PROJECT REPORT ON

Detecting Parkinson's Disease Using Machine Learning

TEAM ID:PNT2022MID02435

Submitted by

Pramod	-	190801116
Preejith Raghavender.D.P	-	190801121
Preetha.A	-	190801122
Sanjana.M	-	190801146

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

TABLE OF CONTEXTS

1. INTRODUCTION

1.1 Project Overview

The overall view of this project is for detecting Parkinson’s disease

using Machine Learning. The detection is done by a different kind of data sets and algorithm. It is done by using deflection in the voice and other methods as a medical reports like dopamine level. As a key feature of this project we use patients hand drawn image of the spirals as input.

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.

The researchers found that the drawing speed was slower and the pen pressure is lower among Parkinson’s patients. One of the indications of Parkinson’s is tremors and rigidity in the muscles, making it difficult to draw smooth spirals and waves. It is possible to detect Parkinson’s disease using the drawings alone instead of measuring the speed and pressure of the pen on paper. Our goal is to quantify the visual appearance(using HOG method) of these drawings and then train a machine learning model to classify them. In this project, We are using, Histogram of Oriented Gradients (HOG) image descriptor along with a Random Forest classifier

to automatically detect Parkinson’s disease in hand-drawn images of spirals and waves.

1.2 Purpose

The Parkinson’s disease is progressive neuro degenerative disorder that affects a lot only people significantly affecting their quality of life. It mostly affect the motor functions of human. The main motor symptoms
are called "parkinsonism" or "parkinsonian syndrome". The symptoms of Parkinson’s disease will occur slowly, the symptoms include shaking, rigidity, slowness of movement and difficulty with walking, Thinking and behavior change, Depression and anxiety are also common. There is a model for detecting Parkinson’s using voice. The deflections in the voice will confirm the symptoms of Parkinson’s disease. This project showed 73.8% efficiency. In our model, a huge amount of data is collected from the normal
person and also previously affected person by Parkinson’s disease. these data is trained using machine learning algorithms. From the whole data 60% is used for training and 40% is used for testing. The data of any person can be entered in db to check whether the person is affected by Parkinsons disease or not.

Major drawback of the proposed model is that the accuracy rate is vastly affected or not.It does not intrude in stage detection, Since it could lead to serious problem in final stage as it has no treatment. In order to rectify this problem ,bringing out hand drawn images can give a clarity during the process of recovery.

2. LITERATURE SURVEY

2.1 Existing problem

The most prominent signs and symptoms of Parkinson's disease occur when nerve cells in the basal ganglia, an area of the brain that controls movement, become impaired and/or die. Normally, these nerve cells, or neurons, produce an important brain chemical known as dopamine. When the neurons die or become impaired, they produce less dopamine, which causes the movement problems associated with the disease. Scientists still do not know what causes the neurons to die.

Problem that is faced in this project is that Parkinson's disease influence level cannot be easily figured out until it has been in its final stage

2.2 References

1.Dr. C K GOMATHY[1], Mr. B. DHEERAJ KUMAR REDDY[2], Ms. B. VARSHA[3], Ms. B. VARSHINI

The Parkinson's disease is a Degenerative disorder that affects a lot only people significantly affecting their quality of life. It mostly affect the motor functions of human. The main motor symptoms are called "parkinsonism" or "parkinsonian syndrome". The symptoms of Parkinson's disease will occur slowly, the symptoms include shaking, rigidity, slowness of movement and difficulty with walking, Thinking and behavior change, Depression and anxiety are also common. There is a model for detecting Parkinson's using voice. The deflections in the voice will confirm the symptoms of Parkinson's disease. This project showed 73.8% efficiency. In our model, a huge amount of data is collected from the normal person and also previously affected person by Parkinson's disease. These data is trained using machine learning algorithms. From the whole data 60% is used for training and 40% is used for testing. The data of any person can be entered in db to check whether the person is affected by Parkinsons disease or not. There are 24 columns in the data set each column will indicate the symptom values of a patient except the status column. The status column has 0's and I's.those values will decide the person is effected with Parkinsons disease. I's indicate person is effected, 0's indicate normal conditions. Key Words: Parkinson's disease; machine learning (ML), XGBoost, Decision tree. Parkinson's disease is•a disorder of the central nervous system affecting movement and inducing tremors and stiffness a neurodegenerative disorder affecting dopamine neurons in brain. Parkinson’s disease is difficult to diagnose. Common diagnostic criteria require the medication before. In this model, the huge data is collected from previously affected person and then by using machine learning algorithm will process the user input data with previous data to check he/she affects from the disease.

2.Machine Learning for the Diagnosis of Parkinson's Disease: A Review of Literature Chemosensory Neuroanatomy Lab, Department of Anatomy, Université du Québec à Trois-Rivières (UQTR), Trois-Rivières, QC, Canada

The diagnosis of PD is traditionally based on motor symptoms. Despite the establishment of cardinal signs of PD in clinical assessments, most of the rating scales used in the evaluation of disease severity have not been fully evaluated and validated (Jankovic, 2008). Although non-motor symptoms (e.g., cognitive changes such as problems with attention and planning, sleep disorders, sensory abnormalities such as olfactory dysfunction) are present in many patients prior to the onset of PD (Jankovic, 2008; Tremblay et al., 2017), they lack specificity, are complicated to assess and/or yield variability from patient to patient (Zesiewicz et al., 2006). Therefore, non-motor symptoms do not yet allow for diagnosis of PD independently (Braak et al., 2003), although some have been used as supportive diagnostic criteria (Postuma et al., 2015).Machine learning techniques are being increasingly applied in the healthcare sector. As its name implies, machine learning allows for a computer program to learn and extract meaningful representation from data in a semi-automatic manner. For the diagnosis of PD, machine learning models have been applied to a multitude of data modalities, including handwritten patterns (Drotár et al., 2015; Pereira et al., 2018), movement (Yang et al., 2009; Wahid et al., 2015; Pham and Yan, 2018), neuroimaging (Cherubini et al., 2014a; Choi et al., 2017; Segovia et al., 2019), voice (Sakar et al., 2013; Ma et al., 2014), cerebrospinal fluid (CSF) (Lewitt et al., 2013; Maass et al., 2020), cardiac scintigraphy (Nuvoli et al., 2019), serum (Váradi et al., 2019), and optical coherence tomography (OCT) (Nunes et al., 2019). Machine learning also allows for combining different modalities, such as magnetic resonance imaging (MRI) and single-photon emission computed tomography (SPECT) data (Cherubini et al., 2014b; Wang et al., 2017), in the diagnosis of PD. By using machine learning approaches, we may therefore identify relevant features that are not traditionally used in the clinical diagnosis of PD and rely on these alternative measures to detect PD in preclinical stages or atypical forms.

3. Early Detection of Parkinson's Disease through Patient Questionnaire and Predictive Modelling R. Prashantha,Sumantra Dutta Roya, A Department of Electrical Engineering, Indian Institute of Technology Delhi, India.

Early detection of Parkinson's disease (PD) is important which can enable early initiation of Therapeutic interventions and management strategies. However, methods for early detection still Remain an unmet clinical need in PD. In this study, we use the Patient Questionnaire (PQ) portion From the widely used Movement Disorder Society-Unified Parkinson's Disease Rating Scale (MDS-UPDRS) to develop prediction models that can classify early PD from healthy normal Using machine learning techniques that are becoming popular in biomedicine: logistic regression, Random forests, boosted trees and support vector machine (SVM). We carried out both subjectwise and record-wise validation for evaluating the machine learning techniques. We observe that These techniques perform with high accuracy and high area under the ROC curve (both >95%) in Classifying early PD and healthy normal. The logistic model demonstrated statistically significant Fit to the data indicating its usefulness as a predictive model. It is inferred that these prediction Models have the potential to aid clinicians in the diagnostic process by joining the items of a Questionnaire through machine learning.

4.Imperative Role of Machine Learning Algorithm for DetectionOf Parkinson's Disease: Review, Challenges And Recommendations Arti Rana Ankur Dumka Rajesh Singh 3,4 , Manoj Kumar

Panda 5, Neeraj Priyadarshi And Bhekisipho Twala 7,Computer Science & Engineering, Veer Madho Singh Bhandari Uttarakhand Technical University.

The brain of humans is the main computing unit of the human body, and if there is Any minor accident in any part of the human body, then it will directly affect the other Organs. One of its silent effects is PD [1]. PD is a neurological disease that is incurable And is progressive over time [2]. As of 2020, an estimated 9.4 million people were still Living with this disease worldwide [3]. This disease

mostly affects people over the age Of 60 years, with only 4% of the cases occurring in people under the age of 50 [4]. The Symptoms of this disease are featured as motor and non-motor
[5]. The main motor Symptoms are slowness of movement, tremor, rapid eye movement disorder, shivering,Gait issue, and unstable posture [6,7]. Non-motor symptoms include hypotension, sweating in the body, fatigue, constipation, urinary problems, and loss of weight.

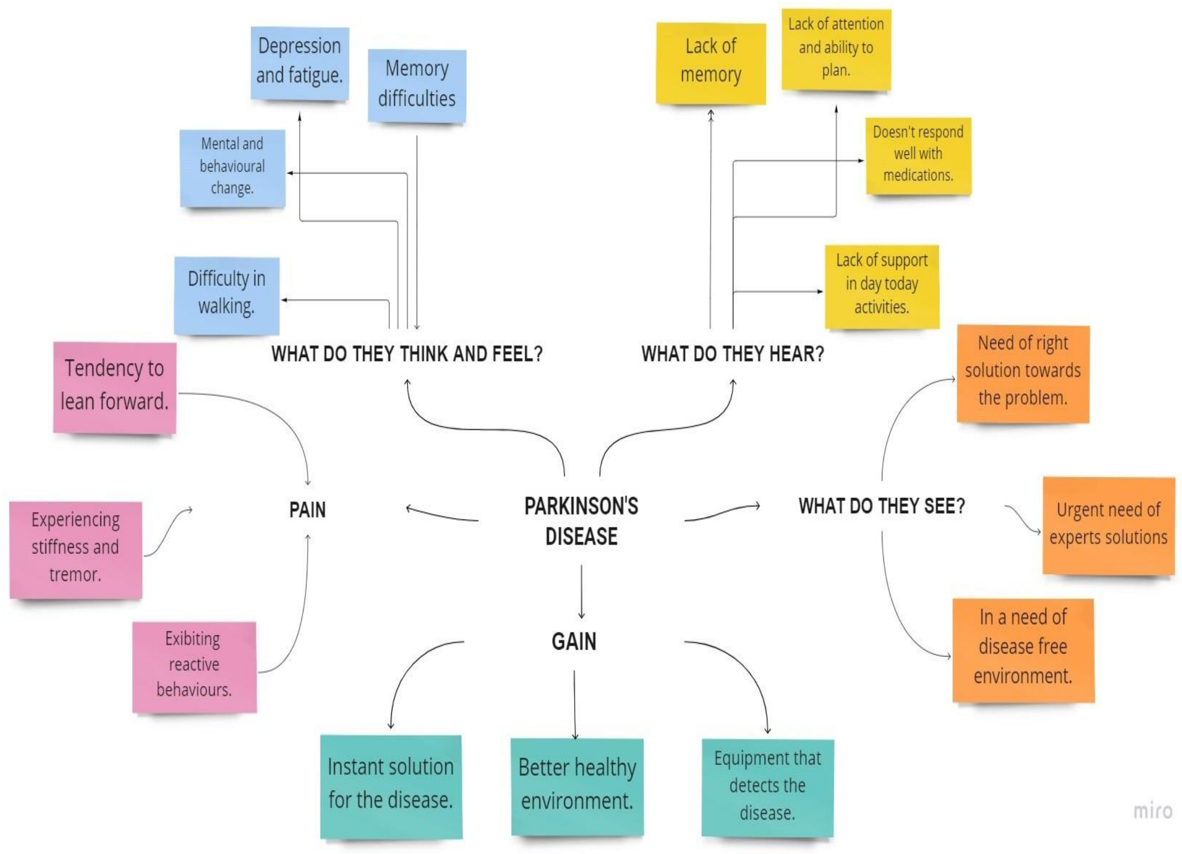
2.3 Problem Statement Definition

Who does the problem affect?	People who are men with minimization of nerve cells in primarily of village areas.
What are the boundaries of the problem?	People who are men with weaknerve cells and age over 50
What is the issue?	In real time life of human, if the person is affected by Parkinson disease then it produces the side effect problems like dry skin and dandruff which majorly affects the quality of the life. As the age gets progresses, it causes the people to face major problem with the nerve cells in thebrain.
When does the issue occur?	During the age excess of over50 as they will affect the people with loss of nerve cells in the brain.
Where is the issue coming?	It majorly occurs due to the age getting over 50 and as maximum in village areas.

Why is it important that we fix the problem?	It is very crucial to develop a application that detects the disease at good prediction rate so that it helps to get a clear line of disease symptoms during the times.
Which solution can be used to address this issue?	An machine learning powered web application model with the strong building of algorithm that helps to identify and predicts the disease with the identification of symptoms. It processes the breathing signals using a neural network that infers 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.
What methodology used to solve the issue?	Supervised and Un-supervised machine learning, Data mining , Computer vision with OpenCV, Python web application interface – Flask , IBM Cloud.

3. **IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas


Depression and fatigue.
Memory difficulties
Mental and behavioural change.
Difficulty in walking.
Lack of memory
Lack of attention and ability to plan.
Doesn't respond well with medications.
Lack of support in day today activities.
WHAT DO THEY THINK AND FEEL?
WHAT DO THEY HEAR?
Tendency to lean forward.
Need of right solution towards the problem.
Experiencing stiffness and tremor.
PAIN
PARKINSON'S DISEASE
WHAT DO THEY SEE?
Urgent need of experts solutions
Exibiting reactive behaviours.
GAIN
In a need of disease free environment.
Instant solution for the disease.
Better healthy environment.
Equipment that detects the disease.
miro

3.2 Ideation & Brainstorming

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.

The researchers found that the drawing speed was slower and the pen pressure is lower among Parkinson’s patients. One of the indications of Parkinson’s is tremors and rigidity in the muscles, making it difficult to draw smooth spirals and waves. It is possible to detect Parkinson’s disease using the drawings alone instead of measuring the speed and pressure of the pen on paper. Our goal is to quantify the visual appearance(using HOG method) of these drawings and then train a machine learning model to classify them. In this project, We are using, Histogram of Oriented Gradients (HOG) image descriptor along with a Random Forest classifier

to automatically detect Parkinson’s disease in hand-drawn images of spirals and waves.

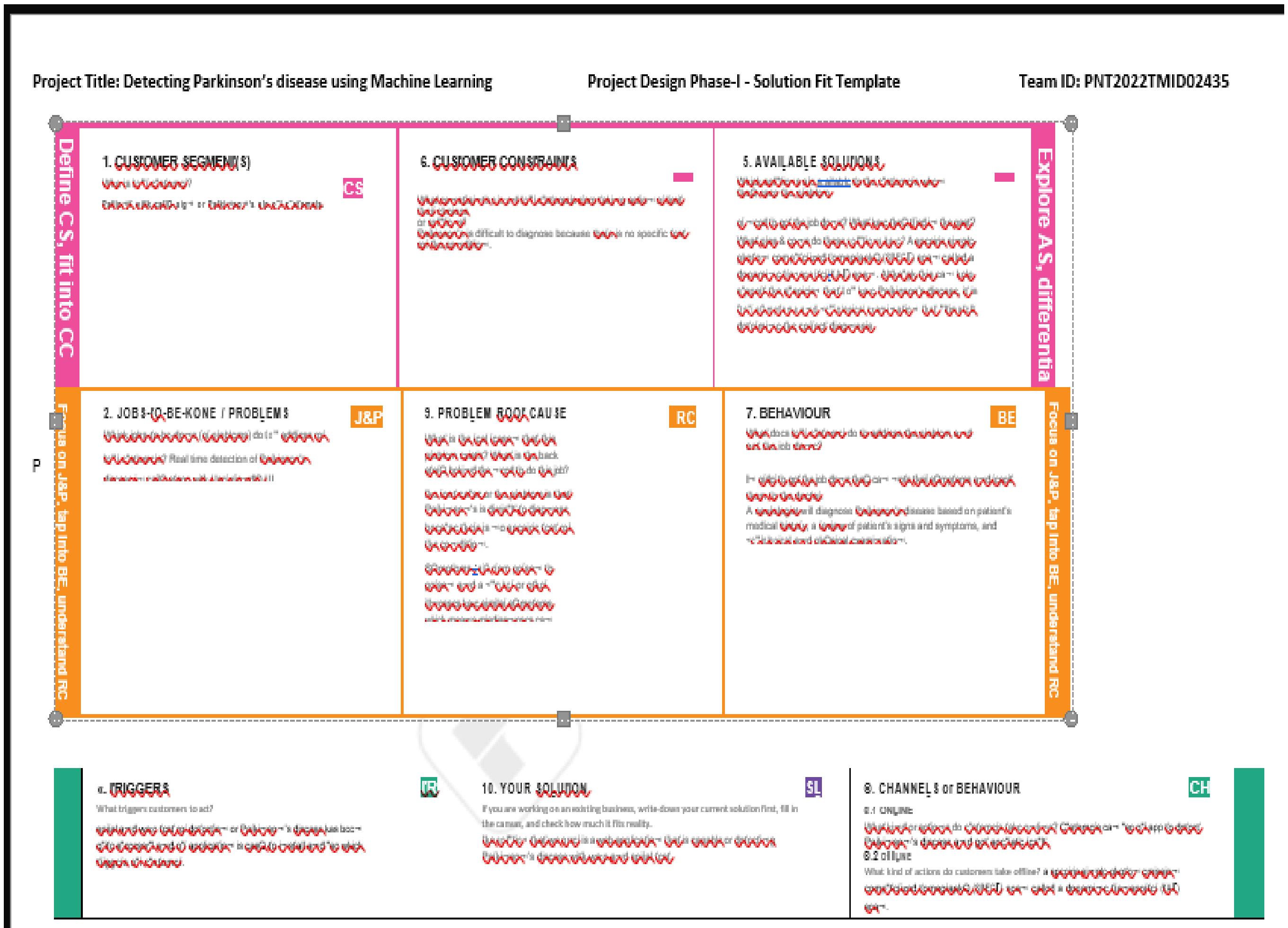
3.3 Proposed Solution

For this Project Design, we use the disease affected patients hand drawn images of spirals and the healthy person hand drawn images as a dataset. Split the Dataset into the Training and Testing data. Apply the data in the classified algorithm. Trains the model and calculate the accuracy of the model and use the model as a design for detecting the Parkinson’s disease. The key feature of the solution is we use the patients hand drawn image of the spiral as inputs.

The main therapy for Parkinson’s is levodopa. Nerve cells use levodopa to make dopamine to replenish the brain’s dwindling supply. Usually, people take levodopa along with another medication called carbidopa. Carbidopa prevents or reduces some of the side effects of levodopa therapy — such as nausea, vomiting, low blood pressure, and restlessness — and reduces the amount of levodopa needed to improve symptoms.

People living with Parkinson’s disease should never stop taking levodopa without telling their doctor. Suddenly stopping the drug may have serious side effects, like being unable to move or having difficulty breathing.

3.4 Problem Solution fit



Project Title: Detecting Parkinson's disease using Machine Learning

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMID02435

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer? [illegible]

6. CUSTOMER CONSTRAINTS

[illegible] is difficult to diagnose because there is no specific test [illegible]

5. AVAILABLE SOLUTIONS

[illegible]

Explore AS, differentia

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

What jobs-to-be-done (or problems) do you address for your customers? Real time detection of Parkinson's disease [illegible]

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? [illegible]

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done? [illegible]

A [illegible] will diagnose Parkinson's disease based on patient's medical history, a review of patient's signs and symptoms, and [illegible]

P

3. TRIGGERS TR

What triggers customers to act? [illegible]

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. [illegible]

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE [illegible]

8.2 OFFLINE

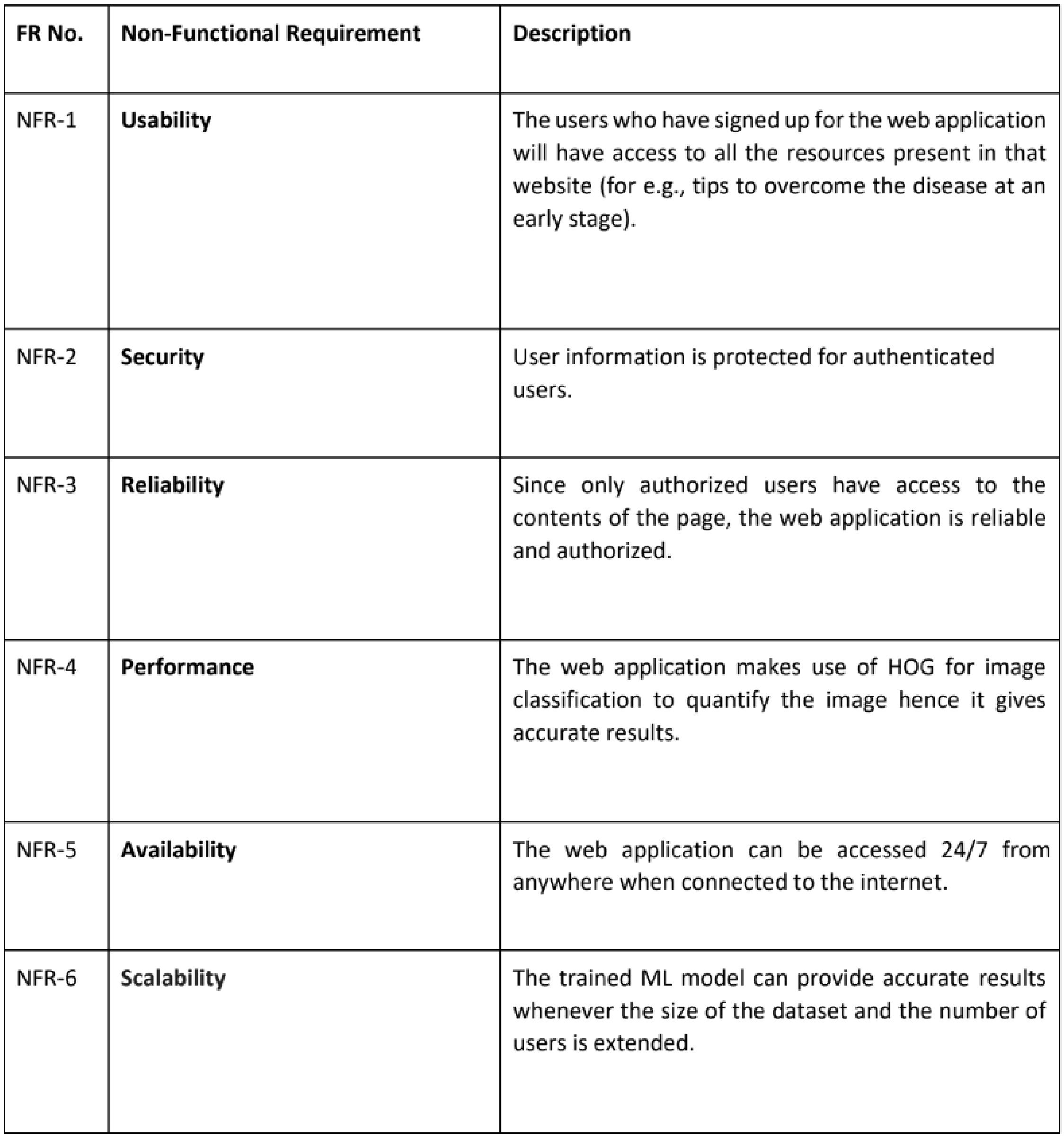
What kind of actions do customers take offline? [illegible]

4. REQUIREMENT ANALYSIS

4.1 Functional requirements

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	**Login Page**	User can login through their Username and password
FR-2	**Test Inputs**	The user inputs the symptoms into the Machine Learning model.
FR-3	**Result**	Accurately, get the result as positive or negative with percentage affected in a person by the Parkinson’s Disease.

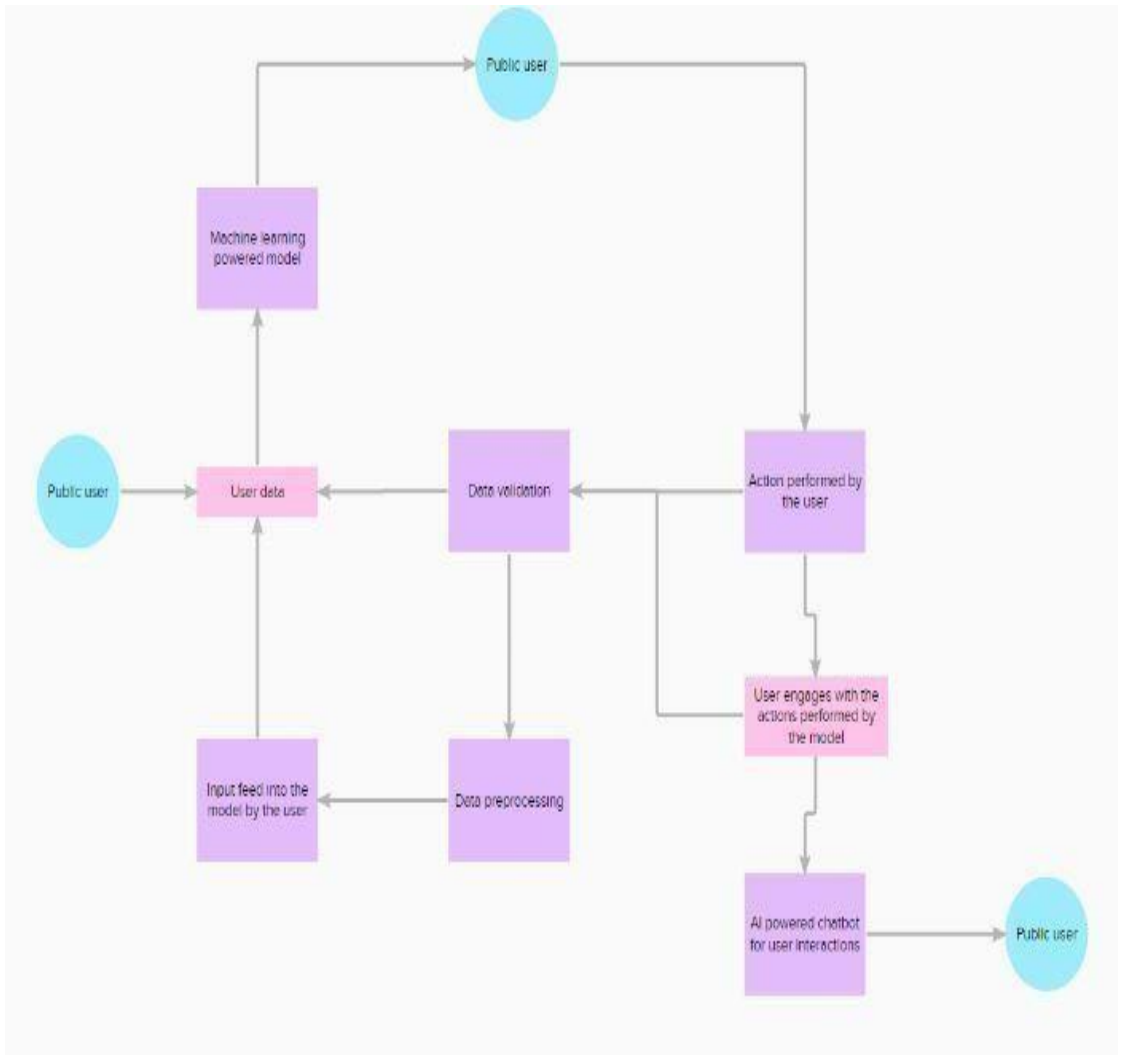
4.2 Non-Functional requirements



FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The users who have signed up for the web application will have access to all the resources present in that website (for e.g., tips to overcome the disease at an early stage).
NFR-2	**Security**	User information is protected for authenticated users.
NFR-3	**Reliability**	Since only authorized users have access to the contents of the page, the web application is reliable and authorized.
NFR-4	**Performance**	The web application makes use of HOG for image classification to quantify the image hence it gives accurate results.
NFR-5	**Availability**	The web application can be accessed 24/7 from anywhere when connected to the internet.
NFR-6	**Scalability**	The trained ML model can provide accurate results whenever the size of the dataset and the number of users is extended.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

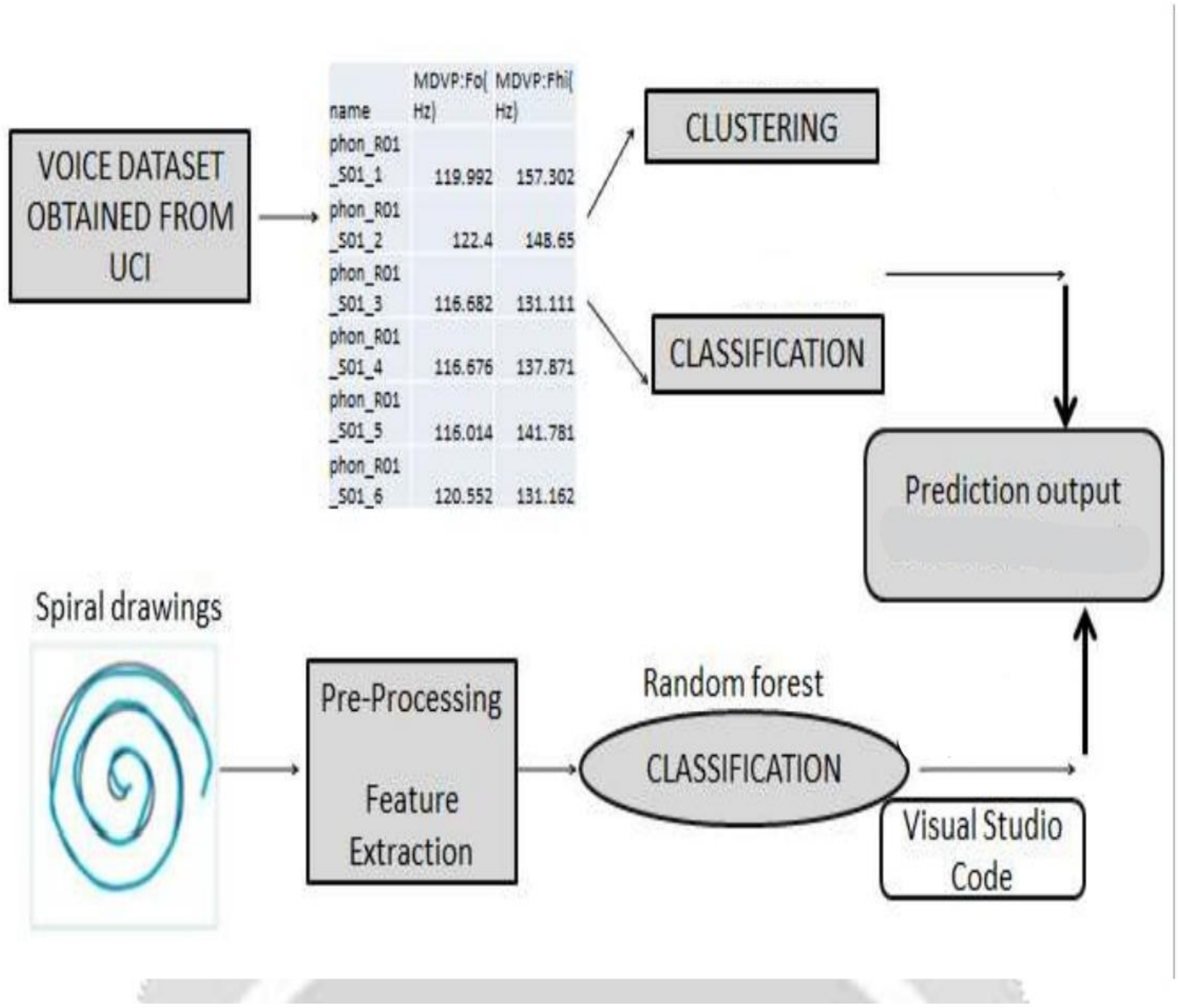

Public user
Machine learning powered model
Public user
User data
Data validation
Action performed by the user
User engages with the actions performed by the model
Input feed into the model by the user
Data preprocessing
AI powered chatbot for user interactions
Public user

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

5.2 Solution & Technical Architecture

ARCHITECTURE DIAGRAM:

The following is the solution architecture for detecting Parkinson’s using Machine learning:



name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)
phon_R01_S01_1	119.992	157.302
phon_R01_S01_2	122.4	148.65
phon_R01_S01_3	116.682	131.111
phon_R01_S01_4	116.676	137.871
phon_R01_S01_5	116.014	141.781
phon_R01_S01_6	120.552	131.162

By using machine learning techniques, the problem can be solved with minimal 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. Thus, combining both the results, the doctor can conclude normality or abnormality and prescribe the medicine based on the affected stage.

TECHNICAL ARCHITECTURE

The Technical Architecture involves the development of a technical blueprint with regard to the arrangement, interaction and interdependence of all elements so that the system-relevant requirements are met.

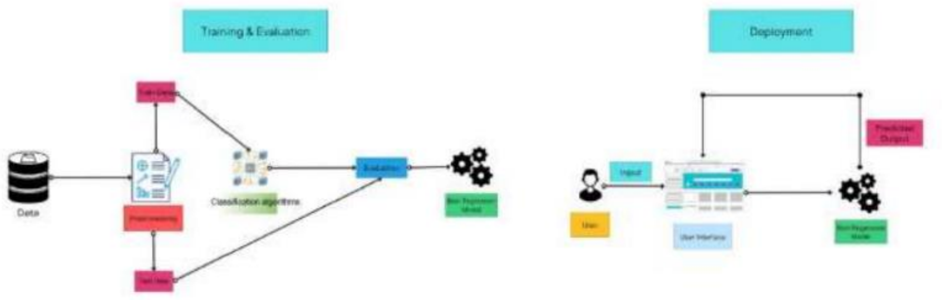
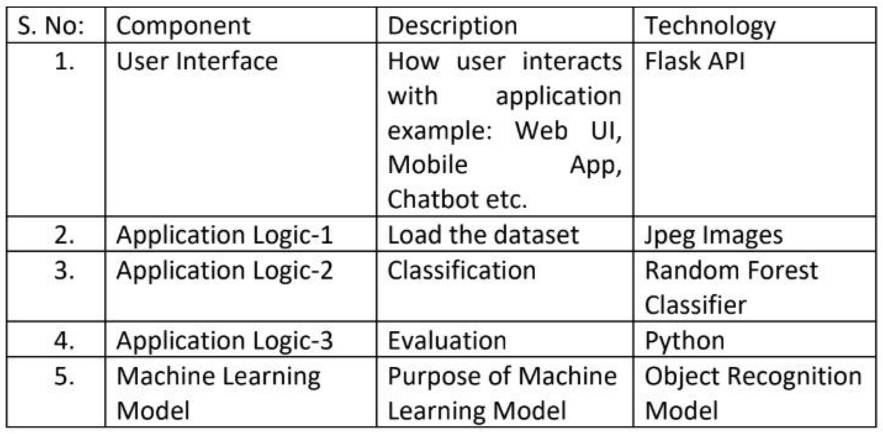
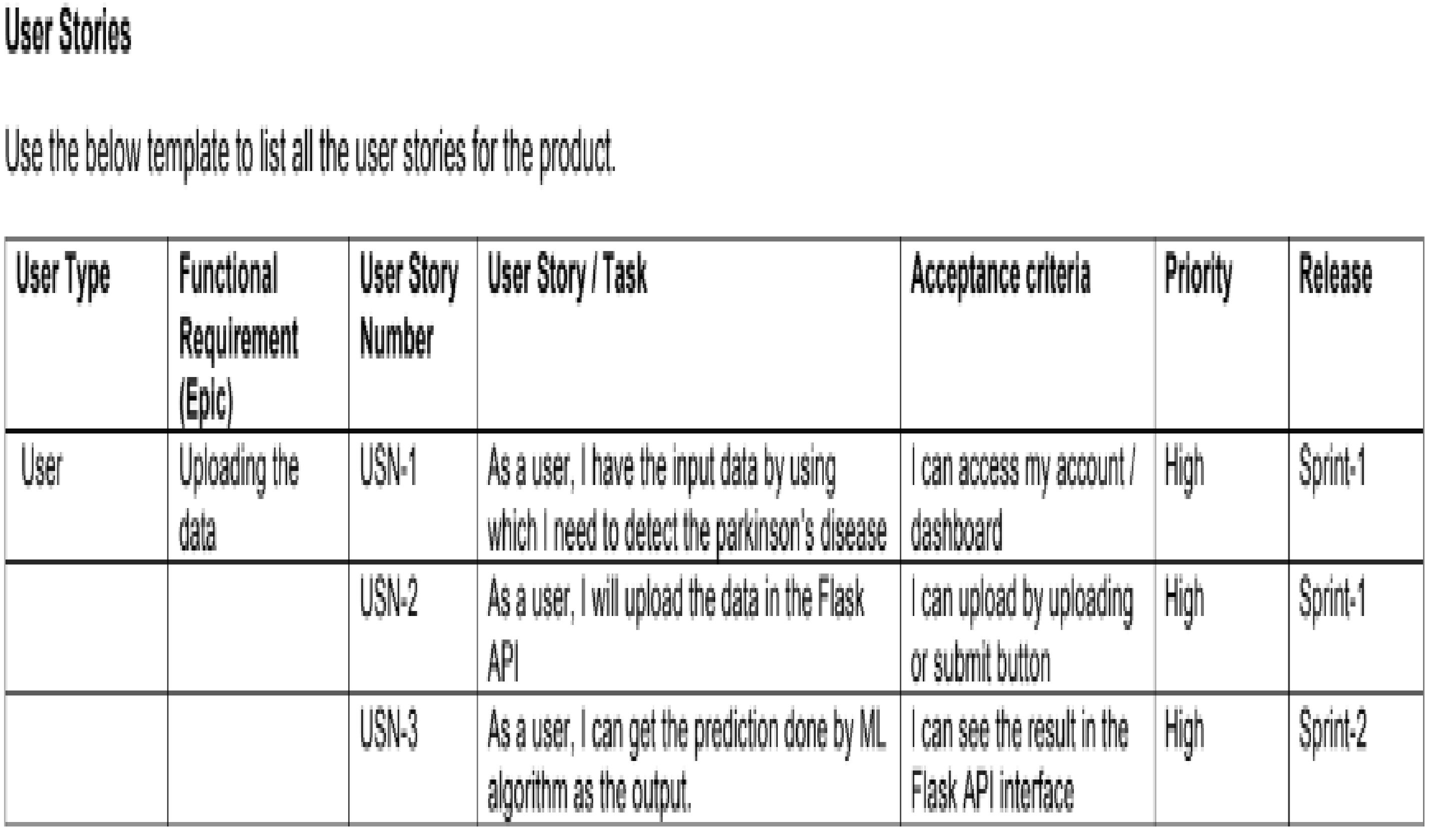


Table-1: Component & Technologies



S. No:	Component	Description	Technology
1.	User Interface	How user interacts with application example: Web UI, Mobile App, Chatbot etc.	Flask API
2.	Application Logic-1	Load the dataset	Jpeg Images
3.	Application Logic-2	Classification	Random Forest Classifier
4.	Application Logic-3	Evaluation	Python
5.	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model

5.3 User Stories



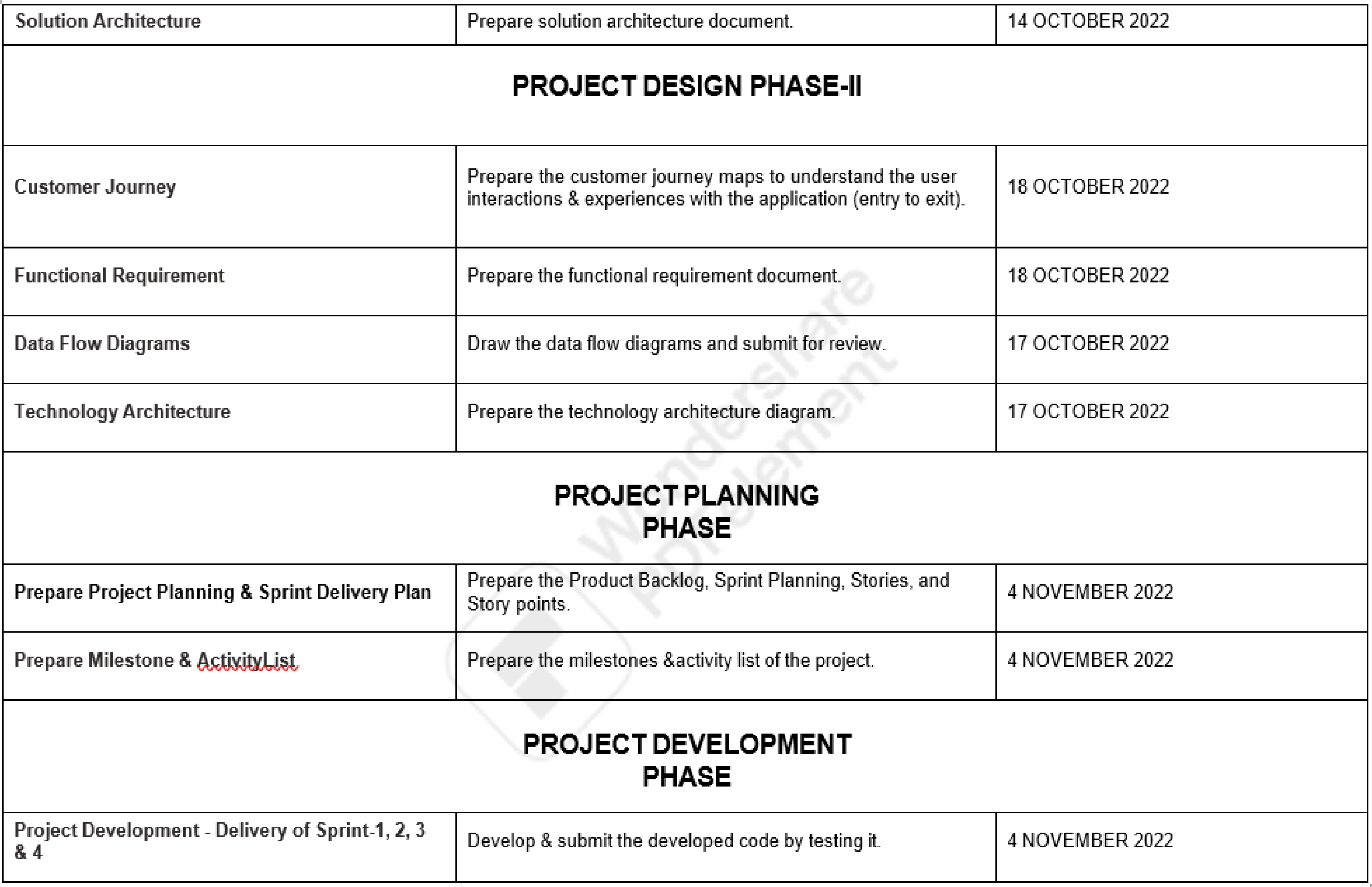
User Stories

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
User	Uploading the data	USN-1	As a user, I have the input data by using which I need to detect the parkinson's disease	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will upload the data in the Flask API	I can upload by uploading or submit button	High	Sprint-1
		USN-3	As a user, I can get the prediction done by ML algorithm as the output.	I can see the result in the Flask API interface	High	Sprint-2

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Delivery Schedule



Solution Architecture	Prepare solution architecture document.	14 OCTOBER 2022
PROJECT DESIGN PHASE-II		
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry to exit).	18 OCTOBER 2022
Functional Requirement	Prepare the functional requirement document.	18 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	17 OCTOBER 2022
Technology Architecture	Prepare the technology architecture diagram.	17 OCTOBER 2022
PROJECT PLANNING PHASE		
Prepare Project Planning & Sprint Delivery Plan	Prepare the Product Backlog, Sprint Planning, Stories, and Story points.	4 NOVEMBER 2022
Prepare Milestone & ActivityList	Prepare the milestones &activity list of the project.	4 NOVEMBER 2022
PROJECT DEVELOPMENT PHASE		
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	4 NOVEMBER 2022

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

Importing The Necessary Libraries

The first step is usually importing the libraries that will be needed in the program.

The required libraries to be imported to Python script are:

OpenCV:

OpenCV is a library of programming functions mainly aimed at real-time computer vision. Here, OpenCV is used for resizing. Rescaling, thresholding the image.

Imutils:

Imutils package has a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, and displaying Matplotlib images and video frames easier with OpenCV.

We will build_montages for visualization. Our paths import will help us to extract the file paths to each of the images in our dataset.

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

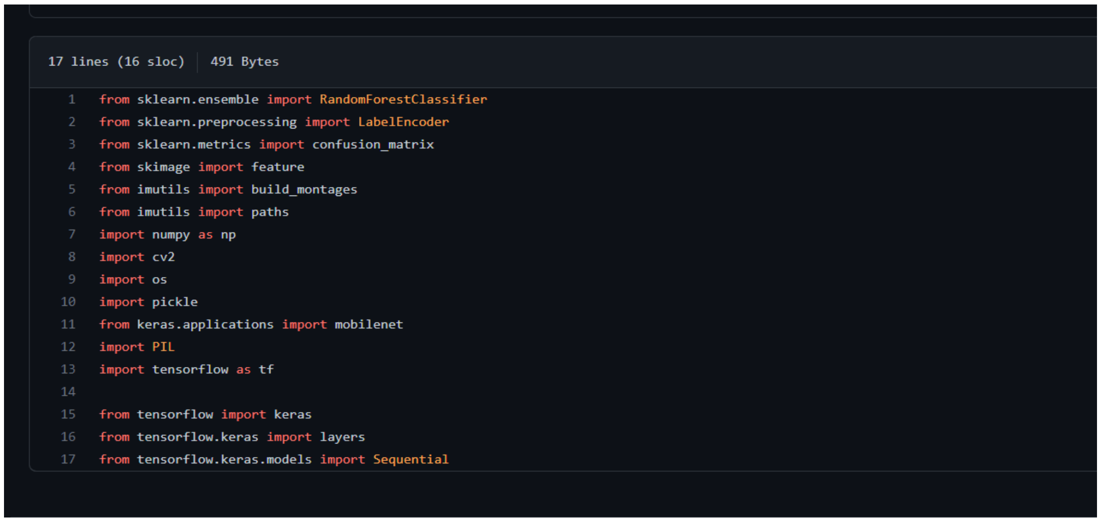
sklearn.preprocessing:
This package provides several common utility functions and transformer classes to change raw feature vectors into a representation that is more suitable for the downstream estimators.

Sklearn.ensemble
This package contains RandomForestClassifier and many more inbuilt algorithms.

Scikit-image

Scikit-image, or skimage, is an open-source Python package designed for image preprocessing.

Histogram of Oriented Gradients (HOG) will come from the feature import of scikit-image.



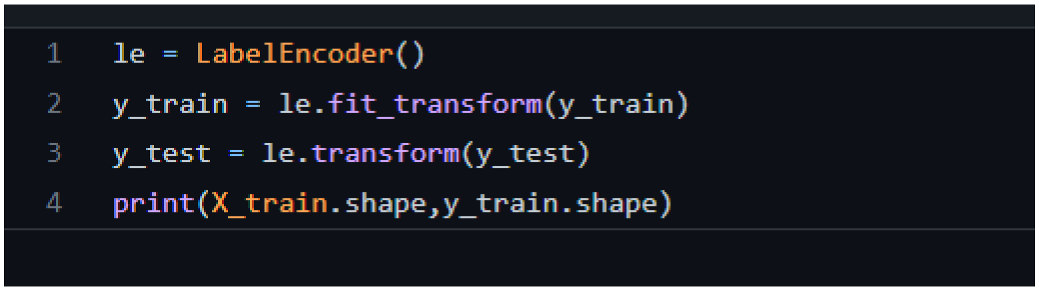
17 lines (16 sloc) | 491 Bytes

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
from skimage import feature
from imutils import build_montages
from imutils import paths
import numpy as np
import cv2
import os
import pickle
from keras.applications import mobilenet
import PIL
import tensorflow as tf

from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
```

Label Encoding

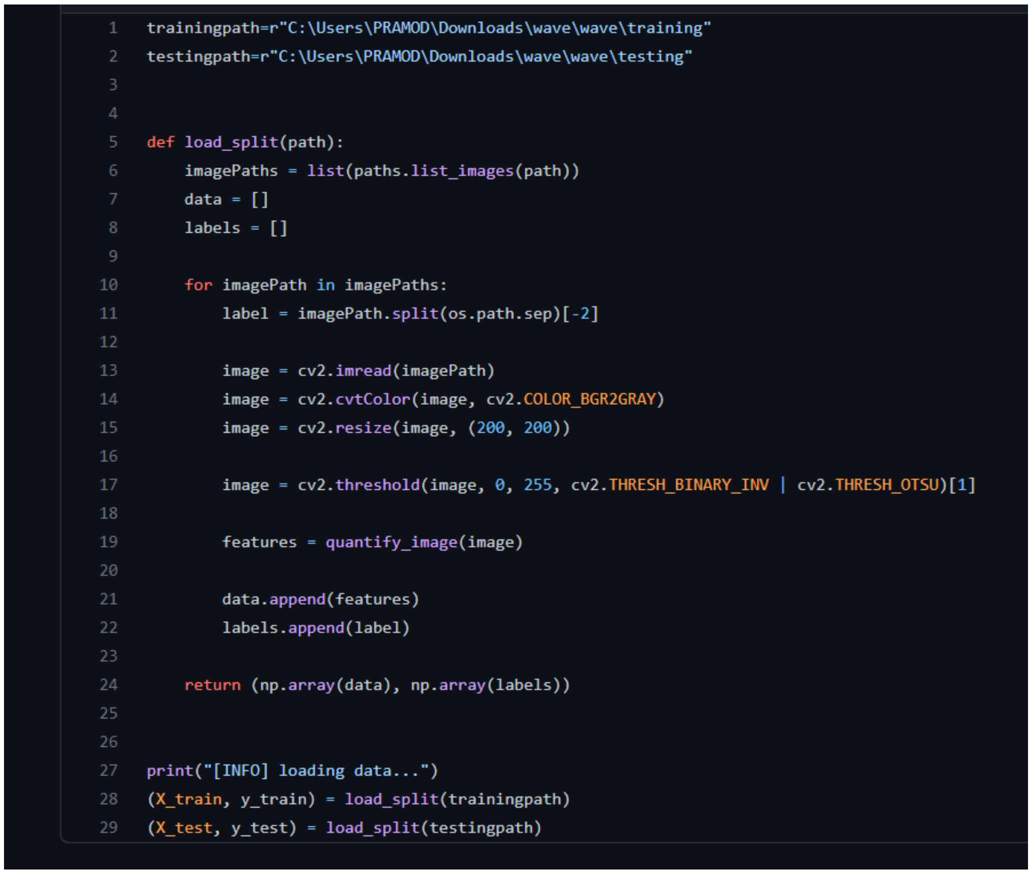
Label Encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. We apply Encoding in order to convert the values into 0's and 1's.

As we have observed that we have labels in our dataset. We need to convert them into binary values by using Label encoding. 0:healthy,1:Parkinson
Create an object **le** and fit the **y_train, y_test** using **fit_transform.**



```python
le = LabelEncoder()
y_train = le.fit_transform(y_train)
y_test = le.transform(y_test)
print(X_train.shape,y_train.shape)
```

Loading Train Data And Test Data



```python
trainingpath=r"C:\Users\PRAMOD\Downloads\wave\wave\training"
testingpath=r"C:\Users\PRAMOD\Downloads\wave\wave\testing"


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

    for imagePath in imagePaths:
        label = imagePath.split(os.path.sep)[-2]

        image = cv2.imread(imagePath)
        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))


print("[INFO] loading data...")
(X_train, y_train) = load_split(trainingpath)
(X_test, y_test) = load_split(testingpath)
```

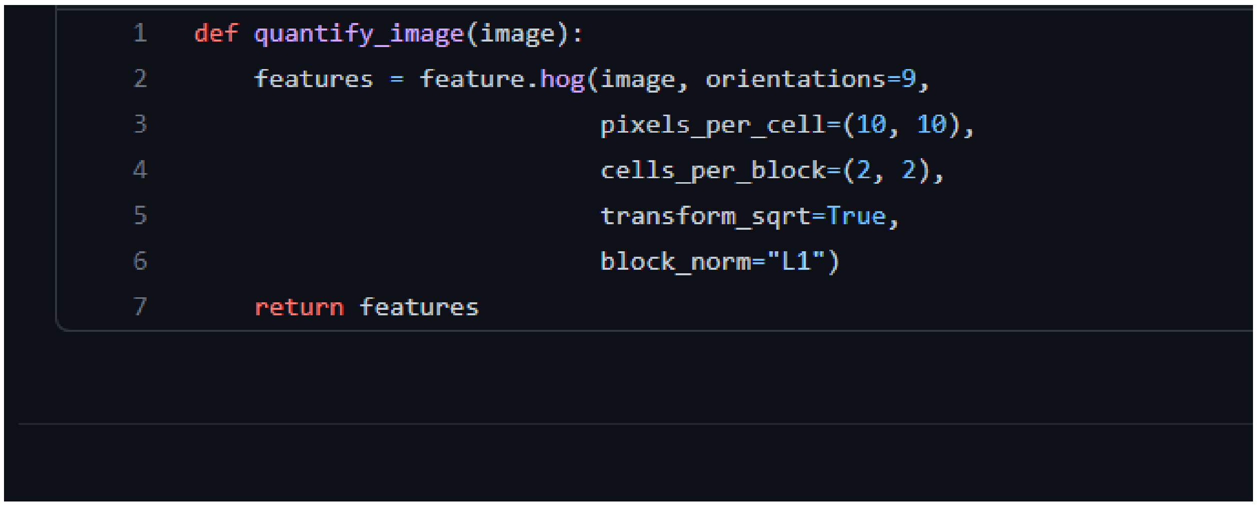
After importing the necessary libraries, the next step is to define the training path and testing path.

Our dataset contains both hand-drawn spiral and wave patterns. Here we are taking spiral patterns into consideration and training the model.
We split the data into train and test. Using the training dataset we train the model and the testing dataset is used to predict the results.

Quantifying Images

We will extract features from each input image with the quantify_image function.
HOG is a structural descriptor that will capture and quantify changes in local gradient in the input image. HOG will naturally be able to quantify how the directions of a both spirals and waves change.

It will be able to capture if these drawings have more of a “shake” to them, as we expect from a Parkinson’s patient.The most important parameters for the HOG descriptor are the **orientations**, **pixels_per_cell,** and the **cells_per_block**. These three parameters (along with the size of the input image) effectively control the dimensionality of the resulting feature vector.The resulting features are a 12,996-dim feature vector (list of numbers) quantifying the wave or spiral. We’ll train a Random Forest classifier on top of the features from all images in the dataset.



```python
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
```

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

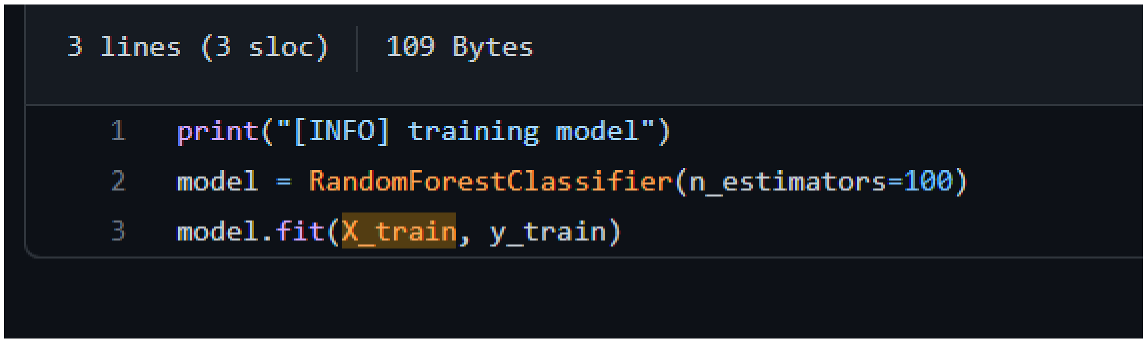
2.Decision Tree Classifier

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



```
3 lines (3 sloc) | 109 Bytes

print("[INFO] training model")
model = RandomForestClassifier(n_estimators=100)
model.fit(X_train, y_train)
```

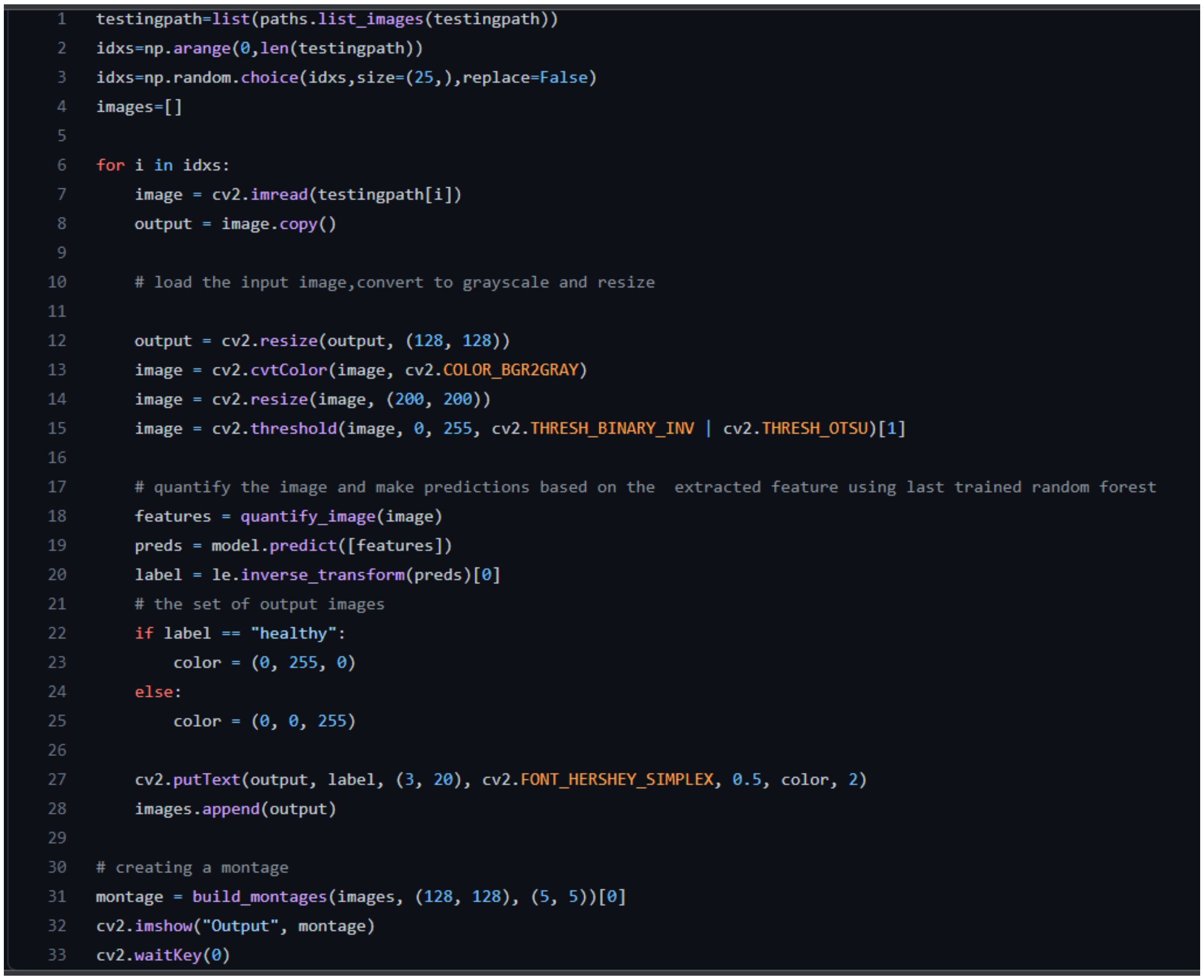
Testing The Model

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.



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

for i in idxs:
    image = cv2.imread(testingpath[i])
    output = image.copy()

    # load the input image,convert 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 feature using last trained random forest
    features = quantify_image(image)
    preds = model.predict([features])
    label = le.inverse_transform(preds)[0]
    # the set of output images
    if label == "healthy":
        color = (0, 255, 0)
    else:
        color = (0, 0, 255)

    cv2.putText(output, label, (3, 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
    images.append(output)

# creating a montage
montage = build_montages(images, (128, 128), (5, 5))[0]
cv2.imshow("Output", montage)
cv2.waitKey(0)
```

Model Evaluation

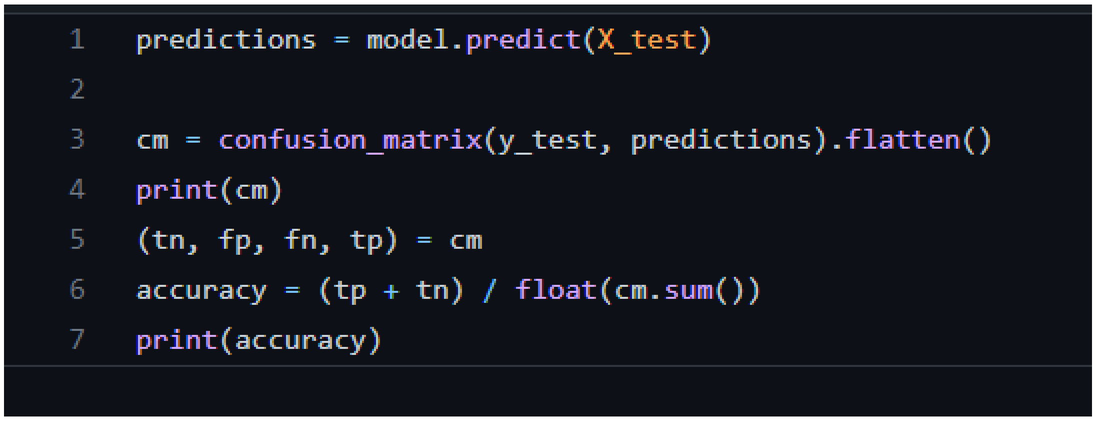
Evaluation is a process during the development of the model to check whether the model is the best fit for the given problem and corresponding data.

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.



```
predictions = model.predict(X_test)

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

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.

This is done by the below code

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

Building HTML Pages

For this project create four HTML files namely

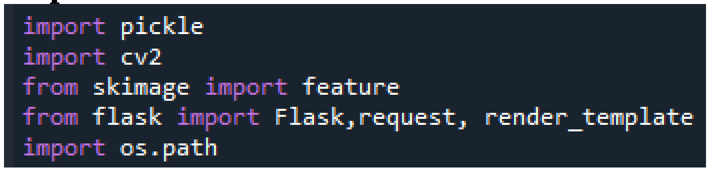
- about.html

- base.html
- index6.html
- info.html

and save them in templates folder
download the HTML files from the given link in the prerequisites section.

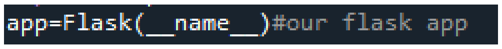
Build Python Code

Import the libraries



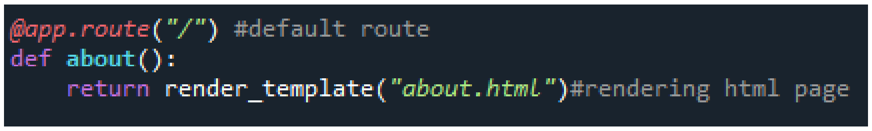
```python
import pickle
import cv2
from skimage import feature
from flask import Flask,request, render_template
import os.path
```

Importing the flask module in the project is mandatory. Flask constructor takes the name of the current module (__name__) as argument.



```python
app=Flask(__name__)#our flask app
```

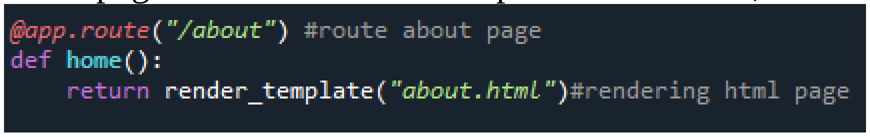
Render HTML Page:



```python
@app.route("/") #default route
def about():
    return render_template("about.html")#rendering html page
```

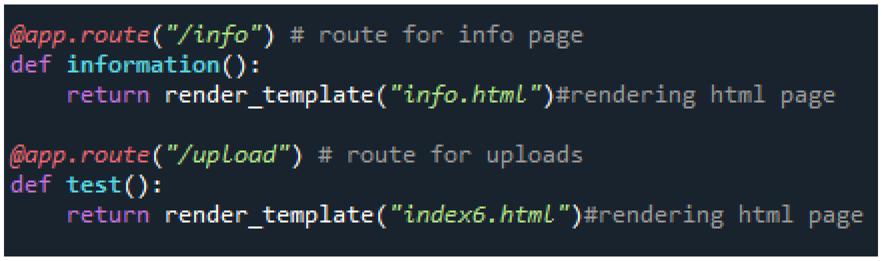
Here, declared constructor is used to route to the HTML page created earlier.

In the above example, '/' URL is bound with about.html function. Hence, when the home page of the web server is opened in browser, the html page is rendered.



```python
@app.route("/about") #route about page
def home():
    return render_template("about.html")#rendering html page
```

Here, **"about.html"** is rendered when home button is clicked on the UI.

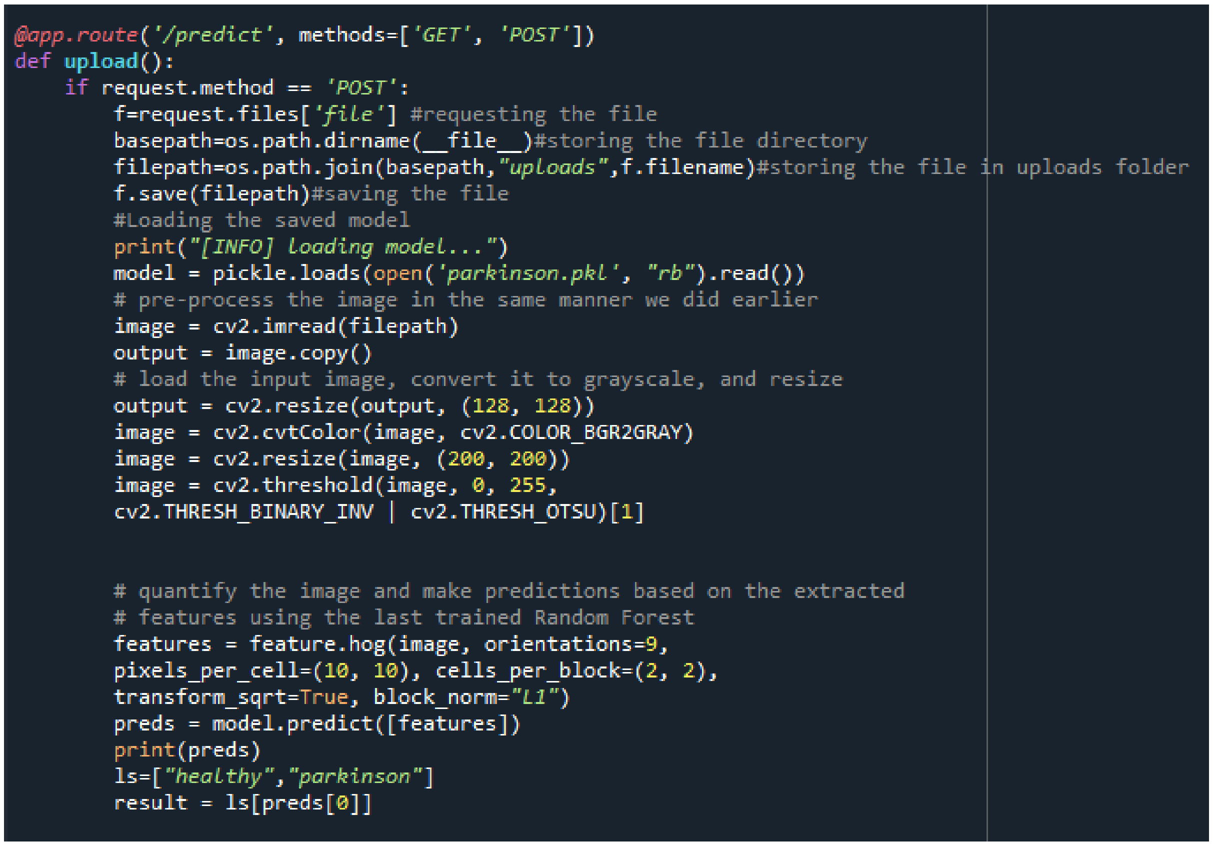


```python
@app.route("/info") # route for info page
def information():
    return render_template("info.html")#rendering html page

@app.route("/upload") # route for uploads
def test():
    return render_template("index6.html")#rendering html page
```

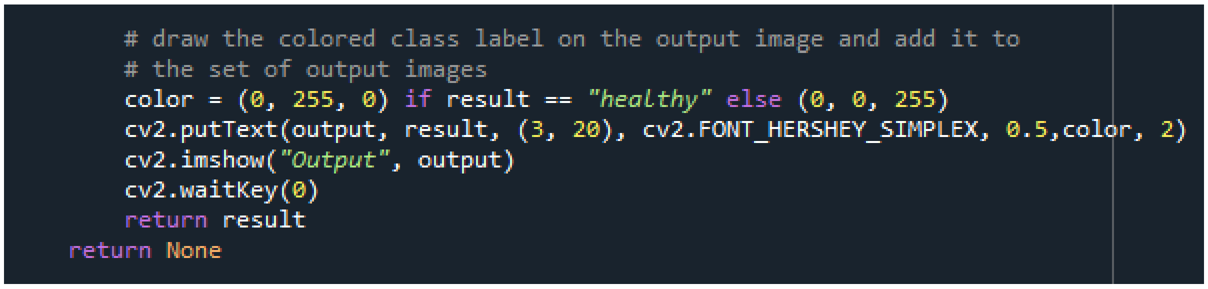
Similarly, **info.html** and **index6.html** are rendered when info and predict buttons are clicked on UI.

Retrieve the Values from UI

Whenever you give the inputs from the html page, the values can be retrieved using POST Method.



```python
@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        f=request.files['file'] #requesting the file
        basepath=os.path.dirname(__file__)#storing the file directory
        filepath=os.path.join(basepath,"uploads",f.filename)#storing the file in uploads folder
        f.save(filepath)#saving the file
        #Loading the saved model
        print("[INFO] loading model...")
        model = pickle.loads(open('parkinson.pkl', "rb").read())
        # 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]]
```

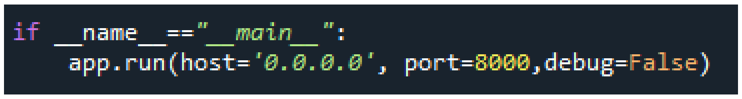


```
        # draw the colored class label on the output image and add it to
        # the set of output images
        color = (0, 255, 0) if result == "healthy" else (0, 0, 255)
        cv2.putText(output, result, (3, 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5,color, 2)
        cv2.imshow("Output", output)
        cv2.waitKey(0)
        return result
    return None
```

Here we are routing our app to upload() function. This function retrieves all the values from the HTML page using Post request. We are requesting to upload image using the request function.

We take the input from the user, and preprocess the image(convert to gray scale, resize, threshold)).It is necessary to preprocess the data so as give it to the model to predict the output. Once the output is predicted, result is shown on opencv window and html page.

Main Function:



```
if __name__=="__main__":
    app.run(host='0.0.0.0', port=8000,debug=False)
```

8. TESTING

8.1 Test Cases

Verify if the user is able to see the Homepage when they click on the link

• Verify the UL elements in Homepage

1. Home icon
2. Information icon
3. Predict icon

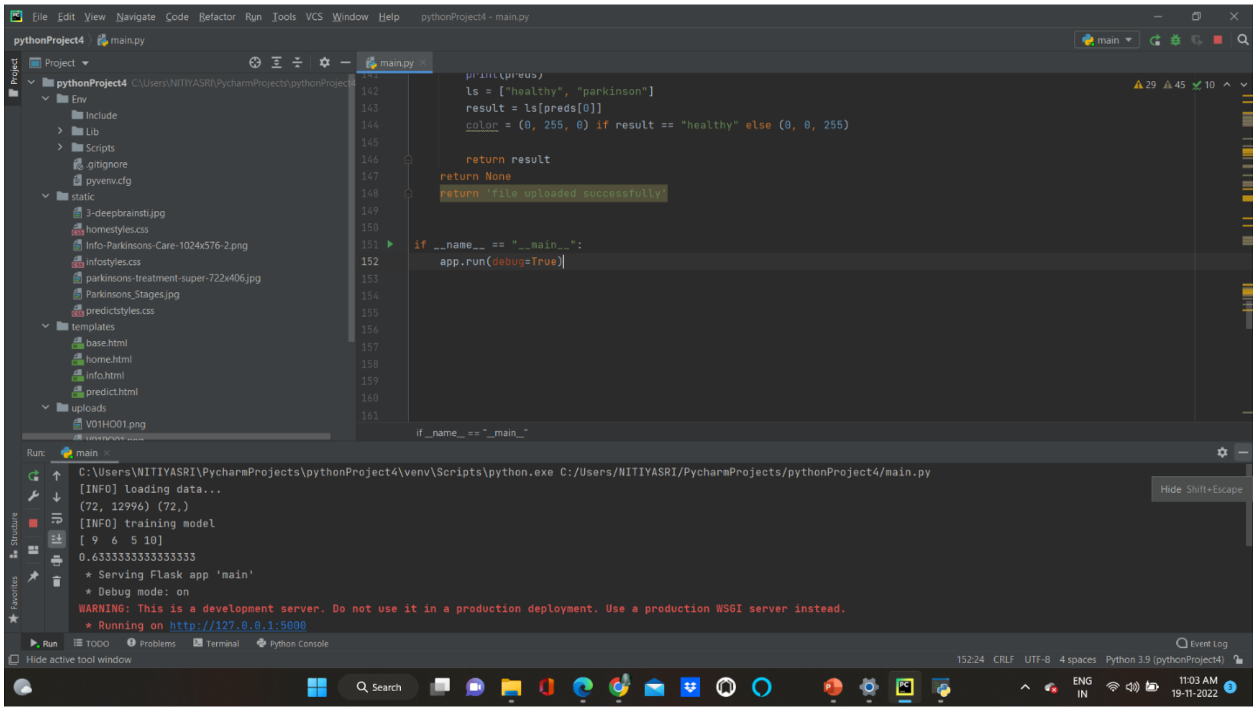
• Verify if the Information and the predict icons functions properly

• Verify if the user is able to view the upload file option and its function

• Verify if the user is able to choose the file from the local file system and click on predict to find the predicted result

• Verify if the user is able to select the invalid file formats

Model Building Results:



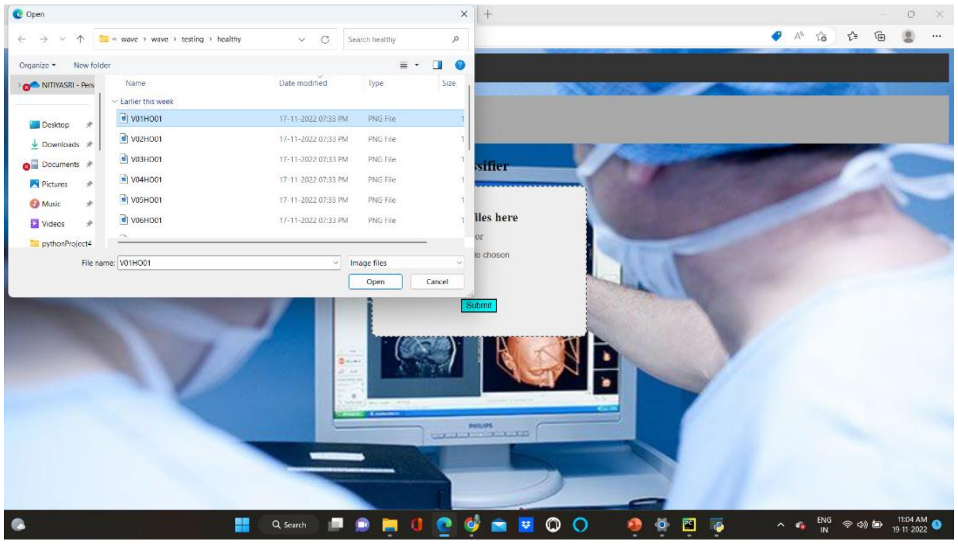
```python
        print(preds)
        ls = ["healthy", "parkinson"]
        result = ls[preds[0]]
        color = (0, 255, 0) if result == "healthy" else (0, 0, 255)

        return result
    return None
    return 'file uploaded successfully'


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

```
C:\Users\NITIYASRI\PycharmProjects\pythonProject4\venv\Scripts\python.exe C:/Users/NITIYASRI/PycharmProjects/pythonProject4/main.py
[INFO] loading data...
(72, 12996) (72,)
[INFO] training model
[ 9  6  5 10]
0.6333333333333333
 * Serving Flask app 'main'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
```

Test case 1:

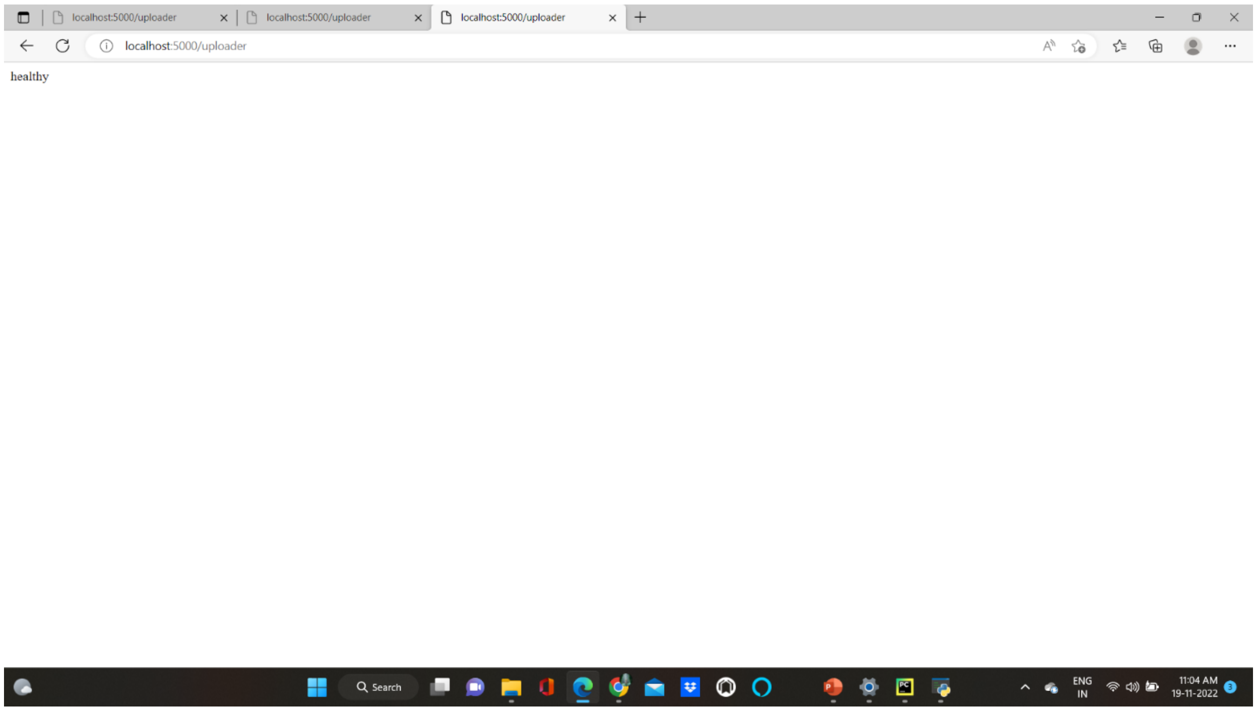
If input is given as healthy persons spiral

Then the output is,



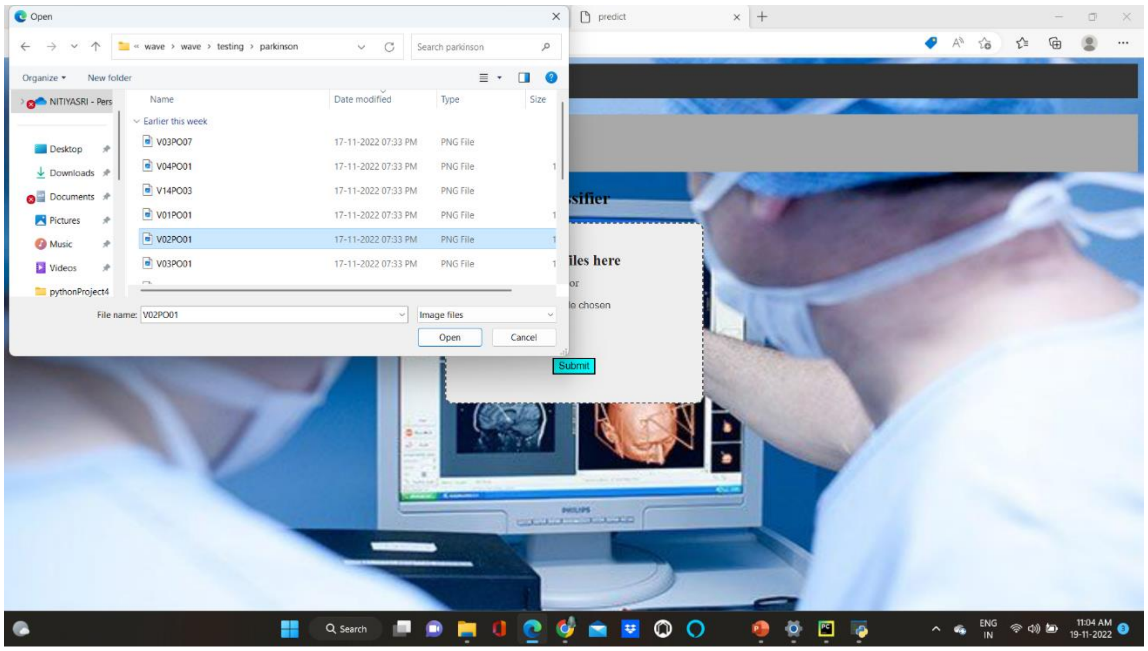
Name	Date modified	Type
V01HO01	17-11-2022 07:33 PM	PNG File
V02HO01	17-11-2022 07:33 PM	PNG File
V03HO01	17-11-2022 07:33 PM	PNG File
V04HO01	17-11-2022 07:33 PM	PNG File
V05HO01	17-11-2022 07:33 PM	PNG File
V06HO01	17-11-2022 07:33 PM	PNG File

File name: V01HO01

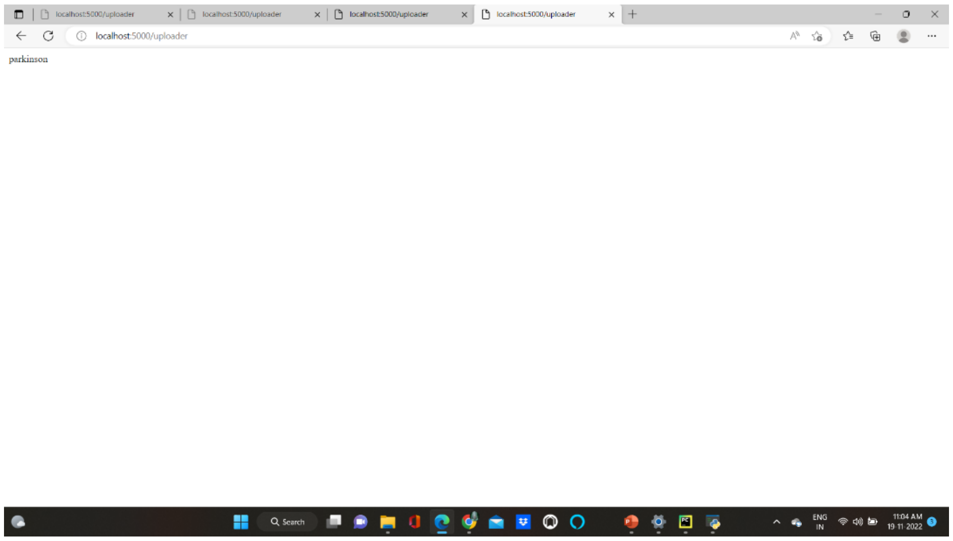

healthy

Test case 2:

If the input is given as Parkinson's person spiral


Open
wave › wave › testing › parkinson
Search parkinson
Organize
New folder
Name
Date modified
Type
Size
Earlier this week
V03PO07 17-11-2022 07:33 PM PNG File
V04PO01 17-11-2022 07:33 PM PNG File
V14PO03 17-11-2022 07:33 PM PNG File
V01PO01 17-11-2022 07:33 PM PNG File
V02PO01 17-11-2022 07:33 PM PNG File
V03PO01 17-11-2022 07:33 PM PNG File
File name: V02PO01
Image files
Open
Cancel
Submit

Then the output will be,



parkinson

8.2 User Acceptance Testing

In UAT design phase, test engineers are preparing UAT testcases as per the business requirements. AT test coverage should be with Alpha and beta testing.After having complete idea about business requirements and have a discussion with BA or Product Owner one can proceed with UAT Test case design/mapping to UAT test suite.

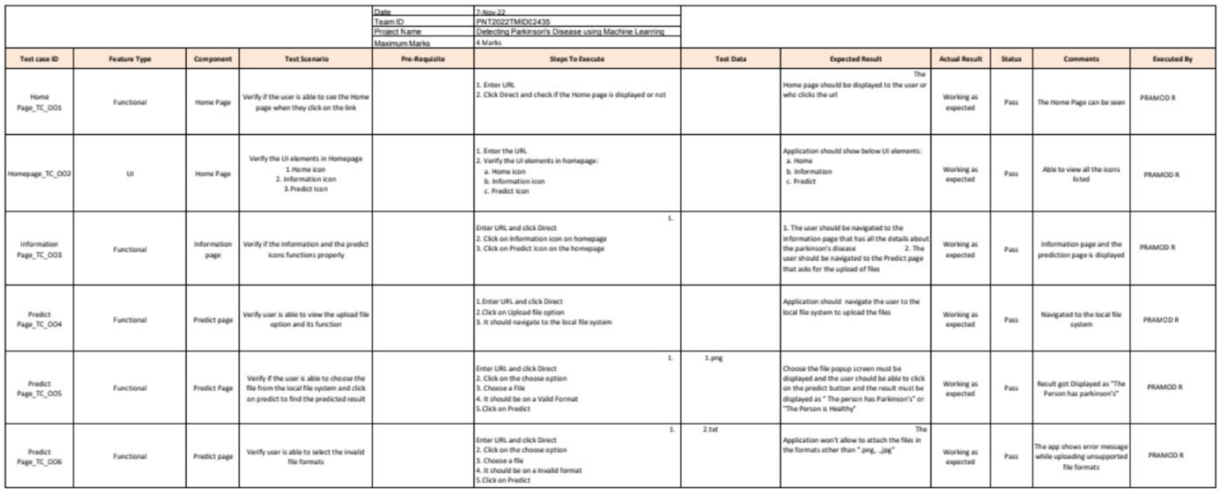
Approach: Alpha Testing and Beta Testing

• UAT Test scenarios & Testcases prepared based on business needs in both functional and non-functional aspects

• UAT Testcases can be set of existing testcases and maintained as a separate UAT Test suite.

• UAT Test scenarios & cases once designed should be reviewed by BA or Product Owner.

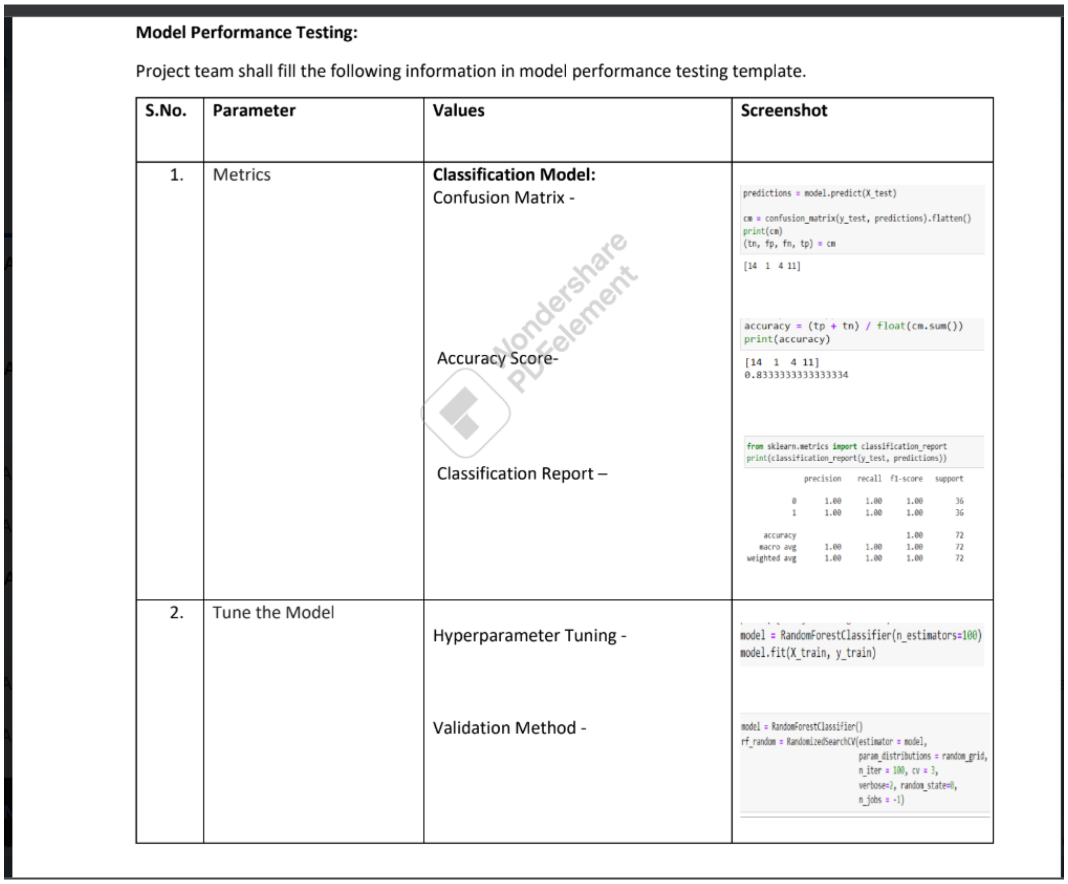
• UAT Testcases target is customer environment based in-terms of Test data and Servers.



				Date	7-Nov-22					
				Team ID	PNT2022TMID02435					
				Project Name	Detecting Parkinson's Disease using Machine Learning					
				Maximum Marks	4 Marks					
Test case ID	**Feature Type**	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**	**Status**	**Comments**	**Executed By**
Home Page_TC_OO1	Functional	Home Page	Verify if the user is able to see the Home page when they click on the link		1. Enter URL 2. Click Direct and check if the Home page is displayed or not		The Home page should be displayed to the user or who clicks the url	Working as expected	Pass	The Home Page can be seen	PRAMOD R
Homepage_TC_OO2	UI	Home Page	Verify the UI elements in Homepage 1.Home icon 2. Information icon 3.Predict Icon		1. Enter the URL 2. Verify the UI elements in homepage: a. Home icon b. Information icon c. Predict Icon		Application should show below UI elements: a. Home b. Information c. Predict	Working as expected	Pass	Able to view all the icons listed	PRAMOD R
Information Page_TC_OO3	Functional	Information page	Verify if the Information and the predict icons functions properly		1. Enter URL and click Direct 2. Click on Information icon on homepage 3. Click on Predict Icon on the homepage		1. The user should be navigated to the Information page that has all the details about the parkinson's disease 2. The user should be navigated to the Predict page that asks for the upload of files	Working as expected	Pass	Information page and the prediction page is displayed	PRAMOD R
Predict Page_TC_OO4	Functional	Predict page	Verify user is able to view the upload file option and its function		1.Enter URL and click Direct 2.Click on Upload file option 3. It should navigate to the local file system		Application should navigate the user to the local file system to upload the files	Working as expected	Pass	Navigated to the local file system	PRAMOD R
Predict Page_TC_OO5	Functional	Predict Page	Verify if the user is able to choose the file from the local file system and click on predict to find the predicted result		1. Enter URL and click Direct 2. Click on the choose option 3. Choose a File 4. It should be on a Valid Format 5.Click on Predict	1.png	Choose the file popup screen must be displayed and the user should be able to click on the predict button and the result must be displayed as " The person has Parkinson's" or "The Person is Healthy"	Working as expected	Pass	Result got Displayed as "The Person has parkinson's"	PRAMOD R
Predict Page_TC_OO6	Functional	Predict page	Verify user is able to select the invalid file formats		1. Enter URL and click Direct 2. Click on the choose option 3. Choose a file 4. It should be on a invalid format 5.Click on Predict	2.txt	The Application won't allow to attach the files in the formats other than ".png, .jpg"	Working as expected	Pass	The app shows error message while uploading unsupported file formats	PRAMOD R

9. RESULTS

9.1 Performance Metrics



Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot

1.	Metrics	**Classification Model:**
		Confusion Matrix -
		
		Accuracy Score-
		
		Classification Report –
			`predictions = model.predict(X_test)`
			
			`cm = confusion_matrix(y_test, predictions).flatten()`
			`print(cm)`
			`(tn, fp, fn, tp) = cm`
			
			`[14  1  4 11]`
			
			`accuracy = (tp + tn) / float(cm.sum())`
			`print(accuracy)`
			
			`[14  1  4 11]`
			`0.8333333333333334`
			
			`from sklearn.metrics import classification_report`
			`print(classification_report(y_test, predictions))`
			
			`              precision    recall  f1-score   support`
			`           0       1.00      1.00      1.00        36`
			`           1       1.00      1.00      1.00        36`
			
			`    accuracy                           1.00        72`
			`   macro avg       1.00      1.00      1.00        72`
			`weighted avg       1.00      1.00      1.00        72`

2.	Tune the Model	Hyperparameter Tuning -
		
		Validation Method -
			`model = RandomForestClassifier(n_estimators=100)`
			`model.fit(X_train, y_train)`
			
			`model = RandomForestClassifier()`
			`rf_random = RandomizedSearchCV(estimator = model,`
			`                               param_distributions = random_grid,`
			`                               n_iter = 100, cv = 3,`
			`                               verbose=2, random_state=0,`
			`                               n_jobs = -1)`

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

• Friendly UI

• Accuracy level is more than 80%

• More efficient as it involves the techniques in Machine Learning

• Accurately detecting Parkinson's disease at an early stage is certainly indispensable for slowing down its progress and providing patients the possibility of accessing to disease-modifying therapy.

• Yields high potential for more systematic clinical decision-making system.

DISADVANTAGES

• The main limitation of random forest is that a large number of trees can make the algorithm too slow and ineffective for real-time predictions.

• In general, these algorithms are very fast to train, but quite slow to create predictions once they are trained.

• Random forest is less interpretable than a single decision tree.

11. CONCLUSION

We all know the fact that two of the most common Parkinson's symptoms include tremors and muscle rigidity which directly impact the visual appearance of a hand-drawn spiral and wave. The variation in visual appearance will enable us to train a machine-learning algorithm to automatically detect Parkinson's disease. we have utilized the random forest algorithm. We also have utilized the Histogram of Oriented Gradients image descriptor to quantify each of the input images. After extracting features from the input images, we trained a Random Forest classifier with 100 total decision trees in the forest, obtaining more than 83.33% accuracy on the spiral dataset that we provided. The Random Forest trained on the spiral dataset obtained 73.33% sensitivity, meaning that the model was capable of predicting a true positive (i.e.,“Yes, the patient has Parkinson’s") nearly 73.33% of the time. Hence this model will be very useful for the early detection of Parkinson's disease and it can be easily used as it has a user-friendly interface. Though Parkinson’s cannot be cured, early detection along with proper medication can significantly improve symptoms and quality of life.

In conclusion, the model that we built using a Random Forest classifier gives us a good evaluation and accuracy score. So, it can be trusted for the early detection of Parkinson's disease and it can be utilized by people very easily.

12.FUTURE SCOPE

In the future we will use different types of attributes for the classification of patients and also identify the different stages of Parkinson's disease.Enrichment of machine learning algorithms with additional sensor data (i.e.,accelerometry, gyrometry ,spirography) may enhance the precision of thealgorithm. More recent versions of iMotor have incorporated some of these capabilities, and additional studies are currently ongoing

13.APPENDIX

Source Code

```
from flask import Flask, render_template, request
from werkzeug.utils import secure_filename
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
from skimage import feature
from imutils import build_montages
from imutils import paths
import numpy as np
import cv2
import os
import pickle
import PIL
#import tensorflow as tf

#from tensorflow import keras
#from tensorflow.keras import layers
#from tensorflow.keras.models import Sequential
trainingpath=r"C:\Users\NITIYASRI\Downloads\wave\wave\training"
testingpath=r"C:\Users\NITIYASRI\Downloads\wave\wave\testing"
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


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

    for imagePath in imagePaths:
        label = imagePath.split(os.path.sep)[-2]

        image = cv2.imread(imagePath)
        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))
```

```
print("[INFO] loading data...")
(X_train, y_train) = load_split(trainingpath)
(X_test, y_test) = load_split(testingpath)
le = LabelEncoder()
y_train = le.fit_transform(y_train)
y_test = le.transform(y_test)
print(X_train.shape,y_train.shape)
print("[INFO] training model")
model = RandomForestClassifier(n_estimators=100)
model.fit(X_train, y_train)
testingpath=list(paths.list_images(testingpath))
idxs=np.arange(0,len(testingpath))
idxs=np.random.choice(idxs,size=(25,),replace=False)
images=[]
for i in idxs:
    image = cv2.imread(testingpath[i])
    output = image.copy()

    # load the input image,convert 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
feature using last trained random forest
    features = quantify_image(image)
    preds = model.predict([features])
    label = le.inverse_transform(preds)[0]
    # the set of output images
    if label == "healthy":
        color = (0, 255, 0)
    else:
        color = (0, 0, 255)

    cv2.putText(output, label, (3, 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
color, 2)
    images.append(output)

# creating a montage
montage = build_montages(images, (128, 128), (5, 5))[0]
cv2.imshow("Output", montage)
predictions = model.predict(X_test)

cm = confusion_matrix(y_test, predictions).flatten()
print(cm)
(tn, fp, fn, tp) = cm
accuracy = (tp + tn) / float(cm.sum())
print(accuracy)
pickle.dump(model,open('parkinson.pkl','wb'))
import pickle
import cv2
```

```
from skimage import feature
from flask import Flask, request, render_template
import os.path
app = Flask(__name__)


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


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


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


@app.route('/uploader', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        f = request.files['file']
        basepath = os.path.dirname(__file__)
        filepath = os.path.join(basepath, "uploads",
secure_filename(f.filename))
        f.save(filepath)
        print(" (INFO] loading model...")
        model = pickle.loads(open('parkinson.pkl', "rb").read())
        image = cv2.imread(filepath)

        output = image.copy
        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 = 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)

        return result
    return None
    return 'file uploaded successfully'


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

GitHub & Project Demo Link
https://drive.google.com/drive/folders/1kmAH4ebtVOECwrQut05nasdeUYcwdS-

r?usp=share_link