PARKINSON'S DISEASE PREDICTION USING MACHINE LEARNING TECHNIQUES

IBM PROJECT REPORT

Submitted by

MANOJ KUMAR S	**(921019104027)**
SHAKUL KANI N	**(921019104046)**
MOHAMED NAINAR A	**(921019104028)**
SIVABALAN P	**(921019104050)**

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

NADAR SARASWATHI COLLEGE OF ENGINEERING AND TECHNOLOGY, THENI-625531

ANNA UNIVERSITY:

CHENNAI

Project Report Format

1. **INTRODUCTION**

2. **LITERATURE SURVEY**
   - **Existing problem**
   - **References**
   - **Problem Statement Definition**
3. **IDEATION & PROPOSED SOLUTION**
   - **Empathy Map Canvas**
   - **Ideation & Brainstorming**
   - **Proposed Solution**
   - **Problem Solution fit**
4. **REQUIREMENT ANALYSIS**
   - **Functional requirement**
   - **Non-Functional requirements**
5. **PROJECT DESIGN**
   - **Data Flow Diagrams**
   - **Solution & Technical Architecture**
   - User Stories
6. **PROJECT PLANNING & SCHEDULING**
   - **Sprint Planning & Estimation**
   - **Sprint Delivery Schedule**
   - **Reports from JIRA**
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
8. **TESTING**
   - **Test Cases**
   - **User Acceptance Testing**
9. **RESULTS**
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**
    - **Source Code**
    - **GitHub & Project Demo Link**

1. **INTRODUCTION**

Parkinson's disease is a global public health concern. Nerve cells, the building blocks of the nervous system in the brain stop producing when they are damaged. Thus, less dopamine is produced that inhibits motor skills and speech. Voice changes in the first stage before brain cells are affected, hence helps identify Parkinson's disease in the early stages and therefore prevent brain cell damage that can lead to reduced fusion and movement. Introduction of different ML algorithms for classification of Parkinson's disease is presented.

Keywords: Parkinson's Disease, Machine Learning, Computer Science, KNN, SVM,

Regression, RF, Decision Tree

1.1 Project Overview

By the end of this project:

You'll be able to understand the problem to classify if it is a regression or a classification kind of problem.
You will be able to know how to pre-process the image by using different data pre-processing techniques.
you will be able to learn how to use OpenCV and machine learning to automatically detect Parkinson's disease in hand-drawn images of spirals and waves
You will be able to know how to find the accuracy of the model.
You will be able to build web applications using the Flask framework.

PROJECT FLOW

- User interacts with the UI (User Interface) to upload the image as input
- The uploaded image is analyzed by the model which is integrated
- Once the model analyzes the uploaded image, the prediction is showcased on the UI and OpenCV window

To accomplish this, we have to complete all the activities and tasks listed below

- **Data Collection**
  - Collect the dataset or Create the dataset
- **Image Preprocessing**.
  - Importing the required libraries
  - Loading Train data and Test data
  - Quantifying images
  - Label Encoding

pre-requisites

In order to develop this project we need to install the following software/packages:

Anaconda Navigator :

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS. Conda is an open-source, cross-platform, package management system. Anaconda comes with so very nice tools like JupyterLab, Jupyter Notebook, QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code. For this project, we will be using Jupyter notebook and Spyder

To install Anaconda navigator and to know how to use Jupyter Notebook & Spyder using Anaconda

To build Machine learning models you must require the following packages

- **Numpy:**
  - It is an open-source numerical Python library. It contains a multidimensional array and matrix data structures and can be used to perform mathematical operations
- **Scikit-learn:**
  - It is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-nearest neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy
- **Scikit-image**
  - Scikit-image or skimage, is an open-source Python package designed for image preprocessing.
- **Install imutils**
  - Imutils are a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, and displaying Matplotlib images easier with OpenCV
  - Open anaconda prompt and type this command
    “pip install imutils”

- **OpenCV**
  - OpenCV is a library of programming functions mainly aimed at real-time computer vision. Here, OpenCV is used to capture frames by accessing the webcam in real-time.
  - Open anaconda prompt and type this command
    "pip install opencv-contrib-python"

- **Flask:**

Web framework used for building Web applicationsIf you are using anaconda navigator, follow below steps to download required packages:

- Open anaconda prompt.
- Type **"pip install numpy"** and click enter.
- Type **"pip install scikit-image"** and click enter.
- Type **"pip install imutils"** and click enter.
- Type **"pip install scikit-learn"** and click enter.
- Type **"pip install opencv-contrib-python"** and click enter.
- Type **"pip install Flask"** and click enter.

2. LITERATURE SURVEY

Parkinson's disease (PD) has affected millions of people worldwide and is more prevalent inpeople, over the age of 50. Even today, with many technologies and advancements, earlydetection of this disease remains a challenge. This necessitates a need for the machine learning-based automatic approaches that help clinicians to detect this disease accuratelyinits early stage. Thus, the focus of this research paper is to provide an insightful surveyandcompare the existing computational intelligence techniques used for PD detection. Tosavetime and increase treatment efficiency, classification has found its place in PDdetection. Theexisting knowledge review indicates that many classification algorithms have been usedtoachieve better results, but the problem is to identify the most efficient classifier for PDdetection. The challenge in identifying the most appropriate classification algorithmlies intheir application on local dataset. Thus, in this paper three types of classifiers, namely, Multilayer Perceptron, Support Vector Machine and K-nearest neighbor have been discussedon the benchmark (voice) dataset to compare and to know which of these classifiers is themost efficient and accurate for PD classification. The Voice input dataset for these classifiershas been obtained from UCI machine learning repository. ANN with Levenberg–Marquardt algorithm was found to be the best classifier, having highest classification accuracy (95.89%). Moreover, we compared our results with those obtained by

Resul Das ["A comparisonof multiple classification methods for diagnosis of Parkinson Disease,"

2.1 Existing problem

In existing system, PD is detected at the secondary stage only (Dopamine deficiency) which leads to medical challenges. Also doctor has to manually examine and suggest medical diagnosis in which the symptoms might vary from person to person so suggesting medicine is also a challenge. Thus the mental disorders are been poorly characterized and have many health complications. PD is generally diagnosed with the following clinical methods as, MRI or CT scan - Conventional MRI cannot detect early signs of Parkinson's disease PET scan - is used to assess activity and function of brain regions involved in movement SPECT scan - can reveal changes in brain chemistry, such as a decrease in dopamine This results in a high misdiagnosis rate (up to 25% by non-specialists) and many years before diagnosis, people can have the disease. Thus existing system is not effective in early prediction and accurate medicinal diagnosis to the affected people

2.2 References

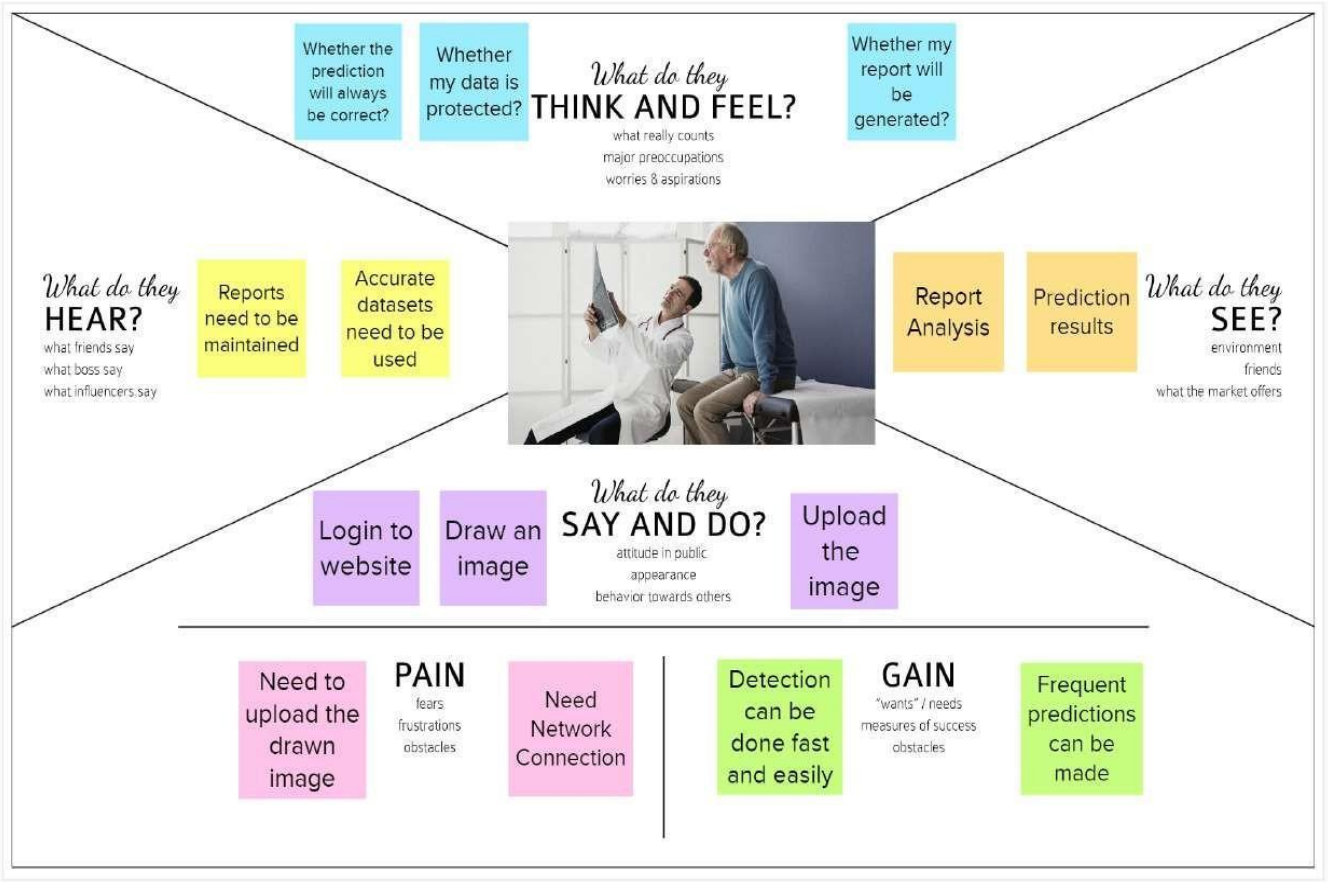
1. Dickson, D.W. Neuropathology of Parkinson disease.
Parkinsonism Relat. Disord. 2018, 46 (Suppl. 1), S30–S33.
2.. Kalia, L.V.; Lang, A.E. Parkinson's Disease. Lancet 2015,
386, 896–912.

2.3 Problem Statement Definition

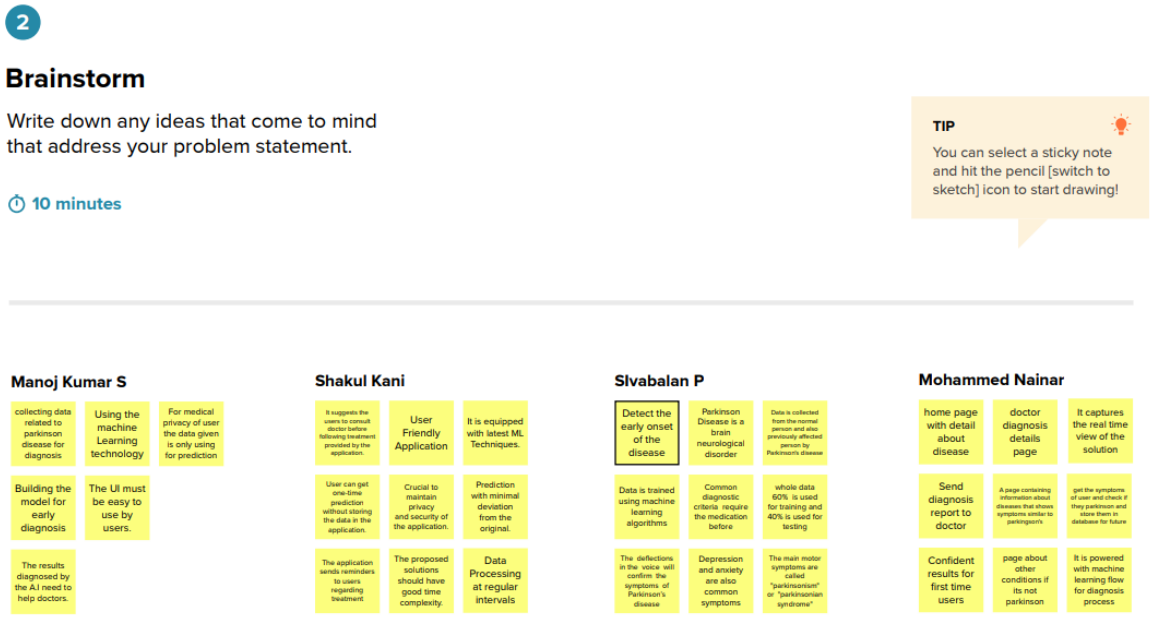
The main aim is to predict the prediction efficiency that would be beneficial for the patients who are suffering from Parkinson and the percentage of the disease will be reduced. Generally in the first stage, Parkinson's can be cured by the proper treatment. 10 So it's important to identify the PD at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



3.2 Ideation & Brainstorming



3.3 Proposed Solution

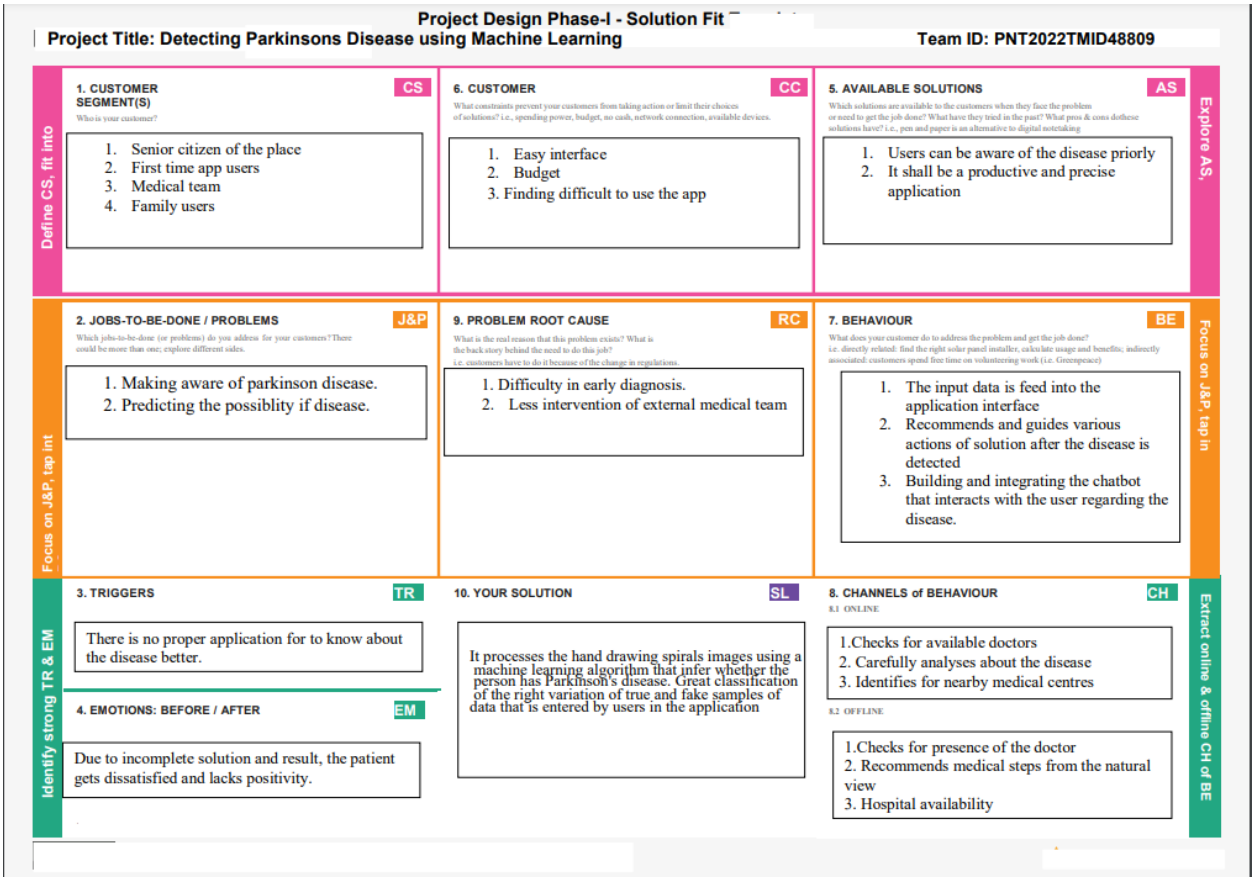
S.No	Parameter	Description
1.	Problem Statement	Diagnosis of Parkinson's disease is commonly based on medical observations and assessment of clinical signs, including the characterization of a variety of motor symptoms. However, traditional diagnostic approaches may suffer from subjectivity as they rely on the evaluation of movements that are sometimes subtle to human eyes and therefore difficult to classify, leading to possible misclassification. In the meantime, early non- motor symptoms of PD may be mild and can be caused by many other conditions. Therefore, these symptoms are often overlooked, making diagnosis of PD at an early stage challenging. To address these difficulties and to refine the diagnosis and assessment procedures of PD, machine learning methods have been implemented for the classification of PD and healthy controls or patients with similar clinical presentations (e.g., movement disorders or other Parkinsonian syndromes). To provide a comprehensive overview of data modalities and machine learning methods that have been used in the diagnosis and differential diagnosis of PD, in this study, we conducted a literature review of studies published until February 14, 2020, using the PubMed and IEEE Xplore databases. A total of 209 studies were included, extracted for relevant information and presented in this review, with an investigation of their aims, sources of data, types of data, machine learning methods and associated outcomes. These studies demonstrate a high potential for adaptation of machine learning methods and novel biomarkers in clinical decision making, leading to increasingly systematic, informed diagnosis of PD.

2.	Idea / Solution description	It processes the spiral image drawn by the patient using a neural network that infer whether the person has Parkinson's disease, and if they are identified then it assesses the severity of their disease in accordance with the Movement Disorder Society Unified Parkinson's Disease using ML algorithms. User can place their values and interact with the friendly user assistance bot which guides the person in using the application. Great classification of the right variation of true and fake samples of data that is entered by users in the application.
3.	Novelty / Uniqueness	Parkinson's Disease is detected at the secondary stage only (Dopamine deficiency) which leads to medical challenges. Also, doctor must manually examine and suggest medical diagnosis in which the symptoms might vary from person to person so suggesting medicine is also a challenge. So hence the disease examination varies at different instances of the medical operations. Here by using machine learning methods, the problem can be addressed with very less error rate. The voice dataset of Parkinson's disease from the UCI Machine learning library is used as input. Also, our proposed system provides accurate results by integrating spiral drawing inputs of normal and Parkinson's affected patients. We propose a hybrid and accurate results analyzing patient both voice and spiral drawing data. This application offers medical advice and solutions as the next step after user is confirmed based on the presence of Parkinson's disease. This can be used direct by medical team for analyzing and offering the solutions at much positive Scaling time.

4.	Social Impact / Customer Satisfaction	Increases interaction with the human and application. Personalize the UI experience. Improves accurate result as expected. An automated chatbot controls the user interaction environment. Accurate prediction at good time complexity.

5.	Business Model	Solutions prospects of improvement. Suits for better saving of involvements. Economic Development. Easy interface.
6.	Scalability of the Solution	Good conversation with ethnicity people. Saves enough time for performing internal operations. It does not require for the users to spend some money in offering their basic data into the model. On the spot result for the users.

Problem Solution fit



Project Design Phase-I - Solution Fit

Project Title: Detecting Parkinsons Disease using Machine Learning

Team ID: PNT2022TMID48809

Define CS, fit into

1. CUSTOMER SEGMENT(S) — CS

Who is your customer?

1. Senior citizen of the place
2. First time app users
3. Medical team
4. Family users

6. CUSTOMER — CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e., spending power, budget, no cash, network connection, available devices.

1. Easy interface
2. Budget
3. Finding difficult to use the app

5. AVAILABLE SOLUTIONS — AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons dothese solutions have? i.e., pen and paper is an alternative to digital notetaking

1. Users can be aware of the disease priorly
2. It shall be a productive and precise application

Explore AS,

Focus on J&P, tap int

2. JOBS-TO-BE-DONE / PROBLEMS — J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

1. Making aware of parkinson disease.
2. Predicting the possiblity if disease.

9. PROBLEM ROOT CAUSE — RC

What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations.

1. Difficulty in early diagnosis.
2. Less intervention of external medical team

7. BEHAVIOUR — BE

What does your customer do to address the problem and get the job done? i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

1. The input data is feed into the application interface
2. Recommends and guides various actions of solution after the disease is detected
3. Building and integrating the chatbot that interacts with the user regarding the disease.

Focus on J&P, tap in

Identify strong TR & EM

3. TRIGGERS — TR

There is no proper application for to know about the disease better.

4. EMOTIONS: BEFORE / AFTER — EM

Due to incomplete solution and result, the patient gets dissatisfied and lacks positivity.

10. YOUR SOLUTION — SL

It processes the hand drawing spirals images using a maching learning algorithm that infer whether the person has Parkinson's disease. Great classification of the right variation of true and fake samples of data that is entered by users in the application

8. CHANNELS of BEHAVIOUR — CH

8.1 ONLINE

1.Checks for available doctors
2. Carefully analyses about the disease
3. Identifies for nearby medical centres

8.2 OFFLINE

1.Checks for presence of the doctor
2. Recommends medical steps from the natural view
3. Hospital availability

Extract online & offline CH of BE

4. **REQUIREMENT ANALYSIS**

Following are the functional requirements of the proposed solution.

Functional Requirements:

A Functional Requirement may be a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software, its behavior, and outputs. It are often a calculation, data manipulation, business process, user interaction, or the other specific functionality which defines what function a system is probably going to perform. Functional Requirements describe the interactions between the system and its environment independent of its application. Applying the algorithms on the test data. Display the result with the description of having Parkinson's or not.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Home Page	▪ Description of Parkinson Disease, causes, symptoms and medications to be used.
FR-2	Test Vital Page-Uploading Image	▪ Uploading through a file input button ▪ Can input image file types like png,jpg,jpeg ▪ Input the required details that has been asked for
FR-3	Result/Prediction of disease	▪ If Positive – suggests to consult a doctor and to undergo treatment like deep brain stimulation, Leison Surgery, neural grafting or tissue transplants. ▪ If Negative – suggests preventive measures and symptoms. ▪ Important note is that the treatment may vary according to the individual.

Non-functional Requirements:

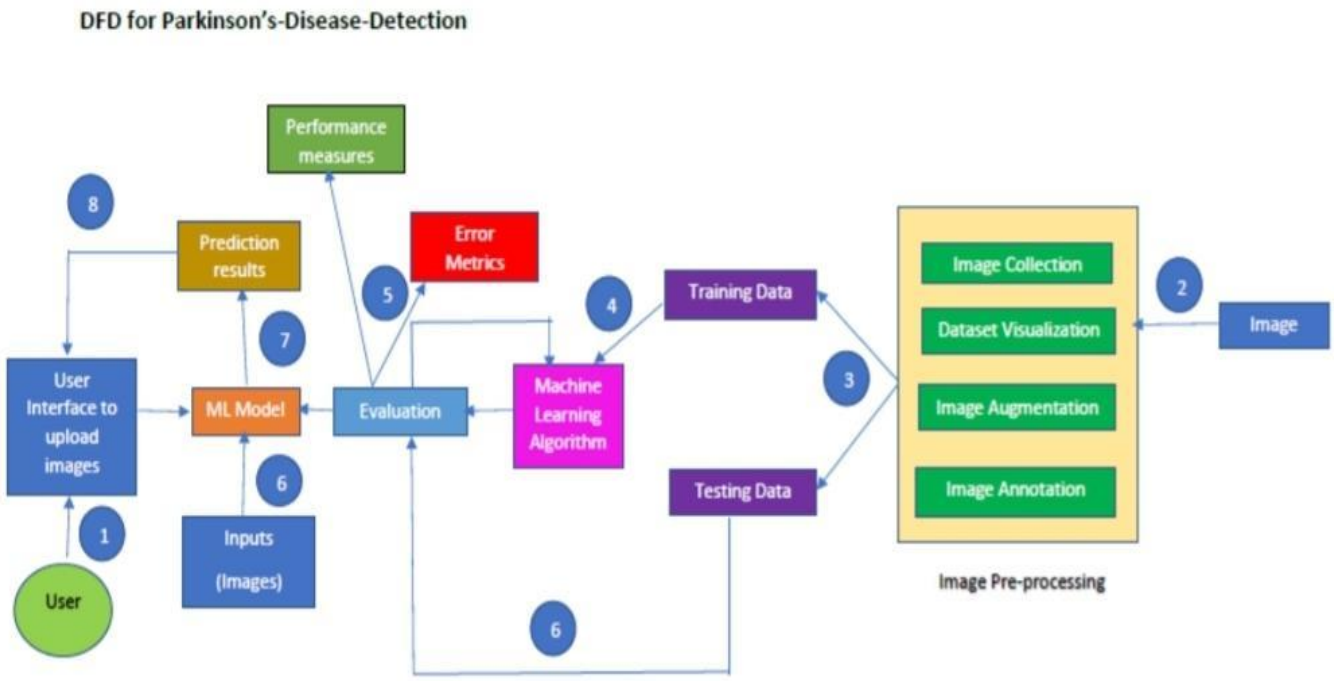
Following are the non-functional requirements of the proposed solution.

Non-Functional Requirements specifies the standard attribute of a software . They judge the software supported Responsiveness, Usability, Security, Portability, and other 34 non-functional standards that are critical to the success of the software. An example of a nonfunctional requirement, "how fast does the website load?" Failing to satisfy non-functional requirements may result in systems that fail to satisfy user needs. Non-functional Requirements allow you to impose constraints or restrictions on the planning of the system across the varied agile backlogs. Accuracy Reliability Flexibility

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	▪ Can be used by people of any age groups. ▪ Open source. ▪ Easily Accessible. ▪ lliterate people can also use efficiently.
NFR-2	**Security**	▪ Only the admin will be able to access the patient details so that the website will be more secured.
NFR-3	**Reliability**	▪ There is no possibility of hacking or misusing the data. ▪ No identity threat is possible ▪ Accuracy of prediction is very high when compared to the existing models since Classification, HOG, CNN makes it more reliable and responsive.
NFR-4	**Performance**	▪ Prediction results are obtained within seconds and the overall time is 3x times less when compared to the waiting time to get the results in medical centres or hospitals.
NFR-5	**Availability**	▪ The website can be accessible the entire day of 24 hrs. We had followed "Anytime, Anywhere accessible" policy.
NFR-6	**Scalability**	▪ Ability to provide proper medications, consultation, suggestions and results immediately. ▪ Able to access even from mobile phones that has internet connectivity

5. PROJECT DESIGN

5.1 Data Flow Diagrams



User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Upload Images	USN-1	As a user, I can upload the images in the website in order to obtain the prediction result of parkinson's disease	I can upload hand drawn spiral or wave images. The file type of image can be any	High	Sprint-1
	Test Vital Page	USN-2	As a user, I will get the prediction result and accuracy on the test vital page.	I can get instant result either positive or negative one click away	High	Sprint-1

	Dashboard	USN-3	Dashboard displays the symptoms, causes and medications for the Parkinson disease	I can register & access the dashboard with Facebook Login	Low	Sprint-2
Administrator	Data Collection	USN-4	As an Administrator, I need to collect data (images of spirals and waves drawn by healthy people and Parkinson's patients).	I have sizable amount of data that is splitted into training dataset and testing dataset.	Medium	Sprint-1
	Data Pre-Processing	USN-5	As an Administrator, I should clean my data and prepare it for model building by doing pre-processing activities such as resizing, visualizing the dataset and converting from RGB to grayscale	Cleaned dataset is ready for doing further process	High	Sprint-1

User Type	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Acceptance criteria**	**Priority**	**Release**
	Model Building	USN-6	As an Administrator, I need to build the model using Random Forest Classifier for spiral images and Convolutional Neural Networks (CNN) for wave images	ML Model is ready for deployment on the testing data.	High	Sprint-2
Customer (Web user)	Deployment of Model	USN-7	As an Administrator, I need to deploy the Machine Learning model that was built.	Model has been deployed successfully.	Medium	Sprint-3
Customer Care Executive	Building the frontend of the application	USN-8	As an Administrator, I need to build the website for the application using HTML, CSS etc.	The website is static and it is designed in order to achieve the user	High	Sprint-3

				interface		
Administrator	Connecting the ML model, Frontend and Backend	USN-9	As an Administrator, I can integrate the deployed model and web application using python flask server.	The web application is dynamic and can be used by the users now.	High	Sprint-4

5.2 Solution & Technical Architecture

Solution 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:

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

Example - Solution Architecture Diagram:

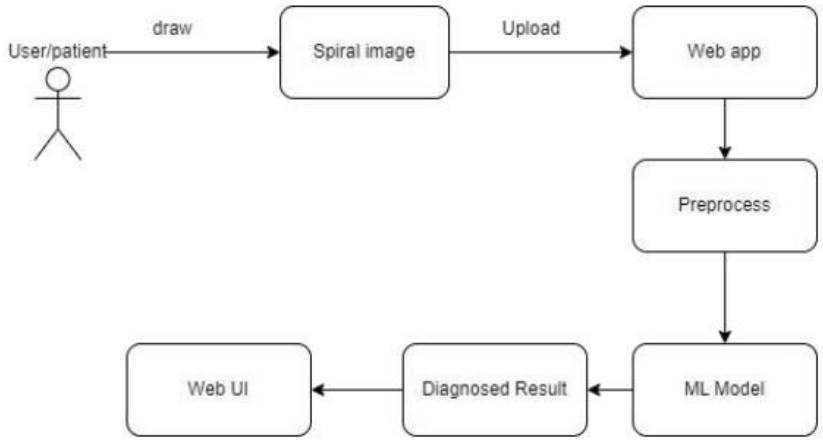


Figure 1: Architecture and data flow of the drawn image of the patient in application.

Technical Architecture:

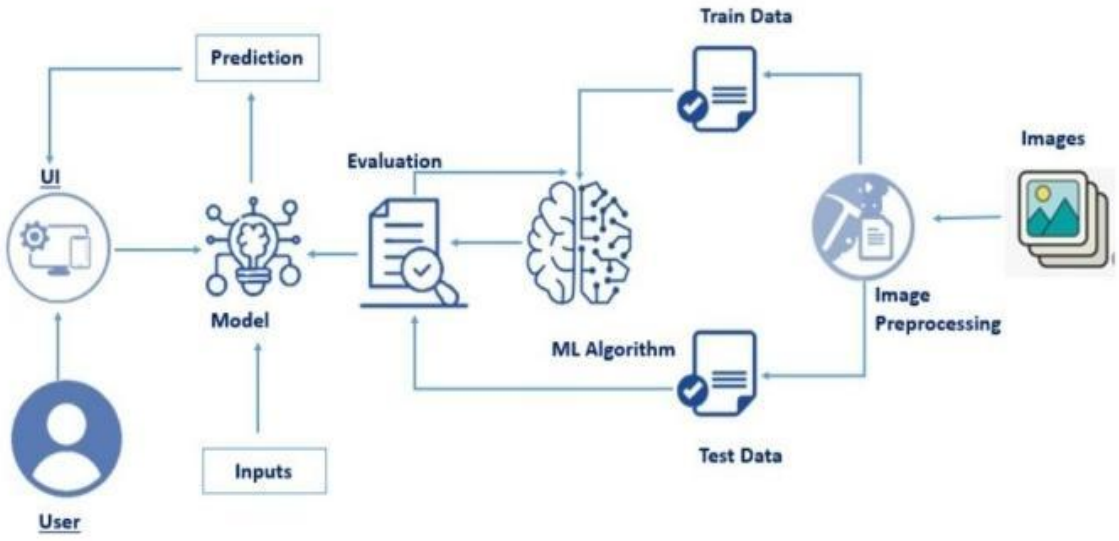


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The User Interact the service using web User Interface.	HTML, CSS, JavaScript.
2.	User Upload	Patient or a doctor hand drawn image to the web.	JavaScript.
3.	Parkinson Predication	The AI model classify the image.	Python.
4.	Data Analysis	The data set is analysis and preprocessed	Python, Computer Vision
5.	Machine Learning Model	Machine Learning Model diagnose the patient for Parkinson disease based on hand drawn spiral image.	CNN, Python, Computer Vision
6.	Model Architecture	It is a CNN Architecture trained by Backpropagation.	Python, Computer Vision
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: localhost:5000 Cloud Server Configuration : Deployed in IBM Cloud	IBM Cloud

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	For data analysis & model training open source frameworks are used.	Sklearn, numpy, pandas, tensorflow, keras.
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	Micro service	Flask
4.	Availability	Deployed in the IBM cloud so 24/7	IBM Cloud
5.	Performance	The asynchrous functionality is used so many concurrent users can access	Flask, Asynchrous

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

The below template shows the product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Upload Images	USN-1	As a user, I can upload the images in the website in order to obtain the prediction result of Parkinson's disease	2	High	1. Manoj Kumar S 2. Sivabalan P 3. Shakul Kani N
Sprint-4	Test Vital Page	USN-2	As a user, I will get the prediction result and accuracy on the test vital page.	3	High	1. Manoj Kumar S 2. Sivabalan P 3. Mohamed Nainar A

Sprint-4	Test Vital Page	USN-2	As a user, I will get the prediction result and accuracy on the test vital page.	3	High	4. Manoj Kumar S 5. Sivabalan P 6. Mohamed Nainar A

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-2	Dashboard	USN-3	Dashboard displays the symptoms, causes and medications for the Parkinson disease	2	Low	1. Manoj Kumar S 2. Shakul Kani N 3. Mohamed Nainar A
Sprint-1	Data Collection	USN-4	As an Administrator, I need to collect data (images of spirals and waves drawn by healthy people and Parkinson’s patients).	2	High	1. Sivabalan P 2. Shakul Kani N 3. Mohamed Nainar A

Sprint-1	Data Pre-Processing	USN-5	As an Administrator, I should clean my data and prepare it for model building by doing pre-processing activities such as resizing, visualizing the dataset and converting from RGB to grayscale	2	High	1. Manoj Kumar S 2. Shakul Kani N
Sprint-2	Model Building	USN-6	As an Administrator, I need to build the model using Random Forest Classifier for spiral images and Convolutional Neural Networks (CNN) for wave images	3	High	1. Sivabalan P 2. Mohamed Nainar A
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-3	Deployment of Model	USN-7	As an Administrator, I need to deploy the Machine Learning model that was built.	2	Medium	1. Sivabalan P 2. Shakul Kani N

Sprint-3	Building Frontend of the application	USN-8	As an Administrator, I need to build the website for the application using HTML, CSS etc.	2	High	1. Shakul Kani N 2. Mohamed Nainar A
Sprint-4	Connecting the ML model, Frontend and Backend	USN-9	As an Administrator, I can integrate the deployed model and web application using python flask server.	3	High	1. Manoj Kumar S 2. Shakul Kani N

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	14 Nov 2022

Velocity:

For example, imagine we have a 10-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)
In our project, we have a 6-days 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)

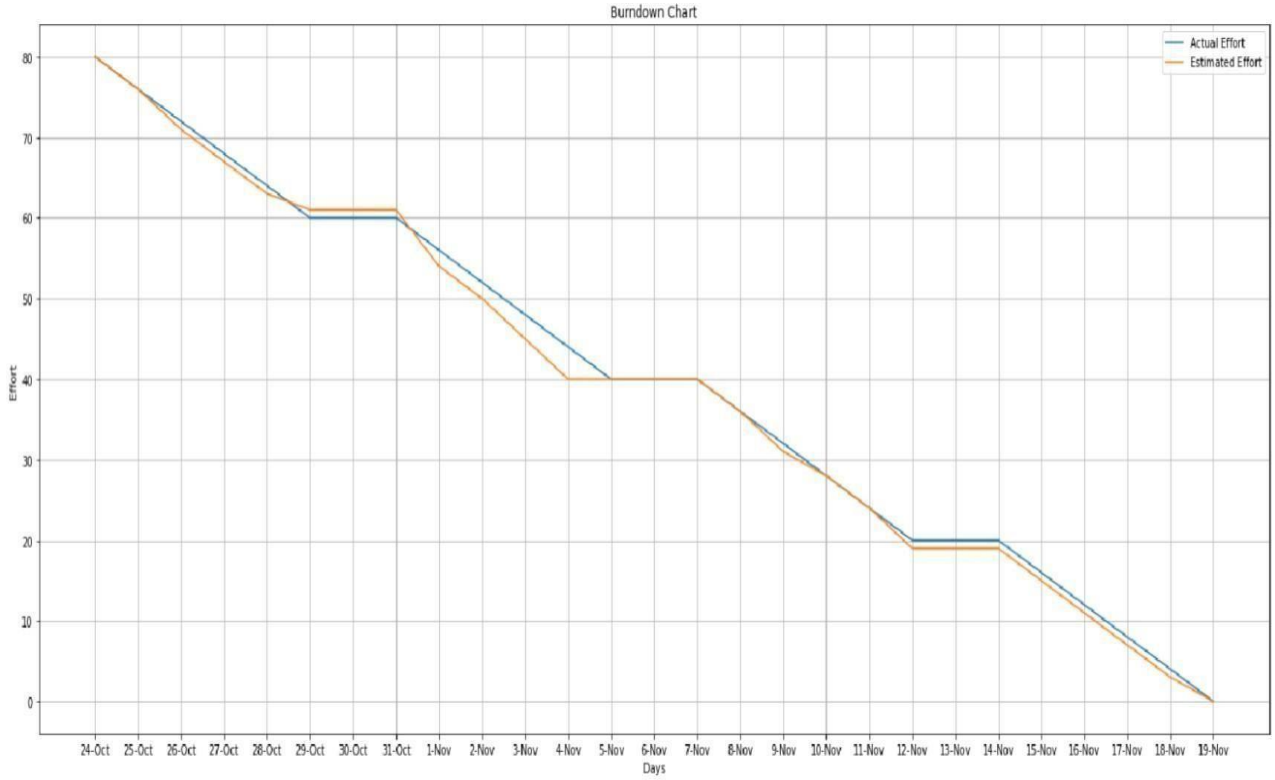
AV = Sprint Duration = 20 = 3.3 (approx.) Velocity 6

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 methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time

A burndown chart is almost a “must” have tool for a Scrum Team for the following main reasons:

- Monitoring the project scope creep
- Keeping the team running on schedule
- Comparing the planned work against the team progression.



-

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

Training The Model

- Once after splitting the data into train and test, the data should be fed to an algorithm to build a model.
- There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms that you can choose according to the objective that you might have it may be Classification algorithms are Regression algorithms.

1.Logistic Regression

2.Decision Tree Classifier

3.Random Forest Classifier

4.KNN

Random Forest classifier

Initialize our **Random Forest classifier** and train the model using a number of estimators as 100

```
waveModel = SVC(kernel="poly", degree=5)
spiralModel = KNeighborsClassifier(21)

#waveModel = KNeighborsClassifier(n)
#spiralModel = KNeighborsClassifier(21)
```

37] Python

```
veModel.fit(wave_train_x, wave_train_y)
spiralModel.fit(spiral_train_x, spiral_train_y)
```

38] Python

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=21)
```


```
waveModel = SVC(kernel="poly", degree=5)
spiralModel = KNeighborsClassifier(21)

#waveModel = KNeighborsClassifier(n)
#spiralModel = KNeighborsClassifier(21)
```

37] Python

```
veModel.fit(wave_train_x, wave_train_y)
spiralModel.fit(spiral_train_x, spiral_train_y)
```

38] Python

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=21)
```


5. CNN

Testing The Model

After training the model, the model should be tested by using the test data which is been separated while splitting the data for checking the functionality of the model.Here we are selecting 25 images from the test data and initialize the output images for montage we're going to create a montage so that we can share our work visually

- First, we randomly sample images from our testing set
- Our images list will hold each spiral image along with annotations added via OpenCV drawing functions .
- We proceed to loop over the random image indices.
- Inside the loop, each image is processed in the same manner as during training(convert to gray scale, resize, threshold) .
- From there we'll automatically classify the image using our new HOG + Random Forest based classifier and add color-coded annotations
- Each image is quantified with HOG features.
- Then the image is classified bypassing those features to model.predict .
- The class label is colored green for "healthy" and red otherwise .The label is drawn in the top left corner of the image using cv2.putText function .
- Each output image is then appended to an images list so that we can develop a montage
- The montage is then displayed until a key is pressed
- The **cv2.imshow()**function always takes two more functions to load and close the image. **cv2.waitKey()** function, you can provide any value to close the image and continue with further lines of code.

Model Evaluation

Evaluation is a process during the development of the model to check whether the model is the best fit for the given problem and corresponding data.

```
print("Validation Evaluation")
print(metrics.classification_report(wave_pred_y, wave_test_y))
```

```
Validation Evaluation
              precision    recall  f1-score   support

           0       0.93      0.78      0.85        18
           1       0.73      0.92      0.81        12

    accuracy                           0.83        30
   macro avg       0.83      0.85      0.83        30
weighted avg       0.85      0.83      0.84        30
```

```
print("Validation Evaluation")
print(metrics.classification_report(spiral_pred_y, spiral_test_y))
```

```
Validation Evaluation
              precision    recall  f1-score   support

           0       0.60      0.69      0.64        13
           1       0.73      0.65      0.69        17

    accuracy                           0.67        30
   macro avg       0.67      0.67      0.67        30
weighted avg       0.68      0.67      0.67        30
```

Classification Evaluation Metrics:

These model evaluation techniques are used to find out the accuracy of models built in the classification type of machine learning models. We have three types of evaluation methods.

- Accuracy_score
- Confusion matrix
- Roc- Auc Curve

Confusion Matrix

It is a matrix representation of the results of any binary testing.

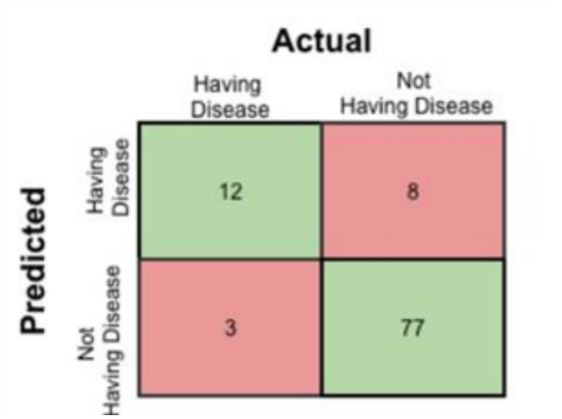


Fig: Confusion Matrix of prediction of a disease

1. True Positive: 12 (You have predicted the positive case correctly!)
2. True Negative: 77 (You have predicted negative case correctly!)

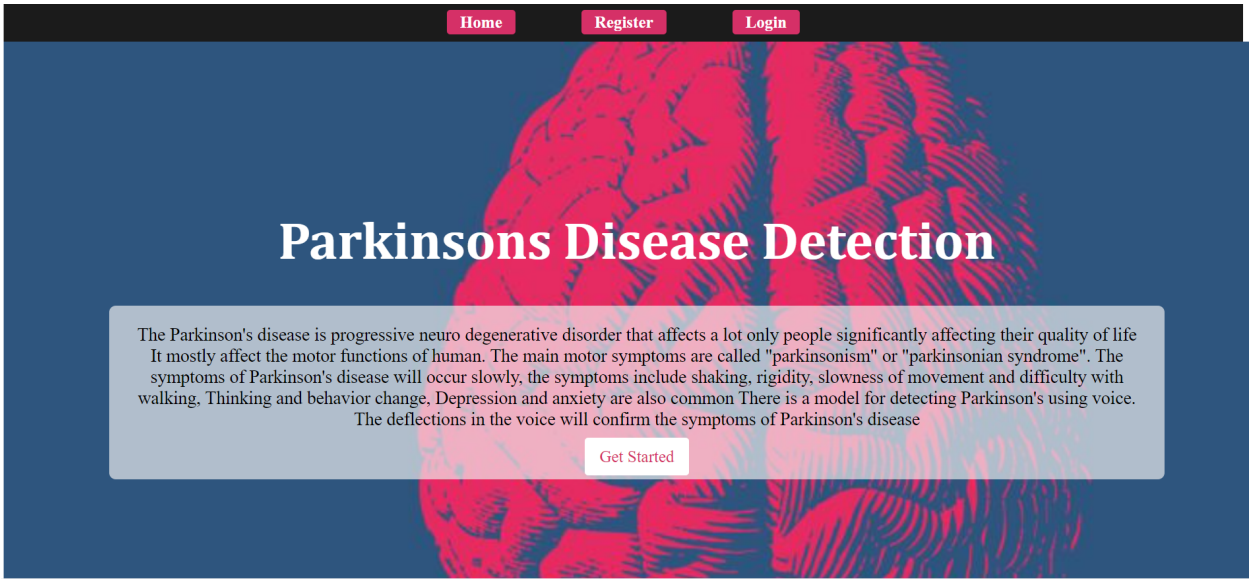
3. False Positive: 8 (You have predicted these people as having disease, but in actual they do not have.)
4. False Negative: 3 (Wrong predictions)
5. We can use the **predict** method on the model and pass **X_test** as a parameter to get the output as predictions.
6. The output of the confusion matrix and accuracy are as follows.

Save The Model

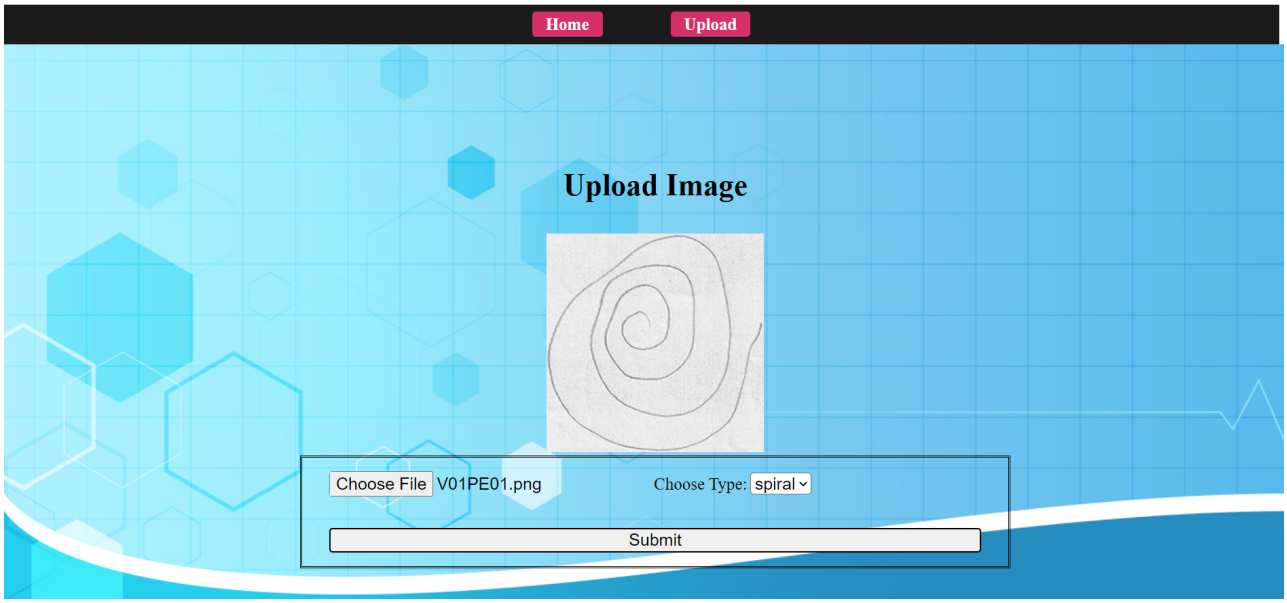
After building the model we have to save the model. Pickle in Python is primarily used in serializing and de-serializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions or transport data over the network. wb indicates write method and rd indicates the read method.

8. RESULTS

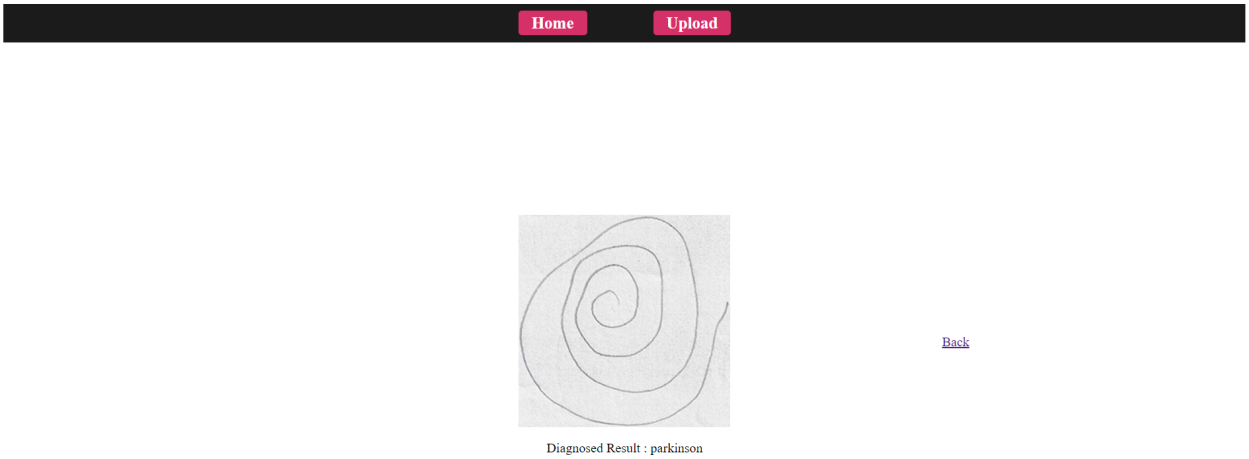
Home Page



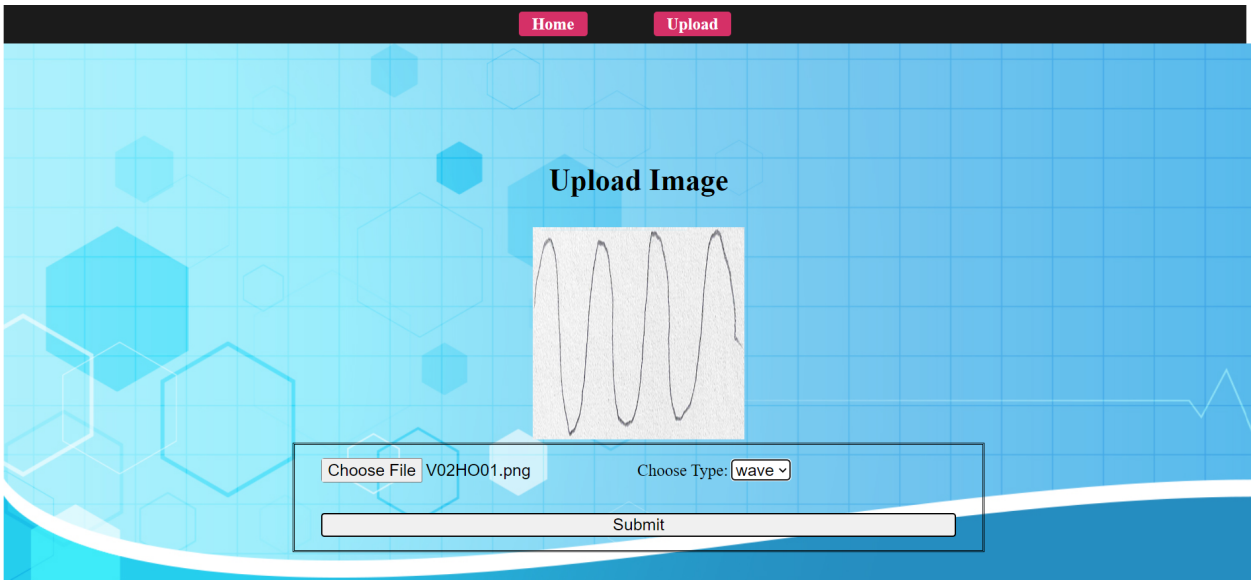
UPLOAD PAGE



OUTPUT PAGE (PARKINSON)

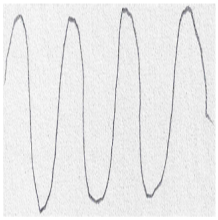


UPLOAD PAGE(HEALTHY)



OUTPUT PAGE (HEALTHY)

Home Upload



Back

Diagnosed Result : healthy

9. 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.

10. 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.

\

11. APPENDIX

Source Code

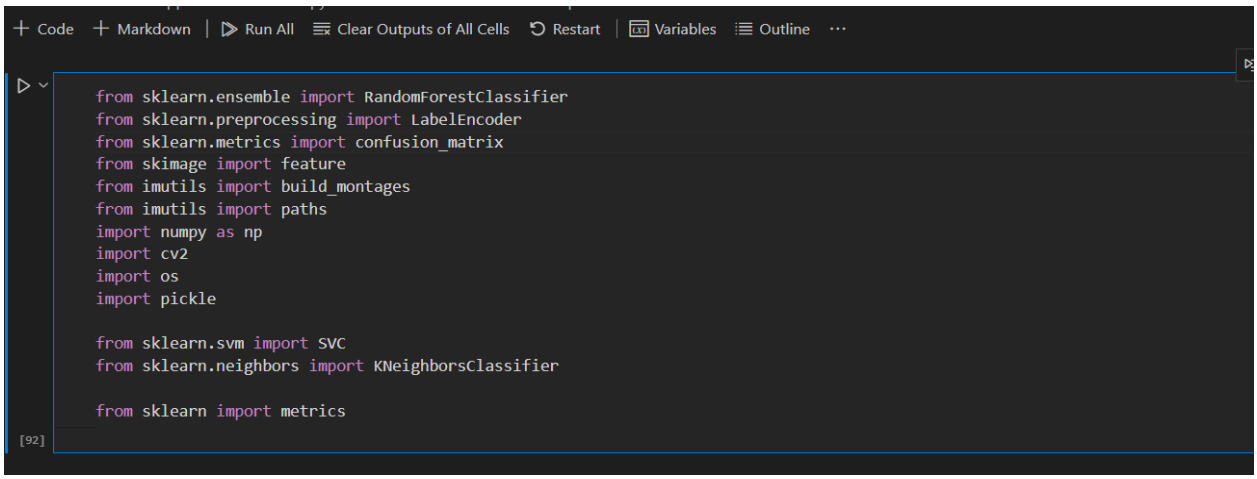
PROCESS_image

```python
def quantify_image(image):
    features = feature.hog(image, orientations=9, pixels_per_cell=(13, 13), cells_per_block=(4, 4), transform_sqrt=True, block_norm="L1")
    return features

def load_split(path):
    imagePaths = list(paths.list_images(path))
    data = []
    labels = []

    for imagePath in imagePaths:
        image = cv2.imread(imagePath)
        label = imagePath.split(os.path.sep)[-2]
        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)
        data.append(features)
        labels.append(label)
    return (np.array(data), np.array(labels))
```

TRAIN DEPENDICIES



MODEL EVALUATION

```
print("Validation Evaluation")
print(metrics.classification_report(wave_pred_y, wave_test_y))
```

```
Validation Evaluation
              precision    recall  f1-score   support

           0       0.93      0.78      0.85        18
           1       0.73      0.92      0.81        12

    accuracy                           0.83        30
   macro avg       0.83      0.85      0.83        30
weighted avg       0.85      0.83      0.84        30
```

```
print("Validation Evaluation")
print(metrics.classification_report(spiral_pred_y, spiral_test_y))
```

```
Validation Evaluation
              precision    recall  f1-score   support

           0       0.60      0.69      0.64        13
           1       0.73      0.65      0.69        17

    accuracy                           0.67        30
   macro avg       0.67      0.67      0.67        30
weighted avg       0.68      0.67      0.67        30
```

MODEL BUILDING_TRAINING

```
waveModel = SVC(kernel="poly", degree=5)
spiralModel = KNeighborsClassifier(21)

#waveModel = KNeighborsClassifier(n)
#spiralModel = KNeighborsClassifier(21)
```

Python

```
waveModel.fit(wave_train_x, wave_train_y)
spiralModel.fit(spiral_train_x, spiral_train_y)
```

Python

```
KNeighborsClassifier(n_neighbors=21)
```

LABEL_ENCODING

```
from flask import Flask, request, redirect, flash, send_from_directory, render_template, send_file



app = Flask(__name__, static_url_path='/')
app.config['SECRET_KEY'] = 'secret_key'
app.config['UPLOAD_FOLDER'] = 'static/uploads/'
```

FLASK APP

```
from flask import Flask, request, redirect, flash, send_from_directory, render_template, send_file



app = Flask(__name__, static_url_path='/')
app.config['SECRET_KEY'] = 'secret_key'
app.config['UPLOAD_FOLDER'] = 'static/uploads/'
```

FLASK ROUTES

Final Deliverables > app > main.py > ...

```python
from app import *
import os
from utils.predict import predict

@app.route("/", methods=["GET"])
def index():
    return render_template('index.html')



@app.route("/signin", methods=["POST", "GET"])
def signin():
    if request.method=="GET":
        return render_template('signin.html')


@app.route("/signup", methods=["POST", "GET"])
def signup():
    if request.method=="GET":
        return render_template('signup.html')


@app.route("/upload", methods=["POST", "GET"])
def upload():
    if request.method=="GET":
        return render_template('upload.html')
    if request.method=="POST":
        imgType = request.form["type"]
        file = request.files['file']

        imgPath = os.path.join(app.config['UPLOAD_FOLDER'], "upImage.jpg")
        file.save(imgPath)
        print(imgPath)
        prediction = predict(imgPath, imgType)
        return render_template('result.html', prediction=prediction, path='/uploads/upImage.jpg')

if __name__ == "__main__":
    app.run(debug=False)
```

GitHub & Project Demo Link :

GitHub Link : https://github.com/IBM-EPBL/IBM-Project-44995-1660727752

Demo Link :

https://drive.google.com/file/d/1X4KAydtVpWr5WMbRu-EvX7icSHK3trDB/view?usp=share_link