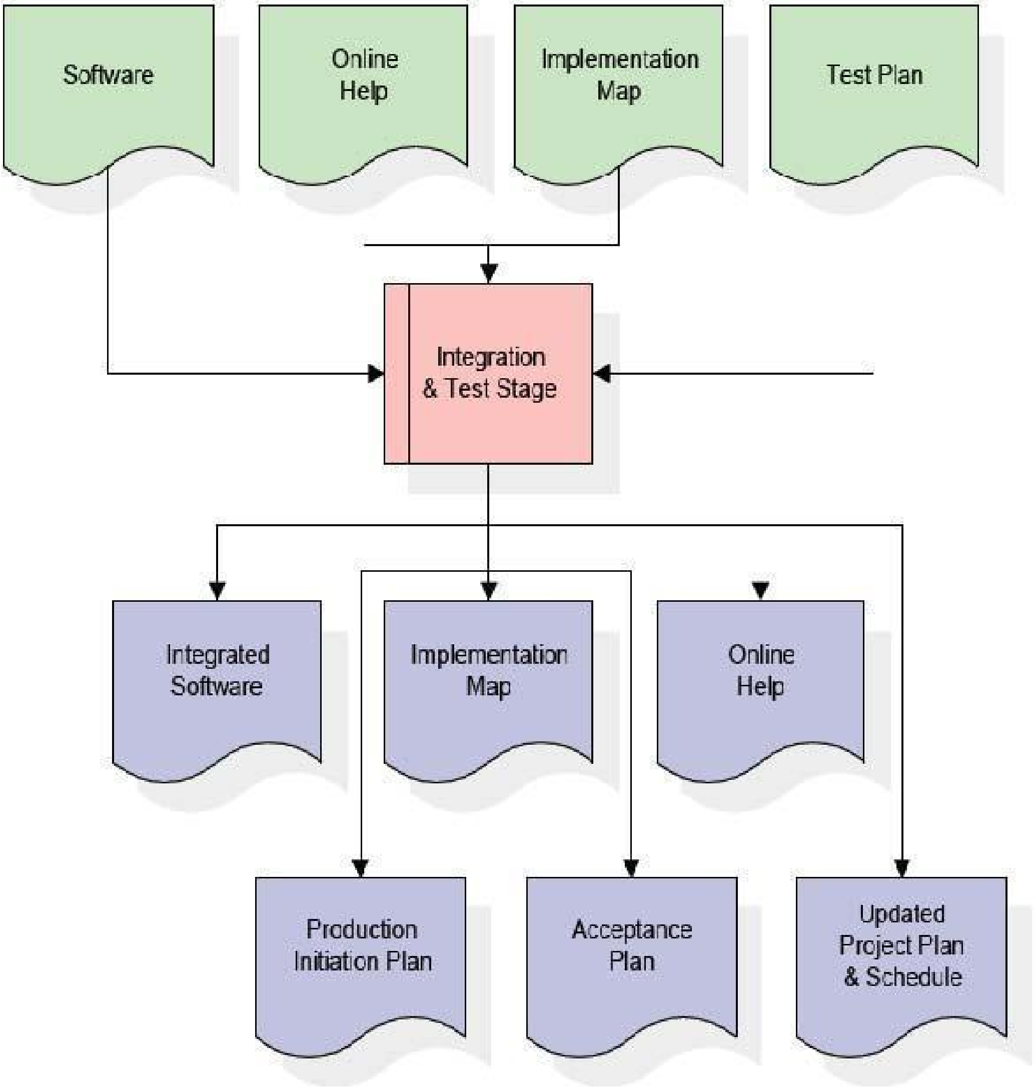


The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

Integration & Test Stage:

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the 17 software. The successful execution of the test suite confirms a robust and complete migration capability.

During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

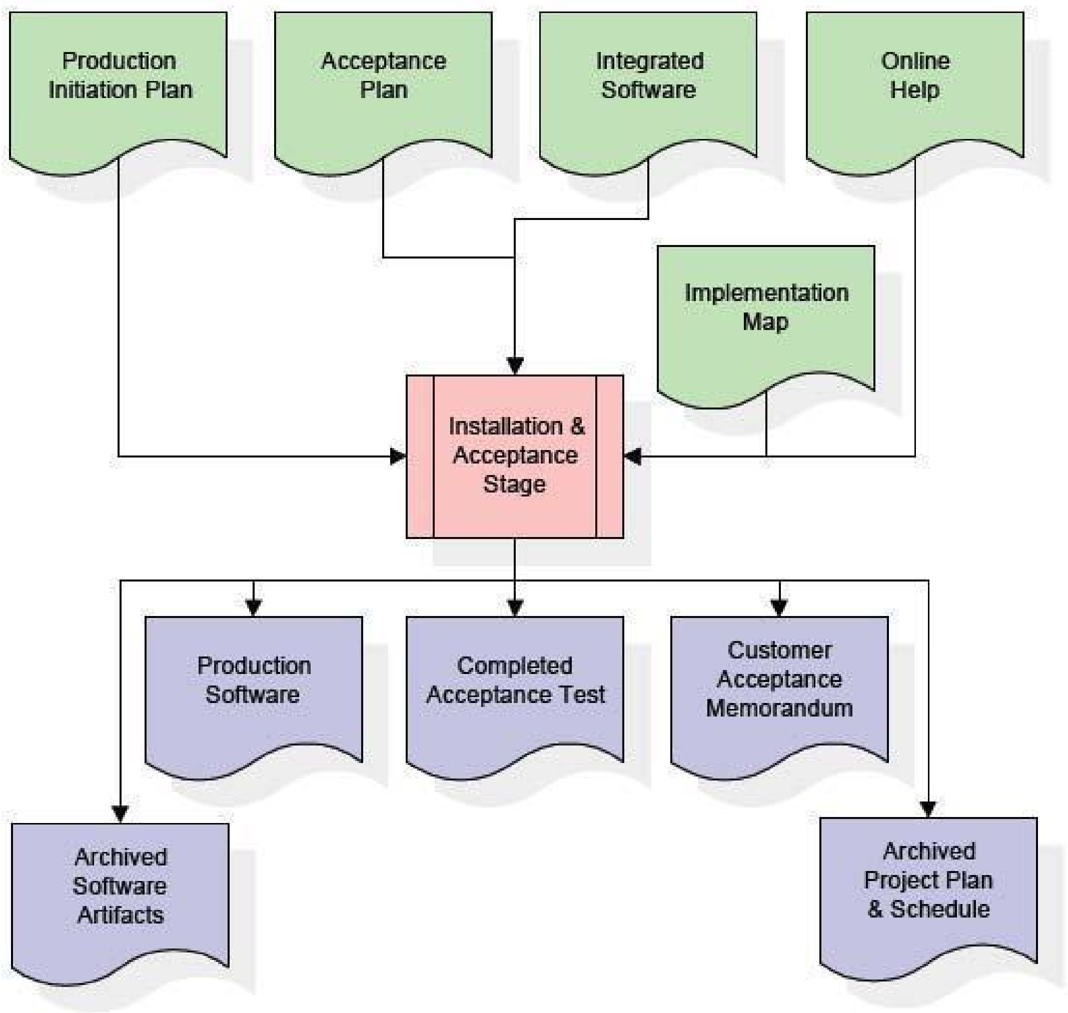


The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

Installation & Acceptance Stage

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel has verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.


Production Initiation Plan
Acceptance Plan
Integrated Software
Online Help
Implementation Map
Installation & Acceptance Stage
Production Software
Completed Acceptance Test
Customer Acceptance Memorandum
Archived Software Artifacts
Archived Project Plan & Schedule

The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point, the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference

SOFTWARE OVERVIEW:

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.

Input as CSV File

Reading data from CSV(comma separated values) is a fundamental necessity in Data Science. Often, we get data from various sources which can get exported to CSV format so that they can be used by other systems. The Panadas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

The CSV file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv. You can create this file using windows notepad by copying and pasting this data. Save the file as input.csv using the save As All files(*.*) option in notepad.

```
import pandas as pd
```

```
data = pd.read_csv('path/input.csv')

print (data)
```

Operations using NumPy

NumPy is a Python package that stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Using NumPy, a developer can perform the following operations :

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- ·Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

Key Features of Pandas

- Fast and efficient DataFrame object with the default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High-performance merging and joining of data.
- Time Series functionality.

SYSTEM CONFIGURATION:

Hardware requirements:

Processer	:	Any Update Processer
Ram	:	Min 4 GB
Hard Disk	:	Min 100 GB

Software requirements:

Operating System	:	Windows family
Technology	:	Python 3.6
IDE	:	PyCharm
Front-End	:	PyQt5

3.3 CONTENT DIAGRAM OR ARCHITECTURE OF PROJECT

The system proposed is composed of feature selection and learning algorithm shown in Figure. The feature selection component is responsible to extract most relevant features or attributes to identify the instance to a particular group or class. The learning algorithm component builds the necessary intelligence or knowledge using the result found from the feature selection component. Using the training dataset, the model gets trained and builds its intelligence. Then the learned bits of intelligence are applied to the testing dataset to measure the accuracy of home much the model correctly classified on unseen data.

3.4 ALGORITHM AND FLOWCHARTS

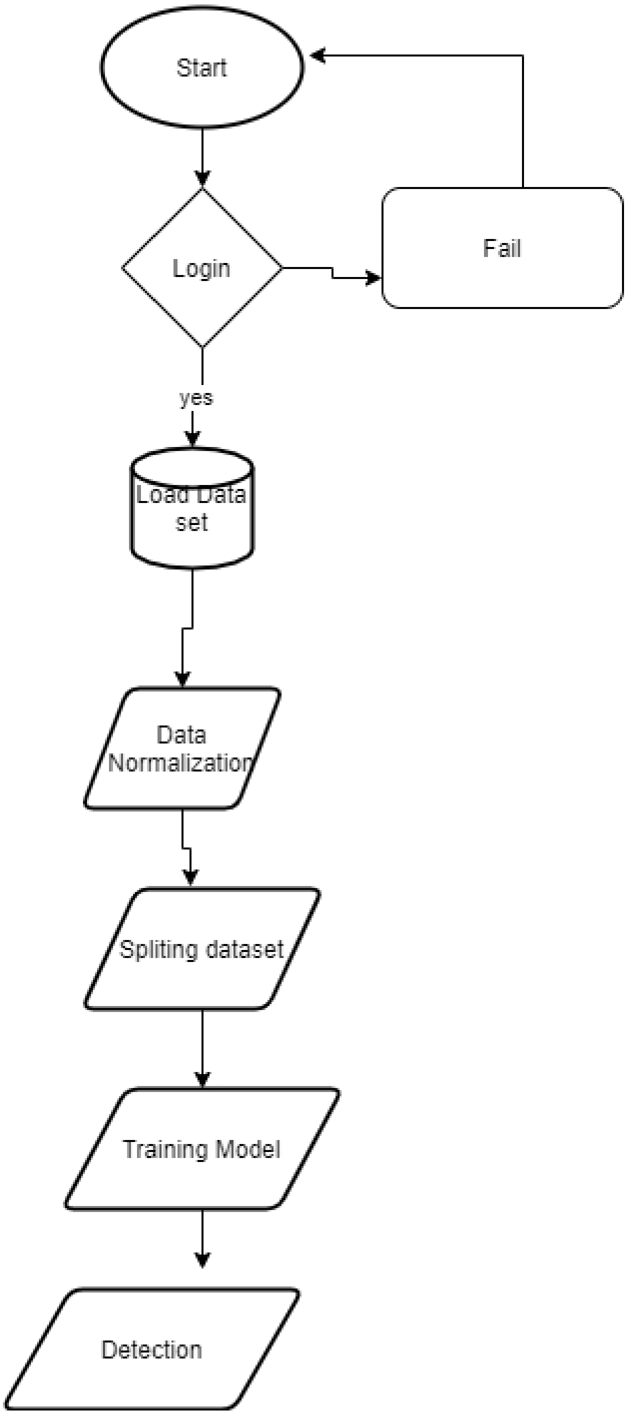
Support Vector Machine (SVM):

Support Vector Machine is an extremely popular supervised machine learning technique (having a pre-defined target variable) which can be used as a classifier as well as a predictor. For classification, it finds a hyper-plane in the feature space that differentiates between the classes. An SVM model represents the training data points as points in the feature space, mapped in such a way that points belonging to separate classes are segregated by a margin as wide as possible. The test data points are then mapped into that same space and are classified based on which side of the margin they fall.

Neural networks:

Artificial neural networks are computing that were inspired by biological neural networks of the animals' brains (though many scientists believe that actual brains are much more complex systems than artificial neural networks it has many more units, signals are transferred differently etc). The artificial neural networks consist of units (generally grouped by layers) and connections between them. Each of these connections has a corresponding weight, which is modified during learning process. There is no formal definition of the deep neural network, but usually, it is assumed that a neural network is deep if it has more than one hidden layer (not input or output layer).

Flow chart:


Start
Login
Fail
yes
Load Data set
Data Normalization
Spliting dataset
Training Model
Detection

4.DESIGN

4.1 INTRODUCTION

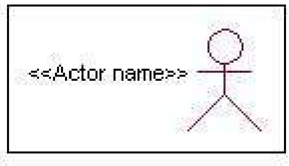
The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

Global Use Case Diagrams:

Identification of actors:

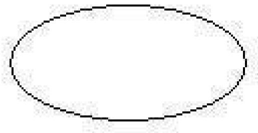
An actor represents the role a user plays with respect to the system. An actor interacts with but has no control over the use cases.

Graphical representation:



Usecase A use case can be described as a specific way of using the system from a user's (actor's) perspective.

Graphical representation



A more detailed description might characterize a use case as:

The pattern of behavior the system exhibits

- A sequence of related transactions performed by an actor and the system
- Delivering something of value to the actor

Use cases provide a means to:

- capture system requirements
- communicate with the end-users and domain experts
- test the sys Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Flow of Events

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. The flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

- When and how the use case starts and ends
- Use case/actor interactions
- Data needed by the use case
- Normal sequence of events for the use case
- Alternate or exceptional flows

Construction of Usecase diagrams:

Use-case diagrams graphically depict system behavior (use cases). These diagrams present a high-level view of how the system is used as viewed from an outsider's (actor's) perspective. A use-case diagram may depict all or some of the use cases of a system.

A use-case diagram can contain:

- actors ("things" outside the system)
- use cases (system boundaries identifying what the system should do)
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

Relationships in use cases:

1. Communication:

The communication relationship of an actor in a usecase is shown by connecting the actor symbol to the usecase symbol with a solid path. The actor is said to communicate with the usecase.

2. Uses:

A Uses relationship between the usecases is shown by the generalization arrow from the usecase.

3. Extends:

The extend relationship is used when we have one usecase that is similar to another usecase but does a bit more. In essence, it is like subclass.

Sequence Diagrams

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence that happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

Object:

An object has state, behavior, and identity. The structure and behavior of similar objects are defined in their common class. Each object in a diagram indicates some instance of a class. An object that is not named is referred to as a class instance.

The object icon is similar to a class icon except that the name is underlined: An object's concurrency is defined by the concurrency of its class.

Message:

A message is a communication carried between two objects that trigger an event. A message carries information from the source focus of control to the destination focus of control. The synchronization of a message can be modified through the message specification. Synchronization means a message where the sending object pauses to wait for results.

Link:

A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes. The existence of a relationship between two classes symbolizes a path of communication between instances of the classes: one object may send messages to another. The link is depicted as a straight line between objects or objects and class instances in a collaboration diagram. If the object links to itself, use the loop version of the icon.

Class Diagram:

A class is a set of objects that share a common structure and common behavior (the same attributes, operations, relationships, and semantics). A class is an abstraction of real-world items.

4.2 DFD/ER/UML DIAGRAM

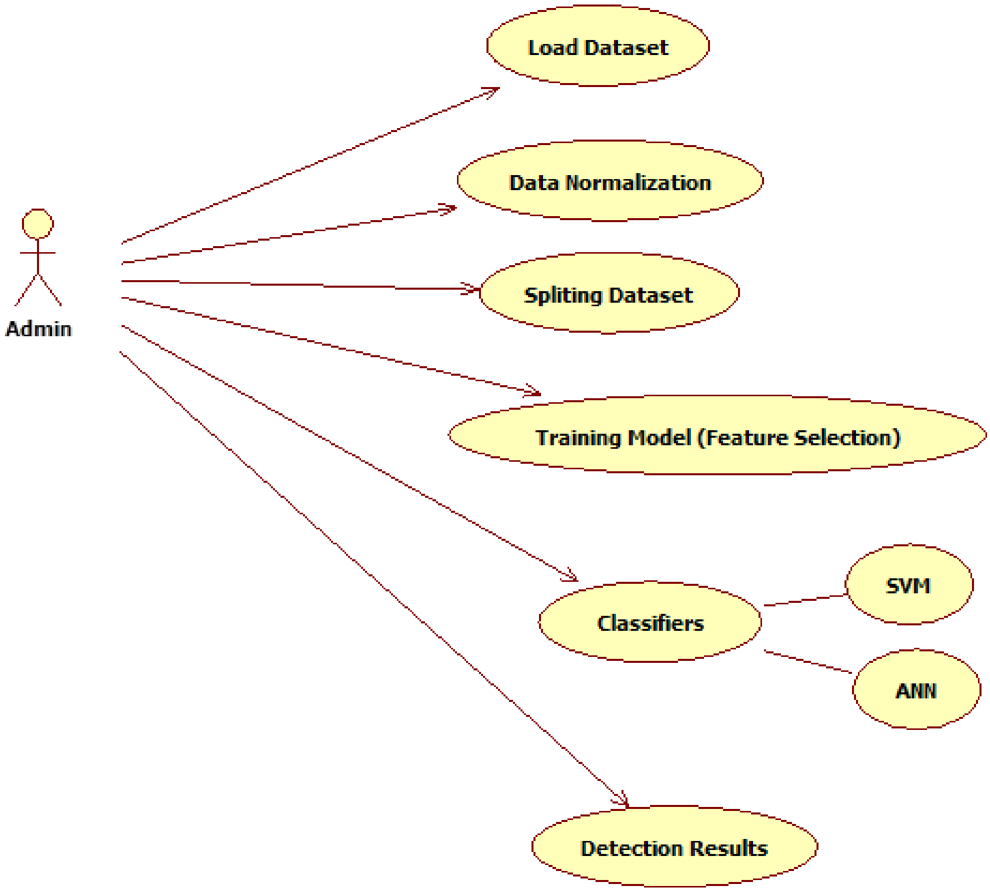
Class

Fig:-4 Project Class

Description:

In PC code planning, a class plot inside the Unified Modeling Language we used in our Project Development is Star (UML) could be a sort of static structure outline that delineates the structure of a system by showing the system's characterizations, their qualities, operations (or methodologies), and moreover the associations among the groupings. It elucidates that the class contains data.

Use case:



Description:

A usage case outline inside the Unified Modeling Language we used in our Project Development is Star (UML) could be a sort of behavioral chart portrayed out by and produced using a Use-case examination. Its inspiration is to gift a graphical layout of the presence of mind given by a system to the extent performing specialists, their targets (addressed as use cases), and any conditions between those use cases. The most explanation behind a use case diagram is to show what structure limits are played out that on-screen character. Parts of the entertainers inside the system will be a diagram.

User Sequence:

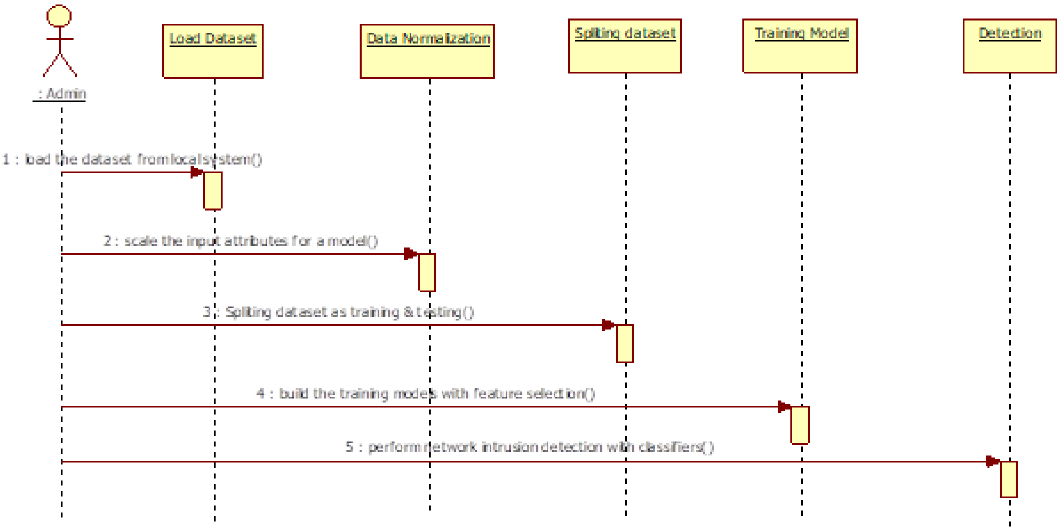

: Admin
Load Dataset
Data Normalization
Spliting dataset
Training Model
Detection
1 : load the dataset from localsystem()
2 : scale the input attributes for a model()
3 : Spliting dataset as training & testing()
4 : build the training models with feature selection()
5 : perform network intrusion detection with classifiers()

Fig:-6 Project User Sequence

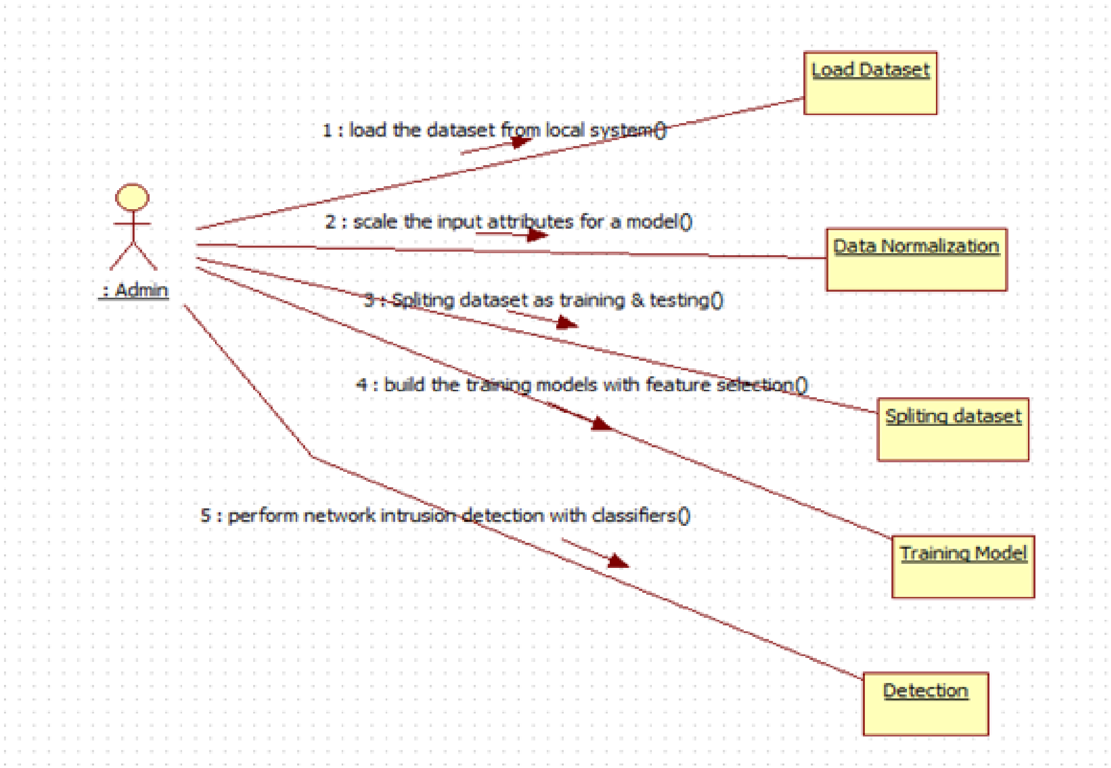
Description:

A gathering diagram in Unified Modeling Language we used in our Project Development is Star (UML) could be a sensible association graph that shows however frames work with each other and in what mastermind. It's a creation of a Message Sequence Chart. Progression diagrams are regularly known as event plots, event conditions, and short-lived approach outlines.

Collaboration:

Description:

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 (UML). These **diagrams** can be used to portray the dynamic behavior of a particular use case and define the role of each object.

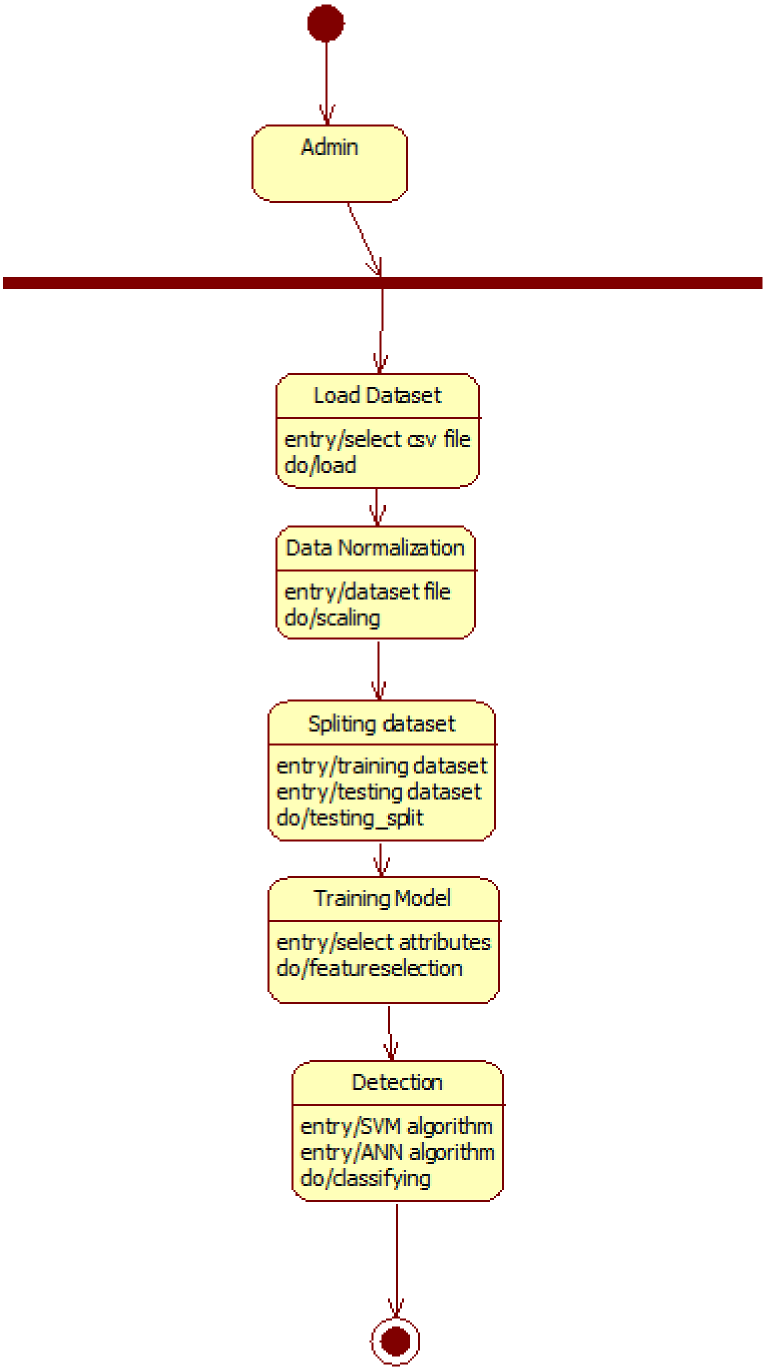

Load Dataset
1 : load the dataset from local system()
: Admin
2 : scale the input attributes for a model()
Data Normalization
3 : Spliting dataset as training & testing()
4 : build the training models with feature selection()
Spliting dataset
5 : perform network intrusion detection with classifiers()
Training Model
Detection

State Chart:

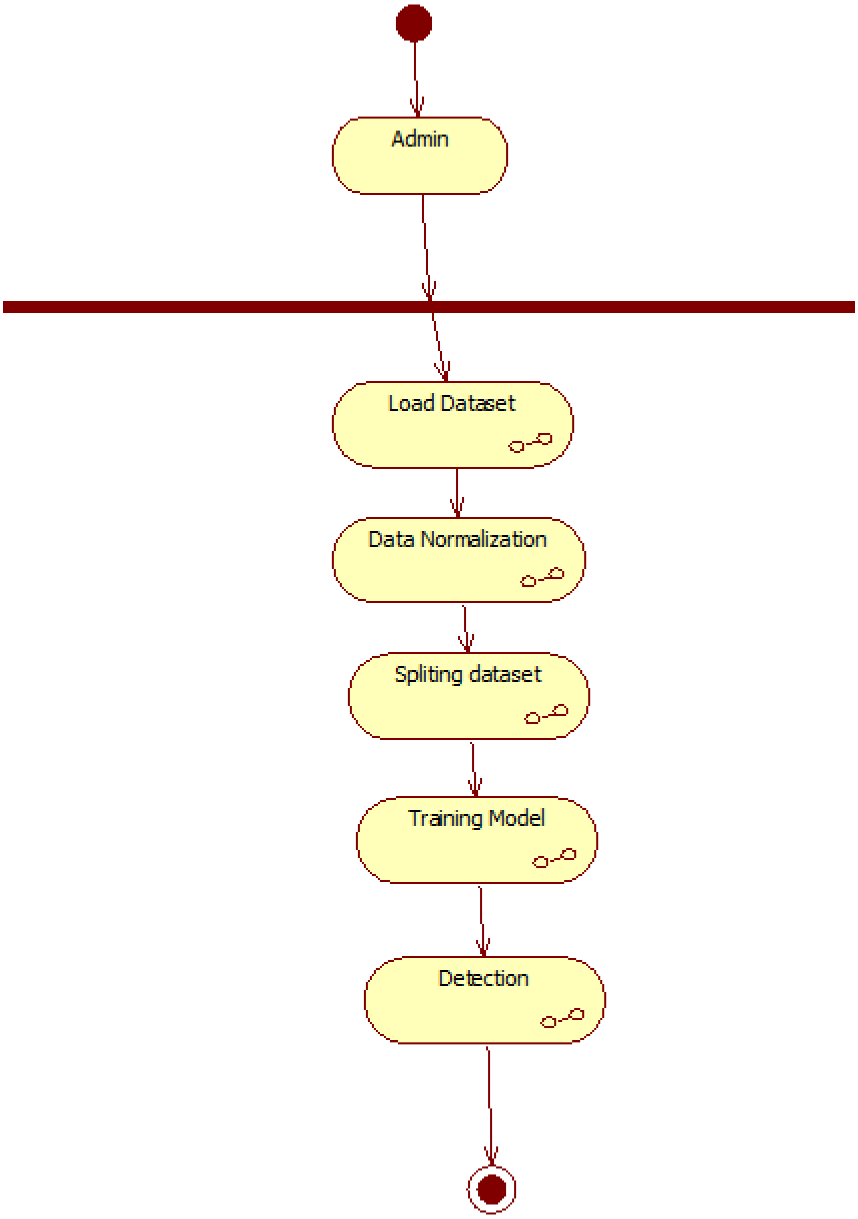
Fig:-9 Project State Chart

Description:

state graph (state machine define or statechart chart) A state graph, likewise referred to as a state machine graph or statechart define, could be an illustration of the states an issue will accomplish and additionally the advances between those states within the Unified Modeling Language we used in our Project Development is Star (UML).


Admin
Load Dataset
entry/select csv file
do/load
Data Normalization
entry/dataset file
do/scaling
Spliting dataset
entry/training dataset
entry/testing dataset
do/testing_split
Training Model
entry/select attributes
do/featureselection
Detection
entry/SVM algorithm
entry/ANN algorithm
do/classifying

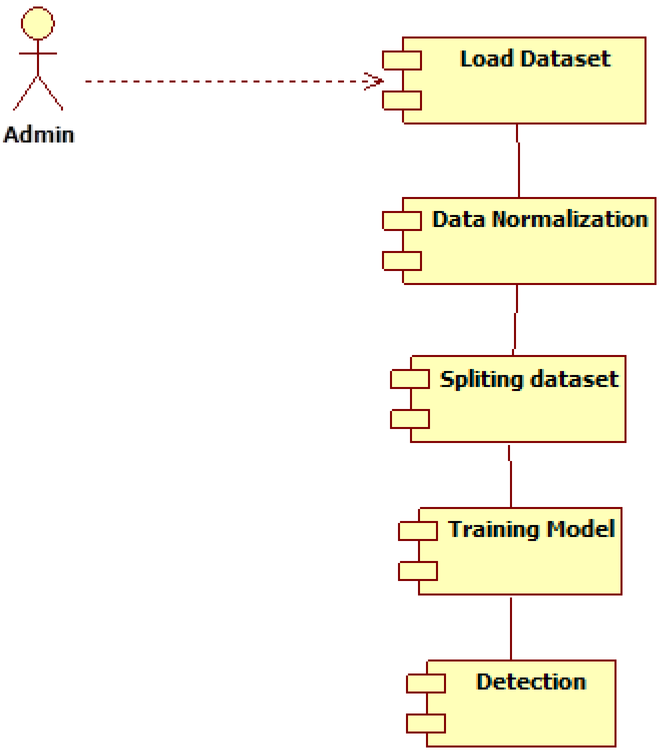
Activity:



Description:

Activity outlines are graphical depictions of work procedures of stepwise activities and exercises with help for the decision, cycle, and synchronization. Inside the Unified Modeling Language we used in our Project Development is Star, developments layouts will be regular depict the business and operational in little stages work procedures of parts in the midst of a structure. Relate in Nursing activity diagram shows the flood of organization.

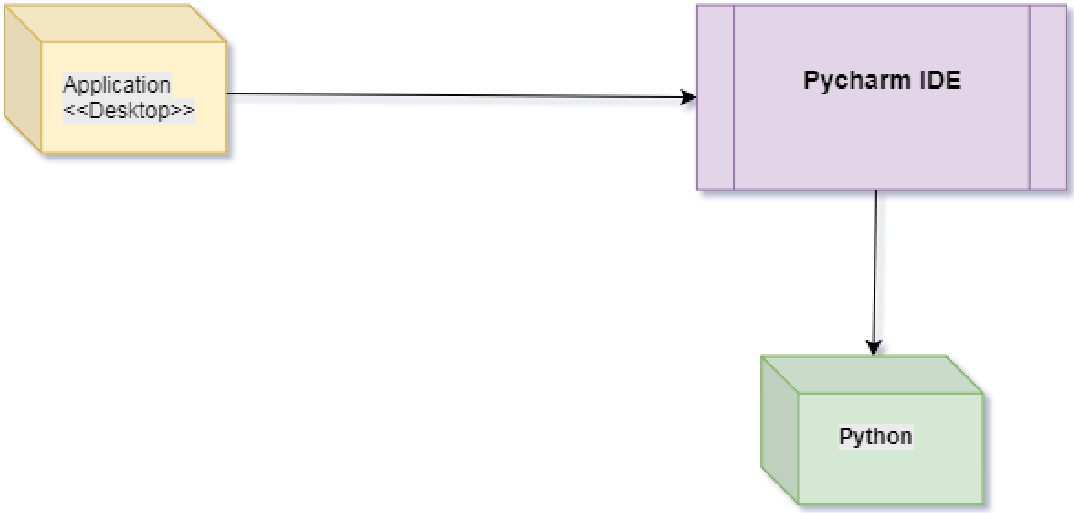
Component:



Description

A part graph otherwise referred to as AN UML phase chart depicts the association and wiring of the physical division in a very framework. Phase graphs are oftentimes interested in facilitating demonstrate usage points of interest and twofold watch that every a part of the framework's needed capacities is secured by organized improvement. Within the main style of UML, elements incorporated into these charts were physical: archives, information table, documents, and executable, each physical element with a region. Within the realm of UML a pair of, these elements aren't most physical however rather a lot of calculated stay solitary define parts, for instance, a business procedure that provides or expects interfaces to speak with completely different develops within the framework.

Deployment:



Description:

A UML **deployment diagram** is a **diagram** that shows the configuration of run-time processing nodes and the components that live on them. **Deployment diagrams** are a kind of structure **diagram** used in modeling the physical aspects of an object-oriented system.

4.3 MODULE DESIGN AND ORGANIZATION

Feature Selection:

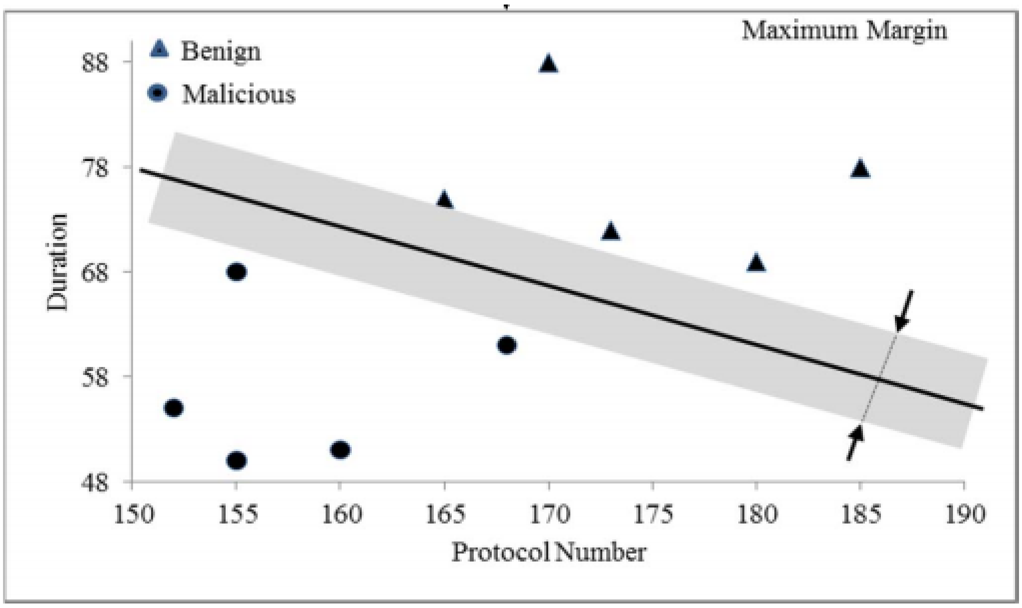
Feature selection is an important part of machine learning to reduce data dimensionality and extensive research carried out for a reliable feature selection method. For feature selection filter method and the wrapper method have been used. In the filter method, features are selected on the basis of their scores in various statistical tests that measure the relevance of features by their correlation with the dependent variable or outcome variable. The wrapper method finds a subset of features by measuring the usefulness of a subset of the feature with the dependent variable. Hence filter methods are independent of any machine learning algorithm whereas in the wrapper method the best feature subset selected depends on the machine learning algorithm used to train the model.

Building Machine Intelligence:

Based on the best features found in the feature selection process, learning models are developed. To develop the learning model, the machine learning algorithm is used. The training dataset is used to train the algorithm with the selected features. In supervised machine learning, each instance in the training dataset has the class it belongs to. The algorithm builds the learning model based on which machine learning algorithm is being used.

Support Vector Machine (SVM):

In SVM a separating hyperplane defines the classifier depending on the type of problem and available datasets. In the case where dataset is one dimensional, the hyperplane is a point, for two-dimensional data, it is a separating line as shown in below Figure.



Artificial Neural Network (ANN):

Artificial Neural Network is another tool used in machine learning. As its name suggests, ANN is a system inspired by human brain system and replicates the learning system of human brain. It consists of input and output layers with one or more hidden layers in most cases as shown in Figure. The ANN uses a technique called backpropagation to adjust the outcome with the expected result or class.

5. IMPLEMENTATION & RESULTS

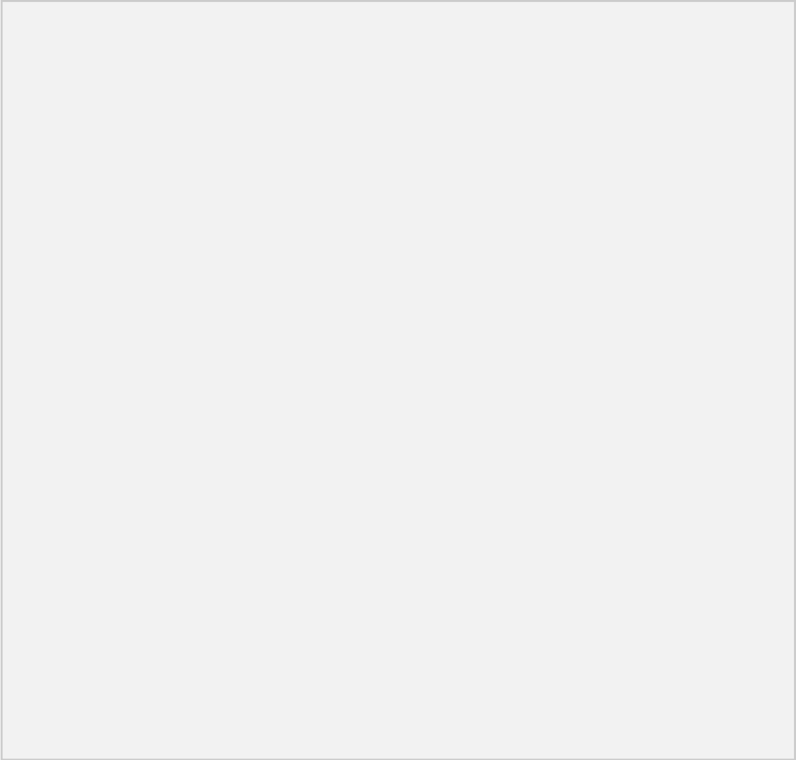
5.1 Introduction

The system proposed is composed of feature selection and learning algorithms. The feature selection component is responsible to extract most relevant features or attributes to identify the instance to a particular group or class.

5.2 Explanation Of Key Functions

The learning algorithm component builds the necessary intelligence or knowledge using the result found from the feature selection component. Using the training dataset, the model gets trained and builds its intelligence. Then the learned intelligence is applied to the testing dataset to measure the accuracy of home much the model correctly classified on unseen data.

5.3 Method of Implementation



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.

Input as CSV File

Reading data from CSV(comma separated values) is a fundamental necessity in Data Science. Often, we get data from various sources which can get exported to CSV format so that they can be used by other systems. The Panadas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

The CSV file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv. You can create this file using windows notepad by copying and pasting this data. Save the file as input.csv using the save As All files(*.*) option in notepad.

```
import pandas as pd

data = pd.read_csv('path/input.csv')

print (data)
```

Operations using NumPy

NumPy is a Python package that stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Using NumPy, a developer can perform the following operations –

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear aglebra. NumPy has in-built functions for linear algebra and random number generation.

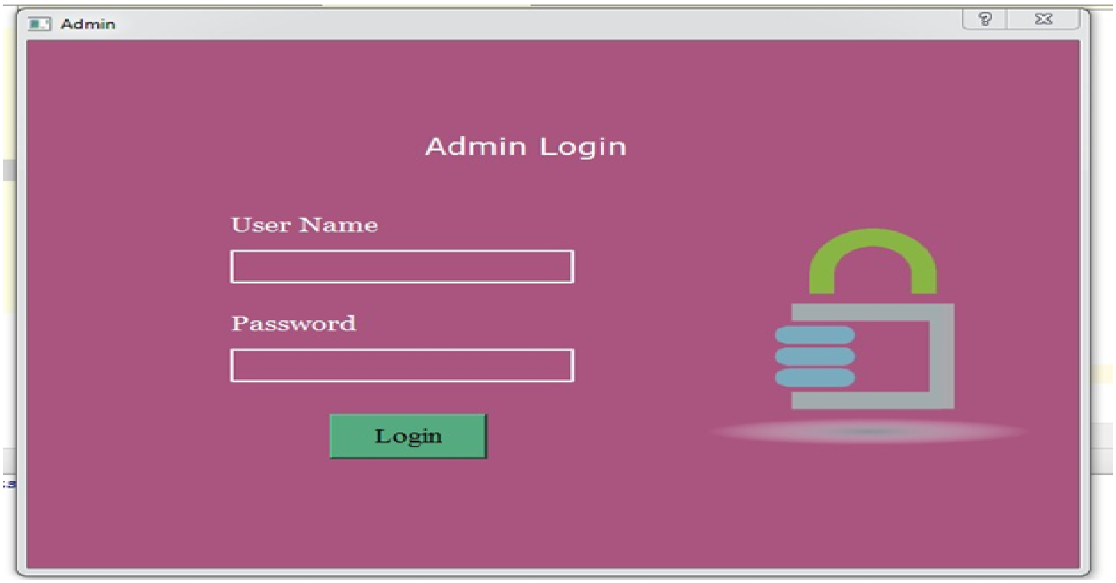
Key Features of Pandas

- Fast and efficient DataFrame object with the default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, indexing, and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High-performance merging and joining of data.
- Time Series functionality.

5.2.2 Output Screens:

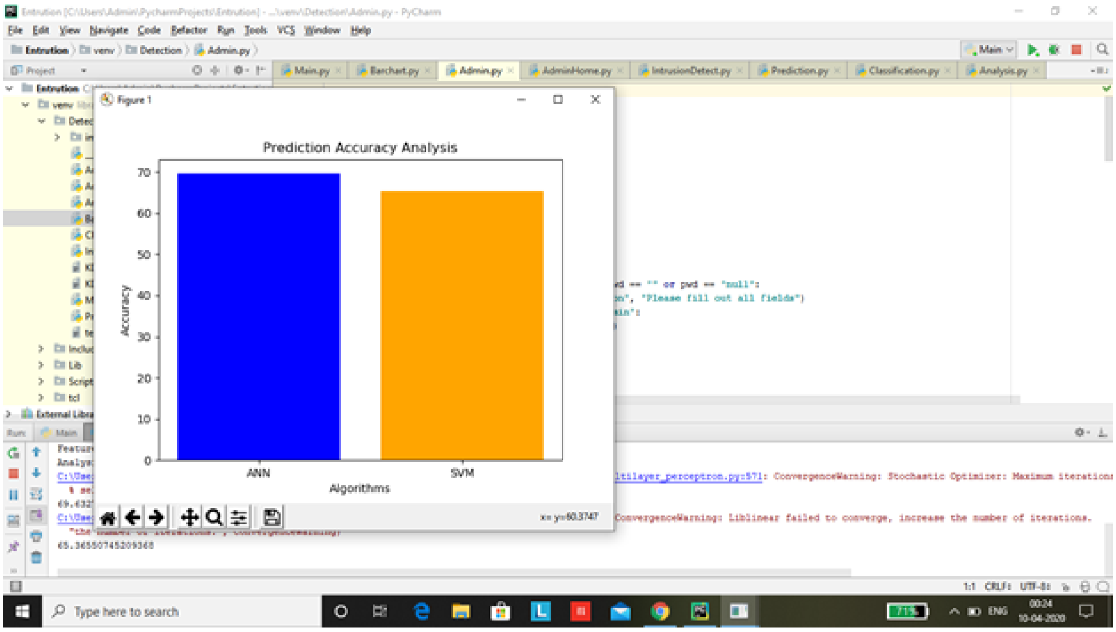
Step 1: On successful execution of code the main tab appears

Step 2: Then on pressing to the right of this main tab a corresponding admin login. The credential tab opens with user name and passport set as admin for our convenience



Step 3: On successful entering of credentials an admin Home tab with network intrusion detection and classification analysis are present

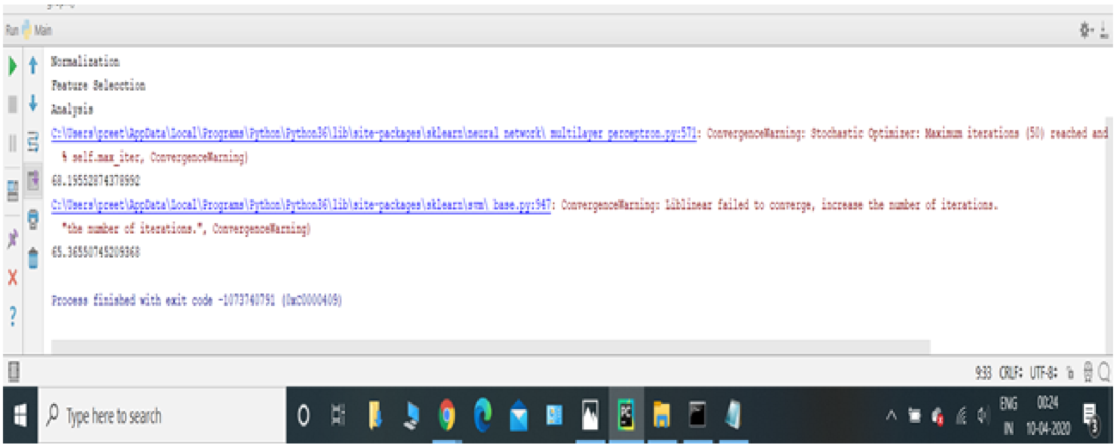
Network Intrusion detection gives us the information of the type of different attacks network is vulnerable to where as classification Analysis shows us the percentage of an attack that might happen in the command prompt or the ide console


Entrution [C:\Users\Admin\PycharmProjects\Entrution] - ...\venv\Detection\Admin.py - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Main.py
Barchart.py
Admin.py
AdminHome.py
IntrusionDetect.py
Prediction.py
Classification.py
Analysis.py
Figure 1
Prediction Accuracy Analysis
Accuracy
Algorithms
ANN
SVM
0
10
20
30
40
50
60
70
x= y=60.3747
Type here to search

```
d == "" or pwd == "null":
n", "Please fill out all fields")
in":
```

```
ltilayer_perceptron.py:571: ConvergenceWarning: Stochastic Optimizer: Maximum iteration
ConvergenceWarning: Liblinear failed to converge, increase the number of iterations.
65.36550745209368
```

Step 4: These screenshots do show the required outputs



```
Normalization
Feature Selection
Analysis
C:\Users\preet\AppData\Local\Programs\Python\Python36\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:571: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (50) reached and
  % self.max_iter, ConvergenceWarning)
68.19552874378992
C:\Users\preet\AppData\Local\Programs\Python\Python36\lib\site-packages\sklearn\svm\_base.py:947: ConvergenceWarning: Liblinear failed to converge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
65.36550745209368

Process finished with exit code -1073740791 (0xC0000409)
```

5.2.3 Result Analysis :

The above screen recordings gives us knowledge about the different types of network attacks that might happen along percentage. Along with the following aspect this also generate an bar chart to depict the accuracy of two algorithms in identify the attacks

6.TESTING & VALIDATION

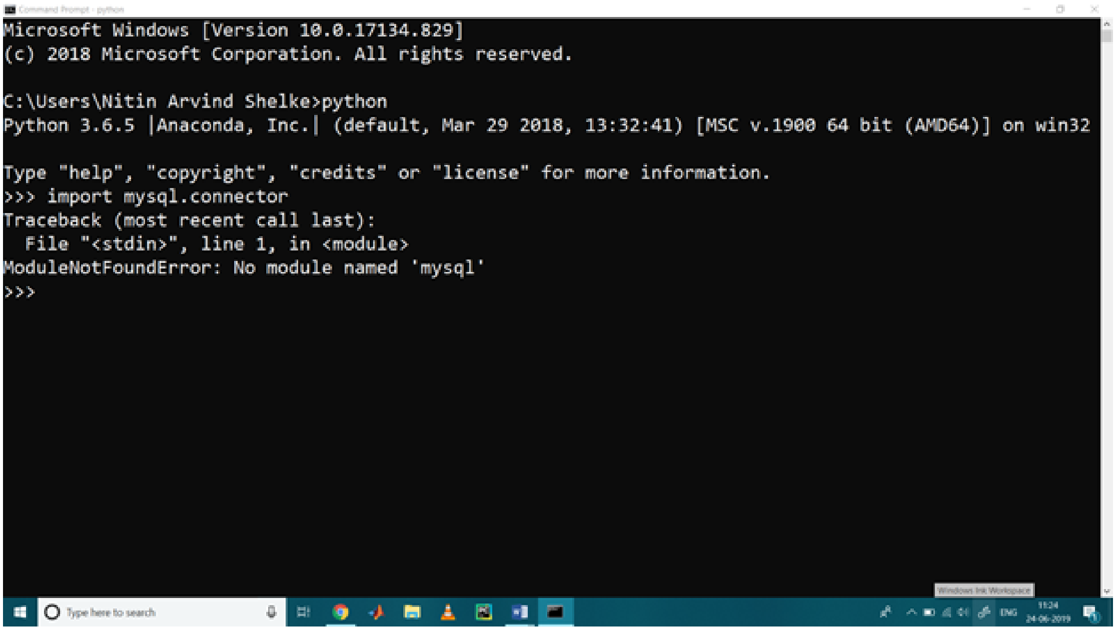
6.1 Introduction

Software testing is one of the main stages of project development life cycle to provide our cessation utilize with information about the quality of the application and ours, in our Project we have undergone some stages of testing like unit testing where it’s done in the development stage of the project when we are in the implementation of the application after the Project is yare we have done manual testing with different Case of all the different modules in the application we have even done browser compatibility testing in different web browsers in the market, even we have done Client-side validation testing on our application

Unit Testing

The unit testing is done in the stage of implementation of the project only the error are solved in development stage some of the error we come across in development are given below

When the pyqt5 module is not found the error will come as shown in fig

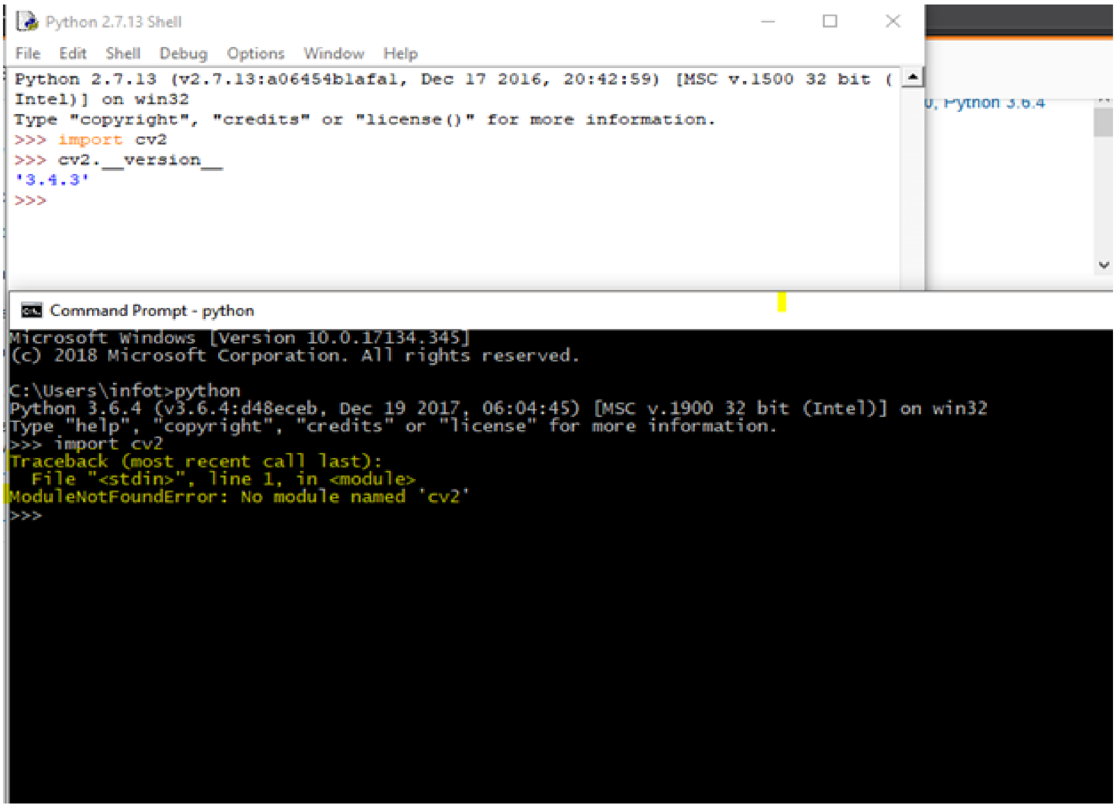


```
Microsoft Windows [Version 10.0.17134.829]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Nitin Arvind Shelke>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.
>>> import mysql.connector
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'mysql'
>>>
```

When the mysql module is not found the error will come as shown in fig



```
Python 2.7.13 (v2.7.13:a06454b1a1a1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import cv2
>>> cv2.__version__
'3.4.3'
>>>
```

```
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\infot>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'cv2'
>>>
```

When the cv2 module is not found the error will come as shown in fig

Manual testing on project application is done as shown in the above figures

6.2 Design of Test Case Scenarios

Code testing

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing

Executing this specification starting what the program should do and how it should be performed under various conditions. Test cases for the various situations and combinations of conditions in all the modules are tested.

Unit testing:

In unit testing, we test each module individually and integrate it with the overall system. Unit testing focuses on verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during the programming stage. In the testing step, each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example, the validation check is done for varying the user input given by the user in which the validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

BLACK BOX TESTING

Black box testing is a software testing technique in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

Input
Black Box
Output

The above Black Box can be any software system you want to test. For example an operating system like Windows, a website like Google, a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

Black box testing - Steps

Here are the generic steps followed to carry out any type of Black Box Testing.

- Initially requirements and specifications of the system are examined.
- Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

Types of Black Box Testing

There are many types of Black Box Testing but the following are the prominent ones

- **Functional testing** – This black box testing type is related to the functional requirements of a system; it is done by software testers.
- **Non-functional testing** – This type of black-box testing is not related to testing of specific functionality, but non-functional requirements such as performance, scalability, usability.
- **Regression testing** – Regression testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

WHITE BOX TESTING

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability. White box testing is also known as **clear, open, structural, and glass box testing**.

It is one of two parts of the **"box testing" approach** of software testing. Its counter-part, black box testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "Whitebox" was used because of the see-through box concept. The clear box or white box name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

What do you verify in White Box Testing ?

White box testing involves the testing of the software code for the following:

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code

System testing:

Once the individual module testing is completed, modules are assembled and integrated to perform as a system. The top-down testing, which began from the upper level to lower-level module, was carried out to check whether the entire system is performing satisfactorily.

There are three main kinds of System testing:

i. Alpha Testing

ii. Beta Testing

iii. Acceptance Testing

Alpha Testing:

This refers to the system testing that is carried out by the test team with the Organization.

Beta Testing:

This refers to the system testing that is performed by a selected group of friendly customers.

Acceptance Testing:

This refers to the system testing that is performed by the customer to determine whether or not to accept the delivery of the system.

Integration Testing:

Data can be lost across an interface, one module can have an adverse effort on the other sub-functions when combined, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The developed system has run successfully for this sample data. The need for the integrated test is to find the overall system performance.

Output testing:

After the performance of the validation testing, the next step is output testing. The output displayed or generated by the system under consideration is tested by asking the user about the format required by the system.

7.CONCLUSION

We have presented different machine learning models using different machine learning algorithms and different feature selection methods to find the best model. The analysis of the result shows that the model built using ANN and wrapper feature selection outperformed all other models in classifying network traffic correctly with a detection rate of 94.02%. We believe that these findings will contribute to research further in the domain of building a detection system that can detect known attacks as well as novel attacks. The intrusion detection system that exists today can only detect known attacks. Detecting new attacks or zero-day attacks still remains a research topic due to the high positive rate of the existing systems.

References

[1] H. Song, M. J. Lynch, and J. K. Cochran, “A macro-social exploratory analysis of the rate of interstate cyber-victimization,” American Journal of Criminal Justice, vol. 41, no. 3, pp. 583–601, 2016.

[2] P. Alaei and F. Noorbehbahani, “Incremental anomaly-based intrusion detection system using limited labeled data,” in Web Research (ICWR), 2017 3th International Conference on, 2017, pp. 178–184.

[3] M. Saber, S. Chadli, M. Emharraf, and I. El Farissi, “Modeling and implementation approach to evaluate the intrusion detection system,” in International Conference on Networked Systems, 2015, pp. 513–517.