DETECTING OF PARKINSON'S DISEASE USING MACHINE LEARNING

A PROJECT REPORT

Submitted by

HARIHARAN M

HARIPRASATH B

HARI PRIYA D

RAJA M

TEAM ID : PNT2022TMID06357

TABLE OF CONTENTS :

1. INTRODUCTION

1.1 Project Overview

The recent report of the World Health Organization shows a visible increase in the number and health burden of Parkinson's disease patients increases rapidly. In China, this disease is spreading so fast and estimated that it reaches half of the population in the next 10 years. Classification algorithms are mainly used in the medical field for classifying data into different categories according to the number of characteristics. Parkinson's disease is the second most dangerous neurological disorder that can lead to shaking, shivering, stiffness, and difficulty walking and balance. It caused mainly due by the breaking down of cells in the nervous system. Parkinson's can have both motor and non-motor symptoms. The motor symptoms include slowness of movement, rigidity, balance problems, and tremors. If this disease continues, the patients may have difficulty walking and talking. The non-motor symptoms include anxiety, breathing problems, depression, loss of smell, and change in speech. If the above-mentioned symptoms are present in the person then the details are stored in the records. In this paper, the author considers the speech features of the patient, and this data is used for predicting whether the patient has Parkinson's disease or not. Neurodegenerative disorders are the results of progressive tearing and neuron loss in different areas of the nervous system. Neurons are functional units of the brain. They are contiguous rather than continuous. A good healthy looking neuron as shown in fig 1 has extensions called dendrites or axons, a cell body, and a nucleus that contains our DNA. DNA is our genome and a hundred billion neurons contain our entire genome which is packaged into it. When a neuron gets sick, it loses its extension and hence its ability to communicate which is not good for it and its metabolism becomes low so it starts to accumulate junk and it tries to contain the junk in the little packages in little pockets. When things become worse and if the neuron is a cell culture it completely loses its extension, becomes round and full of vacuoles. This work deals with the prediction of Parkinson's disorder which is now a day is tremendously increasing incurable disease. Parkinson's disease is a most spreading disease which gets its name from James Parkinson who earlier described it as a paralysis agitans and later

gave his surname was known as PD. It generally affects the neurons which are responsible for overall body movements. The main chemicals are dopamine and acetylcholine which affect the human brain. There is a various environmental factor which has been implicated in PD below are the listed factor which caused Parkinson's disease in an individual.

1.1.1 Parkinson's disease symptoms

The symptoms of Parkinson's disease broadly divided into two categories.

- Motor symptoms: This is a symptom where any voluntary action involved. It indicates the movement-related disorders like tremors, rigidity, freezing, Bradykinesia or any voluntary muscle movement.

- Non-Motor symptoms: Non motor symptoms include disorders of mood and affect with apathy, cognitive dysfunction as well as complex behavioral disorders. There are two other categories of PD which are divided by doctors: Primary symptom and Secondary symptom.

- Primary symptoms: It is the most important symptom. Primary symptoms are rigidity, tremor and slowness of movement.

- Secondary symptoms: It is a symptom that directly impacts the life of an individual. These can be either motor or non-motor. Its effect depends on person to person. A very wide range of symptoms is associated with Parkinson's,. Besides these symptoms, there are some other symptoms found that lead to Parkinson's disease. These symptoms are micrographic, decreased olfaction & postural instability, slowing of the digestive system, constipation, fatigue, weakness, and Hypotension. Speech difficulties i.e. dysphonia (impaired speech production) and dysarthria (speech articulation difficulties) are found in patients with Parkinson's.

1.2 Purpose

The main aim is to predict the prediction efficiency that would be beneficial for the patients who are suffering from Parkinson and the percentage of the disease will be reduced. Generally in the first stage, Parkinson's can be cured by the proper treatment. 10 So it's important to identify the PD at the early stage for the

betterment of the patients. The main purpose of this research work is to find the best prediction model i.e. the best machine learning technique which will distinguish the Parkinson's patient from the healthy person. The techniques used in this problem are KNN, Naïve Bayes, and Logistic Regression. The experimental study is performed on the voice dataset of Parkinson's patients which is downloaded from the Kaggle. The prediction is evaluated using evaluation metrics like confusion matrix, precision, recall accuracy, and f1-score. The author used feature selection where the important features are taken into consideration to detect Parkinson's.

2. LITERATURE SURVEY

S.NO	JOURNAL NAME	AUTHORS	OBSERVATION
1	Bio composite MultipleUses foraNewApproach in theDiagnosisof Parkinson's Disease Using aMachine Learning Algorithm	AbdallahAl-Husban Mustafa Mahdi Abdulridha A.A.Hamad Mohamad	These results were obtained when the data set in the whole group was classified with the most relevant features of 45% (accuracy: 85%, sensitivity:0.94, specificity:00.78, measure: 0.86, kappa: 0.72, Acu: 0.86).

2	Imperative Role ofMachineLearning Algorithm for Detection of Parkinson's Disease: Review, Challenges and Recommendations	Arti Rana Ankur Dumka Rajesh Singh Manoj Kumar Panda Neeraj Priyadarshi	PD was obtained by L1-Norm SVM with K- fold cross- validation, with 99%; in handwritten patterns, it was obtained by bagging ensemble,with 97.96%; and for gaitanalysis, it was obtained by SVM with 100%. This reviewaddressedvarious challenges and also provided some futurerecommendations an dopportunities, as we observed that there is still a lot of work that has to be performed in thefuture.
3	Predicting SeveriOf Parkinson's DiseaseUsing Deep Learning.	Sridhti Grover SaloniBhartia Akshama Abhilasha Yadav Seeja K.R	we have used a dataset of 5875 instances, the accuracy of our approach can be further improved by implementing it on a larger dataset, having a greater number of instances of each severity class as well as on a combined database of patients' voice data and other patient .

4	Detection of Parkinson's Disease UsingMachine Learning Algorithm	Shrihari K Kulkarni K R Sumana	The results for algorithms based on accuracy are like this:Decision Tree 93.25%, Logistic Regression 91.25%, Naive Bayes 94.5%, and RNN88.75%.
5	Parkinson's Disease Detection using machine Learning Techniques	C K Gomathy	Wecanpredict the parkinson's disea se in patient's body using machine learning technology and this method makes the process easy to our user. Our analysis provides very accurate performance in detecting Parkinson's disease using XGBOOST algorithm
6	Parkinson disease onsetdetection UsingMachine Learning	Sonia Singla	Parkinson disease data and find out XGBoost is the best Algorithm to predict the onset of the disease which will enable early treatment and savea life.
7	A Survey of Detection of Parkinson Disease using DeepLearning Technique	Sakshi Jadhav Seema Thorat Sakshi Fokane Rahul Chakre	The work is mainly focusing on advancement of predictive models to achieve good accuracy in predicting valid disease outcomes using deep learning methods like prediction based on Artificial Neural Network (ANN). In thispaper, Deep Learningtechniques are proposed for theprediction of Parkinson Disease in early stage.
8	Machine Learning for theDiagnosis of	Jei Mei Christian Deserosiers	Machine learning approaches therefore have the potential to provide clinicians with

	Parkinson's Disease		additional tools to screen, detect or diagnose PD.
9	Parkinson’s disease ratingscales	Jamir Pitton Rissardo AnaLeticiaFornari Caprara	Instead of developing new scales, the studies should focus on the clinimetric evaluation, and assess these scales in different ethnic origins translating the scales and evaluating their quality.
10	Parkinson Disease diagnosis using machine learning	Anila M Dr. G.Pradeepini	Developing a very faster classifier using novel architecture of neural network combined with specific approach may work better.

2.1 Existing problem :

The main idea behind the implementation is to classify a person as Healthy or having Parkinson’s disease by looking at the Spiral Drawing made by the person. The Spiral Drawing created by The project aims at presenting a solution for Parkinson’s disease detection using Spiral Drawings and CNN. a healthy person will look almost similar to a standard spiral shape. However, a spiral drawn by a person with Parkinson’s disease will highly deviate from a perfect spiral shape and look distorted due to slow motor movements and decreased coordination between hand and brain**.**

2.2 References :

1.Mahlknecht, P.; Krismer, F.; Poewe, W.; Seppi, K. Meta-Analysis of Dorsolateral Nigral Hyperintensity on MagneticResonance Imaging as a Marker for Parkinson’s Disease.Mov. Disord. 2017, 32, 619–623.

2. Dickson, D.W. Neuropathology of Parkinson disease.Parkinsonism Relat. Disord. 2018, 46 (Suppl. 1), S30–S33.

3. Aich, S., Sain, M., Park, J., Choi, K.W. and Kim, H.C., 2017, November.A mixed classification approach for the prediction of Parkinson's disease

4. M. Abdar and M. Zomorodi-Moghadam, "Impact of Patients' Gender on Parkinson's disease using Classification Algorithms" Journal of AI and Data Mining, vol. 6, 2018.

5. Satish Srinivasan, Michael Martin & Abhishek Tripathi, "ANN based Data Mining Analysis of Parkinson's Disease" International Journal of Computer Applications, vol. 168, June 2017.

6. Ramzi M. Sadek et al., "Parkinson's Disease Prediction using Artificial Neural Network" International Journal of Academic Health and Medical Research, vol. 3, Issue 1, January 2019.

7. Arvind Kumar Tiwari, "Machine Learning based Approaches for Prediction of Parkinson's Disease", Machine Learning and Applications: An International Journal (MLAU) vol. 3, June 2016.

8. Dr. Anupam Bhatia and Raunak Sulekh, "Predictive Model for Parkinson's Disease through Naive Bayes Classification" International Journal of Computer Science & Communication vol. 9, Dec. 2017, pp. 194- 202, Sept 2017 - March 2018.

9. Dragana Miljkovic et al, "Machine Learning and Data Mining Methods for Managing Parkinson's Disease" LNAI 9605, pp 209-220, 2016.

10. R. P. Duncan, A. L. Leddy, J. T. Cavanaugh et al., "Detecting and predicting balance decline in Parkinson disease: a prospective cohort study," Journal of Parkinson's Disease, vol. 5, no. 1, pp. 131–139, 2015.

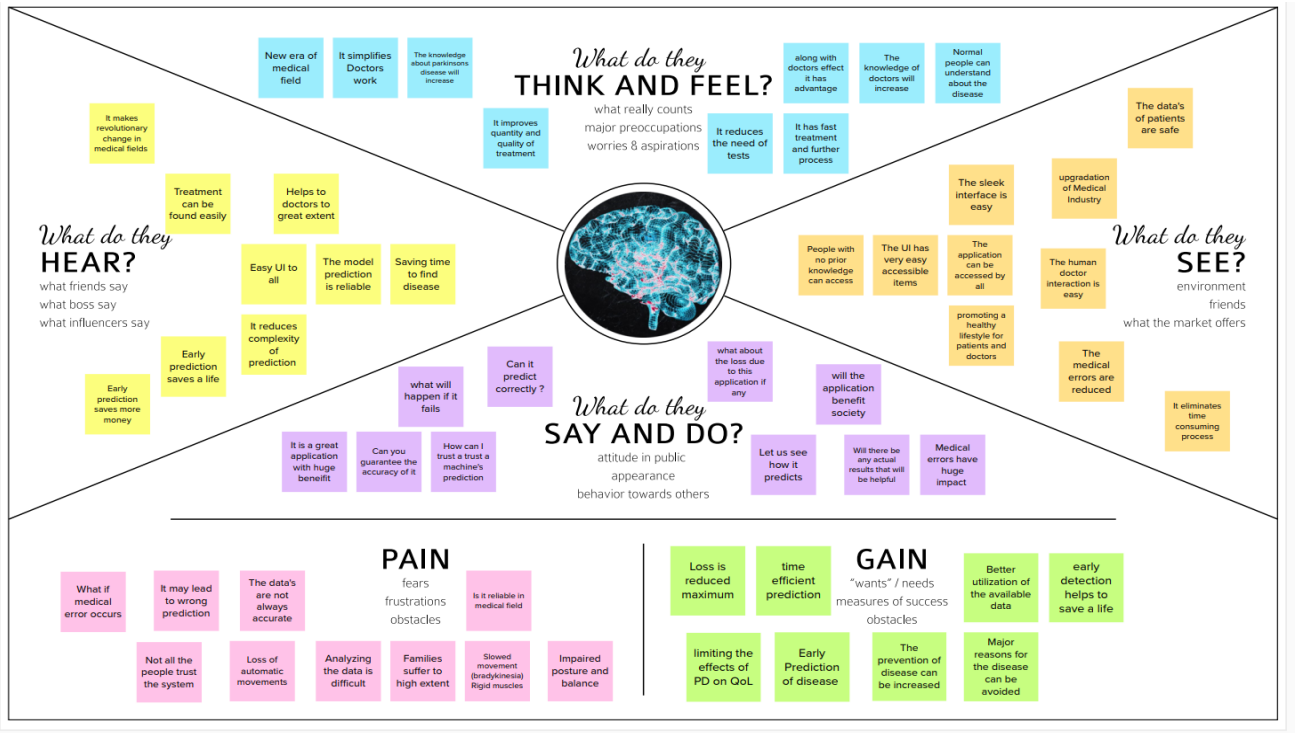
2.3 Problem Statement :

The main aim is to predict the prediction efficiency that would be beneficial for the patients who are suffering from Parkinson and the percentage of the disease will be reduced. Generally in the first stage, Parkinson's can be cured by the proper treatment. 10 So it‘s important to identify the PD at the early stage for the betterment of the patients. The main purpose of this research work is to find the best prediction model i.e. the best machine learning technique which will distinguish the Parkinson’s patient from the healthy person. The techniques used in this problem are KNN, Naïve Bayes, and Logistic Regression. The experimental study is performed on the voice dataset of Parkinson’s patients which is downloaded from the Kaggle. The prediction is evaluated using evaluation metrics like confusion matrix, precision, recall accuracy, and f1-score. The author used feature selection where the important features are taken into consideration to detect Parkinson’s.

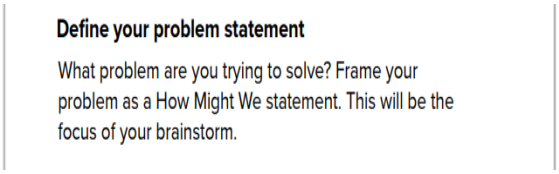
3. IDEATION & PROPOSED SOLUTION

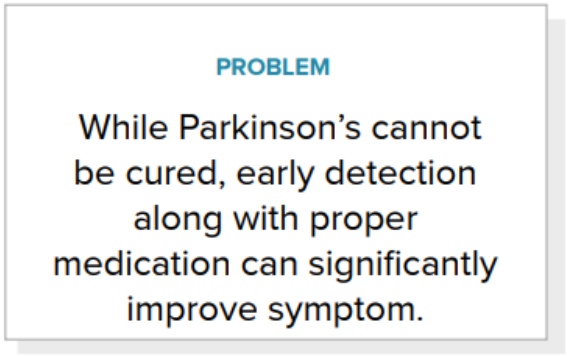
3.1 Empathy Map Canvas :

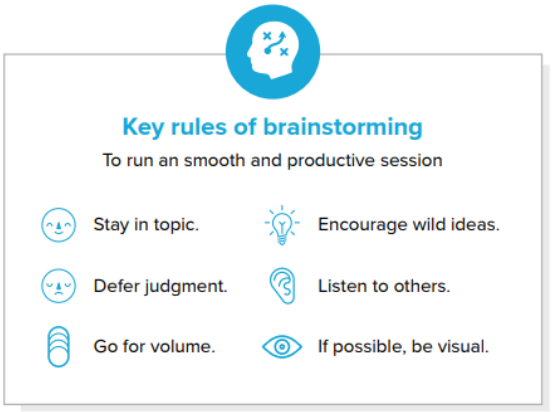
More than 10 million people are living with Parkinson’s Disease worldwide, according to the Parkinson’s Foundation. While Parkinson’s cannot be cured, early detection along with proper medication can signifcantly improve symptoms and quality of life

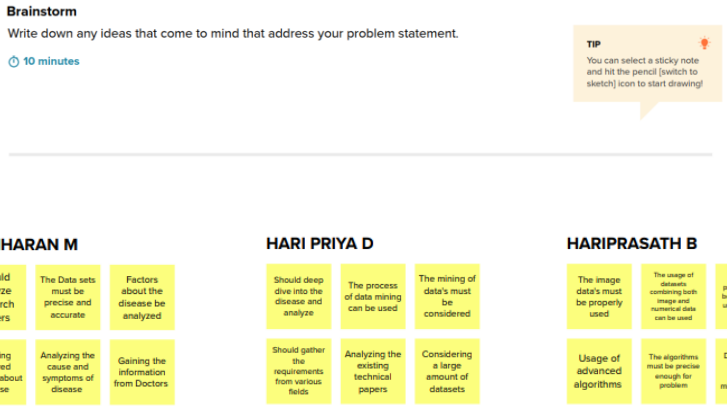


3.2 Ideation & Brainstorming :







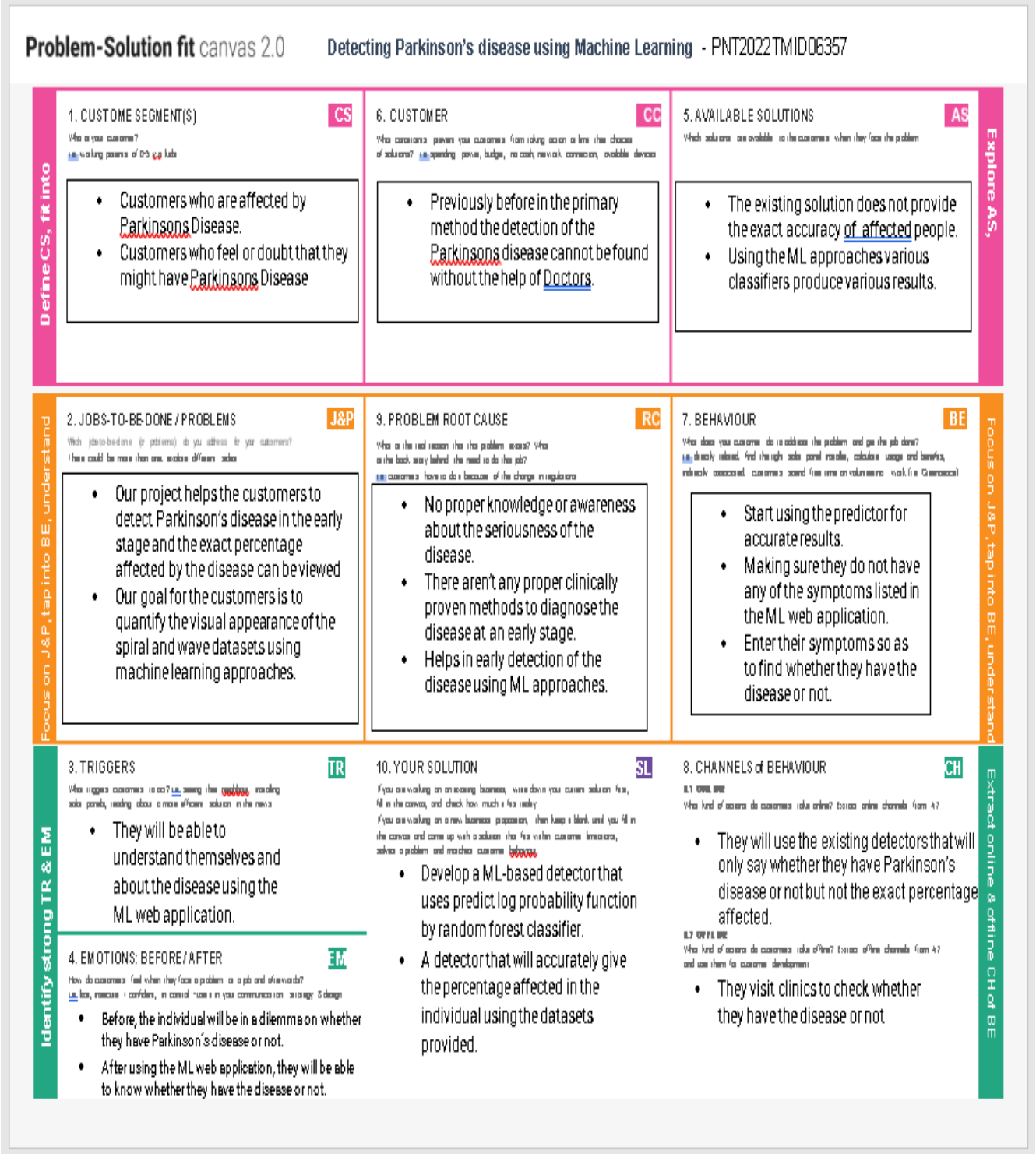


3.3 Proposed Solution :

S.NO	Parameter	Description
1.	Problem Statement (Problem to be solved)	More than 10 million people are living with Parkinson's Disease worldwide, according to the Parkinson 's Foundation. While Parkinson's cannot be cured, early detection along with proper medication can significantly improve symptoms and quality of life.
2.	Idea / Solution description	Prediction of Parkinson's disease with higher accuracy and estimation using web application which will help stakeholders such as the government and health insurance companies . It can identify patients at risk of disease or health conditions.
3.	Novelty / Uniqueness	It identifies patients at risk of disease or health conditions at early stages . The use of OpenCV techniques to eliminate even the use of paper for the drawing test also contributes to the novelty factor. The application in case of a prediction leaning to a confirmation of the condition can provide awareness and various information about the condition including location and other details of treatment centres and specialists. Since the application must work with the patients physical and personal information, the security factor

		is of paramount importance. The usage of OTP verified authentication means is a novelty factor
4.	Social Impact / Customer Satisfaction	The prediction of disease can effectively control and prevent large scale outbreaks and epidemics .It also detects the abnormal distribution of disease in prior and hence the customers could be saved from traumas which will lead them to the road of happiness.
5.	Business Model (Revenue Model)	The platform is free. It can used by people pertaining to all age groups with limited technical knowledge and can be prescribed to others .The model helps to increase economic status and has easy user interface.
6.	Scalability of the Solution	Highly scalable .Produces accurate results with small and large amount of data . It may accessed by any number of people and advancement of chatbots can be introduced

3.4 Problem Solution fit :



4.REQUIREMENT ANALYSIS

4.1 Functional Requirement :

S.NO	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
1	User Authentication	The users must be registered first and can be only able to access the web application . This is to ensure that the web application is used for a good reason.
2	Web Service Management Process	Web Service Management process by Web Portal admin in registering web client to do SSO or member data communication. The web page is hosted in cloud.
3	Data Management	The Web server and Portal manager can have access to data to edit and update again to server.
4	Testing	Applying the algorithms on the test data
5	Confirmation	Display the result with the description of having Parkinson’s or not

4.2 Non-Functional Requirements :

S.NO	Non-Functional Requirements	Description
1	Usability	The webpage loading for users submitting their image input details at the web application must be loaded fast than rendering more time.
2	Security	Authorization access scenarios and definitions, hand-over procedures for patient records. The image and other inputs of patients must be highly secured and can't be accessible to others.
3	Reliability	The prediction of the system must be with higher accuracy so that the output from the application can be trusted by the users without any doubts and can be used for further dragonising process with Doctors.
4	Performance	The landing page supporting 5,000 users per hour must provide 6 second or less response time in a Chrome desktop browser, including the rendering of text and images and over an LTE connection and the uploading of Data (image) must

		also should be fast and the output page should be rendered within seconds.
5	Availability	The web application should be available to all Doctors across the globe and can be implemented in every hospital so that the patients can use it effectively .
6	Scalability	The System must function using Cloud and during a down process also it must satisfy the maximum number of clients. . The system must use higher RAM and CPU processing in Server to handle multiple request at same time.

5. PROJECT DESIGN

5.1 Data Flow Diagrams :

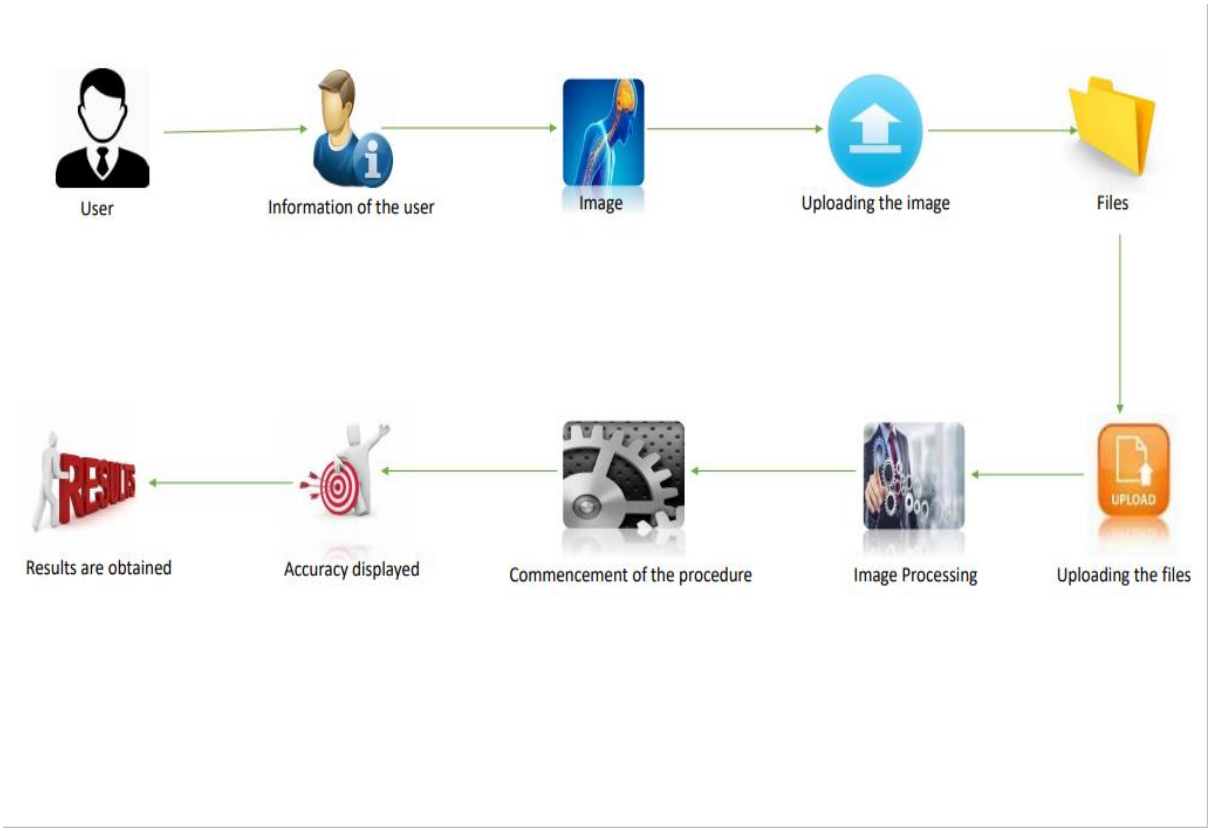
• DFD is the abbreviation for Data Flow Diagram

• The flow of data of a system or a process is represented by DFD.

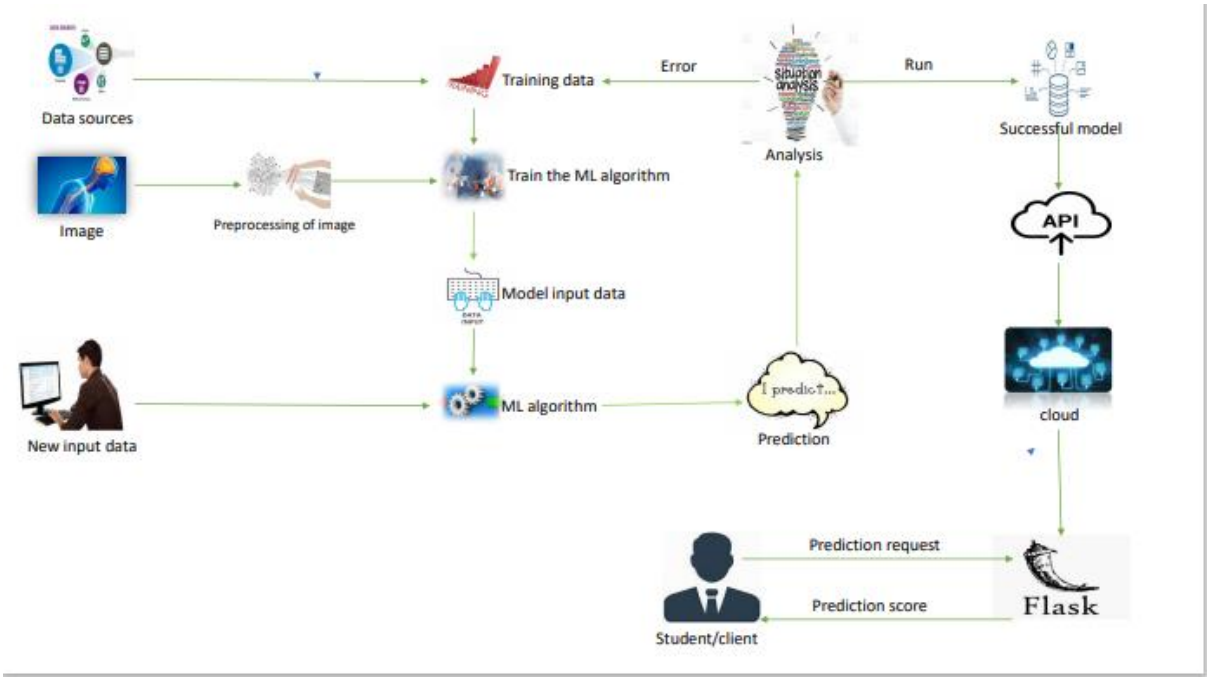
• It also gives insight into the inputs and outputs of each entity and the process itself

• DFD does not have control flow and no loops or decision rules are present

• Specific operations depending on the type of data can be explained by a flowchart



5.2 Solution& Technical Architecture:



5.3 User Stories:

User type	Functional requirement (Epic)	User story number	User story/task	Acceptance criteria	Priority	Release
User	Account creation	USN-1	User can connect to the application	User can access the account being created	High	Sprint-1
Input data	Adding data	USN-2	Inputs can be given to the system for its learning purposes	Data entered could be verified by the user	High	Sprint-1
Data validation	Checking accuracy	USN-3	Ability and accuracy of the model can be checked by the user	On logging in to account , the capability could be checked	Medium	Sprint-2
Classification	Data classification	USN-4	Data can be viewed by the user	Verify the user data with real data	Medium	Sprint-2
App work	Work flow	USN-5	Working action of the application model could be viewed	Application working and responses to the actions imposed can be viewed	Medium	Sprint-2
Image classification	Checking for the disease	USN-6	With the help of trained and test data ,user can verify with application that the image is identified with the actual image	User can confirm that the data shows accurate results	Low	Sprint-3
User interaction	AI-Powered chatbot	USN-7	User can interact with the automated chatbot to engage my time till the application processed the accurate result	Results could be viewed from the interactions from the chatbot	Low	Sprint-3
Medical assistance	Medical suggestions	USN-8	User can get medical advises	Enough assistance could be obtained	High	Sprint-3
Data extraction	Obtaining the data	USN-9	User can retrieve the result data from the application for data storage	Result could be downloaded in the form data to be shown to the medical teams	Medium	Sprint-4

6.PROJECT PLANNING & SCHEDULING:

6.1 Sprint Planning & Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Tas	Story Points	Priority	Team Members
Sprint-2	homepage	USN-1	Description about Parkinson's disease.	8	Low	Hariharan M Raja M
Sprint-2		USN-2	Details about the test vitals required for the testing.	13	Low	Raja M Hariprasath B
Sprint-3	Registration	USN-3	As a user, I can register for the application by entering my username, email, phone number, and password, and confirming my password	5	Medium	Hari Priya D Hariharan M
Sprint-3	Login	USN-4	As a user, I can log in to the web application by entering my email id & password.	5	Medium	Hari Priya D Raja M
Sprint-3	Main Page(Test vitals)	USN-5	As a user, I submit the required image for the prediction	5	Medium	Hariharan M Hariprasath B

Sprint-3	Result	USN-6	Results will be displayed	8	High	Hariprasath B Raja M Hari Priya D
Sprint-1	Data collection	USN-7	Collect the required data for the detection of Parkinson's disease	8	High	Hariharan M Hari Priya D Raja M
Sprint-1	Data preprocessing	USN-8	Clean and analyze the data to avoid noise and duplications	8	High	Hari Priya D Hariharan M Hariprasath B
Sprint-1	Model Building	USN-9	Build the model using a Random forest classifier to classify the images.	5	High	Raja M Hariprasath B Hariharan M
Sprint-4	Deploy the model	USN-10	Deployment of ML model using IBM Watson Studio, object storage.	13	High	Hariharan M Hari Priya D Hariprasath B Raja M
Sprint-4	Integrate the web app with the IBM model	USN-11	Use flask for the integration purpose	8	Medium	Hari Priya D Hariprasath B

6.2 Sprint Delivery Schedule :

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	21	6 Days	04 Nov 2022	29 Oct 2022	21	29 Oct 2022
Sprint-2	21	6 Days	31 Oct 2022	05 Nov 2022	21	02 Nov 2022
Sprint-3	21	6 Days	07 Nov 2022	12 Nov 2022	21	12 Nov 2022
Sprint-4	21	6 Days	18 Nov 2022	19 Nov 2022	21	18 Nov 2022

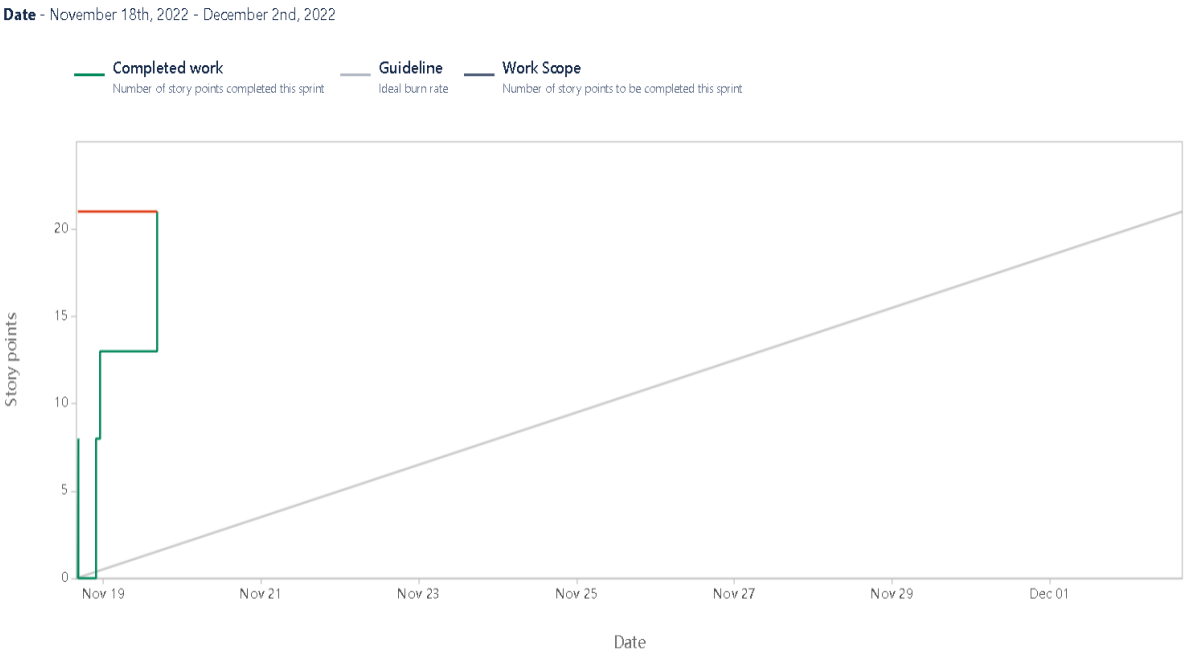
Velocity: we have a 6-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day).

AV = sprint duration = 21/6 = 3.5 velocity.

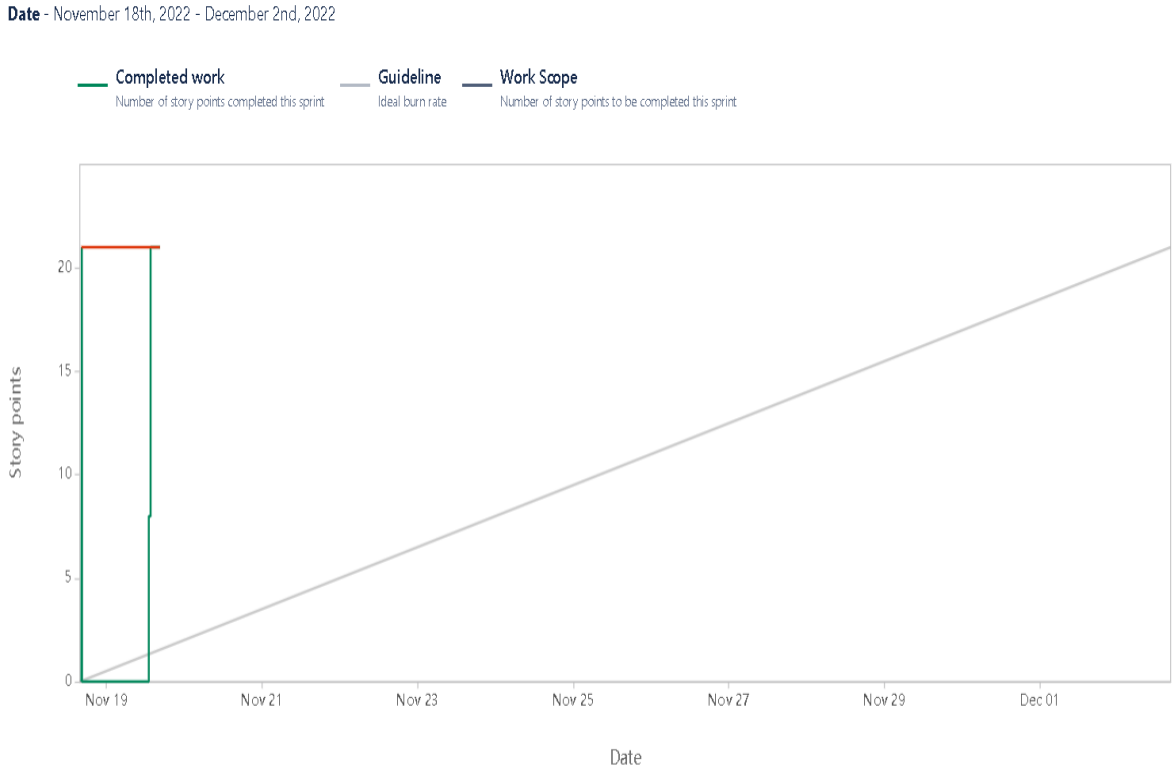
6.3 Reports from JIRA :

Burnup Report

Sprint 1 :

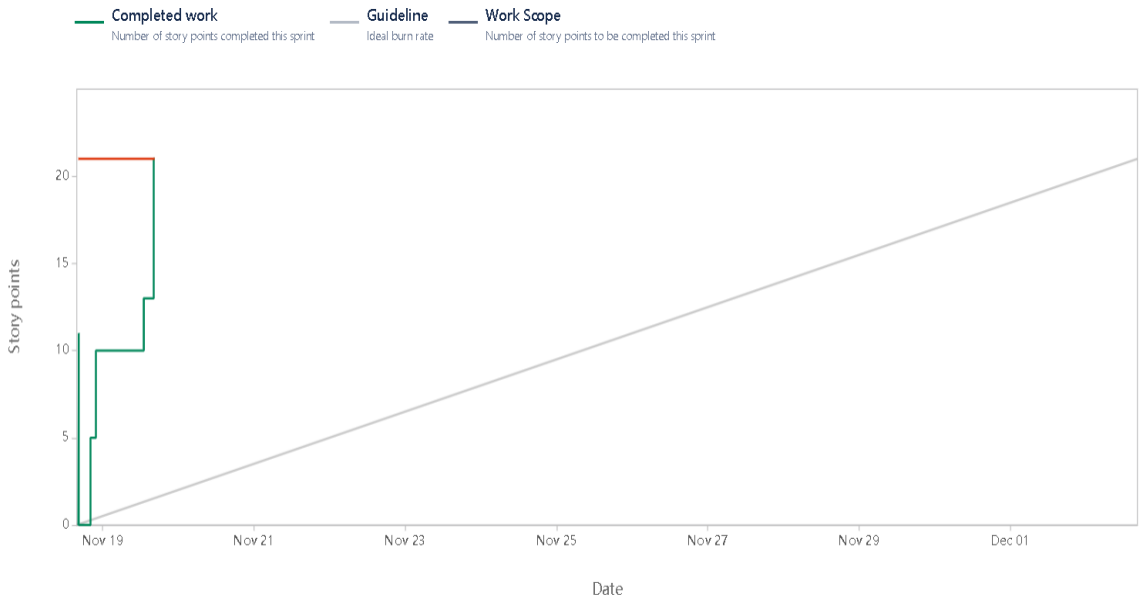


Sprint 2 :



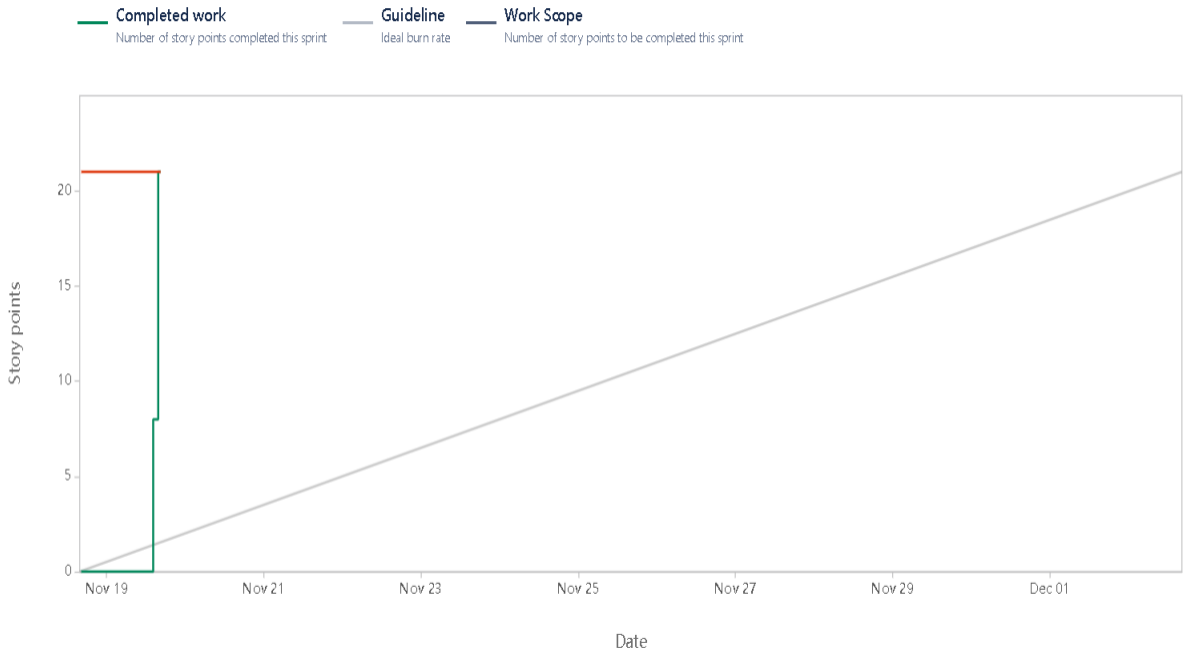
Sprint 3 :

Date - November 18th, 2022 - December 2nd, 2022



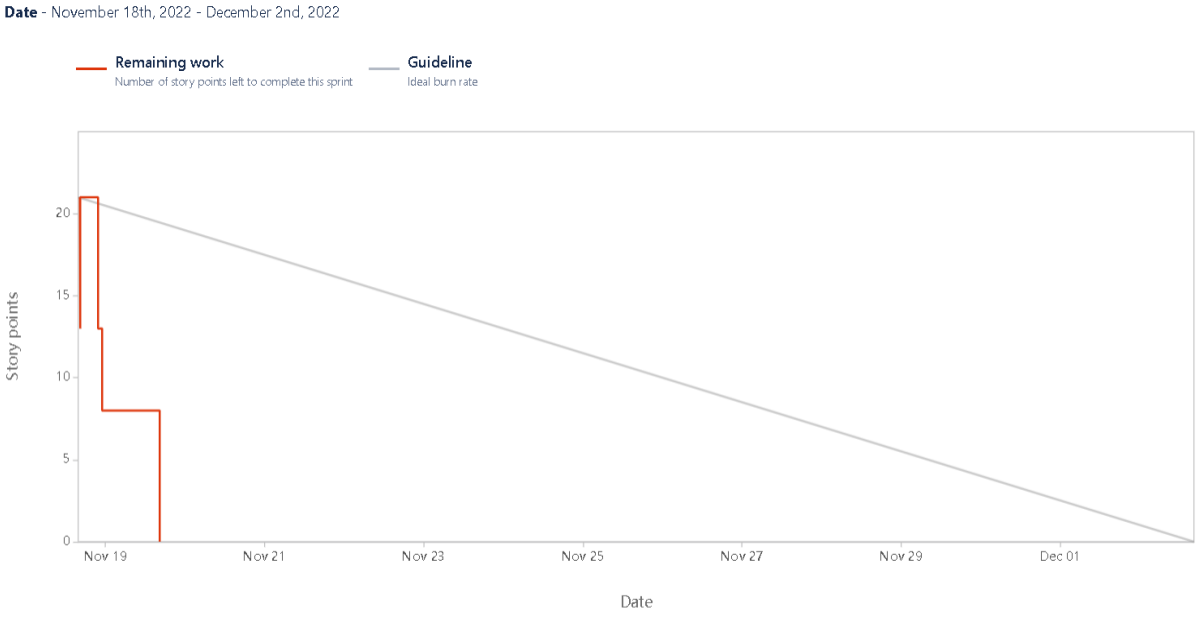
Sprint 4 :

Date - November 18th, 2022 - December 2nd, 2022

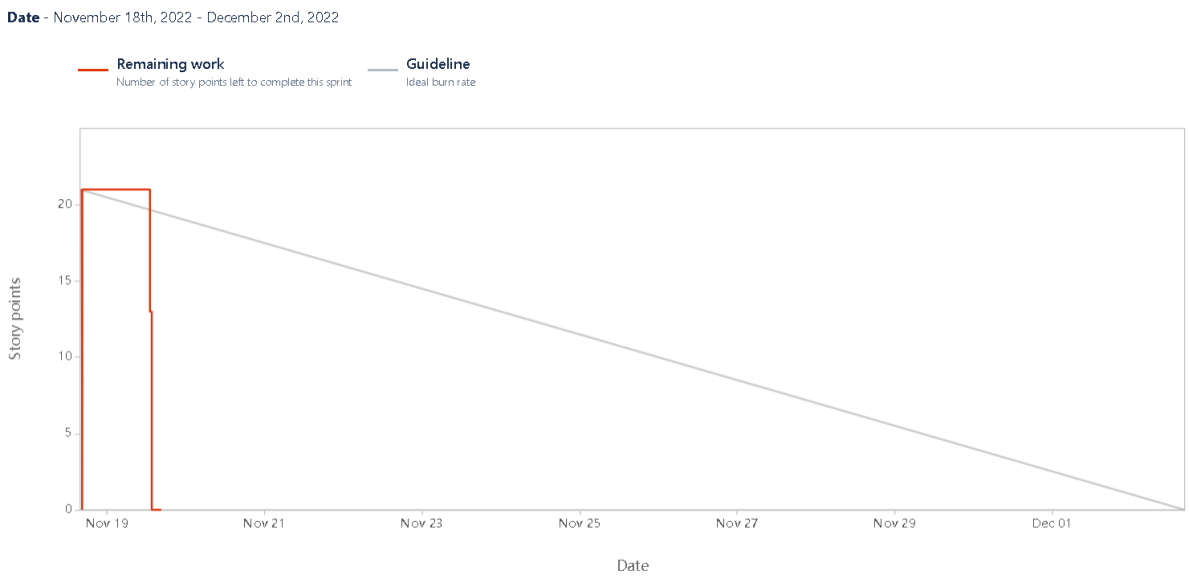


Sprint burndown chart

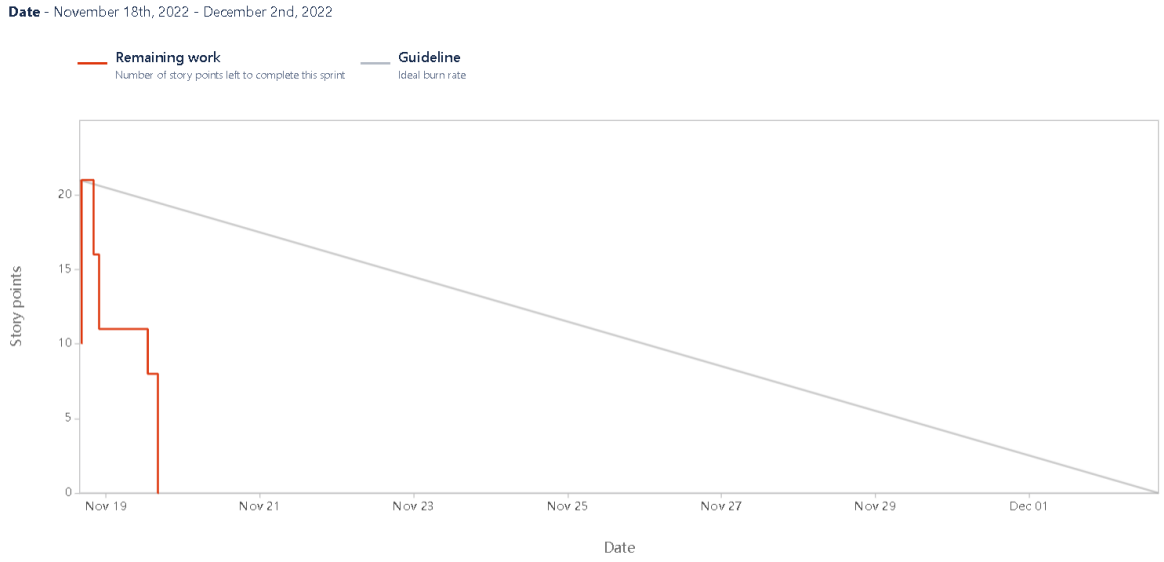
Sprint 1 :



Sprint 2 :



Sprint 3 :

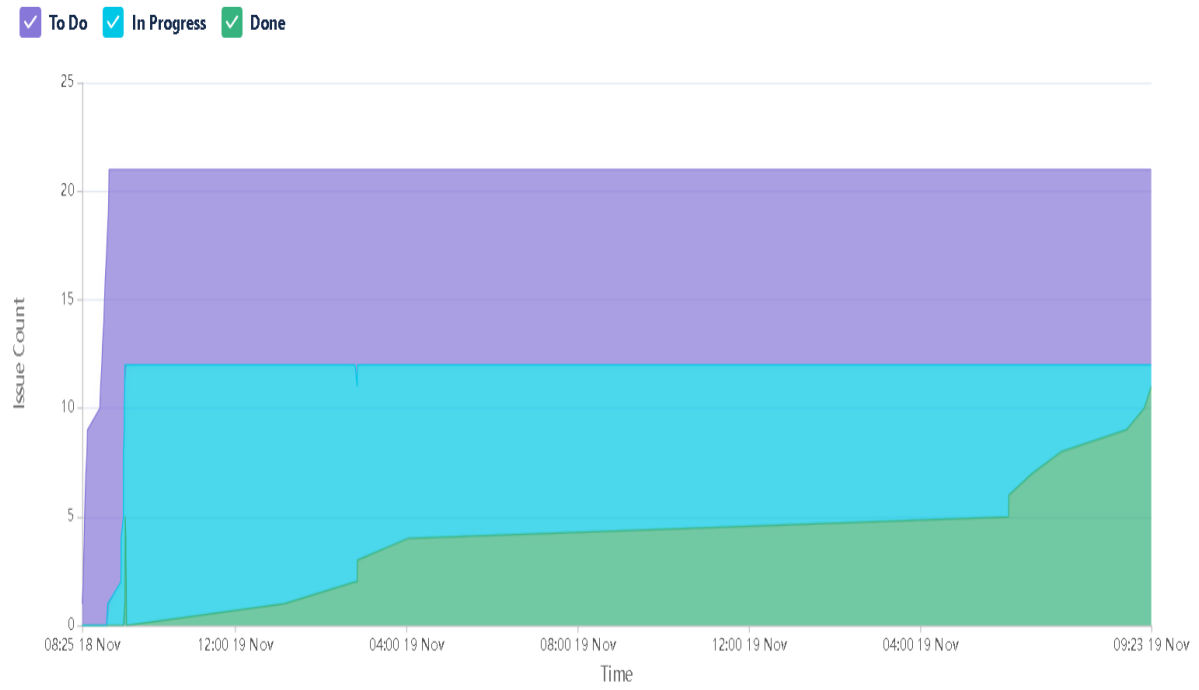


Sprint 4:

Velocity report

Sprint	Commitment	Completed
PDP Sprint 1	21	21
PDP Sprint 2	21	21
PDP Sprint 3	21	21
PDP Sprint 4	21	21

Cumulative flow diagram



7.CODINGS &SOLUTIONING (Explain the features added in the project along with code)

7.1 Features 1

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import zipfile as zf
import os
import random
import cv2
import pickle
from imutils import build_montages
from imutils import paths
from sklearn.metrics import classification_report,confusion_matrix
from sklearn import metrics
from sklearn.preprocessing import LabelEncoder,LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier,ExtraTreesClassifier
from skimage import feature
#from google.colab.patches import cv2_imshow
```

```
sns.set()
os.getcwd()
```

```
'C:\\Users\\harih\\OneDrive\\Desktop\\IBM Project'
```

```
handle_spiral = zf.ZipFile(r'dataset.zip')
handle_spiral.extractall('dataset')
handle_spiral.close()
```

```
spiral_train_healthy = os.listdir('dataset/dataset/spiral/training/healthy/')
spiral_train_park = os.listdir('dataset/dataset/spiral/training/parkinson/')

fp_spiral_train_healthy = 'dataset/dataset/spiral/training/healthy/'
fp_spiral_train_park = 'dataset/dataset/spiral/training/parkinson/'

spiral_test_healthy = os.listdir('dataset/dataset/spiral/testing/healthy/')
spiral_test_park = os.listdir('dataset/dataset/spiral/testing/parkinson/')

fp_spiral_test_healthy = 'dataset/dataset/spiral/testing/healthy/'
fp_spiral_test_park = 'dataset/dataset/spiral/testing/parkinson/'
```

```
def quantify_image(image):
    features = feature.hog(image,orientations=9,
                           pixels_per_cell=(10,10),cells_per_block=(2,2),transform_sqrt=True,block_norm="L1")

    return features
```

```
trainX = []
testX = []
outputs = []
trainY = []
testY = []

for i in spiral_train_healthy:
  image = cv2.imread(fp_spiral_train_healthy+i)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image =cv2.threshold(image, 0, 255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  trainX.append(features)
  trainY.append('healthy')

for i in spiral_train_park:
  image = cv2.imread(fp_spiral_train_park+i)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  trainX.append(features)
  trainY.append('parkinson')


for i in spiral_test_healthy:
  image = cv2.imread(fp_spiral_test_healthy+i)
  outputs.append(image)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  testX.append(features)
  testY.append('healthy')

for i in spiral_test_park:
  image = cv2.imread(fp_spiral_test_park+i)
  outputs.append(image)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  testX.append(features)
  testY.append('parkinson')
```

```
    trainX = np.array(trainX)
    testX = np.array(testX)
    trainY = np.array(trainY)
    testY = np.array(testY)
    trainX
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

```
    trainY
```

```
array(['healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson'], dtype='<U9')
```

```
testX
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

```
testY
```

```
array(['healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
       'healthy', 'healthy', 'healthy', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
       'parkinson', 'parkinson', 'parkinson'], dtype='<U9')
```

```
le = LabelEncoder()
trainY = le.fit_transform(trainY)
testY = le.transform(testY)
print(trainX.shape,trainY.shape)
```

```
(72, 12996) (72,)
```

```
trainY
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1], dtype=int64)
```

```
testY
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1])
```

```
print("Training model....")
model = RandomForestClassifier(n_estimators=100)
model.fit(trainX,trainY)
```

```
Training model....

RandomForestClassifier()
```

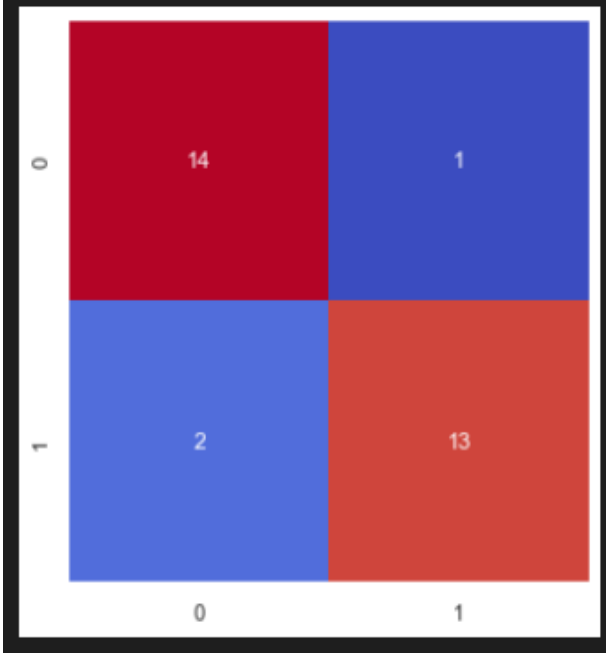
```
    preds = model.predict(testX)
    preds
```

```
array([0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 0, 1, 1, 1], dtype=int64)
```

```
    cnf = confusion_matrix(testY,preds)
    cnf
```

```
array([[14,  1],
       [ 2, 13]], dtype=int64)
```

```
    plt.figure(figsize=(5,5))
    sns.heatmap(cnf , annot=True , cmap="coolwarm" , cbar=False)
    plt.show()
```



```
    acc = metrics.accuracy_score(testY,preds)
    acc
```

```
0.9
```

```
    indexes = np.random.randint(0,30,25)
    indexes
```

```
array([17, 20, 21,  0, 22, 23, 19, 22,  6, 19, 18, 20, 19,  1,  4, 26,  4,
       18, 25, 24, 14, 20,  9, 11, 25])
```

```
    testpath=list(paths.list_images(fp_spiral_train_healthy))
    idxs=np.arange(0,len(testpath))
    idxs=np.random.choice(idxs,size=(25,),replace=False)
    images=[]

    for i in idxs:
        image=cv2.imread(testpath[i])
        output=image.copy()
        output=cv2.resize(output,(128,128))
        image=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
        image=cv2.resize(image,(200,200))
        image=cv2.threshold(image,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]

        features= quantify_image(image)
        preds=model.predict([features])
        label=le.inverse_transform(preds)[0]
        if label=="healthy":
          color=(0,255,0)
        else:
          (0,0,255)
        cv2.putText(output,label, (3,20),cv2.FONT_HERSHEY_SIMPLEX,0.5,color,2)
        images.append(output)
```

```
    montage=build_montages(images,(128,128),(5,5))[0]
    cv2.imshow('',montage)
    cv2.waitKey(0)
```

```
-1
```

```
predictions = model.predict(testX)

cm = confusion_matrix(testY, predictions).flatten()
print(cm)
(tn, fp, fn, tp) = cm
accuracy = (tp + tn) / float(cm.sum())
print(accuracy)
```

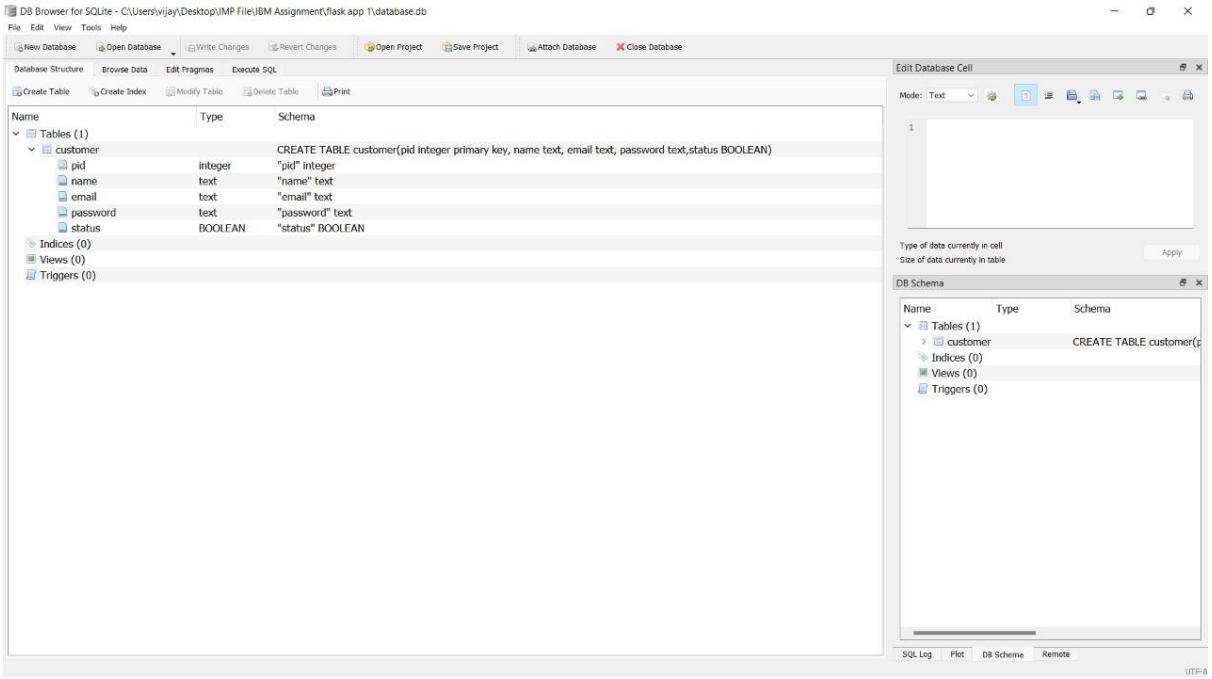
```
[14  1  2 13]
0.9
```

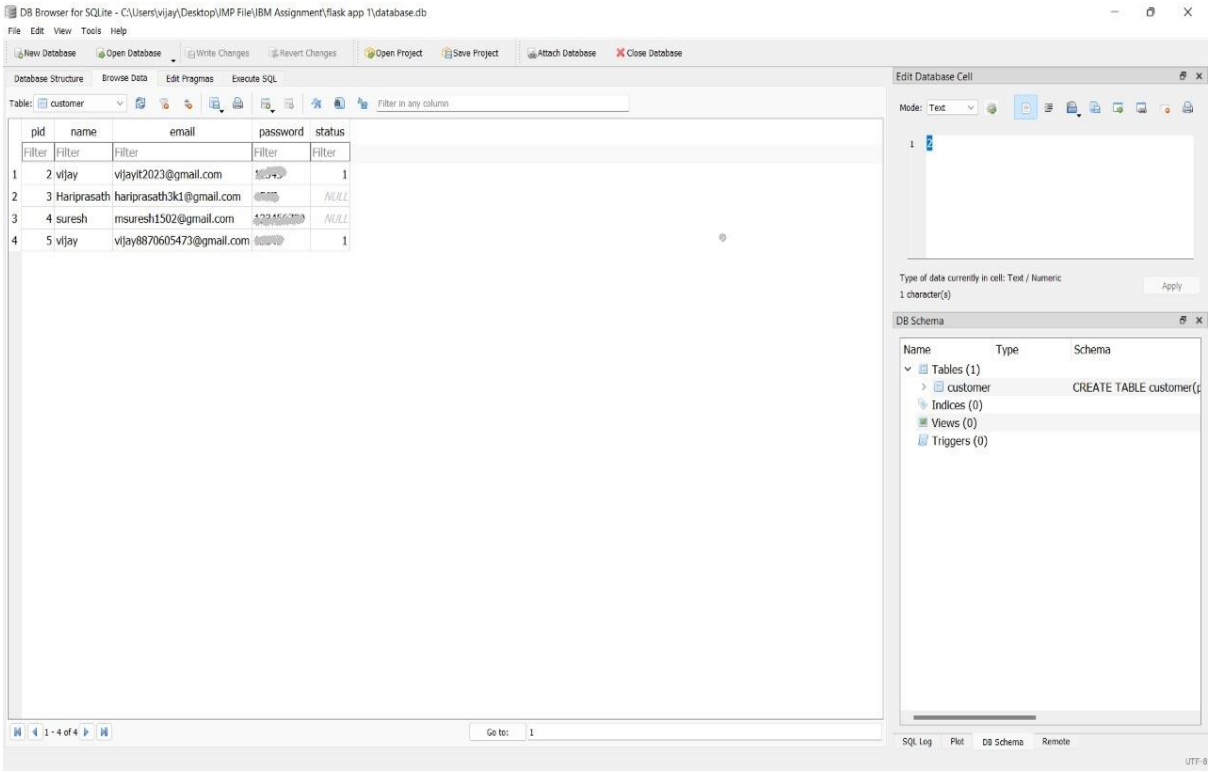
```
pickle.dump(model,open('parkinson.pkl','wb'))
```

7.2 Features 2

```
@app.route('/upload', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        f = request.files['file2']  # requesting the file
        #filename_secure = secure_filename(f.filename)
        basepath = os.path.dirname(
            '__file__')  # storing the file directory
        # storing the file in uploads folder
        filepath = os.path.join(basepath, "uploads", f.filename)
        f.save(filepath)  # saving the file

        # Loading the saved model
        print("[INFO] loading model...")
        model = pickle.loads(open('parkinson.pkl', "rb").read())
        ''' local_filename = "./uploads/"
        local_filename += filename_secure
        print(local_filename)'''

        # Pre-process the image in the same manner we did earlier
        image = cv2.imread(filepath)
        output = image.copy()

        # Load the input image, convert it to grayscale, and resize
        output = cv2.resize(output, (128, 128))
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        image = cv2.resize(image, (200, 200))
        image = cv2.threshold(image, 0, 255,
                              cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]

        # Quantify the image and make predictions based on the extracted features using the last trained Random Forest
        features = feature.hog(image, orientations=9,
                               pixels_per_cell=(10, 10), cells_per_block=(2, 2),
                               transform_sqrt=True, block_norm="L1")
        preds = model.predict([features])
        print(preds)
        ls = ["healthy", "parkinson"]
        result = ls[preds[0]]
        '''color = (0, 255, 0) if result == "healthy" else (0, 0, 255)
        cv2.putText(output, result, (3, 20),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
        cv2.imshow("Output", output)
        cv2.waitKey(0)'''
        if result =="healthy":
          return redirect("healthy")
        else:
          return redirect("parkinson")
```

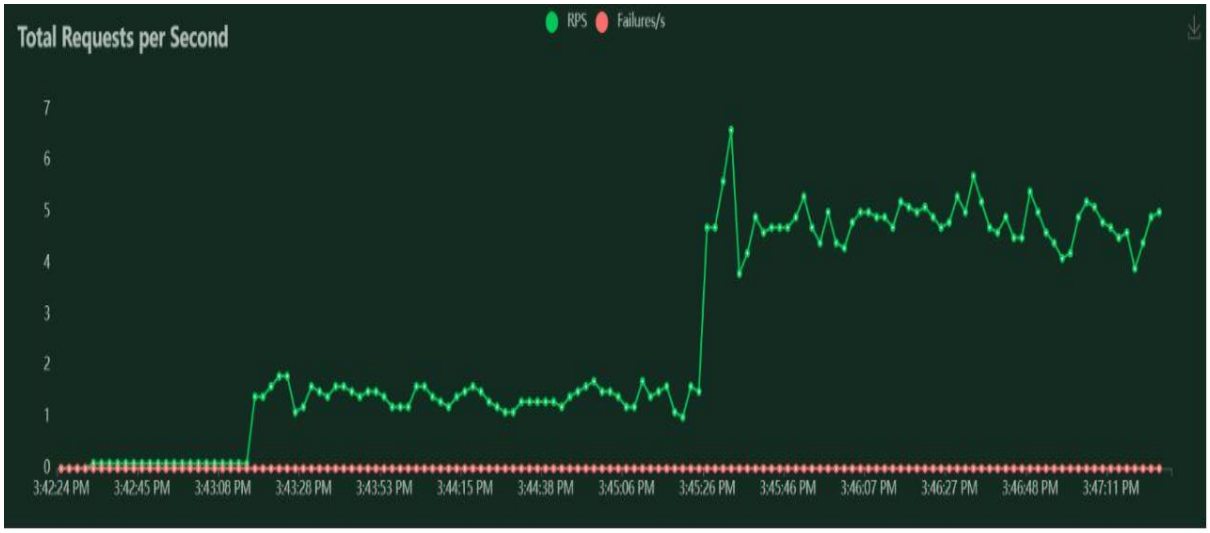
7.3 Database schema (if applicable)

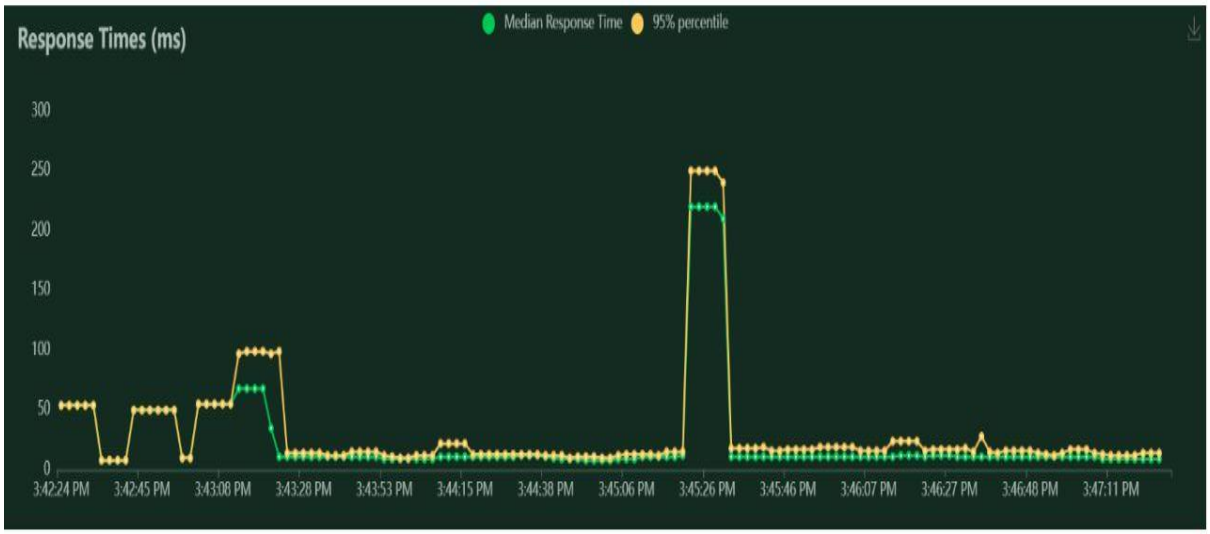


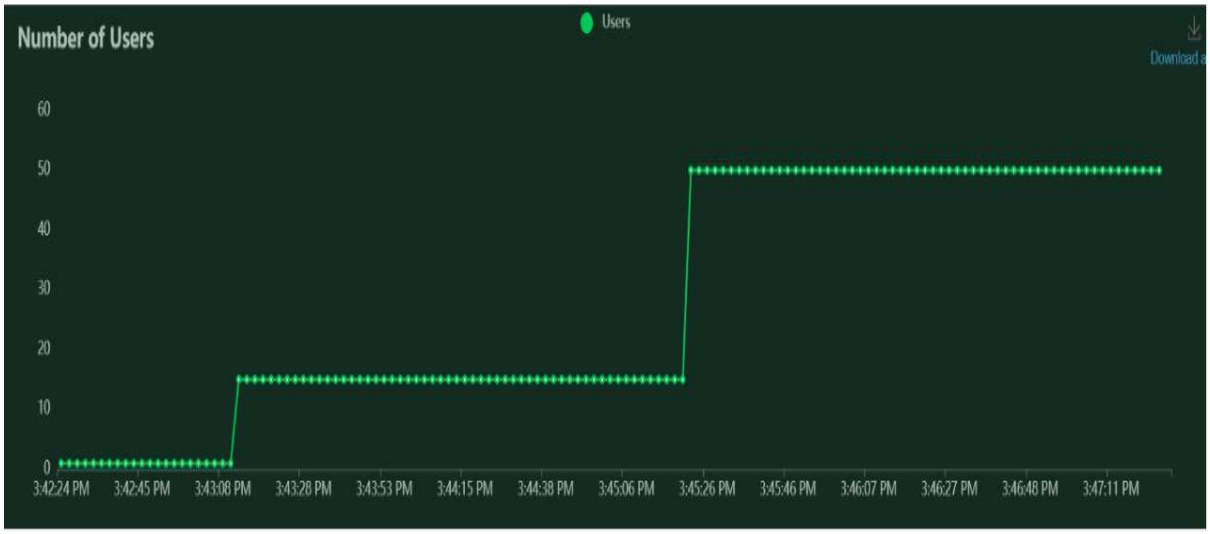


8.TESTING

8.1 Test cases







Locust Test Report

During: 18/11/2022, 3:42:15 pm - 18/11/2022, 3:47:27 pm

Target Host: http://127.0.0.1:5000

Script: locust.py

Request Statistics

Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	/	131	0	16	5	221	1632	0.4	0.0
GET	/contact	127	0	30	5	241	4007	0.4	0.0
GET	/explore	115	0	18	4	260	2233	0.4	0.0
GET	/home	104	0	26	5	252	1182	0.3	0.0
GET	/images	127	0	18	5	240	1207	0.4	0.0
GET	/prediction	134	0	20	5	229	380	0.4	0.0
GET	/signup	80	0	19	5	237	2021	0.3	0.0
	Aggregated	**818**	**0**	**21**	**4**	**260**	**1794**	**2.6**	**0.0**

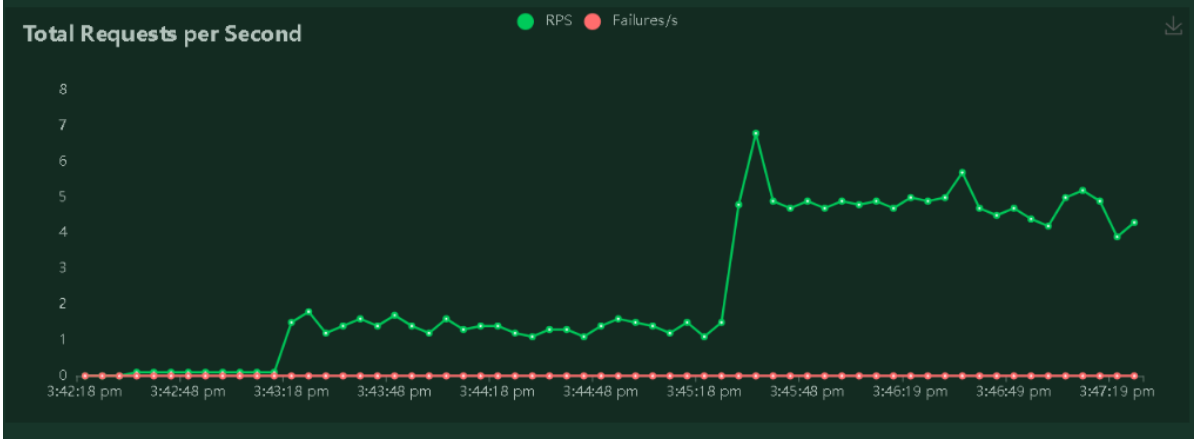
Response Time Statistics

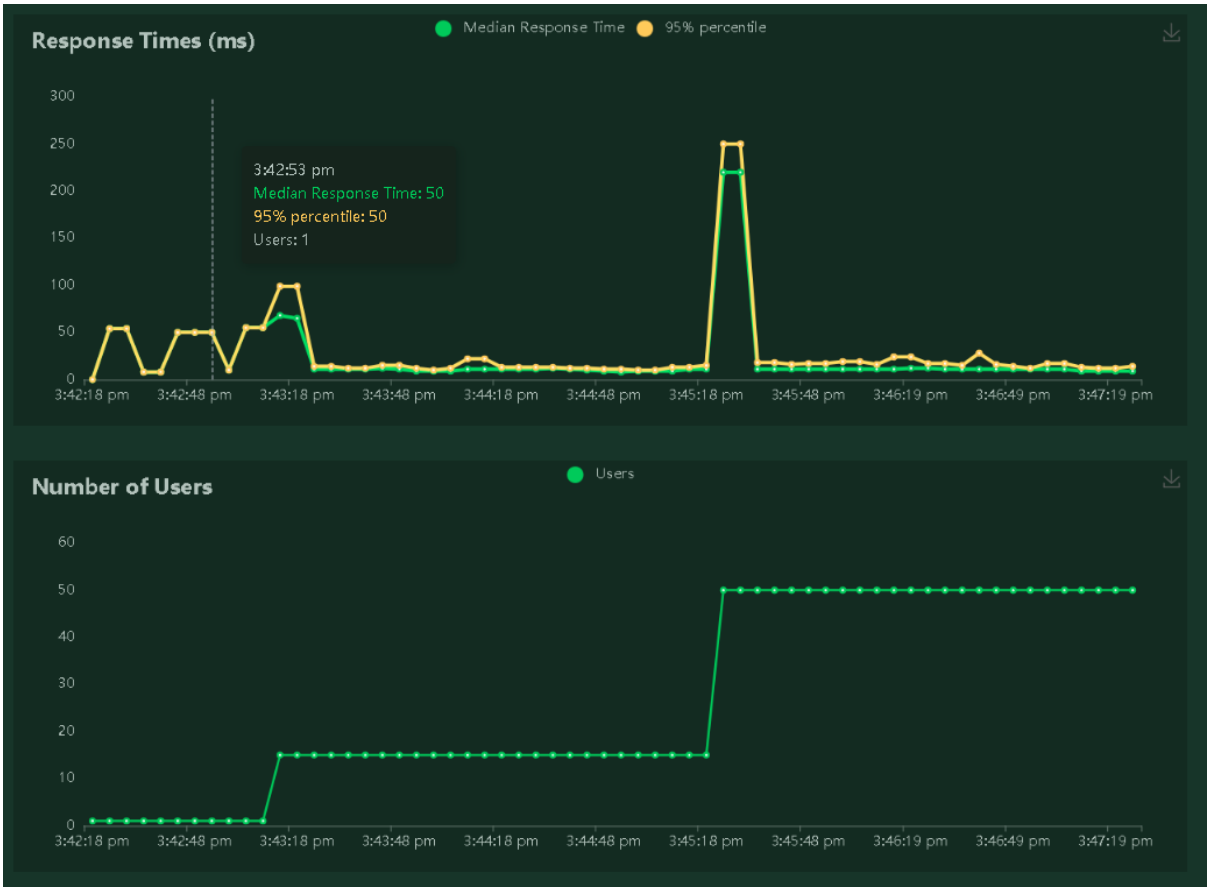
Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	/	11	11	11	12	14	17	220	220
GET	/contact	11	11	12	13	88	220	230	240
GET	/explore	11	11	12	13	16	22	240	260
GET	/home	11	11	12	14	20	220	250	250
GET	/images	11	11	12	13	17	32	220	240
GET	/prediction	11	11	11	13	15	65	230	230
GET	/signup	11	11	12	13	15	50	240	240
	Aggregated	**11**	**11**	**12**	**13**	**17**	**75**	**230**	**260**

Exceptions Statistics

Count	Message	Traceback	Nodes
40	tuple index out of range	File "C:\Users\harih\anaconda3\lib\site-packages\locust\user\task.py", line 347, in run self.execute_next_task() File "C:\Users\harih\anaconda3\lib\site-packages\locust\user\task.py", line 372, in execute_next_task self.execute_task(self._task_queue.pop(0)) File "C:\Users\harih\anaconda3\lib\site-packages\locust\user\task.py", line 493, in execute_task task(self.user) File "C:\Users\harih\OneDrive\Desktop\IBM App\locust.py", line 23, in home data=data[random.randint(0,3)], headers=post_headers)	local

Charts





Final ratio

Ratio per User class

- 100.0% WebsiteUser
  - 14.3% index
  - 14.3% home
  - 14.3% home1
  - 14.3% contact
  - 14.3% prediction
  - 14.3% images
  - 14.3% explore

Total ratio

- 100.0% WebsiteUser
  - 14.3% index
  - 14.3% home
  - 14.3% home1
  - 14.3% contact
  - 14.3% prediction
  - 14.3% images
  - 14.3% explore

8.2 User Acceptance Testing ;

Test case ID	Feature Type	Component	Test Scenario
Login page 1	Functional	Login Page	Verify user is able to see the Login popup when user is clicked
LoginPage 2	UI	Log inPage	Verify the UI elements in Login popup
LoginPage 3	Functional	Log in page	Verify user is able to log into application with Valid credentials
LoginPage 4	Functional	Login page	Verify user is able to log into application with InValid credentials
signup page1	Functional	sign up	Verify the user to access the register page

signup page2	Functional	sign up	Verify the user's detail is correct or incorrect
Signup page3	UI	Sign up	Verify the UI elements in Login popup
Home page1	Functional	Home Page	Verify user is able to see the Home page popup when user is clicked
Home page2	Functional	Home page	verify the menu bar and heading is working or not

Home Page3	Functional	Home Page	Verify the explore and helpline button is working or not
Home page4	UI	Home Page	Verify the UI elements in Home page popup
Explorepage1	Functional	Explore page	Verify the Explore page is viewed or not
Explore page2	UI	Explore page	Verify the UI elements in the Explore page

Prediction page1	Functional	Prediction page	Verify the prediction page is viewed and working or not
Prediction page2	UI	Prediction page	Verify the UI elements in the Prediction page
Image page1	Functional	Image page	Verify the Image page is viewed and working or not
Image page2	UI	Image page	Verify the UI elements in the image page

contact page 1	Functional	contact page	Verify the Image page is viewed and working or not
contact page 2	UI	contact page	Verify the UI elements in the contact page
Helpline	Functional	Helpline page	Verify the Helpline is working or not

Date	03-Nov-22	
Team ID	PNT2022TMID06357	
Project Name	Project - Detecting of Parkinson's disease using Machine Learning	
Maximum Marks	4 marks	

Pre-Requisite	Steps To Execute	Test Data
	1.Enter URL and click go 2.Verify login popup displayed or not	http://localhost//5000
	1.Enter URL and click go 2.Verify login popup with below UI elements: a.email text box b.password text box c.Login button d.New user? Create account link	http://localhost//5000
	1.Enter URL(http://localhost//5000) and click go 2.Enter Valid username/email in Email text box 3.Enter valid password in password text box 4.Click on login button	Username: chalam@gmail.com password: Testing123
	1.Enter URL(http://localhost//5000) and click go 2.Click on My Account dropdown button 3.Enter InValid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: chalam@gmail password: Testing123
	1.Enter URL(http://localhost//5000) and click go 2.Click on My Account dropdown button. 3.Click on Register button	

	1.Enter URL(http://localhost//5000) and click go 2.Click on log in page 3.Enter InValid username/email in Email text box 4.Enter Invalid password in password text box 5.Click on Register button	username:chalam@gmail password :Testing123 Re-enter password First name:XYZ Last name:ABC
	1.Enter url (http://localhost//5000) 2.click on log in page 3. Verify popup menu with below UI elements: a) First name b)Last name c)email d)passsword e)Re-enter password f)register button	http://localhost//5000
	1)click on the register page 2)verify popup menu is displayed or not 3)Check the display menu bar a)Home b)Prediction c)Images d)contact e)Helpline button f)Explore	
	1)click on the register page 2)verify popup menu is displayed or not 3)Heading is alligned in left side 4)Check the display menu bar a)Home b)Prediction c)Images d)contac	

	1)click on the register page 2)verify popup menu bar is displayed or not 3)click the explore and helpline button	
	1)Enter the url(http://localhost//5000) 2)Verify menu bar with UI elements: a)Home b)Images c)Prediction d) contact 3)Explore button and helpline button 4)Headings and background image	
	1)Enter the url(http://localhost//5000) 2)Click on the home page 3)click on the explore button viewed in the center 4)Learn more button and background image is viewed 5)Check the menu bar is displayed or not: a)Home b)Images c)Prediction d) contact e)Helpline	
	1)Enter the url(http://localhost//5000) 2)Verify menu bar UI elements: a)Home b)Images c)Prediction d) contact 3)Helpline button and Learn more button 4)Headings and background image	

	1)Enter the URL(http://localhost//5000) 2) click on the home page 3) Verify your mail to access Prediction page 4)If your mail is verified ,user is viewed the Prediction page 5) Displayed of choose file location button and Prediction button	Username: chalam@gmail password: Testing123
	1)Enter the URL(http://localhost//5000) 2) click on the home page 3) Verify your mail to access Prediction page 4)If your mail is verified ,user is viewed the Prediction page 5) choose file	
	1)Enter the url(http://localhost//5000) 2)Click on the home page 3)click on the image button viewed in the menu bar 4) Background colour is viewed 5)Check the menu bar is displayed or not: a)Home b)Images c)Prediction d) contact e)Helpline button 6)In that image page some of the images are displayed	
	1)Enter the url(http://localhost//5000) 2)Verify menu bar with UI elements: a)Home b)Images c)Prediction d) contact 3)helpline button 4)background colour and heading 5)Six images are displayed	

	1)click on the home page 2)verify popup menu is displayed or not 3)Check the display menu bar a)Home b)Prediction c)Images d)contact e)Helpline button f)Explore button 4)Our contact is displayed 5)If any queries is occured ,please in the given box and fill the details 6)click the "Send anyway"button	Username: chalam@gmail
	1)Enter the url(http://localhost//5000) 2)Verify menu bar with UI elements: a)Home b)Images c)Prediction d) contact 3)Explore button and helpline button 4)Headings and background colour	
	1)Enter the url(http://localhost//5000) 2)Verify menu bar with all pages helpline button lile: a)Home page b)Image page c) contact page d)Explore page e)click the helpline button it send to URL in the right side top 3) viewed page of Helpline page is a Paragraph(Description)	http://localhost//5000

Expected Result	Actual Result	Status	Commnets
Login popup should display	Working as expected	Pass	
Application should show below UI elements: a.email text box b.password text box c.Login button with green colour d.New user? Create account link and background image	Working as expected	Pass	
User should navigate to user account homepage	working as expected	Pass	
Application should show 'Incorrect email or password ' validation message.	Working as expected	Pass	
Application should show the register page	Working as expected	Pass	

Application should show 'Incorrect email or password or First name or Last name or re-enter password ' validation message.	Working as expected	Pass	
Applications should show below UI elements: a)First name text box b)Second name text box c)email text box d) password text box e)Re-enter password text box f)Register button green color and backgroung image	Working as expected	Pass	
Application should display the page or not a)Home b)Prediction c)Images d)contact e)Helpline button f)Explore	Working as expected	Pass	
Application should display the page or not a)Home b)Prediction c)Images d)contact Heading is alligned in left side or not or invalid	Working as expected	Pass	

Application sshould be viewed in explore and helpline button	Working as expected	Pass	
1)Application should show UI elements below: a)Home b)Images c)Prediction d) contact 2)heading and background image 3)Explore button and helpline button in blue color or inavalid	Working as expected	Pass	
1)Application should show the explore page ,learn more button and backgroung image 2)Application should show elements below: a)Home b)Images c)Prediction d) contact e)Helpline 3)Paragraph	Working as expected	Pass	
1)Application should show UI elements below (menu bar): a)Home b)Images c)Prediction d) contact 2)heading and background image 3) Learn more button andhelpline button in blue color or inavalid	Working as expected	Pass	

1)Application should show the Prediction page or "Verify your mail" is viewed to the user or inavalid 2)) Displayed of choose file location button and Prediction button 3) predict the image using machine learning	Working as expected	Pass	
Application should show the Prediction page with UI elements: 1)Background colour black 2)Choose file location button blue colour 3)Prediction button blue colour	Working as expected	Pass	
1)Application should show elements below (menu bar): a)Home b)Images c)Prediction d) contact 2)Background image 3) Helpline button or inavalid 2)Some of the images are displayed in this page or invalid	Working as expected	Pass	
1)Application should show the image page , backgroung colour 2)Application should show elements below: a)Home b)Images c)Prediction d) contact e)Helpline 3)Six images rae displayed in the box type inside the box is white colour 4)Helpline button blue colour and Heading	Working as expected	Pass	

1)Application should show UI elements below: a)Home b)Images c)Prediction d) contact 2)heading and background colour 3)Our contact is displayed or not 4)queries table is viewed or not 5)email 6)queries 7)send anyway button 3)Explore button and helpline button	Working as expected	Pass	
1)Application should show UI elements below (menu bar): a)Home b)Images c)Prediction d) contact 2)Background colour and heading(bold) 3) Helpline button 4)email text box 5)queries text box 6)send anyway button blue clour 7)Explore button and helpline button in blue color or inavalid	Working as expected	Pass	
1)Application should show UI elements helpline button in all pages given below: a)Home b)Images c) contact d)Image 2)Helpline button in blue color or inavalid in the right side top 3)Paragraph and lines	Working as expected	Pass	

TC for Automation(Y/N)	BUG ID	Executed By
N	nil	Hariharan M Hari Priya D
N	nil	Hariprasath B Raja M
N	nil	Hari Priya D Raja M
N	nil	Hariprasath B Hari Priya D
N	nil	Raja M Hari Priya D

N	nil	Hariharan M Hariprasath B
N	nil	Raja M Hariharan M
N	nil	Hariprasath B Hari Priya D
N	nil	Hariharan M Hariprasath B

N	nil	Hari Priya D Raja M
N	nil	Hariharan M Hari Priya D
N	nil	Hari Priya D Raja M
N	nil	Hariprasath B Raja M

N	nil	Hariharan M Raja M
N	nil	Hari Priya D Hariprasath B
N	nil	Raja M Hariharan M
N	nil	Hariprasath B Hariharan M

N	nil	Hari Priya D Hariharan M
N	nil	Hariprasath B Raja M
N	nil	Hariharan M Hariprasath B

9.RESULTS

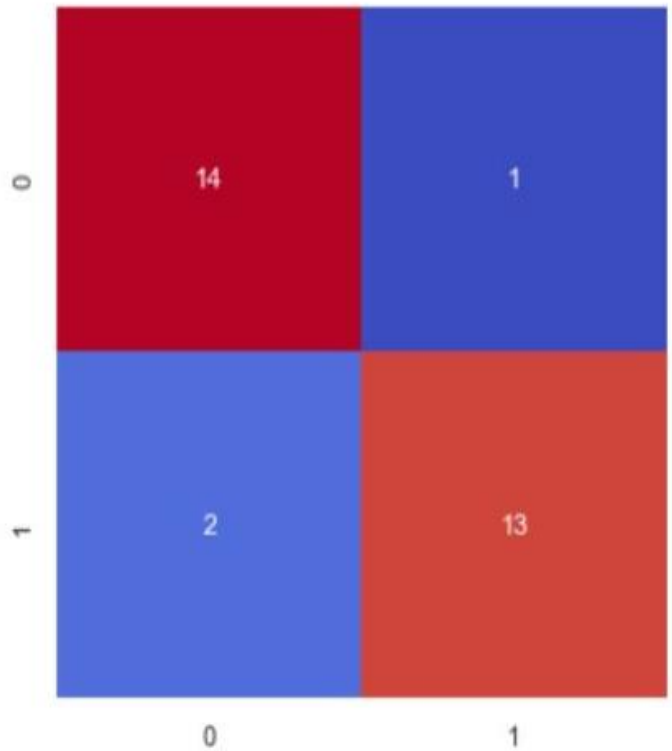
9.1 Performance Metrics

Model Evaluation

In [36]:
```
cnf = confusion_matrix(testY,preds)
cnf
```

Out[36]:
```
array([[14,  1],
       [ 2, 13]], dtype=int64)
```

In [37]:
```
plt.figure(figsize=(5,5))
sns.heatmap(cnf , annot=True , cmap="coolwarm" , cbar=False)
plt.show()
```



Model accuracy and Prediction ¶

In [45]:
```
predictions = model.predict(testX)

cm = confusion_matrix(testY, predictions).flatten()
print(cm)
(tn, fp, fn, tp) = cm
accuracy = (tp + tn) / float(cm.sum())
print(accuracy)
```

```
[14  1  2 13]
0.9
```

10.ADVANTAGES &DISADVANTAGES :

- Machine learning algorithm (MLA) can be used for early detection of disease to increase the chances of elderly people's lifespan and improved lifestyle with Parkinson.
- Due to diseases diagnosis importance to mankind, several studies have been conducted on developing methods for the classification of Parkinson disease
- An efficient diagnosis system for Parkinson's disease using kernel-based extreme learning machine with subtractive clustering features weighting approach.
- Machine learning algorithms are able to support intelligent decisions by using different data types (demographic, laboratory, and image data) to make predictions of disease risk, diagnosis, prognosis, and appropriate treatments
- Early prediction of disease could be performed with the provided input

- In our research work, Parkinson’s disease is detected using image as input and we used machine learning technology for classification with higher accuracy
- The benefits of early prediction and management of PD would affect not only the individual (and their families) but also the wider society and research community.
- early detection of PD, ML models have been applied to multiple data modalities, including movement, neuroimaging, and voice and handwriting patterns

DISADVANTAGES :

- Predicted Parkinson’s are 31 on a heat map. Parkinson’s disease affects the CNS of the brain and has yet no treatment unless it’s detected early. Late detection leads to no treatment and loss of life.
- If input not given in the specified mentioned format results will not be displayed
- Prediction is made as yes or no ,the type will not be mentioned
- Parkinson’s disease prediction using machine learning requires massive data sets to train on and these should be inclusive/unbiased, and of good quality
- Parkinson’s disease prediction using machine learning needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you
- Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

.

11.CONCLUSION :

Parkinson's disease is the second most dangerous neurodegenerative disease which has no cure till now and to make it reduce prediction is important. In this project, we have used three various prediction models to predict the Parkinson's disease which are Machine Learning Techniques i.e. KNN, Naïve Bayes and Logistic Regression. The dataset is trained using these models and we also compared these different models built using different methods and identifies the best model that fits. The aim is to use various evaluation metrics such as Accuracy, Precision, Recall, Specificity, F1-score, LR+, LR- and Youden score that produce the predicts the disease efficiently. We have used the Speech dataset that contains voice features of the patients which is available in the Kaggle website. The dataset consists of more than 700 features and 750 patient details. The models are built using the five best features which were identified by feature selection. From this results, Naïve Bayes outstands from the other two machine learning algorithms with an accuracy of 81%. This system we designed can make the predictions of the Parkinson's disease.

12.FUTURE SCOPE :

In future, these models can be trained with different datasets that have best features and can be predicted more accurately. If the accuracy rate increases, it can be used by the laboratories and hospitals so that it is easy to predict in early stages. This models can be also used with different medical and disease datasets. In future the work can be extended by building a hybrid model that can find more than one disease with an accurate dataset and that dataset has common features of two diseases. In future the work can extended to build a model that may extract more important features among all features in the dataset so that it produce more accuracy.

13. APPENDIX

Source Code :

Importing required Libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import zipfile as zf
import os
import random
import cv2
import pickle
from imutils import build_montages
from imutils import paths
from sklearn.metrics import classification_report,confusion_matrix
from sklearn import metrics
from sklearn.preprocessing import LabelEncoder,LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import
RandomForestClassifier,GradientBoostingClassifier,ExtraTreesClassifier
from skimage import feature
#from google.colab.patches import cv2_imshow
sns.set()
os.getcwd()
```

Loading the dataset

```
handle_spiral = zf.ZipFile(r'dataset.zip')
handle_spiral.extractall('dataset')
handle_spiral.close()
spiral_train_healthy = os.listdir('dataset/dataset/spiral/training/healthy/')
spiral_train_park = os.listdir('dataset/dataset/spiral/training/parkinson/')

fp_spiral_train_healthy = 'dataset/dataset/spiral/training/healthy/'
fp_spiral_train_park = 'dataset/dataset/spiral/training/parkinson/'

spiral_test_healthy = os.listdir('dataset/dataset/spiral/testing/healthy/')
spiral_test_park = os.listdir('dataset/dataset/spiral/testing/parkinson/')

fp_spiral_test_healthy = 'dataset/dataset/spiral/testing/healthy/'
```

```
fp_spiral_test_park = 'dataset/dataset/spiral/testing/parkinson/'
```

Qunatifying the images

```
def quantify_image(image):
  features = feature.hog(image,orientations=9,

pixels_per_cell=(10,10),cells_per_block=(2,2),transform_sqrt=True,block_norm="L1")

  return features
```

Splitting of Training and Testing Data

```
trainX = []
testX = []
outputs = []
trainY = []
testY = []

for i in spiral_train_healthy:
  image = cv2.imread(fp_spiral_train_healthy+i)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image =cv2.threshold(image, 0, 255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  trainX.append(features)
  trainY.append('healthy')

for i in spiral_train_park:
  image = cv2.imread(fp_spiral_train_park+i)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  trainX.append(features)
  trainY.append('parkinson')
```

```
for i in spiral_test_healthy:
  image = cv2.imread(fp_spiral_test_healthy+i)
  outputs.append(image)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV |
cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  testX.append(features)
  testY.append('healthy')

for i in spiral_test_park:
  image = cv2.imread(fp_spiral_test_park+i)
  outputs.append(image)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV |
cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  testX.append(features)
  testY.append('parkinson')
trainX = np.array(trainX)
testX = np.array(testX)
trainY = np.array(trainY)
testY = np.array(testY)
trainX
trainY
testX
testY
```

Label Encoding

```
le = LabelEncoder()
trainY = le.fit_transform(trainY)
testY = le.transform(testY)
print(trainX.shape,trainY.shape)
trainY
testY
```

Model Building

Train the model

```
print("Training model....")
model = RandomForestClassifier(n_estimators=100)
model.fit(trainX,trainY)
preds = model.predict(testX)
preds
```

Model Evaluation

```
cnf = confusion_matrix(testY,preds)
cnf
plt.figure(figsize=(5,5))
sns.heatmap(cnf , annot=True , cmap="coolwarm" , cbar=False)
plt.show()
acc = metrics.accuracy_score(testY,preds)
acc
indexes = np.random.randint(0,30,25)
indexes
```

Test the model

```
testpath=list(paths.list_images(fp_spiral_train_healthy))
idxs=np.arange(0,len(testpath))
idxs=np.random.choice(idxs,size=(25,),replace=False)
images=[]

for i in idxs:
   image=cv2.imread(testpath[i])
   output=image.copy()
   output=cv2.resize(output,(128,128))
   image=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
   image=cv2.resize(image,(200,200))
   image=cv2.threshold(image,0,255,cv2.THRESH_BINARY_INV |
cv2.THRESH_OTSU)[1]

   features= quantify_image(image)
   preds=model.predict([features])
   label=le.inverse_transform(preds)[0]
   if label=="healthy":
     color=(0,255,0)
   else:
     (0,0,255)
   cv2.putText(output,label,
(3,20),cv2.FONT_HERSHEY_SIMPLEX,0.5,color,2)
```

```
    images.append(output)
montage=build_montages(images,(128,128),(5,5))[0]
cv2.imshow('',montage)
cv2.waitKey(0)
```

Model accuracy and Prediction

```
predictions = model.predict(testX)
cm = confusion_matrix(testY, predictions).flatten()
print(cm)
(tn, fp, fn, tp) = cm
accuracy = (tp + tn) / float(cm.sum())
print(accuracy)
```

Save the model

```
pickle.dump(model,open('parkinson.pkl','wb'))
```

Application Source code using flask:

```
from flask import Flask, flash, redirect, render_template, request, session,
url_for

from flask_mail import Mail, Message
from itsdangerous import URLSafeTimedSerializer, SignatureExpired
import pickle
import cv2
from skimage import feature
import os.path

import sqlite3


app=Flask(__name__)


app.secret_key="#@universityflaskapp@#"

# email verification

app.config.from_pyfile('config.cfg')

mail=Mail(app)

s = URLSafeTimedSerializer(app.config['SECRET_KEY'])
```

```
# database creation
con=sqlite3.connect("database.db")
print("Opened database successfully")
con.execute("create table if not exists customer(pid integer primary key, name
text, email text, password text,status BOOLEAN)")
print("Table created successfully")
con.close()



@app.route('/signup',methods=['GET','POST'])
def signup():
  if request.method == 'POST':
        try:
            name=request.form['name']
            email=request.form['email']
            password=request.form['password']
            con=sqlite3.connect("database.db")
            cur=con.cursor()
            cur.execute("INSERT INTO customer(name,email,password) VALUES
(?,?,?)",(name,email,password))
            con.commit()
            flash("Registered successfully","success")
        except:
            con.rollback()
            flash("Problem in Registration, Please try again","danger")
        finally:
            return redirect(url_for("index"))
            con.close()
  else:
        return render_template('signup.html')

@app.route('/login',methods=['POST','GET'])
def login():
  if request.method =='POST':
        email = request.form['email']
        password = request.form['password']
        con=sqlite3.connect("database.db")
        con.row_factory=sqlite3.Row
        cur=con.cursor()
        cur.execute("SELECT * FROM customer where email=? and
password=?",(email,password))
        data=cur.fetchone()

        if data:
            session["email"]=data["email"]
            print("sent to home")
```

```
            return redirect(url_for("home"))

        else:
            flash("Username or Password is incorrect","danger")
            print("not sent to home")
  return redirect(url_for("home"))

@app.route('/')
def index():
  return render_template("index.html")

@app.route('/home')
def home():
  return render_template("homepage.html")

@app.route('/prediction')
def prediction():
  return render_template("prediction.html")

@app.route('/images')
def images():
  return render_template("image.html")

@app.route('/contact')
def contact():
  return render_template("contact.html")

@app.route('/explore')
def explore():
  return render_template("explore.html")

@app.route('/logout')
def logout():
    session.clear()
    return render_template('index.html')


@app.route('/healthy')
def healthy():
    return render_template('healthy.html')

@app.route('/parkinson')
def parkinson():
    return render_template('parkinson.html')

@app.route('/check')
def check():
    email = session["email"]
    con=sqlite3.connect("database.db")
```

```
    cur=con.cursor()
    cur.execute("SELECT status FROM customer where email=?",[email])
    data=cur.fetchone()
    con.commit()
    print(data)
    if data[0]==1:
        return render_template('prediction.html')
    else:
        return render_template('verify.html')

@app.route('/verify')
def verify():
    email = session["email"]

    token = s.dumps(email, salt='email-confirm')

    msg=Message('Confirm Email', sender='ibmproject2023@gmail.com.',
recipients=[email])

    link=url_for('confirm_email', token=token, _external=True)

    msg.body= 'Please click the link to verify your account to continue  : {}
'.format(link)

    mail.send(msg)
    return render_template("homepage.html")

@app.route('/confirm_email/<token>')
def confirm_email(token):
  try:
    email=s.loads(token, salt='email-confirm' , max_age=3600*5)
  except SignatureExpired:
    return render_template("verify.html")
  con=sqlite3.connect("database.db")
  con.row_factory=sqlite3.Row
  cur=con.cursor()
  cur.execute("UPDATE customer SET status = 1 WHERE email = ?",(email,))
  con.commit()
  con.close()
  return redirect(url_for("prediction"))



@app.route('/upload', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        f = request.files['file2']  # requesting the file
        #filename_secure = secure_filename(f.filename)
        basepath = os.path.dirname(
```

```
            '__file__')  # storing the file directory
        # storing the file in uploads folder
        filepath = os.path.join(basepath, "uploads", f.filename)
        f.save(filepath)  # saving the file

        # Loading the saved model
        print("[INFO] loading model...")
        model = pickle.loads(open('parkinson.pkl', "rb").read())
        ''' local_filename = "./uploads/"
        local_filename += filename_secure
        print(local_filename)'''

        # Pre-process the image in the same manner we did earlier
        image = cv2.imread(filepath)
        output = image.copy()

        # Load the input image, convert it to grayscale, and resize
        output = cv2.resize(output, (128, 128))
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        image = cv2.resize(image, (200, 200))
        image = cv2.threshold(image, 0, 255,
                              cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]

        # Quantify the image and make predictions based on the extracted
features using the last trained Random Forest
        features = feature.hog(image, orientations=9,
                               pixels_per_cell=(10, 10), cells_per_block=(2,
2),
                               transform_sqrt=True, block_norm="L1")
        preds = model.predict([features])
        print(preds)
        ls = ["healthy", "parkinson"]
        result = ls[preds[0]]
        '''color = (0, 255, 0) if result == "healthy" else (0, 0, 255)
        cv2.putText(output, result, (3, 20),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
        cv2.imshow("Output", output)
        cv2.waitKey(0)'''
        if result =="healthy":
          return redirect("healthy")
        else:
          return redirect("parkinson")

    return None
if __name__=='__main__':
  app.run(Debug=True)
```

GitHub & Project Demo Link :

https://github.com/IBM-EPBL/IBM-Project-54753-1662453739