Indian Institute of Information Technology, Allahabad
Software Engineering

Instructors: Dr. Sonali Agarwal

SOFTWARE DESIGN SPECIFICATION

GROUP MEMBERS-

Medha Balani	IIT2019021
Vidushi Pathak	IIT2019027
Aarushi	IIT2019032
Jyotika Bhatti	IIT2019036

Table of contents:

1.Introduction:

The Software Design Document is a document to provide documentation which will be used to aid in software development by providing the details for how the software should be built. Within the Software Design Document are narrative and graphical documentation of the software design for the project including use case models, sequence diagrams, collaboration models, object behaviour models, and other supporting requirement information.

1.1Purpose of this document

This document will define the design of our software that is to identify change in Hepatitis data patterns in India. It contains specific information about the expected input, output, classes, and functions. The interaction between the classes to meet the desired requirements are outlined in detailed figures at the end of the document

1.2 Scope of the development project

We describe what features are in the scope of the software and what are not in the scope of the software to be developed.

<u>In Scope</u>

a. Users can see the change in the data pattern of Hepatitis in India.
b. Users can see the correlation matrix between various attributes.
c. Users can see the mortality rate.
d. Users can visualize various attributes.
e. Users can get the probability for mortality and survival chances according to the patterns.

<u>Out of scope</u>

a. Users can not visualize the data of other countries.
b. Users can only visualize the listed attributes.
c. The data set being static, the results will be constrained towards the same.

1.3 Overview of document

This SDS is divided into seven sections with various sub-sections. The sections of the Software Design Document are:.

1.Introduction: describes the document, purpose, scope of development project definitions and abbreviations used in the document.

2. Conceptual Architecture/Architecture Diagram: describes the overview of components, modules, structure and relationships and user interface issues.

3. Logical Architecture: describes Logical Architecture Description and Components.

4. Execution Architecture: defines the runtime environment, processes, deployment view.

5. Design Decisions and Trade-offs: describes the decisions taken along with the reason as to why they were chosen over other alternatives.

6. Pseudocode for components: describes pseudocode, as the name indicates.

7. Appendices: describes subsidiary matter if any

2. Design considerations

2.1 Assumptions and dependencies
Software would be used in the presence of the internet otherwise, the app shall be downloaded within the computer system for use. No any specific of any operating system or any such thing required. This software can be accessed through any smartphone or computer system.

2.2 General Constraints
The constraint for this software is the presence of a static data set. All the results and visualisations and analysis will be completely based on the available data set. The probabilities shown and the correlation matrices shall be completely based on the data set.
Users shall not change any values in the data set, they can just get the analysis results and correlation matrix, and can visualise the changes. They can give any custom inputs to check the mortality rates and probabilities corresponding to that input.
Presence of internet connection will be preferred, otherwise if you wish to download and run the code, the presence of python and its libraries should be pre-installed.
No security or verification required, anyone can access and check the analysis results.

2.3 Goals and Guidelines
The main purpose of these softwares related to the health sector is to provide patterns of a pre-existing disease and assist for the same. So this software should be made with utmost simplicity and highly user friendly. So, rather than a complex software, we will use a simple interface and use easy to deal things like simple buttons and drop down menus, which would be very precise about providing the information.
By providing symptoms and patterns of the hepatitis disease on a software available to all, we don’t aim to scare or panic any patients. Relying on proper advice from a practitioner is recommended, while this analysis can help doctors, practitioners and hospitals.

2.4 Development Methods

This software will be developed purely using python libraries such as pandas, matplotlib, streamlit, numpy, tkinter, sweet viz etc.
Pandas can be used to perform data manipulation and analysis. Pandas provide powerful and easy-to-use data structures, as well as the means to quickly perform operations on these structures.
Numpy is another very useful python library, it also has functions for working in the domain of linear algebra, fourier transform, and matrices. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. Arrays are very frequently used in data science, where speed and resources are very important.
Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter.
Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science.
Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.
Sweetviz is an open-source Python library that generates beautiful, high-density visualizations to kickstart EDA (Exploratory Data Analysis) with just two lines of code. Output is a fully self-contained HTML application.

EDA(Exploratory data analysis) is done using **pandas and sweet viz** module
, which is a critical process of performing initial investigations on data so as to discover patterns,to spot anomalies,to test hypotheses and to check assumptions with the help of summary statistics and graphical representations. EDA is all about making sense of data in hand,before getting them dirty with it.

3. ARCHITECTURAL STRATEGIES

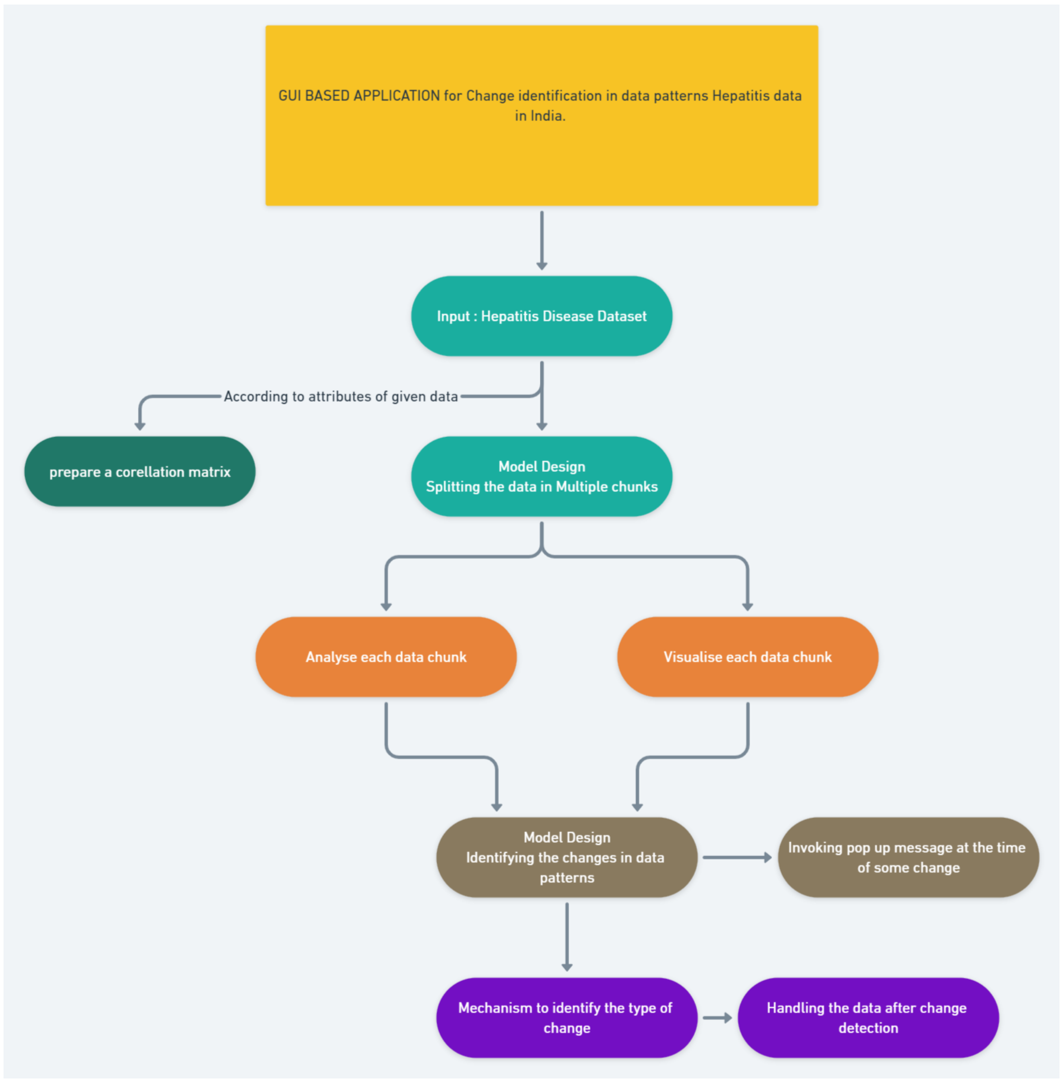
As stated above, the software is built to assist the health sector so architecture for the same would be kept as simple as possible. The architecture will include basic functionalities such as analysing the dataset using pandas and then plotting the corresponding graphs and correlation matrices for all the attributes in the data set used using matplotlib.

4. SYSTEM ARCHITECTURE

The front page shall provide the overview of what all things can be done Using the software, for example it will provide the basic outline, of how the Users can take advantage of the results and where he/she can find the results of the correlation matrices, visualisations of the data set.

The user can change, drop and select the attributes according to which he can get the results completely, however, he cannot change the values of any of the attributes.

The flow chart diagram is shown below, describes briefly about how the software shall function :

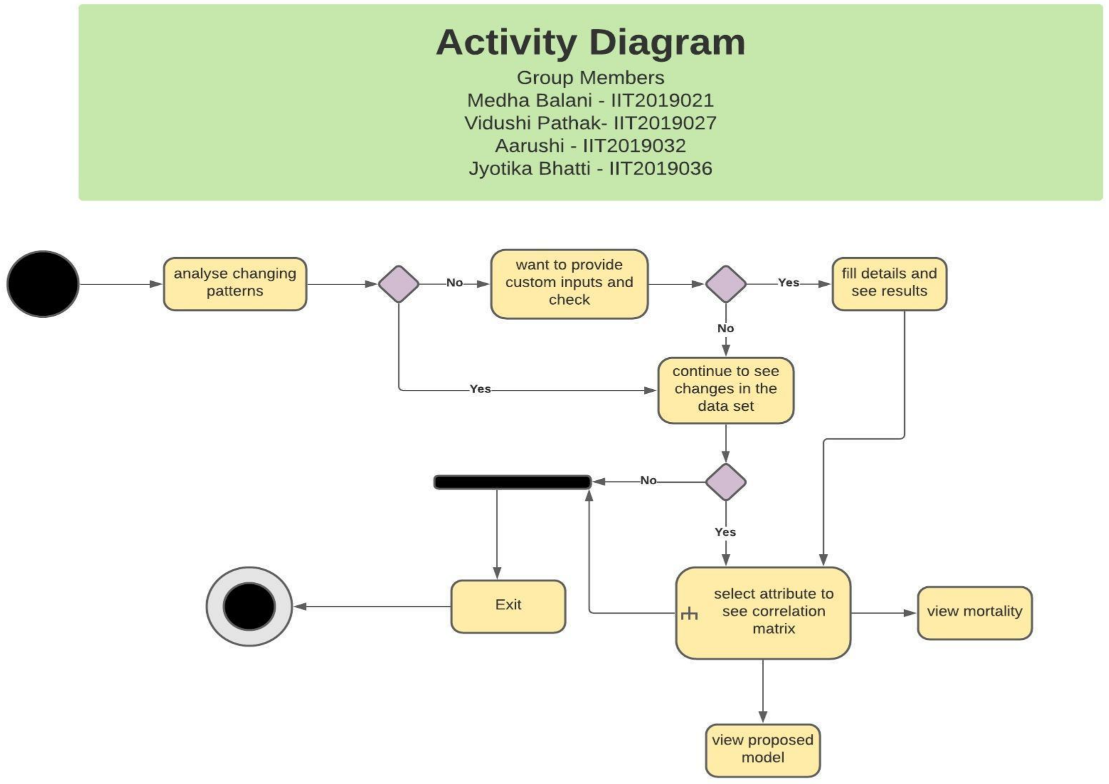

GUI BASED APPLICATION for Change identification in data patterns Hepatitis data in India.
Input : Hepatitis Disease Dataset
According to attributes of given data
prepare a corellation matrix
Model Design
Splitting the data in Multiple chunks
Analyse each data chunk
Visualise each data chunk
Model Design
Identifying the changes in data patterns
Invoking pop up message at the time of some change
Mechanism to identify the type of change
Handling the data after change detection

Following is the description of all the diagrams which explain our system architecture:

Activity Diagram:

Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use case relate to one another, in particular, use cases where activities may overlap and require coordination. It is also suitable for modeling how a collection of use cases coordinate to represent business workflows

Identifying candidate use cases, through the examination of business workflows, Identifying pre- and post-conditions (the context) for use cases, Model workflows between/within use cases, Model complex workflows in operations on objects, Model in detail complex activities in a high level activity Diagram


Activity Diagram
Group Members
Medha Balani - IIT2019021
Vidushi Pathak- IIT2019027
Aarushi - IIT2019032
Jyotika Bhatti - IIT2019036
analyse changing patterns
No
want to provide custom inputs and check
Yes
fill details and see results
No
Yes
continue to see changes in the data set
No
Yes
Exit
select attribute to see correlation matrix
view mortality
view proposed model

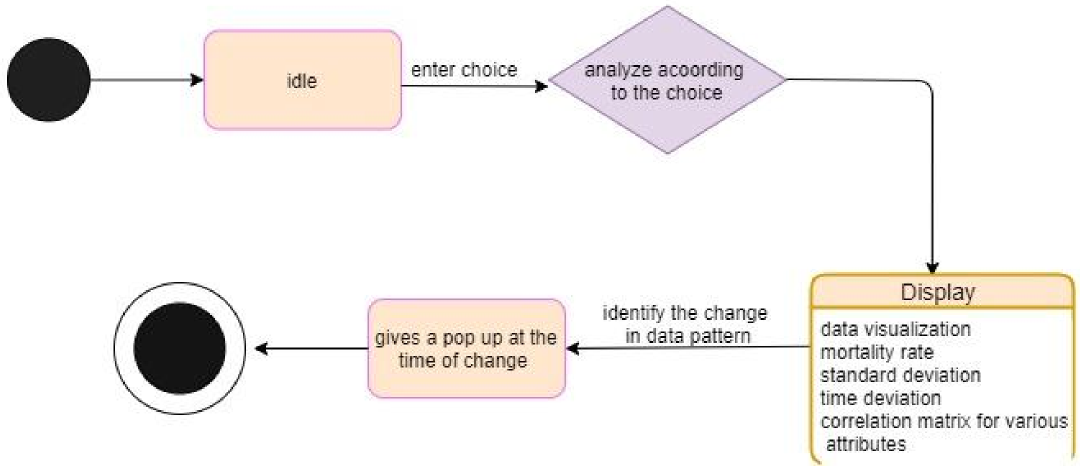
State Diagram:

A state diagram is used to represent the condition of the system or part of the system at finite instances of time. It’s a behavioral diagram and it represents the behavior using finite state transitions. State diagrams are also referred to as State machines and State-chart Diagrams. These terms are often used interchangeably. So simply, a state diagram is used to model the dynamic behavior of a class in response to time and changing external stimuli. We can say that each and every class has a state but we don’t model every class using State diagrams.

STATE DIAGRAM

Group Members

Medha Balani - IIT2019021
Vidushi Pathak - IIT2019027
Aarushi - IIT2019032
Jyotika Bhatti - IIT2019036


idle
enter choice
analyze acoording to the choice
Display
data visualization
mortality rate
standard deviation
time deviation
correlation matrix for various attributes
identify the change in data pattern
gives a pop up at the time of change

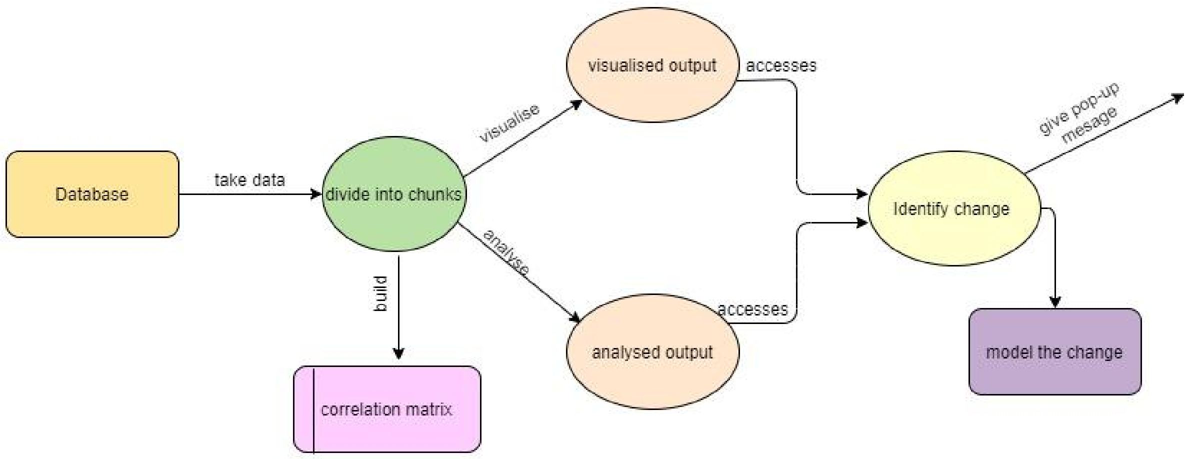
DFD :

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. The diagram completely describes how the work shall flow in the software.

Data Flow Diagram

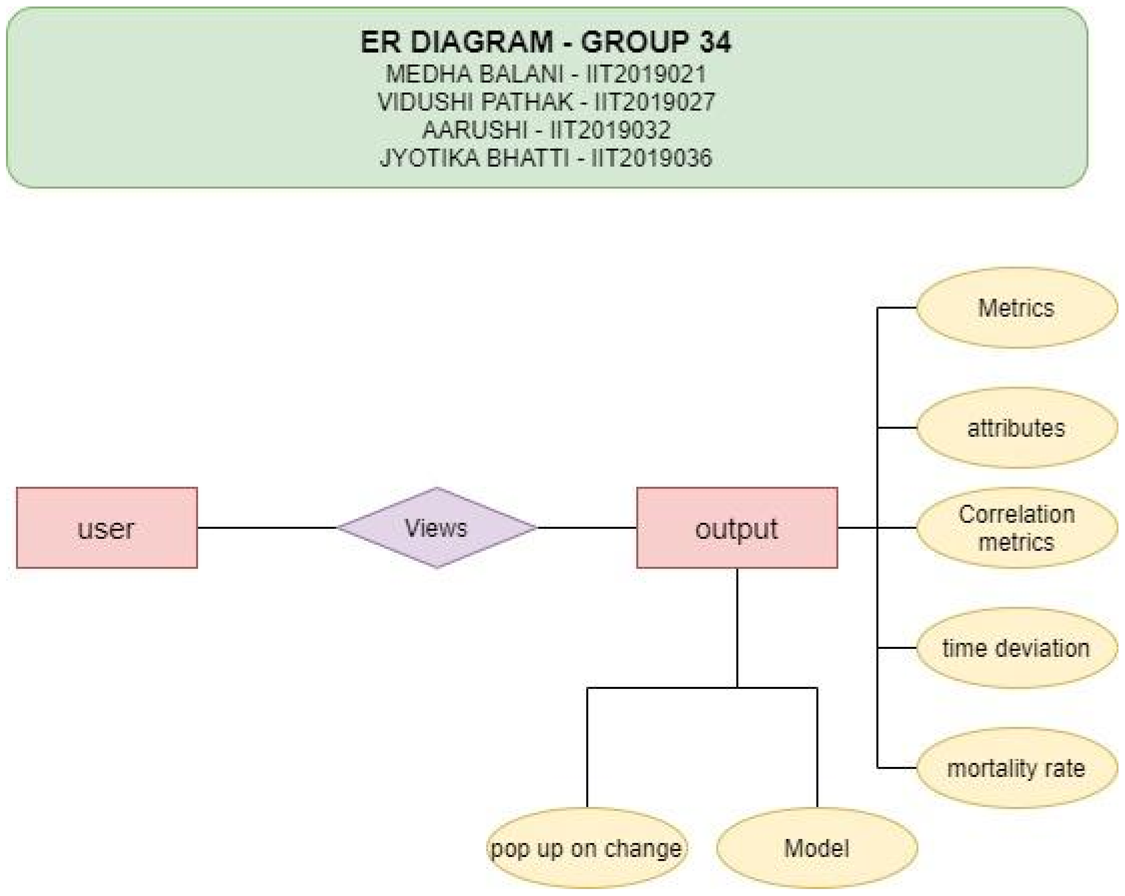
Members:

Medha Balani - IIT2019021	**Vidushi pathak - IIT2019027**
Aarushi - IIT2019032	**Jyotika Bhatti - IIT2019036**



ER Diagram:

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.



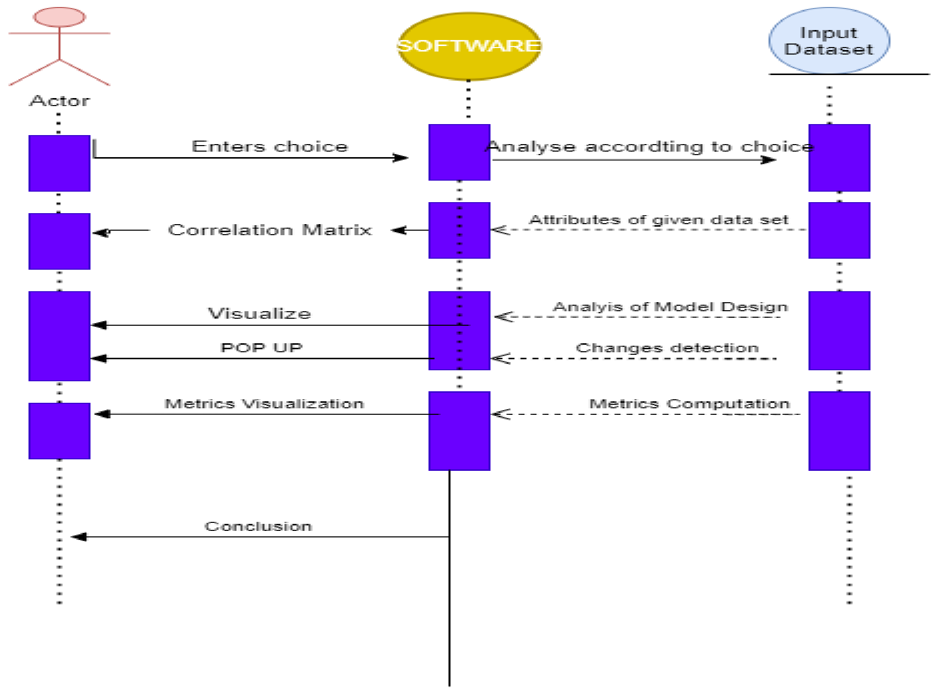
Sequence Diagram:

Sequence Diagram is an interaction diagram that details how operations are carried out -- what messages are sent and when, when the user sends the request for visualization or finding some kind of visualization or the metrics computation upon the given data set. Sequence diagrams are organized according to time. The time progresses as you go down the page. The objects involved in the operation are listed from left to right according to when they take part in the message sequence.
Below is the sequence diagram for making our dataset analysis and various graphs and metrics.

SEQUENCE DIAGRAM

Group Members

Medha Balani - IIT2019021
Vidushi Pathak - IIT2019027
Aarushi - IIT2019032
Jyotika Bhatti - IIT2019036



5. PSEUDO CODE:

Method : home page()
Function : shows the different functions that can be performed from the portal which includes,

1. Correlation Matrix
2. Metrics for computation , example: for probability , mean, standard deviation.
3. Visualization of the attributes
4. The data set of Hepatitis virus
5. Changes detection , dividing into chunks and pop -up on the model

Method : Correlation Matrix()
function:

```
User enters the system selects from options:
If (show correlation matrix){
    Select attributes:
    If (attributes)
        Show matrix
}
```

Method: Computation metrics

Function: another html file opens , which have a dropdown option of various metrics computation like probability of survival, standard deviation, accuracy of the model, execution time taken, complexity of model

```
Select
if(want to check probability){
    if(General pattern from data set){
        Show probability
        Show graphs
        Show standard deviation
    }

    Else if(provide custom inputs and check){
        Show graphs
        Show probability
        Show standard deviation
    }
}
```

Method : Visualization of data chunks()
Function:

Using the EDA Analysis of the data set , the data is to be divided into chunks , taking the different rows

```
Select attribute
if(attribute){
        Show visualisation output
        }
```

Method : Data set()
Function:
The original cleaned data set present in .csv format for visualisation

Method : Changes detection()
Function:

```
Check positions for change detected
if(position for change detected){
        Pop -up messages for change detection of the data
}
```