A MINI-PROJECT (2) REPORT ON

# ACCURATE STRESS DETECTION OF ELDERLY PATIENTS USING MACHINE LEARNING ALGORITHMS

**Submitted in partial fulfillment of requirements for the award of the degree of**

**BACHELOR OF TECHNOLOGY**
**IN**
**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by:**

| | |
|---|---|
| G. THEJA SEKHAR REDDY | (19091A05G7) |
| P. PAVAN | (19091A0596) |
| N. LOKESHWAR REDDY | (20095A0506) |
| S. SHAHID BASHA | (20095A0516) |

**Under the Guidance of**
**Ms. D. KARISHMA M. Tech**

**Designation, Dept. of CSE**

EDUCATION FOR PEACE
(ESTD-1995)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY**
**(AUTONOMOUS)**
AFFILIATED TO J.N.T UNIVERSITY ANANTAPUR. ACCREDITED BY NBA (TIER-1) & NAAC OF UGC. NEW DELHI, WITH A+ GRADE
RECOGNIZED UGC-DDU KAUSHAL KENDRA
NANDYAL-518501, (Estd-1995)
**YEAR: 2022-2023**

**Rajeev Gandhi Memorial College of Engineering &Technology**

**(AUTONOMOUS)**

AFFILIATED TO J.N.T UNIVERSITY ANANTAPUR. ACCREDITED BY NBA (TIER-1) & NAAC OF UGC. NEW DELHI, WITH A+ GRADE
RECOGNIZED UGC-DDU KAUSHAL KENDRA
NANDYAL-518501, (Estd-1995)

**(ESTD – 1995)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

# CERTIFICATE

This is to certify that **G. THEJA SEKHAR REDDY** (19091A05G7), **P. PAVAN** (19091A0596), **N. LOKESHWAR REDDY**(20095A0506) and **S. SHAHID BASHA** (20095A0516) of IV- B. Tech I- semester, have carried out the mini-project (2) work entitled "**ACCURATE STRESS DETECTION OF ELDERLY PATIENTS USING MACHINE LEARNING ALGORITHMS**" under the supervision and guidance of **Ms. D. KARISHMA,** Assistant Professor, CSE Department, in partial fulfillment of the requirements for the award of Degree of **Bachelor of Technology** in **Computer Science and Engineering** from **Rajeev Gandhi Memorial College of Engineering & Technology (Autonomous)**, Nandyal is a bonafied record of the work done by them during 2021-2022.

| | |
|---|---|
| **Project Guide** | **Head of the Department** |
| **Ms. D. Karishma** **M. Tech.** | **Dr. K. Subba Reddy** **M.Tech, Ph.D.** |
| **Assistant Professor, Dept. of CSE** | **Professor, Dept. of CSE** |

**Place:** Nandyal
**Date:**

**External Examiner**

## *Candidate's Declaration*

We hereby declare that that the work done in this project entitled **"ACCURATE STRESS DETECTION OF ELDERLY PATIENTS USING MACHINE ALGORITHMS"** submitted towards completion of mini-project (2) in *IV Year I Semester of B. Tech (CSE)* at the **Rajeev Gandhi Memorial College of Engineering & Technology**, Nandyal. It is an authentic record our original work done under the esteemed guidance of **Ms. D. Karishma,** Assistant Professor, department of **COMPUTER SCIENCE AND ENGINEERING**, RGMCET, Nandyal.

We have not submitted the matter embodied in this report for the award of any other Degree in any other institutions.

**By**

| | |
|---|---|
| **G. Theja Sekhar Reddy** | (19091A05G7) |
| **P. Pavan** | (19091A0596) |
| **N. Lokeshwar Reddy** | (20095A0506) |
| **S. Shahid Basha** | (20095A0516) |

Dept. of CSE,
RGMCET.

**Place:** Nandyal
**Date:**

# ACKNOWLEDGEMENT

We manifest our heartier thankfulness pertaining to your contentment over our project guide **Ms. D. Karishma,** Assistant Professor of Computer Science Engineering department, with whose adroit concomitance the excellence has been exemplified in bringing out this project to work with artistry.

We express our gratitude to **Dr. K. Subba Reddy garu,** Head of the Department of Computer Science Engineering department, all teaching and non-teaching staff of the Computer Science Engineering department of Rajeev Gandhi memorial College of Engineering and Technology for providing continuous encouragement and cooperation at various steps of our project.

Involuntarily, we are perspicuous to divulge our sincere gratefulness to our Principal, **Dr. T. Jaya Chandra Prasad garu,** who has been observed posing valiance in abundance towards our individuality to acknowledge our project work tangentially.

At the outset we thank our honourable **Chairman Dr. M. Santhi Ramudu garu,** for providing us with exceptional faculty and moral support throughout the course.

Finally we extend our sincere thanks to all the **Staff Members** of CSE Department who have co-operated and encouraged us in making our project successful.

Whatever one does, whatever one achieves, the first credit goes to the **Parents** be it not for their love and affection, nothing would have been responsible. We see in every good that happens to us their love and blessings.

**BY**

| | |
|---|---|
| **G. Theja Sekhar Reddy** | (19091A05G7) |
| **P. Pavan** | (19091A0596) |
| **N. Lokeshwar Reddy** | (20095A0506) |
| **S. Shahid Basha** | (20095A0516) |

# ABSTRACT

In these fast-paced times, psychological health issues like stress, anxiety, and depression became quite common among people. The symptoms of stress are feeling upset or agitated, an inability to relax, low energy levels, chronic headaches, frequent overreaction, and persistent colds or infections. Due to mental health issues, one-third of the population in the world undergoes depression, suicide (Shafiee et al. 2020).

This project aims to detect human mental stress in elderly patients using the K Nearest Neighbor (KNN) classifier. To detect mental stress, heart rate and blood pressure data is used. For this project we need a dataset to estimate the stress level. The SWELL dataset is publicly available on Kaggle website which comprises heart rate variability (HRV) indices computed from the multimodel SWELL knowledge work (SWELL-KW) dataset for research on stress and user modelling. From the dataset we consider 70% of data as trained dataset and 30 % of data as the comparison data. At last we compare accuracy between KNN and Navie Basis.

**Keywords:** *KNN, Naive Bayes, Machine Learning, Kaggle, Heart Rate, Blood Pressure..*

# CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

# 1. INTRODUCTION

## 1.1 Introduction:

In these fast-paced times, psychological health issues like stress, anxiety, and depression became quite common among people (Priya, Garg, and Tigga 2020). The symptoms of stress are feeling upset or agitated, an inability to relax, low energy levels, chronic headaches, frequent overreaction, and persistent colds or infections. Due to mental health issues, one-third of the population in the world undergoes depression, suicide (Shafiee et al. 2020). In this paper, the proposed work gives classification and detection of mental stress in the elderly using machine learning algorithms like KNN (K Nearest Neighbor) in comparison with Naive Bayes (P. Kumar, Garg, and Garg 2020). An increase in mental stress in people leads to major problems like hypertension, anxiety, chronic illness (Can et al. 2019a). The proposed work can be applied to students, employed and unemployed individuals, and elderly people (Ahuja and Banga 2019).

## 1.2 Objectives:

- Preventing and managing age-associated chronic diseases including mental, neurological and substance use disorders.
- To develop a long and short version of an index to measure experiences during hospitalization perceived by elderly patients as stressful.
- Based on the results of the system it will be helpful to patients to take care before any serious things happened.
- The system providers better results comparing to previous system and also helpful for doctors, how to treat the patients and also knows the patient current status.
- The comparison of both KNN and the Naïve Bayes Algorithm gives better accuracy in stress detection.

# 2. SYSTEM ANALYSIS

## 2.1 Existing System:

There is no software existing system but, a hardware system called Continuous Stress Detection Using Wearable Sensors in Real Life. The existing system performs only one kind of classification.

The hardware system needs to be wear all time in order to measure the stress levels. The sensors regularly sense the required things in order to calculate or perform the operations in order to get the person is under stress or not.

### 2.1.1 Disadvantages of Existing system:

- The existing system results are not that much accurate.
- This system uses only one classifier for stress level classification.
- The accuracy levels are low.

## 2.2 Motivation to the Problem:

The purpose of the project is to find or detect the stress of elderly patient using machine learning algorithms K-Nearest Neighbour and Naive Bayes and the accuracies are compared to provide better results. The objective is to predict the patient has stress or not by using the heart rate and blood pressure of the patient which are available on the dataset.

## 2.3 Proposed system:

It aims to detect human mental stress in elderly patients using machine learning algorithms. The dataset is over sampled using KNN (K Nearest Neighbour) and the algorithm classifier. After KNN classification we perform Naïve Bayes Classification by using its classifier. And calculate the accuracies mean from both classifiers results. Finally, we compare the accuracy differences between KNN and Naïve Bayes to deliver better results to the user.

### 2.3.1 Advantages of Introduced System:

- High accuracy on detection rate.
- Comparison with the Naïve Bayes algorithm gives better accuracy.
- Comparison of various algorithms can be done easily.

# 3. FESABILITY STUDY

Feasibility study is an important phase in the software development process. Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of thefeasibility study is to test the Technical, Solution and Economical feasibility for adding new modules and debugging old running system.

Feasibility study should be performed based on various criteria and parameters. The various feasibility studies are:

- Technical Feasibility
- Solution Feasibility
- Economic Feasibility

## 3.1 Technical Feasibility:

To check the technical feasibilities. the technical requirements of the system, Since machine learning algorithms are based on pure mathematics. There is very little requirement for any professional software. Most ofthe tools are open source. We can run this software in any software in any system without any software requirements which makes them highly portable.

## 3.2 Solution Feasibility:

To check the level of acceptance of the system by the user. This includes the process of providing solutionefficiently. The user must not feel threatened by the system. The objective is to predict network attack with betteraccuracy and to find suitable algorithm to detect intrusions.

## 3.3 Economic Feasibility:

Since the project is Machine Learning based, the cost spent in executing this project would not demand cost for software's and related products, as most of the products are open source and free to us. Our project must be minimal cost.

# 4. SYSTEM REQUIREMENT SPECIFICATION

## 4.1 Requirement Analysis:

Requirements Analysis is the process of defining the expectations of the users for an application that is to bebuilt or modified. It involves all the tasks that are conducted to identify the need of different stakeholders.

Requirement Analysis is a software engineering task that bridges the gap between system level software allocation and software design. Requirement Analysis is to specify software function and performance indicatessoftware interface with other system elements.

Requirement Analysis of the system starts with the specification given by the user. This user-required specification could be formal or informal. In formal method, the user clearly states the purpose of the software. This acts as the good basis for the software engineers for the requirement analysis. In informal method purpose and the outputs are not clearly specified by the user. The responsibility is on the software engineer to get the purpose and outputs by interacting more the user.

## 4.2 Software Requirements Specification (SRS):

A software requirements specification, a requirements specification for a software system is complete description of the behavior of a system to be developed and may include a set of use cases that describe interactions the user will have with the software. In addition, it also contains non-functional requirements. Non- functional requirements impose constraints on the design or implementation.

The software requirements specification document enlists all necessary requirements that are required for theproject development. To derive the requirements, we need to have clear and through understanding of the products to be developed. This is prepared after detailed communications with the project team and customer.

A Software Requirements Specification minimizes the time and effort required by developers to achieve desired goals and minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs, and human users in a wide variety of real-world situations.

## 4.3 Functional Requirements:

In software engineering a functional requirement defines as a function of system or its components where afunction is described as a specification of behavior between output and input. It provides the following actions.

**Technical Issues:**

Many software projects have failed due to an incomplete or inaccurate analysis process, especially technicalissues. It is a very key step in process.

**Risk Analysis:**

Project Risk Analysis is for the estimates of known accuracy and risk on capital investment projects. The main challenge is to determine how to model and visualize the complex relationships between normal data and data affected by attacks, define and monitor the risk impacts, analyse the probability of risk occurrence, mitigate the negative impact of risks, and monitor the course of the project with risks and uncertainties.

**Performance Requirements:**

The project has the following performance requirements :

- The prime requirement is that no error condition causes a project to exit abruptly.
- Any error occurred in any process should return an understandable error message.
- The response should be fast and accurate, the analysis should not be confused at any point of time about actionthat is happening.
- The system performance should be adequate.

## 4.4 Non-Functional Requirements:

**Reliability:**

The project is guaranteed to provide reliable results for every test dataset. The system shall operate with expectedpotency. The accuracy of chosen classifiers should be adequate and should be comparable with accuracy of other classifiers.

**Usability:**

The IDS developed can be used in many public and hybrid networks to detect intrusions that occur in them. Theclassifiers used should have good accuracy and provides results in reliable manner.

**Scalability:**

The need for scalability has been a driver for much of the technology innovations of the past few years. The system should perform with same accuracy even when performed on large test data sets. This adaption makes system scalable towards any kind of datasets.

**Maintainability:**

Maintainability is the ability to make changes to the product over time. The system should be maintainable to be consistent in performance. For example, it should be able to cope with another better classifiers in future to meetrequired performance and accuracy.

## 4.5 Software Requirements:

| | | |
|---|---|---|
| • Operating System | : | **Windows 10/11** |
| • IDE | : | **Google Colab** |
| • Programming Language | : | **PYTHON 3.X** |
| • Libraries | : | **NumPy, Pandas, matplotlib, seaborn, sklearn, SciPy** |

## 4.6 Hardware Requirements:

| | | |
|---|---|---|
| • Processor Type | : | **intel i3** |
| • RAM | : | **4GB DD2 RAM** |
| • Hard disk | : | **500 GB** |

# 5. SYSTEM DESIGN

Design is a meaningful engineering representation of something that is to be built. It is the most crucial phase in the developments of a system. Software design is a process through which the requirements are translatedinto a representation of software. The logical system design arrived at because of systems analysis is converted into physical system design.

## 5.1 Modules:

A module is defined as the unique and addressable components of the software which can be solved andmodified independently without disturbing (or affecting in very small amount) other modules of the software. Thus, very software design should follow modularity. The process of breaking down a software into multiple independent modules where each module is developed separately is called Modularization.

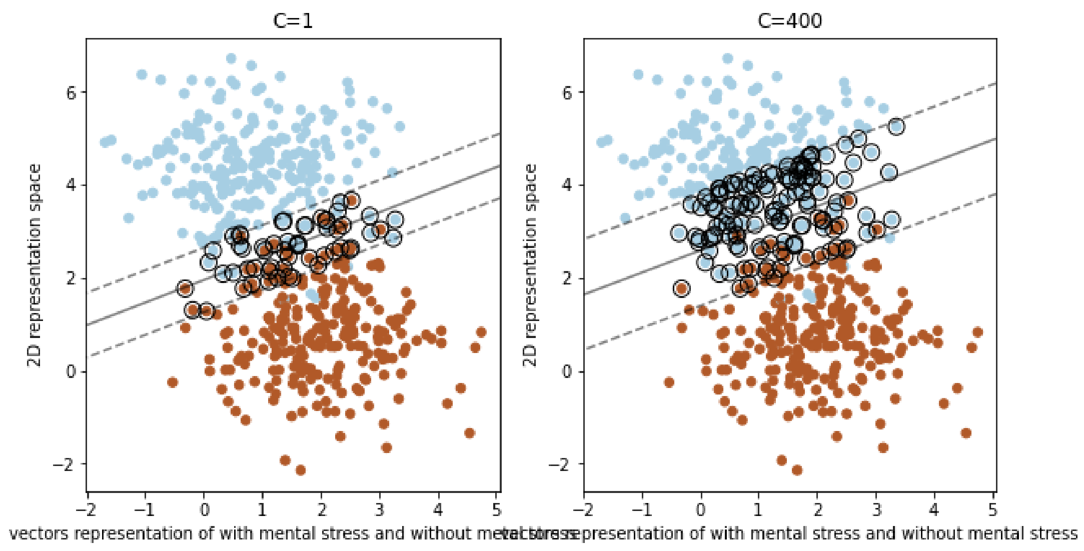
### 5.1.1 Data Preprocessing:

The dataset is explored, the categorical attributes are converted into numeric attributes by Label Encoding. The class label is explored, the attack types with negligible values are replaced with the label as ‘others.

normal = 0, neptune = 1, others = 2

### 5.1.2 Sampling Technique:

SVM or Support vector Machine is a linear model for classification and regression problems. The linear SVM technique is used to classify the data from data set. The idea of SVM is simple: The algorithm creates a line or a hyperplane which separates the data into classes.



### 5.1.3 Modeling:

KNN (K Nearest Neighbor):

KNN is a supervised machine learning algorithm used for solving regression and classification problems. It is a simple and non-parametric algorithm. The working of KNN is as follows:

- Load the data.
- Initialize K to the chosen number of neighbors.
- For each data in the dataset.
- Calculate the distance between the query data and the current data from the dataset.
- Add the distance and the index of the data to an ordered collection.
- Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.
- Pick the first K entries from the sorted collection.
- Get the labels of the selected K entries.
- If regression, return the mean of the K labels.
- If classification, return the mode of the K labels.

**Naïve Bayes Classification:**

Naive Bayes is a classification algorithm for binary (two-class) and multiclass classification problems. Novel Naive Bayes or idiot Bayes is used to calculate the probabilities for each class which is simplified to make their calculations tractable. The probability of the classes is calculated by the equation:

$$\mathbf{P(class|data) = (P(data|class) * P(class)) / P(data)}$$

Where P(class|data) is the probability of class to the provided data.

The working of Naïve Basis is:

- Separate into Class.
- Summarize Dataset.
- Summarize Data by Class.
- Gaussian Probability Density Function.
- Class Probabilities.

## 5.2 System Architecture:

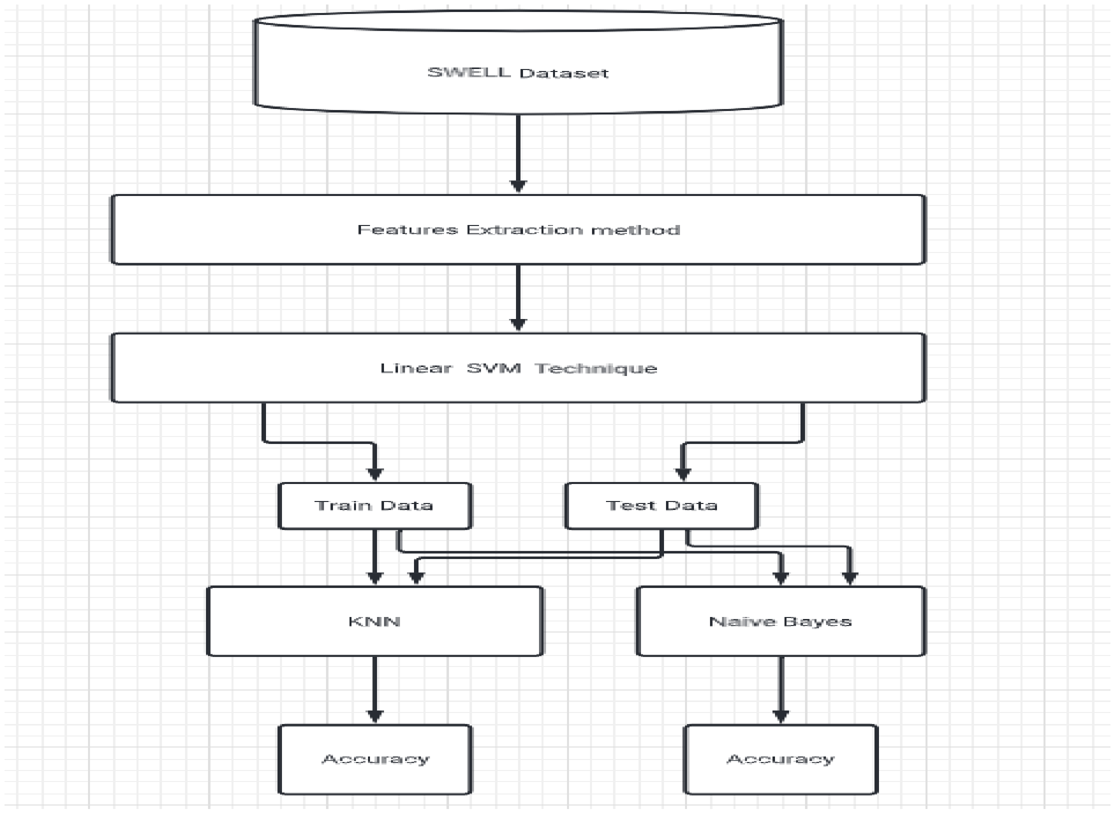


**Fig 5.1: System Architecture**

The stress detection model uses machine learning algorithms such as KNN (K Nearest Neighbor) and Naïve Bayes. The performance of the classifier was improved by optimizingthe algorithms for similarity and combining it with data imbalance processing techniques.

The architecture of the stress detection is as follows:

- The processed dataset is given for both training and testing. Data processing includes missing data removal, replacement of null values and float values with mean or median values, and standardization of data.
- By using Linear SVM we separate the dataset set as trained dataset and test dataset.
- The preprocessed dataset with features is given as input to KNN and Naive Bayes. From the total sample size, 70% of the data is given for training and the remaining 30% is given for testing.
- Both the datasets are used in this proposed work for group1 and group2 binary classification as given in Table 1.

- Table 1 represents the sample features and classes of the input dataset for analysis. The SWELL dataset contains 12 features with 2 different classes and the Biometrics for stress monitoring dataset contains 16 attributes with 2 classes.
- Finally, we compare the accuracies of both KNN Classifier and the Naïve Bayes Classifier.

## 5.3 Introduction to UML:

A UML diagram is a diagram based on the UML (Unified Modelling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts, or classes, in order to better understand, alter, maintain, or document information about the system.

Each UML diagram is designed to let developers and customers view a software system from a differentperspective and in varying degrees of abstraction. UML diagrams commonly created in visual modelling tools include.

### 5.3.1 Class Diagram:

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not onlyused for visualizing, describing, and documenting different aspects of a system but also for constructing executablecode of the software application. It describes the attributes and operations of a class and also the constraints imposed on the system

### 5.3.2 Use case Diagram:

A use case diagram at its simplest is a representation of user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by otherof diagrams as well. The use cases are represented by either circles or ellipses.

### 5.3.3 Activity Diagram:

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control is drawn from one operation to another. Thisflow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join etc.

### 5.3.4 Sequence Diagram:

Sequence diagrams in UML shows how object interact with each other and the order those interactions occur.It's important to note that show the interactions for a particular scenario. The processes are represented vertically,and interactions are show as arrows.

## 5.4 UML Diagrams:

The **figure 5.2** shows the use case diagram of the system. The system consists of mainly two actors or users interacts with it, they are Admin and Student. The use cases they involved and their interaction with the system are depicted in the below figure.

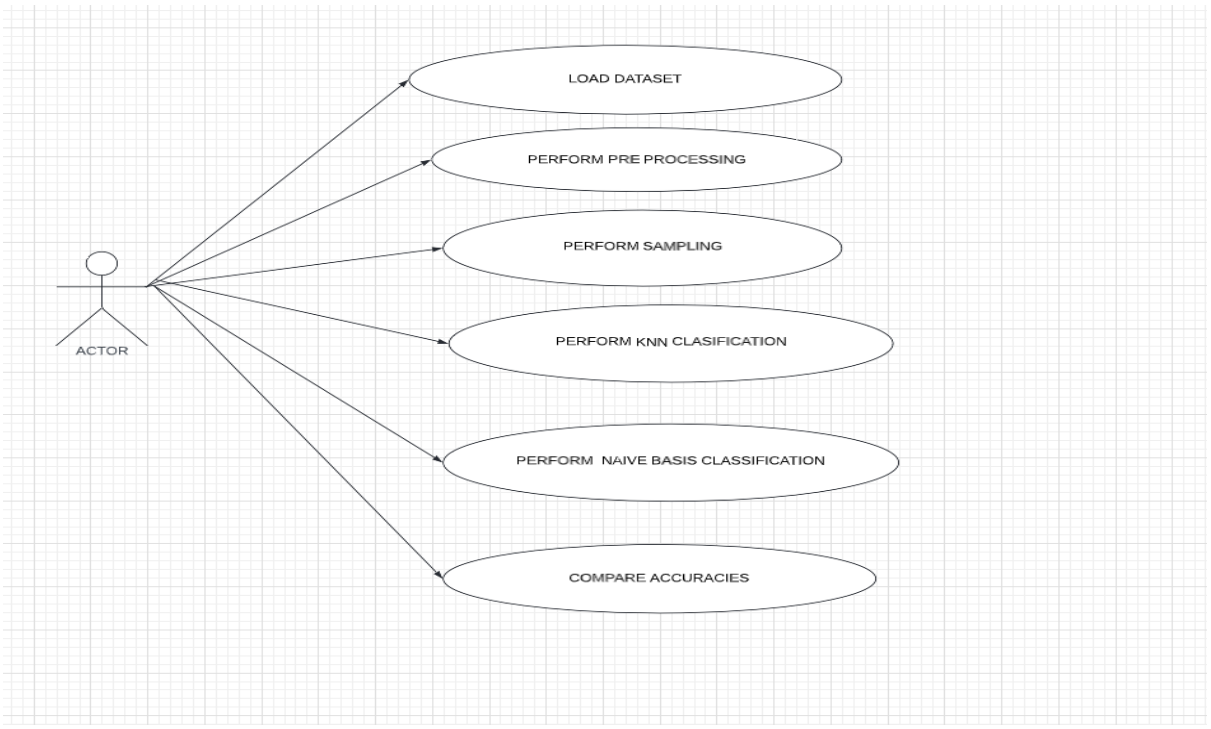


**Fig 5.2: Use Case Diagram**

The **figure 5.3** shows the class diagram of the system. It represents static view of application. The below figuredescribes attributes and operations of classes and the constraints imposed on the system.

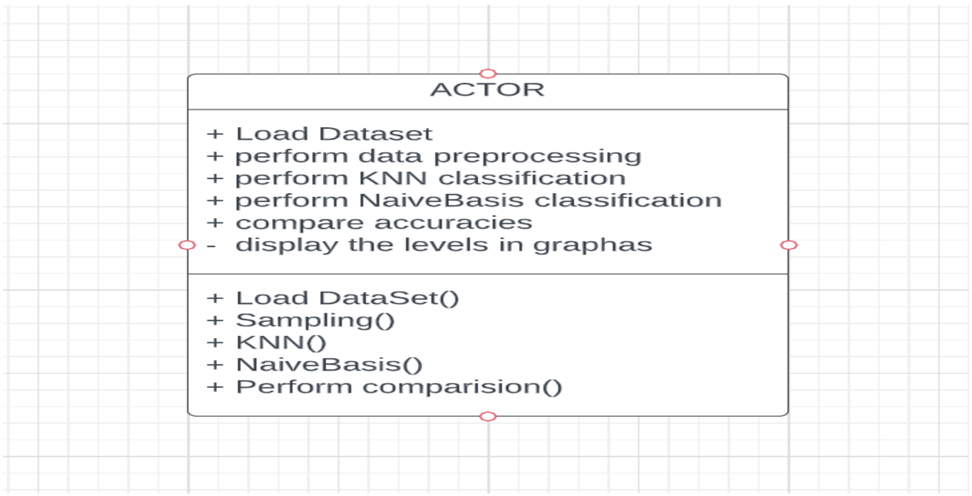

ACTOR
+ Load Dataset
+ perform data preprocessing
+ perform KNN classification
+ perform NaiveBasis classification
+ compare accuracies
- display the levels in graphas
+ Load DataSet()
+ Sampling()
+ KNN()
+ NaiveBasis()
+ Perform comparision()

**Fig 5.3: Class Diagram**

The **figure 5.4** shows the sequence diagram for admin module of the system. The below figure depicts the processes involved and the sequences of messages exchanged between the processes needed to carry out the functionality in admin module.

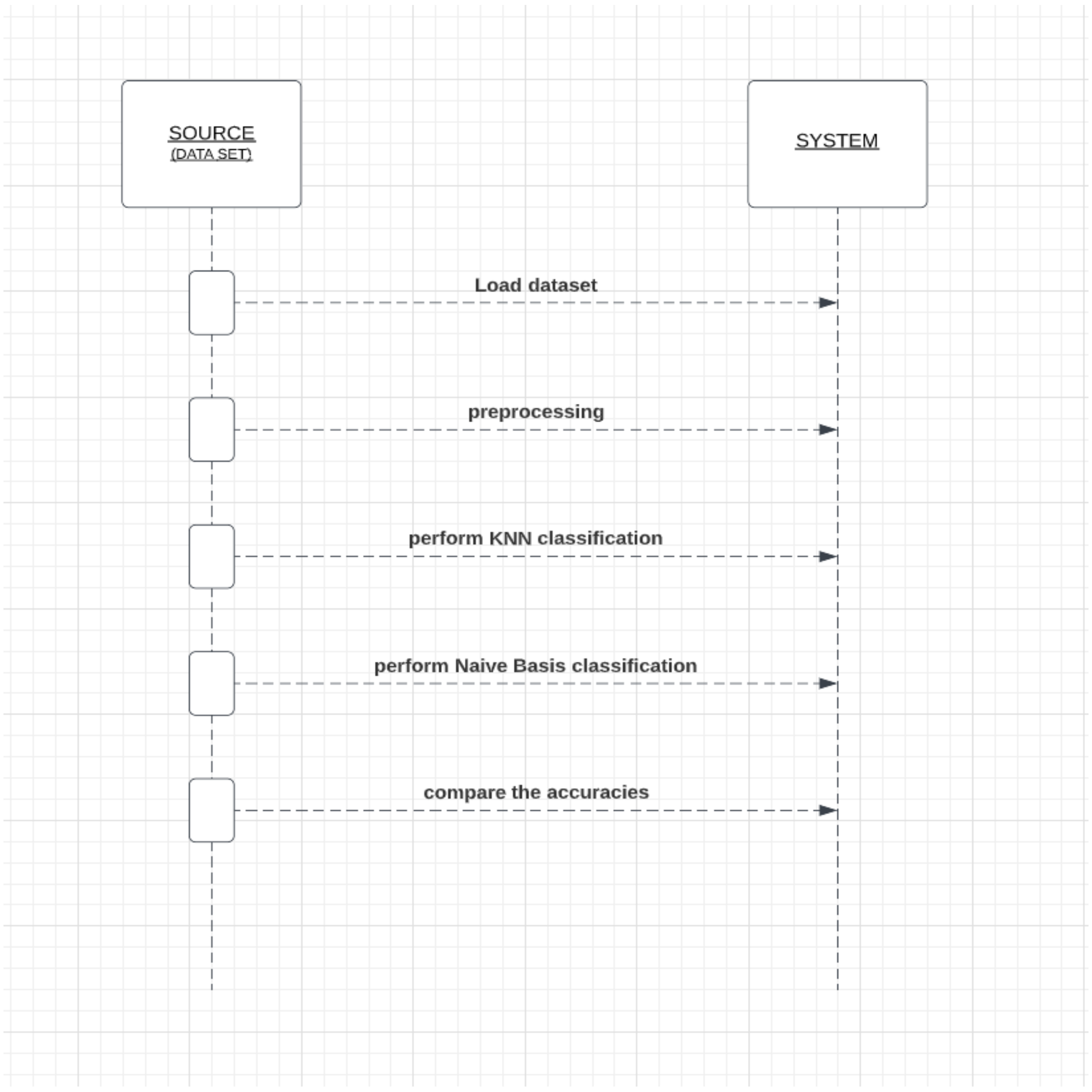


**Fig 5.4: Sequence Diagram**

The **figure 5.5** shows the activity diagram for admin module of the system. The below figure basically represents the flow from one activity to another activity in admin module. In the below figure each activity can be described as an operation of admin module of the system.

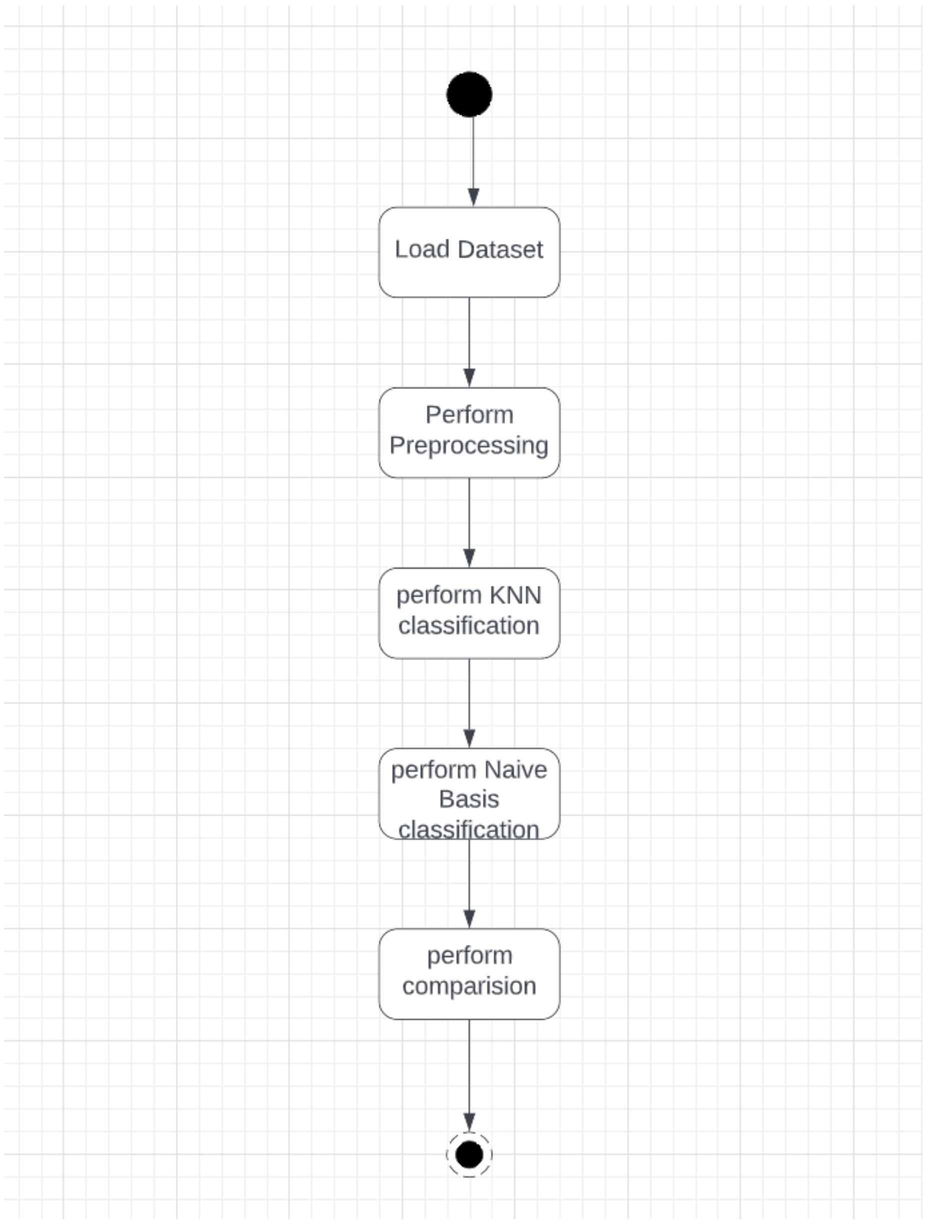

Load Dataset
Perform Preprocessing
perform KNN classification
perform Naive Basis classification
perform comparision

**Fig 5.5: Activity Diagram**

# 6. SYSTEM IMPLEMENTATION

## 6.1 Technologies

### 6.1.1 Python:

Python is a high-level programming language that is designed to be simple to read and use. It's free to use,even for commercial purposes, because it's open-source. Python is available for Mac, Windows, and Unix systems,as well as Java and .NET virtual machines.

Python, like Ruby or Perl, is a scripting language that is frequently used to create Web applications and dynamic Web content. Python is also supported by a variety of 2D and 3D imaging programs, allowing users to write custom plug-ins and extensions. GIMP, Inkscape, Blender, and Autodesk Maya are examples of applicationsthat support a Python API. Python scripts (.PY files) can be parsed and executed right away. They can also be saved as compiled programs (.PYC files), which are commonly used as programming modules that other Python programs can reference.

**Python Features:**

Python has a variety of useful features that distinguish it from other programming languages. It supports object-oriented programming, procedural programming, and memory allocation that is dynamic. A few essential features are listed below:

**Easy to Learn and Use:**

Python is a simple programming language to learn when compared to other programming languages. Its syntax issimple and like that of the English language. The semicolon and curly brackets are not used; instead, the indentation defines the code block. For beginners, it is the recommended programming language.

**Expressive Language:**

Python can perform complex tasks with just a few lines of code. For example, to run the hello world program, simply type print ("Hello World"). It will only require one line of code to run, whereas Java or C will require multiple lines.

**Interpreted Language:**

Python is an interpreted language, which means that each line of a Python program is executed separately. The benefit of being an interpreted language is that debugging is simple and portable.

**Cross-platform Language:**

Python can run on a variety of platforms, including Windows, Linux, UNIX, and Macintosh.

As a result, we can say that Python is a portable programming language. It allows programmers to create software for multiple competing platforms by writing only one program.

**Free and Open Source:**

Python is a free programming language that anyone can use. On its official website, www.python.org, it is freelyavailable. It has a large community all over the world working hard to create new Python modules and functions. The Python community welcomes contributions from anyone. "Anyone can download its source code without paying a penny," says open-source.

**Object Oriented Language:**

Python supports object-oriented programming, which introduces the concepts of classes and objects. It allows foran inheritance, polymorphism, and encapsulation, among other things. The object-oriented method aids programmers in writing reusable code and developing applications with less code.

**Extensible:**

It means that other languages, such as C/C++, can be used to compile the code, allowing us to use it in our Pythoncode. It converts the program to byte code, which can be run on any platform.

**Large Standard Library:**

It offers a diverse set of libraries for a variety of fields, including machine learning, web development, and scripting. Tensor flow, Pandas, NumPy, Keras, and Pytorch are just a few examples of machine learning libraries.Python web development frameworks include Django, Flask, and Pyramids.

**GUI Programming Support:**

For the development of a desktop application, a graphical user interface is used. The libraries used to develop theweb application are PyQT5, Tkinter, and Kivy.

**Integrated:**

It's simple to integrate with languages like C, C++, and JAVA, among others. Python, like C, C++, and Java,executes code line by line. It makes it easy to debug the code.

**Embeddable:**

Other programming languages' code can be used in the Python source code. Python source code can also be usedin other programming languages. It can embed other languages into our code.

**Dynamic Memory Allocation:**

Other programming languages' code can be used in the Python source code. Python source code can also be usedin other programming languages. It can embed other languages into our code.

### 6.1.2 Google Colab:

Google is quite aggressive in AI research. Over many years, Google developed AI framework called **TensorFlow** and a development tool called **Colaboratory**. Today TensorFlow is open-sourced and since 2017, Google made Colaboratory free for public use. Colaboratory is now known as Google Colab or simply **Colab**.

Another attractive feature that Google offers to the developers is the use of GPU. Colab supports GPU and it is totally free. The reasons for making it free for public could be to make its software a standard in the academicsfor teaching machine learning and data science. It may also have a long term perspective of building a customer base for Google Cloud APIs which are sold per-use basis.

Irrespective of the reasons, the introduction of Colab has eased the learning and development of machine learningapplications. If you have used **Jupyter** notebook previously, you would quickly learn to use Google Colab. To be precise, Colab is a free Jupyter notebook environment that runs entirely in the cloud. Most importantly, it does notrequire a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook.

**Advantages of Google Colab:**

There are the following advantages of Google Colab

1. **Sharing:**

You can share your Google Colab notebooks very easily. Thanks to Google Colab everyone with a Google account can just copy the notebook on his own Google Drive account. No need to install any modulesto run any code, modules come preinstalled within Google Colab.

2. **Versioning:**

You can save your notebook to Github with just one simple click on a button. No need to write “git addgit commit git push git pull” codes in your command client (this is if you did use versioning already)!

**3. Code Snippets:**

Google Colab has a great collection of snippets you can just plug in on your code. For example, if youwant to write data to a Google Sheet automatically, there's a snippet for it in the Google Library

**4. Forms for non-technical users:**

Not only programmers have to analyze data and Python can be useful for almost everyone in an officejob. The problem is non-technical people are scared to death of making even the tiniest change to the code.

But Google Colab has the solution for that. Just insert the comment #@param {type:"string"} and youturn any variable field in a easy-to-use form input field. Your non-technical user needs to change form fields and Google Colab will automatically update the code.

**5. Performance and Price:**

Use the computing power of the Google servers instead of your own machine. Running python scripts often requiresa lot of computing power and can take time. By running scripts in the cloud, you don't need to worry. Your local machine performance won't drop while executing your Python scripts. Best of all its freeware and cost-free to use

**6.1.3 Model Information:**

Machine Learning Classification:

Classification is a process of categorizing a given set of data into classes; it can be performed on both structured and unstructured data. The process starts with predicting the class of given data points. The classes areoften referred to as target, label, or categories.

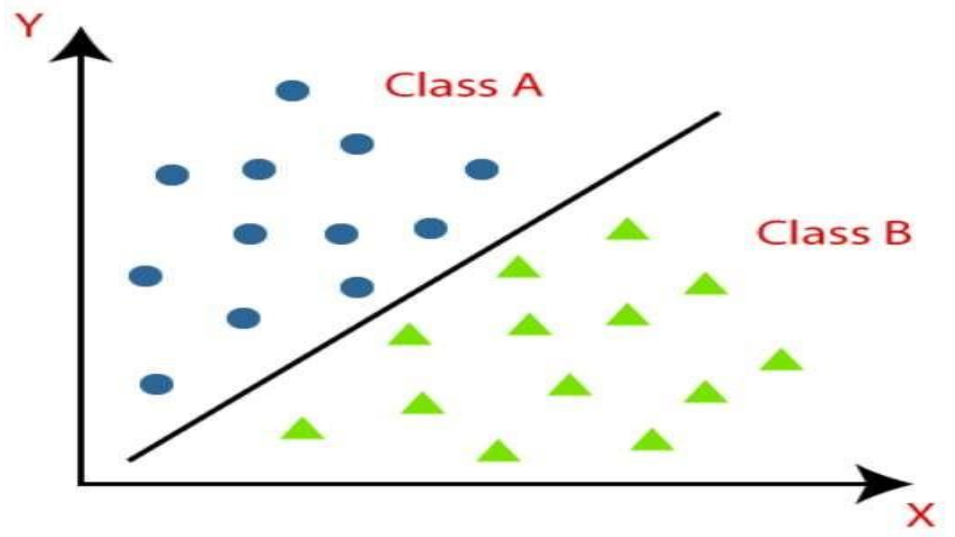


**Fig 6.1 Machine Learning Classifier**

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations based on training data. In Classification, a program learns from the given dataset or observationsand then classifies new observation into several classes or groups. Such as**, Yes or No, 0(Goodware) or 1(Malware),** etc. Classes can be called as targets/labels or categories.

**KNN (K Nearest Neighbour)**

**The KNN algorithm:**

KNN is a supervised machine learning algorithm used for solving regression and classification problems. It is a simple and non-parametric algorithm. The working of KNN is as follows:

- Load the data.
- Initialize K to the chosen number of neighbors.
- For each data in the dataset.
- Calculate the distance between the query data and the current data from the dataset.
- Add the distance and the index of the data to an ordered collection.
- Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.
- Pick the first K entries from the sorted collection.
- Get the labels of the selected K entries.
- If regression, return the mean of the K labels.
- If classification, return the mode of the K labels.

**Naive Bayes Algorithm:**

Naive Bayes is a classification algorithm for binary (two-class) and multiclass classification problems. Novel Naive Bayes or idiot Bayes is used to calculate the probabilities for each class which is simplified to make their calculations tractable. The probability of the classes is calculated by the equation:

$$\mathbf{P(class|data) = (P(data|class) * P(class)) / P(data)} \qquad \mathbf{(1)}$$

Where P(class|data) is the probability of class to the provided data. The working of KNN is:

- Separate into Class.
- Summarize Dataset.
- Summarize Data by Class.
- Gaussian Probability Density Function.
- Class Probabilities.

**K-Nearest Neighbour Classifier:**

K-Nearest Neighbour or K-NN is a Supervised Non-linear classification algorithm. K-NN is a non-parametric algorithm i.e. it does not make any assumption about underlying data or its distribution. It is one of the simplest and widely used algorithm which depends on it's k value (Neighbour's) and finds it's applications in many industries like finance industry, healthcare industry etc.

**Theory:**

In the KNN algorithm, K specifies the number of neighbours and its algorithm is as follows:

- Choose the number K of neighbour.
- Take the K Nearest Neighbour of unknown data point according to distance.
- Among the K-neighbours, Count the number of data points in each category.
- Assign the new data point to a category, where you counted the most neighbours.

For the Nearest Neighbour classifier, the distance between two points is expressed in the form of **Euclidean Distance.**

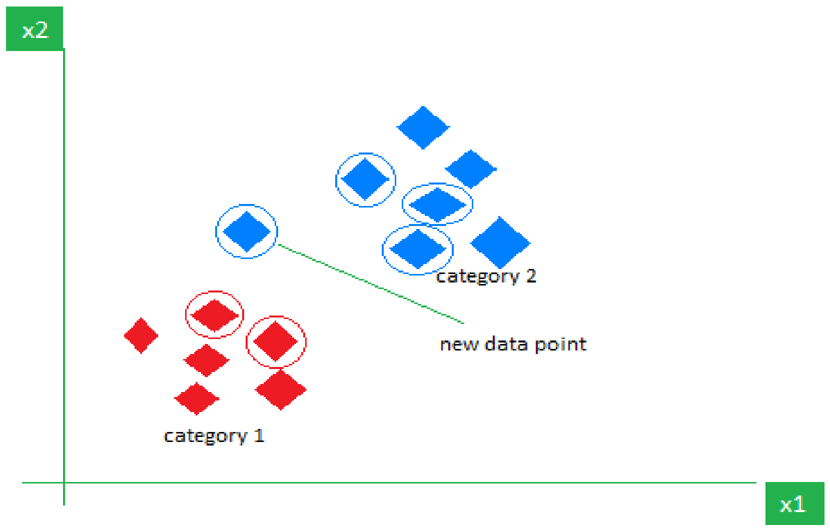


**Fig 6.2 KNN Sampling Technique**

**Naive Bayes Classifier:**

Naive Bayes is a Supervised Non-linear classification algorithm. Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Baye's theorem with strong (Naive) independence assumptions between the features or variables. The Naive Bayes algorithm is called "Naive" because it makes the assumption that the occurrence of a certain feature is independent of the occurrence of other features.

**Theory:**

Naive Bayes algorithm is based on Bayes theorem. Bayes theorem gives the conditional probability of an event A given another event B has occurred.

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

where,

P(A|B) = Conditional probability of A given B.

P(B|A) = Conditional probability of B given A.

P(A) = Probability of event A.

P(B) = Probability of event B.

For many predictors, we can formulate the posterior probability as follows:

$$P(A|B) = P(B1|A) * P(B2|A) * P(B3|A) * P(B4|A) \dots$$

### 6.1.4 Libraries:

**NumPy:**

NumPy is a Python package. It stands for 'Numerical Python'. It is a library consisting of multidimensionalarray objects and a collection of routines for processing of array.
Numeric, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open-sourceproject.

**Operations using NumPy**

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 numbergeneration.

**NumPy – A Replacement for MATLAB**

NumPy is often used along with packages like SciPy (Scientific Python) and Mat−plotlib (plotting library).This combination is widely used as a replacement for MATLAB, a popular platform for technical computing.However, Python alternative to MATLAB is now seen as a more modern and complete programming language. It is open source, which is an added advantage of NumPy.

**Pandas**

Pandas is an open-source library that is made mainly for working with relational or labelled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

Pandas were initially developed by Wes McKinney in 2008 while he was working at AQR Capital Management. He convinced the AQR to allow him to open source the Pandas. Another AQR employee, Chang She, joined as the second major contributor to the library in 2012. Over time many versions of pandas have been released. The latest version of the pandas is 1.4.1.

**Advantages:**

- Fast and efficient for manipulating and analysing data.
- Data from different file objects can be loaded.
- Easy handling of missing data (represented as NaN) in floating point as well as non-floating-point data
- Size mutability: columns can be inserted and deleted from Data Frame and higher dimensional objects
- Data set merging and joining.
- Flexible reshaping and pivoting of data sets
- Provides time-series functionality.
- Powerful group by functionality for performing split-apply-combine operations on data sets.

**Scikit learn (Sklearn):**

In Python, Scikit-learn (Sklearn) is the most usable and robust machine learning package. It uses a Pythonconsistency interface to give a set of fast tools for machine learning and statistical modelling, such as classification,regression, clustering, and dimensionality reduction. NumPy, SciPy, and Matplotlib are the foundations of this package, which is mostly written in Python. SciKit Learn includes everything from dataset manipulation to processing metrics One of the best things about SciKit Learn are the built-in algorithms for Machine Learning which you can just try out with minimal adjustments Functions such as classification, regression, clustering, mode,model selection and others are generally built-in.

- **Train_Test_Split:**

  train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data. With this function, you don't need to divide the dataset manually. By default, Sklearntrain_test_split will make random partitions for the two subsets.

**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 thebroader SciPy stack.
- It was introduced by John Hunter in the year 2002.
- One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easilydigestible visuals.

- Matplotlib consists of several plots like line, bar, scatter, histogram etc.

**Installation:**

Windows, Linux and macOS distributions have matplotlib and most of its dependencies as wheel packages.

Run the following command to install matplotlib package:

python -mpip install -U matplotlib

Basic plots in Matplotlib:

Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to makecorrelations.

They are typically instruments for reasoning about quantitative information.

**Evaluation Metrics:**

The performance of the proposed architecture is evaluated based on several statistical measures in additionto our new metric, defined as the corona score.

**Accuracy:**

Accuracy is a metric that quantifies the competency of the method in defining the correct predicted cases.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN},$$

- True Positive: True Positive is equal to the number of correct predicted positive cases.
- False Positive: False Positive is equal to the number of incorrect predicted positive cases.
- True Negative: True Negative is equal to the number of correct predicted negative cases.
- False Negative: False Negative is equal to the number of incorrect predicted negative cases

**Recall:**

Recall is the sensitivity of the method

$$Recall = \frac{TP}{TP+FN}.$$

**Precision:**

Precision is the ratio of the unnecessary positive case to the total number of positives.

$$Precision = \frac{TP}{TP+FP}.$$

**Specificity:**

Specificity is the ratio of correct predicted negatives over negative observations.

$$Specificity = \frac{TN}{TN+FP}.$$

F1-Score

F1-Score is the measure of the quality of detection.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall}.$$

**6.2 Dataset Used:**

This system uses SWELL dataset is publicly available on Kaggle website. It is the most classic dataset in the field of intrusion detection, each sample in the SWELL includes 16 features. The SWELL was collected by researchers at the Institute for Computing and Information Sciences at Radboud University. It is a result of experiments conducted on 25 subjects doing typical office work (for example writing reports, making presentations, reading email, and searching for information) (qiriro n.d.). The subject went through typical working stressors such as receiving unexpected email interruptions and pressure to complete their work on time. The dataset was set up, then the training and testing datasets were checked for missing values and thenconcatenated to form a combined dataset

| Attribute | **Description** |
|---|---|
| 1-3 | Patient details like name, age, education etc, |
| 4-7 | Patient habits and drugs usage. |
| 8-12 | Patient health conditions and diseases. |
| 14-17 | Patient Blood levels, heart rate, glucose,BMI |

**6.2.1 Data Presentation:**

**Data Pre-Processing:**

Data pre-processing is the one of the important concepts to perform on each data before training the model.

**Checking Missing Values:**

The concept of missing values is important to understand to successfully manage data. If the missingvalues are not handled properly, then it may lead to drawing an inaccurate inference about the data. Due to improper handling, the result obtained will differ.

**Finding the Outliers:**

It starts with the collection of data and that's when outliers first introduced to the population. The outliers will not be known at the collection phase, they can be the result of a mistake during data collection, orit can be just an indication of variance in the data.

**Normalize the Data:**

Normalization is a technique often applied as part of data preparation for machine learning. The goalof normalization is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.

**Splitting the Data:**

Splitting the data into train and test is based on all random images. Train data is sent to the model andthe test data is used for the model performance. Always train data must be high because it will be trained.

**Model Applying:**

Classification models are a subset of supervised machine learning. A classification model reads some input and generates an output that classifies the input into some category. For example, a model might read an email and classify it as either spam or not - binary classification.

**Model Performance:**

To calculate the model performance, the type 1 and type 2 errors are checked with TP, FP, FN, TN tocreate confusion matrix and calculating the accuracy, precision, recall and sensitivity.

**Predictions:**

When the data is passed into the saved model, model must extract the features of the data and classifythe predictions.

### 6.2.2 Linear SVM Technique for Sampling:

Support Vector Machine (SVM) is a relatively simple Supervised Machine Learning Algorithm used for classification and/or regression. It is more preferred for classification but is sometimes very useful for regression as well. Basically, SVM finds a hyper-plane that creates a boundary between the types of data. In 2-dimensional space, this hyper-plane is nothing but a line. In SVM, we plot each data item in the dataset in an N-dimensional space, where N is the number of features/attributes in the data. Next, find the optimal hyperplane to separate the data. So by this, you must have understood that inherently, SVM can only perform binary classification (i.e., choose between two classes). However, there are various techniques to use for multi-class problems. Support Vector Machine for Multi-Class Problems To perform SVM on multi-class problems, we can create a binary classifier for each class of the data. The two results of each classifier will be :

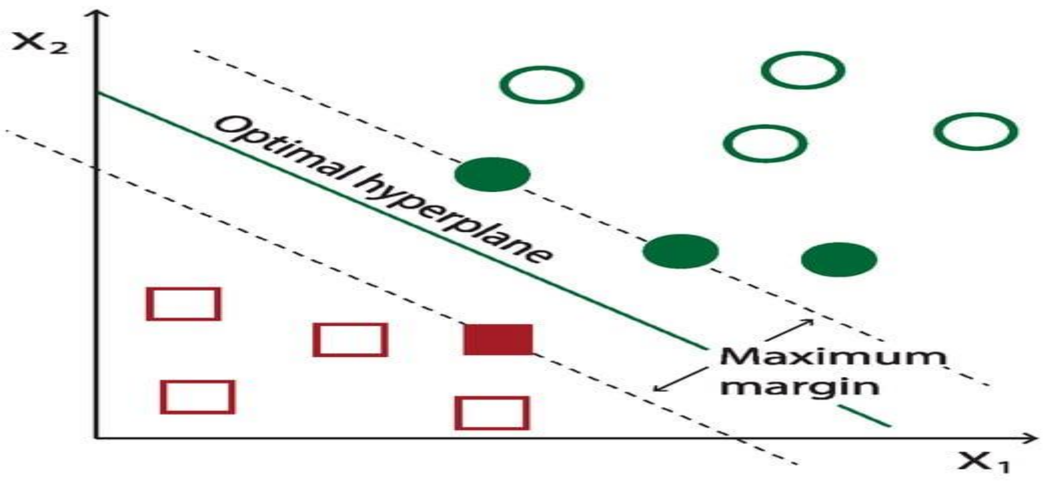
- The data point belongs to that class OR
- The data point does not belong to that class.

**Linear SVM:**

Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

**Hyperplane:**

There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.



### 6.2.3 K Nearest Neighbour Algorithm:

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

**KNN Algorithm Pseudocode:**

1. Calculate “$d(x, x_i)$” i =1, 2, ….., **n**; where **d** denotes the Euclidean distance between the points.
2. Arrange the calculated **n** Euclidean distances in non-decreasing order.
3. Let **k** be a +ve integer, take the first **k** distances from this sorted list.
4. Find those **k**-points corresponding to these **k**-distances.
5. Let **k**$_i$ denotes the number of points belonging to the i$^{th}$ class among **k** points i.e. $k \geq 0$
6. If $k_i > k_j \; \forall \; i \neq j$ then put x in class i.

### 6.2.4 Naïve Bayes Algorithm:

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.

- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

**Bayes' Theorem:**

- Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

**Where,**

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Implementation of this model is done using Google Colab, using sklearn library (Which is an opensource neural network library written in Python). In Google Colab, pre-processed Dataset is split into 70% for training Data 30% for testing purposes. The accuracy can be calculated using true positive, true negative, false positive, and false negative values.

**Flow of Implementation:**

Step 1: Importing necessary libraries and installing the modules like sklearn, NumPy, etc.

Step 2: Import the dataset and preprocess the data to remove inconsistency and the redundancy present in the data set.

Step 3: Separate the data set using SVM classifier into two sets Trained set as 70% and the 30% for the Test set.

Step 4: Plot a Linear SVM vectors representation of with mental stress and without mental stress of both trained dataset and test dataset.

Step 5: Use KNN Classifier on the data sets and find the accuracy by the iterations. By means for different iteration the accuracy changes get the mean value of the accuracy as result.

Step 6: Use Naïve Bayes classifier to get the patients with mental stress and without mental stress and here also find the accuracies based on changing the probability acquire the result by the mean of all acquired results.

Step 7: In final step, Compare the accuracies of KNN and Naïve Bayes models.

**6.3 Sample Code:**

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

# Input data files are available in the read-only "../input/" directory

# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

```
from google.colab import drive
drive.mount('/content/drive/',force_remount=True)
df = pd.read_csv('/content/drive/MyDrive/framingham (1).csv')
from sklearn.svm import SVC
from sklearn import datasets
from sklearn.preprocessing import StandardScaler
import numpy as np
#import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
df.describe()
df.shape
(4240, 16)
df['heartRate']
df['sysBP']
df['diaBP']

# check for missing values as percentage
df.isnull().sum()*100/len(df)

# cross check if all missing values are gone
```

```
df.isnull().sum().sum()

df.columns
Index(['male', 'age', 'education', 'currentSmoker', 'cigsPerDay', 'BPMeds',
       'prevalentStroke', 'prevalentHyp', 'diabetes', 'totChol', 'sysBP',
       'diaBP', 'BMI', 'heartRate', 'glucose', 'TenYearCHD'],
      dtype='object')

# education can not have relation with CHD. So dropping it

df.drop(['education'], axis=1, inplace=True)

df['gender']=df['male'].map({0:'female', 1:'male'})
df['currentSmoker'] = df['currentSmoker'].astype('object')
df['prevalentHyp']=df['prevalentHyp'].astype('object')
df['prevalentStroke']=df['prevalentStroke'].astype('object')
df['diabetes']=df['diabetes'].astype('object')
df['BPMeds']=df['BPMeds'].astype('object')

df.drop(['male'], axis=1, inplace=True)
df.info()
```

**To draw a Linear SVM:**

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs
from sklearn.svm import LinearSVC

X, y = make_blobs(n_samples=490, centers=2, random_state=0)

plt.figure(figsize=(10, 5))
for i, C in enumerate([1, 400]):
    # "hinge" is the standard SVM loss
    clf = LinearSVC(C=C, loss="hinge", random_state=42).fit(X, y)
    # obtain the support vectors through the decision function
    decision_function = clf.decision_function(X)
    # we can also calculate the decision function manually
    # decision_function = np.dot(X, clf.coef_[0]) + clf.intercept_[0]
    # The support vectors are the samples that lie within the margin
    # boundaries, whose size is conventionally constrained to 1
    support_vector_indices = np.where(
        np.abs(decision_function) <= 1 + 1e-15)[0]
    support_vectors = X[support_vector_indices]

    plt.subplot(1, 2, i + 1)
    plt.scatter(X[:, 0], X[:, 1], c=y, s=30, cmap=plt.cm.Paired)
    ax = plt.gca()
    xlim = ax.get_xlim()
    ylim = ax.get_ylim()
    xx, yy = np.meshgrid(np.linspace(xlim[0], xlim[1], 50),
                         np.linspace(ylim[0], ylim[1], 50))
    Z = clf.decision_function(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    plt.contour(xx, yy, Z, colors='k', levels=[-1, 0, 1], alpha=0.5,
                linestyles=['--', '-', '--'])
    plt.scatter(support_vectors[:, 0], support_vectors[:, 1], s=100,
                linewidth=1, facecolors='none', edgecolors='k')
    plt.xlabel('vectors representation of with mental stress and without metal
stress')
    plt.ylabel('2D representation space')
    plt.title("C=" + str(C))
#plt.tight_layout()
plt.xlabel('vectors representation of with mental stress and without mental
stress')
plt.ylabel('2D representation space')
plt.show()
```

**KNN classifier:**

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7,
test_size=0.3, random_state=100)
# Standarize features
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
# Create support vector classifier object
svc = SVC(kernel='linear', random_state=0)

# Train classifier
model = svc.fit(X_train, y_train)
# View support vectors
model.support_vectors_

# View number of support vectors for each class
model.n_support_

x_train=x_train.reshape(-1,1)
y_train=y_train.reshape(-1,1)
x_test=x_test.reshape(-1,1)


from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=15)
knn.fit(X_train,y_train)
print('knn',knn.score(X_train,y_train))
```

**Naïve Bayes Classifier:**

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7,
test_size=0.3, random_state=100)
# Standarize features
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
# Create support vector classifier object
svc = SVC(kernel='linear', random_state=0)

# Train classifier
model = svc.fit(X_train, y_train)
# View support vectors
model.support_vectors_

# View number of support vectors for each class
model.n_support_

x_train=x_train.reshape(-1,1)
y_train=y_train.reshape(-1,1)
x_test=x_test.reshape(-1,1)


from sklearn.naive_bayes import GaussianNB
gnb=GaussianNB()
gnb.fit(X_train,y_train)
print('gnb',gnb.score(X_train,y_train))
```

## 6.4 Results:

The KNN algorithm achieved an accuracy of 91.3%, and 87% whereas Naive Bayes achieved an accuracy of 88%, and 83% ($P<0.05$) in SWELL and Biometrics for stress monitoring datasets respectively.

Table 6.1 represents the sample features and classes of the input dataset for analysis. The SWELL dataset contains 12 features with 2 different classes and the Biometrics for stress monitoring dataset contains 16 attributes with 2 classes.

Table 6.2 shows the comparison of KNN and Naive Bayes algorithms on SWELL and Biometrics for stress monitoring datasets. KNN achieved an accuracy of 91.3% and Naive Bayes achieved an accuracy of 88% in the SWELL dataset. Similarly in the Biometrics for stress monitoring dataset the KNN achieved an accuracy of 87% and the Naive Bayes achieved an accuracy of 83%.

From Fig. 6.1, it is observed that the accuracy increases as the number of iterations increases. The accuracy value becomes constant after the 300th iteration. We took 10 different iteration samples for checking the accuracy values for the algorithm.

Table 6.3 shows the analysis of KNN and Naive Bayes algorithms in terms of mean accuracy, standard deviation, standard error mean values. The standard deviation of KNN is 0.01273 which shows higher performance than the Naive Bayes which had standard deviation as 0.01384.

Figure 6.2 shows the comparison of KNN and Naive Bayes algorithms as the number of iterations varied. As the iterations vary the accuracy of the algorithm varies, after the 300th iteration the accuracy remains constant.

Figure 6.3 shows the comparison of accuracy values of KNN and Naive Bayes algorithms as a bar chart by using the SPSS tool. Where the mean accuracy of KNN is higher than the Naive Bayes. The standard deviation of KNN is lesser than the Naive Bayes so, KNN performs better than Naive Bayes.

**Tables and Figures:**

Table 6.1: Input data for analysis. Both SWELL and Mental Stress are suitable for binary classification.

| **Dataset** | **No. of Patients** | **Features** | **Classes** |
|---|---|---|---|
| SWELL | 4240 | 16 | 2 |
| Biometrics for stress monitoring | 32794 | 12 | 2 |

Table 6.2: Comparison of KNN and Naive Bayes with accuracy

| **Datasets** | **KNN** | **Naive Bayes** |
|---|---|---|
| SWELL dataset | Accuracy – 91.3% | Accuracy – 88% |
| Biometrics for stress monitoring dataset | Accuracy – 87% | Accuracy – 83% |

Table 6.3: Represents the performance of KNN and Naive Bayes algorithms in terms of mean accuracy, standard deviation, and standard error mean. The mean accuracy of KNN is higher than Naive Bayes.

| | Groups | N | Mean | Std. Deviation | Std. Error Mean |
|---|---|---|---|---|---|
| Accuracy | KNN | 10 | 0.8625 | 0.01273 | 0.00592 |
| | Naïve Bayes | 10 | 0.8339 | 0.01384 | 0.00438 |

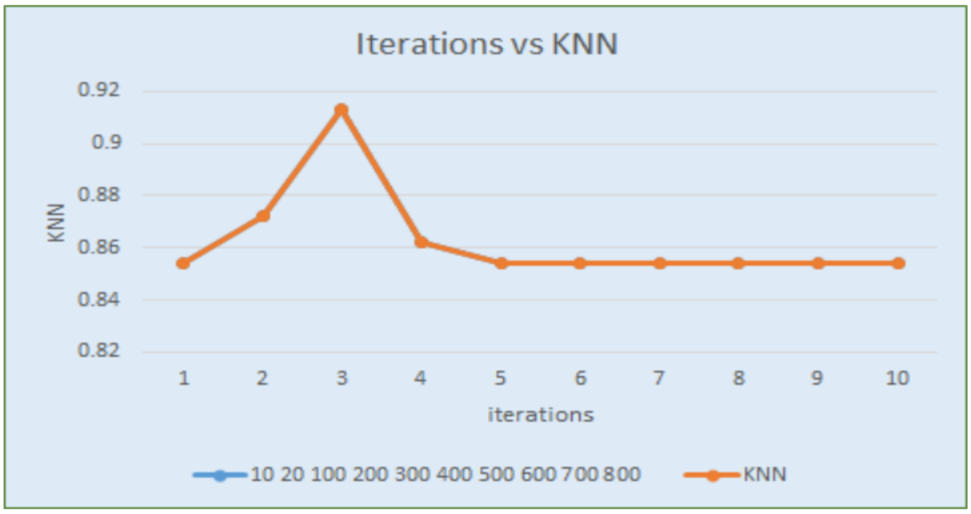


**Fig.6.3.** Accuracy of KNN for different iterations. Fluctuations occur before the 300th iteration after that the accuracy remains constant.

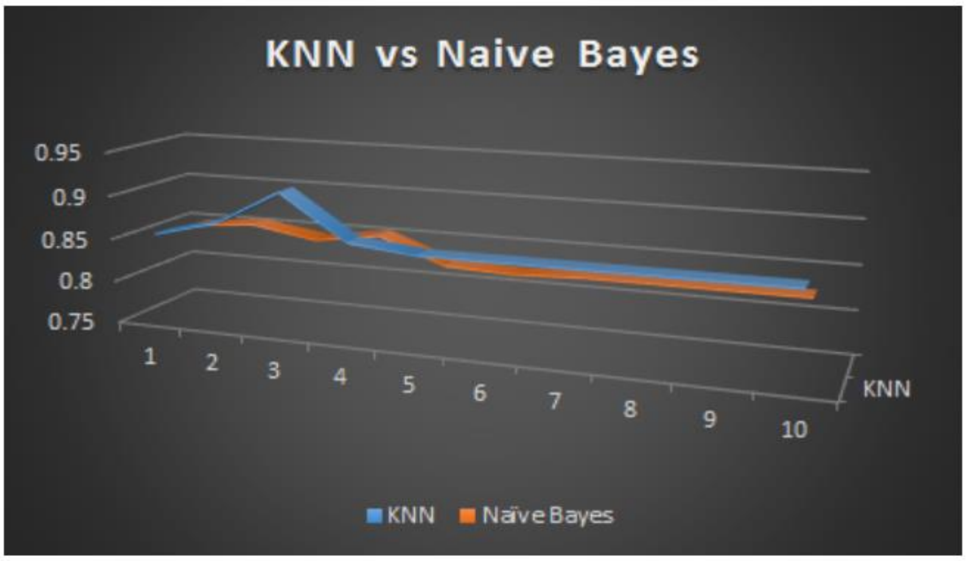


Fig.6.4. Comparison of KNN and Naive Bayes at different iteration values

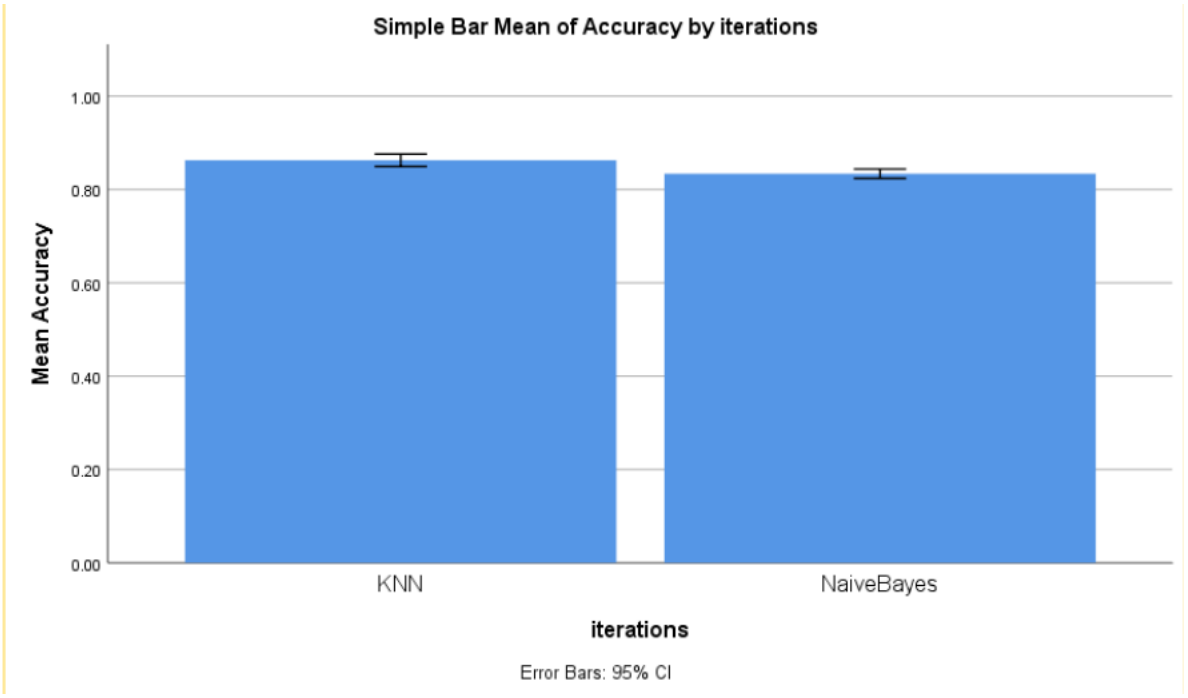


**Fig.6.5.** In terms of mean accuracy, the KNN and Naive Bayes are compared. The X-axis represents the groups (algorithms) and Y-axis represents the mean accuracy. It shows that the KNN mean accuracy is higher than the Naive Bayes and the Mean accuracy is ± 1 SD.

# 7. SYSTEM TESTING

## 7.1 Introduction:

Testing forms an integral part of any software development project. Testing helps in ensuring that the final product is by and large, free of defects and it meets the desired requirements. Proper testing in the development phase helps in identifying the critical errors in the design and implementation of various functionalities thereby ensuring product reliability. Even though it is a bit time-consuming and a costly process at first, it helps in the long run of software development.

Although machine learning systems are not traditional software systems, not testing them properly for their intended purposes can lead to a huge impact in the real world. This is because machine learning systems reflect the biases of the real world. Not accounting or testing for them will inevitably have lasting and sometimes irreversible impacts.

**Testing Traditional Software Systems v/s Machine Learning Systems**

In traditional software systems, code is written for having a desired behaviour.as the outcome. Testing them involves testing the logic behind the actual behaviour. and how it compares with the expected behaviour.

In machine learning systems, however, data and desired behavior are the inputs and the models learn the logic as the outcome of the training and optimization processes. In this case, testing involves validating the consistency of the model's logic and our desired behavior.

Due to the process of models learning the logic, there are some notable obstacles in the way of testing Machine Learning systems. They are:

- Indeterminate outcomes-On retraining, it's highly possible that the model parameters vary significantly
- Generalization-It's a huge task for Machine Learning models to predict sensible outcomes for data not encountered in their training.
- Coverage-There is no set method of determining test coverage for a Machine Learning model. Interpretability-Most ML models are black boxes and don't have a comprehensible logic for a certain decision made during prediction.

These issues lead to a lower understanding of the scenarios in which models fail and the reason for that behavior; not to mention, making it more difficult for developers to improve their behaviours.

**Difference between Model Testing and Model Evaluation**

From the discussion above, it may feel as if model testing is the same as model evaluation but that's not true. Model evaluations focus on the performance metrics of the models like accuracy, precision, the area under the curve, f1 score, log loss, etc. These metrics are calculated on the validation dataset and remain confined to that. Though the evaluation metrics are necessary for assessing a model, they are not sufficient because they don't shed light on the specific behaviours of the model.

It is fully possible that a model's evaluation metrics have improved but its behavior on a core functionality has regressed. Or retraining a model on new data might introduce a bias for marginalized sections of society all the while showing no difference in the metrics values. This is extra harmful in the case of ML systems since such problems might not come to light easily but can have devastating impacts.

In summary, model evaluation helps in covering the performance on validation datasets while model testing helps in explicitly validating the nuanced behaviours of our models. During the development of ML models, it is better to have both model testing and evaluation to be executed in parallel.

**Writing Test Cases:**

We usually write two different classes of tests for Machine Learning systems:

- Pre-train tests
- Post-train tests

**Pre-train tests:**

The intention is to write such tests which can be run without trained parameters so that we can catch implementation errors early on. This helps in avoiding the extra time and effort spent in a wasted training job. We can test the following in the pre-train test:

- The model predicted output shape is proper or not.
- Test dataset leakage i.e., checking whether the data in training and testing datasets have no duplication.
- Temporal data leakage which involves checking whether the dependencies between training and test data do not lead to unrealistic situations in the time domain like training on a future data point and testing on a past data point.

- Check for the output ranges. In the cases where we are predicting outputs in a certain range (for example when predicting probabilities), we need to ensure the final prediction is not outside the expected range of values.
- Ensuring a gradient step training on a batch of data leads to a decrease in the loss.
- Data profiling assertions

**Post-train tests:**

Post-train tests are aimed at testing the model's behavior. We want to test the learned logic and it could be tested on the following points and more:

- Invariance tests which involve testing the model by tweaking only one feature in a data point and checking for consistency in model predictions. For example, if we are working with a loan prediction dataset then change in sex should not affect an individual's eligibility for the loan given all other features are the same or in the case of titanic survivor probability prediction data, change in the passenger's name should not affect their chances of survival.
- Directional expectations wherein we test for a direct relation between feature values and predictions. For example, in the case of a loan prediction problem, having a higher credit score should increase a person's eligibility for a loan.
- Apart from this, you can also write tests for any other failure modes identified for your model.

## 7.2 Test Cases:

| Test No | Test Cases | Expected Output | Actual Output | Pass/Fail |
|---|---|---|---|---|
| **1** | Importing the Libraries | Imported | Imported without errors | Pass |
| **2** | Collecting the data and by using the Authentications and mounting | Collected | Got Access to the data using the authentications | Pass |
| **3** | Pre-process the data | Data is cleaned | Data is cleaned without errors | Pass |
| **4** | Normalizing the data | Normalize the data | Normalize the data | Pass |
| **5** | Splitting data into Train and Test | Train and Test 70-30% | Train and Test 70-30% | Pass |
| **6** | Running model | Training the KNN Classifier and Naïve Bayes Classifier. | Trained the KNN Classifier and Naïve Bayes Classifier. | Pass |
| **7** | Model Evaluation | Classification Report | Classification Report Generated | Pass |
| **8** | Predictions | User sending the input from the validation for the prediction to classify | Model predicting the output. | Pass |

**Table 7.2.1 Test Cases**

# 8. CONCLUSION

From this proposed work, one can perform an initial analysis for mental stress detection to help the individuals to undergo treatment at the earliest. It is also observed that the proposed KNN algorithm performed significantly better than the Naive Bayes algorithm in mental stress detection. There is an improvement in the accuracy of the proposed algorithm. In this study, it is observed that the KNN algorithm performs significantly better than the Naive Bayes algorithm.

# 9. FUTURE ENHANCEMENTS

- In future, we can apply pre-defined models to check the better accuracy.
- We can also implement the system by using Random Forest and Decision Tress Algorithms for better accuracy prediction and compare the accuracies.

# BIBLIOGRAPHY

1. Ahuja, Ravinder, and Alisha Banga. 2019. "Mental Stress Detection in University Students Using Machine Learning Algorithms." Procedia Computer Science. https://doi.org/10.1016/j.procs.2019.05.007.
2. Attallah, Omneya. 2020. "An Effective Mental Stress State Detection and Evaluation System Using Minimum Number of Frontal Brain Electrodes." Diagnostics (Basel, Switzerland) 10 (5). https://doi.org/10.3390/diagnostics10050292.
3. Can, Yekta Said, Niaz Chalabianloo, Deniz Ekiz, and Cem Ersoy. 2019a. "Continuous Stress Detection Using Wearable Sensors in Real Life: Algorithmic Programming Contest Case Study." Sensors. https://doi.org/10.3390/s19081849.
4. Dalmeida, Kayisan Mary, and Giovanni Luca Masala. n.d. "Stress Classification of ECG-Derived HRV Features Extracted from Wearable Devices." https://doi.org/10.20944/preprints202103.0644.v1.
5. Elzeiny, Sami, and Marwa Qaraqe. 2018. "Machine Learning Approaches to Automatic Stress Detection: A Review." 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA). https://doi.org/10.1109/aiccsa.2018.8612825.Fernández, José Raúl Machado, and Lesya Anishchenko. 2018. "Mental Stress Detection Using Bioradar Respiratory Signals." Biomedical Signal Processing and Control. https://doi.org/10.1016/j.bspc.2018.03.006.
6. Hasan, Md Junayed, and Jong-Myon Kim. 2019. "A Hybrid Feature Pool-Based Emotional Stress State Detection Algorithm Using EEG Signals." Brain Sciences 9 (12). https://doi.org/10.3390/brainsci9120376.
7. Koldijk, Saskia, Mark A. Neerincx, and Wessel Kraaij. 2018. "Detecting Work Stress in Offices by Combining Unobtrusive Sensors." IEEE Transactions on Affective Computing. https://doi.org/10.1109/taffc.2016.2610975.
8. Kumar, Muthusamy Senthil, Gelli Vamsi, Ramasamy Sripriya, and Praveen Kumar Sehgal. 2006. "Expression of Matrix Metalloproteinases (MMP-8 and -9) in Chronic Periodontitis Patients with and without Diabetes Mellitus." Journal of Periodontology 77 (11): 1803–8.
9. Kumar, Prince, Shruti Garg, and Ashwani Garg. 2020. "Assessment of Anxiety, Depression and Stress Using Machine Learning Models." Procedia Computer Science. https://doi.org/10.1016/j.procs.2020.04.213.