An Industry Oriented Mini Project Report (CS705PC)

On

SCHIZOPHRENIA SEVERITY LEVEL IDENTIFICATION

Submitted in partial fulfilment of the requirements for the award of the degree

Bachelor of Technology

in

Computer Science and Engineering (Data Science)

by

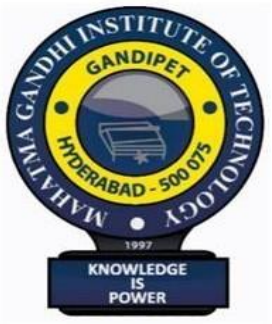
Ms. K. Lakshmi Sai Supriya (20261A6728)

Mr. K. Sampreeth (20261A6725)

Under the Guidance of

Mrs. J. Sreedevi

(Assistant Professor)



DEPARTMENT OF EMERGING TECHNOLOGIES

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY (Autonomous)

(Affiliated to Jawaharlal Nehru Technological University Hyderabad) Gandipet, Hyderabad-500075, Telangana (India)

2023-2024

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY
(AUTONOMOUS)

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)

Gandipet, Hyderabad – 500075, Telangana.

CERTIFICATE

This is to certify that the project entitled **"SCHIZOPHRENIA SEVERITY LEVEL IDENTIFICATION"** is being submitted by **K. LAKSHMI SAI SUPRIYA (20261A6728)** and **K. SAMPREETH (20261A6725)** in partialfulfillment of the requirements for the Mini Project in COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE) is a record of bonafide work carried out by them.

The results of the investigations enclosed in this report have been verified and found satisfactory.

Project Guide	Head of the Department
Mrs. J. Sreedevi	**Dr. M. Rama Bai**
Assistant Professor	Professor, Dept. of ET

External Examiner

DECLARATION

This is to certify that the work reported in the project titled "**Schizophrenia Severity Level Identification**" is a record of work done by us in the Department of Emerging Technologies, Mahatma Gandhi Institute of Technology, Hyderabad.

No part of the work is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the work done entirely by us and not copied from any other source.

K. Lakshmi Sai Supriya

20261A6728

K. Sampreeth

20261A6725

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crowns all efforts with success. They have been a guiding light and source of inspiration towards the completion of the project.

We would like to express our sincere thanks to **Dr. G. ChandraMohan Reddy**, **Principal MGIT**, providing the working facilities in college.

We wish to express our sincere thanks and gratitude to **Dr.M. Rama Bai, Professor and HoD**, Department of ET, MGIT, for all the timely support and valuable suggestions during the period of project.

We are extremely thankful to **Dr. B. Yadaiah, Assistant Professor**, and **M. Srikanth, Assistant Professor**, Department of ET, MGIT, mini project coordinators for their encouragement and support throughout the project.

We are extremely thankful and indebted to our internal guide **Mrs. J. Sreedevi, Assistant Professor**, Department of ET, for her constant guidance, encouragement and moral support throughout the project.

Finally, We would also like to thank all the faculty and staff of ET Department who helped us directly or indirectly, for completing this project.

K. Lakshmi Sai Supriya

20261A6728

K. Sampreeth

20261A6725

TABLE OF CONTENTS

Certificate i

Declaration ii

Acknowledgement iii

List of Figures v

List of Tables vi

Abstract vii

1. Introduction 1

    1.1. Problem Definition 2

    1.2. Existing System 2

    1.3. Proposed System 4

    1.4. Requirements Specifications 4

        1.4.1. Software Requirements 4

        1.4.2. Hardware Requirements 4

2. Literature Survey 5

3. Design and Methodology 9

    3.1. System Design 9

    3.2. Tools used 11

    3.3. Design Methodology 15

        3.3.1. UML Diagrams 15

        3.3.2. Use Case Diagrams 15

        3.3.3. Sequence Diagrams 16

        3.3.4. Activity Diagrams 17

    3.4. Modules Imported in Python 18

    3.5. Model Evaluation and Selection 19

4. Results 22

    4.1. System Testing Results 22

5. Conclusion and Future Scope 25

    5.1. Conclusion 25

    5.2. Future Scope 25

Bibliography 26

Appendix 28

LIST OF FIGURES

Figure No.	Figure Name	Page No.
Figure 1.1	MRI Images	2
Figure 1.2	Graph Kernel Based Brain states	3
Figure 3.1	Architecture Diagram	9
Figure 3.2	Data Flow Diagram	10
Figure 3.3	Files created using project	10
Figure 3.4	User Interface Diagram	11
Figure 3.5	Library/Bin folder of Anaconda tool	12
Figure 3.6	Creating shortcut of designer.exe	12
Figure 3.7	Creating a push button	13
Figure 3.8	Applying color to the text label	14
Figure 3.9	Use Case Diagram	15
Figure 3.10	Sequence Diagram	16
Figure 3.11	Activity Diagram	17
Figure 4.1	Metrics of Different Classification Algorithms.	22
Figure 4.2	Metrics results for the dataset	23
Figure 4.3	Prediction of the Schizophrenia	23
Figure 4.4	Storing the observable symptoms using UI	23
Figure 4.5	Storing the testable symptoms using UI	24
Figure 4.6	Accuracy of the Decision Tree	24

LIST OF TABLES

Table No.	Table Name	Page No.
Table 1.1	Demographic and clinical characteristics	3
Table 2.1	Comparison of Literature Survey	6

ABSTRACT

Schizophrenia is a complex mental health disorder characterized by a range of symptoms that vary in severity among individuals. Early and accurate identification of the severity level of schizophrenia is crucial for effective treatment planning and intervention[1]. This study proposes a machine learning-based approach for assessing the severity of schizophrenia in individuals based on a comprehensive analysis of diverse clinical and behavioral data.

The methodology involves the collection of multi-modal data, including neuroimaging scans, clinical assessments, and behavioral observations, from a cohort of individuals diagnosed with schizophrenia. Feature engineering techniques are applied to extract relevant information from the diverse data sources. A machine learning model, possibly a combination of traditional classifiers and deep learning architectures, is trained on this data to predict the severity level of schizophrenia[2].

The study aims to contribute to the development of a reliable and objective tool for clinicians to assist in the early and accurate identification of schizophrenia severity. Ultimately, the project seeks to advance our understanding of schizophrenia and improve clinical decision-making by providing a quantitative and standardized measure of severity, facilitating personalized treatment plans and improving outcomes for individuals living with schizophrenia.

KEYWORDS:

Schizophrenia Severity Level, Machine Learning, Neuroimaging, Clinical Assessment, Behavioral Data, Feature Engineering, Traditional Classifiers, Deep Learning, Diagnostic Tool, Early Identification, Treatment Planning, Ethical Considerations, Data Privacy, Model Evaluation, Personalized Treatment, Clinical Decision-making, Outcome Improvement.

1. INTRODUCTION

Schizophrenia is a neuro-developmental disorder associated with impairments in social andlingual abilities. It is a serious developmental disorder that impairs the ability to communicate and interact. Schizophrenia disorder impacts the nervous system and affects the overall cognitive, emotional, social and physical health of the affected individual. The range and severity of symptoms can vary widely. Common symptoms include difficulty with communication,difficulty with social interactions, obsessive interests and repetitive behaviors. Symptoms are divided into two categories viz., observable symptoms and testable symptoms. Observable symptoms include: improper eye to eye contact, Not responding when called, engaging himself in imaginative play, staying alone, epilepsy and depression. Testable symptoms include poor mathematical, logical skills and in ability to speak words and sentences[1]. This project aims at predicting the Schizophrenia severity level by analyzing a dataset consisting of the above mentioned observable and Testable symptoms, using decision Tree classification with different levels of depth.

The project comprises four modules. First module deals with storing the observable and testable symptoms data into the DB, The second module deals with the creation of comma separated value files of the data stored in the DB. The third module deals with the development of python routines to execute the three level decision tree Classification on the symptoms data set and the final module deals with the four level decision tree analysis of symptoms data[3].

The project uses the 'sklearn' module of Python to perform the Extra Tree Classification and decision tree analysis. Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed[2]. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches. Decision trees can handle both categorical and numerical data. In statistics, the logistic model is used to model the probability of a certain class or event existing such as pass/fail, win/lose, or healthy/sick.

Algorithms Used: Decision tree algorithm with varying levels of depth

The Schizophrenia diagnostic observation schedule is the current system being used for diagnosing Schizophrenia along with the expert clinical judgment. The current system doesn't use any machine learning technique in the diagnosis. The proposed system is to predict the Schizophrenia severity level by using decision tree classification with varying levels of depth.

1.1. PROBLEM STATEMENT

Schizophrenia is a complex neuro-developmental disorder that significantly affects an individual's social, linguistic, cognitive, emotional, and physical well-being. Current diagnostic methods rely on the Schizophrenia diagnostic observation schedule and expert clinical judgment, lacking the integration of machine learning techniques. This project aims to address this gap by developing a predictive model using decision tree algorithms to assess the severity level of Schizophrenia based on observable and testable symptoms.

1.2. EXISTING SYSTEM

- In recent studies, it has been shown that neuropsychiatric disorders are related to abnormal structure of the brain. Researchers are trying to find some potential neuroimaging biomarkers related to schizophrenia.
- Researchers like liang et al used diffusion tensor imaging data and found that the patients with first episode schizophrenia has lower white volume in the right temporal occipital region.

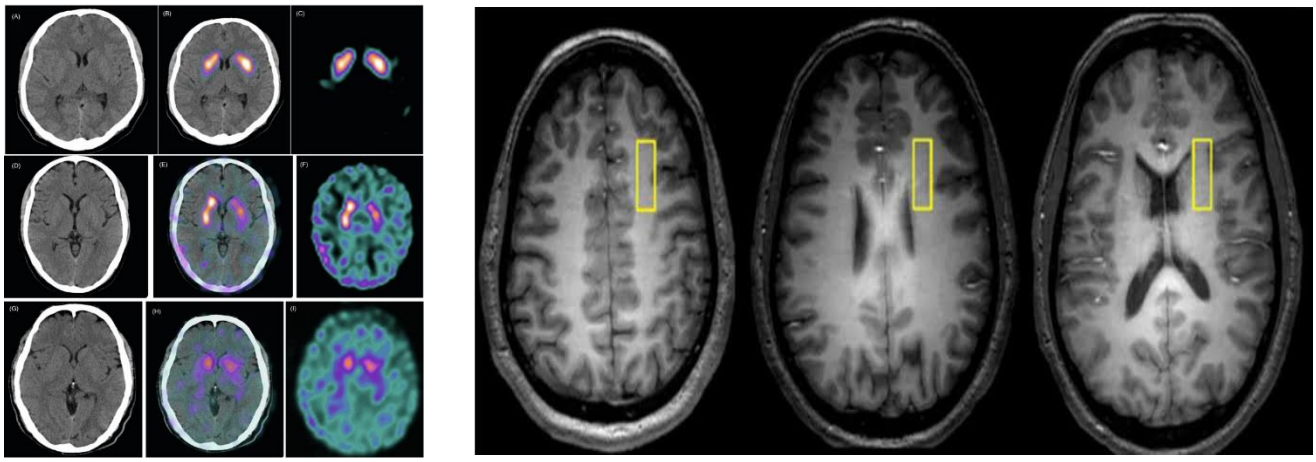


Fig 1.1 : MRI Images

- Leroux et al jointly analyzed the functional magnetic resonance imaging (fMRI) and DTI, and found that abnormal frontal-temporal pathways in the brain of schizophrenic patients may be one of the factors leading to the disease.

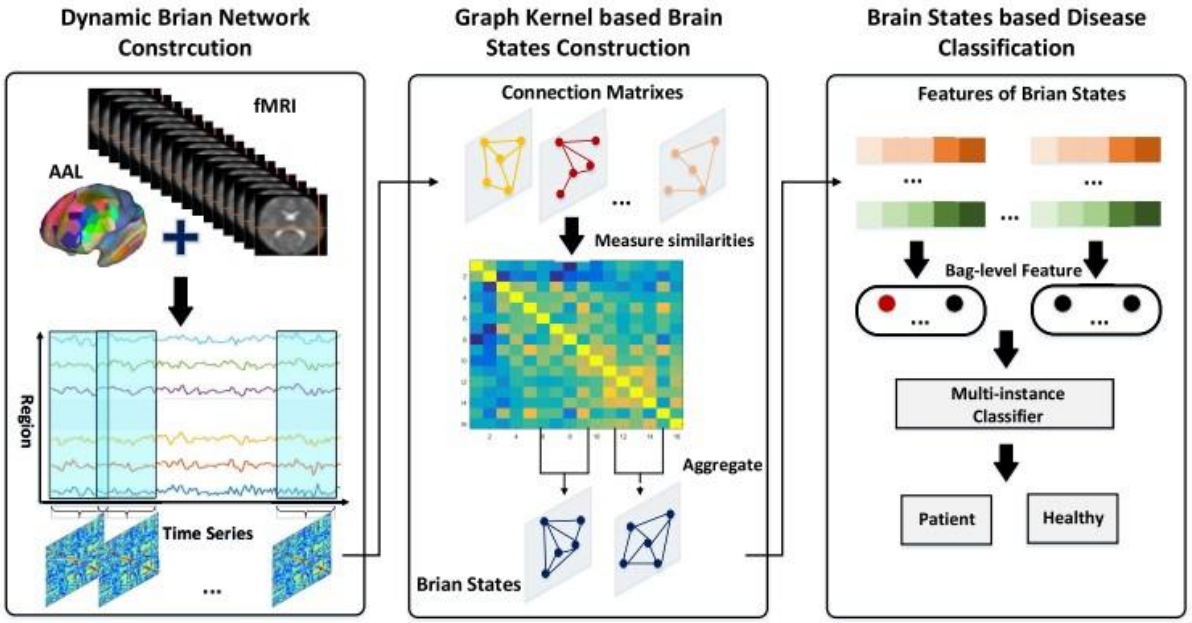


Fig 1.2 : Illustration of the graph kernel based brain states construction framework

Group	Age	Gender (M/F)	PANSS-positive	PANSS-negative	PANSS-general
Healthy	34.82 ± 11.28	46/21	-	-	-
Patient	36.75 ± 13.68	42/11	14.84 ± 4.53	14.42 ± 4.97	29.88 ± 8.27

Table 1.1 : Demographic and clinical characteristics of participants

- We use the shortest path graph kernel to measure the similarity between these connection matrices. The shortest path graph kernel compares the similarity by counting the number of shortest paths with the same length on two graphs. Therefore, the similarity between connection matrices of each subject can be written as follows:

  K(G, G 0) = X vi, uj ∈G X v 0 k ,u 0 l∈G0 δ(d(vi, uj), d(v 0 k , u 0 l)),

- where d(vi, uj) is the length of the shortest path between nodes of graph G.
- d(v 0 k , u 0 l) is the length of the shortest path between node v 0 k and node u 0 l on the graph G 0 .
- When these two shortest paths are equal in length, δ(d(vi, uj), d(v 0 k , u 0 l)) = 1, otherwise δ(d(vi, uj), d(v 0 k , u 0 l)) = 0.
- The value can be obtained by comparing all the shortest paths in two networks.
- This value measures the similarity between the two graphs.

1.3. PROPOSED SYSTEM

- To detect the disease more accurately and early we can use machine learning algorithms.
- ML algorithms can predict the severity of the disease in the patient more accurately by dividing the dataset into groups according to the characteristics of the symptoms.
- For this we take patients symptoms and create a dataset and use it in ML algorithm to predict.

1.4. REQUIREMENTS SPECIFICATION

1.4.1. Software Requirements

1. It requires a 64-bit Windows Operating System.
2. Python Qt Designer for designing user interfaces.
3. SQLite3 for storing database Entities.
4. Pyuic for converting the layout designed user interface (UI) to python code.
5. Python language for coding.

1.4.2. Hardware Requirements

1. It requires a minimum of 2.16 GHz processor.
2. It requires a minimum of 4 GB RAM.
3. It requires 64-bit architecture.
4. It requires a minimum storage of 500GB.

2. LITERATURE SURVEY

The literature survey was conducted thoroughly in order to gain in-depth understanding of the disease Schizophrenia, causes of Schizophrenia and how it can be identified easily. Various research papers have been read and the approaches used in those research papers have been understood.

1. **"GK-BSC: Graph Kernel-Based Brain States Construction with Dynamic Brain Networks and Application to Schizophrenia Identification" by XINYAN YUAN, LINGLING GU AND JIA SHUANG HUANG [1]**: The study introduces GK-BSC, a novel method using graph kernels to enhance the accuracy of identifying schizophrenia-related brain abnormalities. By capturing dynamic brain network topological properties more accurately, the approach employs multi-instance support vector machines for patient identification, offering a nuanced understanding of schizophrenia severity.

2. **"Current concepts and treatments of schizophrenia" by P. Stepnicki, M. Kondej, and A. A. Kaczor [2]:** This review highlights the public health impact of schizophrenia, emphasizing the incomplete understanding of its pathomechanism and limitations of current antipsychotics. It explores the role of G protein-coupled receptors (GPCRs) and innovative signaling mechanisms as essential considerations for drug discovery.

3. **"Endophenotypes in schizophrenia: Digging deeper to identify genetic mechanisms" by T. A. Greenwood, A. Shutes-David, and D. W. Tsuang [3]:** The review discusses the genetic heterogeneity of schizophrenia and the challenges in identifying risk variants. It emphasizes the use of neurocognitive and neurophysiological endophenotypes as stable trait markers to objectively measure underlying brain dysfunction, potentially revealing the genetic architecture of schizophrenia.

4. **"Extending schizophrenia diagnostic model to predit in first-degree relatives" by Sunil Vasu Kalmady, Animesh Kumar Paul, Russell Greiner, Rimjhim Agrawal, Serdar M. Dursun & Ganesan Venkatasubramanian [4]:** The study applies the machine-learning algorithm "EMPaSchiz" to first-degree relatives of schizophrenia patients. It demonstrates the model's potential to predict state-independent vulnerability, even in individuals without active psychosis or schizophrenia symptoms.

5. **"Early detection of schizophrenia: current evidence and future perspectives" by HEINZ HÄFNER and KURT MAURER [5]:** The research identifies prepsychotic prodromal and psychotic prephase stages in schizophrenia, emphasizing the importance of early intervention for functional impairment. Cognitive-behavioral therapy at the prepsychotic prodromal stage and a combination with low-dose antipsychotics in the psychotic prephase are discussed as potential interventions for early recognition.

Table 2.1 : Comparison of Literature Survey

S.NO	TITLE	AUTHORS	ALGORITHM / METHODOLOGY	MERITS OR ADVANTAGES	DEMERITS OR FUTURE SCOPE
1	GK-BSC: Graph Kernel-Based Brain States Construction with Dynamic Brain Networks and Application to Schizophrenia Identification	XINYAN YUAN, LINGLING GU and JIA SHUANG HUANG	The architecture consists GK-BSC method consists of three parts, including dynamic brain network construction graph kernel based brain states construction and brain states based disease classification.	The paper proposes an innovative method for constructing brain states and identifying schizophrenia. Using a graph kernel enhances topological property capture, while individualized feature extraction improves sensitivity to structural abnormalities. Experimental results demonstrate increased accuracy in classification and the identification of potential schizophrenia biomarkers, marking a significant advance in disease understanding and diagnosis.	While GK-BSC method significantly enhances schizophrenic diagnosis compared to existing brain state construction methods, future considerations include incorporating weight information from Dynamic Brain Networks (DBNs), designing more efficient features than clustering coefficients, and exploring a unified framework for joint brain state construction and classifier training to further improve recognition accuracy.

2	Current concepts and treatments of schizophrenia	P.Stepnicki, M. Kondej, A.A.Kaczor	The research paper targets Novel GPCR Signaling Mechanisms in Schizophrenia and Other Non-Classical Approaches for the Treatment of Schizophrenia.	The paper gives knowledge about various treatments for schizophrenia and focuses primarily on how GPCR signaling mechamism is used for the treatement.	Single-target drugs may not address all symptoms.
3	Endophenotypes in schizophrenia: Digging deeper to identify genetic mechanisms	T. A. Greenwood, A. Shutes-David, and D. W. Tsuang	This paper employs a review methodology to explore and evaluate promising schizophrenia endophenotypes, including prepulse inhibition, mismatch negativity, oculomotor antisaccade, letter-number sequencing, and continuous performance tests.	The research systematically reviews stable neurocognitive and neurophysiological markers (endophenotypes) for schizophrenia, shedding light on underlying genetic factors and suggesting novel treatment targets for the disorder.	The time and effort required to assess endophenotypes. Additionally, their translational relevance to the broader disorder may be limited, posing challenges in comprehensive understanding and treatment.
4	Extending schizophrenia diagnostic model to predit in first-degree relatives	Sunil Vasu Kalmady, Animesh Kumar Paul, Russell Greiner, Rimjhim Agrawal, Serdar M. Dursun & Ganesan Venkatasubramanian	The project involves a neuroimaging approach. Resting-state fMRI data from 57 first-degree relatives of schizophrenia patients were acquired, pre-processed, and subjected to machine-learned prediction using the EMPaSchiz model.	The project demonstrates robust subject selection, advanced neuroimaging techniques, and a machine learning approach, offering potential diagnostic improvements. Ethical considerations and comprehensive pre-processing enhance research integrity.	Limitations include a relatively small sample size, reliance on a single imaging modality, and potential generalization issues with the machine learning model. Subjective self-reported scores and limited clinical context present further challenges in interpretation.

5	Early detection of schizophrenia: current evidence and future perspectives	HEINZ HÄFNER and KURT MAURER	Study tells about several instruments available for screening and early recognition, including PRODscreen, PQ, Y-PARQ, SIPS screen, ERIraos, CAARMS, SIPS/SOPS, and BSABS for schizophrenia.	Early intervention focuses on symptomatic manifestations, lacking insight into the unknown disease process. Cognitive-behavioral therapy during the prepsychotic prodromal stage positively influences short-term outcomes, while combining it with low-dose antipsychotics shows efficacy in the psychotic prephase.	Long-term effects have not yet been demonstrated.

3. DESIGN AND METHODOLOGY

3.1. SYSTEM DESIGN

The project comprises four modules. First module deals with storing the observable and testable symptoms data into the DB, The second module deals with the creation of comma separated value files of the data stored in the DB. The third module deals with the development of python routines to execute the three level decision tree Classification on the symptoms data set and the final module deals with the four level decision tree analysis of symptoms data.

Architecture Diagram:

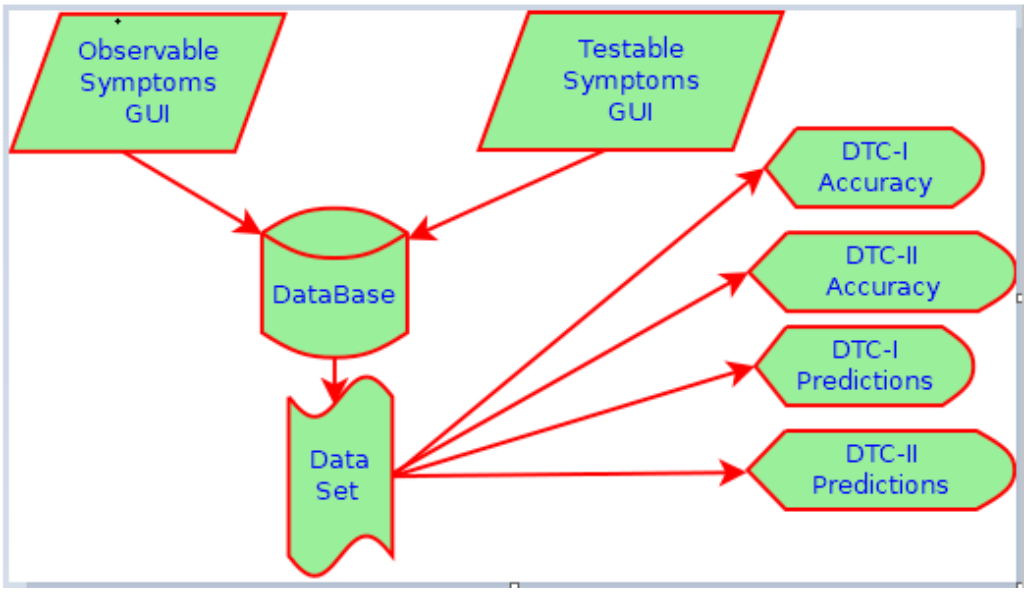


Fig 3.1 : Architecture Diagram

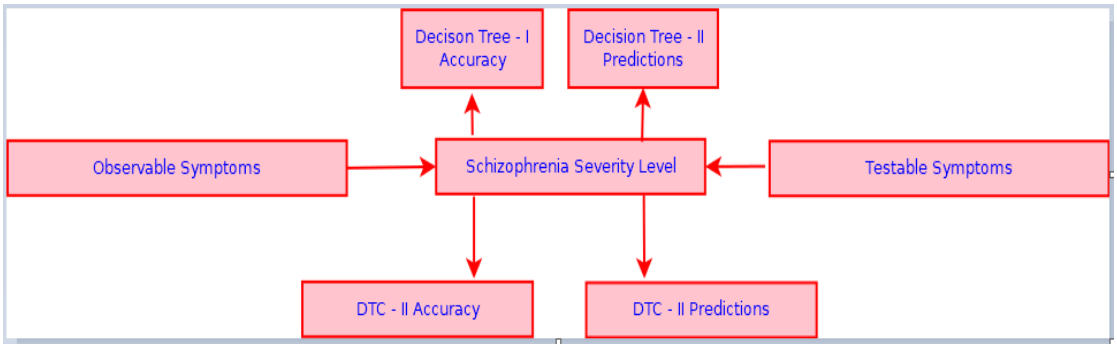


Fig 3.2 : Data Flow Diagram

The observable symptoms as well as the testable symptoms are to be provided as Input to the system, using the corresponding user interfaces. The system then generates the accuracies and predictions of the decision tree classifier with varying levels of depth.

Screen shots and descriptions:

Fig 3.3 : Screenshot showing the files created during the project

The above screen shot shows different files created during this project. There are four different types of files: (1) .py files (2) .ui file (3).txt file and (4).csv file

.ui files are the user interface files, created by using PyQt layout editor

.py files are python program files created either manually, or automatically. For instance, each .ui file has a corresponding .py file that is created automatically by using the PyUIC tool. .txt files contain the generic useful information about the project.

Schizo1.py, is the entry program for this project. Execution of this python program leads to the entry screen as follows:

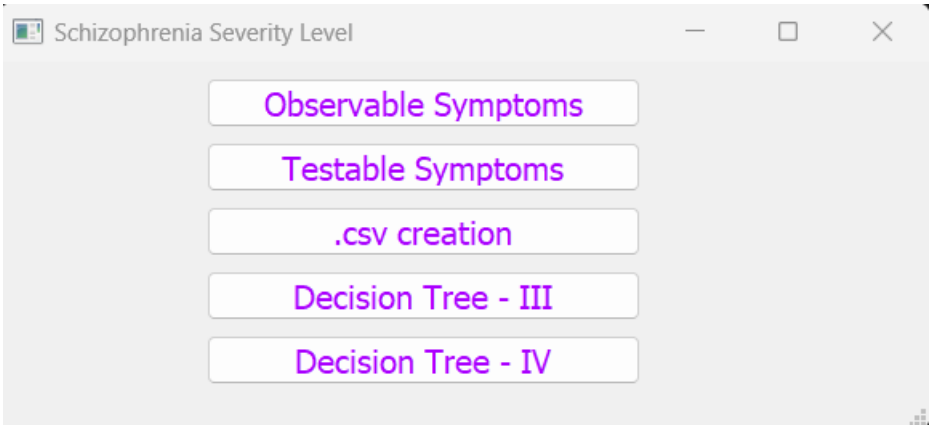


Fig 3.4: User Interface Diagram

The first two buttons are used to store the symptom sets in the DB, the third button is used to create the .csv file, by exporting the data from the DB, fourth button is used to find out the accuracy of Extra Tree Classification, and fifth button is used for the accuracy of the decision Tree Classifier.

3.2 Tools Used

Qt Designer:

Qt Designer is the tool used to create the Graphical user interfaces in this project. The tool can be found in the Library/Bin folder of Anaconda tool as shown in the following screenshot.

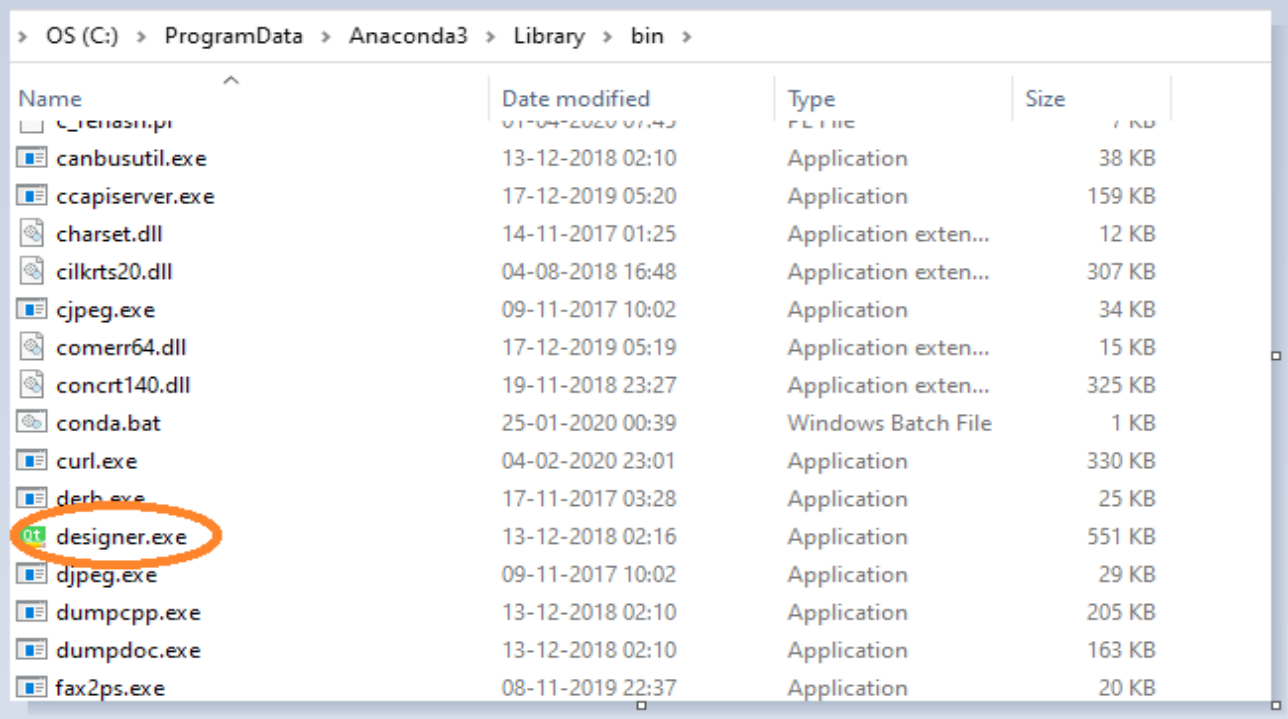


Fig 3.5 : Library/Bin folder of Anaconda tool

As this designer tool is used again and again in the project, it's better to create a shortcut on the desktop,by using a right-click on designer.exe, as shown in the following figure.

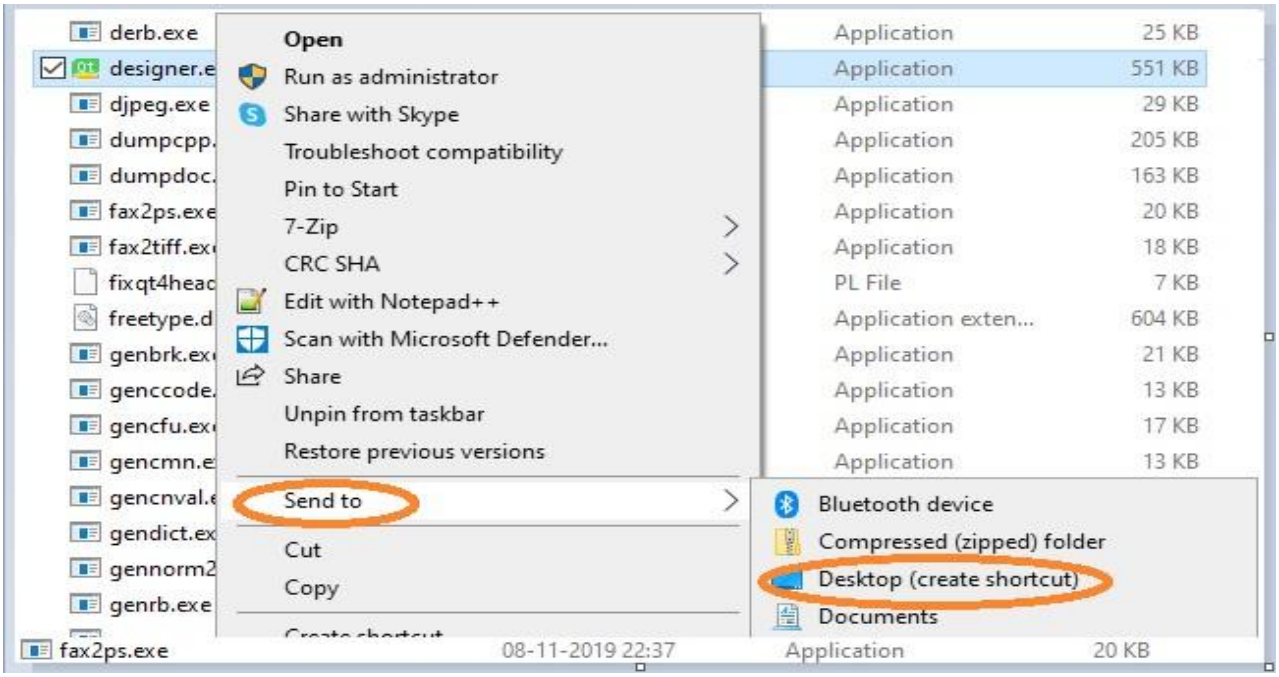


Fig 3.6 :Creating shortcut of designer.exe

The main window can be created by clicking on the Create button, after highlighting the mainwindow.

The main window can be given a title, by using the properties window,that is in the right handside middle portion**.**

The menu bar in the main window can be removed by right clicking in the white portion after 'Type here's label.

A push button can be created by dragging and dropping it, from the buttons section, on the left hand side as shown in below figure.

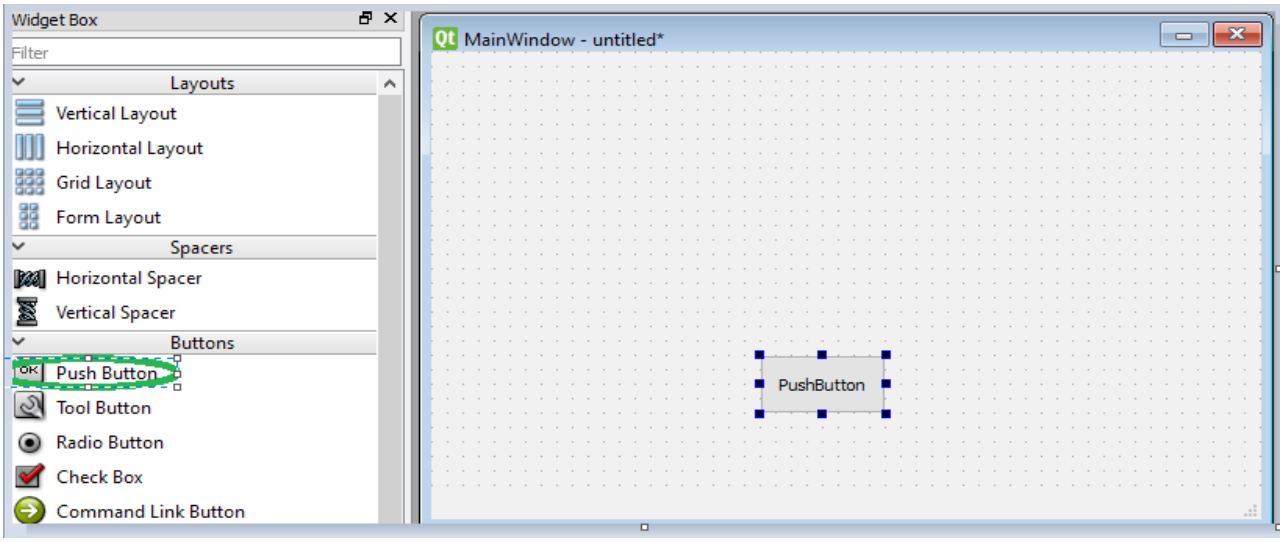


Fig 3.7 : Creating a push button

Likewise, A Label can be created by dragging and dropping it, from the Display Widgets section,on the left hand side .

Likewise, A Line edit can be created by dragging and dropping it, from the Input Widgets section, on the left hand side.

The color of a pushbutton/label can be changed by right clicking on it and selecting the 'change style' option.

Clicking on the inverted triangle symbol, on the right of the ‘Add color’ option .

The user can choose any of his/her interesting colors from the color palette, and click the ‘ok’ button.

Now the user has to click on the ‘Apply button’, in order to change the color of the text label, as shown in the following figure.

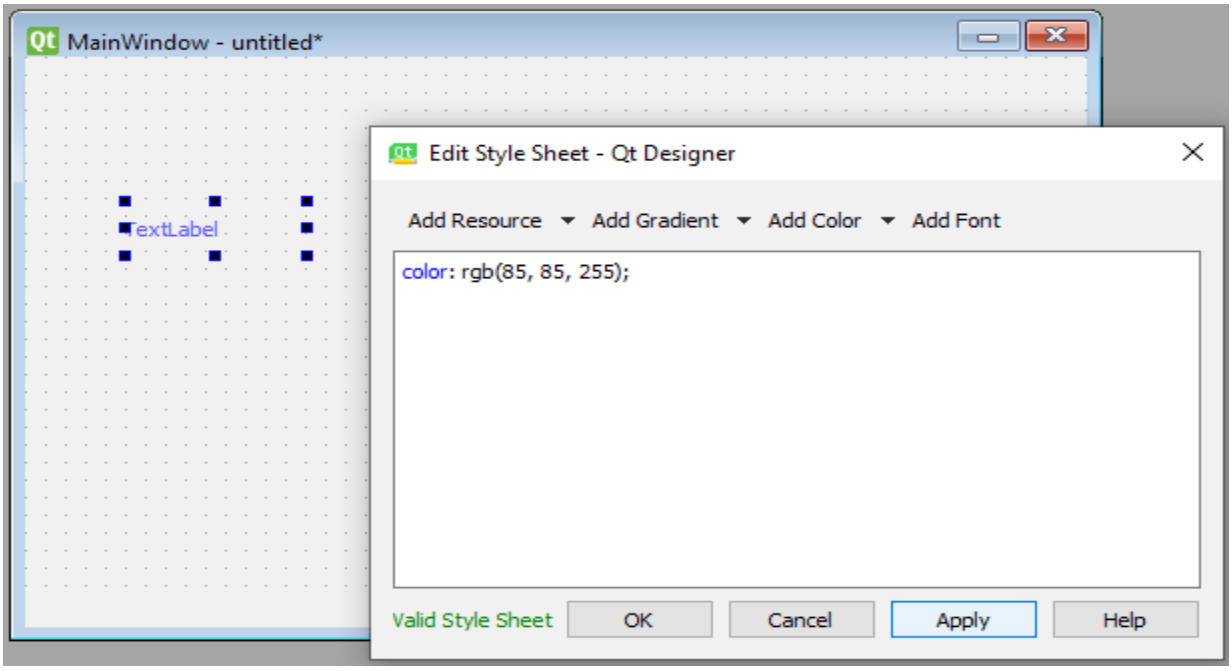


Fig 3.8 : Applying color to the text label

The user can change the font of the label/pushbutton, by clicking on the Addfont option.

The user can choose his interested font, font style and size and click on the ‘ok’ button .

The user needs to click on ‘Apply’ button to change the font .

3.3 DESIGN METHODOLOGY

3.3.1. UML DIAGRAMS

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order tobetter understand, alter, maintain, or document information about the system. UML has been used as ageneral-purpose modeling language in the field of software engineering. However, it has now found itsway into the documentation of several business processes or workflows. For example, activity diagrams,a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modelling workflows as well as a wider range of features to improve readability and efficiency.

3.3.2. USE CASE DIAGRAM

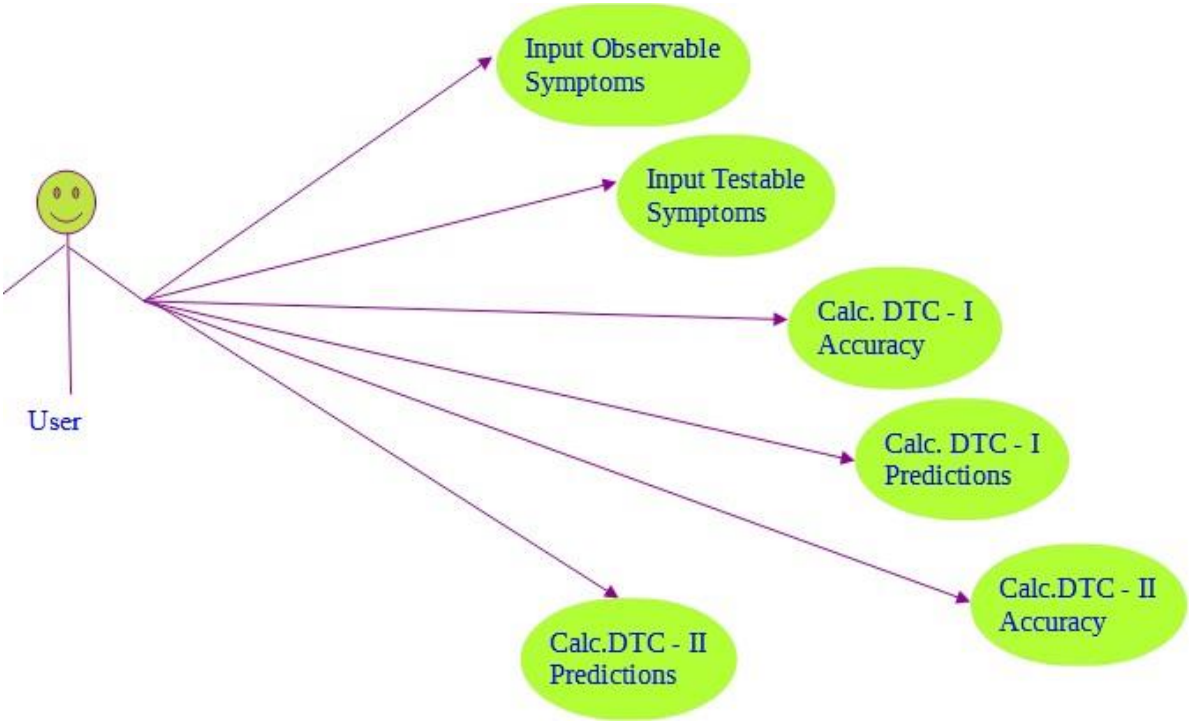


Fig : 3.9 Use Case Diagram

Description of Use Case Diagram

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

As we can see the user is interacting with the system by providing symptoms as input, and then calculating the accuracies and predictions of Decision Tree Analysis with varying levels of depth.

3.3.3. SEQUENCE DIAGRAM

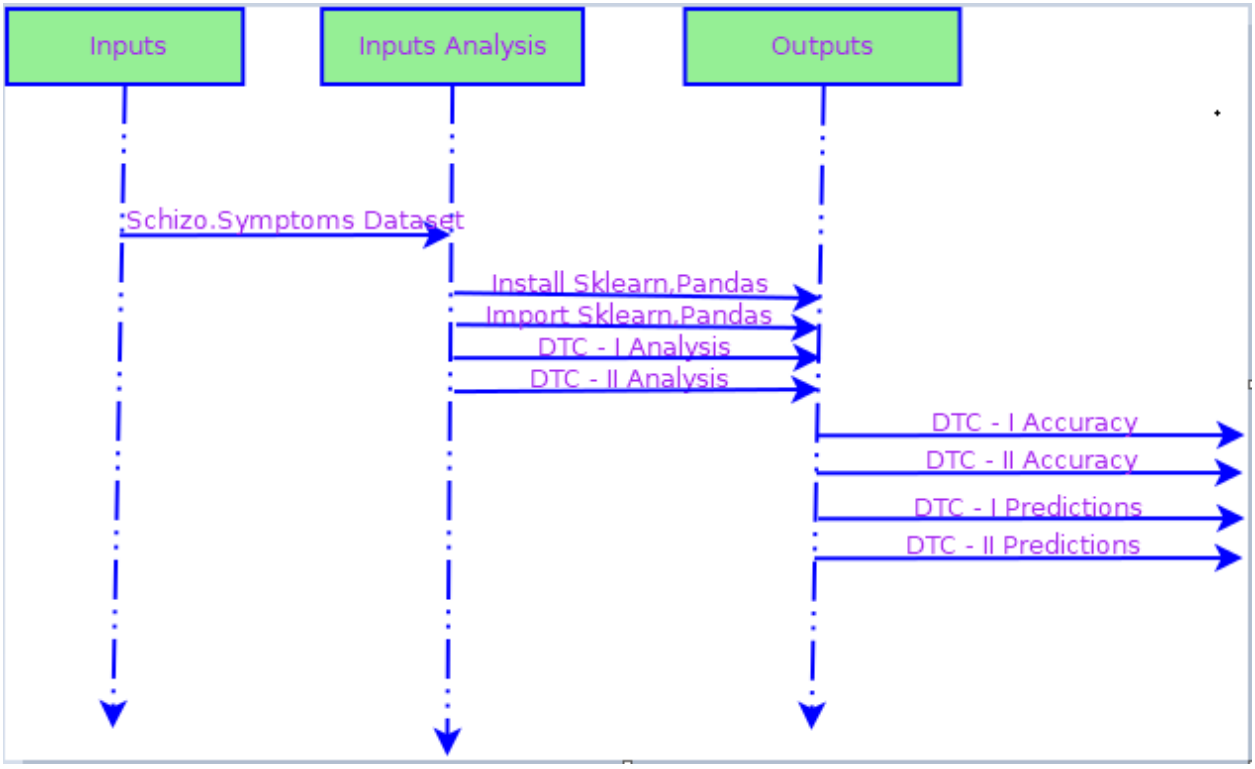


Fig 3.10 : Sequence Diagram

Description of sequence diagram

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence.

From the above mentioned sequence diagram we have to go in sequence: Enter the needed details as shown in the above figure, Provide the symptoms dataset and then generate the outputsviz.., the accuracies along with the predictions of DTC with varying levels of depth.

3.3.4. ACTIVITY DIAGRAM

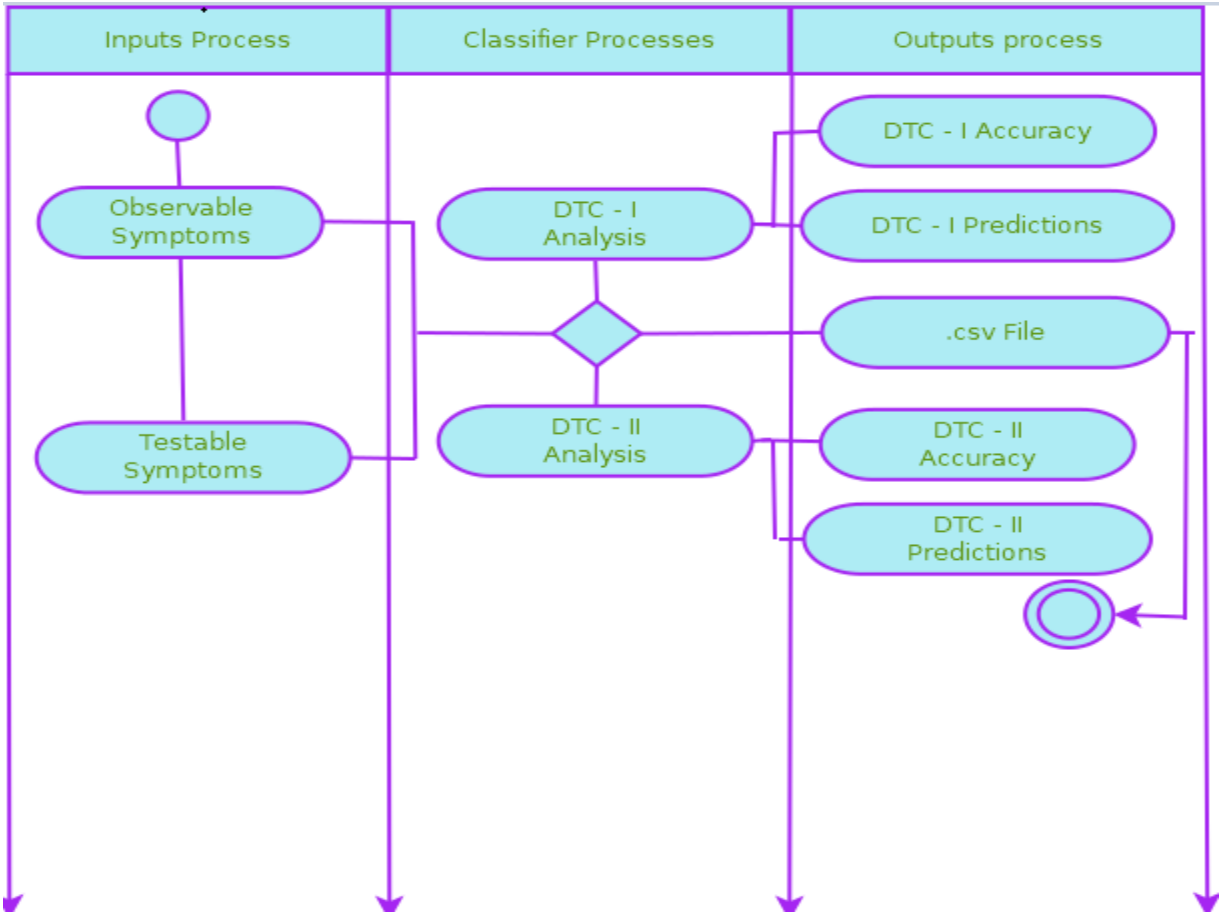


Fig 3.11: Activity Diagram

Description of Activity diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity toanother activity. The activity can be described as an operation of the system. So, the control flowis drawn from one operation to another. In the activity diagram we can see that first provide the Schizophrenia symptoms data set, Train the models, Test the models and calculate Decision Tree analysis accuracies and predictions with varying levels of prediction.

3.4 Modules Imported in Python routines:

OS Module:

The OS module in Python provides a way of using operating system dependent functionality.

The functions that the OS module provides allows you to interface with the underlying operating system that Python is running on – be that Windows, Mac or Linux.

OS functions:

Executing a shell command

os.system()

Get the users environment

os.environ()

Returns the current working directory.os.getcwd()

Return the real group id of the current process.

os.getgid()

Return the current process's user id.

os.getuid()

Returns the real process ID of the current process.

os.getpid()

Set the current numeric umask and return the previous umask.

os.umask(mask)

Return information identifying the current operating system.

os.uname()

Change the root directory of the current process to path.

os.chroot(path)

Return a list of the entries in the directory given by path.
os.listdir(path)

Create a directory named path with numeric mode mode.
os.mkdir(path)

Recursive directory creation function.
os.makedirs(path)

Remove (delete) the file path.

os.remove(path)

Remove directories recursively.
os.removedirs(path)

Rename the file or directory src to dst.
os.rename(src, dst)

Remove (delete) the directory path.
os.rmdir(path)

3.5. Model Evaluation and Selection:

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future

- There are several evaluation metrics, like confusion matrix, cross-validation, AUC-ROC curve, etc.
- Different evaluation metrics are used for different kinds of problems.

1. Confusion Matrix :

Also known as an Error Matrix, the Confusion Matrix is a two-dimensional matrix that allows visualization of the algorithm’s performance. While this isn't an actual metric to use for evaluation,

it's an important starting point. Predictions are highlighted and divided by class (true/false), before being compared with the actual values. The matrix's size is compatible with the amount of classes in the label column. In a binary classification, the matrix will be2X2. If there are 3 classes, the matrix will be 3X3, and so on.

This matrix essentially helps you determine if the classification model is optimized. It shows what errors are being made and helps to determine their exact type.

		predicted class		
Abstract Class			T	F
			Class Yes	Class No
	T	Class Yes	TT	FN
	F	Class No	FP	TN

TP- TRUE POSITIVE- Predicted values correctly predicted as positive

Ex: buyscomp=yes predicted as buyscomp=yes

TN- TRUE NEGATIVE- Predicted values are correctly predicted as Negative

Ex: buyscomp=no predicted as buyscomp=no

FN- FALSE NEGATIVE- Positive values are predicted as Negative

Ex: buyscomp=yes predicted as buyscomp=No

FP- FALSE POSITIVE- Negative values are predicted as Positive

Ex: buyscomp=no predicted as buyscomp=yes

2. **Accuracy :** Percentage of prediction that were correct.

Accuracy = TP + TN / TP + TN + FP + FN

3. **Sensitivity (true positive rate):** The true positive rate, also known as sensitivity, corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points.

Sensitivity = TP / FN + TP

4. **Specificity (false positive rate):** False positive rate, also known as specificity, corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points.

Specificity = FP / FP + TN

Please note that both FPR and TPR have values in the range of 0 to 1.

5. **Precision:** This metric is the number of correct positive results divided by the number of positive results predicted by the classifier.

Precision = TP / TP + FP

6. **Recall:** Recall is the number of correct positive results divided by the number of all samples that should have been identified as positive.

Recall = TP / TP + FN

Measure	Formula
Accuracy	(TP + TN)/ (TP + TN + FP + FN)
Error rate	(FP + FN)/(TP + TN + FP + FN)
Sensitivity, true positive rate, Recall	TP / FN + TP
Specificity, true negative rate	FP / FP + TN
Precision	TP / TP + FP
F1 SCORE	$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$ NOTHING BUT $2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$

7. **F1 score:** The F1 score is basically the harmonic mean between precision and recall. It issued to measure the accuracy of tests and is a direct indication of the model's performance. The range of the F1 score is between 0 to 1, with the goal being to get as close as possible to 1. It is calculated as per:

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

4. RESULTS

4.1. System Testing Results

Initially, an extensive evaluation was conducted, assessing the accuracy, precision, recall, F1 score, and computational efficiency across various classification algorithms for our designated model. Subsequently, the decision tree classifier emerged as the optimal choice, demonstrating superior accuracy, efficient processing, and ease of implementation.

Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
LinearSVC	0.99	0.99	0.99	0.99	0.06
LogisticRegression	0.99	0.99	0.99	0.99	0.06
SVC	0.99	0.99	0.99	0.99	0.23
CalibratedClassifierCV	0.99	0.99	0.99	0.99	0.15
SGDClassifier	0.99	0.99	0.99	0.99	0.05
PassiveAggressiveClassifier	0.99	0.99	0.99	0.99	0.04
LGBMClassifier	0.98	0.98	0.98	0.98	0.20
AdaBoostClassifier	0.98	0.98	0.98	0.98	0.35
XGBClassifier	0.98	0.98	0.98	0.98	1.00
LabelSpreading	0.98	0.98	0.98	0.98	0.34
ExtraTreesClassifier	0.98	0.98	0.98	0.98	0.63
LabelPropagation	0.98	0.98	0.98	0.98	0.29
ExtraTreeClassifier	0.98	0.98	0.98	0.98	0.05
RandomForestClassifier	0.98	0.98	0.98	0.98	0.60
DecisionTreeClassifier	0.98	0.98	0.98	0.98	0.06
BaggingClassifier	0.97	0.97	0.97	0.97	0.15
Perceptron	0.96	0.96	0.96	0.96	0.03
KNeighborsClassifier	0.92	0.92	0.92	0.92	0.09

Fig 4.1 : Metrics of Different Classification Algorithms.

After testing both decision tree classifier three level and four level algorithms we getaccuracy of 87.08% in three level and 89.53 in four level algorithm.

```
The accuracy of DT Classifier with three levels on testing data is: 87.08240534521158
The accuracy of DT Classifier with four levels on testing data is: 89.53229398663697
Precision : 0.8715103160812177
Sensitivity : 0.8708240534521158
Recall / Specificity : 0.8708240534521158
f1 score : 0.8708907474155426
Confusion matrix :
[[202  33]
 [ 25 189]]
```

Fig 4.2 : Metrics results for the dataset

```
Improper eye contact: 1  Proper eye contact:0 = 1
NoIntimeResponse:1 IntimeResponse:0 = 1
ImaginativePlay:1 NoImaginativePlay:0 = 1
SpendsAlone:1 SpendswithOthers:0 = 1
Epilepsy:1 NoEpilepsy:0 = 0
Depression:1 NoDepression:0 = 0
PoorLogicalReasoning:1 LogicalReasoningOK:0 = 1
UnabletoSpeakSentns:1 AbletoSpeakSentns:0 = 1
UnabletoSpeakWords:1 AbletoSpeakWords:0 = 0
PoorinMaths:1 NotpoorinMaths:0 = 1
DT prediction on the given test set is: [1]
```

Fig 4.3 : Prediction of the Schizophrenia

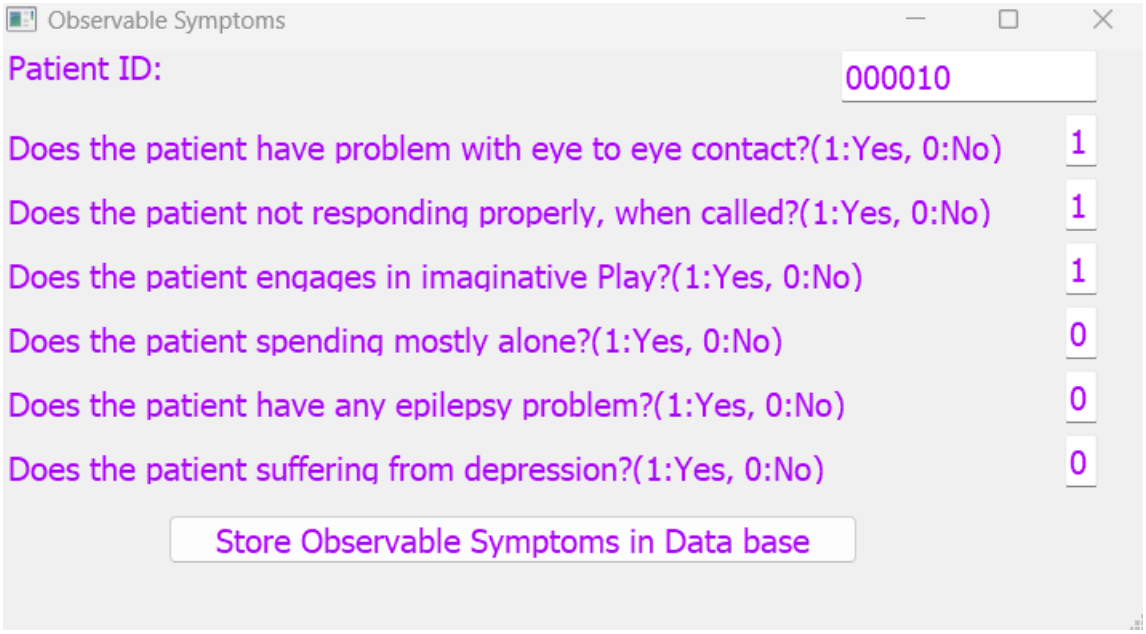


Fig 4.4 : Storing the observable symptoms using UI

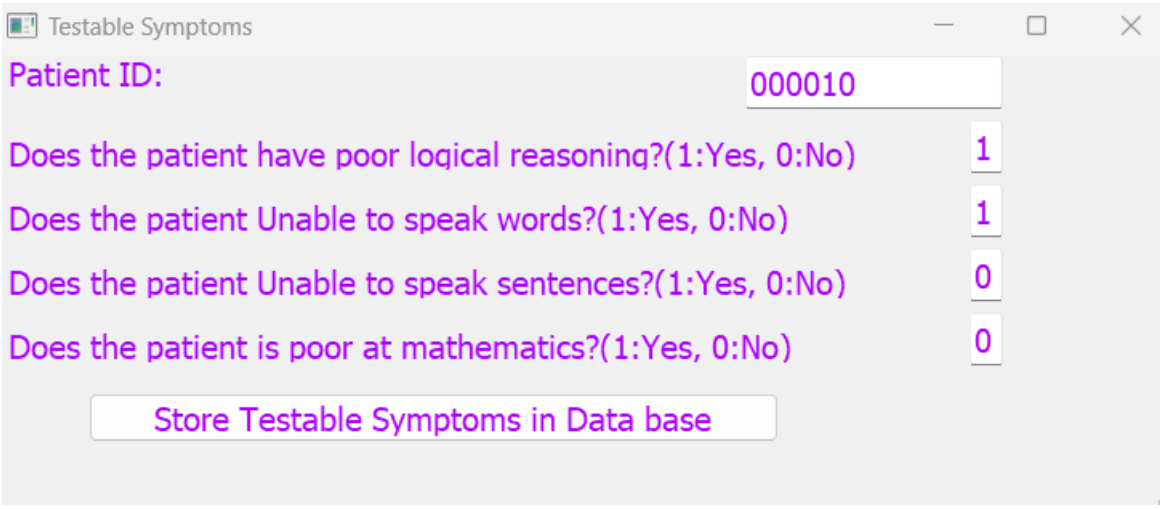


Fig 4.5 : Storing the testable symptoms using UI

```
C:\mini>python schiz1.py
obsymps1.csv file successfully created
testsymps1.csv file successfully created
The accuracy of DT Classifier with three levels on testing data is: 86.7892016082711
DT prediction on the first test set is: [1]
DT prediction on the second test set is: [1]
The accuracy of DT Classifier with four levels on testing data is: 87.65077541642734
DT prediction on the first test set is: [1]
DT prediction on the second test set is: [0]
```

Fig 4.6 : Accuracy of the Decision Tree with three and four levels

5. CONCLUSION AND FUTURE SCOPE

5.1. CONCLUSION

This project is entitled **"Schizophrenia Severity Level Identification."** holds significant promise in addressing the challenges faced by economically disadvantaged patients who are unable to undergo regular medical checkups due to high healthcare expenses. By leveraging technology to automate the identification of schizophrenia severity levels, the project serves as a valuable tool for individuals who may otherwise struggle to access timely and comprehensive healthcare.

For doctors, the project provides a streamlined and efficient means of diagnosis, eliminating the need to memorize extensive details about each symptom associated with schizophrenia. The automated identification of severity levels empowers healthcare professionals to make quicker, more informed decisions, ultimately leading to more effective treatment strategies for patients.

Beyond the immediate benefits for patients and healthcare providers, the project contributes to the broader goal of enhancing the overall quality of life for individuals affected by schizophrenia. By providing a systematic and accessible method for assessing severity levels, the project facilitates early intervention and personalized treatment plans, contributing to the improvement of long-term health outcomes.

As we continue to advance in the field of healthcare technology, projects like these play a crucial role in democratizing access to quality care and promoting the well-being of vulnerable populations. The "Schizophrenia Severity Level Identification" project stands as a testament to the positive impact that innovative technology can have on the lives of individuals facing mental health challenges, ultimately contributing to the extension and enhancement of the average human lifespan for those diagnosed with schizophrenia.

5.2. FUTURE SCOPE

As of now, the project is implemented by calculating the accuracies of decision tree analysis with varying levels of depth. The project can be further extended by considering other classifiers like extra tree classifiers.

BIBLIOGRAPHY

[1] XINYAN YUAN, LINGLING GU AND JIA SHUANG HUANG: GK-BSC: Graph Kernel-Based Brain States Construction With Dynamic Brain Networks and Application toSchizophrenia Identification, IEEE Access

[2] P. Stepnicki, M. Kondej, and A. A. Kaczor, "Current concepts and treatments of schizophrenia," Molecules, vol. 23, no. 8, p. 2087.

[3] T. A. Greenwood, A. Shutes-David, and D. W. Tsuang, "Endophenotypes in schizophrenia: Digging deeper to identify genetic mechanisms", J. Psychiatry Brain Sci., vol. 4, no. 2, 2019,
Art. no. e190005

[4] "Extending schizophrenia diagnostic model to predit in first-degree relatives" by Sunil Vasu Kalmady, Animesh Kumar Paul, Russell Greiner, Rimjhim Agrawal, Serdar M. Dursun & Ganesan Venkatasubram.

[5] "Early detection of schizophrenia: current evidence and future perspectives" by HEINZ HÄFNER and KURT MAURER.

[6] Jobe TH, Harrow M. Long-term outcome of patients with schizophrenia: a review. Can J Psychiatry.

[7] McGue, M., Gottesman, I. I. & Rao, D. C. The transmission of schizophrenia under a multifactorial threshold model.

[8] Grant, P., Green, M. J. & Mason, O. J. Models of schizotypy: the importance of conceptual clarity.

[9] Soler, J. et al. Familial aggregation of schizotypy in schizophrenia-spectrum disorders and its relation to clinical and neurodevelopmental characteristics.

[10] S. Maher, T. Ekstrom, D. Ongur, D. L. Levy, D. J. Norton, L. D. Nickerson,

and Y. Chen, ‘‘Functional disconnection between the visual cortex and right fusiform face area in schizophrenia,’’ Schizophrenia Res., vol. 209, pp. 72–79, Jul. 2019.

[11] Barrantes-Vidal, N., Grant, P. & Kwapil, T. R. The role of schizotypy in the study of the etiology of schizophrenia spectrum disorders.

[12] De Berardis D., Rapini G., Olivieri L., Di Nicola D., Tomasetti C., Valchera A., Fornaro M., Di Fabio F., Perna G., Di Nicola M., et al. Safety of antipsychotics for the treatment of schizophrenia: A focus on the adverse effects of clozapine.

[13] Corcoran C.M., Carrillo F., Fernandez-Slezak D., Bedi G., Klim C., Javitt D.C., Bearden C.E., Cecchi G.A. Prediction of psychosis across protocols and risk cohorts using automated language analysis.

[14] Schizophrenia Detection Using Machine Learning Approach from Social Media Content Yi Ji Bae, Midan Shim and Won Hee Lee.

[15] https://www.python.org/

[16] https://github.com/baoboa/pyqt5/blob/master/pyuic/uic/pyuic.py

[17] https://www.numpy.org/

[18] https://riverbankcomputing.com/software/pyqt/intro

APPENDIX

dtc3.py

```
import numpy as np

import pandas as pd

from sklearn import *

from sklearn.tree import DecisionTreeClassifier

from lazypredict.Supervised import LazyClassifier

from sklearn.model_selection import train_test_split

from matplotlib import pyplot as plt

from sklearn import tree

from matplotlib import pyplot as plt

import seaborn as sns

import scipy

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score,

f1_score,precision_score, classification_report,

confusion_matrix, recall_score

from sklearn.metrics import roc_auc_score

from sklearn.metrics import roc_curve
```

```
from sklearn.decomposition import PCA

from sklearn.preprocessing import MinMaxScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.naive_bayes import GaussianNB

from statistics import mean

from sklearn.model_selection import StratifiedKFold

from sklearn.tree import plot_tree

df = pd.read_csv('C:/mini/schizoset.csv')

df["Obs_Sympt1"]=df["Obs_Sympt1"].map

({'ImproperEyeContact':1 ,'ProperEyeContact':0})

df["Obs_Sympt2"]=df["Obs_Sympt2"].map

({'NoIntimeResponse':1 ,'IntimeResponse':0})

df["Obs_Sympt3"]=df["Obs_Sympt3"].map

({'ImaginativePlay':1 ,'NoImaginativePlay':0})

df["Obs_Sympt4"] =df["Obs_Sympt4"].map

({'SpendsAlone':1 ,'SpendswithOthers':0})

df["Obs_Sympt5"] = df["Obs_Sympt5"].map

({'Epilepsy':1 ,'NoEpilepsy':0})

df["Obs_Sympt6"] = df["Obs_Sympt6"].map

({'Depression':1 ,'NoDepression':0})

df["Test_Sympt1"]=df["Test_Sympt1"].map
```

```
({'PoorLogicalReasoning':1 ,'LogicalReasoningOK':0})

df["Test_Sympt2"]=df["Test_Sympt2"].map

({'UnabletoSpeakSentns':1 ,'AbletoSpeakSentns':0})

df["Test_Sympt3"]=df["Test_Sympt3"].map

({'UnabletoSpeakWords':1 ,'AbletoSpeakWords':0})

df["Test_Sympt4"] =df["Test_Sympt4"].map

({'PoorinMaths':1 ,'NotpoorinMaths':0})

df["SchizoSeverity"] = df["SchizoSeverity"].map

({'High':1 ,'Low':0})

data=df[["Obs_Sympt1","Obs_Sympt2","Obs_Sympt3",

"Obs_Sympt4","Obs_Sympt5","Obs_Sympt6","Test_Sympt1",

"Test_Sympt2","Test_Sympt3","Test_Sympt4",

"SchizoSeverity"]].to_numpy()

inputs = data[:,:-1]

outputs = data[:, -1]

training_inputs,testing_inputs,training_outputs,

testing_outputs=train_test_split(inputs,outputs,test_size=

0.2,random_state=12)

clf=LazyClassifier(predictions=True)

models,predictions=clf.fit(training_inputs,testing_inputs,

training_outputs,testing_outputs)
```

```
models

classifier = DecisionTreeClassifier(max_depth=3)

classifier.fit(training_inputs, training_outputs)

predictions = classifier.predict(testing_inputs)

accuracy = 100.0 * accuracy_score(testing_outputs,

predictions)

print ("The accuracy of DT Classifier with three levels on

testing data is: " + str(accuracy))

classifier1 = DecisionTreeClassifier(max_depth=4)

classifier1.fit(training_inputs, training_outputs)

predictions1 = classifier1.predict(testing_inputs)

accuracy1 = 100.0 * accuracy_score(testing_outputs,

predictions1)

print ("The accuracy of DT Classifier with four levels on

testing data is: " + str(accuracy1))

print(f"Precision : {precision_score(testing_outputs,

predictions, average = 'weighted')}")

# ability of the classifier not to label as negative a sample that

is positive

print(f"Sensitivity : {recall_score(testing_outputs,

predictions,average = 'weighted')}")

# ability of the classifier to capture all the positive cases
```

```
print(f"Recall / Specificity : {recall_score(testing_outputs,
predictions, average = 'weighted')}")
# ability of the classifier to capture all the negative cases
print(f"f1 score : {f1_score(testing_outputs, predictions,
average = 'weighted')}")
cf = confusion_matrix(testing_outputs, predictions)
print(f'Confusion matrix :')
print(cf)
classifier = DecisionTreeClassifier(max_depth=3)
classifier.fit(training_inputs, training_outputs)
predictions = classifier.predict(testing_inputs)
o1=int(input('Impropereyecontact: 1  Propereyecontact:0 = '))
o2=int(input('NoIntimeResponse:1 IntimeResponse:0 = '))
o3=int(input('ImaginativePlay:1 NoImaginativePlay:0 = '))
o4=int(input('SpendsAlone:1 SpendswithOthers:0 = '))
o5=int(input('Epilepsy:1 NoEpilepsy:0 = '))
o6=int(input('Depression:1 NoDepression:0 = '))
t1=int(input('PoorLogicalReasoning:1
LogicalReasoningOK:0 = '))
t2=int(input('UnabletoSpeakSentns:1
AbletoSpeakSentns:0 = '))
```

```
t3=int(input('UnabletoSpeakWords:1

AbletoSpeakWords:0 = '))

t4=int(input('PoorinMaths:1 NotpoorinMaths:0 = '))

r=[[]]

r[0].append(o1)

r[0].append(o2)

r[0].append(o3)

r[0].append(o4)

r[0].append(o5)

r[0].append(o6)

r[0].append(t1)

r[0].append(t2)

r[0].append(t3)

r[0].append(t4)

predictions = classifier.predict(r)

print('DT prediction on the given test set is:',predictions)
```

createcsv1h.py

```
import sys

import csv

import os
```

```
import sqlite3

con = sqlite3.connect('schizo1')

cur = con.cursor()

cur.execute('SELECT * FROM obsymps;')

names = [description[0] for description in cur.description]

rows = cur.fetchall()

fp = open('obsymps1.csv', 'w')

myFile = csv.writer(fp)

myFile.writerows([names])

myFile.writerows(rows)

fp.close()

print('obsymps1.csv file successfully created')

cur.execute('SELECT * FROM testsymps;')

names = [description[0] for description in cur.description]

rows = cur.fetchall()

fp = open('testsymps1.csv', 'w')

myFile = csv.writer(fp)

myFile.writerows([names])

myFile.writerows(rows)

fp.close()
```

```
print('testsymps1.csv file successfully created')
con.commit()
```

schiz.py

```
from PyQt5 import QtCore, QtGui, QtWidgets
class Ui_MainWindow(object):
  def setupUi(self, MainWindow):
   MainWindow.setObjectName("MainWindow")
   MainWindow.resize(582, 229)
   MainWindow.setStyleSheet("color: rgb(170, 0, 255);\n"
"font: 75 12pt \"MS Shell Dlg 2\";")
   self.centralwidget = QtWidgets.QWidget(MainWindow)
   self.centralwidget.setObjectName("centralwidget")
   self.pushButton=QtWidgets.QPushButton(self.centralwidget)
   self.pushButton.setGeometry(QtCore.QRect
  (130, 50, 271, 31))
  self.pushButton.setObjectName("pushButton")
 self.pushButton_2 = QtWidgets.QPushButton(self.centralwidget)
 self.pushButton_2.setGeometry(QtCore.QRect
 (130, 90, 271, 31))
self.pushButton_2.setObjectName("pushButton_2")
self.pushButton_3 = QtWidgets.QPushButton(self.centralwidget)
```

```
self.pushButton_3.setGeometry(QtCore.QRect(130, 170, 271, 31))
self.pushButton_3.setObjectName("pushButton_3")
self.pushButton_4 = QtWidgets.QPushButton(self.centralwidget)
self.pushButton_4.setGeometry(QtCore.QRect(130, 130, 271, 31))
self.pushButton_4.setObjectName("pushButton_4")
self.pushButton_5 = QtWidgets.QPushButton(self.centralwidget)
self.pushButton_5.setGeometry(QtCore.QRect(130, 10, 271, 31))
self.pushButton_5.setObjectName("pushButton_5")
MainWindow.setCentralWidget(self.centralwidget)
self.statusbar = QtWidgets.QStatusBar(MainWindow)
self.statusbar.setObjectName("statusbar")
MainWindow.setStatusBar(self.statusbar)
self.retranslateUi(MainWindow)
QtCore.QMetaObject.connectSlotsByName(MainWindow)
def retranslateUi(self, MainWindow):
_translate = QtCore.QCoreApplication.translate
MainWindow.setWindowTitle(_translate("MainWindow",
"Schizophrenia Severity Level"))
self.pushButton.setText(_translate("MainWindow", "Testable
Symptoms"))
self.pushButton_2.setText(_translate("MainWindow", ".csv
creation"))
self.pushButton_3.setText(_translate("MainWindow","Decision
```

```
Tree - IV"))
self.pushButton_4.setText(_translate("MainWindow", "Decision
Tree - III"))
self.pushButton_5.setText(_translate("MainWindow", "Observable
Symptoms"))
```

obsymps1.py

```
import sys
import os
from obsymps import *
from PyQt5 import QtWidgets, QtGui, QtCore
import sqlite3
con = sqlite3.connect('schizo1')
class MyForm(QtWidgets.QMainWindow):
  def __init__(self,parent=None):
     QtWidgets.QWidget.__init__(self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.insertvalues)
     #self.ui.pushButton_2.clicked.connect(self.testdetails)
     def insertvalues(self):
    with con:
```

```
cur = con.cursor()

id = str(self.ui.lineEdit_9.text())

s1 = str(self.ui.lineEdit_3.text())

s2 = str(self.ui.lineEdit_4.text())

s3 = str(self.ui.lineEdit_5.text())

s4 = str(self.ui.lineEdit_6.text())

s5 = str(self.ui.lineEdit_2.text())

s6 = str(self.ui.lineEdit_7.text())

cur.execute('INSERT INTO obsymps(id,s1,s2,s3,s4,s5,s6)

VALUES(?,?,?,?,?,?,?)',(id,s1,s2,s3,s4,s5,s6))

con.commit()

#  def testdetails(self):

#    os.system("python ptests1.py")

if __name__ == "__main__":

app = QtWidgets.QApplication(sys.argv)

myapp = MyForm()

myapp.show()

sys.exit(app.exec_())
```

testsymps1.py

```
import sys

import os
```

```
from testsymps import *
from PyQt5 import QtWidgets, QtGui, QtCore
import sqlite3
con = sqlite3.connect('schizo1')
class MyForm(QtWidgets.QMainWindow):
def __init__(self,parent=None):
QtWidgets.QWidget.__init__(self,parent)
self.ui = Ui_MainWindow()
self.ui.setupUi(self)
self.ui.pushButton.clicked.connect(self.insertvalues)
#self.ui.pushButton_2.clicked.connect(self.testdetails)
def insertvalues(self):
with con:
cur = con.cursor()
id = str(self.ui.lineEdit_9.text())
s1 = str(self.ui.lineEdit_3.text())
s2 = str(self.ui.lineEdit_4.text())
s3 = str(self.ui.lineEdit_5.text())
s4 = str(self.ui.lineEdit_6.text())
cur.execute('INSERT INTO testsymps(id,s1,s2,s3,s4)
VALUES(?,?,?,?,?)',(id,s1,s2,s3,s4))
```

```
con.commit()

#  def testdetails(self):

#    os.system("python ptests1.py")

if __name__ == "__main__":

app = QtWidgets.QApplication(sys.argv)

myapp = MyForm()

myapp.show()

sys.exit(app.exec_())
```