VISUALIZING AND PREDICTING HEART DISEASES WITH AN INTERACTIVE DASHBOARD

Project Report

INTRODUCTION

CHAPTER-1

INTRODUCTION

1.1 Project Overview

In this Project, we will be closely working with the heart disease prediction and for that, we will be looking into the heart disease dataset from that dataset we will derive various insights that help us know the weightage of each feature and how they are interrelated to each other but this time our sole aim is to detect the probability of person that will be affected by a saviour heart problem or not.

Machine learning proves to be effective in assisting in making decisions and predictions from the large quantity of data produced by the health care industry. This project aims to predict future heart disease by analysing data of patients which classifies whether they have heart disease or not using machine-learning algorithm. Machine Learning techniques can be a boon in this regard. Even though heart disease can occur in different forms, there is a common set of core risk factors that influence whether someone will ultimately be at risk for heart disease or not. By collecting the data from various sources, classifying them under suitable headings & finally analysing to extract the desired data we can say that this technique can be very well adapted to do the prediction of heart disease.

1.2 Purpose

Considering the anomalies in the existing system computerization of the whole activity is being suggested after initial analysis. It might have happened so many times that you or someone yours need doctors help immediately, but they are not available due to some reason. The Heart Disease Prediction application is an end user support and online consultation project. Here, we propose a web application that allows users to get instant guidance on their heart disease through an intelligent system online.

Various details are fed in the application and the heart disease associated with those details. Users can share their heart related issues with this application. It then processes user specific details to check for various illness that could be associated with it. After getting the result from the system, system suggests various doctors for treatment. The system allows user to view doctor's details. The system can be use in case of emergency.

LITERATURE SURVEY

CHAPTER-2

LITERATURE SURVEY

2.1 Existing problem

Heart disease is even being highlighted as a silent killer which leads to the death of a person without obvious symptoms. The nature of the disease is the cause of growing anxiety about the disease & its consequences. Hence continued efforts are being done to predict the possibility of this deadly disease in prior. So that various tools & techniques are regularly being experimented with to suit the present-day health needs. Machine Learning techniques can be a boon in this regard. Even though heart disease can occur in different forms, there is a common set of core risk factors that influence whether someone will ultimately be at risk for heart disease or not. By collecting the data from various sources, classifying them under suitable headings & finally analysing to extract the desired data we can conclude. This technique can be very well adapted to the do the prediction of heart disease. As the well-known quote says “Prevention is better than cure”, early prediction & its control can be helpful to prevent & decrease the death rates due to heart disease.

2.2 References

[1] Soni J, Ansari U, Sharma D & Soni S (2011). Predictive data mining for medical diagnosis: an overview of heart disease prediction. International Journal of Computer Applications, 17(8), 43-8

[2] Dangare C S & Apte S S (2012). Improved study of heart disease prediction system using data mining classification techniques. International Journal of Computer Applications, 47(10), 44-8.

[3] Ordonez C (2006). Association rule discovery with the train and test approach for heart disease prediction. IEEE Transactions on Information Technology in Biomedicine, 10(2), 334-43.

[4] Shinde R, Arjun S, Patil P & Waghmare J (2015). An intelligent heart disease prediction system using k-means clustering and Naïve Bayes algorithm. International Journal of Computer Science and Information Technologies, 6(1), 637-9.

[5] Bashir S, Qamar U & Javed M Y (2014, November). An ensemble-based decision support framework for intelligent heart disease diagnosis. In International Conference on Information Society (i-Society 2014) (pp. 259-64). IEEE. ICCRDA 2020 IOP Conf. Series: Materials Science and Engineering 1022 (2021) 012072 IOP Publishing doi:10.1088/1757-899X/1022/1/012072 9.

[6] Jee S H, Jang Y, Oh D J, Oh B H, Lee S H, Park S W & Yun Y D (2014). A coronary heart disease prediction model: the Korean Heart Study. BMJ open, 4(5), e005025.

[7] Ganna A, Magnusson P K, Pedersen N L, de Faire U, Reilly M, Ärnlöv J & Ingelsson E (2013). Multilocus genetic risk scores for coronary heart disease prediction. Arteriosclerosis, thrombosis, and vascular biology, 33(9), 2267-72.

[8] Jabbar M A, Deekshatulu B L & Chandra P (2013, March). Heart disease prediction using lazy associative classification. In 2013 International MutliConference on Automation, Computing,Communication, Control and Compressed Sensing (iMac4s) (pp. 40- 6). IEEE.

[9] Brown N, Young T, Gray D, Skene A M & Hampton J R (1997). Inpatient deaths from acute myocardial infarction, 1982-92: analysis of data in the Nottingham heart attack register. BMJ, 315(7101), 159-64.

[10] Folsom A R, Prineas R J, Kaye S A & Soler J T (1989). Body fat distribution and self-reported prevalence of hypertension, heart attack, and other heart disease in older women. International journal of epidemiologyy, 18(2), 361-7.

[11] Chen A H, Huang S Y, Hong P S, Cheng C H & Lin E J (2011, September). HDPS: Heart disease prediction system. In 2011 Computing in Cardiology (pp. 557- 60). IEEE.

[12] Parthiban, Latha and R Subramanian. "Intelligent heart disease prediction system using CANFIS and genetic algorithm." International Journal of Biological, Biomedical and Medical Sciences 3.3 (2008).

2.3 Problem Statement Definition

The major challenge in heart disease is its detection. There are instruments available which can predict heart disease but either it are expensive or are not efficient to calculate chance of heart disease in human. Early detection of cardiac diseases can decrease the mortality rate and overall complications. However, it is not possible to monitor patients every day in all cases accurately and consultation of a patient for 24 hours by a doctor is not available since it requires more sapience, time and expertise. Since we have a good amount of data in today's world, we can use various machine learning algorithms to analyse the data for hidden patterns. The hidden patterns can be used for health diagnosis in medicinal data.

IDEATION & PROPOSED SOLUTION

CHAPTER-3

IDEATION & PROPOSED SOLUTION

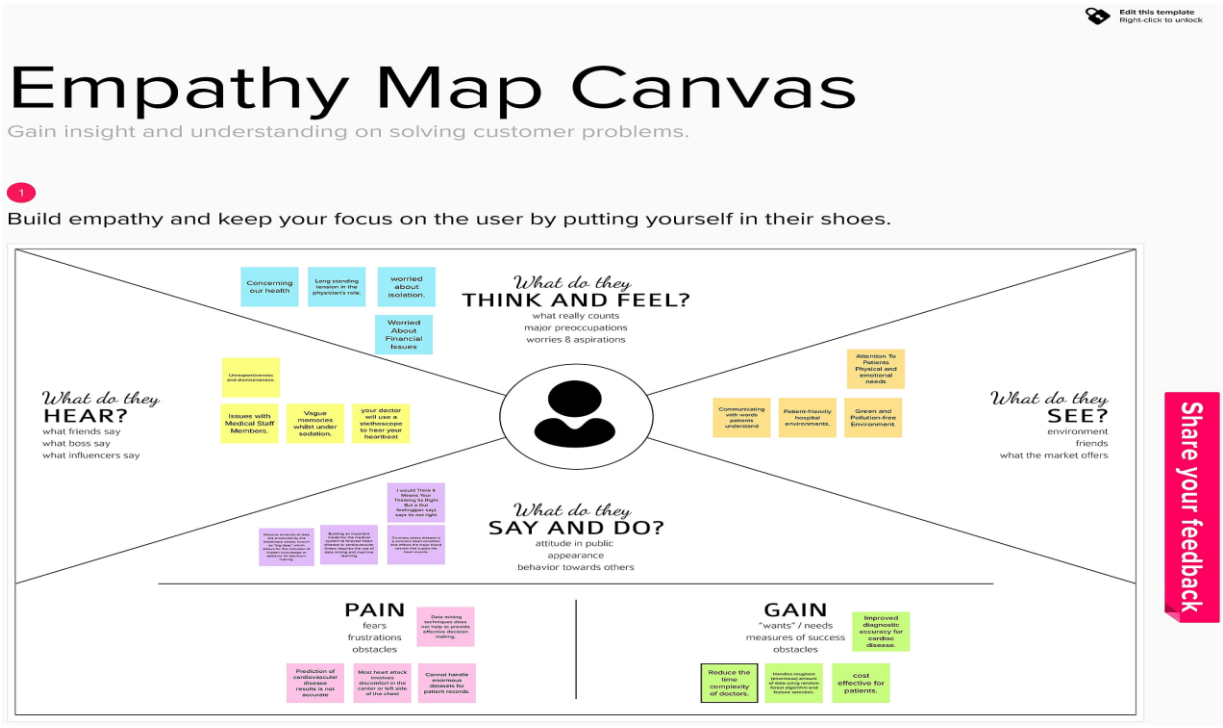
3.1 Empathy Map Canvas

An empathy map canvas is a more in-depth version of the original empathy map, which helps identify and describe the user's needs and pain points. And this is valuable information for improving the user experience.

Teams rely on user insights to map out what is important to their target audience, what influences them, and how they present themselves. This information is then used to create personas that help teams visualize users and empathize with them as individuals, rather than just as a vague marketing demographic or account number.

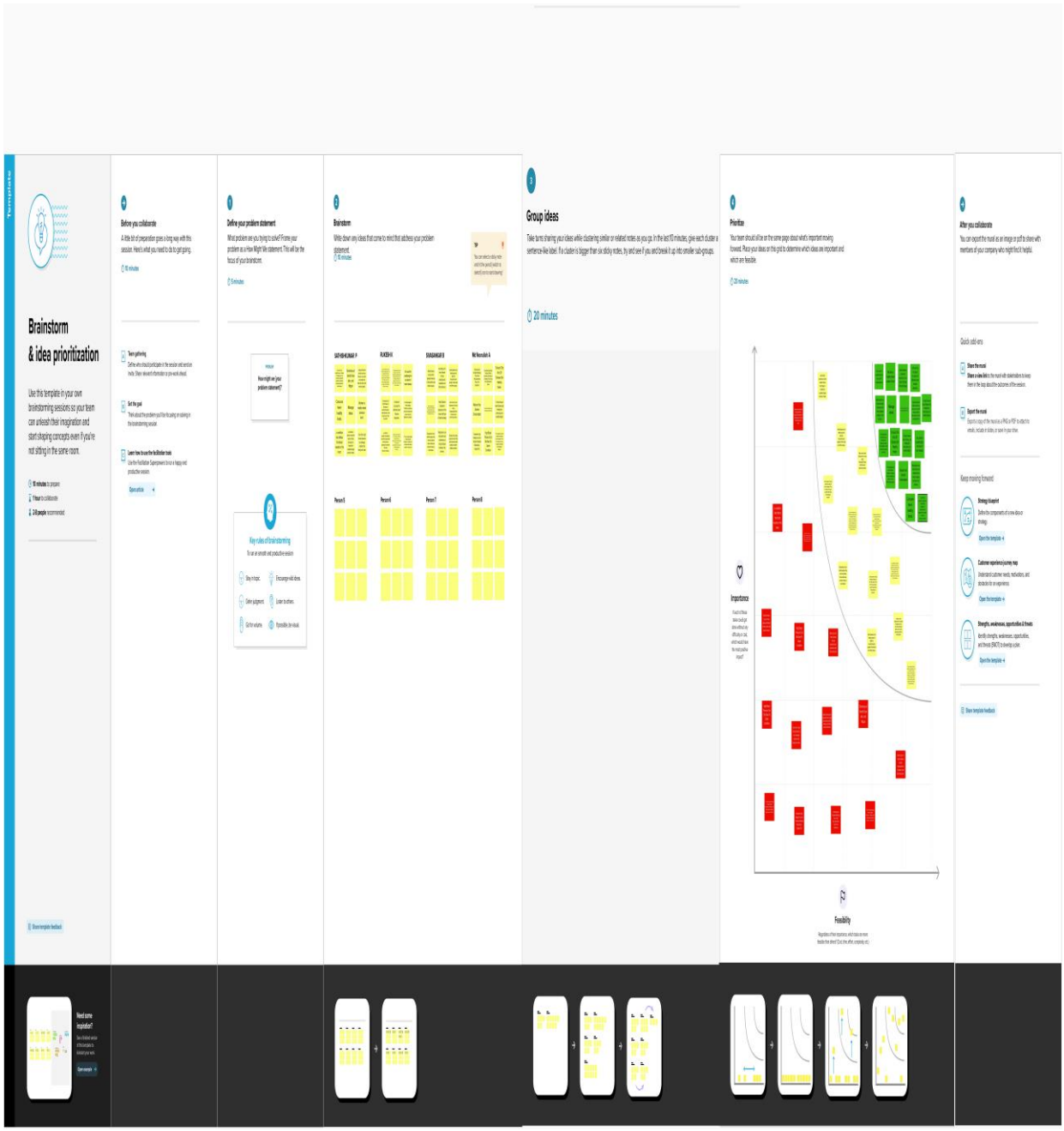
An empathy map canvas helps brands provide a better experience for users by helping teams understand the perspectives and mindset of their customers. Using a template to create an empathy map canvas reduces the preparation time and standardizes the process so you create empathy map canvases of similar quality.

Empathy Map Canvas Visualizing and Predicting Heart Diseases with an Interactive Dashboard:



3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.



3.3 Proposed Solution

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	• Angina is the discomfort caused when the muscles of heart is not supplied with sufficient oxygen rich blood. • High blood pressure is one of the major causes of heart disease as it damages arteries. • Blood pressure combined with diabetes can increase the risk even more. Heart rate with high blood pressure increases the risk of heart diseases. • Heart beat rate is directly proportional to the risk of coronary disease. The symptom of heart disease includes feeling gripping and tight usually on the chest but spread to shoulders up to the stomach. • The types of angina are atypical angina, typical angina, asymptomatic and non-anginal pain
2.	Idea / Solution description	The use of analytics in healthcare improves care by facilitating preventive care and EDA is a vital step while analysing data. The use of data analytics and virtualization tool to find the risk factors that causes heart disease is considered and predicted using K-means algorithm and the analysis is carried out using a publicly available data for heart disease.
3.	Novelty / Uniqueness	if three-dimensional (3D) imaging of congenital heart disease (CHD) could be more useful than two-dimensional (2D) visualisation for clarifying CHD anatomy and raising awareness. Stakeholders including the government and health insurance providers could gain from disease prediction. Patients who are at risk for certain illnesses or disorders can be identified.
4.	Social Impact / Customer Satisfaction	Patient satisfaction is also determined by exploring the particularity between the expected and perceived health services.
5.	Business Model (Revenue Model)	The Successful Business Strategies to Prevent Heart Disease and Stroke Toolkit provides information, materials, and tools that state programs can reference and distribute to businesses, primarily through employer and professional organizations. The Toolkit also assists state programs in addressing these CVH priority areas: • Providing health care coverage for employees and their families that includes primary and secondary prevention services addressing heart disease and stroke, as well as rehabilitation services for heart attack and stroke survivors.

		• Assuring detection and follow-up services with employees at the worksite to control high blood pressure and cholesterol. • Promoting adequate cost coverage or reimbursement for prescription drugs required
6.	Scalability of the Solution	Using this approach, we show that up to 98% accuracy is achieved. We also present a comparison against Naïve-Bayes classifier, where we show the random forest approach outperforms the former by a significant.

3.4 Problem Solution fit

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.

REQUIREMENT ANALYSIS

CHAPTER-4

REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposedsolution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Signup	Signup through Form. Signup throughGmail.
FR-2	User Confirmation	Confirmation via Email. Confirmation viaOTP.
FR-3	User Login	Login through Form.
FR-4	Data collection	User(patient) uploads the data Adminuploads the data to trainthe machine.
FR-5	Visualizing Data	User can visualize the trends on the heart disease through Dashboard created using IBM Cognos Analytics.
FR-6	Generating Report	User can viewhis/her health report and can make decisions accordingly.

4.2 Non-Functional requirements

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The application will have an easy-to-use graphicaluser interface. Users will find it simple to comprehend and utilize all the capabilities of the application. Any activity must be carried out with only a few clicks.

NFR-2	**Security**	The technique known as database replication shouldbe utilized for the application's security to ensure the safety of all crucial data. In the event of a crash, the systemto protect and restore the data.
NFR-3	**Reliability**	The structure must provide the functionality withstrength and dependability. The Programmer's advancements must also include Test Designer and Project Pioneer roles.
NFR-4	**Performance**	When various userswould employ the entire structure at once, thestructure shouldn't give in.

PROJECT DESIGN

CHAPTER-5

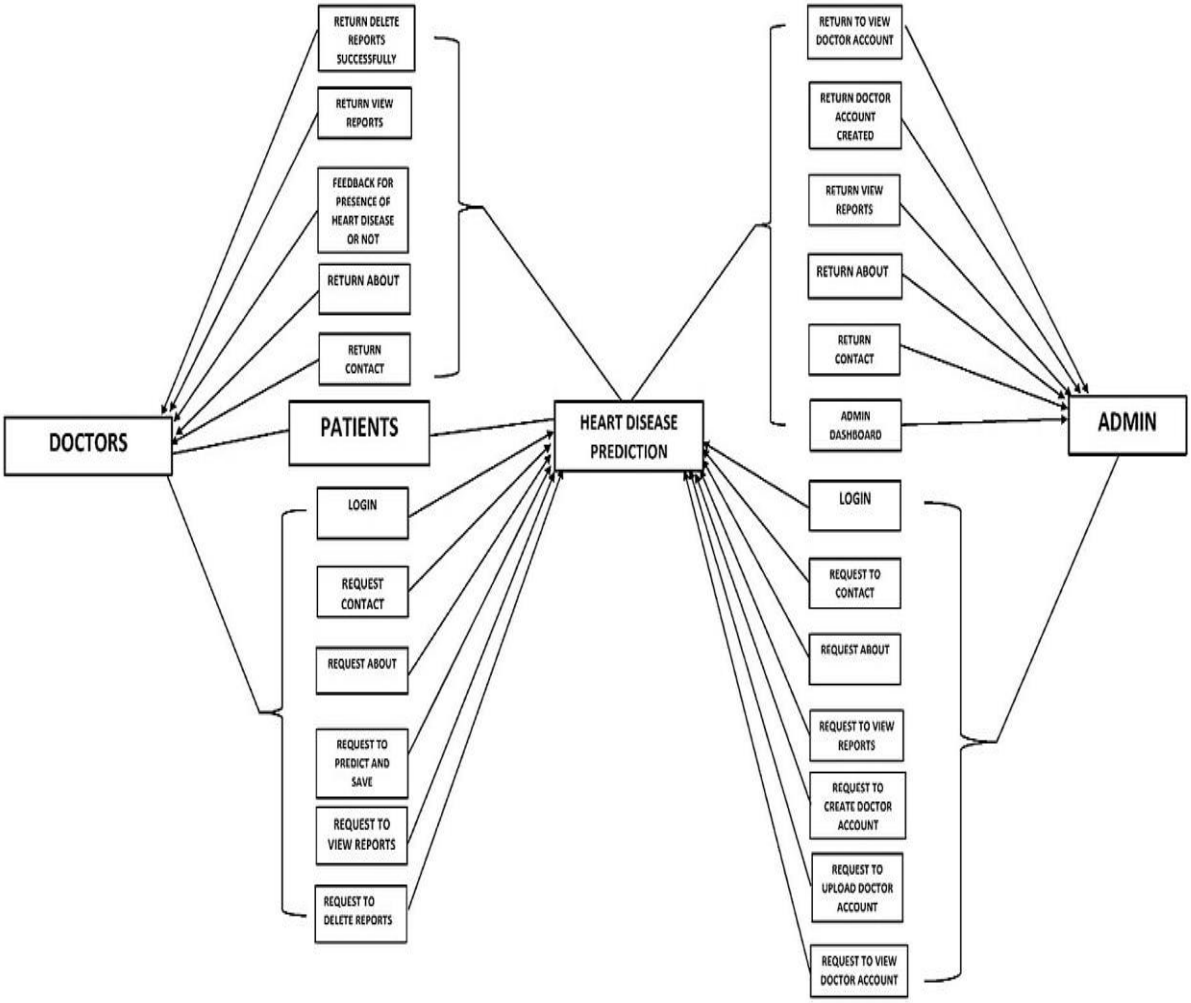
PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changesthe information, and where data is stored.

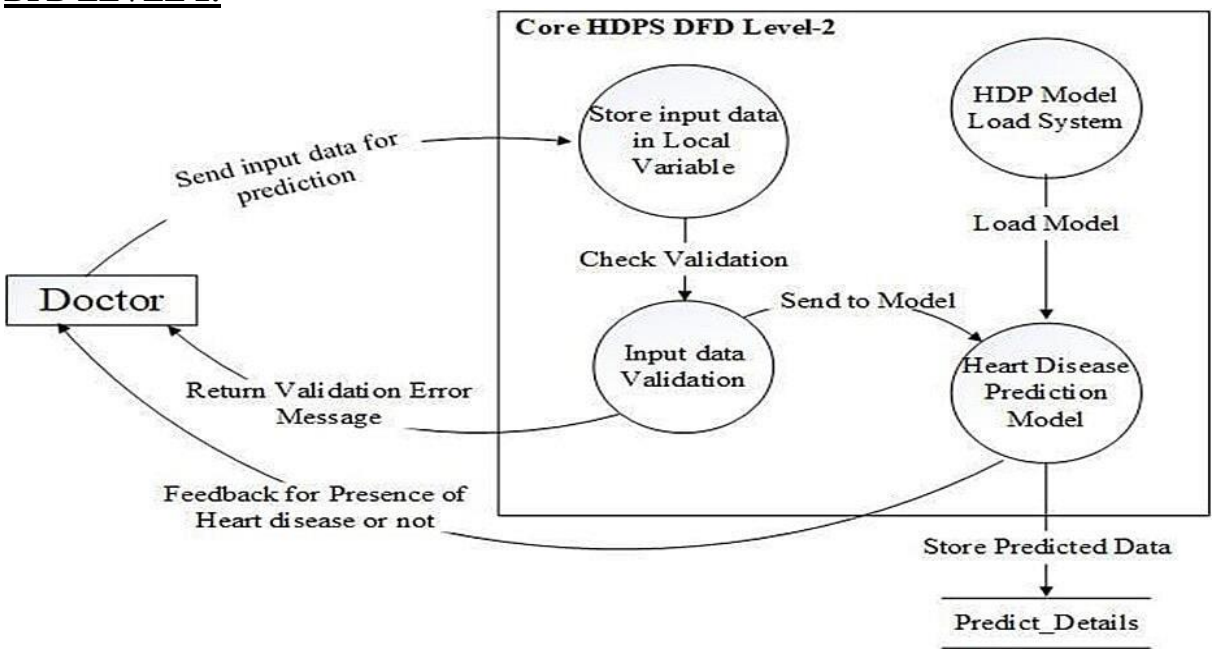
Data Flow Diagram for Heart Disease Prediction Dashboard:

DFD LEVEL 0:

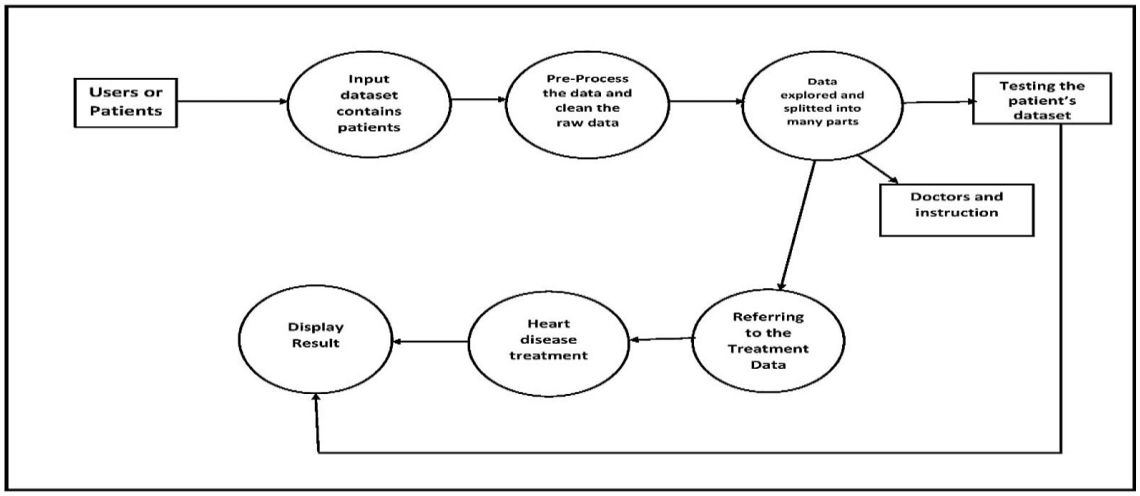


DFD LEVEL 1:

DFD LEVEL 2:



Flow:



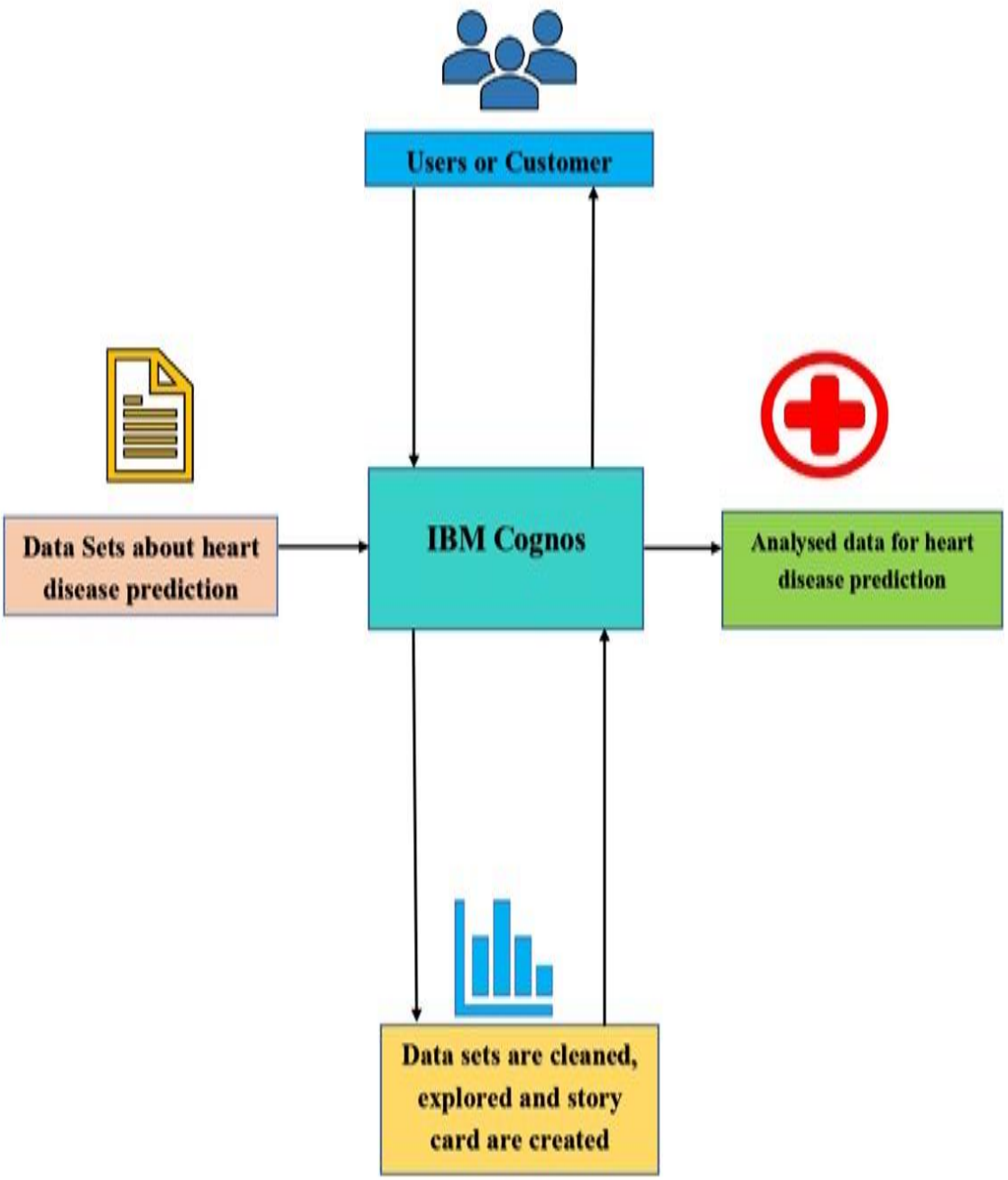
1. User createsan account in the application.
2. User enters the medicalrecords in the dashboard.
3. User can view the visualizations of trends in the form of graphs and charts for his/her medical records with the trained dataset.
4. User can view the accuracy of probability of occurrence of heart diseasein the dashboard.

5.2 Solution & Technical Architecture

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

1. Find the best tech solution to solve existing business problems.
2. Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.
3. Define features, development phases, and solution requirements.
4. Provide specifications according to which the solution is defined, managed, and delivered.

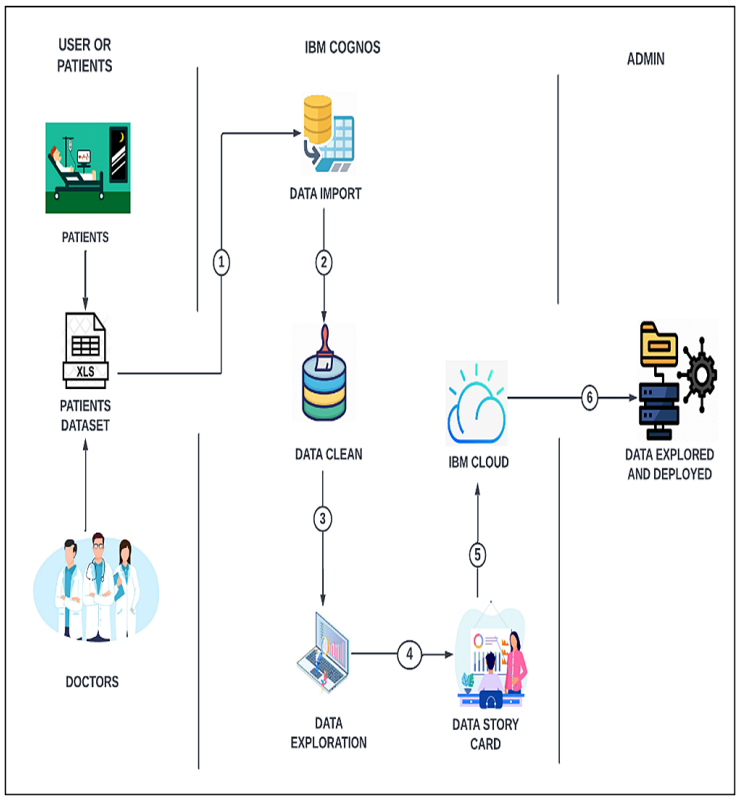
Solution Architecture Diagram:



Technology Stack (Architecture & Stack):

Technical Architecture:

Visualizing and Predicting Heart Diseases with an Interactive Dash Board



Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.

7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
9.	External API-2	Purpose of External API used in the application	Aadhar API, etc.
10	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model, etc.
11	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Opensource framework
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Technology used
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Technology used
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Technology used

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d

5.3 User Stories

User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
USER	Datasets	USN-1	As a user, I can gather the details of the patients.	I can access datasets my account Kaggle	High	Sprint-1
		USN-2	As an Analyst, I will check the data set andclean the dataset to create an efficient model.	I can clean the datasets using Cognos Analytics	High	Sprint-1
	Login	USN-3	As an Analyst I willalso correct the raw data and createa data module.	I can create Data module using Cognos Analytics	High	Sprint-1
USER	Cleaning, exploring data and creating model	USN-4	As a Data analyst, I create a predicted modelby also preparing story card with using explored data	I can explore datasets using Cognos Analytics	High	Sprint-2
		USN-5	As a Data analyst, I create a predicted modelby also preparing story card with using explored data	I can create story of heart disease prediction using Cognos Analytics	High	Sprint-2
Data Analyst	Data Prediction	USN-6	As a Dataanalyst, I willcreate different types of models in explored datato identify suitable model witheffectively and efficiently.	I can post my queriesin the dashboard	Medium	Sprint-3
		USN-7	As a DataAnalyst, I willanalysis of theheart disease patient's datasets.	I can to predict heart disease patients using Cognos Analytics	High	Sprint-3
Admin	Creation of deployed data UI	USN-8	As a Dataanalyst, I will create a heart disease prediction iterative dashboard.	I can view my updated health details.	High	Sprint-4

		USN-9	As an Analyst, I will import my analyzed modelinto suitable framework.		High	Sprint-4

PROJECT PLANNING & SCHEDULING

CHAPTER-6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation:

Use the below templateto create productbacklog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	TeamMembers
Sprint-1	Datasets	USN-1	As a user, I can gather the details of the patients.	2	High	SATHISHKUMAR P
Sprint-1		USN-2	As an Analyst, I will check the data set andclean thedataset to create an efficient model.	3	High	RUKESH K
Sprint-1		USN-3	As an Analyst I will also correctthe raw data and createa data module.	5	High	SIVASANKAR B, MD NOORULLAH A
Sprint-2	Cleaning, exploringdata and creating model	USN-4	As an Analyst I can create an Exploratory dataanalysis to identify the important factors of patient dataset	5	High	SATHISHKUMAR P, RUKESH K
Sprint-2		USN-5	As a Data analyst, I create a predicted modelby also preparing story card with using explored data	5	High	SIVASANKAR B, MD NOORULLAH A
Sprint-3	Data Prediction	USN-6	As a Data analyst, I will createdifferent types of modelsin explored data to identify	5	Medium	SATHISHKUMAR P, MD NOORULLAH A

Sprint	Functional Requirement (Epic)	User Story Number	User Story/ Task	StoryPoints	Priority	Team Members
			suitable modelwith effectively and efficiently.			
Sprint-3		USN-7	As a Data Analyst, I will analysis of the heart disease patient's datasets.	5	High	SATHISHKUMAR P
Sprint-4	Creation of deployed data UI	USN-8	As a Data analyst, I will create a heart disease prediction iterative dashboard.	5	High	SIVASANKAR B
Sprint-4		USN-9	As an Analyst, I will import my analyzedmodel intosuitable framework.	5	High	RUKESH K, MD NOORULLAH A

6.2 Sprint Delivery Schedule

Project Tracker,Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	SprintStart Date	Sprint End Date (Planned)	Story Points Completed (as on PlannedEnd Date)	Sprint Release Date (Actual)
Sprint-1	10	5 Days	24 Oct 2022	29 Oct 2022	10	29 Oct 2022
Sprint-2	10	5 Days	31 Oct 2022	05 Nov 2022	10	05 Nov 2022
Sprint-3	10	5 Days	07 Nov 2022	12 Nov 2022	10	12 Nov 2022
Sprint-4	10	5 Days	14 Nov 2022	19 Nov 2022	10	19 Nov 2022

Velocity:

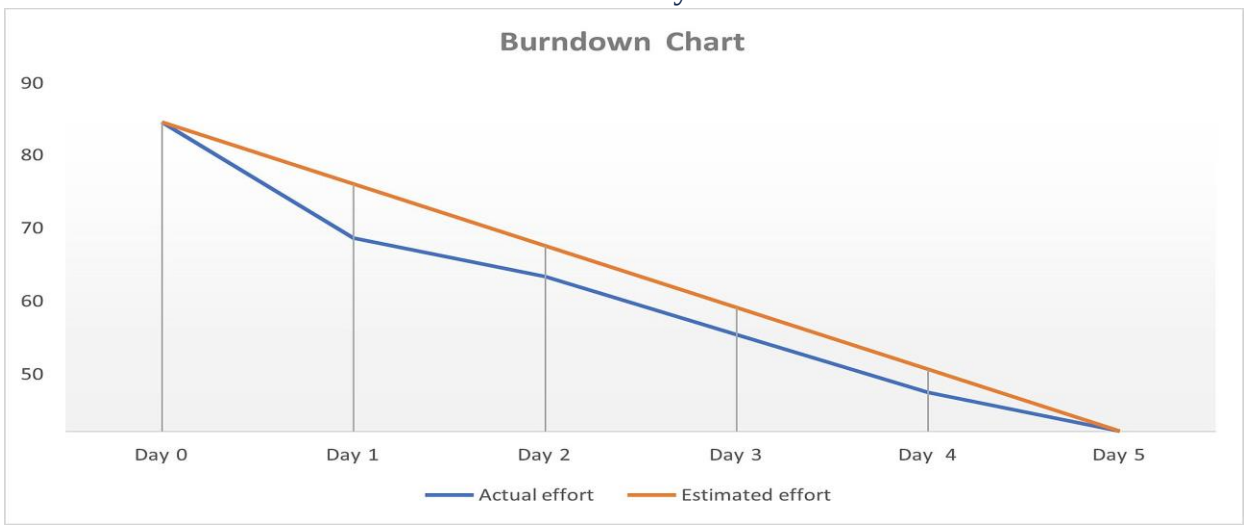
Imagine we have a 05-day sprint duration, and the velocity of the team is 10 (points per sprint). Let's calculate the team's average velocity (AV)per iteration unit (storypoints per day)

AV=Sprint Duration/Velocity=10/5

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologiessuch as Scrum. However, burn down chartscan be applied to any project containing measurable progress over time.

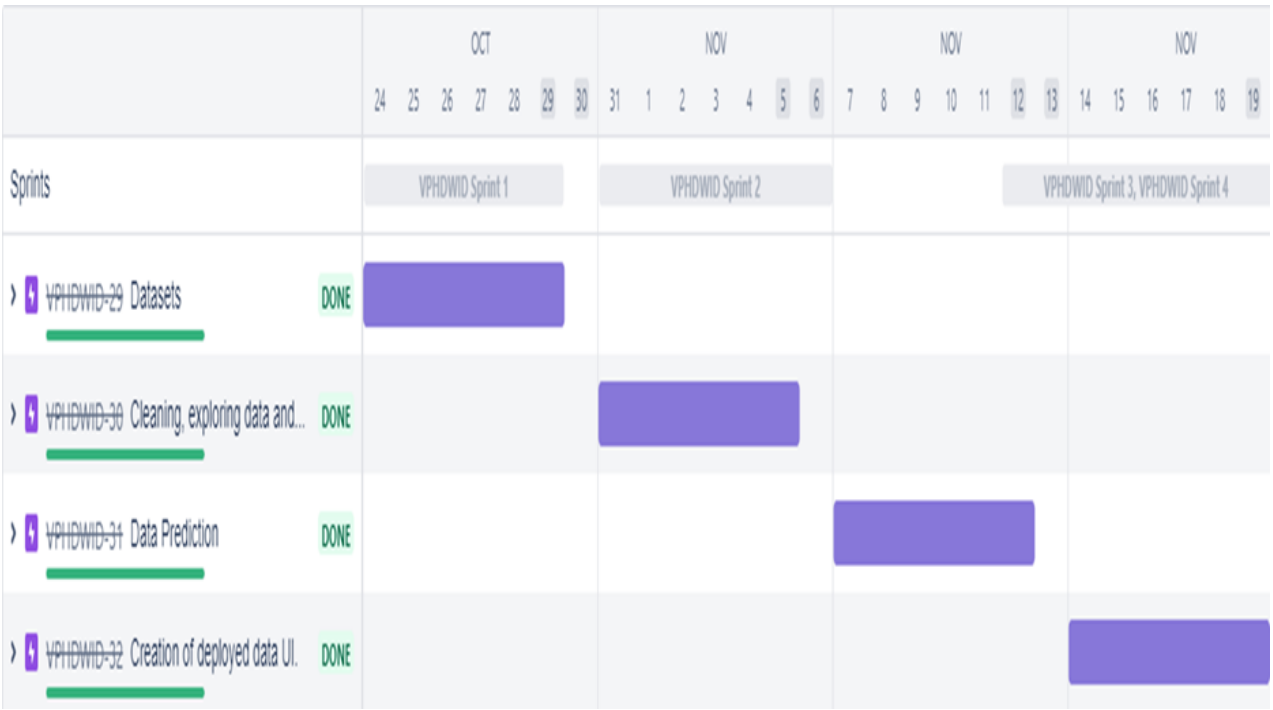
Goal:60 hoursin 5 days

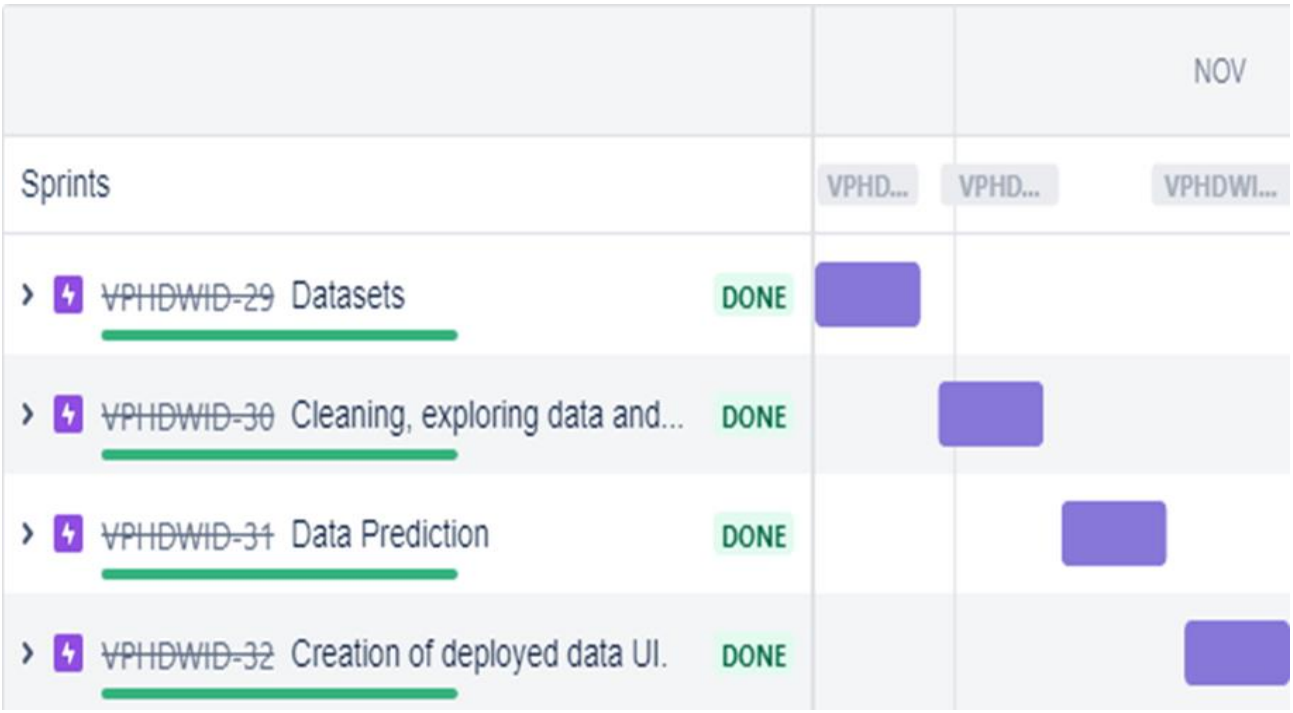


6.3 Reports from JIRA

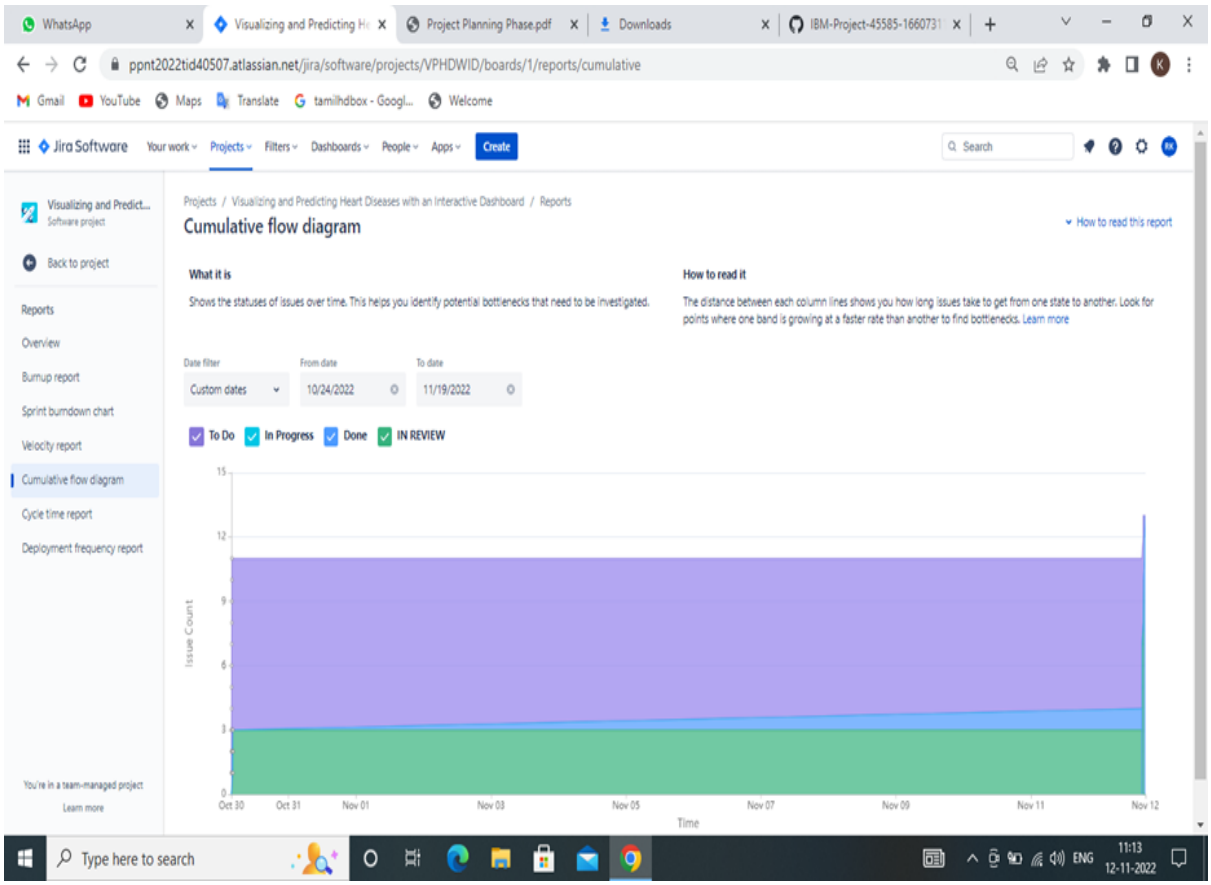
Jira helps teams plan, assign, track, report, and manage work and brings teams together for everything from agile software development and customer support to start-ups and enterprises. Software teams build better with Jira Software, the #1 tool for agile teams.

As a Jira administrator, you can create project categories so your team can view work across related projects in one place. Your team can use categories in advanced search, filters, reports, and more.

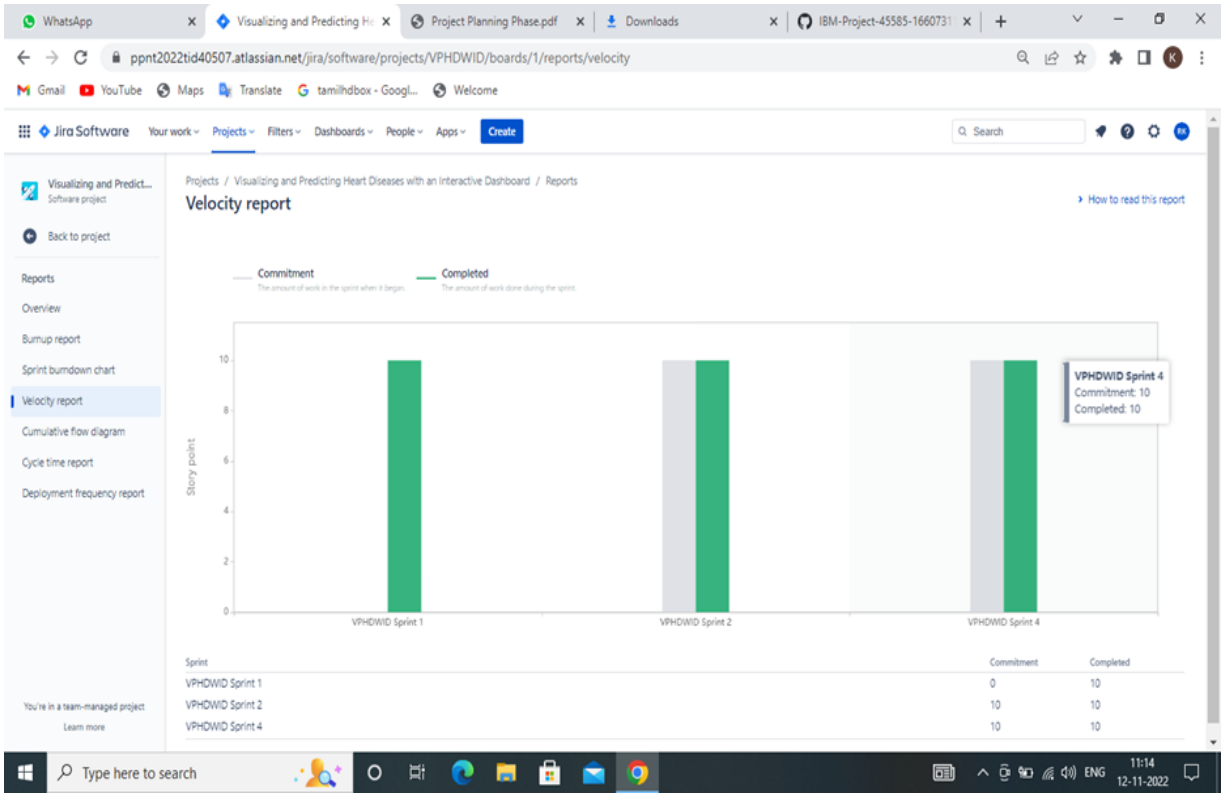




CUMULATIVE JIRA FILE:

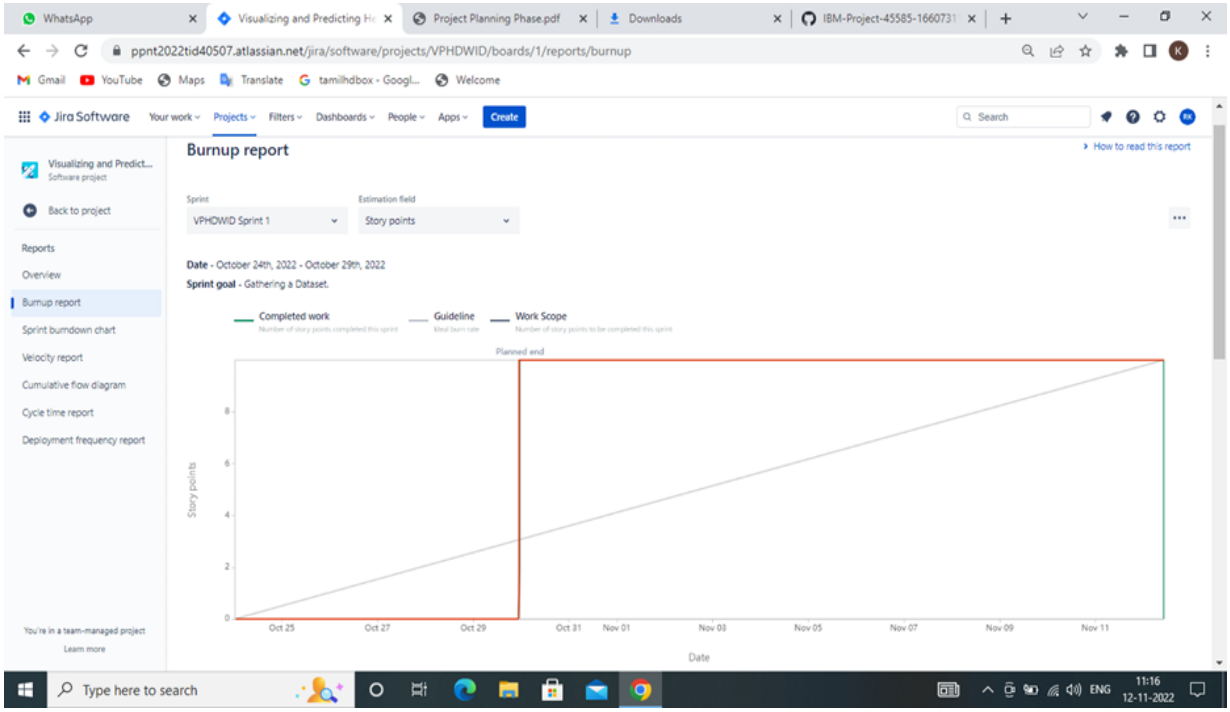


VELOCITY:

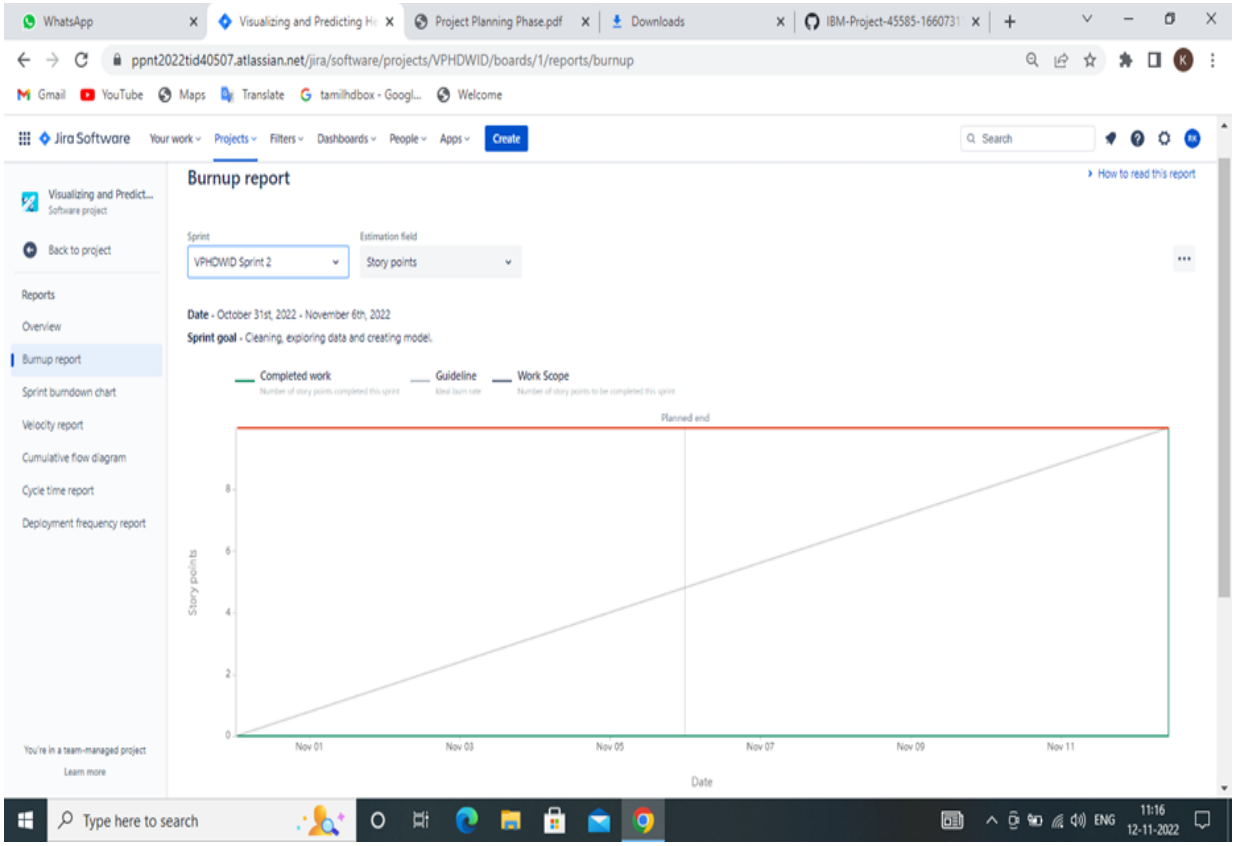


BURNUP REPORT:

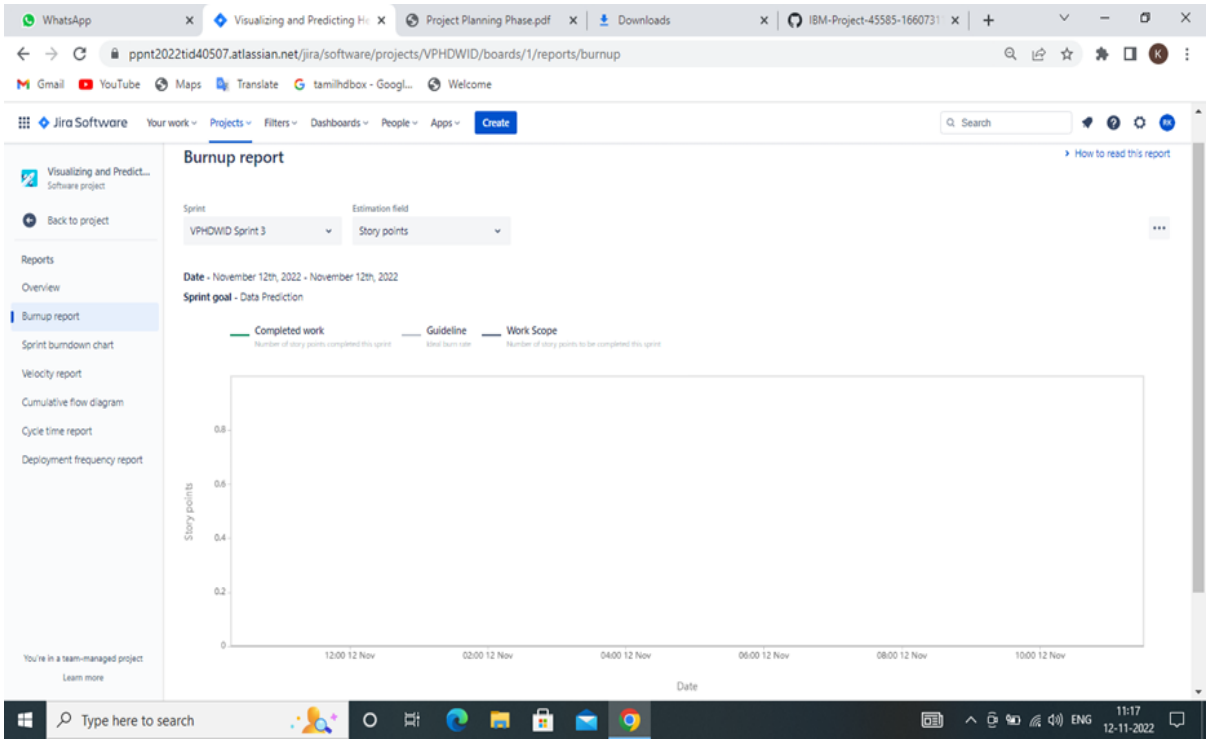
SPRINT-1:



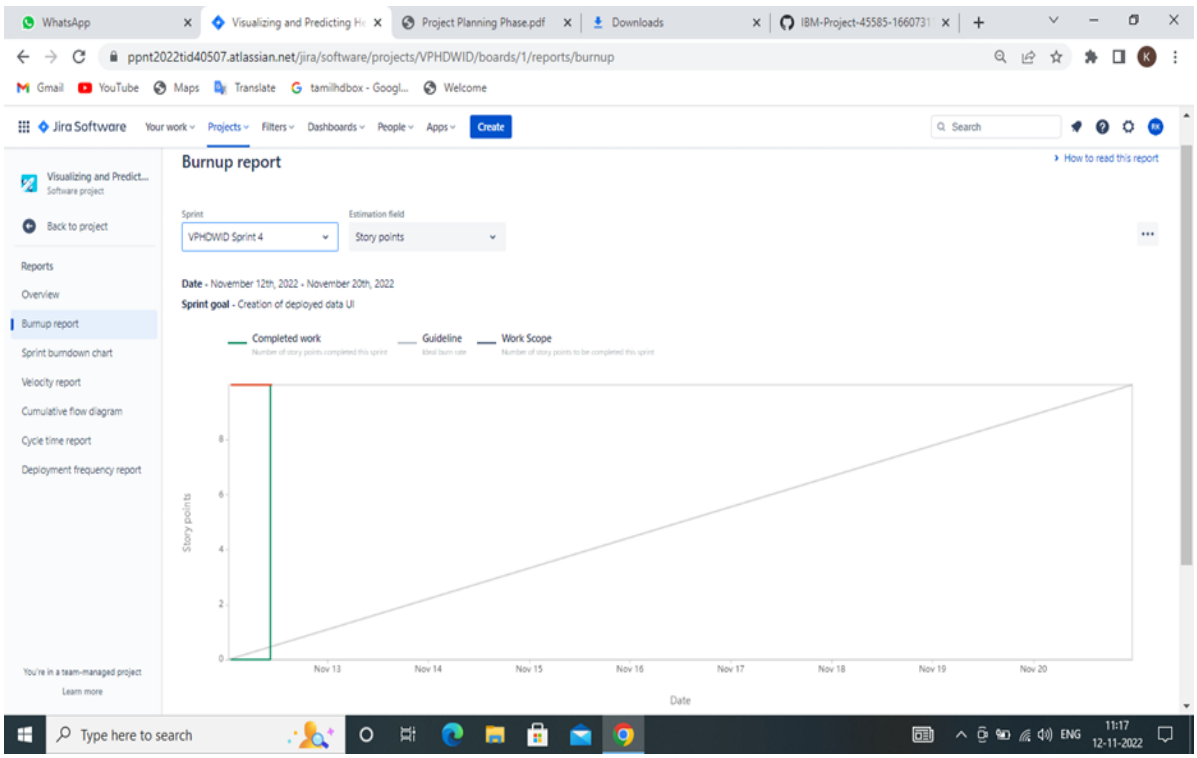
SPRINT-2



SPRINT-3

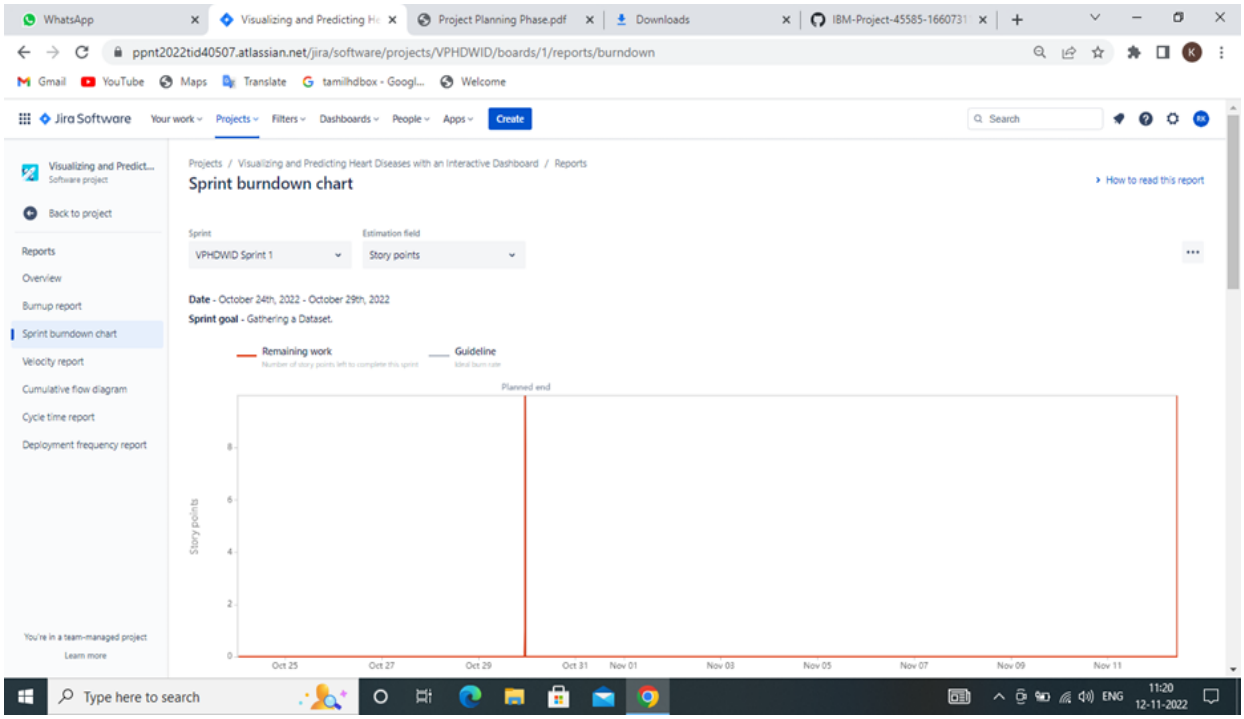


SPRINT-4

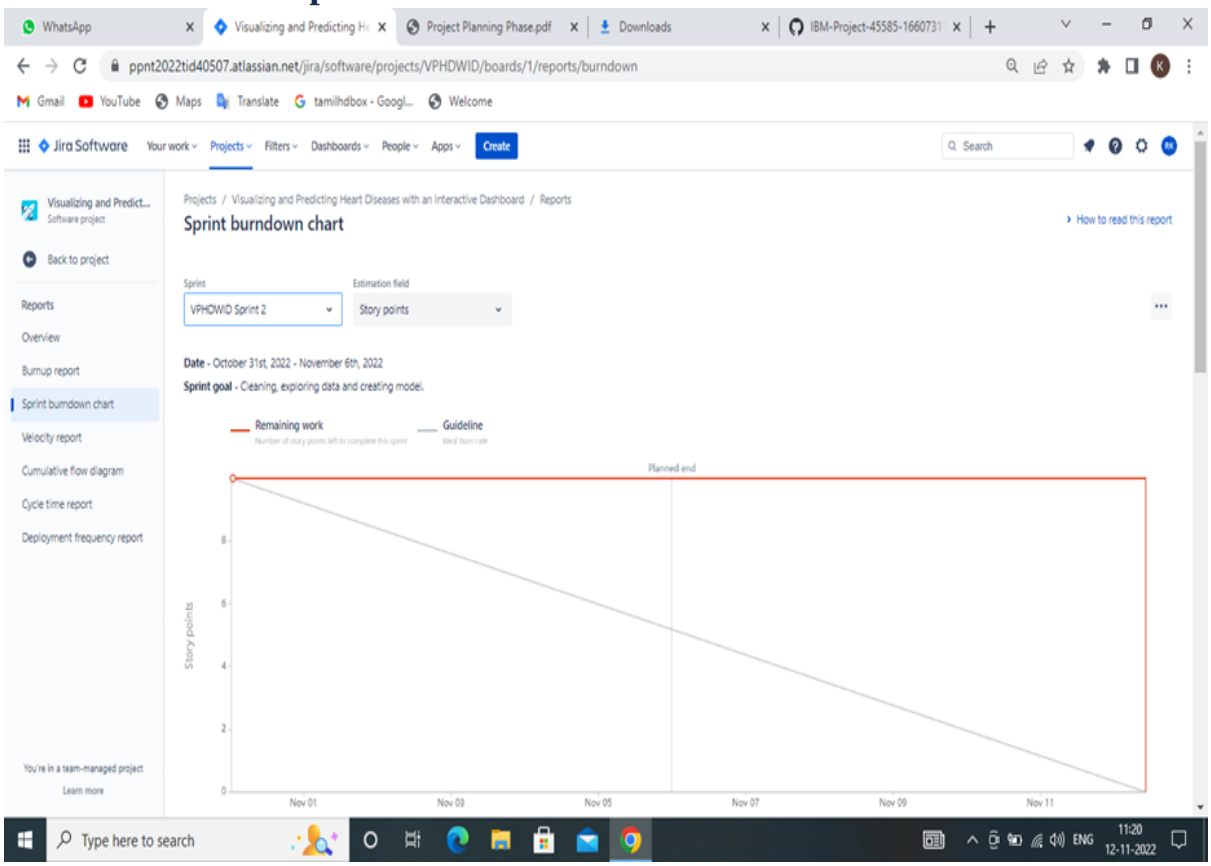


Burndown Chart:

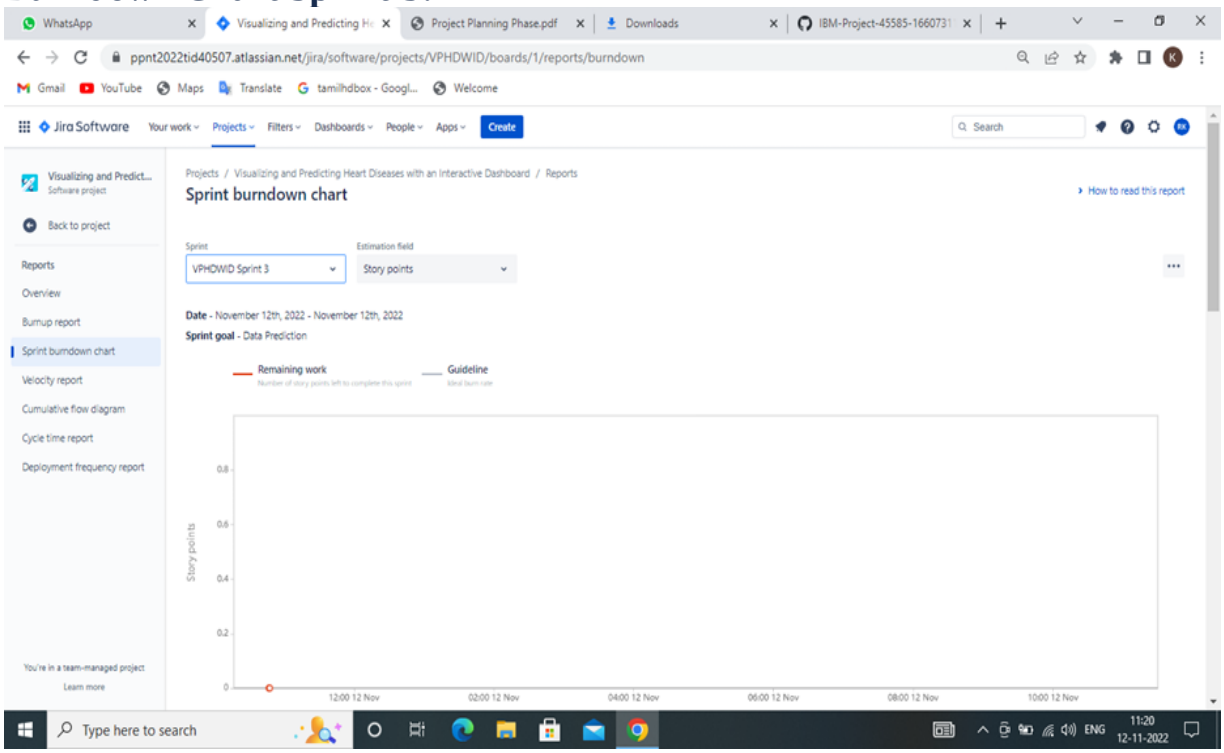
Burndown Chart Sprint-1:



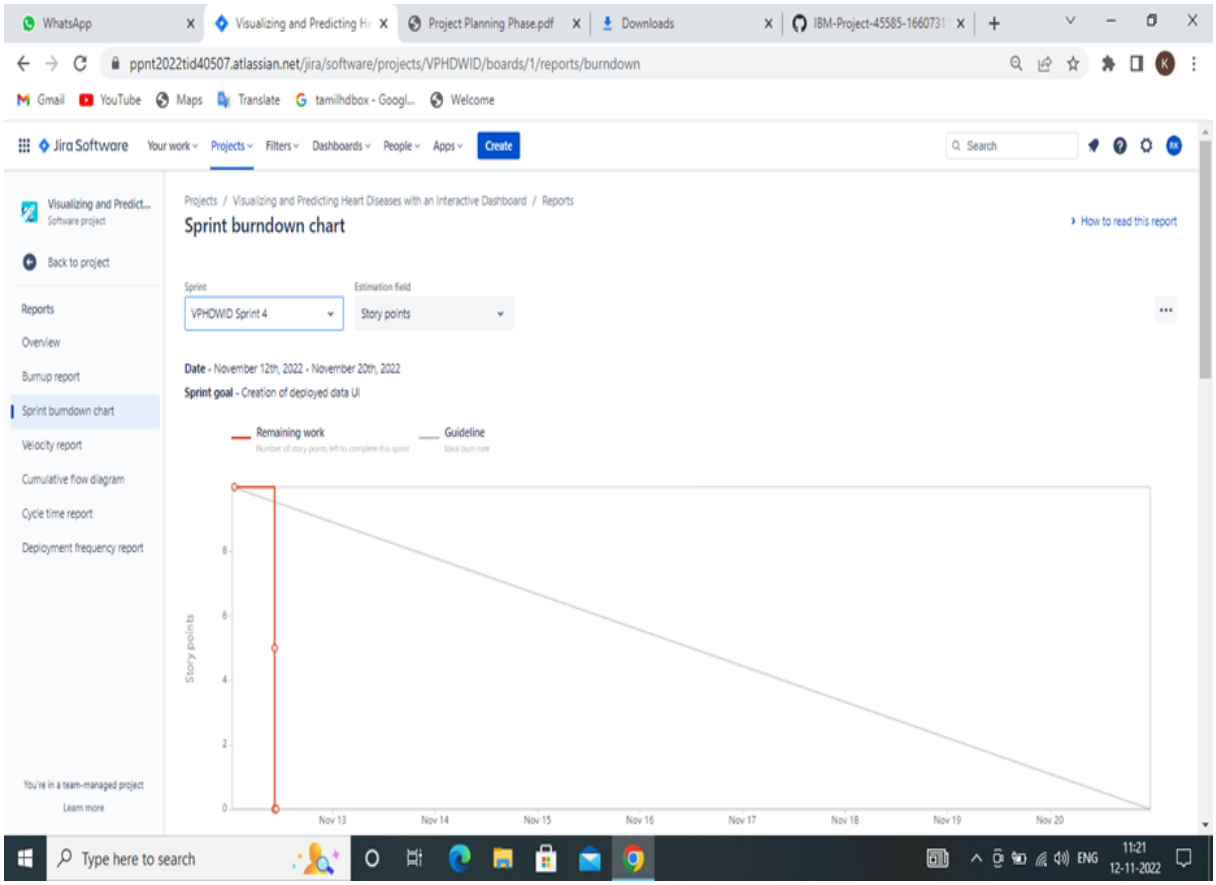
Burndown Chart Sprint-2:



Burndown Chart Sprint-3:



Burndown Chart Sprint-4:



CODING & SOLUTIONING

CHAPTER-7

CODING & SOLUTIONING

7.1 Random Forest

Random Forest is a supervised learning algorithm. Random forest can be used for both classification and regression problems, by using random forest regressor we can use random forest on regression problems. But we have used random forest on classification in this internship project so we will only consider the classification part.

7.1.1 Random Forest pseudocode

- Randomly select “k” features from total “m” features. Where k << m
- Among the “k” features, calculate the node “d” using the best split point.
- Split the node into daughter nodes using the best split.
- Repeat 1 to 3 steps until the “l” number of nodes has been reached.
- Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees.

7.1.2 Random Forest prediction pseudocode

1. Takes the test features and use the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome (target).
2. Calculate the votes for each predicted target.
3. Consider the highly voted predicted target as the final prediction from the random forest algorithm.

Code:

```
max_accuracy = 0

for x in range(500):

rf_classifier = RandomForestClassifier(random_state=x)

rf_classifier.fit(X_train,Y_train)

Y_pred_rf = rf_classifier.predict(X_test)

current_accuracy =

round(accuracy_score(Y_pred_rf,Y_test)*100,2)

if(current_accuracy>max_accuracy):

max_accuracy = current_accuracy

best_x = x

print(max_accuracy)

print(best_x)

rf_classifier =

RandomForestClassifier(random_state=best_x)

rf_classifier.fit(X_train,Y_train)
```

```
Y_pred_rf = rf_classifier.predict(X_test)
Y_pred_rf.shape
score_rf = round(accuracy_score(Y_pred_rf,Y_test)*100,2)
score_rf
```

7.2. K-Nearest Neighbors

We can implement a KNN model by following the below steps:

● Load the data

● Initialize the value of k

● For getting the predicted class, iterate from 1 to total number of training data points

Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.

○ Sort the calculated distances in ascending order based on distance values

○ Get top k rows from the sorted array

○ Get the most frequent class of these rows

○ Return the predicted class

Code:

```
knn_classifier=
KNeighborsClassifier(n_neighbors=31,leaf_size=30)
knn_classifier.fit(X_train,Y_train)
Y_pred_knn = knn_classifier.predict(X_test)
score_knn = round(accuracy_score(Y_pred_knn,Y_test)*100,2)
score_knn
```

7.3 Decision Tree

Pseudocode:

● Place the best attribute of the dataset at the root of the tree.

● Split the training set into subsets. Subsets should be made in such a way that each subset contains data with the same value for an attribute.

● Repeat step 1 and step 2 on each subset until you find leaf nodes in all the branches of the tree.

Assumptions while creating a Decision Tree- At the beginning, the whole training set is considered as the root. Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model. Records are distributed recursively on the basis of attribute values. Order to place attributes as root or internal node of the tree is done by using some statistical approach.

The popular attribute selection measures:

- Information gain
- Gini index

Attribute selection method- A dataset consists of "n" attributes then deciding which attribute to place at the root or at different levels of the tree as internal nodes is a complicated step. By just randomly selecting any node to be the root can't solve the issue. If we follow a random approach, it may give us bad results with low accuracy. To solve this attribute selection problem, researchers worked and devised some solutions. They suggested using some criterion like information gain, Gini index, etc. These criterions will calculate values for every attribute. The values are sorted, and attributes are placed in the tree by following the order i.e., the attribute with a high value (in case of information gain) is placed at the root. While using information Gain as a criterion, we assume attributes to be categorical, and for Gini index, attributes are assumed to be continuous.

Gini Index - Gini Index is a metric to measure how often a randomly chosen element would be incorrectly identified. It means an attribute with a lower Gini index should be preferred.

Code:

```
dt_classifier = DecisionTreeClassifier(
max_depth=20,
min_samples_split=2,
min_samples_leaf=1,
min_weight_fraction_leaf=0.00001,
max_features='auto',
random_state=46)
dt_classifier.fit(X_train, Y_train)
Y_pred_dt=dt_classifier.predict(X_test)
score_dt = round(accuracy_score(Y_pred_dt,Y_test)*100,2)
score_dt
```

7.4 Naïve Bayes

Bayes'Theorem is stated as:

P(h|d) = (P(d|h) * P(h)) / P(d)

P(h|d) is the probability of hypothesis h given the data d. This is called the posterior probability.

P(d|h) is the probability of data d given that the hypothesis h was true.

P(h) is the probability of hypothesis h being true (regardless of the data). This is called the prior probability of h.

P(d) is the probability of the data (regardless of the hypothesis).

We are interested in calculating the posterior probability of P(h|d) from the prior probability p(h) with P(D) and P(d|h). After calculating the posterior probability for a number of different hypotheses, we will select the hypothesis with the highest probability. This is the maximum probable hypothesis and may formally be called the (MAP) hypothesis.

This can be written as:

MAP(h) = max(P(h|d)) or

MAP(h) = max((P(d|h) * P(h)) / P(d)) or

MAP(h) = max(P(d|h) * P(h))

The P(d) is a normalizing term which allows us to calculate the probability. We can drop it when we are interested in the most probable hypothesis as it is constant and only used to normalize. Back to classification, if we have an even number of instances in each class in our training data, then the probability of each class (e.g. P(h)) will be equal. Again, this would be a constant term in our equation, and we could drop it so that we end up with:

MAP(h) = max(P(d|h))

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values. It is called Naive Bayes or Idiot Bayes because the calculation of the probabilities for each hypothesis are simplified to make their calculation tractable. Rather than attempting to calculate the values of each attribute value P (d1, d2, d3|h), they are assumed to be conditionally independent given the target value and calculated as P(d1|h) * P(d2|H) and so on. This is a very strong assumption that is most unlikely in real data, i.e. that the attributes do not interact. Nevertheless, the approach performs surprisingly well on data where this assumption does not hold.

MAP(h) = max(P(d|h) * P(h))

Gaussian Naïve Bayes:

mean(x) = 1/n * sum(x)

Where n is the number of instances and x are the values for an input variable in your training data. We can calculate the standard deviation using the following equation:

standard deviation(x) = sqrt (1/n * sum(xi-mean(x)^2))

This is the square root of the average squared difference of each value of x from the mean value of x, where n is the number of instances, sqrt() is the square root function, sum() is the sum function, xi is a specific value of the x variable for the i'th instance and mean(x) is described above, and ^2 is the square. Gaussian PDF with a new input for the variable, and in return the Gaussian PDF will provide an estimate of the probability of that new input value for that class.

pdf (x, mean, sd) = (1 / (sqrt (2 * PI) * sd)) * exp (-((x-mean^2)/(2*sd^2)))

Where pdf(x) is the Gaussian Probability Density Function (PDF), sqrt () is the square root, mean and sd are the mean and standard deviation calculated above, Pi is the numerical constant, exp () is the numerical constant e or Euler's number raised to power and x is the input value for the input variable.

Code:

```
nb_classifier = GaussianNB( var_smoothing=1e-50)
nb_classifier.fit(X_train,Y_train)
nb_classifier.predict(X_test)
Y_pred_nb = nb_classifier.predict(X_test)
score_nb = round(accuracy_score(Y_pred_nb,Y_test)*100,2)
score_nb
```

7.5 Web App Code

```
import numpy as np
import pickle
from flask import Flask, request, render_template
# Load ML model
model = pickle.load(open('models.pkl', 'rb'))
# Create application
app = Flask(__name__)
# Bind home function to URL
@app.route('/')
def home():
return
render_template('Heart_Disease_Classifier.html')
# Bind predict function to URL
@app.route('/predict', methods =['POST'])
def predict():
# Put all form entries values in a list
features = [float(i) for i in request.form.values()]
# Convert features to array
array_features = [np.array(features)]
# Predict features
prediction = model.predict(array_features)
```

output = prediction

Check the output values and retrieve the result with

html tag based on the value

if output == 1:

return

render_template('Heart_Disease_Classifier.html',result = 'The patient is not likely to have heart disease!')

else:

return

render_template('Heart_Disease_Classifier.html',

result = 'The patient is likely to have heart disease!')

if __name__ == '__main__':

#Run the application

app.run()

7.6 Libraries used

Python has a vast reserve of inbuilt standard libraries which includes areas like web services tools, string operation, data analysis, and machine learning, etc. The complex programming tasks can be dealt with ease using these inbuilt libraries as it reduces the size of code with many inbuilt functions that do the job pretty well for its user.\

7.6.1 Data Visualization

● Matplotlib: Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical mathematics extension NumPy, a big data numerical handling resource.

○ pyplot

○ rcParams

○ rainbow

● Seaborn: Seaborn is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis. Seaborn works easily with dataframes and the Pandas library. The graphs created can also be customized easily.

7.6.2 Data Manipulation

● NumPy: The NumPy library in python is used for scientific computing and array manipulation. It can perform different operations such as indexing of an array, sequencing, and slicing, etc.

● Pandas: The Pandas library in python is used for structuring, manipulating, and organizing data in a tabular structure called the data frame which is further used for data analysis.

● Scikit-learn:

○ sklearn.model_selection

■ train_test_split

○ sklearn.preprocessing

■ StandardScaler

■ LabelEncoder

7.6.3 Data Modeling

● Scikit-learn: Scikit-learn is one of the most useful libraries that python offers. It has various statistical learning algorithms such as regression models (linear regression, logistic regression), SVM's, random forest for classification tasks and k-means for clustering, etc.

○ sklearn.ensemble.RandomForestClassifier

○ sklearn.neighbors.KNeighborsClassifier

○ sklearn.tree.DecisionTreeClassifier

○ sklearn.naive_bayes.GaussianNB

7.6.4 Data Validation

● Scikit-learn-metrics: The sklearn.metrics module implements several loss, score, and utility functions to measure classification performance.

sklearn.metrics - log_loss, roc_auc_score, precision_score, f1_score, recall_score, roc_curve, auc, plot_roc_curve, classification_report, confusion_matrix, accuracy_score, fbeta_score, matthews_corrcoef

● Mlxtend: Mlxtend (machine learning extensions) is a Python library of useful tools for day-to-day data science tasks. mlxtend.plotting - plot_confusion_matrix

TESTING

CHAPTER-8

TESTING

8. Testing

8.1 Acceptance Testing

UAT Execution & Report Submission

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [Visualizing and Predicting Heart Diseases]project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This reportshows the numberof resolved or closedbugs at each severity level, and how they

were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
BP	10	4	2	3	19
Cholesterol	10	7	3	4	24
Thallium	2	3	0	1	6
ECG	11	2	4	20	37
Obesity	9	6	1	0	16
St depression	3	4	1	1	9
Totals	45	21	11	29	111

1. Test Case Analysis

This report shows the number of test cases that have passed, failed,and untested

Section	Total Cases	Not Tested	Fail	Pass
BP	22	0	0	22
Cholesterol	31	0	0	31

Thallium	4	0	0	4
ECG	51	0	0	51

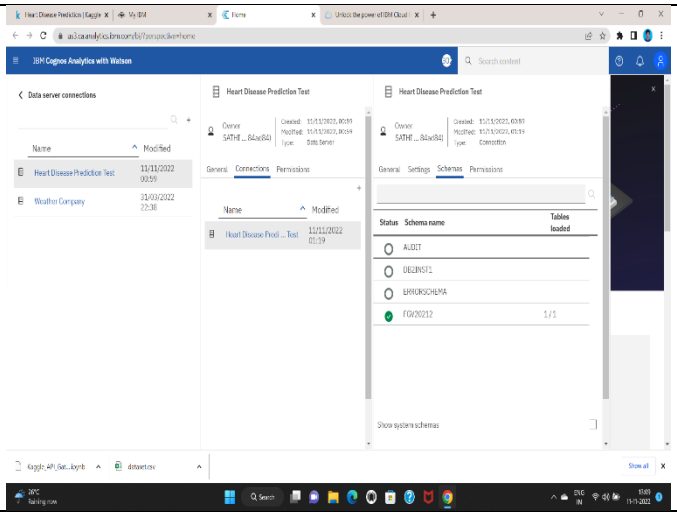
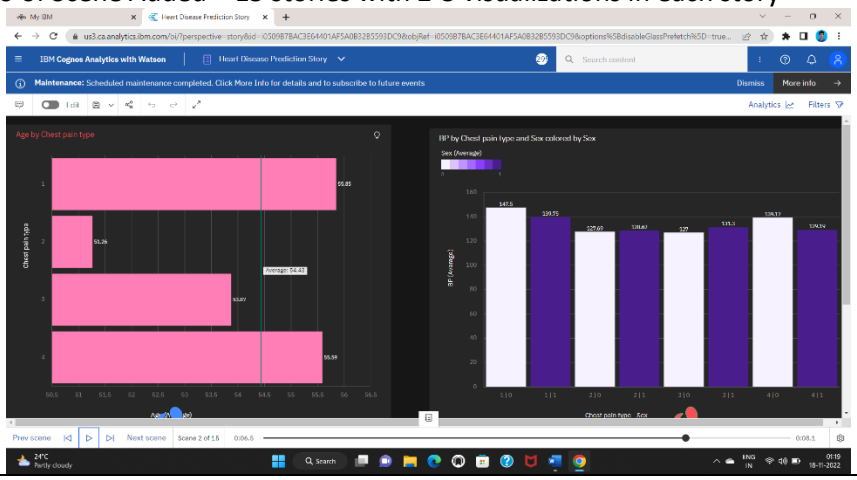
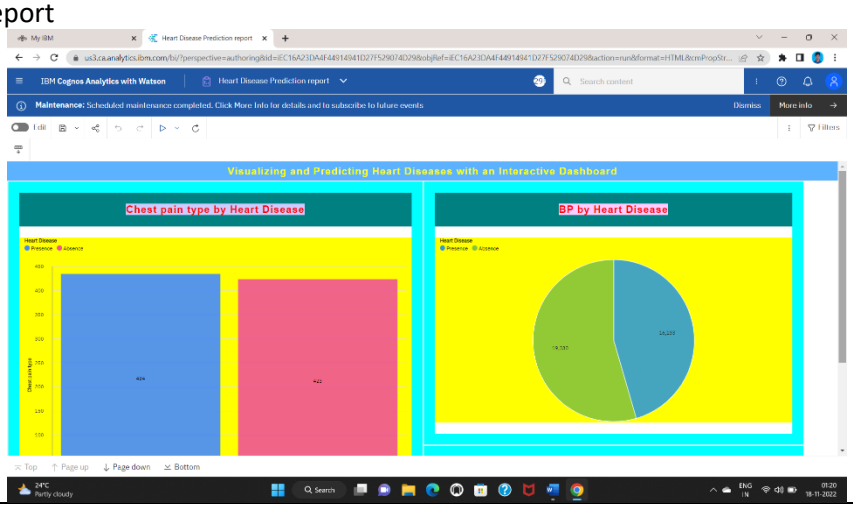
Test Cases Report:

Date	13-Nov-22
Team ID	PPNT2022TMID40507
Project Name	Project - Visualizing and Predicting Heart Diseases with an Interactive Dashboard
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
BP	dashboard/report, story	Cognos Analytics	Verify the dataset for accurate performance	A quality dataset	1.Upload the dataset 2.Explore the data 3.Create dashboard/Report, Story	https://github.com/IBM-EPBL/IBM-Project-45585-1660731143/blob/main/Final%20Deliverable/Datasets/Heart_Disease_Prediction.csv	Accurate Prediction	Working as expected	Pass	Cognos analytics to accurate predict of patients Bp	yes	high	SATHISHKUMAR P
Cholesterol	Dashboard/report, report	Cognos Analytics	Verify the dataset for accurate performance	A quality dataset	1.Upload the dataset 2.Explore the data 3.Create dashboard/Report, Story	https://github.com/IBM-EPBL/IBM-Project-45585-1660731143/blob/main/Final%20Deliverable/Datasets/Heart_Disease_Prediction.csv	Accurate Prediction	Working as expected	pass	Cognos analytics to accurate predict of patients Cholesterol	yes	high	RUKESH K
Thallium	Dashboard/report, Story	Cognos Analytics	Verify the dataset for accurate performance	A quality dataset	1.Upload the dataset 2.Explore the data 3.Create dashboard/Report, Story	https://github.com/IBM-EPBL/IBM-Project-45585-1660731143/blob/main/Final%20Deliverable/Datasets/Heart_Disease_Prediction.csv	Accurate Prediction	Not Working as expected	fail	some data not accuracy	no	low	SIVASANKAR B
ECG	Dashboard/report, Story	Cognos Analytics	Verify the dataset for accurate performance	A quality dataset	1.Upload the dataset 2.Explore the data 3.Create dashboard/Report, Story	https://github.com/IBM-EPBL/IBM-Project-45585-1660731143/blob/main/Final%20Deliverable/Datasets/Heart_Disease_Prediction.csv	Accurate Prediction	Working as expected	pass	Cognos analytics to accurate predict of patients Bp	yes	high	MD NOORULLAH A
Obesity	Dashboard/report, Story	Cognos Analytics	Verify the dataset for accurate performance	A quality dataset	1.Upload the dataset 2.Explore the data 3.Create dashboard/Report, Story	https://github.com/IBM-EPBL/IBM-Project-45585-1660731143/blob/main/Final%20Deliverable/Datasets/Heart_Disease_Prediction.csv	Accurate Prediction	Not Working as expected	fail		no	medium	SATHISHKUMAR P
ST Depression	Dashboard/report, Story	Cognos Analytics	Verify the dataset for accurate performance	A quality dataset	1.Upload the dataset 2.Explore the data 3.Create dashboard/Report, Story	https://github.com/IBM-EPBL/IBM-Project-45585-1660731143/blob/main/Final%20Deliverable/Datasets/Heart_Disease_Prediction.csv	Accurate Prediction	Working as expected	pass		yes	high	RUKESH K

8.2 Acceptance Testing

S.No.	Parameter	Screenshot / Values
1.	Dashboard designs	No of Visualizations / Graphs – 8 dashboard tabs with 7 - 23 visualizations in each dashboard
2.	Data Responsiveness	It hides certain aspects of the visualization if the size is limited, to maximize the space that is available to display data. • Its Create with relationship with another explorations • There was another data exploration with various continuous vales , those values was grouped as common.
3.	Amount Data to Rendered (DB2 Metrics)	There are one relevant datasets are uploaded in the IBM DB2.

		
4.	Utilization of Data Filters	IN Cognos Dashboard utilization of the filtration to be filtered of all explorations of the dashboard
5.	Effective User Story	No of Scene Added – 15 stories with 2-3 visualizations in each story 
6.	Descriptive Reports	No of Visualizations / Graphs – 8 reports with 3 – 5 visualization in each report 

8.3 Testing and Validations

Validation is a complex process with many possible variations and options, so specifics vary from database to database, but the general outline is:

- **Requirement Gathering**

o The Sponsor decides what the database is required to do based on regulations, company needs, and any other important factors.

o The requirements are documented and approved.

- **System Testing**

o Procedures to test the requirements are created and documented.

o The version of the database that will be used for validation is set up.

o The Sponsor approves the test procedures.

o The tests are performed and documented.

o Any needed changes are made. This may require another, shorter round of testing and documentation.

- **System Release**

o The validation documentation is finalized.

The database is put into production.

8.4 Testing Levels

8.4.1 Functional Testing:

This type of testing is done against the functional requirements of the project.

Types:

Unit testing: Each unit /module of the project is individually tested to check for bugs. If any bugs found by the testing team, it is reported to the developer for fixing.

Integration testing: All the units are now integrated as one single unit and checked for bugs. This also checks if all the modules are working properly with each other.

System testing: This testing checks for operating system compatibility. It includes both functional and non functional requirements.

Sanity testing: It ensures change in the code doesn't affect the working of the project.

Smoke testing: this type of testing is a set of small tests designed for each build.

Interface testing: Testing of the interface and its proper functioning.

Regression testing: Testing the software repetitively when a new requirement is added, when bug fixed etc. Beta/Acceptance testing: User level testing to obtain user feedback on the product.

8.4.2 Non-Functional Testing:

This type of testing is mainly concerned with the non-functional requirements such as performance of the system under various scenarios.

Performance testing: Checks for speed, stability and reliability of the software, hardware or even the network of the system under test.

Compatibility testing: This type of testing checks for compatibility of the system with different operating systems, different networks etc.

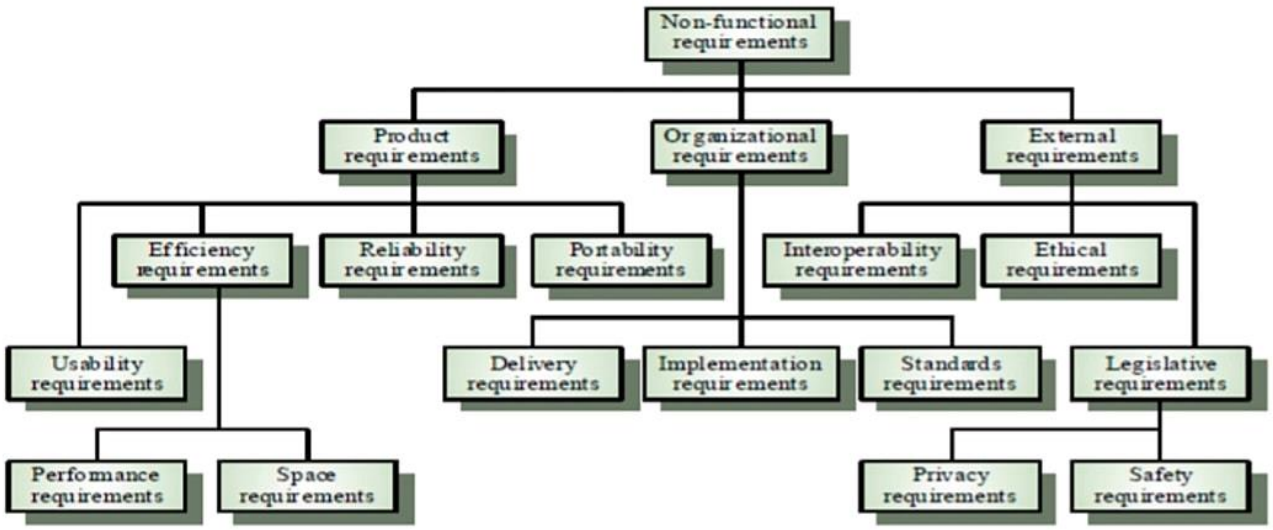
Localization testing: This checks for the localized version of the product mainly concerned with UI.

Security testing: Checks if the software has vulnerabilities and if any, fix them.

Reliability testing: Checks for the reliability of the software

Stress testing: This testing checks the performance of the system when it is exposed to different stress levels. Usability testing: Type of testing checks the easily the software is being used by the customers.

Compliance testing: Type of testing to determine the compliance of a system with internal or external standards.



Reliability

The structure must be reliable and strong in giving the functionalities. The movements must be made unmistakable by the structure when a customer has revealed a couple of enhancements. The progressions made by the Programmer must be Project pioneer and in addition the Test designer.

Maintainability

The system watching and upkeep should be fundamental and focus in its approach. There should not be an excess of occupations running on diverse machines such that it gets hard

to screen whether the employments are running without lapses.

Performance

The framework will be utilized by numerous representatives all the while. Since the system will be encouraged on a single web server with a lone database server outside of anyone's ability to see, execution transforms into a significant concern. The structure should not capitulate when various customers would use everything the while. It should allow brisk accessibility to each and every piece of its customers. For instance, if two test specialists are all the while attempting to report the vicinity of a bug, then there ought not to be any irregularity at the same time.

Portability

The framework should to be effectively versatile to another framework. This is obliged when the web server, which s facilitating the framework gets adhered because of a few issues, which requires the framework to be taken to another framework.

Scalability

The framework should be sufficiently adaptable to include new functionalities at a later stage. There should be a run of the mill channel, which can oblige the new functionalities.

Flexibility

Flexibility is the capacity of a framework to adjust to changing situations and circumstances, and to adapt to changes to business approaches and rules. An adaptable framework is one that is anything but difficult to reconfigure.

8.5 White Box Testing

White Box Testing is defined as the testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers.

It is one of two parts of the "Box Testing" approach to software testing. Its counterpart, Blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing.

The term "WhiteBox" was used because of the see-through box concept. The clear box or WhiteBox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "Black Box Testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested.

Internal security holes

Broken or poorly structured paths in the coding processes

The flow of specific inputs through the code

Expected output

The functionality of conditional loops

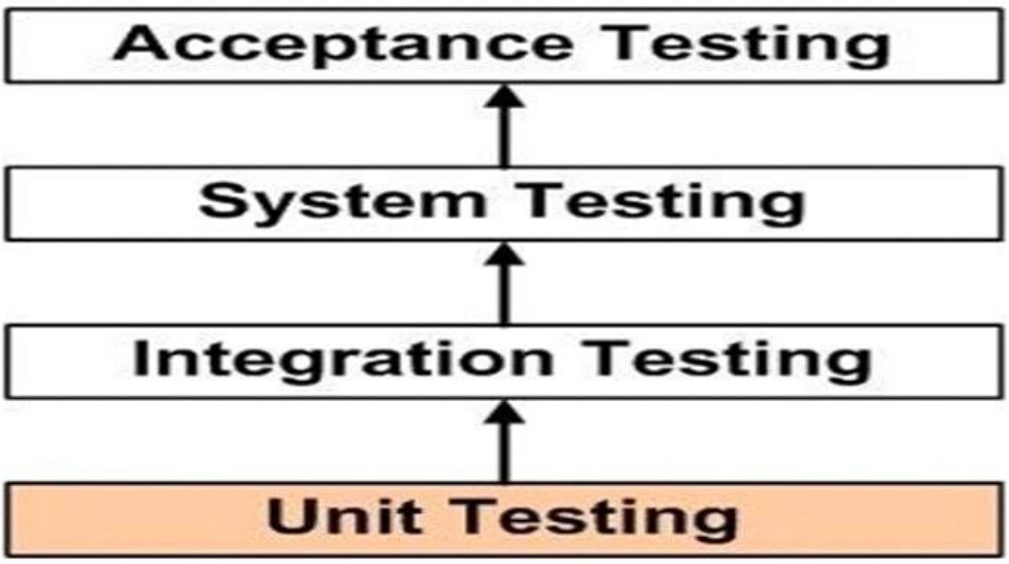
Testing of each statement, object, and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

8.6 Different Stages of Testing

8.6.1 Unit Testing

UNIT TESTING is a level of software testing where individual units/ components of software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.



Unit Test Plan:

1) Unit Test Plan
o Prepare

o Review

o Rework

o Baseline

2)Unit Test Cases/Scripts

o Prepare

o Review

o Rework

o Baseline

 Unit Test

o Perform

Benefits

- Unit testing increases confidence in changing/ maintaining code. If good unit tests are written and if they are run every time any code is changed, we will be able to promptly catch any defects introduced due to the change. Also, if codes are already made less interdependent to make unit testing possible,the unintended impact of changes to any code is less.
- Codes are more reusable. In order to make unit testing possible, codes need to be modular. This means that codes are easier to reuse.
- Development is faster. How? If you do not have unit testing in place, you write your code and perform that fuzzy 'developer test' (You set some breakpoints, fire up the GUI, provide a few inputs that hopefully hit your code and hope that you are all set.) But, if you have unit testing in place, you write the test, write the code and run the test. Writing tests takes time but the time is compensated by the less amount of time it takes to run the tests;

You need not fire up the GUI and provide all those inputs. And, of course, unit tests are more reliable than 'developer tests'. Development is faster in the long run too. How? The effort required to find and fix defects found during unit testing is very less in comparison to the effort required to fix defects found during system testing or acceptance testing.

- The cost of fixing a defect detected during unit testing is lesser in comparison to that of defects detected at higher levels. Compare the cost (time, effort, destruction, humiliation) of a defect detected during acceptance testing or when the software is live.

 Debugging is easy. When a test fails, only the latest changes need to be debugged. With testing at higher levels, changes made over the span of several days/weeks/months need to be scanned.

8.6.2 Integration Testing

INTEGRATION TESTING is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

Integration testing: Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. See also component integration testing, system integration testing.

Component integration testing: Testing performed to expose defects in the interfaces and interaction between integrated components.

System integration testing: Testing the integration of systems and packages; testing interfaces to external organizations (e.g. Electronic Data Interchange,Internet).

Tasks

Integration Test Plan

o Prepare

o Review

o Rework

o Baseline

Integration Test Cases/Scripts

o Prepare

o Review

o Rework

o Baseline

Integration Test

8.6.3 System Testing

SYSTEM TESTING is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements.

system testing: The process of testing an integrated system to verify that it meets specified requirements.

8.6.4 Acceptance Testing

ACCEPTANCE TESTING is a level of software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.

acceptance testing: Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

TESTING HEART DISEASE PRIDICTION:

Testing is the process used to help identify the correctness, completeness, security and quality of the developed computer software. Testing is the process of technical investigation and includes the process of executing a program or application with the intent of finding errors.

In the training process, our model learns to associate a particular input (i.e. features) to the corresponding output (tag) based on the test samples used for training. Input features and tags (e.g. 1-normal 2-heart disease) are fed into the machine learning algorithm to generate a model.

A comparative analysis of different classifiers was performed for the classification of the Heart Disease dataset in order to correctly classify and predict Heart Disease cases with minimal attributes.

Input	Expected Output	Actual Output
Data Visualization	Various visual representations of the data to understand more about the relationship between various features.	Pass
Data Processing	Convert some categorical variables into dummyvariables and scaleall the valuesbefore training theMachine Learning models.	Pass
Dataset	Split the datasetinto training and testing datasets.	Pass
Training dataset	Train the model using the training dataset.	Pass
Testing dataset	Tests if the model is accurate basedon the outputof the testing dataset.	Pass

Training and subsequent testing

Input	Expected Output	Actual Output
No Heart Disease	Should be labeledas 1 (no heart disease) and should showoutput as “The patientis not likely to have heart disease”.	Pass
Heart Disease	Should be labeled as 2 (heart disease) and should showoutput as “The patient is likely to have heart disease”.	Pass

Heart DiseaseTest

8.7 Model Evaluation

The most important evaluation metrics for this problem domain are Accuracy,Sensitivity, Specificity, Precision, F1-measure, Log Loss, ROC and Mathew correlation coefficient.

- **Accuracy:** which refers to how close a measurement is to the true value and can be calculated using the following formula:

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

$$\mathrm{MCC} = \frac{\mathrm{TP} \cdot \mathrm{TN} - \mathrm{FP} \cdot \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP}) \cdot (\mathrm{TP} + \mathrm{FN}) \cdot (\mathrm{TN} + \mathrm{FP}) \cdot (\mathrm{TN} + \mathrm{FN})}}$$

(worst value: −1; best value: +1)

- **Precision:** which is how consistent results are when measurements are repeated and can be calculated using the following formula:

Precision = True Positive / (True Positive + False Positive)

- **Sensitivity:**

Sensitivity is a measure of the proportion of actual positivecases that got predicted as positive (or true positive). Sensitivity is also termed as Recall.

Sensitivity = True Positive / (True Positive + False Negative)

- **Specificity:**

Specificity is defined as the proportion of actual negatives, which got predicted asthe negative (or true negative).

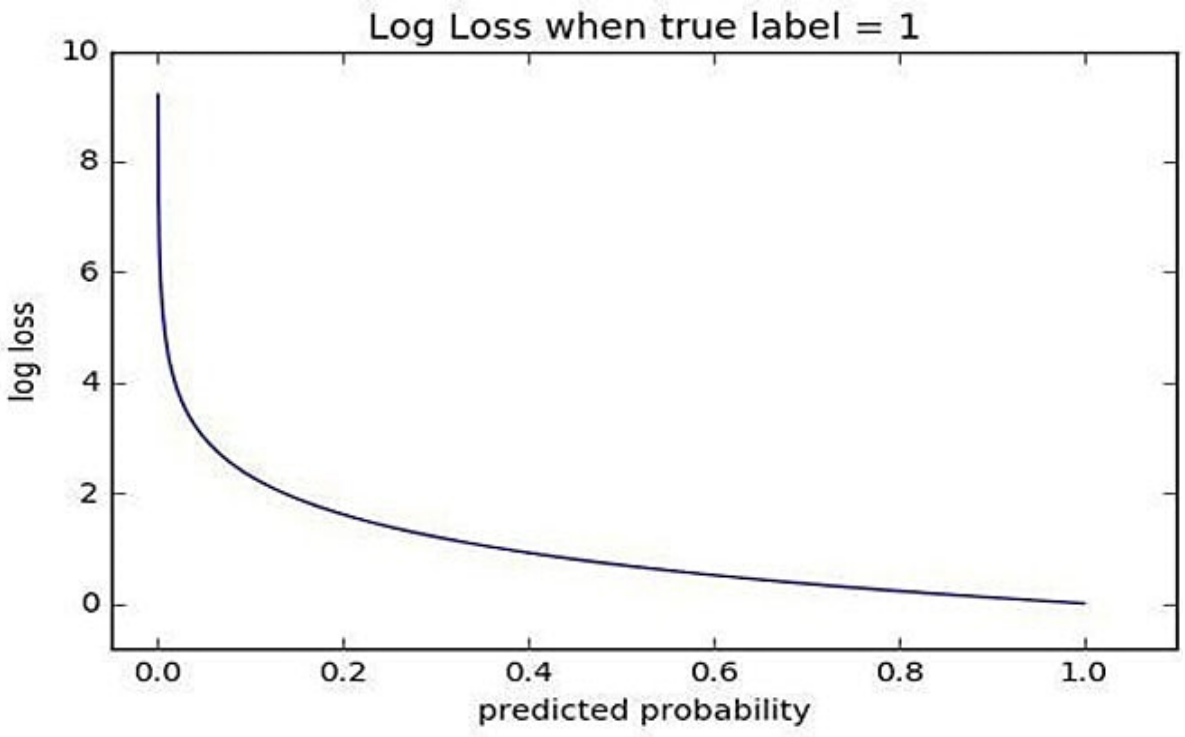
Specificity = True Negative / (True Negative + False Positive)

- **Mathew Correlation coefficient (MCC):**

The Matthews correlation coefficient (MCC), instead, is a more reliable statisticalrate which produces a high score only if the prediction obtained good results in all ofthe four confusion matrix categories (true positives, false negatives, true negatives, and false positives), proportionally both to the size of positive elements and the size of negative elements in the dataset.

- **Logic loss**

Logarithmic loss measures the performance of a classification model where the prediction input is a probability value between 0 and 1. The goal of our machine learningmodels is to minimize this value. A perfect modelwould have a log loss of 0. Log loss increases as the predicted probability diverges from the actual label. So, predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high log loss.



Log Loss Graph

- **F1 Score**

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positivesand false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 score is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost.

F1 Score = 2(Recall Precision) / (Recall + Precision)

- **ROC Curve**

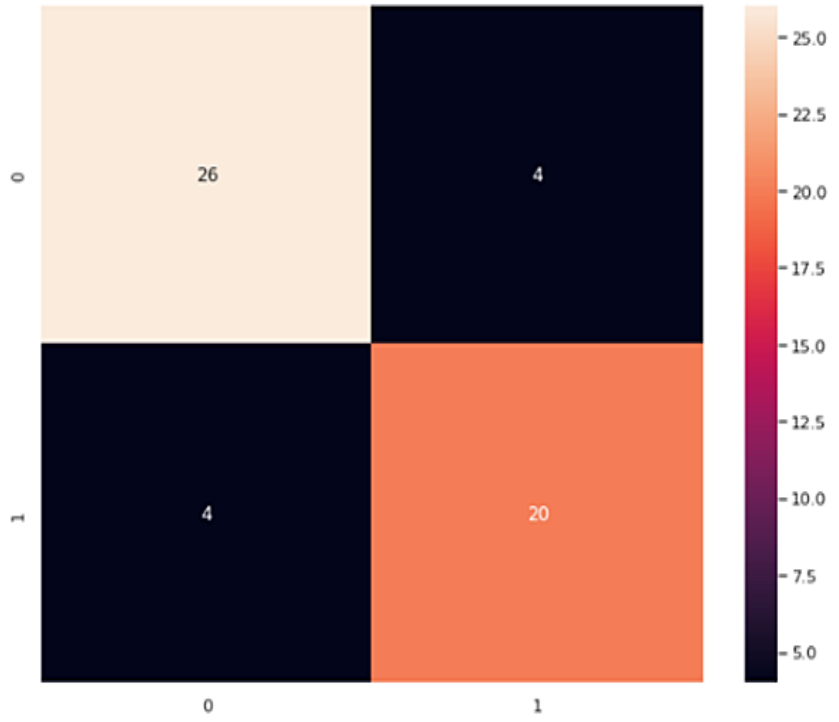
  An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate & False Positive Rate.

8.7.1 Random Forest Classifier

Code:

```
y_pred_rfe = rf_classifier.predict(X_test)

plt.figure(figsize=(10, 8)) CM=confusion_matrix(Y_test,y_pred_rfe) sns.heatmap(CM, annot=True)

TN = CM[0][0]

FN = CM[1][0]

TP = CM[1][1]

FP = CM[0][1]

specificity = TN/(TN+FP)

loss_log = log_loss(Y_test, y_pred_rfe) acc= accuracy_score(Y_test, y_pred_rfe) roc=roc_auc_score(Y_test, y_pred_rfe)

prec = precision_score(Y_test, y_pred_rfe) rec = recall_score(Y_test, y_pred_rfe)

f1 = f1_score(Y_test, y_pred_rfe)

mathew = matthews_corrcoef(Y_test, y_pred_rfe)

model_results =pd.DataFrame([['Random Forest',acc, prec,rec,specificity, f1,roc, loss_log,mathew]],

columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity', 'F1 Score','ROC','Log_Loss','mathew_corrcoef'])

model_results
```

	Model	Accuracy	Precision	Sensitivity	Specificity	F1 Score	ROC	Log_Loss	mathew_corrcoef
0	Random Forest	0.851852	0.866667	0.866667	0.866667	0.866667	0.85	19.188653	0.7



Random ForestConfusion Matrix

Y_pred_rf = np.around(Y_pred_rf)
print(metrics.classification_report(Y_test,Y_pred_rf))

```
Y_pred_rf = np.around(Y_pred_rf)
print(metrics.classification_report(Y_test,Y_pred_rf))
```

```
              precision    recall  f1-score   support

           1       0.87      0.87      0.87        30
           2       0.83      0.83      0.83        24

    accuracy                           0.85        54
   macro avg       0.85      0.85      0.85        54
weighted avg       0.85      0.85      0.85        54
```

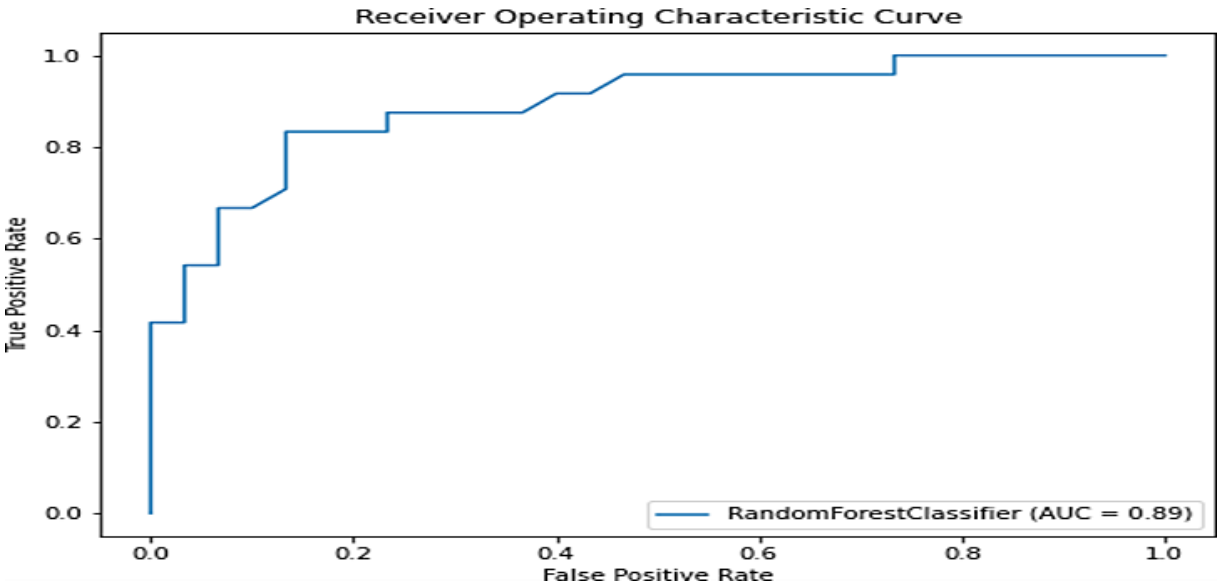
Random Forest Classification Repor

plot_roc_curve(rf_classifier,X_test,Y_test)

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic Curve')



Random ForestROC Curve

8.7.2 K-Nearest Neighbors Classifier

y_pred_knne = knn_classifier.predict(X_test)

plt.figure(figsize=(10, 8)) CM=confusion_matrix(Y_test,y_pred_knne) sns.heatmap(CM, annot=True)

TN = CM[0][0]

FN = CM[1][0]

TP = CM[1][1]

FP = CM[0][1]

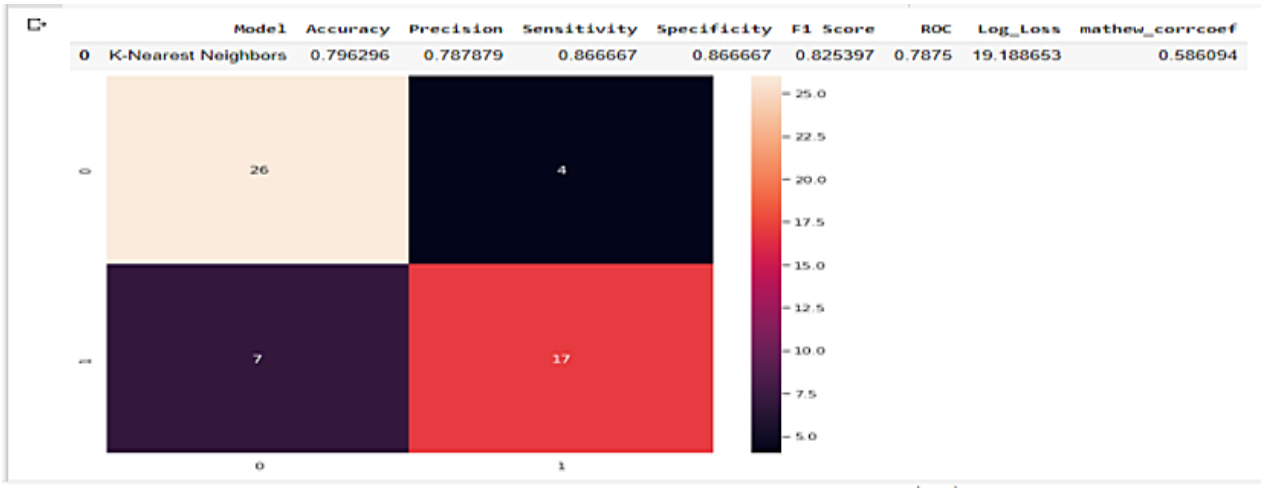
specificity = TN/(TN+FP)

loss_log = log_loss(Y_test, y_pred_knne) acc= accuracy_score(Y_test, y_pred_knne) roc=roc_auc_score(Y_test, y_pred_knne)

prec = precision_score(Y_test, y_pred_knne) rec = recall_score(Y_test, y_pred_knne)

f1 = f1_score(Y_test, y_pred_knne)

```
mathew = matthews_corrcoef(Y_test, y_pred_knne)

model_results =pd.DataFrame([['K-Nearest Neighbors ',acc, prec,rec,specificity, f1,roc, loss_log,mathew]],

columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity', 'F1 Score','ROC','Log_Loss','mathew_corrcoef'])

model_results
```



	Model	Accuracy	Precision	Sensitivity	Specificity	F1 Score	ROC	Log_Loss	mathew_corrcoef
0	K-Nearest Neighbors	0.796296	0.787879	0.866667	0.866667	0.825397	0.7875	19.188653	0.586094

K-Nearest Neighbors Confusion Matrix

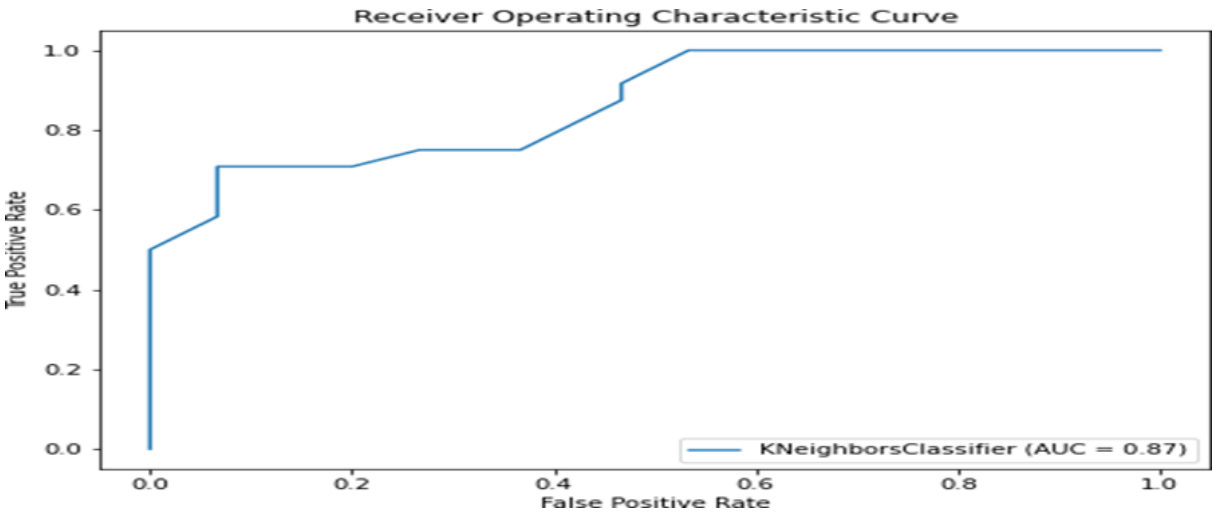
```
Y_pred_knn = np.around(Y_pred_knn)

print(metrics.classification_report(Y_test,Y_pred_knn))
```

```
Y_pred_knn = np.around(Y_pred_knn)
print(metrics.classification_report(Y_test,Y_pred_knn))
```

```
              precision    recall  f1-score   support

           1       0.79      0.87      0.83        30
           2       0.81      0.71      0.76        24

    accuracy                           0.80        54
   macro avg       0.80      0.79      0.79        54
weighted avg       0.80      0.80      0.79        54
```

K-Nearest Neighbors Classification Report

plot_roc_curve(knn_classifier,X_test,Y_test)plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic Curve')



K-Nearest Neighbors ROC Curve

8.7.3 Decision Tree Classifier

y_pred_dte = dt_classifier.predict(X_test)

plt.figure(figsize=(10, 8)) CM=confusion_matrix(Y_test,y_pred_dte) sns.heatmap(CM, annot=True)

TN = CM[0][0]

FN = CM[1][0]

TP = CM[1][1]

FP = CM[0][1]

specificity = TN/(TN+FP)

loss_log = log_loss(Y_test, y_pred_dte) acc= accuracy_score(Y_test, y_pred_dte) roc=roc_auc_score(Y_test, y_pred_dte)

prec = precision_score(Y_test, y_pred_dte) rec = recall_score(Y_test, y_pred_dte)

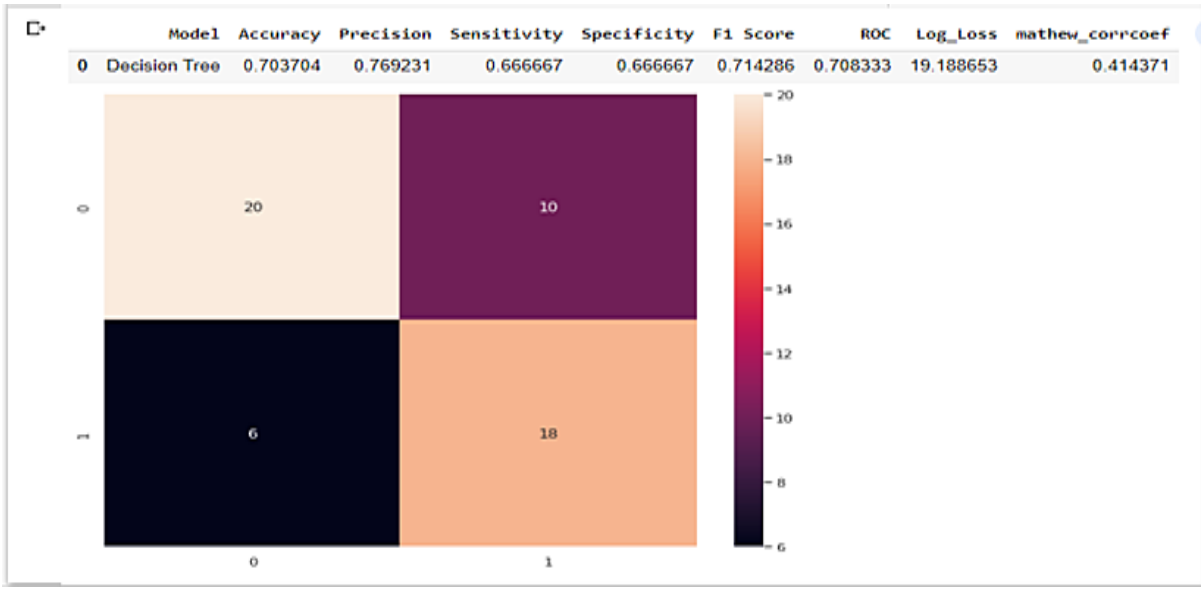
f1 = f1_score(Y_test, y_pred_dte)

mathew = matthews_corrcoef(Y_test, y_pred_dte)

model_results =pd.DataFrame([['Decision Tree',acc, prec,rec,specificity, f1,roc, loss_log,mathew]],

columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity',
'F1 Score','ROC','Log_Loss','mathew_corrcoef'])

model_results

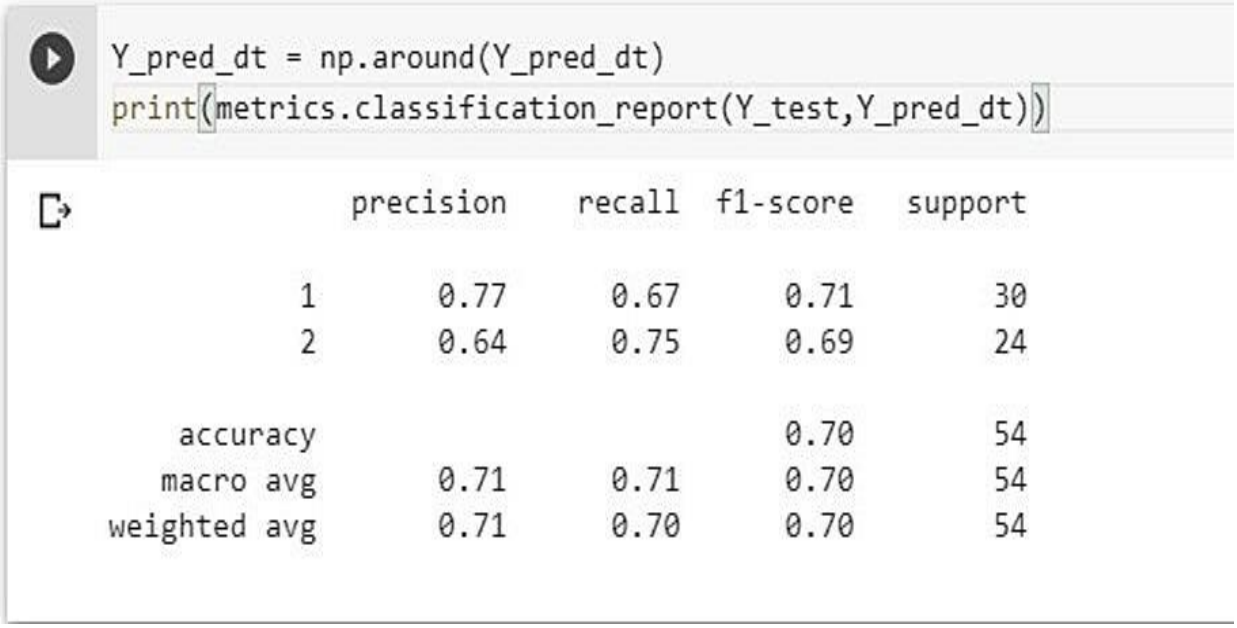


	Model	Accuracy	Precision	Sensitivity	Specificity	F1 Score	ROC	Log_Loss	mathew_corrcoef
0	Decision Tree	0.703704	0.769231	0.666667	0.666667	0.714286	0.708333	19.188653	0.414371

Decision Tree Confusion Matrix

Y_pred_dt = np.around(Y_pred_dt)

print(metrics.classification_report(Y_test,Y_pred_dt))



```
Y_pred_dt = np.around(Y_pred_dt)
print(metrics.classification_report(Y_test,Y_pred_dt))
```

	precision	recall	f1-score	support
1	0.77	0.67	0.71	30
2	0.64	0.75	0.69	24
accuracy			0.70	54
macro avg	0.71	0.71	0.70	54
weighted avg	0.71	0.70	0.70	54

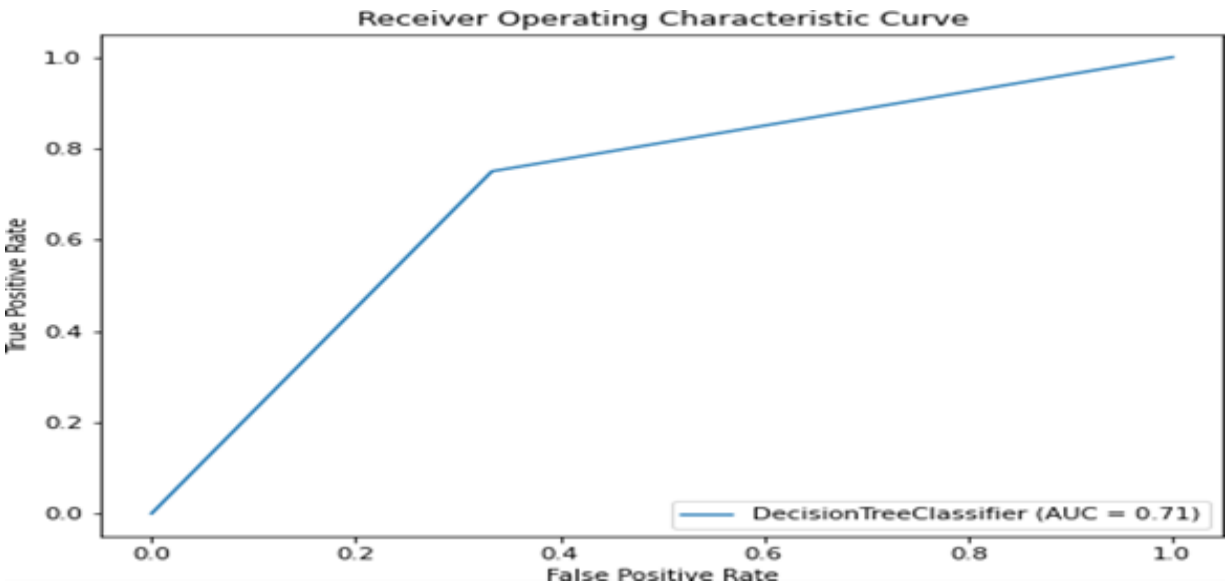
Decision Tree Classification Report

plot_roc_curve(dt_classifier,X_test,Y_test)

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic Curve')



Decision Tree ROC Curve

8.7.4 Naive Bayes Classifier

y_pred_nbe = nb_classifier.predict(X_test)

plt.figure(figsize=(10, 8)) CM=confusion_matrix(Y_test,y_pred_nbe) sns.heatmap(CM, annot=True)

TN = CM[0][0]

FN = CM[1][0]

TP = CM[1][1]

FP = CM[0][1]

specificity = TN/(TN+FP)

loss_log = log_loss(Y_test, y_pred_nbe) acc= accuracy_score(Y_test, y_pred_nbe) roc=roc_auc_score(Y_test, y_pred_nbe)

prec = precision_score(Y_test, y_pred_nbe) rec = recall_score(Y_test, y_pred_nbe)

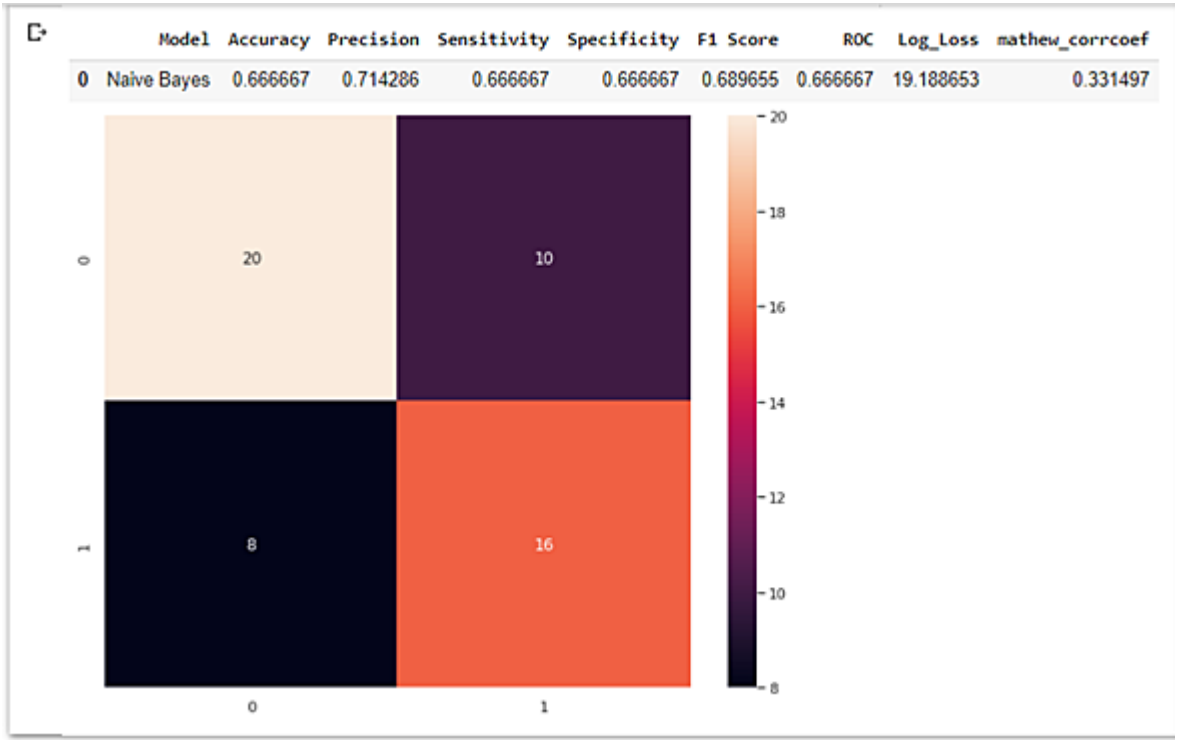
f1 = f1_score(Y_test, y_pred_nbe)

mathew = matthews_corrcoef(Y_test, y_pred_nbe)

model_results =pd.DataFrame([['Naive Bayes ',acc, prec,rec,specificity, f1,roc, loss_log,mathew]],

columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity', 'F1 Score','ROC','Log_Loss','mathew_corrcoef'])

model_results



	Model	Accuracy	Precision	Sensitivity	Specificity	F1 Score	ROC	Log_Loss	mathew_corrcoef
0	Naive Bayes	0.666667	0.714286	0.666667	0.666667	0.689655	0.666667	19.188653	0.331497

Naive BayesConfusion Matrix

Y_pred_nb = np.around(Y_pred_nb)

print(metrics.classification_report(Y_test,Y_pred_nb))

```
Y_pred_nb = np.around(Y_pred_nb)
print(metrics.classification_report(Y_test,Y_pred_nb))
```

```
              precision    recall  f1-score   support

           1       0.71      0.67      0.69        30
           2       0.62      0.67      0.64        24

    accuracy                           0.67        54
   macro avg       0.66      0.67      0.66        54
weighted avg       0.67      0.67      0.67        54
```

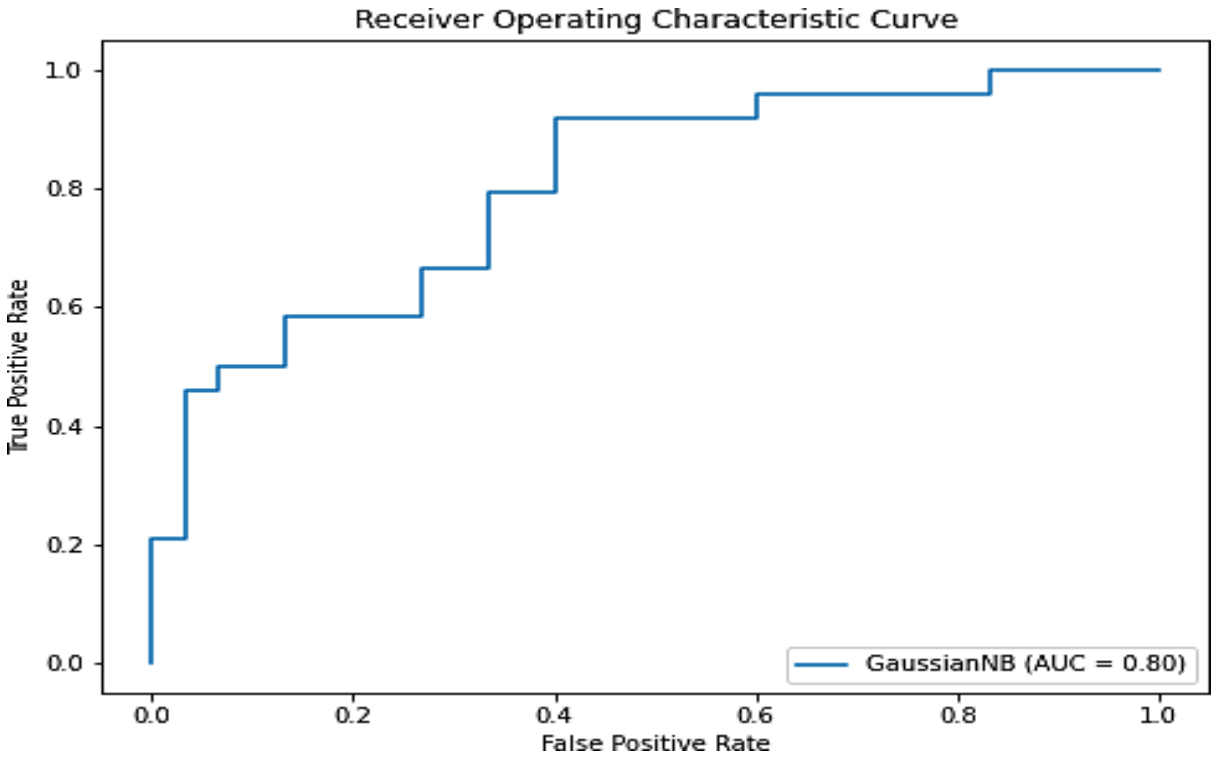
Naive BayesClassification Report

plot_roc_curve(nb_classifier,X_test,Y_test)

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic Curve')



Naive Bayes ROC Curve

RESULTS

CHAPTER-9

RESULTS

9.1 Performance Metrics

Final Result

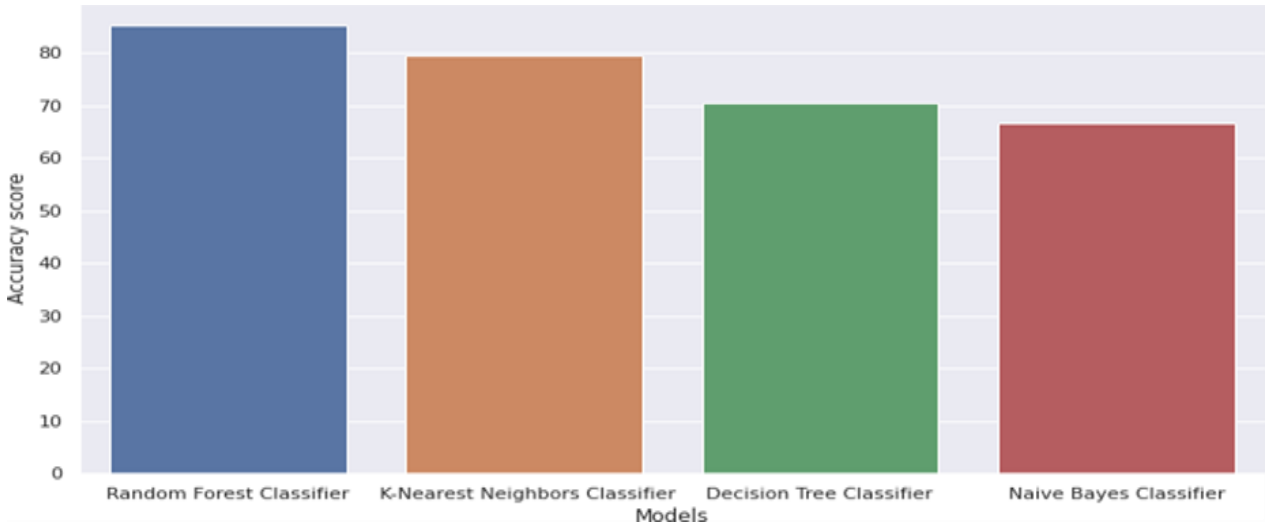
Model	Accuracy	Precision	Sensitivity	Specificity	F1 Score	ROC	Log_Loss	Mathew_ correcoef
Random Forest	0.8519	0.8667	0.8667	0.8667	0.8667	0.85	19.1886	0.7
KNN	0.7963	0.7879	0.8667	0.8667	0.8254	0.7875	19.1886	0.5861
Decision Tree	0.7037	0.7692	0.6667	0.6667	0.7142	0.7083	19.1886	0.4144
Naive Bayes	0.6667	0.7143	0.6667	0.6667	0.6896	0.6667	19.1886	0.3315

Final Accuracy Score

```
scores = [score_rf,score_knn,score_dt,score_nb]
Models = ["Random Forest Classifier"," K-Nearest Neighbors Classifier",
          "Decision Tree Classifier","Naive Bayes Classifier"]

for i in range(len(Models)):
    print("The accuracy score achieved using "+Models[i]+" is: "+str(scores[i])+" %")
```

```
The accuracy score achieved using Random Forest Classifier is: 85.19 %
The accuracy score achieved using  K-Nearest Neighbors Classifier is: 79.63 %
The accuracy score achieved using Decision Tree Classifier is: 70.37 %
The accuracy score achieved using Naive Bayes Classifier is: 66.67 %
```

Accuracy ScoreBar Graph



Snapshots:

Sample test-1

```
Input = (67, 0, 3, 115, 564, 0, 2, 160, 0, 1.6, 2, 0, 7)

Input_array= np.asarray(Input)
Input_reshaped = Input_array.reshape(1,-1)

prediction = rf_classifier.predict(Input_reshaped)
prediction = np.around(prediction)

print(prediction)

if (prediction[0]== 1):
  print('The Person does not have a Heart Disease')
else:
  print("The Person is likely to have Heart Disease by %f "%(prediction))
```

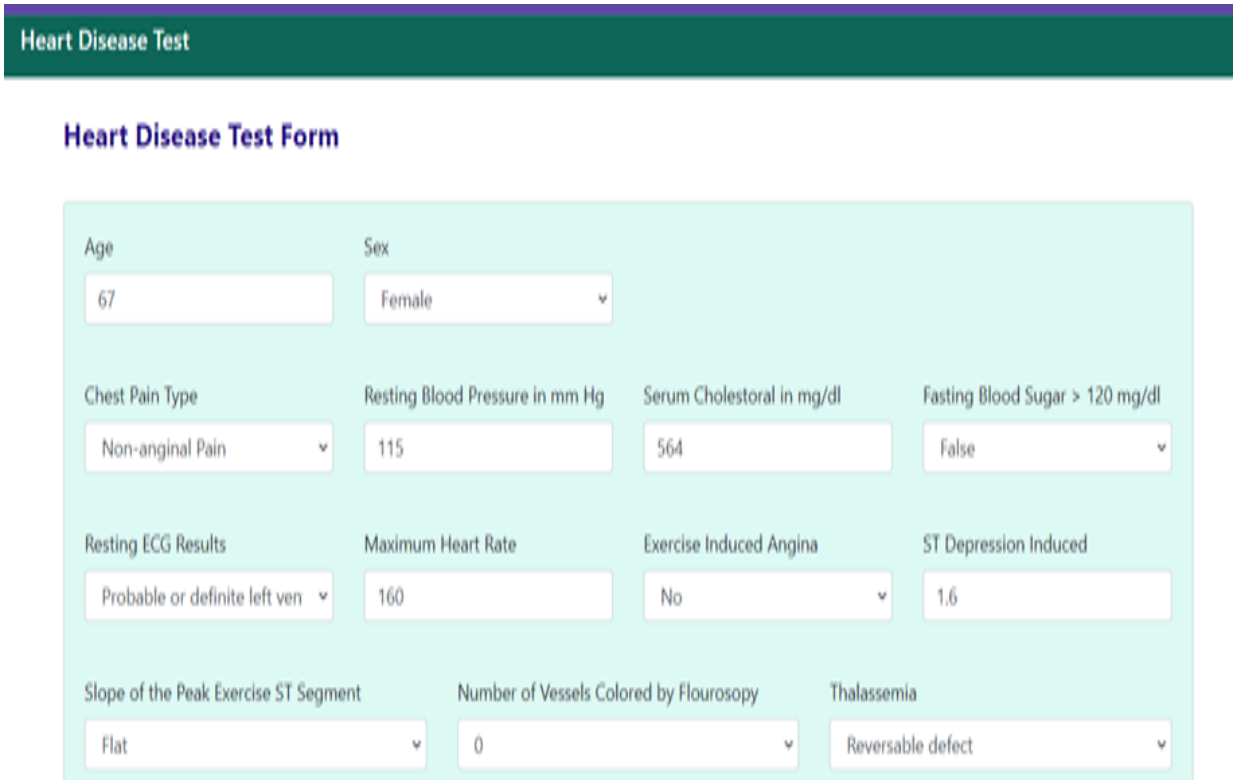
```
[1]
The Person does not have a Heart Disease
```

Sample test-2

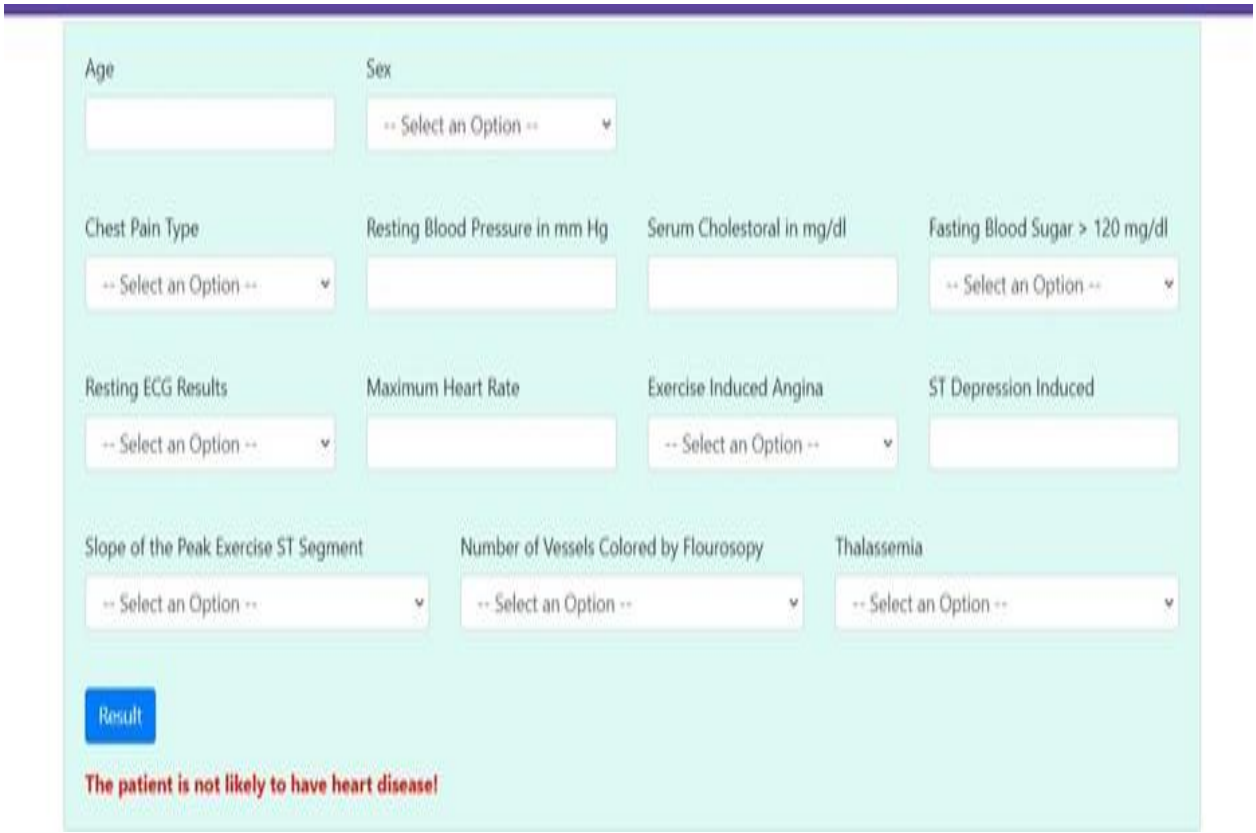
```
Input = (70,1,4,130,322,0,2,109,0,2.4,2,3,3)

Input_array= np.asarray(Input)
Input_reshaped = Input_array.reshape(1,-1)

prediction = rf_classifier.predict(Input_reshaped)
prediction = np.around(prediction)

print(prediction)

if (prediction[0]== 1):
  print('The Person does not have a Heart Disease')
else:
  print("The Person is likely to have Heart Disease by %f "%(prediction))
```

```
[2]
The Person is likely to have Heart Disease by 2.000000
```

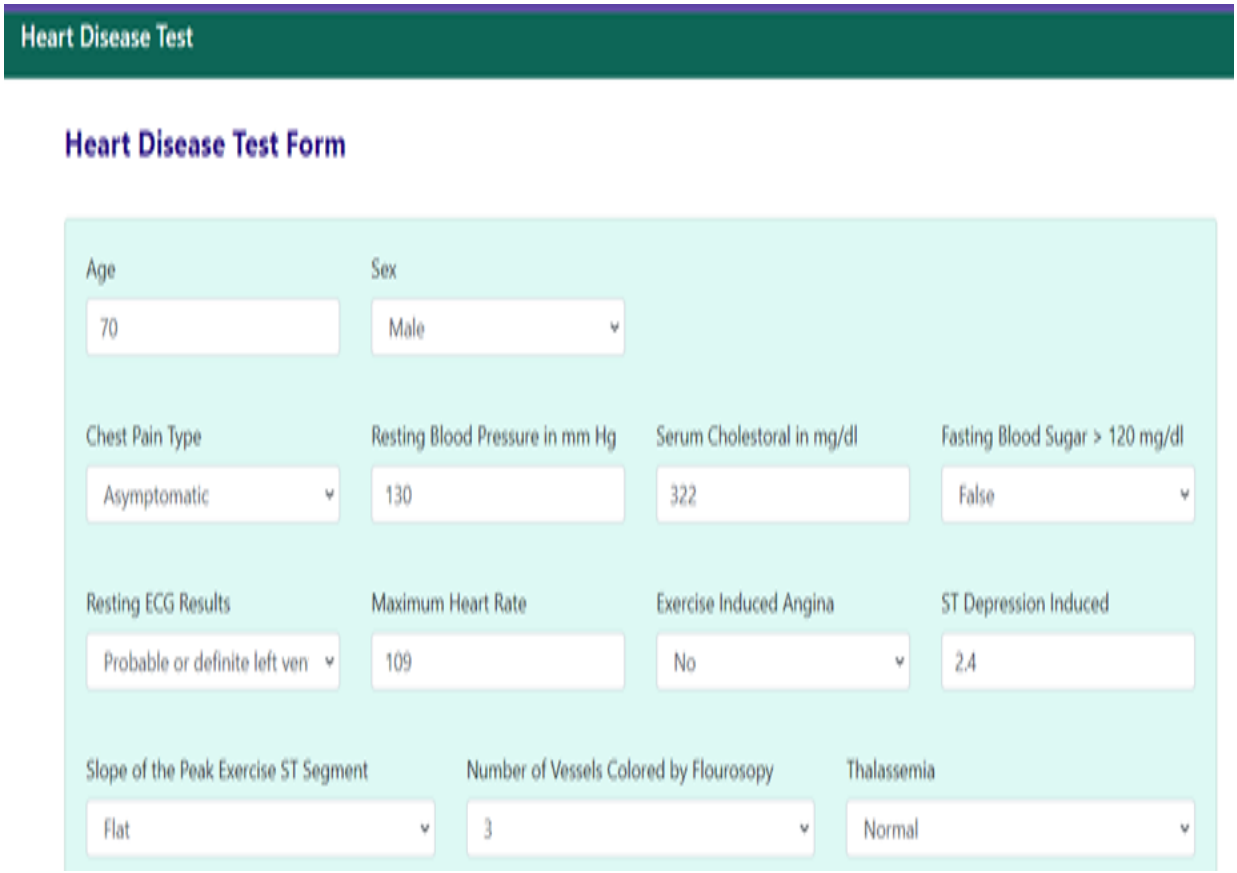
Heart DiseaseTest 1



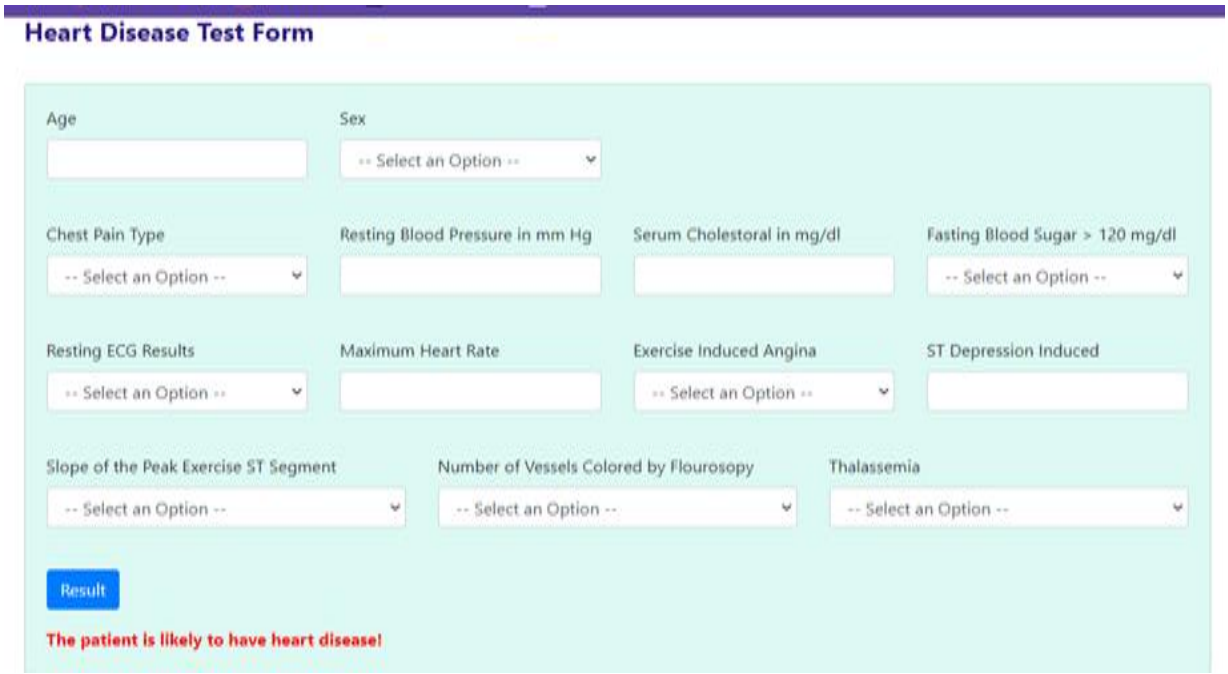
Heart DiseaseTest Result 1



Heart DiseaseTest 2



Heart DiseaseTest Result 2



CHAPTER-10

ADVANTAGES & DISADVANTAGES

10.1 ADVANTAGES

1. User can search for doctor's help at any point of time.
2. User can talk about their heart disease and get instant diagnosis.
3. Doctors get more clients online.
4. Very useful in case of emergency.

10.2 DISADVANTAGES

- Prediction of cardiovascular disease results is not accurate.
- Data mining techniques does not help to provide effective decision making.
- Cannot handle enormous datasets for patient records.
- Disadvantages
- Accuracy Issues: A computerized system alone does not ensure accuracy, and the warehouse data is only as good as the data entry that created it.
- The system is not fully automated, it needs data from user for full diagnosis.

CHAPTER-11

CONCLUSION

11.1 CONCLUSION

Heart diseases are a major killer in India and throughout the world, application of promising technology like machine learning to the initial prediction of heart diseases will have a profound impact on society. The early prognosis of heart disease can aid in making decisions on lifestyle changes in high-risk patients and in turn reduce the complications, which can be a great milestone in the field of medicine. The number of people facing heart diseases is on a raise each year. This prompts for its early diagnosis and treatment. The utilization of suitable technology support in this regard can prove to be highly beneficial to the medical fraternity and patients.

CHAPTER-12

FUTURE SCOPE

12.1 FUTURE SCOPE

Future enhance of the HDPS is to predict a specific HD type such Heart attracts, CVD, CAD, etc. the potential of the HDPS in a different area are hospital, Clinic, smartphone, smart wear, hospital/police emergency system and integrate with fitness mobile application. We will integrate this model in hospital and clinic system to predict heart disease. We will implement this HDP Model into smart wears to detect essential attributes of HD and suggest to the precaution of HD. we will also apply this model into a mobile app to easily test ourselves HD. we will integrate smart wear to the hospital and police emergency system to save the life of the patient at the emergency condition.

CHAPTER-13

APPENDIX

14. APPENDIX

Python:

Python is an interpreted, high-level, general purpose programming language created by Guido Van Rossum and first released in 1991, Python's design philosophy emphasizes code Readability with its notable use of significant White space. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

Sklearn:

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Numpy:

NumPy is a library for the python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim with contributions from several other developers. In 2005, Travis created NumPy by

incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

Librosa:

Librosa is a Python package for music and audio analysis. Librosa is basically used when we work with audio data like in music generation (using LSTMs), Automatic Speech Recognition.

It provides the building blocks necessary to create the music information retrieval systems. Librosa helps to visualize the audio signals and also do the feature extractions in it using different signal processing techniques.

Matplotlib:

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, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a statemachine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.

Seaborn:

Seaborn is a Python data visualization library based on matplotlib. It provides a highlevel interface for drawing attractive and informative statistical graphics. Seaborn is a library in Python predominantly used for making statistical graphics. Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

SciPy:

SciPy contains modules for optimization, linearalgebra, integration, interpolation, special functions, FFT, signal and imageprocessing, ODE solvers and other tasks common in science and engineering. SciPy is also a family of conferences for users and developers of these tools: SciPy (in the United States), EuroSciPy (in Europe) and SciPy.in (in India). Enthought originated the SciPy conference in the United States and continues to sponsor many of the international conferences as well as host the SciPy website. SciPy is a scientific computation library that uses NumPy underneath. It provides more utility functions for optimization, stats and signal processing.

Visualizing and Predicting Heart Diseases with an Interactive Dashboard source code:

Heart Disease Prediction.py

'''

Heart Disease (including Coronary Heart Disease, Hypertension, and Stroke) remains the No. 1 cause of death in the US.The Heart Disease and Stroke Statistics—2019 Update from the American Heart Association indicates that:

Machine Learning Algorithms

* Random Forest Classifier
* K-Nearest Neighbors Classifier
* Decision Tree Classifier
* Naive Bayes Classifier

Import libraries

Let's first import all the necessary libraries. We will use `numpy` and `pandas` to start with. For visualization, we will use`pyplot` subpackage of `matplotlib`, use `rcParams` to add styling to the plots and `rainbow` for colors and `seaborn`. For implementing Machine Learning models and processing of data, we will use the `sklearn` library.

'''

Commented out IPython magic to ensure Python compatibility.

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import rcParams

from matplotlib.cm import rainbow

import seaborn as sns

%matplotlib inline

"""For processing the data, we'll import a few libraries. To split the available dataset for testing and training, we'll use the `train_test_split` method. To scale the features, we are using `StandardScaler`."""

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.preprocessing import LabelEncoder

from sklearn import tree

from warnings import filterwarnings

filterwarnings("ignore")

"""For model validation, we'll import a few libraries."""

#model validation

from sklearn.metrics import log_loss,roc_auc_score,precision_score,f1_score,recall_score,roc_curve,auc,plot_roc_curve

from sklearn.metrics import classification_report, confusion_matrix,accuracy_score,fbeta_score,matthews_corrcoef

from sklearn import metrics

from mlxtend.plotting import plot_confusion_matrix

#extra

from sklearn.pipeline import make_pipeline, make_union

from sklearn.preprocessing import PolynomialFeatures

from sklearn.feature_selection import SelectFwe, f_regression

"""Next, we will import all the Machine Learning algorithms

* K-Nearest Neighbors Classifier

* Random Forest Classifier

* Decision Tree Classifier

* Naive Bayes Classifier

"""

from sklearn.ensemble import RandomForestClassifier

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.naive_bayes import GaussianNB

"""## **Import dataset**

Now that we have all the libraries we will need, we can import the dataset and take a look at it. The dataset is stored in the file `dataset.csv`. We'll use the pandas `read_csv` method to read the dataset.

"""

dataset = pd.read_csv('dataset.csv',sep=',',encoding="utf-8")

"""### **Data Preparation and Data Exploration**"""

type(dataset)

dataset.shape

"""The dataset is now loaded into the variable dataset. We'll just take a glimpse of the data using the `desribe()` and `info()` methods before we actually start processing and visualizing it."""

dataset.info()

"""Looks like the dataset has a total of 270 rows and there are no missing values. There are a total of 13 features along with one target value which we wish to find."""

dataset.columns

dataset.describe()

"""The scale of each feature column is different and quite varied as well. While the maximum for age reaches 77, the maximum of chol (serum cholestoral) is 564."""

dataset

dataset.head()

dataset.isnull().sum()

"""So, we have no missing values"""

dataset.apply(lambda x:len(x.unique()))

print('cp ',dataset['cp'].unique())

print('fbs ',dataset['fbs'].unique())

print('restecg ',dataset['restecg'].unique())

print('exang ',dataset['exang'].unique())

print('slope ',dataset['slope'].unique())

print('ca ',dataset['ca'].unique())

print('thal ',dataset['thal'].unique())

"""### **Dataset Description:**

This dataset consists of 13 features and a target variable. The detailed description of all the features are as follows:

1. **Age**: Patients Age in years (Numeric)

---

2. **Sex**: Gender of patient (Male - 1, Female - 0)(Nominal)

---

3. **Chest Pain Type**: Type of chest pain experienced by patient categorized into :(Nominal)

* Value 1: Typical angina

* Value 2: Atypical angina

* Value 3: Non-anginal pain

* Value 4: Asymptomatic

(Angina: Angina is caused when there is not enough oxygen-rich blood flowing to a certain part of the heart. The arteries of the heart become narrow due to fatty deposits in the artery

walls. The narrowing of arteries means that blood supply to the heart is reduced, causing angina.)

---

4. **resting bps**: Level of blood pressure at resting mode in mm/HG (Numerical)

---

5. **cholestrol**: Serum cholestrol in mg/dl (Numeric)

(Cholesterol means the blockage for blood supply in the blood vessels)

---

6. **fasting blood sugar**: Blood sugar levels on fasting > 120 mg/dl represents as 1 in case of true and 0 as false (Nominal)

(blood sugar taken after a long gap between a meal and the test. Typically, it's taken before any meal in the morning.)

---

7. **resting ecg**: Result of electrocardiogram while at rest are represented in 3 distinct values: (Nominal)

* Value 0: Normal

* Value 1: Having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)

* Value 2: Showing probable or definite left ventricular hypertrophy by Estes' criteria.

(ECG values taken while person is on rest which means no exercise and normal functioning of heart is happening)

---

8. **oldpeak**: Exercise induced ST-depression in comparison with the state of rest (Numeric)

(ST Depression is the difference between value of ECG at rest and after exercise.

An electrocardiogram records the electrical signals in your heart. It's a common and painless test used to quickly detect heart problems and monitor your heart's health. Electrocardiograms — also called ECGs or EKGs — are often done in a doctor's office, a clinic or a hospital room. ECG machines are standard equipment in operating rooms and ambulances. Some personal devices, such as smart watches.)

---

9. **ST slope**: ST segment measured in terms of slope during peak exercise (Nominal)

* Value 1: Upsloping

* Value 2: Flat

* Value 3: Downsloping

---

10. **ca**: Number of major blood vessels (0-3)(Numeric)

(Fluoroscopy is an imaging technique that uses X-rays to obtain real-time moving images of the interior of an object. In its primary application of medical imaging, a fluoroscope allows a physician to see the internal structure and function of a patient, so that the pumping action of the heart or the motion of swallowing, for example, can be watched)

---

11. **exang**: Exercise induced angina (1 = yes; 0 = no)

(is chest pain while exercising or doing any physical activity.)

---

12. **thal**: Thalium stress test

* Value 3: normal

* Value 6: fixed defect

* Value 7: reversibe defect

---

13. **thalach**: Maximum heart rate achieved in bpm(Numeric)

Target variable

14. **target**: It is the target variable which we have to predict 2 means patient is suffering from heart risk and 1 means patient is normal. (1 = no disease; 2 = disease)

Data Visualization

Now let's see various visual representations of the data to understand more about relationship between various features.

Distribution of Heart disease (target variable)

It's always a good practice to work with a dataset where the target classes are of approximately equal size. Thus, let's check for the same.

"""

fig, (ax1) = plt.subplots(nrows=1, ncols=1, sharey=False, figsize=(14,6))

ax1 = dataset['target'].value_counts().plot.pie(x="Heart disease" ,y ='no.of patients',

autopct = "%1.0f%%",labels=["Heart Disease","Normal"], startangle = 60,ax=ax1);

ax1.set(title = 'Percentage of Heart disease patients in Dataset')

plt.show()

"""The two classes are not exactly 50% each but the ratio is good enough to continue without dropping/increasing our data."""

```
y = dataset["target"]
rcParams['figure.figsize'] = 8,6
plt.bar(dataset['target'].unique(), dataset['target'].value_counts(), color = ['blue', 'green'])
plt.xticks([1, 2])
plt.xlabel('Target Classes (1 =no disease; 2 = disease)')
plt.ylabel('Samples')
plt.title('Count of each Target Class')
target_temp = dataset.target.value_counts()
print(target_temp)
"""From the total dataset of 270 patients, 150 (56%) have a heart disease (target=2)
Next, we'll take a look at the histograms for each variable.
"""
dataset.hist(edgecolor='black',layout = (7, 2),
        figsize = (10, 30),
        color=['purple'])
"""Taking a look at the histograms above, I can see that each feature has a different range of distribution. Thus, using scaling before our predictions should be of great use. Also, the categorical features do stand out.
## **Exploratory Data Analysis (EDA)**
### **Gender distribution based on heart disease**
"""
dataset["sex"].unique()
# Number of males and females
F = dataset[dataset["sex"] == 0].count()["target"]
M = dataset[dataset["sex"] == 1].count()["target"]
# Create a plot
figure, ax = plt.subplots(figsize = (6, 4))
ax.bar(x = ['Female', 'Male'], height = [F, M])
plt.xlabel('Gender')
plt.title('Number of Males and Females in the dataset')
plt.show()
"""**Heart Disease frequency for gender**"""
```

```
pd.crosstab(dataset.sex,dataset.target).plot(kind="bar",figsize=(20,10),color=['blue','#AA1111' ])

plt.title('Heart Disease Frequency for Sex')

plt.xlabel('Sex (0 = Female, 1 = Male)')

plt.xticks(rotation=0)

plt.legend(["Don't have Disease", "Have Disease"])

plt.ylabel('Frequency')

plt.show()

countFemale = len(dataset[dataset.sex == 0])

countMale = len(dataset[dataset.sex == 1])

print("Percentage of Female Patients:{:.2f}%".format((countFemale)/(len(dataset.sex))*100))

print("Percentage of Male Patients:{:.2f}%".format((countMale)/(len(dataset.sex))*100))

"""### **Age distribution based on heart disease**"""

# Display age distribution based on heart disease

sns.distplot(dataset[dataset['target'] == 1]['age'], label='Do not have heart disease')

sns.distplot(dataset[dataset['target'] == 2]['age'], label = 'Have heart disease')

plt.xlabel('Frequency')

plt.ylabel('Age')

plt.title('Age Distribution based on Heart Disease')

plt.legend()

plt.show()

"""Get min, max and average of the age of the people do not have heart disease"""

print('Min age of people who do not have heart disease: ', min(dataset[dataset['target'] == 1]['age']))

print('Max age of people who do not have heart disease: ', max(dataset[dataset['target'] == 1]['age']))

print('Average age of people who do not have heart disease: ', dataset[dataset['target'] == 1]['age'].mean())

"""Get min, max and average of the age of the people have heart disease"""

print('Min age of people who have heart disease: ', min(dataset[dataset['target'] == 2]['age']))

print('Max age of people who have heart disease: ', max(dataset[dataset['target'] == 2]['age']))

print('Average age of people who have heart disease: ', dataset[dataset['target'] == 2]['age'].mean())
```

"""From the data, we can say that the heart disease infects the old and young people, and the probability of the old people te be infected is higher than young people.

Heart Disease Frequency for ages

"""

```
pd.crosstab(dataset.age,dataset.target).plot(kind="bar",figsize=(20,6))
plt.title('Heart Disease Frequency for Ages')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.savefig('heartDiseaseAndAges.png')
plt.show()
plt.figure(figsize=(12, 10))
dataset.age.hist(bins=80)
print(f"The most of the patients have a mean age of : {dataset.age.mean()}")
```

"""### **Distribution of Categorial features**"""

```
categorial = [('sex', ['female', 'male']),
              ('cp', ['typical angina', 'atypical angina', 'non-anginal pain', 'asymptomatic']),
              ('fbs', ['fbs > 120mg', 'fbs < 120mg']),
              ('restecg', ['normal', 'ST-T wave', 'left ventricular']),
              ('exang', ['yes', 'no']),
              ('slope', ['upsloping', 'flat', 'downsloping']),
              ('thal', ['normal', 'fixed defect', 'reversible defect'])]
def plotGrid(isCategorial):
    if isCategorial:
        [plotCategorial(x[0], x[1], i) for i, x in enumerate(categorial)]
    else:
        [plotContinuous(x[0], x[1], i) for i, x in enumerate(continuous)]
def plotCategorial(attribute, labels, ax_index):
    sns.countplot(x=attribute, data=dataset, ax=axes[ax_index][0])
    sns.countplot(x='target', hue=attribute, data=dataset, ax=axes[ax_index][1])
    avg = dataset[[attribute, 'target']].groupby([attribute], as_index=False).mean()
    sns.barplot(x=attribute, y='target', hue=attribute, data=avg, ax=axes[ax_index][2])
```

```
    for t, l in zip(axes[ax_index][1].get_legend().texts, labels):
        t.set_text(l)
    for t, l in zip(axes[ax_index][2].get_legend().texts, labels):
        t.set_text(l)
fig_categorial, axes = plt.subplots(nrows=len(categorial), ncols=3, figsize=(15, 30))
plotGrid(True)
"""### **Distribution of Continuous features**"""
continuous = [('trestbps', 'blood pressure in mm Hg'),
              ('chol', 'serum cholestoral in mg/d'),
              ('thalach', 'maximum heart rate achieved'),
              ('oldpeak', 'ST depression by exercise relative to rest'),
              ('ca', '# major vessels: (0-3) colored by flourosopy')]
def plotContinuous(attribute, xlabel, ax_index):
    sns.distplot(dataset[[attribute]], ax=axes[ax_index][0])
    axes[ax_index][0].set(xlabel=xlabel, ylabel='density')
    sns.violinplot(x='target', y=attribute, data=dataset, ax=axes[ax_index][1])
fig_continuous, axes = plt.subplots(nrows=len(continuous), ncols=2, figsize=(15, 22))
plotGrid(isCategorial=False)
"""### **PiePlots**"""
fig, ax = plt.subplots(4,2, figsize = (14,14))
((ax1, ax2), (ax3, ax4), (ax5, ax6), (ax7, ax8)) = ax
labels = ["Male", "Female"]
values = dataset['sex'].value_counts().tolist()[:2]
ax1.pie(x=values,                                                    labels=labels,
autopct="%1.1f%%",colors=['#AAb3ff','#CC80FF'],shadow=True,
startangle=45,explode=[0.1, 0.1])
ax1.set_title("Sex", fontdict={'fontsize': 12},fontweight ='bold')
labels = ["Typical angina", "Atypical angina","non-anginal pain","asymptomatic"]
values = dataset['cp'].value_counts().tolist()
ax2.pie(x=values,                                                    labels=labels,
autopct="%1.1f%%",colors=['#AAb3ff','#CC80FF','#DD00AA','#FF0099'],shadow=True,star
tangle=45,explode=[0.1, 0.1, 0.1, 0.2])
ax2.set_title("Chest Pain", fontdict={'fontsize': 12},fontweight ='bold')
```

```
labels = dataset['fbs'].value_counts().index.tolist()[:2]

values = dataset['fbs'].value_counts().tolist()

ax3.pie(x=values, labels=labels, autopct="%1.1f%%",colors=['#AAb3ff','#CC80FF'],shadow=True, startangle=45,explode=[0.1, 0.15])

ax3.set_title("Fasting Blood Sugar", fontdict={'fontsize': 12},fontweight ='bold')

labels = dataset['restecg'].value_counts().index.tolist()[:3]

values = dataset['restecg'].value_counts().tolist()

ax4.pie(x=values, labels=labels, autopct="%1.1f%%", colors=['#AAb3ff','#CC80FF','#DD00AA'],shadow=True,startangle=45,explode=[ 0.05, 0.05, 0.05])

ax4.set_title("Resting Blood Pressure", fontdict={'fontsize': 12},fontweight ='bold')

labels = dataset['exang'].value_counts().index.tolist()[:2]

values = dataset['exang'].value_counts().tolist()

ax5.pie(x=values, labels=labels, autopct="%1.1f%%", colors=['#AAb3ff','#CC80FF'],shadow=True, startangle=45,explode=[0.1, 0.1])

ax5.set_title("Exercise induced Angina", fontdict={'fontsize': 12},fontweight ='bold')

labels = dataset['slope'].value_counts().index.tolist()[:3]

values = dataset['slope'].value_counts().tolist()

ax6.pie(x=values, labels=labels, autopct="%1.1f%%", colors=['#AAb3ff','#CC80FF','#DD00AA'],shadow=True,startangle=45,explode=[ 0.1, 0.1, 0.1])

ax6.set_title("Peak exercise ST_segment Slope", fontdict={'fontsize': 12},fontweight ='bold')

labels = dataset['ca'].value_counts().index.tolist()[:4]

values = dataset['ca'].value_counts().tolist()

ax7.pie(x=values, labels=labels, autopct="%1.1f%%", shadow=True, startangle=45,explode=[0.05, 0.07, 0.1, 0.1],colors=['#AAb3ff','#CC80FF','#DD00AA','#FF0099'])

ax7.set_title("Major vessels", fontdict={'fontsize': 12},fontweight ='bold')

labels = dataset['thal'].value_counts().index.tolist()[:3]

values = dataset['thal'].value_counts().tolist()

ax8.pie(x=values, labels=labels, autopct="%1.1f%%", shadow=True, startangle=45,explode=[0.1, 0.1, 0.1],colors=['#AAb3ff','#CC80FF','#DD00AA'])

ax8.set_title("Types of Thalassemia", fontdict={'fontsize': 12},fontweight ='bold')

plt.tight_layout()

plt.show()
```

```
plt.savefig("PiePlots.png")

"""### **Target Correlations**"""

plt.figure(figsize=(10,10))

sns.heatmap(pd.DataFrame(dataset.corr()['target']).sort_values(by='target').transpose().drop('target',axis=1).transpose(),annot=True,cmap='twilight')

plt.savefig("TargetCorrelations.png")

"""### **Feature Importance**"""

X = dataset.drop('target',axis=1)

Y = dataset['target']

from sklearn.feature_selection import SelectKBest, chi2

fs = SelectKBest(score_func=chi2, k='all')

fs.fit(X, Y)

per = []

for i in fs.scores_:

    per.append(round(((i/sum(fs.scores_))*100),3))

features_data = pd.DataFrame({'Feature':X.columns,'Scores':fs.scores_,'Importance (%)':per}).sort_values(by=['Scores'],ascending=False)

plt.figure(figsize=(9,4))

sns.barplot( 'Importance (%)','Feature',orient='h',data=features_data,palette='twilight_shifted_r')

insignificant = features_data.loc[features_data['Importance (%)']<0.005]['Feature'].unique()

features_data = features_data.set_index('Feature')

features_data

plt.savefig("FeatureImportance.png")

"""### **Analysing Fasting Blood sugar** [fbs]

Heart disease according to Fasting Blood sugar

"""

# Display fasting blood sugar in bar chart

dataset.groupby(dataset['fbs']).count()['target'].plot(kind = 'bar', title = 'Fasting Blood Sugar', figsize = (8, 6))

plt.xticks(np.arange(2), ('fbs < 120 mg/dl', 'fbs > 120 mg/dl'), rotation = 0)

plt.show()

"""#### **Display fasting blood sugar based on the target**"""
```

```
pd.crosstab(dataset.fbs,dataset.target).plot(kind = "bar", figsize = (8, 6))

plt.title('Heart Disease Frequency According to Fasting Blood Sugar')

plt.xlabel('Fasting Blood Sugar')

plt.xticks(np.arange(2), ('fbs < 120 mg/dl', 'fbs > 120 mg/dl'), rotation = 0)

plt.ylabel('Frequency')

plt.show()

"""### **Analysing the Chest Pain** [cp] (4 types of chest pain)

[Value 1: typical angina, Value 2: atypical angina, Value 3: non-anginal pain, Value 4: asymptomatic]

"""

dataset["cp"].unique()

plt.figure(figsize=(26, 10))

sns.barplot(dataset["cp"],y)

"""**Display chest pain types based on the target**"""

pd.crosstab(dataset.cp,dataset.target).plot(kind = "bar", figsize = (8, 6))

plt.title('Heart Disease Frequency According to Chest Pain Type')

plt.xlabel('Chest Pain Type')

plt.xticks(np.arange(4), ('typical angina', 'atypical angina', 'non-anginal pain', 'asymptomatic'), rotation = 0)

plt.ylabel('Frequency')

plt.show()

"""### **Analysing Resting Blood Pressure** [trestbps]

mm Hg on admission to the hospital

"""

dataset["trestbps"].unique()

plt.figure(figsize=(26, 10))

sns.barplot(dataset["trestbps"],y)

"""#### **Display blood pressure distribution based on heart disease**"""

fig, (axis1, axis2) = plt.subplots(1, 2,figsize=(25, 5))

ax = sns.distplot(dataset[dataset['target'] == 1]['trestbps'], label='Do not have heart disease', ax = axis1)

ax.set(xlabel='People Do Not Have Heart Disease')
```

```
ax = sns.distplot(dataset[dataset['target'] == 2]['trestbps'], label = 'Have heart disease', ax = axis2)

ax.set(xlabel='People Have Heart Disease')

plt.show()

# Get min, max and average of the  blood pressure of the people do not have heart diseas

print('Min blood pressure of people who do not have heart disease: ', min(dataset[dataset['target'] == 1]['trestbps']))

print('Max blood pressure of people who do not have heart disease: ', max(dataset[dataset['target'] == 1]['trestbps']))

print('Average blood pressure of people who do not have heart disease: ', dataset[dataset['target'] == 1]['trestbps'].mean())

# Get min, max and average of the blood pressure of the people have heart diseas

print('Min blood pressure of people who have heart disease: ', min(dataset[dataset['target'] == 2]['trestbps']))

print('Max blood pressure of people who have heart disease: ', max(dataset[dataset['target'] == 2]['trestbps']))

print('Average blood pressure of people who have heart disease: ', dataset[dataset['target'] == 2]['trestbps'].mean())

"""### **Analysing the Resting Electrocardiographic Measurement** [restecg]

(0 = normal, 1 = having ST-T wave abnormality, 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria)

"""

dataset["restecg"].unique()

# Display electrocardiographic results in bar chart

dataset.groupby(dataset['restecg']).count()['target'].plot(kind = 'bar', title = 'Resting Electrocardiographic Results', figsize = (8, 6))

plt.xticks(np.arange(3), ('normal', 'ST-T wave abnormality', 'probable or left ventricular hypertrophy'))

plt.show()

# Display resting electrocardiographic results based on the target

pd.crosstab(dataset.restecg,dataset.target).plot(kind = "bar", figsize = (8, 6))

plt.title('Heart Disease Frequency According to Resting Electrocardiographic Results')

plt.xticks(np.arange(3), ('normal', 'ST-T wave abnormality', 'probable or left ventricular hypertrophy'))

plt.xlabel('Resting Electrocardiographic Results')

plt.ylabel('Frequency')
```

plt.show()

"""Usually the people who do not have heart disease have normal electrocardiographic, whereas the people who have heart disease have probable or left ventricular hypertrophy.

Analysing Exercise Induced Angina [exang]

(1 = yes; 0 = no)

"""

dataset["exang"].unique()

Display exercise induced angina in bar chart

dataset.groupby(dataset['exang']).count()['target'].plot(kind = 'bar', title = 'Exercise Induced Angina', figsize = (8, 6))

plt.xticks(np.arange(2), ('No', 'Yes'), rotation = 0)

plt.show()

"""#### **Display exercise induced angina based on the target**"""

pd.crosstab(dataset.exang,dataset.target).plot(kind = "bar", figsize = (8, 6))

plt.title('Heart Disease Frequency According to Exercise Induced Angina')

plt.xlabel('Exercise Induced Angina')

plt.xticks(np.arange(2), ('No', 'Yes'), rotation = 0)

plt.ylabel('Frequency')

plt.show()

"""The people who suffer from exercise induced angina are more likely to be infected with the heart disease.

Analysing the Slope of the peak exercise ST segment [slope]

(Value 1: upsloping, Value 2: flat, Value 3: downsloping)

"""

dataset["slope"].unique()

Display slope of the peak exercise ST segment in bar chart

dataset.groupby(dataset['slope']).count()['target'].plot(kind = 'bar', title = 'Slope of the Peak Exercise ST Segment', figsize = (8, 6))

plt.xticks(np.arange(3), ('upsloping', 'flat', 'downsloping'), rotation = 0)

plt.show()

"""#### **Display slope of the peak exercise ST segment based on the target**"""

pd.crosstab(dataset.slope,dataset.target).plot(kind = "bar", figsize = (8, 6))

plt.title('Heart Disease Frequency According to Slope of the Peak Exercise ST Segment')

```
plt.xlabel('Slope')

plt.xticks(np.arange(3), ('upsloping', 'flat', 'downsloping'), rotation = 0)

plt.ylabel('Frequency')

plt.show()

"""As it is clear, the people with flatslope peak ST segment are likely to have heart disease and usually the people who do not have heart disease have upsloping peak ST segment.

### **Analysing Number of Major Vessels (0-3) colored by flourosopy** [ca]

"""

dataset["ca"].unique()

"""#### **Display number of major vessels in bar chart**"""

dataset.groupby(dataset['ca']).count()['target'].plot(kind = 'bar', title = 'Number of Major Vessels Colored by Flourosopy',

                                                      figsize = (8, 6))

plt.show()

"""#### **Display number of vessels based on the target**"""

pd.crosstab(dataset.ca,dataset.target).plot(kind = "bar", figsize = (8, 6))

plt.title('Heart Disease Frequency According to Number of Major Vessels Colored by Flourosopy')

plt.xlabel('number of vessels')

plt.xticks(rotation = 0)

plt.ylabel('Frequency')

plt.show()

"""As it is clear, the people who do not have heart disease usually do not have major vessels colored by flourosopy.

### **Analysing a Blood Disorder called Thalassemia** [thal]

(3 = normal ; 6 = fixed defect ; 7 = reversable defect)

"""

dataset["thal"].unique()

"""#### **plotting the thalassemia distribution** (3,6,7)"""


sns.distplot(dataset["thal"])

"""#### **Display thalassemia in bar chart**"""

dataset.groupby(dataset['thal']).count()['target'].plot(kind = 'bar', title = 'Thalassemia')
```

```
plt.xticks(np.arange(3), ('normal', 'fixed defect', 'reversible defect'), rotation = 0)
plt.show()
"""#### **Thalassemia compared with target**"""
pd.crosstab(dataset.thal,dataset.target).plot(kind = "bar", figsize = (8, 6))
plt.title('Heart Disease Frequency According to Thalassemia')
plt.xlabel('Thalassemia')
plt.xticks(np.arange(3), ('normal', 'fixed defect', 'reversible defect'), rotation = 0)
plt.ylabel('Frequency')
plt.show()
"""As it is clear, the people with reversible defect are likely to have heart disease.
### **Thalassemia vs Cholesterol Scatterplot**
"""
plt.figure(figsize=(20,10))
sns.scatterplot(x='chol',y='thal',data=dataset,hue='target')
plt.show()
"""### **Thalassemia vs Resting blood pressure Scatterplot**"""
plt.figure(figsize=(20,10))
sns.scatterplot(x='thal',y='trestbps',data=dataset,hue='target')
plt.show()
"""### **Thalassemia vs Age Scatterplot**"""
plt.figure(figsize=(20, 10))
plt.scatter(x=dataset.age[dataset.target==2], y=dataset.thal[(dataset.target==2)], c="green")
plt.scatter(x=dataset.age[dataset.target==1], y=dataset.thal[(dataset.target==1)])
plt.legend(["Disease", "Not Disease"])
plt.xlabel("Age")
plt.ylabel("Maximum Heart Rate")
plt.show()
"""### **Maximum Heart Rate vs Oldpeak(Exercise induced ST-depression in comparison with the state of rest)** """
plt.figure(figsize=(15,5))
plt.subplot(1,2,1)
sns.histplot(data=dataset,hue='target',x='thalach',bins=20,element='poly')
```

plt.subplot(1,2,2)

sns.histplot(data=dataset,hue='target',x='oldpeak',bins=20,element='poly')

plt.savefig("Thalach&oldpeak_Histplot.png")

"""### **Resting Blood Pressure vs Cholestrol**"""

plt.figure(figsize=(15,5))

plt.subplot(1,2,1)

sns.histplot(data=dataset,hue='target',x='trestbps',bins=20,element='poly')

plt.subplot(1,2,2)

sns.histplot(data=dataset,hue='target',x='chol',bins=20,element='poly')

plt.savefig("Resting_blood_pressure&chol_Histplot.png")

"""### **Pair Plots**"""

sns.pairplot(data=dataset)

"""### **Correlation Matrix**

The best way to compare relationship between various features is to look at the correlation matrix between those features.

"""

corr_matrix = dataset.corr()

top_corr_feature = corr_matrix.index

plt.figure(figsize=(20, 20))

sns.heatmap(dataset[top_corr_feature].corr(), annot=True, cmap="RdYlGn", annot_kws={"size":15})

"""Taking a look at the correlation matrix above, it's easy to see that a few features have negative correlation with the target value while some have positive.

Data Processing

After exploring the dataset, we observed that we need to convert some categorical variables into dummy variables and scale all the values before training the Machine Learning models. First, we'll use the `get_dummies` method to create dummy columns for categorical variables.

"""

dataset = pd.get_dummies(dataset, columns = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal'])

"""Now, we will use the `StandardScaler` from `sklearn` to scale my dataset."""

standardScaler = StandardScaler()

columns_to_scale = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']

dataset[columns_to_scale] = standardScaler.fit_transform(dataset[columns_to_scale])

```
dataset.head()

dataset.describe()

"""### **Train Test Split**

We'll now import train_test_split to split our dataset into training and testing datasets. Then, we'll import all Machine Learning models we'll be using to train and test the data.

"""

Y = dataset['target'].values

X = dataset.drop('target',axis=1).values

X.shape

Y.shape

# Split the dataset into training and testing.

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=0)

print("Training features have {0} records and Testing features have {1} records.".\

    format(X_train.shape[0], X_test.shape[0]))

"""**Checking distribution of traget variable in train test split**"""

print('--------------Training Set----------------')

print(X_train.shape)

print(Y_train.shape)

print('----------------Test Set------------------')

print(X_test.shape)

print(Y_test.shape)

"""## **Machine Learning Model**

### **1. Random Forest Classifier**

Now, we'll use the ensemble method, Random Forest Classifier, to create the model and vary the number of estimators to see their effect.

"""

max_accuracy = 0

for x in range(500):

    rf_classifier = RandomForestClassifier(random_state=x)

    rf_classifier.fit(X_train,Y_train)

    Y_pred_rf = rf_classifier.predict(X_test)

    current_accuracy = round(accuracy_score(Y_pred_rf,Y_test)*100,2)

    if(current_accuracy>max_accuracy):
```

```
        max_accuracy = current_accuracy
        best_x = x
print(max_accuracy)
print(best_x)
rf_classifier = RandomForestClassifier(random_state=best_x)
rf_classifier.fit(X_train,Y_train)
Y_pred_rf = rf_classifier.predict(X_test)
Y_pred_rf.shape
score_rf = round(accuracy_score(Y_pred_rf,Y_test)*100,2)
score_rf
"""#### **Model Evaluation:**
y_pred_rfe = rf_classifier.predict(X_test)
plt.figure(figsize=(10, 8))
CM=confusion_matrix(Y_test,y_pred_rfe)
sns.heatmap(CM, annot=True)
TN = CM[0][0]
FN = CM[1][0]
TP = CM[1][1]
FP = CM[0][1]
specificity = TN/(TN+FP)
loss_log = log_loss(Y_test, y_pred_rfe)
acc= accuracy_score(Y_test, y_pred_rfe)
roc=roc_auc_score(Y_test, y_pred_rfe)
prec = precision_score(Y_test, y_pred_rfe)
rec = recall_score(Y_test, y_pred_rfe)
f1 = f1_score(Y_test, y_pred_rfe)
mathew = matthews_corrcoef(Y_test, y_pred_rfe)
model_results =pd.DataFrame([['Random Forest',acc, prec,rec,specificity, f1,roc, loss_log,mathew]],
            columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity', 'F1 Score','ROC','Log_Loss','mathew_corrcoef'])
model_results
Y_pred_rf = np.around(Y_pred_rf)
```

```
print(metrics.classification_report(Y_test,Y_pred_rf))

plot_roc_curve(rf_classifier,X_test,Y_test)

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic Curve');

plt.savefig("RF.png")

"""### **2. K-Nearest Neighbors Classifier**
```

The classification score varies based on different values of neighbors that we choose. Thus, we'll plot a score graph for different values of K (neighbors) and check when do we achieve the best score.

```
"""

knn_classifier= KNeighborsClassifier(n_neighbors=31,leaf_size=30)

knn_classifier.fit(X_train,Y_train)

Y_pred_knn = knn_classifier.predict(X_test)

score_knn = round(accuracy_score(Y_pred_knn,Y_test)*100,2)

score_knn


"""#### **Model Evaluation:**"""

y_pred_knne = knn_classifier.predict(X_test)

plt.figure(figsize=(10, 8))

CM=confusion_matrix(Y_test,y_pred_knne)

sns.heatmap(CM, annot=True)

TN = CM[0][0]

FN = CM[1][0]

TP = CM[1][1]

FP = CM[0][1]

specificity = TN/(TN+FP)

loss_log = log_loss(Y_test, y_pred_knne)

acc= accuracy_score(Y_test, y_pred_knne)

roc=roc_auc_score(Y_test, y_pred_knne)

prec = precision_score(Y_test, y_pred_knne)

rec = recall_score(Y_test, y_pred_knne)
```

```
f1 = f1_score(Y_test, y_pred_knne)

mathew = matthews_corrcoef(Y_test, y_pred_knne)

model_results =pd.DataFrame([['K-Nearest Neighbors ',acc, prec,rec,specificity, f1,roc, loss_log,mathew]],

          columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity', 'F1 Score','ROC','Log_Loss','mathew_corrcoef'])

model_results

Y_pred_knn = np.around(Y_pred_knn)

print(metrics.classification_report(Y_test,Y_pred_knn))

plot_roc_curve(knn_classifier,X_test,Y_test)

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic Curve');

plt.savefig("KNN.png")

"""### **3. Decision Tree Classifier**

Here, we'll use the Decision Tree Classifier to model the problem at hand. We'll vary between a set of `max_features` and see which returns the best accuracy.

"""

dt_classifier = DecisionTreeClassifier(

   max_depth=20,

   min_samples_split=2,

   min_samples_leaf=1,

   min_weight_fraction_leaf=0.00001,

   max_features='auto',

   random_state=46)

dt_classifier.fit(X_train, Y_train)

Y_pred_dt=dt_classifier.predict(X_test)

score_dt = round(accuracy_score(Y_pred_dt,Y_test)*100,2)

score_dt

"""#### **Model Evaluation:**"""

y_pred_dte = dt_classifier.predict(X_test)

plt.figure(figsize=(10, 8))

CM=confusion_matrix(Y_test,y_pred_dte)
```

```
sns.heatmap(CM, annot=True)

TN = CM[0][0]

FN = CM[1][0]

TP = CM[1][1]

FP = CM[0][1]

specificity = TN/(TN+FP)

loss_log = log_loss(Y_test, y_pred_dte)

acc= accuracy_score(Y_test, y_pred_dte)

roc=roc_auc_score(Y_test, y_pred_dte)

prec = precision_score(Y_test, y_pred_dte)

rec = recall_score(Y_test, y_pred_dte)

f1 = f1_score(Y_test, y_pred_dte)

mathew = matthews_corrcoef(Y_test, y_pred_dte)

model_results =pd.DataFrame([['Decision Tree',acc, prec,rec,specificity, f1,roc, loss_log,mathew]],

            columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity', 'F1 Score','ROC','Log_Loss','mathew_corrcoef'])

model_results

Y_pred_dt = np.around(Y_pred_dt)

print(metrics.classification_report(Y_test,Y_pred_dt))

plot_roc_curve(dt_classifier,X_test,Y_test)

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic Curve');

"""### **4. Naive Bayes Classifier**

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.”””

nb_classifier = GaussianNB( var_smoothing=1e-50)

nb_classifier.fit(X_train,Y_train)

nb_classifier.predict(X_test)

Y_pred_nb = nb_classifier.predict(X_test)

score_nb = round(accuracy_score(Y_pred_nb,Y_test)*100,2)

score_nb
```

Web App Code for ipynb file to flask framework view Visualizing and Predicting Heart Diseases with an Interactive Dashboard:

Heart_prediction_app.py

```
# -*- coding: utf-8 -*-
import numpy as np
import pickle
from flask import Flask, request, render_template
# Load ML model
model = pickle.load(open('models.pkl', 'rb'))
# Create application
app = Flask(__name__)
# Bind home function to URL
@app.route('/')
def home():
    return render_template('Heart_Disease_Classifier.html')
# Bind predict function to URL
@app.route('/predict', methods =['POST'])
def predict():
    # Put all form entries values in a list
    features = [float(i) for i in request.form.values()]
    # Convert features to array
    array_features = [np.array(features)]
    # Predict features
    prediction = model.predict(array_features)
    output = prediction
    # Check the output values and retrive the result with html tag based on the value
 if output == 1:
        return render_template('Heart_Disease_Classifier.html',
                               result = 'The patient is not likely to have heart disease!')

  else:
        return render_template('Heart_Disease_Classifier.html',
```

result = 'The patient is likely to have heart disease!')

if __name__ == '__main__':

#Run the application

Heart_Disease_Classifier.html

```
<html>

<head>

<!-- Bootstrap CSS -->

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
integrity="sha384-
JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z"
crossorigin="anonymous">

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"
integrity="sha384-
9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN"
crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"
integrity="sha384-
B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV"
crossorigin="anonymous"></script>

<title>Heart Disease Test</title>

</head>

<body>

<!-- Java Script -->

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"
integrity="sha384-
9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN"
crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"
integrity="sha384-
B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV"
crossorigin="anonymous"></script>

<!-- Navbar-->
```

```
<nav class="navbar navbar-dark" style="background-color: rgb(13, 102, 87);">
<span class="navbar-brand mb-0 h1">Heart Disease Test</span>
</nav>
<div class="container">
<br>
<!--Form-->
<form action = "{{url_for('predict')}}" method ="POST" >
<fieldset>
<legend style="color: rgb(41, 15, 134);"><b>Heart Disease Test Form</b></legend><br>
<div class="card card-body" style="background-color: rgb(194 245 236 / 56%);">
<div class="form-group  row">
<div class="col-sm-3">
<label for="age">Age</label>
<input type="number" class="form-control" id="age" name="age" required>
</div>
<div class="col-sm-3">
<label for="sex">Sex</label>
<select class="form-control" id="sex"  name="sex" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "0">Female</option>
<option value = "1">Male</option>
</select>
</div>
</div>
<br>
<div class="form-group  row">
<div class="col-sm">
<label for="cp">Chest Pain Type</label>
<select class="form-control" id="cp" name = "cp" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "1">Typical Angina</option>
<option value = "2">Atypical Angina</option>
```

```
<option value = "3">Non-anginal Pain</option>
<option value = "4">Asymptomatic</option>
</select>
</div>
<div class="col-sm">
<label for="trestbps">Resting Blood Pressure in mm Hg</label>
<input type="number" class="form-control" id="trestbps" name="trestbps"  required>
</div>
<div class="col-sm">
<label for="chol">Serum Cholestoral in mg/dl</label>
<input type="number" class="form-control" id="chol" name="chol" required>
</div>
<div class="col-sm">
<label for="fbs">Fasting Blood Sugar > 120 mg/dl</label>
<select class="form-control" id="fbs" name="fbs" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "0">False</option>
<option value = "1">True</option>
</select>
</div>
</div>

<br>
<div class="form-group row">
<div class="col-sm">
<label for="restecg">Resting ECG Results </label>
<select class="form-control" id="restecg" name="restecg" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "0">Normal  </option>
<option value = "1">Having ST-T wave abnormality  </option>
<option value = "2">Probable or definite left ventricular hypertrophy</option>
</select>
```

```
</div>
<div class="col-sm">
<label for="thalach">Maximum Heart Rate</label>
<input type="number" class="form-control" id="thalach" name="thalach" required>
</div>
<div class="col-sm">
<label for="exang">Exercise Induced Angina </label>
<select class="form-control" id="exang" name="exang" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "0">No</option>
<option value = "1">Yes</option>
</select>
</div>
<div class="col-sm">
<label for="oldpeak">ST Depression Induced</label>
<input type="number" step="any" class="form-control" id="oldpeak" name="oldpeak"
required>
</div>
</div>
<br>
<div class="form-group  row">
<div class="col-sm">
<label for="slope">Slope of the Peak Exercise ST Segment </label>
<select class="form-control" id="slope" name="slope" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "1">Upsloping</option>
<option value = "2">Flat</option>
<option value = "3">Downsloping</option>
</select>
</div>
<div class="col-sm">
<label for="ca">Number of Vessels Colored by Flourosopy</label>
```

```
<select class="form-control" id="ca" name = "ca" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "0">0</option>
<option value = "1">1</option>
<option value = "2">2</option>
<option value = "3">3</option>
</select>
</div>
<div class="col-sm">
<label for="thal">Thalassemia</label>
<select class="form-control" id="thal" name = "thal" required>
<option disabled selected value> -- Select an Option -- </option>
<option value = "3">Normal</option>
<option value = "6">Fixed defect</option>
<option value = "7">Reversable defect</option>
</select>
</div>
</div>
<br>
<div class="form-group">
<input class="btn btn-primary" type="submit" value="Result">
</div>
<!--Prediction Result-->
<div id ="result">
<strong style="color:red">{{result}}</strong>
</div>
</div>
</fieldset>
</form>
</div>
</body>
</html>
```

Web App Code for Embedding Dashboard Visualizing and Predicting Heart Diseases with an Interactive Dashboard:

embedded cognos dashboard.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<title>Heart Disease Prediction</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"></script>

</head>

<body>

<div class="container-fluid p-5 bg-primary text-white text-center"> <h1> Visualizing and Predicting Heart Diseases with an Interactive Dashboard</h1>

<p>Heart Disease Prediction dashboard</p>

<p><iframe src="https://us3.ca.analytics.ibm.com/bi/?pathRef=.my_folders%2FIBM%2BPROJECT%2FReport%2FHeart%2BDisease%2BPrediction%2Breport&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=run&amp;format=HTML&amp;prompt=false" width="1300" height="600" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe></p>

</div>

</body>

</html>
```

Web App Code for Embedding Story Visualizing and Predicting Heart Diseases with an Interactive Dashboard:

embedded cognos story.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<title>Heart Disease Prediction</title>
```

```
<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"></script>

</head>

<body>

<div class="container-fluid p-5 bg-primary text-white text-center"> <h1> Visualizing and Predicting Heart Diseases with an Interactive Dashboard</h1>

<p>Story of Heart Disease Prediction</p>

<p><iframe src="https://us3.ca.analytics.ibm.com/bi/?perspective=story&amp;pathRef=.my_folders%2FIBM%2BPROJECT%2FStory%2FHeart%2BDisease%2BPrediction%2BStory&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=view&amp;sceneId=model000001843796678f_00000000&amp;sceneTime=0" width="1300" height="650" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe></p>

</div>

</body>

</html>
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-45585-1660731143.git

PROJECT DEMO LINK:

https://drive.google.com/file/d/1tfezFRAoAT1yBNe6fR_DIrF3lKTEDcq7/view?usp=drivesdk