

“Early Detection of Alzheimer’s disease with Blood Plasma Proteins Using Support Vector Machines”

A Project Report submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR.

In Partial Fulfillment of the Requirements for the Award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
BY

M.PRUDHVINATH	(18121A05D5)
M.SATHVIKA	(18121A05F3)
N.MAHESH BABU	(18121A05G0)
O.AISWARYA	(18121A05G9)

Under the Guidance of

Dr.K.Suresh ,Ph.D.
Professor
Dept. of CSE, SVEC



Department of Computer Science and Engineering
SREE VIDYANIKETHAN ENGINEERING COLLEGE
(Affiliated to JNTUA, Anantapuramu)
Sree Sainath Nagar, Tirupathi – 517 102
2018-2022



SREE VIDYANIKETHAN ENGINEERING COLLEGE

(Affiliated to Jawaharlal Nehru Technological University Anantapur)
Sree Sainath Nagar, A. Rangampet, Tirupati – 517 102, Chittoor Dist., A.P.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project Work entitled

**“Early Detection of Alzheimer’s disease with Blood Plasma
Proteins Using Support Vector Machines “**

is the bonafide work done by

M.PRUDHVINATH

(18121A05D5)

M.SATHVIKA

(18121A05F3)

N.MAHESH BABU

(18121A05G0)

O.AISWARYA

(18121A05G9)

In the Department of Computer Science and Engineering, Sree Vidyanikethan Engineering College, A. Rangampet. is affiliated to JNTUA, Anantapuramu in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering during 2018-2022.

This work has been carried out under my guidance and supervision.

The results embodied in this Project report have not been submitted in any University or Organization for the award of any degree or diploma.

Internal Guide

Dr.K.Suresh,Ph.D.

Professor
Dept. of CSE
Sree Vidyanikethan Engineering College
Tirupati

Head

Dr. B. Narendra Kumar Rao

Prof & Head
Dept. of CSE
Sree Vidyanikethan Engineering College
Tirupati

INTERNAL EXAMINER

EXTERNAL EXAMINER

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION AND MISSION

VISION

To become a Centre of Excellence in Computer Science and Engineering by imparting high quality education through teaching, training and research.

MISSION

The Department of Computer Science and Engineering is established to provide undergraduate and graduate education in the field of Computer Science and Engineering to students with diverse background in foundations of software and hardware through a broad curriculum and strongly focused on developing advanced knowledge to become future leaders.

Create knowledge of advanced concepts, innovative technologies and develop research aptitude for contributing to the needs of industry and society.

Develop professional and soft skills for improved knowledge and employability of students.

Encourage students to engage in life-long learning to create awareness of the contemporary developments in computer science and engineering to become outstanding professionals.

Develop attitude for ethical and social responsibilities in professional practice at regional, National and International levels.

Program Educational Objectives (PEO's)

1. Pursuing higher studies in Computer Science and Engineering and related disciplines
2. Employed in reputed Computer and I.T organizations and Government or have established startup companies.
3. Able to demonstrate effective communication, engage in team work, exhibit leadership skills, ethical attitude, and achieve professional advancement through continuing education.

Program Specific Outcomes (PSO's)

1. Demonstrate knowledge in Data structures and Algorithms, Operating Systems, Database Systems, Software Engineering, Programming Languages, Digital systems, Theoretical Computer Science, and Computer Networks. (PO1)
2. Analyze complex engineering problems and identify algorithms for providing solutions (PO2)
3. Provide solutions for complex engineering problems by analysis, interpretation of data, and development of algorithms to meet the desired needs of industry and society. (PO3, PO4)
4. Select and Apply appropriate techniques and tools to complex engineering problems in the domain of computer software and computer based systems (PO5)

Program Outcomes (PO's)

1. Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems (**Engineering knowledge**).
2. Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences (**Problem analysis**).
3. Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations (**Design/development of solutions**).
4. Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions (**Conduct investigations of complex problems**).
5. Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations (**Modern tool usage**).
6. Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice (**The engineer and society**).
7. Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development (**Environment and sustainability**).

8. Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice (**Ethics**).

9. Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings (**Individual and team work**).

10. Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions (**Communication**).

11. Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments (**Project management and finance**).

12. Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change (**Life-long learning**).

Course Outcomes

CO1. Knowledge on the project topic (PO1)

CO2. Analytical ability exercised in the project work.(PO2)

CO3. Design skills applied on the project topic. (PO3)

CO4. Ability to investigate and solve complex engineering problems faced during the project work. (PO4)

CO5. Ability to apply tools and techniques to complex engineering activities with an understanding of limitations in the project work. (PO5)

CO6. Ability to provide solutions as per societal needs with consideration to health, safety, legal and cultural issues considered in the project work. (PO6)

CO7. Understanding of the impact of the professional engineering solutions in environmental context and need for sustainable development experienced during the project work. (PO7)

CO8. Ability to apply ethics and norms of the engineering practice as applied in the project work.(PO8)

CO9. Ability to function effectively as an individual as experienced during the project work. (PO9)

CO10. Ability to present views cogently and precisely on the project work. (PO10)

CO11. Project management skills as applied in the project work. (PO11)

CO12. Ability to engage in life-long learning as experience during the project work. (PO12)

CO-PO Mapping

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3	PSO 4
C01	3												3			
C02		3												3		
C03			3												3	
C04				3											3	
C05					3											3
C06						3										
C07							3									
C08								3								
C09									3							
C010										3						
C011											3					
C012												3				

(Note: 3-High, 2-Medium, 1-Low)

DECLARATION

We hereby declare that this project report titled "**Early Detection of Alzheimer's disease with Blood Plasma Proteins Using Support Vector Machines**" is a genuine project work carried out by us, in **B.Tech (Computer Science and Engineering)** degree course of **Jawaharlal Nehru Technological University Anantapur** and has not been submitted to any other course or University for the award of any degree by us.

1.M. Prudhvinath

2.M. Sathvika

3.N.Mahesh Babu

4.O.Asiwarya

Signature of the student

ACKNOWLEDGEMENT

We are extremely thankful to our beloved Chairman and founder **Dr. M. Mohan Babu** who took keen interest to provide us the infrastructural facilities for carrying out the project work.

We are highly indebted to **Dr.B.M.Satish**, Principal of Sree Vidyanikethan Engineering College for his valuable support and guidance in all academic matters.

We are very much obliged to **Dr. B. Narendra Kumar Rao**, Professor & Head, Department of CSE, for providing us the guidance and encouragement in completion of this project.

We would like to express our indebtedness to the project coordinator, **Dr. K. Kannan**, Professor, Department of CSE for his valuable guidance during the course of project work.

We would like to express our deep sense of gratitude to **Dr.K.Suresh**, Professor, Department of CSE, for the constant support and invaluable guidance provided for the successful completion of the project.

We are also thankful to all the faculty members of CSE Department, who have cooperated in carrying out our project. We would like to thank our parents and friends who have extended their help and encouragement either directly or indirectly in completion of our project work.

ABSTRACT

The successful development of amyloid-based biomarkers and tests for Alzheimer's Disease (AD) represents an important milestone in AD diagnosis. However, two major limitations remain. Amyloid-based diagnostic biomarkers and tests provide limited information about the disease process and they are unable to identify individuals with the disease before significant amyloid-beta accumulation in the brain develops. The objective in this project is to develop a method to identify potential blood-based non-amyloid biomarkers for early AD detection. The use of blood is attractive because it is accessible and relatively inexpensive. Our method is mainly based on Machine Learning (ML) techniques (support vector machines in particular) because of their ability to create multivariable models by learning patterns from complex data. Using novel feature selection and evaluation modalities we identified 5 novel panels of non-amyloid proteins with the potential to serve as biomarkers of early AD. In particular, we found that the combination of A2M, ApoE, BNP, Eot3, RAGE and SGOT may be a key biomarker profile of early disease. Existing ML models performed poorly in comparison at this stage of the disease suggesting that the underlying protein panels may not be suitable for early disease detection. Our results demonstrate the feasibility of early detection of AD using non amyloid based biomarkers.

KEYWORDS: Alzheimer's disease, blood biomarker, dementia, machine learning, support vector machine

TABLE OF CONTENTS

CONTENTS	PAGE NO.
1. INTRODUCTION	
1.1 INTRODUCTION	1
1.2 STATEMENT OF THE PROBLEM	4
1.3 OBJECTIVES	5
1.4 SCOPE	5
1.5 APPLICATION	6
1.6 LIMITATION	6
2. LITERATURE SURVEY	7
3. ANALYSIS	11
4. DESIGN	14
5. IMPLEMENTATION	23
6. EXECUTION PROCEDURE AND TESTING	36
7. RESULTS & PERFORMANCE EVALUATIION	43
8. CONCLUSION AND FUTURE WORK	45
LIST OF FIGURES	46
REFERENCES	47

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

ALZHEIMER'S Disease (AD) is the leading cause of dementia and poses a significant social and economic challenge. It is responsible for more than half of all cases of dementia. Over 50 million individuals currently suffer from dementia worldwide with a projected increase to 152 million by 205. No cure for AD has been discovered, but there is intense effort to develop new clinical interventions that may slow or halt the disease. Such interventions are aimed at early stages of the disease prior to extensive cell damage, when it is thought treatment is more likely to be effective. To facilitate early diagnosis, the use of established biomarkers such as those based on amyloid-beta in cerebral spinal fluid and molecular imaging of brain amyloid deposition using positron emission tomography is recommended. However, despite progress with the development of amyloid based biomarkers and tests for early AD diagnosis, they have two major constraints. In addition, tests based on these biomarkers are unable to identify individuals at risk of AD prior to a significant amyloid-beta deposition in the brain. There is a need for biomarkers that have the potential to detect biological processes that precede brain amyloid-beta accumulation (amyloid pathology) during the disease development. Such biomarkers may advance understanding of the disease.

Studies suggest that AD is characterized by metabolic alterations that may precede amyloid pathology. Signatures of such metabolic abnormalities may therefore serve as biomarkers of earlier stages of the disease than amyloid biomarkers. Such biomarkers may be obtained from blood since blood has rich metabolic information content. The use of blood is also attractive because blood biomarker-based test is relatively non-invasive compared to CSF and may be more cost-effective than PET imaging. A number of studies have attempted to find non-amyloid biomarkers of disease by profiling a large array of non-amyloid proteins in blood and examining their association with the disease but this approach is difficult to apply in practice.



Figure 1.1.1:Memory losing

A promising approach is the use of machine learning techniques to find appropriate combinations of non-amyloid proteins to detect AD as no single non-amyloid protein has been shown to reliably detect the disease. ML makes it possible to fit multivariable data to a model by learning complex patterns from data. Several studies have applied ML to develop classifiers to differentiate between

AD subjects and healthy controls. For example, O'Bryant et al, developed a model with a panel of 30 serum proteins that classified Alzheimer's Disease Dementia (ADD) subjects and HCs with Sensitivity (SN), Specificity (SP), and area under receiver operating curve (AUC) of 88%, 82%, and 0.91, respectively. Similarly, with 14 plasma proteins, a classifier model constructed by Llano et al. classified ADD and HC subjects with 86.5% SN, 84.2% SP and AUC of 0.85. More recently, a panel of inflammatory markers in plasma was identified that classified ADD and HC with 84% SN, 70% SP, and AUC of 0.79 using a logistic regression model. In another study, a 12-marker panel classified ADD and HC with 90% SN and 66.7% specificity, and higher performance in post-mortem confirmed AD cases. Furthermore, a study that explored the use of deep learning, random forest, and algorithms for classification of ADD and HC achieved AUC of 0.85 with deep learning and random forest. Despite the promising results from these studies, most of the models were developed and evaluated using data from cognitively healthy controls and subjects at the later stages of the disease.

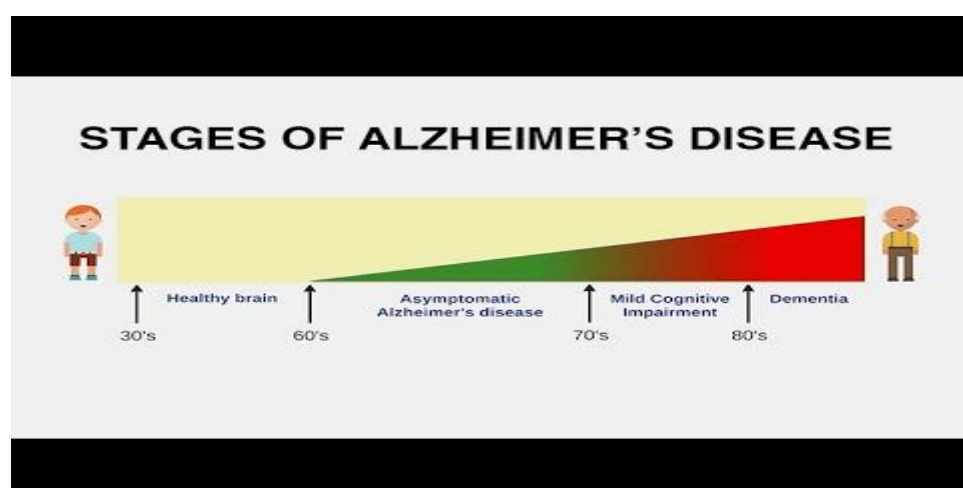


Figure 1.1.2: Ages of AD

The models were not evaluated in individuals at the early stages of the disease. Therefore, the panels underlying such models may not be suitable as biomarker signatures of early AD. In this study, the main objective is to develop a ML-based method (support vector machines (SVM) in particular – see later) to identify blood biomarkers of early AD based on non-amyloid proteins with the potential to identify the disease prior to accumulation of amyloid-beta in the brains. We also assess the potential of existing ML-based methods to achieve early disease detection.



Figure 1.1.3: Support Vector Machine

1.2 STATEMENT OF THE PROBLEM

Memory loss is often the first and main symptom in early Alzheimer's disease. It is also seen, although less often, in early vascular dementia and Dementia with Lewy Bodies (DLB). Memory loss is not common in early Front Temporal Dementia (FTD). It is difficult to detect the Alzheimer's disease dementia. So to overcome this problem by using machine learning algorithms.

1.3 OBJECTIVES

- The main objective of this project is to enhance the overall performance analysis.
- To effectively classify and predict the emotions.
- To predict or detect the Alzheimer's dementia disease based on symptoms.
- To implement the machine learning algorithm.

1.4 SCOPE

The goal of this study is to devise a method for identifying potential blood-based non-amyloid biomarkers for early identification of Alzheimer's disease. Blood is appealing since it is readily available and relatively affordable. Because of their ability to generate multivariable models by learning patterns from complex data, our method is primarily based on machine learning techniques (support vector machines in particular). We identified 5 unique panels of non-amyloid proteins with the potential to act as biomarkers for early Alzheimer's disease using novel feature selection and evaluation methodologies. The combination of A2M, ApoE, BNP, Eot3, RAGE, and SGOT, in particular, was discovered to constitute a critical biomarker profile of early illness.

1.5 APPLICATION

- Detecting Alzheimer's disease Early can help to know how to cure the disease.
- Less time taking process.
- More effective and accurate results.

1.6 LIMITATIONS

- Large datasets are unmanageable.
- SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.
- It is included in the related SVC method of Python scikit-learn library.
- SVM does not perform very well when the data set has more noise.

CHAPTER 2

LITERATUE SURVEY

[1] Predicting Prodromal Dementia Using Linguistic Patterns and Deficits 2020

Methodology:

Data was derived from the cookie theft picture corpus of Dementia Bank, from which all language samples of the identified aetiologies were used, with a random subsampling technique that handles the skewness of the classes. Several original lexical and syntactic (i.e., lexicosyntactic) features were introduced and used alongside previously established lexicosyntactics to train machine learning classifiers against these aetiologies’. Further, a statistical analysis was conducted to uncover the deficiency across these aetiologies’. Our models resulted in benchmarks for differentiating all the identified classes with accuracies ranging between 95 to 98% and corresponding F1 values falling between 94 and 98%. The statistical analysis of our lexicosyntactic biomarkers shows that linguistic deviations are associated with prodromal as well as advanced neurodegenerative pathologies, being greatly impacted as cognitive decline increases and suggesting that language biomarkers may aid the early diagnosis of these pathologies.

[2] DEMNET: A Deep Learning Model for Early Diagnosis of Alzheimer Diseases and Dementia from MR Images, 2021***Methodology:***

Alzheimer's disease is the most common cause of dementia globally. It steadily worsens from mild to severe, impairing one's ability to complete any work without assistance. It begins to outstrip due to the population ages and diagnosis timeline. For classifying cases, existing approaches incorporate medical history, neuropsychological testing, and Magnetic Resonance Imaging (MRI), but efficient procedures remain inconsistent due to lack of sensitivity and precision. The Convolutional Neural Network (CNN) is utilized to create a framework that can be used to detect specific Alzheimer's disease characteristics from MRI images. By considering four stages of dementia and conducting a particular diagnosis, the proposed model generates high-resolution disease probability maps from the local brain structure to a multilayer perceptron and provides accurate, intuitive visualizations of individual Alzheimer's disease risk. To avoid the problem of class imbalance, the samples should be evenly distributed among the classes. The obtained MRI image dataset from Kaggle has a major class imbalance problem. The DEMNET achieves an accuracy of 95.23%, Area under Curve of 97% and Cohen's Kappa value of 0.93 from the Kaggle dataset, which is superior to existing methods. We also used the Alzheimer's disease Neuroimaging Initiative (ADNI) dataset to predict AD classes in order to assess the efficacy of the proposed model.

[3] Alzheimer’s Diseases Detection by Using Deep Learning Algorithms: A Mini-Review, 2020

Methodology:

The accurate diagnosis of Alzheimer’s disease plays an important role in patient treatment, especially at the disease’s early stages, because risk awareness allows the patients to undergo preventive measures even before the occurrence of irreversible brain damage. Although many recent studies have used computers to diagnose AD, most machine detection methods are limited by congenital observations. AD can be diagnosed-but not predicted-at its early stages, as prediction is only applicable before the disease manifests itself. Deep Learning (DL) has become a common technique for the early diagnosis of AD. Here, we briefly review some of the important literature on AD and explore how DL can help researchers diagnose the disease at its early stages. From a computational perspective, this recent advancement has spawned the development of tools that incorporate several patient-specific observations into predictions and improve the clinical outcomes of patients suffering from such disorders.

**[4] Artificial Intelligence for Caregivers of Persons
with Alzheimer’s disease and Related Dementias:
Systematic Literature Review, 2020**

Methodology:

Following the Preferred Reporting Items for Systematic Reviews and Meta-Analyses guidelines for conducting systematic literature reviews, during August and September 2019, we performed 3 rounds of selection. First, we searched predetermined keywords in PubMed, Cumulative Index to Nursing and Allied Health Literature Plus with Full Text, PsycINFO, IEEE Xplore Digital Library, and the ACM Digital Library. This step generated 113 nonduplicate results. Next, we screened the titles and abstracts of the 113 topics according to inclusion and exclusion criteria, after which 52 topics were excluded and 61 remained. Finally, we screened the full text of the remaining topics to ensure that they met the inclusion or exclusion criteria; 31 topics were excluded, leaving a final sample of 30 topics for analysis.

CHAPTER 3

ANALYSIS

The goal of this chapter is to look at the exact hardware, software, and design requirements, as well as their functions.

SPECIFIC REQUIREMENTS

Jupyter Notebook that supports (Python version 3.7) as we are using Machine Learning and SVM is used for the creation of data visualization libraries. Jupyter Notebook is more flexible environment for executing these. ipynb files.

EXISTING SYSTEM

Based on Machine learning techniques (Logistic regression in particular) because of their process of modelling the probability of a discrete outcome given an input variable. The most common logistic regression models a binary outcome; something that can take two values such as true/false, yes/no, and so on. Alzheimer's disease is thought to be caused by the abnormal build-up of proteins in and around brain cells. One of the proteins involved is called amyloid, deposits of which form plaques around brain cells. The other protein is called tau, deposits of which form tangles within brain cells. Disease detection models based on the identified panels achieved sensitivity $< 70\%$, specificity $< 73\%$, and area under receiver operating curve of at least 0.60 at prodromal stage (with higher performance at later stages) of the disease. Existing ML models performed poorly in comparison at this stage of the disease, suggesting that the underlying protein panels may not be suitable for early disease detection. Our results

demonstrate the feasibility of early detection of AD using non-amyloid based biomarkers.

DISADVANTAGES:

- The results is low when compared with proposed.
- It doesn't efficient for large volume of data's .
- Theoretical limits.

PROPOSED SYSTEM

Our method is mainly based on machine learning techniques (support vector in particular) because of their ability to create multivariable models by learning patterns from complex data. Using novel feature selection and evaluation modalities, we identified 5 novel panels of amyloid proteins with the potential to serve as biomarkers of early AD. In particular, we found that the combination of A2M, ApoE, BNP, Eot3, RAGE and SGOT may be a key biomarker profile of early disease. Disease detection models based on the identified panels achieved sensitivity > 80%, specificity > 70%, and area under receiver operating curve of at least 0.80 at prodromal stage (with higher performance at later stages) of the disease. Existing ML models performed poorly in comparison at this stage of the disease, suggesting that the underlying protein panels may not be suitable for early disease detection. Our results demonstrate the feasibility of early detection of AD using non-amyloid based biomarkers.

HARDWARE REQUIRMENTS

Processor	: I3/Intel
	Processor
Hard Disk	: 500GB
RAM	: 4GB

SOFTWARE REQUIREMENTS

OS	: Windows 7/8/10
IDE	: Anaconda Prompt
Libraries	: SVM,sklearn,seaborn
Technology	: Python 3.7+

Anaconda prompt environment is used to execute the system program that need to be done for training the data set and to perform data preprocessing on the datasets.

It is easy to execute this file in Anaconda prompt environment. These system programs are written in Python as it is more flexible to perform Machine Learning techniques.

CHAPTER 4

DESIGN

The most creative and hard component of system development is system design. "System Design" is the process of establishing a device, architecture, modules, interfaces, and data for a system to meet specified criteria utilizing various approaches and ideas.

4.1 DESIGN USING UML DIAGRAMS

UML (Unified Modeling Language) is a language for describing, visualising, building, and documenting software objects.

Connecting shapes that represent an item or class to other shapes to depict connections and the flow of information and data is the key to creating a UML diagram. Business analysts, software architects, and developers use UML to define, specify, create, and record current and new business processes, as well as the structure and behaviour of software artefacts. The following objectives can be met by employing UML diagrams:

- * To use object-oriented principles to depict full systems (rather than just the software part).
- * To create a system that has clear idea coupling as well as executable code.
- * To think about the scaling issues that come with sophisticated and vital systems.
- * To develop a modelling language that can be used by both people and machines.

Building Blocks of UML:

There are three sorts of building blocks in the UML vocabulary. They're:

- Things
- Relationships
- Diagrams

Things:

Things are anything that is a real-world creature or item. There are four different categories of things. Structured Things, Behavioral Things, Grouping Things, and A notational Things are the four categories.

Relationships:

It shows how things are connected in significant ways. It depicts the relationships between things and determines an application's functionality. Dependency, Association, Generalization, and Realization are the four forms of relationships.

Diagrams:

The diagrams are the visual representations of the models, which include symbols and text. In the context of the UML diagram, each symbol has a particular meaning. Structural Diagrams, Behavioral Diagrams, and Interaction Diagrams are the three types of UML diagrams.

4.2 USE CASE DIAGRAM

A use case diagram is a diagram that depicts the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It describes the activities, services, and functionalities that a system/subsystem of an application requires. It shows the high-level functionality of a system as well as how the user interacts with it.

The following are some guidelines to follow while creating a use case diagram:

1. The actor or use case of a system should be given a relevant and meaningful name.
2. An actor's communication with a use case must be defined in a comprehensible manner.
3. Predetermined notations to be utilized as and when needed.
4. Among the numerous interactions between the use case and actors, the most significant interactions should be depicted.

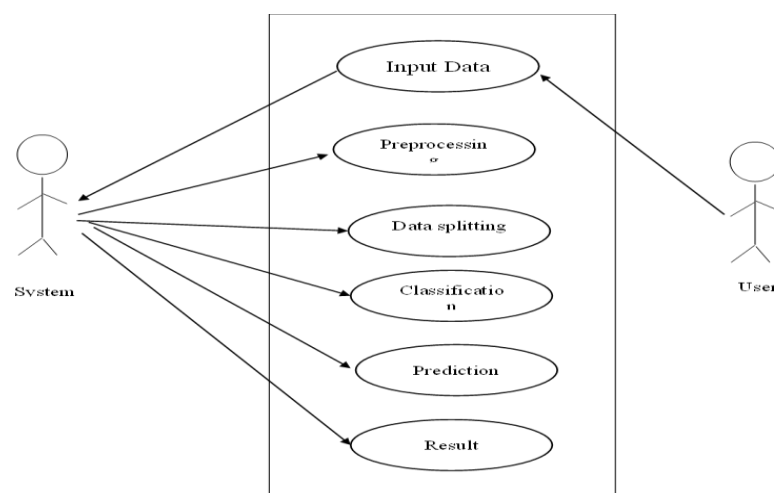


Fig4.2.1 Use case diagram

4.3 CLASS DIAGRAM

A static diagram is a class diagram. It depicts an application's static view. A class diagram is used not only for visualizing, describing, and documenting many parts of a system, but also for creating executable code for a software program. A class diagram depicts a class's properties and actions, as well as the system's limitations. Because class diagrams are the only UML diagrams that can be directly mapped with object-oriented languages, they are frequently utilized in the modelling of object-oriented systems. A collection of classes, interfaces, affiliations, collaborations, and restrictions are shown in a class diagram. A structural diagram is another name for it.

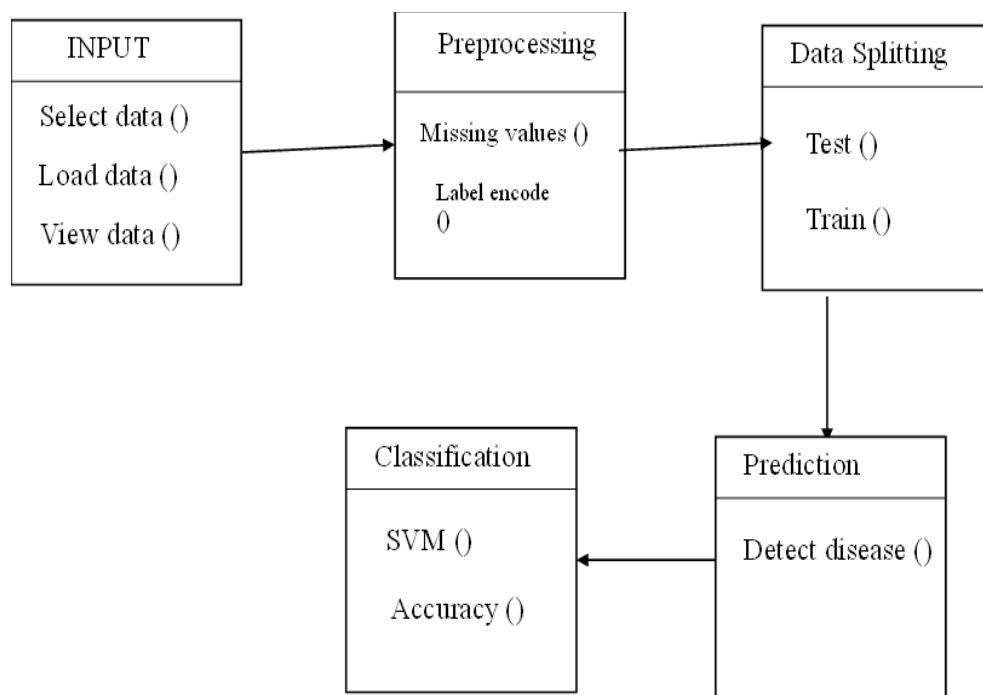


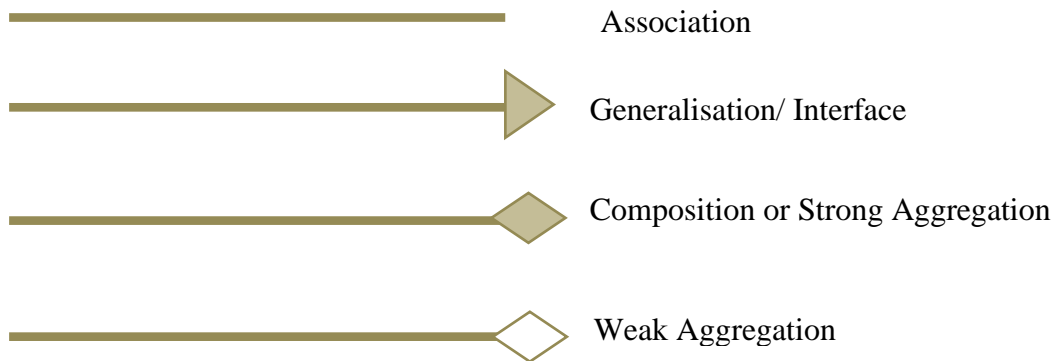
Fig4.3.1 Class diagram

Each class is represented by a rectangle that is divided into three compartments: name, attributes, and operation. Modifiers are used to determine the visibility of characteristics and actions. There are three sorts of modifiers.

- + is used for public visibility

- # is used for protected visibility
- -is used for private visibility

The Exact Meaning of the Arrows



4.4 SEQUENCE DIAGRAM

The most frequent type of interaction diagram is a sequence diagram.

Interaction diagram -

An interaction diagram is a diagram that depicts a system's interacting behavior. We employ several types of interaction diagrams to capture various elements and components of interaction in a system since visualizing the interactions in a system can be difficult. The flow of messages through the system is depicted in the sequence diagram, also known as an event diagram. It assists

display of a wide range of dynamic environments.

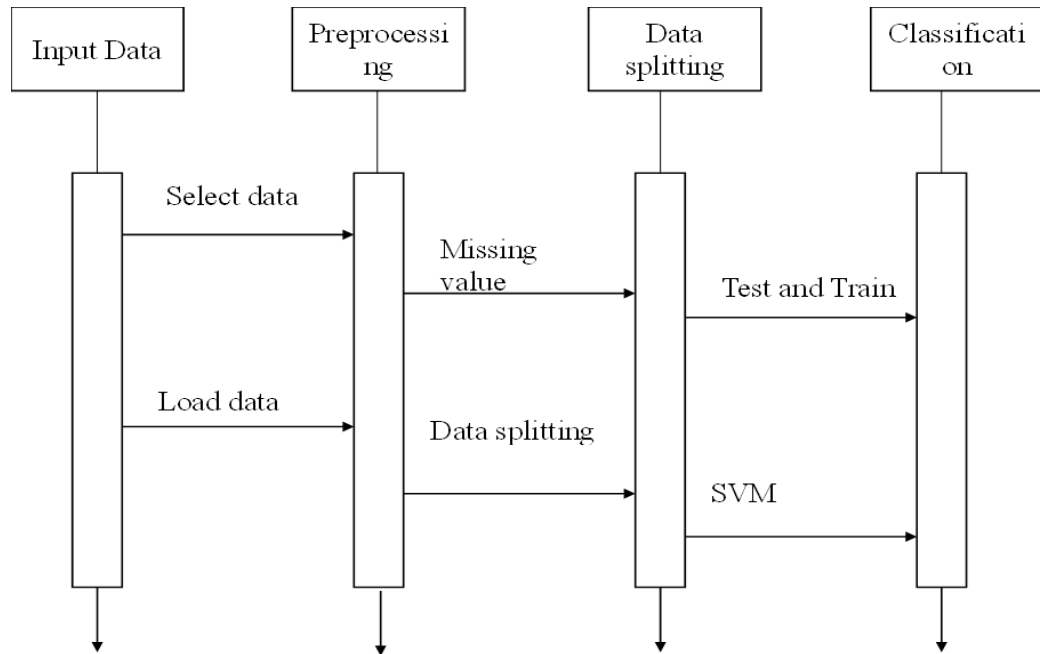


Fig4.4.1 Sequence diagram

The user gives the system the input frame initially in this Sequence design. When the user starts performing hand motions, the system is forced to create random RPS. Before choosing the winner based on the winning rules, the system must now examine and recognize the characteristics.

4.5 COLLABORATION DIAGRAM

The cooperation diagram is used to depict how items in a system are related to one another. The sequence and cooperation diagrams both depict the same data, but in different ways. It illustrates the architecture of the object living in the system, rather than the flow of messages, because it is based on object-oriented programming. A feature is one of numerous that make up an item. Several items in the system are related to one another. The collaboration diagram, also known as a communication diagram, is used to depict the architecture of an item in a system.

When object organisation is the major focus, the collaboration diagram is employed, and these are better suited for portraying simpler interactions of a smaller number of items.

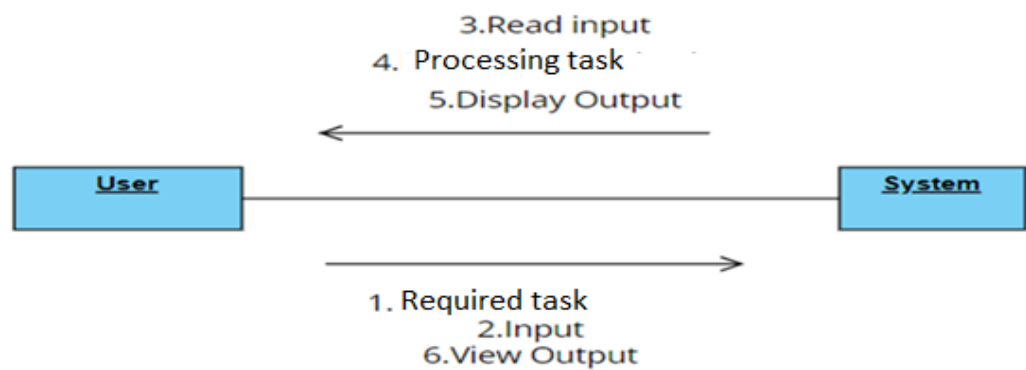


Fig4.5.1 Collaboration diagram

4.7 ACTIVITY DIAGRAM

In UML, the activity diagram is used to show the system's flow of control rather than its implementation. It can imitate both concurrent and sequential tasks. The endeavour diagram makes it easier to visualise the flow of labour from one action to the next. It focused on the condition of flow as well as the order in which it occurs. The roles of effort diagrams are identical to those of the other four diagrams. It captures the system's dynamic behaviour. The other four diagrams demonstrate message flow from one object to another, while the exercise diagram shows message flow from one recreation to another.

An activity diagram depicts the process flow from a start point to a completion point, highlighting the several other pathways that exist while the task is being carried out. A leisure diagram can be used to show both sequential and concurrent processing of tasks. They are commonly used in business and procedure modelling, where they are primarily utilized to describe the dynamic features of a

system. A flowchart and an activity sketch are extremely similar.

Activity Diagram Notations –

- * Initial State-The initial state depicts the starting state before an activity takes place.
- * Action or Activity State- An activity is the result of an action being performed on or by objects. A rectangle with rounded sides is used to depict an activity. In general, each action or event that occurs is represented as an activity.
- * Action Flows or Control Flows — Also known as routes and edges, action flows or control flows are a type of flow. They're utilized to depict the change from one activity state to the next.
- * Final State or End State - A Final State or End State is the state that the system achieves when a certain operation or action is completed. In a state machine diagram, the ultimate state is represented by a filled circle within a circle notation. Multiple end states might exist in a system or process.

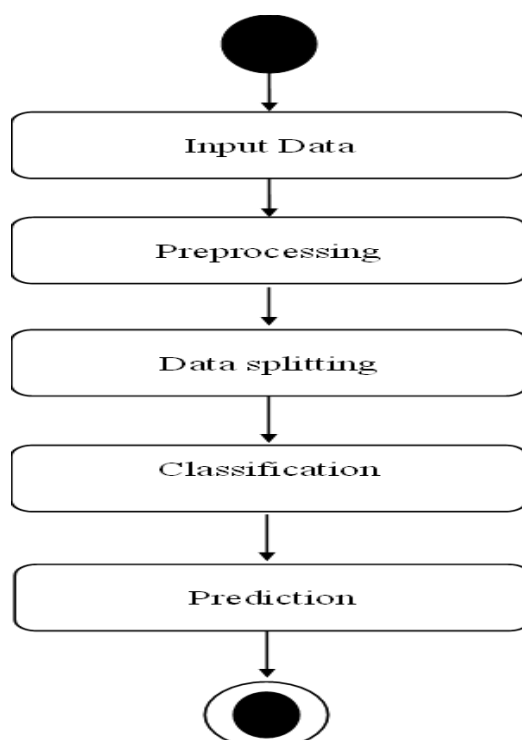


Fig4.7.1 Activity diagram

CHAPTER 5

IMPLEMENTATION

5.1 MODULES:

- Data selection
- Data preprocessing
- Data splitting
- Classification
- Result Generation

5.2 MODULES DESCRIPTION:

5.2.1: DATA SELECTION:

- The input data was collected from dataset repository.
- In our process, the Alzheimer’s disease dataset is used.
- This set consists of a longitudinal collection of 150 subjects aged 60 to 96.
- Each subject was scanned on two or more visits, separated by at least one year for a total of 373 imaging sessions.
- For each subject, 3 or 4 individual T1-weighted MRI scans obtained in single scan sessions are included.
- The subjects are all right-handed and include both men and women.
- 72 of the subjects were characterized as non-demented throughout the study.
- 64 of the included subjects were characterized as demented at the time of their initial visits and remained so for subsequent scans, including 51 individuals with mild to moderate Alzheimer’s disease.
- Another 14 subjects were characterized as non-demented at the time of their initial visit and were subsequently characterized as demented at a later visit.

5.2.2: DATA PREPROCESSING:

- Data pre-processing is the process of removing the unwanted data from the dataset.
- Pre-processing data transformation operations are used to transform the dataset into a structure suitable for machine learning.
- This step also includes cleaning the dataset by removing irrelevant or corrupted data that can affect the accuracy of the dataset, which makes it more efficient.
- Missing data removal
- Encoding Categorical data
- Missing data removal: In this process, the null values such as missing values and Nan values are replaced by 0.
- Missing and duplicate values were removed and data was cleaned of any abnormalities.
- Encoding Categorical data: That categorical data is defined as variables with a finite set of label values.
- That most machine learning algorithms require numerical input and output variables.

5.2.3: DATA SPLITTING:

- During the machine learning process, data are needed so that learning can take place.
- In addition to the data required for training, test data are needed to evaluate the performance of the algorithm in order to see how well it works.

- In our process, we considered 70% of the Alzheimer's disease dataset to be the training data and the remaining 30% to be the testing data.
- Data splitting is the act of partitioning available data into two portions, usually for cross-validator purposes.
- One Portion of the data is used to develop a predictive model and the other to evaluate the model's performance.
- Separating data into training and testing sets is an important part of evaluating data mining models.
- Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing.

5.2.4: CLASSIFICATION:

- In machine learning, classification refers to a predictive modelling problem where a class label is predicted for a given example of input data.
- Classification is the task of predicting a discrete class label. Regression is the task of predicting a continuous quantity.
- In machine learning, classification is a supervised learning concept which basically categorizes a set of data into classes.
- Before classification, we should have split the data into test and train.
- Most of data's are used for training and smaller portion of the data's are used for testing.
- Training data is used for evaluate the model and testing data is used for predictive the model.
- After data splitting, we have to implement the classification algorithm.
- In our process, we have to use, support vector machine (SVM).

- SVM It is basically a representation of different classes in a hyper plane in multidimensional space.
- The hyper plane will be generated in an iterative manner by SVM so that the error can be minimized the goal of SVM is to divide the datasets into classes to find a maximum marginal hyper plane.
- Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges.
- Support Vectors are simply the coordinates of individual observation. The SVM classifier is a frontier that best segregates the two classes (hyper-plane/ line).
- The choice of SVM for the model development task was informed by the fact that it is robust even with limited training data, and not prone to local extremum.
- SVM classifies training instances belonging to either of two classes by fitting a separation boundary (hyper plane) between the classes such that the margin between the boundary and either class is maximized.

5.2.5: RESULT GENERATION:

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,

- **Accuracy**

Accuracy of classifier refers to the ability of classifier. It predicts the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.

$$AC = (TP+TN) / (TP+TN+FP+FN)$$

- **Precision**

Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives.

$$\text{Precision} = TP / (TP+FP)$$

- **Recall**

Recall is the number of correct results divided by the number of results that should have been returned. In binary classification, recall is called sensitivity. It can be viewed as the probability that a relevant document is retrieved by the query.

$$\text{Recall} = TP / (TP+FN)$$

5.3 ALGORITHM

- In machine learning, classification refers to a predictive modelling problem where a class label is predicted for a given example of input data.
- Classification is the task of predicting a discrete class label. Regression is the task of predicting a continuous quantity.
- In machine learning, classification is a supervised learning concept which basically categorizes a set of data into classes.
- Before classification, we should have split the data into test and train.
- Most of data's are used for training and smaller portion of the data's are used for testing.
- Training data is used for evaluate the model and testing data is used for predictive the model.
- After data splitting, we have to implement the classification algorithm.
- In our process, we have to use, support vector machine (SVM).
- **SVM** It is basically a representation of different classes in a hyper plane in multidimensional space.
- The hyper plane will be generated in an iterative manner by SVM so that the error can be minimized the goal of SVM is to divide the datasets into classes to find a maximum marginal hyper plane.
- Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges.
- Support Vectors are simply the coordinates of individual observation. The SVM classifier is a frontier that best segregates the two classes (hyper-plane/ line).

- The choice of SVM for the model development task was informed by the fact that it is robust even with limited training data, and not prone to local extremum.

SVM classifies training instances belonging to either of two classes by fitting a separation boundary (hyper plane) between the classes such that the margin between the boundary and either class is maximized.

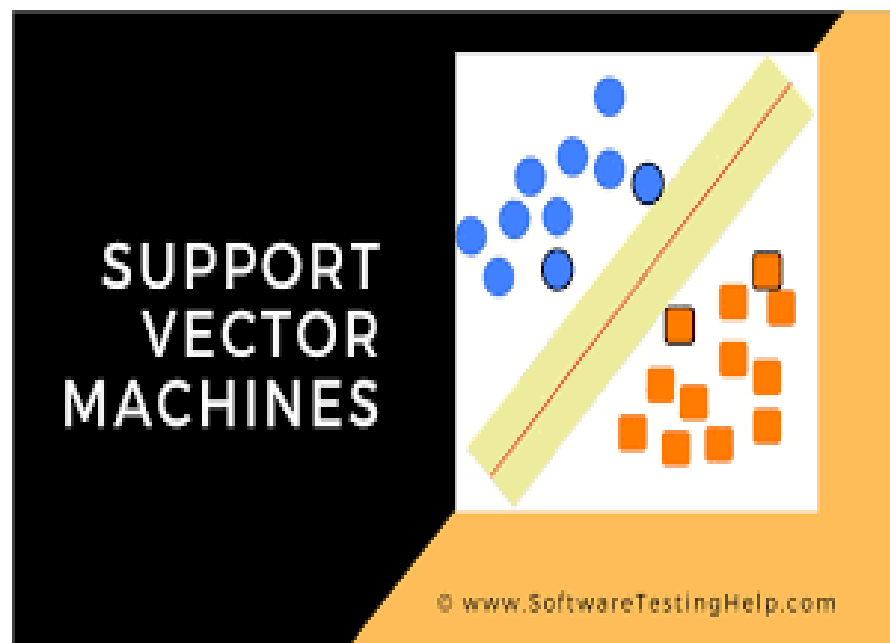


Figure 5.3.1: Support vector machine

SAMPLE CODE

```

#import packages-----
import seaborn as sns
import pandas as pd
from sklearn.svm import SVC
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

#1.data selection-----
-----

dataframe=pd.read_csv("dataset2.csv")
print("-----")
print()
print("Data Selection")
print(dataframe.head())
print()

#2.pre processing-----
-----

#checking missing values
print("-----")
print()
print("Before Handling Missing Values")
print()
print(dataframe.isnull().sum())
print()

#replace the missing values by 0
median = dataframe['MMSE'].median()
dataframe['MMSE'].fillna(median, inplace=True)

print("-----")
print("After Handling Missing Values")

```

```

print("1.Remove missing values in MMSE-----")
print()
print(dataframe.isnull().sum())
print()
median = dataframe['SES'].median()
dataframe['SES'].fillna(median, inplace=True)
print("-----")
print()
print("2.Remove missing values in SES-----")
print(dataframe.isnull().sum())
print()

#visulaization-----
dataframe['Group'] =
dataframe['Group'].replace(['Converted'], ['Demented'])
dataframe['Group'] =
dataframe['Group'].replace(['Converted'], ['Demented'])
sns.countplot(x='Group', data=dataframe)

#label encoding
#Encode columns into numeric
print("-----")
print()
print("Before Label Encoding")
print()
print(dataframe['Group'].head())
label_encoder = preprocessing.LabelEncoder()
print("-----")
print()
print("After Label Encoding")
print()
dataframe['Group']=
label_encoder.fit_transform(dataframe['Group'])
print(dataframe['Group'].head())

```

```

dataframe['M/F']=
label_encoder.fit_transform(dataframe['M/F'])
dataframe['Hand']=
label_encoder.fit_transform(dataframe['Hand'])
#3.data splitting-----
-----

feature_col_names = ["M/F", "Age", "EDUC", "SES",
"MMSE", "eTIV", "nWBV", "ASF"]
predicted_class_names = ['Group']

X = dataframe[feature_col_names].values
y = dataframe[predicted_class_names].values

#splitting the x and y into test and train
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.25, random_state=2)

#4.classification algorithms-----
-----

#svm
svm = SVC(kernel="linear", C=0.1,random_state=0)
svm.fit(X_train, y_train.ravel())
pred = svm.predict(X_test)

#confusion matrix
print("-----")

print("Performance Metrics")
cm1=confusion_matrix(y_test,pred)
print()
print("1.Confusion Matrix",cm1)
print()

```

#find the performance metrics

```
TP = cm1[0][0]
```

```
FP = cm1[0][1]
```

```
FN = cm1[1][0]
```

```
TN = cm1[1][1]
```

```
#Total TP,TN,FP,FN
```

```
Total=TP+FP+FN+TN
```

#Accuracy Calculation

```
accuracy1=((TP+TN)/Total)*100
```

```
print("2.Accuracy",accuracy1,'%')
```

```
print()
```

#Precision Calculation

```
precision=TP/(TP+FP)*100
```

```
print("3.Precision",precision,'%')
```

```
print()
```

#Sensitivity Calculation

```
Sensitivity=TP/(TP+FN)*100
```

```
print("4.Sensitivity",Sensitivity,'%')
```

```
print()
```

#specificity Calculation

```
specificity = (TN / (TN+FP))*100
```

```
print("5.specificity",specificity,'%')
```

```
print()
```

#predict the disease

```
if y[0]== 0:
```

```
    print("-----")
```

```
    print('\n Non Dementia ')
```

```
    print("-----")
```

```
else:
```

```
    print("-----")
```

```
    print('\n Demtia ')
```

```
    print("-----")
```

CHAPTER 6

EXECUTION PROCEDURE AND TESTING

1.JUPYTER NOTEBOOK:

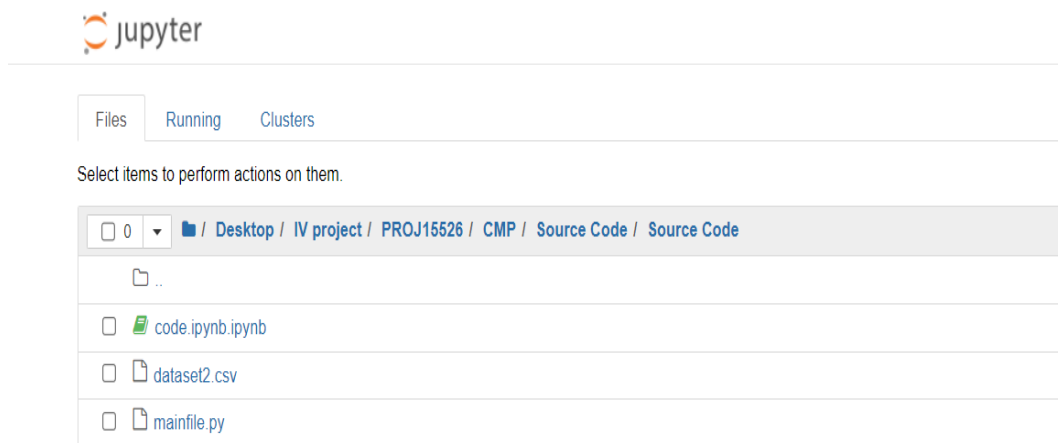


Figure 6.1: List of files

2.DATASET:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Subject ID	MRI ID	Group	Visit	MR Delay	M/F	Hand	Age	EDUC	SES	MMSE	CDR	eTIV	nWBV	ASF
2	OAS2_000	OAS2_000	Nondemer	1	0	M	R	87	14	2	27	0	1987	0.696	0.883
3	OAS2_000	OAS2_000	Nondemer	2	457	M	R	88	14	2	30	0	2004	0.681	0.876
4	OAS2_000	OAS2_000	Demented	1	0	M	R	75	12		23	0.5	1678	0.736	1.046
5	OAS2_000	OAS2_000	Demented	2	560	M	R	76	12		28	0.5	1738	0.713	1.01
6	OAS2_000	OAS2_000	Demented	3	1895	M	R	80	12		22	0.5	1698	0.701	1.034
7	OAS2_000	OAS2_000	Nondemer	1	0	F	R	88	18	3	28	0	1215	0.71	1.444
8	OAS2_000	OAS2_000	Nondemer	2	538	F	R	90	18	3	27	0	1200	0.718	1.462
9	OAS2_000	OAS2_000	Nondemer	1	0	M	R	80	12	4	28	0	1689	0.712	1.039
10	OAS2_000	OAS2_000	Nondemer	2	1010	M	R	83	12	4	29	0.5	1701	0.711	1.032
11	OAS2_000	OAS2_000	Nondemer	3	1603	M	R	85	12	4	30	0	1699	0.705	1.033
12	OAS2_000	OAS2_000	Demented	1	0	M	R	71	16		28	0.5	1357	0.748	1.293
13	OAS2_000	OAS2_000	Demented	3	518	M	R	73	16		27	1	1365	0.727	1.286
14	OAS2_000	OAS2_000	Demented	4	1281	M	R	75	16		27	1	1372	0.71	1.279
15	OAS2_000	OAS2_000	Nondemer	1	0	F	R	93	14	2	30	0	1272	0.698	1.38
16	OAS2_000	OAS2_000	Nondemer	2	742	F	R	95	14	2	29	0	1257	0.703	1.396
17	OAS2_000	OAS2_000	Demented	1	0	M	R	68	12	2	27	0.5	1457	0.806	1.205
18	OAS2_000	OAS2_000	Demented	2	576	M	R	69	12	2	24	0.5	1480	0.791	1.186
19	OAS2_001	OAS2_001	Demented	1	0	F	R	66	12	3	30	0.5	1447	0.769	1.213
20	OAS2_001	OAS2_001	Demented	2	854	F	R	68	12	3	29	0.5	1482	0.752	1.184
21	OAS2_001	OAS2_001	Nondemer	1	0	F	R	78	16	2	29	0	1333	0.748	1.316
22	OAS2_001	OAS2_001	Nondemer	2	730	F	R	80	16	2	29	0	1323	0.738	1.326
23	OAS2_001	OAS2_001	Nondemer	3	1598	F	R	83	16	2	29	0	1323	0.718	1.327
24	OAS2_001	OAS2_001	Nondemer	1	0	F	R	81	12	4	30	0	1230	0.715	1.427
25	OAS2_001	OAS2_001	Nondemer	2	643	F	R	82	12	4	30	0	1212	0.72	1.448
26	OAS2_001	OAS2_001	Nondemer	3	1456	F	R	85	12	4	29	0	1225	0.71	1.433
27	OAS2_001	OAS2_001	Demented	1	0	M	R	76	16	3	21	0.5	1602	0.697	1.096
28	OAS2_001	OAS2_001	Demented	2	504	M	R	77	16	3	16	1	1590	0.696	1.104
29	OAS2_001	OAS2_001	Demented	1	0	M	R	88	8	4	25	0.5	1651	0.66	1.063

Figure 6.2: Dataset

3. DATA SELECTION:

```

data = data[data['Source Code'] == 'OAS2']

Data Selection
  Subject ID      MRI ID      Group  Visit  ...  CDR  eTIV  nWBV  ASF
0  OAS2_0001  OAS2_0001_MR1  Nondemented    1  ...  0.0  1987  0.696  0.883
1  OAS2_0001  OAS2_0001_MR2  Nondemented    2  ...  0.0  2004  0.681  0.876
2  OAS2_0002  OAS2_0002_MR1   Demented    1  ...  0.5  1678  0.736  1.046
3  OAS2_0002  OAS2_0002_MR2   Demented    2  ...  0.5  1738  0.713  1.010
4  OAS2_0002  OAS2_0002_MR3   Demented    3  ...  0.5  1698  0.701  1.034

[5 rows x 15 columns]

```

Figure 6.3: Head of dataset

4. PREPROCESSING:

```

Before Handling Missing Values

```

```

Subject ID      0
MRI ID          0
Group           0
Visit           0
MR Delay        0
M/F             0
Hand            0
Age             0
EDUC            0
SES             29
MMSE            2
CDR              0
eTIV            0
nWBV            0
ASF             0
dtype: int64

```

```

After Handling Missing Values

```

```

1.Remove missing values in MMSE-----

```

```

Subject ID      0
MRI ID          0
Group           0
Visit           0
MR Delay        0
M/F             0
Hand            0
Age             0
EDUC            0
SES             29
MMSE            0
CDR              0
eTIV            0
nWBV            0
ASF             0
dtype: int64

```

```

-----
2.Remove missing values in SES-----
Subject ID    0
MRI ID       0
Group        0
Visit        0
MR Delay     0
M/F         0
Hand        0
Age         0
EDUC        0
SES         0
MMSE        0
CDR         0
eTIV        0
nWBV        0
ASF         0
dtype: int64
-----

```

Figure 6.4.1: Removing Null Values

```

-----
Before Label Encoding

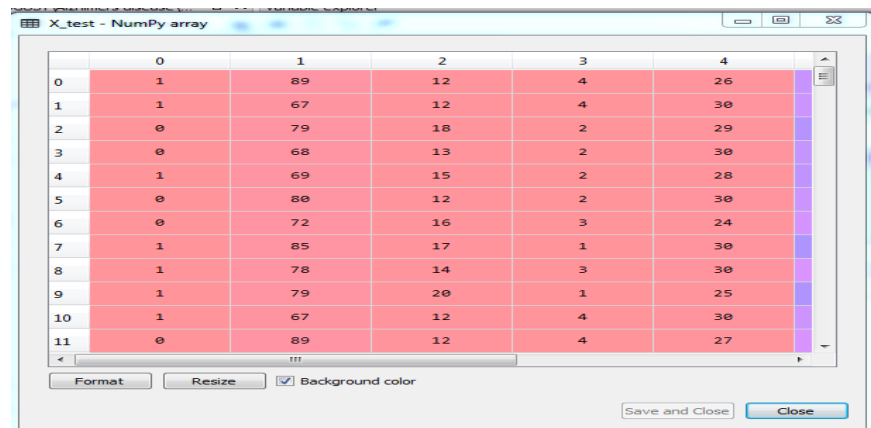
0    Nondemented
1    Nondemented
2     Demented
3     Demented
4     Demented
Name: Group, dtype: object
-----

After Label Encoding

0    1
1    1
2    0
3    0
4    0
Name: Group, dtype: int32
-----

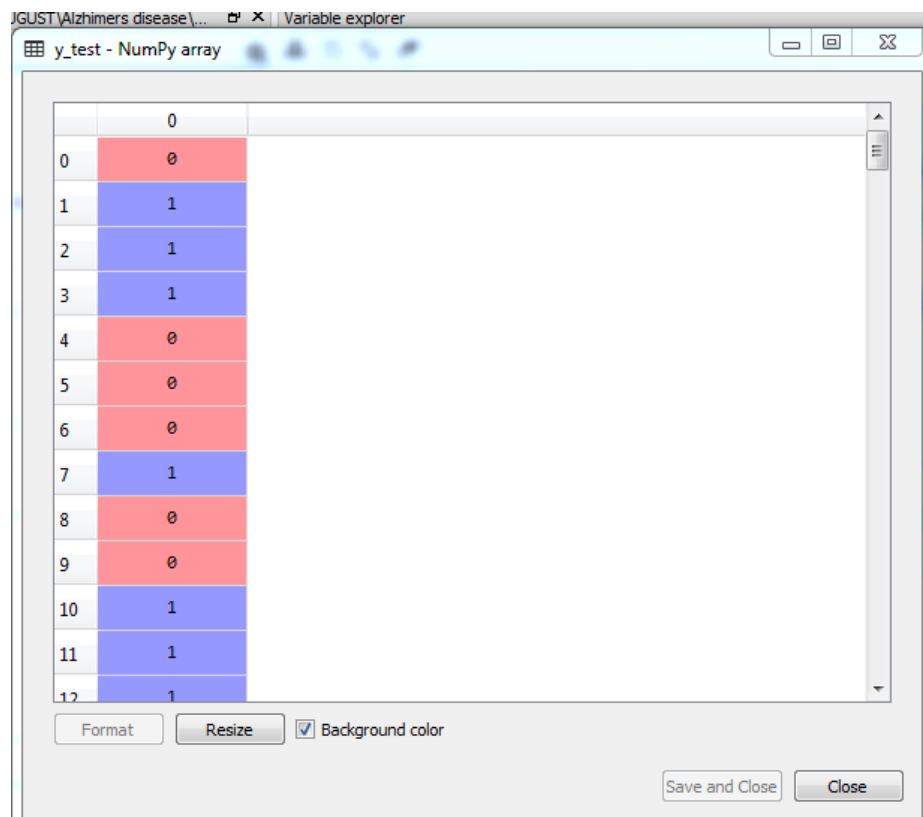
```

Figure 6.4.2: Label Encoding

5.DATA SPLITTING:


	0	1	2	3	4
0	1	89	12	4	26
1	1	67	12	4	30
2	0	79	18	2	29
3	0	68	13	2	30
4	1	69	15	2	28
5	0	80	12	2	30
6	0	72	16	3	24
7	1	85	17	1	30
8	1	78	14	3	30
9	1	79	20	1	25
10	1	67	12	4	30
11	0	89	12	4	27

Figure 6.5.1 : X_test dataset



	0
0	0
1	1
2	1
3	1
4	0
5	0
6	0
7	1
8	0
9	0
10	1
11	1
12	1

Figure 6.5.2: y_test dataset

6.PERFORMANCE ANALYSIS:

Performance Metrics

1.Confusion Matrix $\begin{bmatrix} 50 & 17 \\ 11 & 72 \end{bmatrix}$
 2.Accuracy 81.33333333333333 %
 3.Precision 74.6268656716418 %
 4.Sensitivity 81.9672131147541 %
 5.specificity 80.89887640449437 %

Figure 6.6 : Result Performance

7.VISUALIZATION:

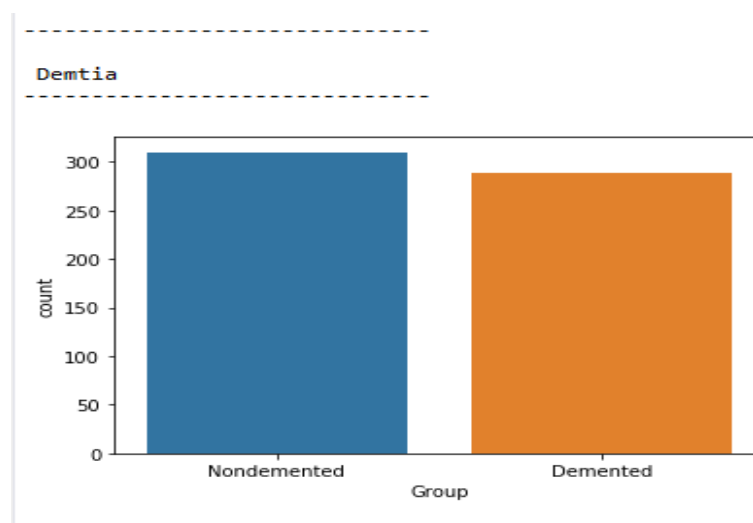


Figure 6.7:Classification of disease

6.2 Testing

Software testing is an important part of software quality assurance since it is the final examination of the specification, graph, and coding. For the software engineer, testing presents an interesting anomaly. The goal of testing is to find flaws. Testing is the process of attempting to find every possible flaw or vulnerability in a work product. It allows you to examine the performance of individual components, sub-assemblies, assemblies, and/or a finished product. It is the process of testing software in order to ensure that it satisfies the requirements and expectations of the user and does not fail in an unacceptable manner.

Objectives of Testing:

- Testing is a process of executing an application with the intent of discovering an error. □ A top check case is one that has an excessive probability of finding an error but undiscovered error.
- A profitable test is one that uncovers an as yet undiscovered error. Testing ought to systematically find unique training of mistakes in a minimum quantity of time and with a minimum quantity of effort. A secondary benefit of testing is that it demonstrates that the software appears to be working as mentioned in the specifications.

Testing Strategies

1.VALIDATION TESTING:

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many, But a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer.

2. USER ACCEPTANCE TESTING:

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

CHAPTER 7

RESULTS AND PERFORMANCE EVALUATION

Feature selection using CFS as discussed earlier was conducted with attribute selection toolbox in. In evaluating the models from previous studies, we used Weka where previous studies had used it for model development. Training of ML models and validation of performance for ADD vs. HC discrimination was based on 10-fold cross-validation scheme repeated 10 times. The data (Dataset) were randomly repartitioned after each run to ensure that data subsets used for training and validation varied from the ones used in the preceding run. This way, a more robust average performance is obtained. Classification performance metrics of primary consideration were measures of SN and SP in accordance with international recommendations for clinically usable AD biomarkers. A performance threshold of 70% for SN and SP was adopted in the model development task. This is on the grounds that the diagnostic accuracy of human experts reaches 77% with sensitivity and specificity reaching 81% and 70%, respectively. Moreover, sensitivity and specificity greater than 80% is the target performance for ideal AD biomarkers.

We successfully replicated 7 models for classification of ADD subjects and HCs. The model proposed by could not be replicated because it was originally trained on a dataset not available to us. Nevertheless, we constructed a model with Dataset based on the ML algorithm and blood protein panel proposed by the study. Only existing models constructed with blood proteins available in our study dataset were investigated in this study. Nearly all the models achieved SN, SP, and AUC greater than 80%, 60%, and 0.70, respectively. However, when evaluated for possible detection of early AD by classifying MCI and HC with Dataset 2, the SN values of the models remained moderately high while their SP values drastically dropped (with only one model achieving up to 50%). This implies that the models may have undesirably high levels of false positives when applied for early disease detection.

CHAPTER 8

CONCLUSION AND FURTHER WORK

Conclusion:

The results suggest that it may be feasible to detect early AD using a profile of non-amyloid proteins that identify the metabolic processes that accompany or precede the disease. It may be therefore possible to detect the disease. This system was proposed for efficient disease detection using machine learning algorithms such as SVM. Experimental results analysis showed that our proposed method is efficient and can achieve better performance results on average when compared with existing.

Further work:

As a future work, a growing understanding of how the disease disrupts the brain has led to potential Alzheimer's treatments that short-circuit basic disease processes. Future Alzheimer's treatments may include a combination of medications, similar to how treatments for many cancers or HIV/AIDS include more than a single drug.

LIST OF FIGURES

Figure	Page no
1.1.1 MEMORY LOSING	2
1.1.2 AGES OF AD	3
1.1.3 SUPPORT VECTOR MACHINE	4
4.2.1 USE CASE DIAGRAM	16
4.3.1 CLASS DIAGRAM	17
4.4.1 SEQUENCE DIAGRAM	19
4.5.1 COLLABORATION DIAGRAM	20
4.6.1 ACTIVITY DIAGRAM	22
5.3.1 SUPPORT VECTOR MACHINE	29
6. 1 LIST OF FILES	33
6.2 DATASET	33
6.3 HEAD OF DATASET	34
6.4.1 REMOVING NULL VALUES	35
6.4.2 LABEL ENCODING	35
6.5.1 X_TEST DATASET	36
6.5.2 Y_TEST DATASET	36
6.6 RESULT PERFORMANCE	37
6.7 CLASSIFICATION OF DISEASE	37

REFERENCES

- [1] Chima S. Eke, Emmanuel Jammeh, Xinzhong Li, Camille Carrol, Stephen Pearson, Emmanuel Ifeachor, "Early Detection of Alzheimer's Disease with Blood Plasma Proteins using Support Vector Machines," IEEE Journal of Biomedical and Health Informatics, Vol. 25, no. 3, pp. 218-226, 2020.
- [2] B. Dubois et al., "Preclinical Alzheimer's disease: definition, natural history, and diagnostic criteria," Alzheimer's & Dementia, vol. 12, no. 3, pp. 292-323, 2016.
- [3] M. S. Albert et al., "The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the National Institute on Aging Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease," Alzheimer's & Dementia, vol. 7, no. 3, pp. 270- 279, 2011.
- [4] G. M. McKhann et al., "The diagnosis of dementia due to Alzheimer's disease: Recommendations from the National Institute on Aging Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease," Alzheimer's & Dementia, vol. 7, no. 3, pp. 263-269, 2011.
- [5] K. H. Tse and K. Herrup, "Re-imagining Alzheimer's disease—the diminishing importance of amyloid and a glimpse of what lies ahead," Journal of Neurochemistry, vol. 143, no. 4, pp. 432-444, 2017.