# DETECTION OF PARKINSON’S DISEASE USING CLASSIFICATION ALGORITHMS

### U18PRIT8P2 - PHASE II REPORT

#### *Submitted by*

**J. VIDYADHAR (U19IT021)**

**L. TEJA REDDY (U19IT032)**

**V. LOHITHA (U19IT033)**

#### *Under the Guidance of*

#### DR. V. KHANAA

#### Dean IT Department

#### *in partial fulfillment for the award of the degree of*

## BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY



## DEPARTMENT OF INFORMATION TECHNOLOGY

## SCHOOL OF COMPUTING

## BHARATH INSTITUTE OF SCIENCE AND TECHNOLOGY

**BHARATH INSTITUTE OF HIGHER EDUCATION AND RESEARCH,**

**CHENNAI - 600073**

### APRIL 2023

# 

# BHARATH INSTITUTE OF SCIENCE AND TECHNOLOGY

# CHENNAI

# DEPARTMENT OF INFORMATION TECHNOLOGY

**U18PRIT8P2 - PHASE II REPORT**

## BONAFIDE CERTIFICATE

##### Certified that this Report titled “**DETECTION OF PARKINSON’S DISEASE USING CLASSIFICATION ALGORITHMS**” is the bonafide work of **J. VIDYADHAR (U19IT021), L.TEJA REDDY(U19IT032), V. LOHITHA(U19IT033)** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Dr .K .Ramesh kumar

Professor and Head

DEPARTMENT OF INFORMATION TECHNOLOGY

BHARATH INSTITUTE OF HIGHER EDUCATION AND RESEARCH

CHENNAI – 600 073

Project Phase- II Viva Voce

Examination held on

INTERNAL EXAMINER

DR. V. Khanaa

Dean IT Department

DEPARTMENT OF INFORMATION TECHNOLOGY

BHARATH INSTITUTE OF HIGHER EDUCATION AND RESEARCH,

CHENNAI – 600 073

EXTERNAL EXAMINER

**DECLARATION**

We declare that this project report titled **Detection of Parkinson’s Disease using Classification Algorithms** submitted in partial fulfillment of the degree of **B. Tech in (Information Technology)** is a record of original work carried out by us under the supervision of **Dr. V. Khanaa**, and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.

J. Vidyadhar (U19IT021)

L.Teja Reddy (U19IT032)

V. Lohitha (U19IT033)

Chennai

/ / 2023

**ACKNOWLEDGMENT**

First, I express my sincere thankfulness to the almighty, for bestowing his blessings throughout this project work. I express my gratitude to my parents for their care, support, prayers and love.

I would like to express my deep gratitude to our Dr. K. Vijaya Bhaskar Raju, Vice Chancellor Dr. M. Sundararajan, Pro Vice-Chancellor (Academics), Dr. R. Hari Prakash, Additional Registrar Dr. R. M. Suresh, Pro Vice-Chancellor with Additional In charge Controller of Examinations, Dr. J. Hameed Hussain Dean Engineering who are responsible for moulding my thoughts in completing my Project.

I am thankful to the Dr. V. Khanna Dean-IT and Head of the Department Dr.K.Ramesh Kumar and all staff members of the Department of INFORMATION TECHNOLOGY, BIHER, for their encouragement and guidance throughout the course of this project work.

I wish to express my sincere profound gratitude to my esteemed and endowed Project guide Dr.V.Khanaa Dean of IT Department for his inspiring guidance, healthy criticism, valuable suggestions and constant encouragement throughout the period of project.

I sincerely wish to express my thanks to the staff members of Library, BIHER for their valuable during this project work. I would like to thank the authors of various journals and books whose works and results are used in the project.

**J.VIDYADHAR(U19IT021)**

**L TEJA REDDY(U19IT032)**

**V LOHITHA(U19IT033)**

**ABSTRACT**

Parkinson's disease (PD) is progressive neurodegenerative disorder that affects a lot of people significantly affecting their quality of life. It mostly affects the motor functions of humans.

The symptoms of Parkinson's disease will occur slowly, the symptoms include shaking, rigidity, slowness of movement and difficulty with walking, Thinking and behaviour change, Depression and anxiety are also common.

If a person has Parkinson disease or not is predicted by comparing different machine learning algorithms such as Random Forest, K-Nearest Neighbours (KNN) and Extreme Gradient Boosting (XG Boost).

In this proposed work, the accuracy score of the algorithms is compared and the best algorithm is chosen that gives the highest accuracy rate in predicting the Parkinson disease.

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **DESCRIPTION** | **PAGE NO** |
| 1 | Introduction | 1 |
| 1.1 | Demerits of Existing System | 2 |
| 1.2 | Merits of Proposed System | 3 |
| 1.3 | Software Requirements | 3 |
| 1.4 | Dataset Description | 3 |
| 1.5 | Motivation | 4 |
| 2 | Objective | 5 |
| 3 | Methodology | 7 |
| 3.1 | Data Analysis and Preparation | 7 |
| 3.1.1 | Exploratory Analysis | 7 |
| 3.1.2 | Correlation Analysis | 8 |
| 3.1.3 | Data Pre-processing | 9 |
| 3.2 | Building Model | 11 |
| 3.2.1 | Extreme Gradient Boosting | 12 |
| 3.2.2 | Random Forest | 13 |
| 3.2.3 | K Nearest Neighbour | 14 |
| 3.3 | Evaluation Metrics | 15 |
| 3.3.1 | Confusion Matrix | 15 |
| 3.3.2 | Recall | 16 |
| 3.3.3 | F1 Score | 16 |
| 4 | Results | 17 |
| 5 | Conclusion and Future Works | 26 |
| 5.1 | Conclusion | 26 |
| 5.2 | Future Works | 26 |
| 6 | Reference | 27 |
| 7 | Appendix (Source code) | 28 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **PageNo.** |
| 1.1 | Stages of Parkinson Disease | 2 |
| 1.2 | Depiction of SVM (Existing Model) | 3 |
| 2.1 | Architecture Diagram | 6 |
| 3.1 | Exploratory Analysis | 7 |
| 3.2 | Univariant Analysis | 7 |
| 3.3 | Correlation Matrix using Heatmap | 9 |
| 3.4 | Outliers using Boxplot | 11 |
| 3.5 | Depiction of working for XG Boost Algorithm | 12 |
| 3.6 | Depiction of working for Random Forest | 13 |
| 3.7 | Depiction of working for KNN | 15 |
| 3.8 | Depiction of Confusion matrix | 16 |
| 3.9 | Formulae of F1 score, precision and recall | 16 |
| 4.1,4.2 | Epochs/iterations for XG Boost algorithm | 17 |
| 4.3 | Confusion Matrix of XG Boost | 18 |
| 4.4 | Classification Report of XG Boost | 18 |
| 4.5 | Confusion Matrix of Random Forest | 19 |
| 4.6 | Classification Report of Random Forest | 19 |
| 4.7 | Confusion Matrix of KNN | 20 |
| 4.8 | Classification Report of KNN | 20 |
| 4.9 | Comparison Plot of Accuracies of Three Algorithms | 21 |
| 4.10 – 4.11 | Prediction | 21 |
| 4.12 | Main window | 22 |
| 4.13 | Accuracy Plot window | 22 |
| 4.14 | Accuracy window | 23 |
| 4.15 | Prediction | 23 |
| 4.16 | Result | 24 |
| 4.17 | Prediction | 24 |
| 4.18 | Result | 25 |

**LIST OF ABBREVIATION**

|  |  |
| --- | --- |
| SVM | Support Vector Machine |
| PCA | Principal Component Analysis |
| ECFS | Ensemble Classifier Feature Selection |
| FS | Feature Selection |
| XGBOOST | Extreme Gradient Boosting |
| RFC | Random Forest Classification |
| KNN | K Nearest Neighbour |
| TP | True Positive |
| TN | True Negative |
| FP | False Positive |
| FN | False Negative |

**CHAPTER – 1**

**INTODUCTION**

Parkinson's problem is a neurological disorder that causes shiver, unfaltering nature, and bother in strolling, changing, and assembling the body. Side effects of Parkinson's defilement traditionally happen consistently and deteriorate after some time. Individuals with Parkinson's defilement could experience issues strolling and talking as the difficulty makes. They may in this way have mental and mental changes, as well as rest issues, discouragement, memory issues, and utilization. Certain nerve cells (neurons) in the cerebrum for a really long time disconnected or kick the compartment in Parkinson's difficulty.

Basically the optional impacts are all accomplished by the passing of neurons in your cerebrum that produce dopamine, a substance courier. Dopamine deficiency causes distorted cerebrum improvement, which prompts heartbreaking flexibility and other Parkinson's defilement coincidental impacts. Subsequently, it's persuasive for have the decision to figure this difficulty.

Among the signs and side effects of Parkinson's defilement are:

* Shiver: A quake, or shaking, if all else fails, begins in one of your furthest points, most all things considered in your grasp or fingers. A pill-moving shiver happens when you rub your thumb and index finger to a great extent.
* Moved back progress: Parkinson's defilement moves back your movability after some time, making routine undertakings more extraordinary and repetitive. Precisely when you walk, your means could end up being more confined. Moving away from a seat can challenge. Precisely when you attempt to walk, your feet could drag.
* Unfaltering Muscles: Muscle heartiness can strike whenever and in any piece of your body. Solid muscles can be unusual and confine your degree of improvement.
* Position and concordance issues: because of Parkinson's defilement, your position could become stooped, and you could encounter balance issues.
* Loss of body Balance: It's conceivable that you'll acquire a couple of more enthusiastically encounters doing careless activities like flickering, grinning, or swinging your arms as you walk.
* Shuddering of Voice: You have the choice of talking cautiously, expedient, slurring, or stopping going before talking. Rather than the standard verbalizations, your discussion might be more repetitive.
* Bother recorded as a printed rendition: It could become testing to make, and your shaping could show up minimal as such.
* Resting Disorders: Sleep issues are run of the mill in individuals with Parkinson's difficulty, and can solidify stimulating endlessly all through the range of the evening, getting rolling early, or nodding off during the day.
* Expedient eye improvement rest direct knot, which consolidates proceeding with dreams, is in this way an open door. Arrangements could have the decision to assist you with your rest issues.



Figure 1.1 Parkinson Disease Stages

. **1.1 Demerits of Existing System:**

* The existing model is used for predicting only one symptom of Parkinson’s disease [Voice Loss].
* The performance of SVM is reduced if new features are added to testing set
* Performance is reduced when dataset has outliers or more noise

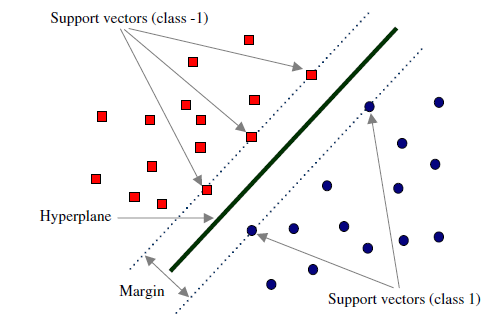


Figure 1.2: SVM Model

**1.2 Merits of Proposed System:**

* In the Existing System they are diagnosing the PD with only voice signals which is only one of the symptoms.
* In our proposed system Parkinson disease is predicted by using tremor frequency and voice frequency which are the symptoms of PD. Tremor frequency contains shivering of body parts which is the main symptom of PD.
* XG Boost, RF Algorithms are so similar and give good results for detection of PD compared to SVM and hence XG Boost algorithm mainly used for prediction of various disease.

**1.3 Software Requirements:**

|  |  |
| --- | --- |
| OS | Windows 11 |
| Language | Python |
| Editor | Jupyter Notebook |
| Frontend | Tkinter |

Table 1

**1.4 Dataset:**

* Created by Max Little of Oxford University.
* This dataset is produced using a degree of biomedical voice evaluations from 31 individuals, 23 with Parkinson's problem (PD).
* Each part in the table is a specific voice measure, and each line takes a gander at one of 480 voice accounts from these people.
* Status fragment is set to 0 for normal and 1 for PD.

**1.5 Motivation:**

1. The main motivation for doing this project is Parkinson disease has increased in world, and since it slowly starts to affect the body, if it is predicted earlier, it can be cured.
2. Parkinson can be predicted by main symptoms like voice shiverness and tremor and muscles.
3. Used both voice and tremor frequencies and tried to predict the Parkinson disease**.**

**Chapter 2**

**Objective**

Recognizing verification of Parkinson Disease should be reliant upon the clinical subtleties of the patients particularly frequencies of voice shuddering and shiver. The going with picture shows the side effects of Parkinson defilement

The most certain signs and consequences of Parkinson's problem happen when nerve cells in the basal ganglia, a district of the psyche that controls improvement, become debilitated besides pass on. Reliably, these nerve cells, or neurons, produce a fundamental frontal cortex substance known as dopamine. Unequivocally when the neurons crash and burn with a vengeance horrendously or become discouraged, they produce less dopamine, which causes the progression issues related with the contamination. Specialists truly acknowledge close to nothing about what drives the neurons come up short disagreeably

People with Parkinson's torture moreover lose the delicate spots that produce norepinephrine, the exceptionally organized messenger of the insightful basic construction, which controls different pieces of the body, for instance, beat and circulatory strain. The lack of norepinephrine could help with getting a handle on a piece of the non-improvement features of Parkinson's, for instance, usage, capricious heartbeat, diminished improvement of food through the gastrointestinal turn of events, and unforeseen drop in circulatory strain when a specific stands up from a sitting or lying position.

Different cerebrum relationship of people with Parkinson's nervousness contains Lewy bodies, extraordinary lots of the protein alpha-synuclein. Specialists are attempting to on a very basic level more conceivable comprehend the conventional and surprising pieces of alpha-synuclein and relationship to procured changes impact Parkinson's and Lewy body dementia.

Several occasions of Parkinson's defilement communicate an impression of being characteristic, and a few cases can be followed to unequivocal brand name changes. While genetic characteristics is endeavoured to expect a segment in Parkinson's, an essential piece of the time the issue doesn't seem to run in families. Various experts right now believe that Parkinson's results from a blend of brand name and standard factors, similar to responsiveness to harms.

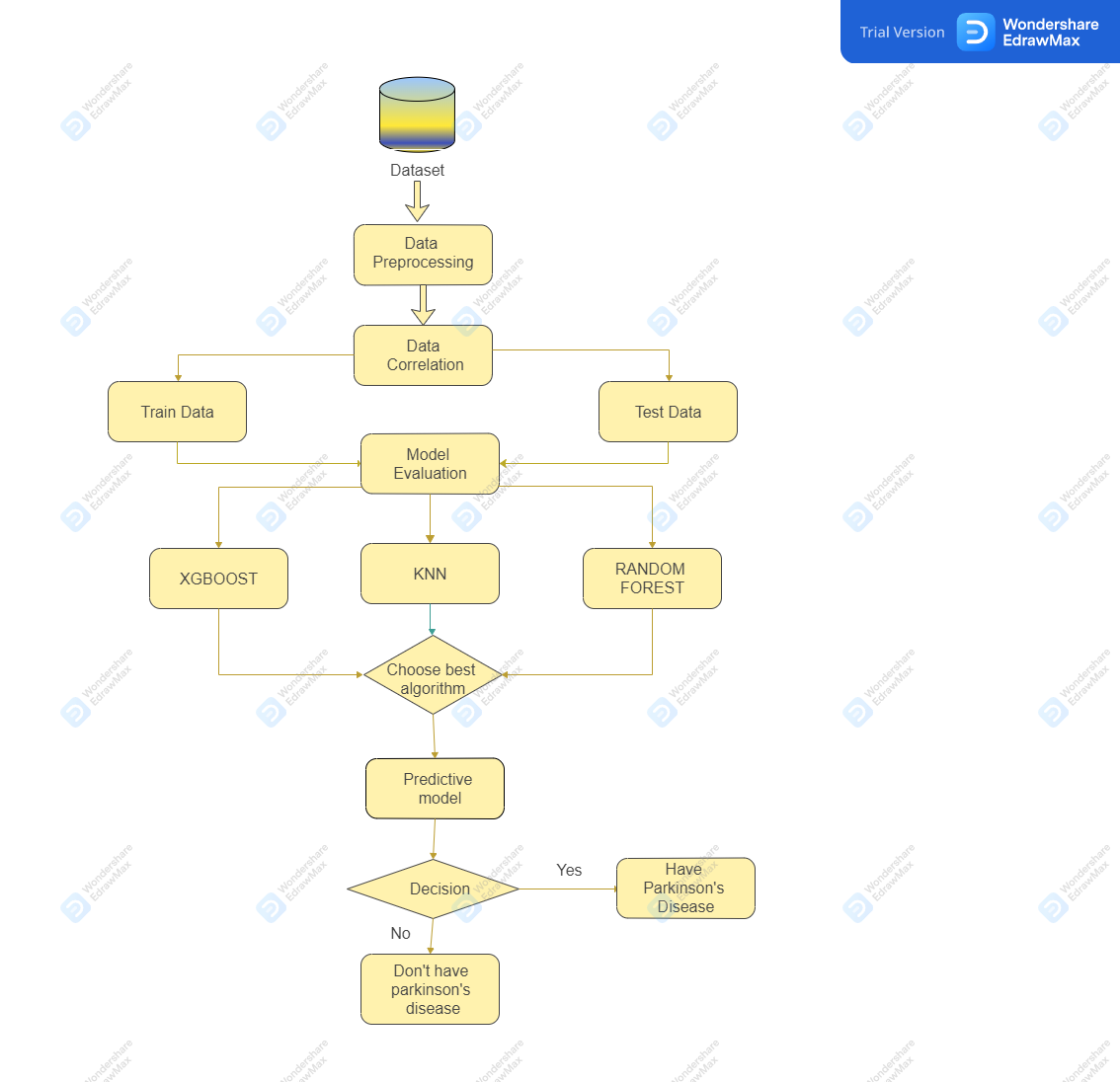


Fig 2.1: Architecture Diagram

**CHAPTER-3**

**METHODOLOGY**

**3.1 Data Analysis and Preparation:**

**3.1.1. Exploratory Data Analysis:**

* Produce a visual portrayal of the dataset's data, including the information, how much segments, and the metropolitan organizations utilized in the exploratory information evaluation.
* The information should be assessed before any superfluous sections are annihilated.
* Utilize the pieces of the test and train datasets to plan the AQI to the differentiating metropolitan organizations.
* Research the dataset and take any characteristics with a high relationship off the turn.

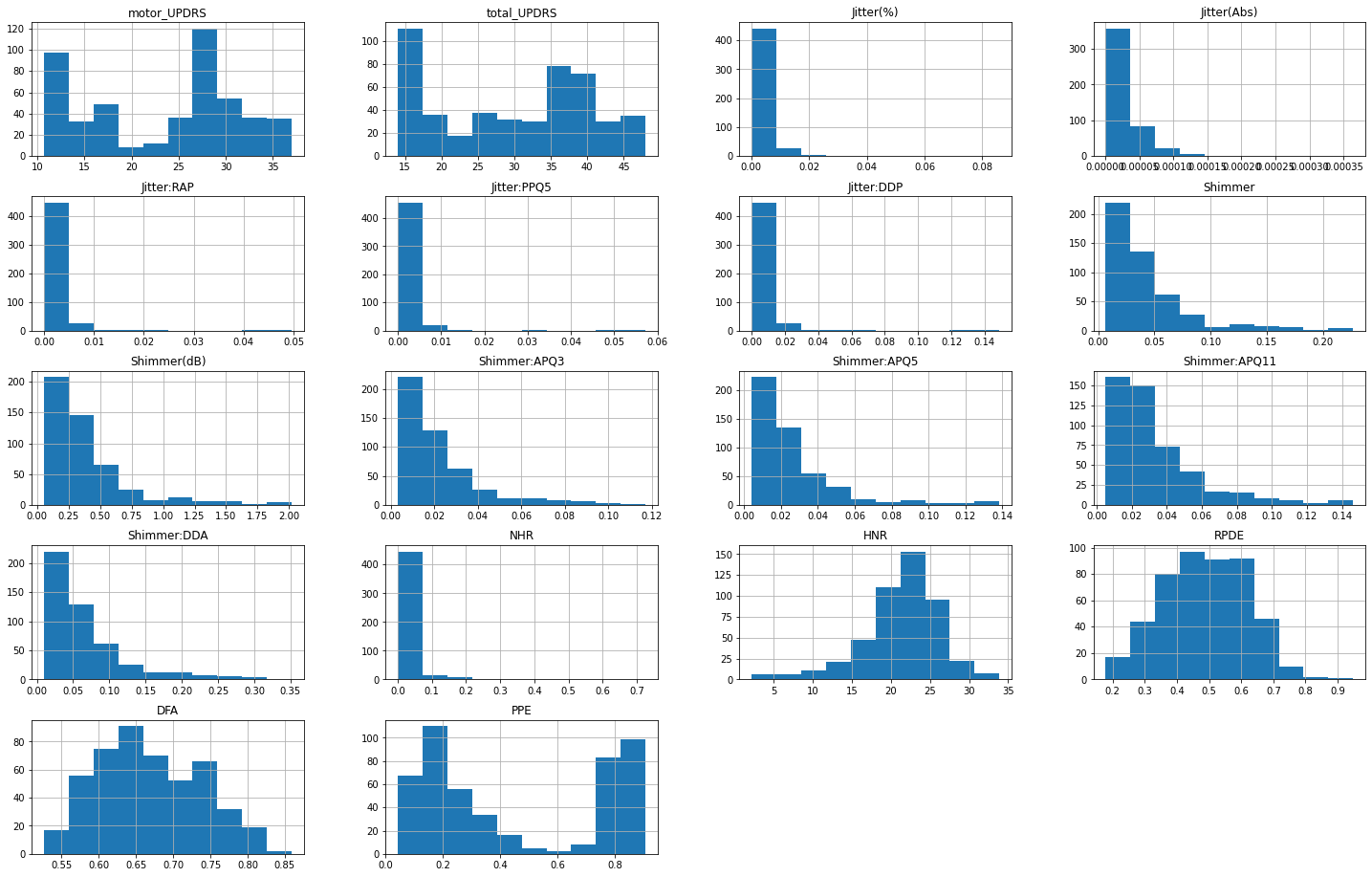
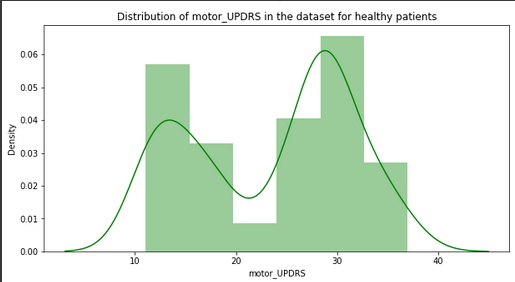
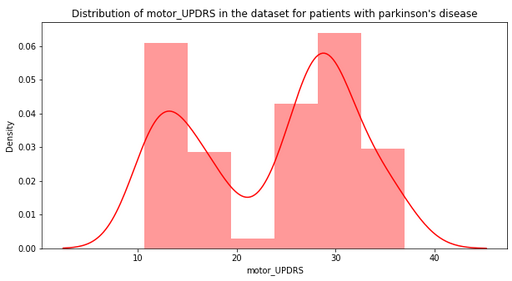
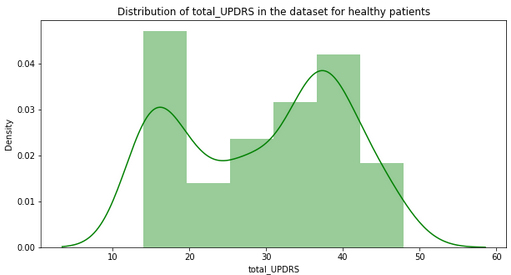
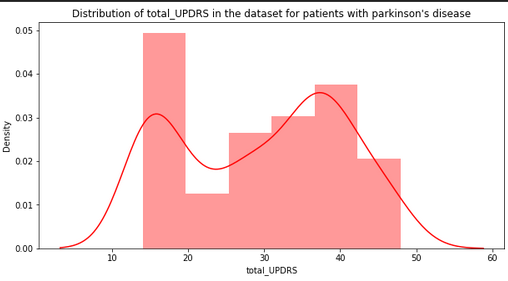
****

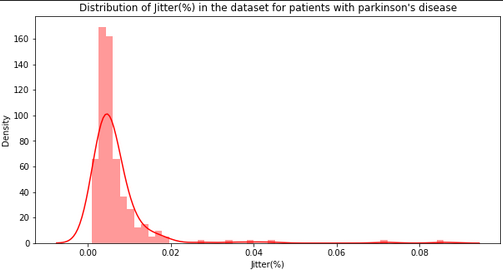
Fig 3.1 Histogram

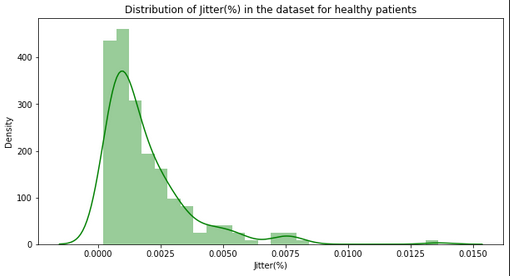
Figure 3.1 shows the relationship between the features of the dataset.

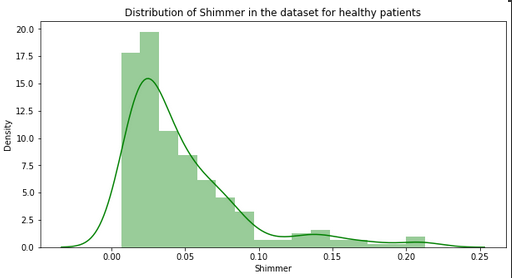
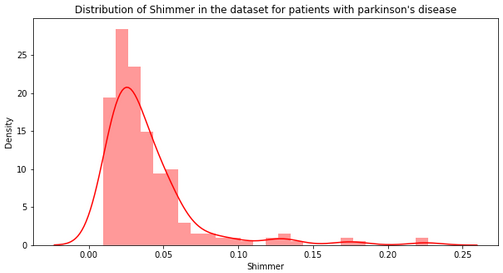
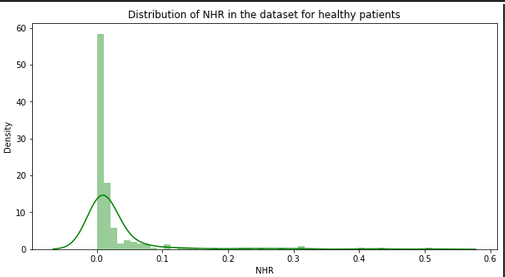
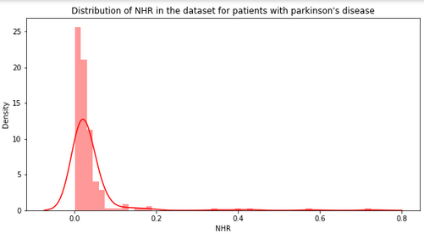
**Univariant Analysis:**











Figures 3.2 Features Analysis in healthy and Parkinson patients

It shows the relationship between the features of the dataset.

**3.1.2 Correlation Analysis:**

* The extraordinarily dependent variable is eliminated the turn using relationship examination, which furthermore recognizes dependent and independent elements.
* Exactly when two variables move in a comparative bearing, there is a positive relationship.
* In case one is raised, both will go up. For instance, running on a treadmill for a longer time period would consume more calories.
* Exactly when two variables have a negative relationship, that suggests they are moving the alternate way for instance, in case rising one measure causes a decreasing in the other, speeding up a vehicle will truncate the distance it takes to get to your area.
* In case there is no association between any two variables, there is none they are unimportant to one another.

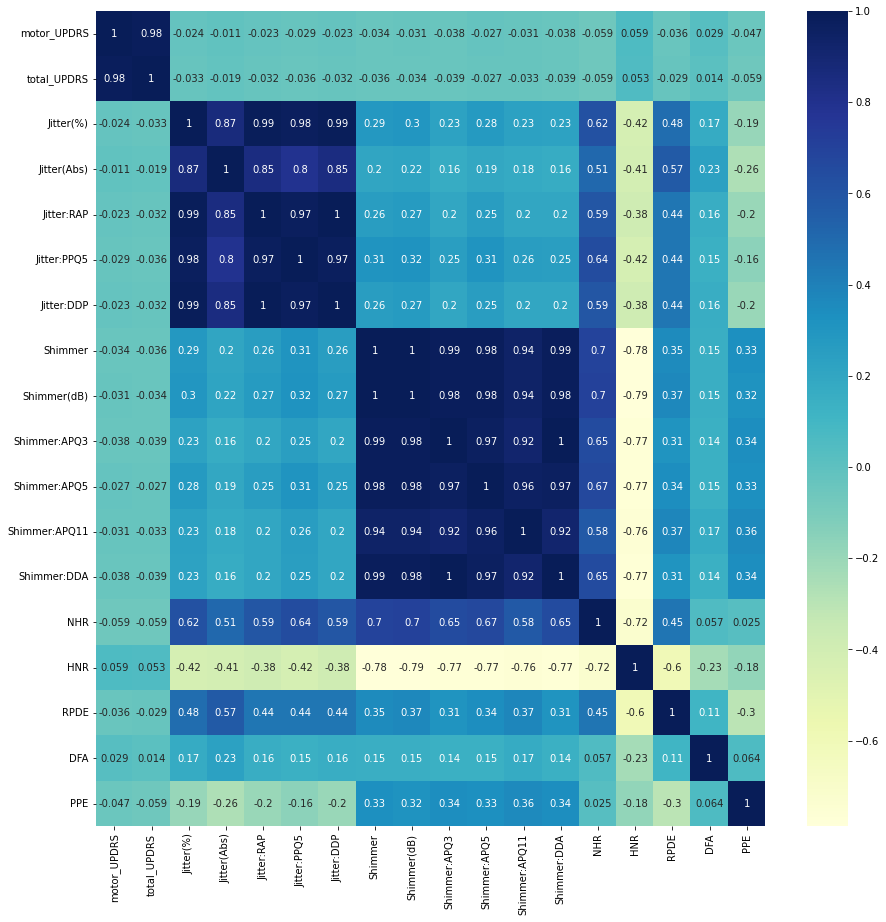


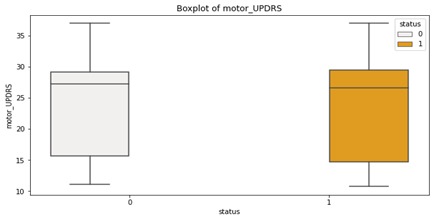
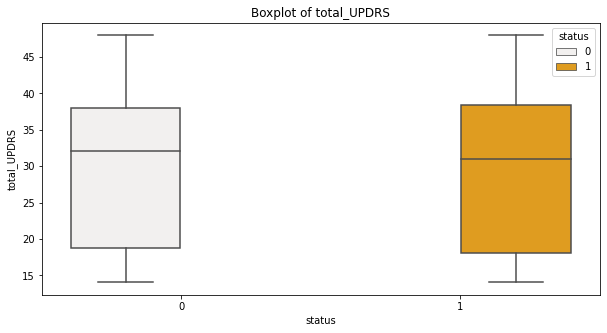
Fig 3.3 shows Correlation Matrix using Heat map

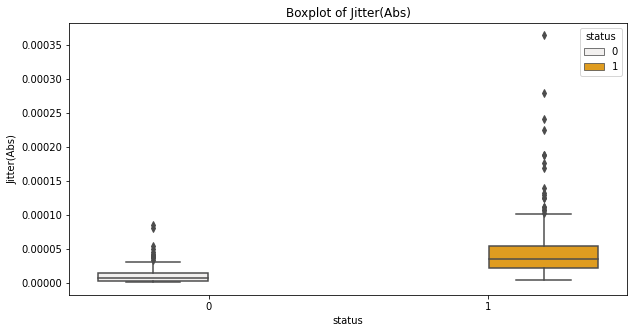
**3.1.3Data Pre-processing**:

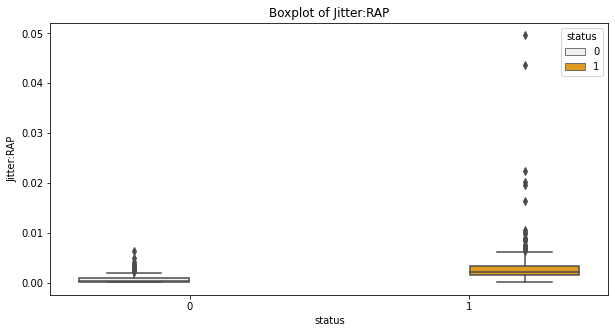
* A data mining approach called data pre-taking care of licenses you to facilitate and use deadbeat data.
* Data cleaning: Examples of data cleaning consolidate darkening boisterous data, abstaining from special cases, filling in the openings made by missing data, and settling data conflicts.
* Missing Values and Raw Data: There are a lot of missing characteristics. Along these lines, data become eccentric, making expecting inconvenient.
* To hold the guess back from being impacted, it is influential for manage the missing numbers fittingly, here boxplot is used to research the special cases and dispense with unfortunate data.

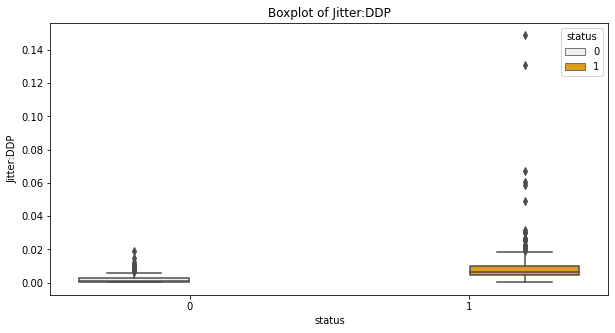
They include:

1. Ignore tuples: This methodology is applied when a sizable piece of the data in a tuple is missing.
2. Use the data's mean worth then again, if essential, extra basically indistinguishable characteristics, to fill in any openings.







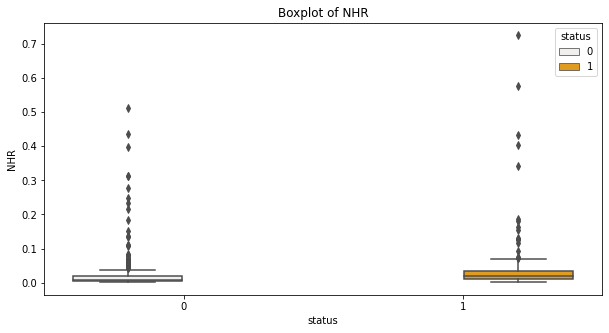












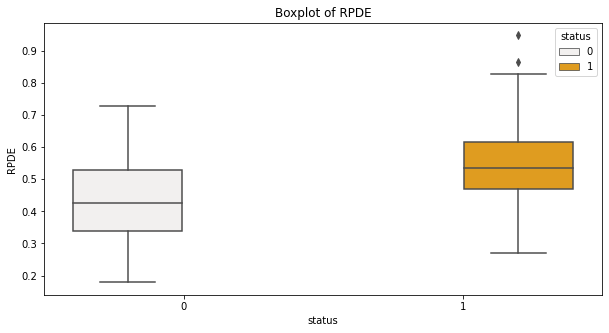




Fig 3.4 shows outliers of features using Boxplot

Here we are removing the outliers of the dataset.

**3.2Building Model:**

* After the parts are added to our dataset, then, at that point, model is created
* learning models with the eccentric backwoods' district classifier and XG Boost computations, to
* research the aftereffects of the two models concerning exactness and other hyper limits.
* The dataset is disengaged as the train and test data and the going with evaluations will be applied to them.

**3.2.1 Extreme Gradient Boost Algorithm:**

It is correspondingly a decision tree-based gathering AI computation which uses the inclination supporting arrangement. The figure issues which consolidate the unstructured data will in this way play out the colossal number of different plans. Right when the data is in the level of little to

medium then the decision tree computation choice is awesome. This usages inclination plunge evaluation inciting the decreasing in messes up and in moderate models. This computation is not unequivocally unclear from others in its momentous ways:

* Can be watched out for in wide collecting of purposes like break conviction, gathering, organizing.
* It has unprecedented genuinely straightforward nature in Windows, Linux, and OS X.
* It stays aware of in all the essential programming vernaculars
* It stays aware of in cloud coordination and other eco structures.

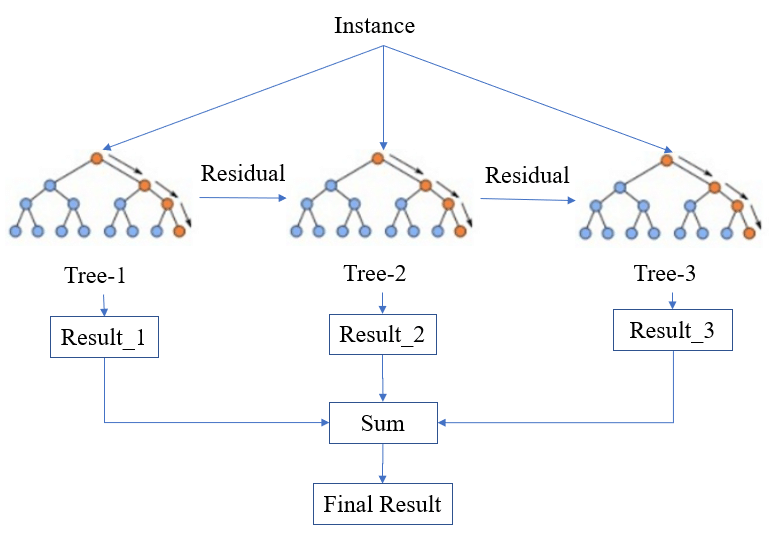


Figure 3.5: XG Boost Work flow

Benefits of Extreme Gradient Boosting:

• It is the mix of the inclination drop and the having an effect

• It keeps up with the different calamity limits.

• Works exceptionally with the affiliations.

Short comings of Extreme Gradient Boosting:

• It requires the tuning of the different hyper-cut-off points to watch out.

• It seeks after for tendency decay

• Its will likely diminish bias, not the variance.

**3.2.2 Random Forest Classifier:**

It is outfit learning reasoning and a coordinated learning evaluation which is used

for both the social gathering and additionally the apostatize, which is meta-assessor which fits the number of decisions trees and uses averaging to chip away at the insightful exactness and controls overfitting (mind blowing execution on the game-plan data, disagreeable hypothesis to other data) issue occurred. Different decision trees are being passed and solidified on to get more exactness and strength of check.

It has same hyper limits as a decision tree as well as holding classifier. It other than

adds a haphazardness to the made model, while encouraging the trees. The more precision then, the greater advancement in how much trees, picking the most preposterous number of parts to be cemented at each split of a center point. It has the restriction of managing the missing ascribes and stay aware of the accuracy for the tremendous degree of data.



Figure 3.6 Random Forest Algorithm Working

Let us see some Benefits and Short comes of RF Algorithm

Benefits of Random Forest:

* Defeats the issue of over fitting
* Gigantic degree of information things than the choice tree handles
* Less qualification wandered from the choice tree
* Especially adaptable
* Scaling of information isn't needed and remains mindful of amazing exactness with no scaling.

Inadequacies of Random Forest:

* More computational assets are required
* Improvement is essentially more tirelessly and time use is more.
* Complexity increases with respect to the problem.

**3.2.3: K Nearest Neighbours:**

KNN: K Nearest Neighbour is one of the principal assessments in AI. Man-made understanding models utilize a great deal of information values to foresee yield values. KNN is one of the most un-complex kinds of AI assessments taking everything into account for get-together. It sorts out the data of interest on how its neighbour is aggregated.

KNN collects the new information sparkles thinking about the similarity level of the past put away data of interest. For instance, on the off chance that have a dataset of tomatoes and bananas. KNN will store practically identical assessments like shape and mix. Unequivocally when another article comes it will really investigate its equivalence with the arrangement (red or yellow) and shape.

K worth in KNN pays special attention to how much the closest neighbours which have used to sort out new data of interest.

Benefits of KNN

* K-NN is truly normal and fundamental
* K-NN has no assumptions and distance between data points decides the position of the new data point in the respective group.
* Can be used both for Classification and Regression
* Very easy to complete for multi-class issue

Short comes of KNN

* K-NN needs homogeneous components
* Optimal number of neighbours
* Outliers affect the function.
* Missing Value treatment.



Figure 3.7 KNN Algorithm work flow

**3.3 EVALUATION METRICS:**

These estimations are used to measure the idea of the verifiable or AI model.

There are different kinds of estimations to test a model. Some of them are accuracy,

confusion framework, f1-score. These are made as Classification report.

**3.3.1 Confusion Matrix:**

It is ordinarily used in depicting the presentation of any gathering model on a lot of the test data that has given by the issue, for which the certifiable characteristics are known. It for the most part summarizes the presentations the issue and alone can be misdirecting when you have a conflicting number of the discernments in each class or more classes in dataset. There are relatively few terms that are to be portrayed like:

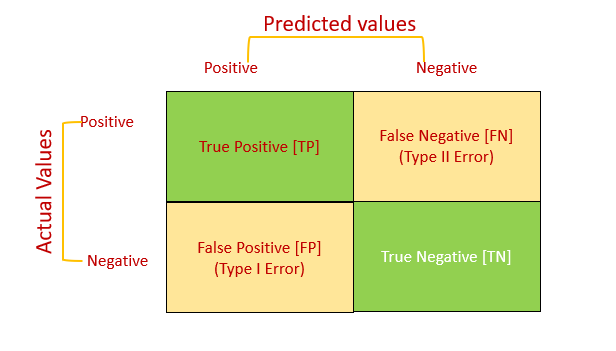


Figure 3.8 Confusion Matrix

**3.3.2 Recall:**

The review is the degree of our model exactly seeing True Positives. In this way, for each of the patients who genuinely have coronary disorder, study lets us know the amount of that unequivocally perceived as having a coronary illness. Numerically:

**3.3.3 F1-Score:**

It is degree of the F score or F measure with an anticipated recipe, which still hanging out there from the precision and review, which can be considered from the disarray framework. It shows the strength between the precision and the review. For the most part, it is the consonant mean of the model's precision and review. Obviously, the higher F1 score, the better the model.

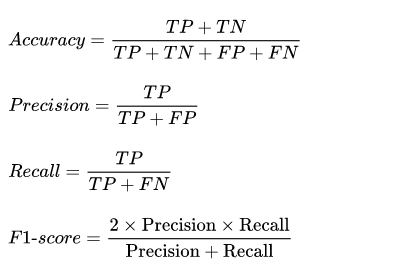


Figure 3.9 Evaluation Metrics

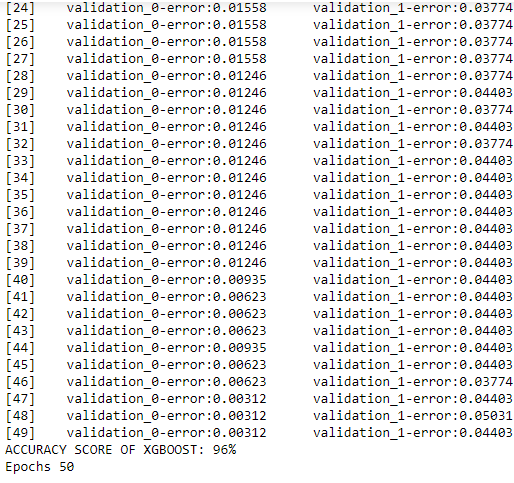
**Chapter 4**

**Results**

**XG Boost:**

**Validation**

1. Accuracy Achieved: 96%
2. Mean Accuracy: 96%
3. Standard deviation percentage: 3.15%



Figures 4.1, 4.2 Shows the Epoch Validations and Accuracy Percentage for XG Boost Algorithm

**Confusion Matrix:**

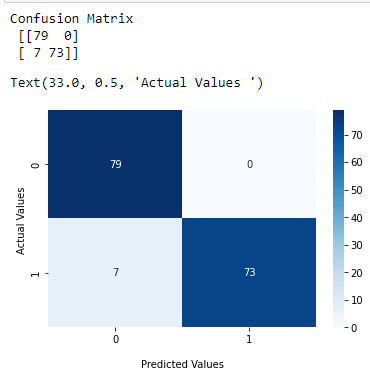


Figure 4.3 Shows Confusion matrix for XG Boost

**Classification Report:**

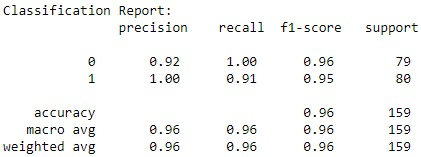


Figure 4.4 Shows the Classification report of XG Boost

**Random Forest:**

**Validation:**

* Accuracy Score of Random Forest: 86%
* Mean Accuracy: 91%
* Standard deviation classification Accuracy: 4.09%

**Confusion Matrix:**

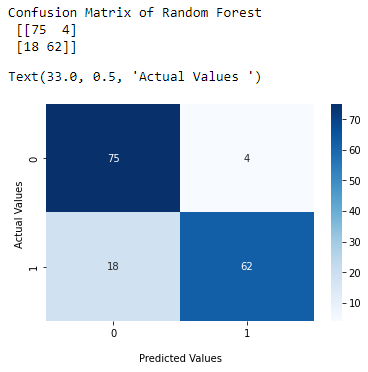


Figure 4.5 Shows Confusion Matrix for Random Forest

**Classification Report:**

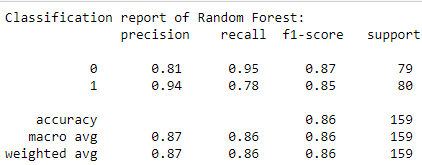


Figure 4.6 Shows Classification Report for Random Forest.

**K Nearest Neighbours:**

**Validation:**

* Accuracy: 86%
* Mean Accuracy: 87%
* Standard Deviation: 3.27%

**Confusion Matrix:**

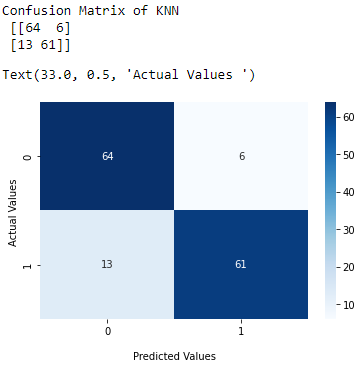


Figure 4.7 Shows Confusion Matrix for K Nearest Neighbours

**Classification Report:**

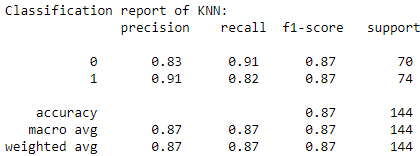


Figure 4.8 Shows Classification Report for K Nearest Neighbours

**Accuracy Plot:**

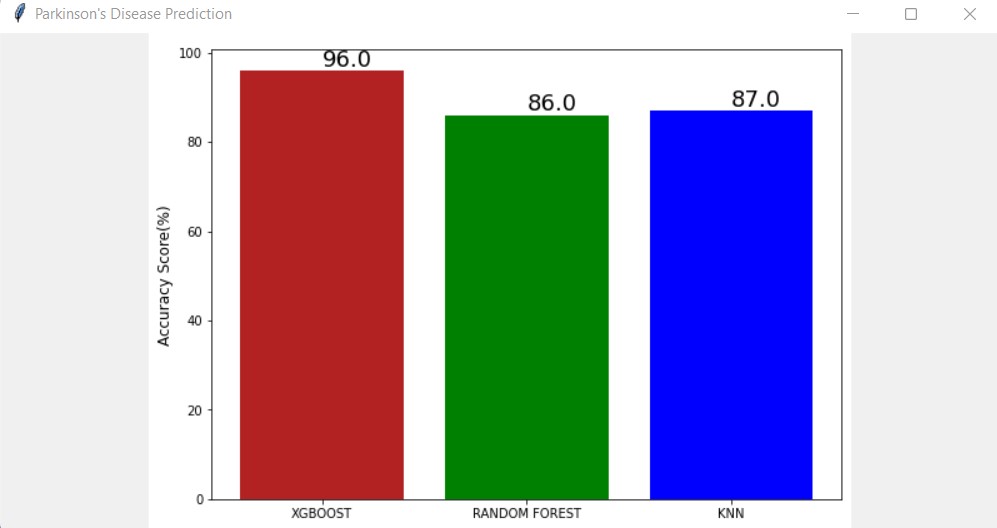
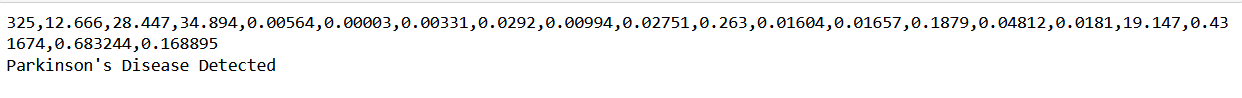


Fig 4.9 Accuracy Plot

Prediction:



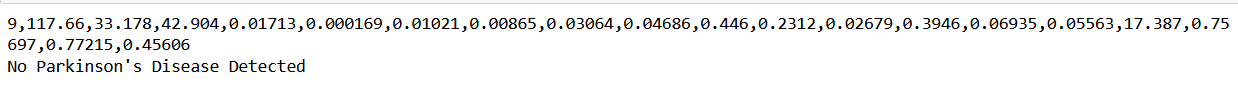


Fig 4.10,4.11 shows prediction image

**OUTPUT :**

**Frontend:**

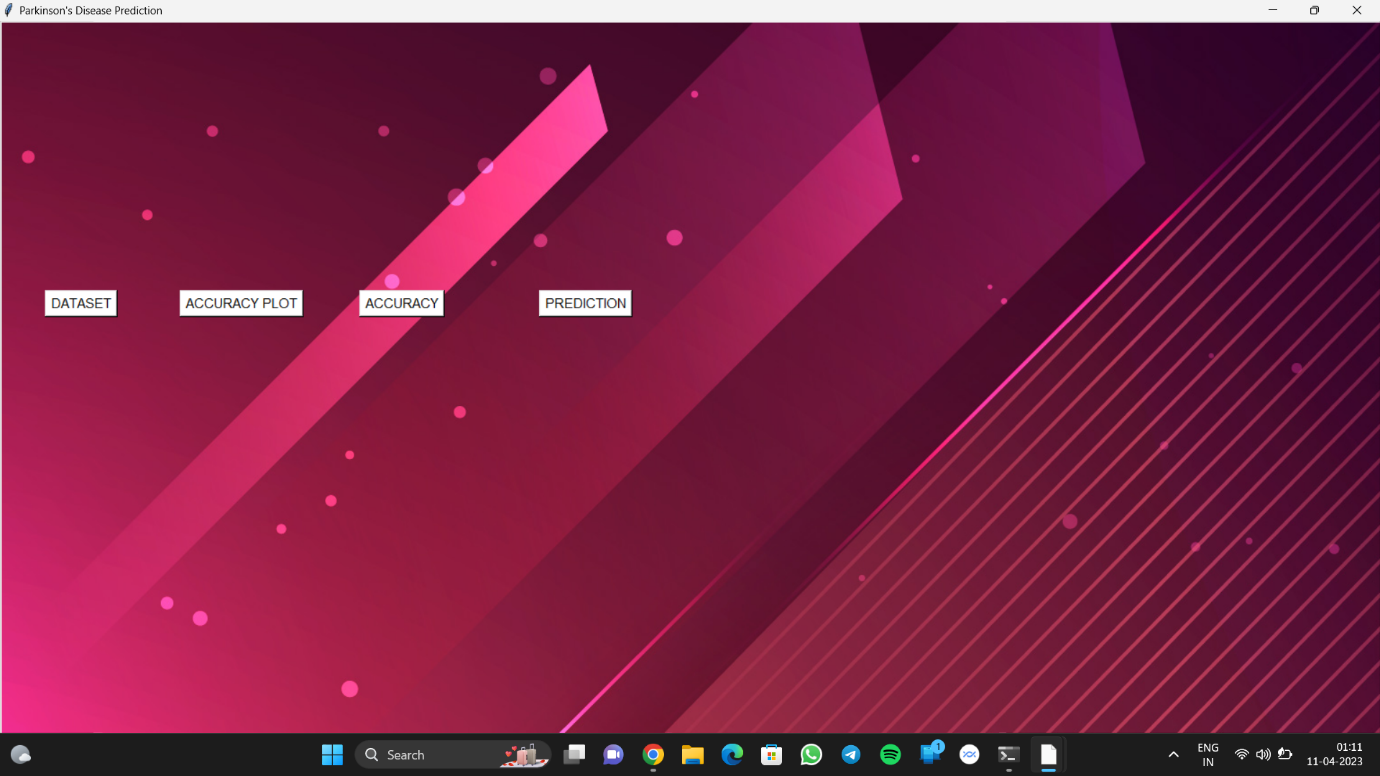
****

Fig 4.12 – Main window

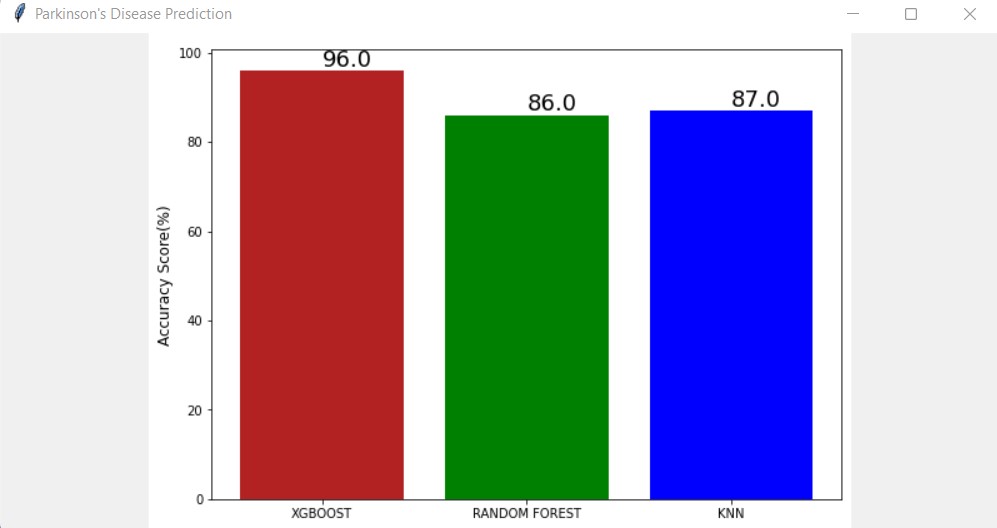


Fig 4.13 Accuracy plot

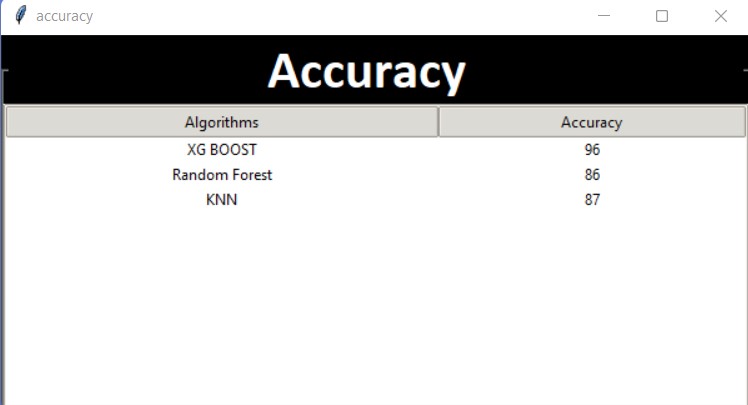


Fig 4.14 Accuracy window

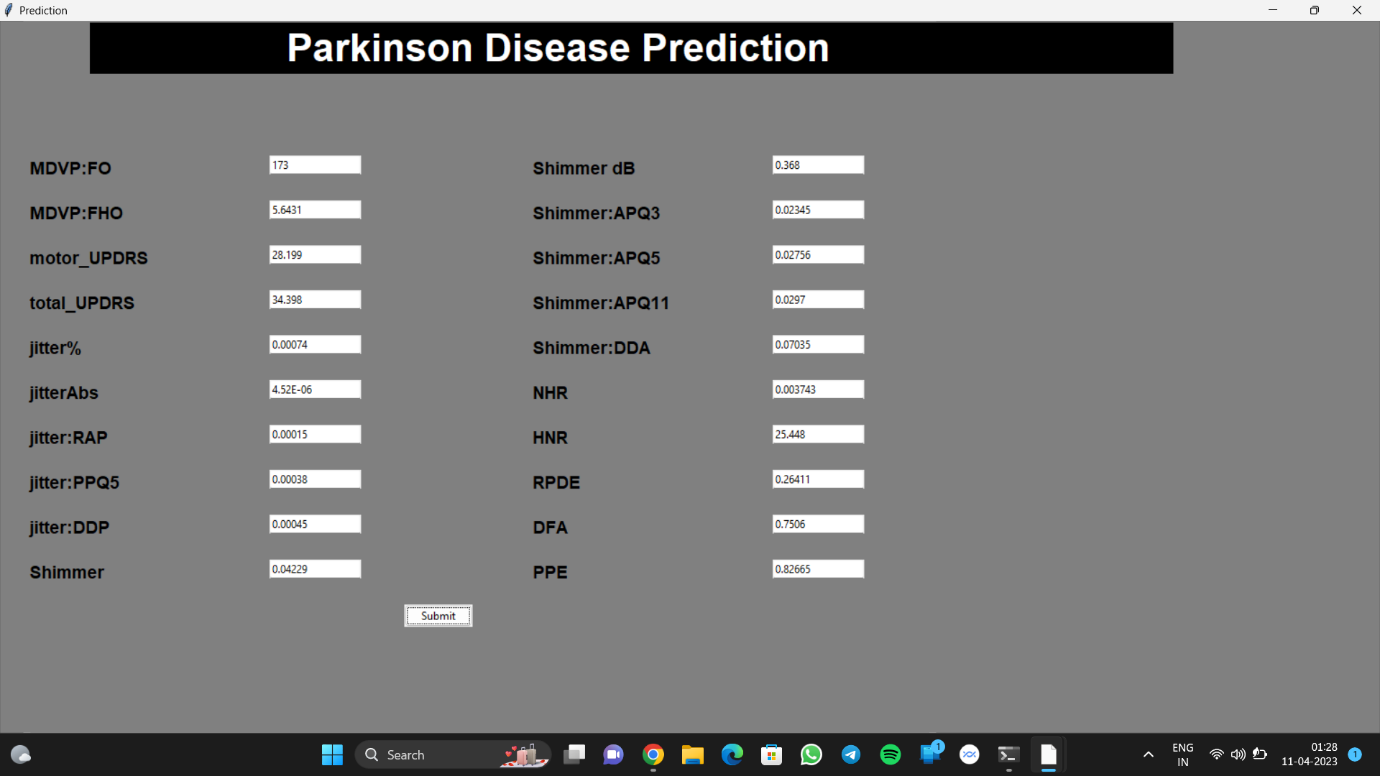


Fig 4.15 Prediction window

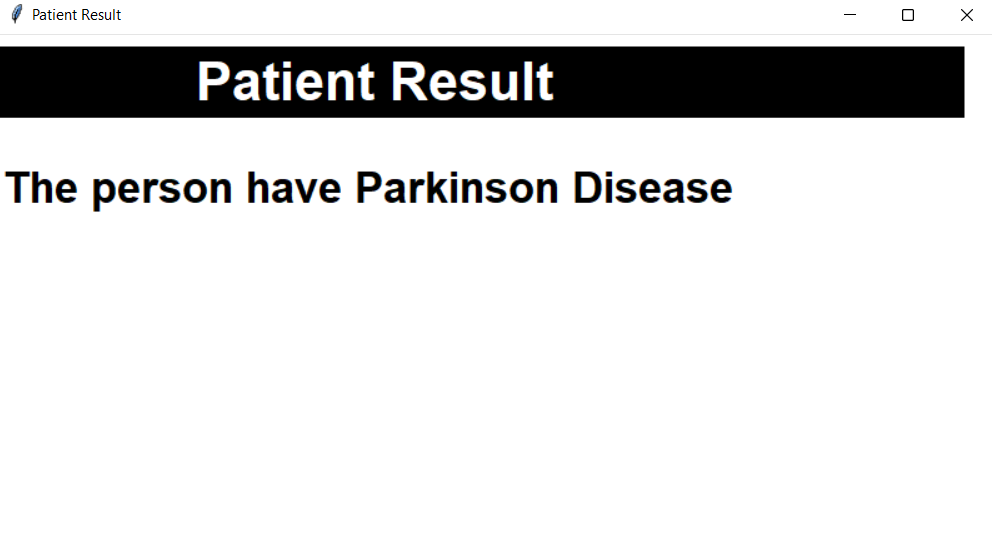


Fig 4.16 Result

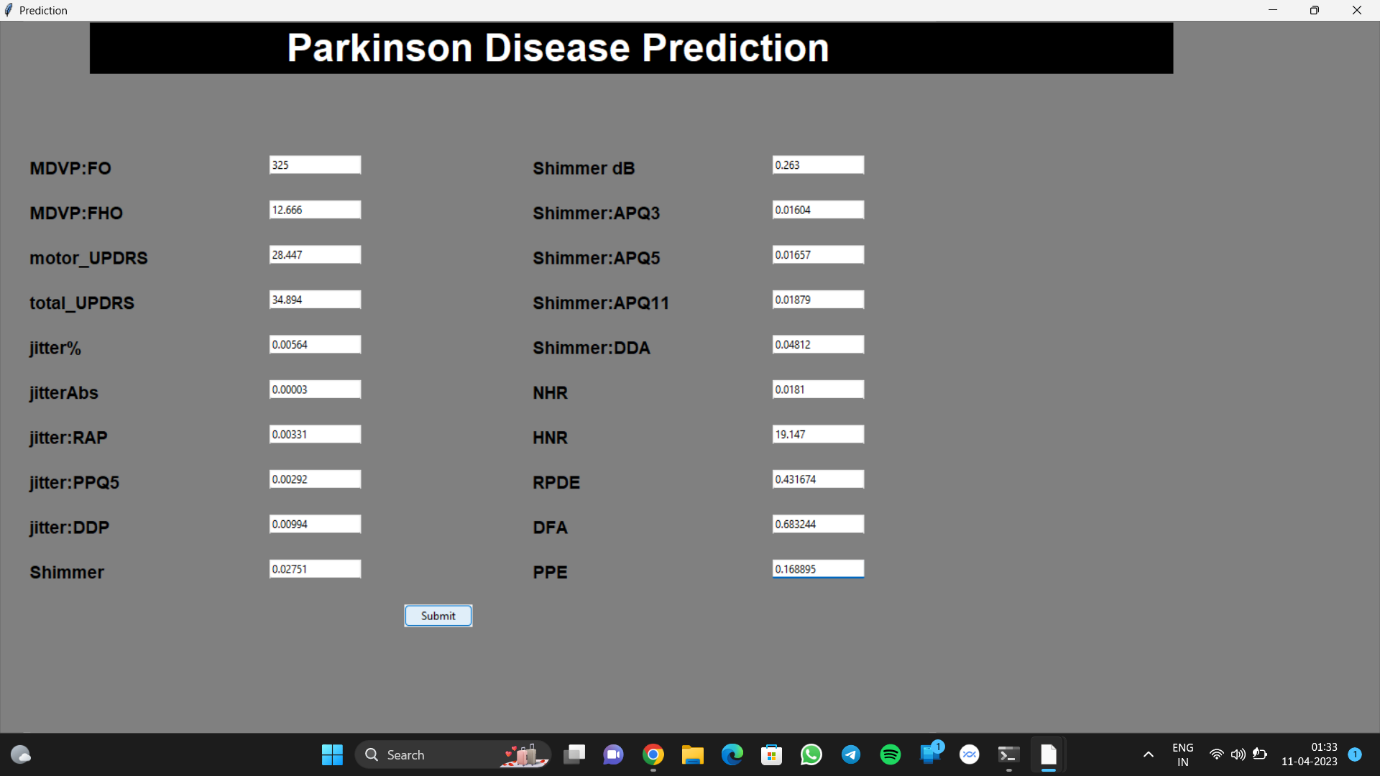


Fig 4.17 Prediction

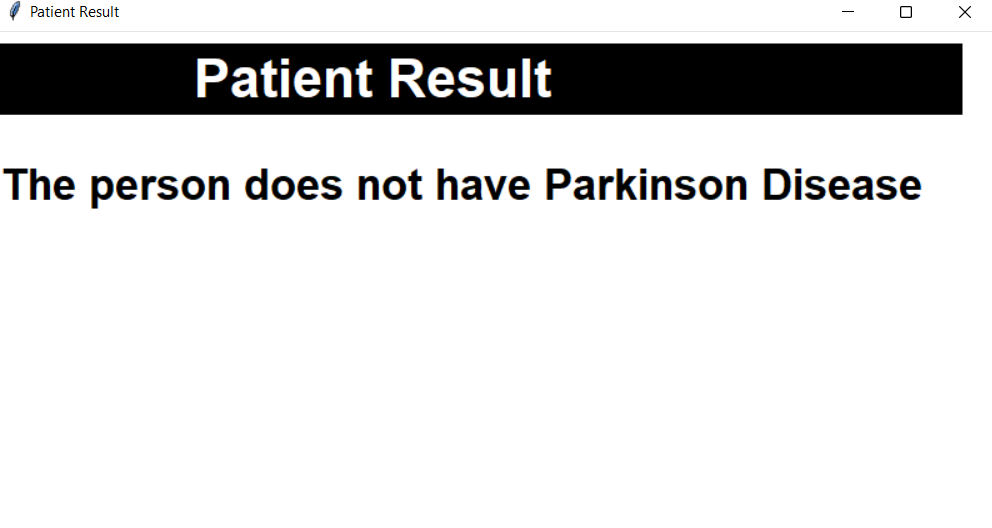


Fig 4.18 Result

**Chapter 5**

**Conclusion and Future Work**

**5.1 Conclusion:**

Parkinson's sickness is a cerebrum problem that influences the focal sensory system (CNS), and there is at present no solution for it except if it is analysed early. Late discovery brings about no treatment and passing. Accordingly, early location is basic. This utilized an assortment of AI methods to identify Parkinson's illness in its beginning phases. Parkinson's infection information is examined and found that XG Boost is the best calculation for anticipating the beginning of the condition, with a 96 percent precision rate, considering early treatment and perhaps saving a daily existence. The anticipated qualities while utilizing the XG Boost calculation to foresee the Parkinson's sickness

**5.2 Future Work:**

This project can be further used by adding image recognition for detecting the handwriting of patients so that this project can completely detect the Parkinson patients. With required modification in dataset this project can be adjusted for detecting diseases like heart disease or kidney disease as these algorithms performs better in healthcare.

**Chapter 6**

**References**

1. Lamba, Tarun Gulati, Anurag Jain, “A hybrid system for Parkinson’s disease diagnosis using machine learning techniques”, [International Journal of Speech TechnologyVolume 25Issue 3](https://dl.acm.org/toc/spijst/2022/25/3) 2022 pp 583–593
2. Varalakshmi P, Tharani Priya B, Bhuvaneaswari R, “Parkinson Disease Detection Based On Speech Using Various Machine Learning Models and Deep Learning Models”, DOI: [10.1109/ICSCAN53069.2021.9526372](https://doi.org/10.1109/ICSCAN53069.2021.9526372)
3. ie Mei, Christian Desrosiers, Johannes Frasnelli, “Machine Learning for the Diagnosis of Parkinson's Disease At early stage” Front. Aging Neurosci., 06 May 2021
4. G. Solana-Lavalle, J.-C. Galán-Hernández, and R. Rosas-Romero, “Automatic Parkinsondisease detection at early stages as a pre-diagnosis tool by using classiers and a small set of vocal features,'” Biocybern. Biomed.Eng., vol. 40, no. 1, pp. 505516, Jan. 2020*.*
5. Anila M, and Dr G Pradeepini, “Diagnosis of Parkinson’s disease using Artificial Neural network” International journal of emerging trends, 2020.
6. A Novel Framework of Two Successive Feature Selection Levels Using Weight-Based Procedure for Voice-Loss Detection in Parkinson’s Disease. IEEE Access PP (99):1-1
7. Anila M Department of CS1, Dr G Pradeepini Department of CSE, “DIAGNOSIS OF PARKINSON’S DISEASE USING ARTIFICIAL NEURAL NETWORK”, JCR, 7(19): 7260-7269, 2020.
8. Arvind Kumar Tiwari, “Machine Learning based Approaches for Prediction of Parkinson’s Disease” Machine Learning and Applications: An International Journal (MLAU) vol. 3, June 2016.
9. Carlo Ricciardi, et al, “Using gait analysis’ parameters to classify Parkinsonism: A data mining approach” Computer Methods and Programs in Biomedicine vol. 180, Oct. 2019.
10. Dr. Anupam Bhatia and Raunak Sulekh, “Predictive Model for Parkinson’s Disease through Naive Bayes Classification” International Journal of Computer Science & Communication vol-9, Dec. 2017, pp. 194- 202, Sept 2017 - March 2018.
11. Heisters. D, “Parkinson’s: symptoms, treatments and research”. British Journal of Nursing, 20(9), 548–554. doi:10.12968/bjon.2011.20.9.548, 2011.

**Chapter 7**

**Appendix (Source code)**

**BACK END**

import warnings

warnings.filterwarnings("ignore")

import pandas as pd

import numpy as np

import seaborn as sns

from matplotlib import pyplot as plt

import warnings

warnings.filterwarnings("ignore")

import pandas as pd

import numpy as np

import seaborn as sns

from matplotlib import pyplot as plt

Loading the data

data=pd.read\_csv("Parkinson.csv")

Dataset Details

data.columns

# using .shape we get to know about the number of rows and columns present in the dataset

data.shape

# displaying the first rows from the dataset

data.head()

# Remove two columns name is 'C' and 'D'

#data=data.drop(['ID', 'test\_time'], axis = 1)

data

data.describe() #statistical data about the dataset

# displaying the dataset of the columns in the dataset

data.dtypes

data['status'] = data['status'].astype('object')

EDA (Exploratory data analysis)

# Plotting the histogram of dataset

data.hist(figsize=(25,16))

plt.show()

list\_of\_non\_object\_cols = data.loc[:, data.dtypes != 'object'].columns.tolist()

Functions to ease up plotting of various continous variables

def Distribution\_Continous\_Variables(series,color,title):

plt.figure(figsize=(10, 5))

sns.distplot(series, color = color).set\_title(title)

def BoxPlot(\*\*kwargs):

plt.figure(figsize=(10, 5))

sns.boxplot(x = kwargs['x'], \

y = kwargs['y'], \

data = kwargs['data'], \

color = kwargs['color'], \

hue = kwargs['hue']).set\_title(kwargs['title'])

Univariate Analysis

Plots to analyse impact of continous on the status

Type Markdown and LaTeX: 𝛼2

Type Markdown and LaTeX: 𝛼2

for col in list\_of\_non\_object\_cols:

Distribution\_Continous\_Variables(data[data['status']==0][col],"green","Distribution of {} in the dataset for healthy patients"\

.format(col))

Distribution\_Continous\_Variables(data[data['status']==1][col],"red","Distribution of {} in the dataset for patients with parkinson's disease"\

.format(col))

for col in list\_of\_non\_object\_cols:

Distribution\_Continous\_Variables(data[data['status']==0][col],"green","Distribution of {} in the dataset for healthy patients"\

.format(col))

Distribution\_Continous\_Variables(data[data['status']==1][col],"red","Distribution of {} in the dataset for patients with parkinson's disease"\

.format(col))

Data Correlation

data.corr()

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

p = sns.heatmap(data.corr(), annot=True,cmap='YlGnBu')

Box Plot

for col in list\_of\_non\_object\_cols:

BoxPlot(x = 'status',\

y = col,\

data = data,\

color = 'orange',\

hue = 'status',\

title = 'Boxplot of {}'.format(col))

Data Preprocessing

data.info(verbose = True)

data.isnull().sum()

Removing Outliers:

from scipy.stats import iqr

import numpy as np

import pandas as pd

from scipy import stats

iqr(data, axis=0)

Q1 = np.quantile(data,0.25)

Q3 = np.quantile(data,0.75)

IQR = Q3 - Q1

for col in list\_of\_non\_object\_cols:

data['z\_score']=stats.zscore(data[col])

data.loc[data['z\_score'].abs()<=3]

for col in list\_of\_non\_object\_cols:

data['ewm\_alpha\_1']=data[col].ewm(alpha=0.1).mean()

data['ewm\_alpha\_3']=data[col].ewm(alpha=0.3).mean()

data['ewm\_alpha\_6']=data[col].ewm(alpha=0.6).mean()

data

Data Correlation

data.corr()

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

p = sns.heatmap(data.corr(), annot=True,cmap='YlGnBu')

data['status'].value\_counts()

sns.countplot(data['status'])

data\_fea=data.drop(['status','z\_score', 'ewm\_alpha\_1','ewm\_alpha\_3','ewm\_alpha\_6'],axis=1)

​

Model evaluation

XGBoost

from sklearn.preprocessing import MinMaxScaler

from xgboost import XGBClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

from sklearn.metrics import classification\_report, confusion\_matrix

​

#print(data\_fea)

​

​

features=data.loc[:,data.columns!='status'].values[:,1:]

labels=data.loc[:,'status'].values

scaler=MinMaxScaler((-1,1))

#scaler.fit(data)

​

x=scaler.fit\_transform(data\_fea)

y=labels

​

​

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x, y, test\_size=0.33, random\_state=7)

​

​

XG\_model =XGBClassifier(learning\_rate =0.1,n\_estimators=50,max\_depth=5,min\_child\_weight=1,gamma=0,subsample=0.8,colsample\_bytree=0.8,objective= 'binary:logistic',nthread=4,

scale\_pos\_weight=1,seed=2)

XG\_model.fit(x\_train, y\_train, eval\_metric="error",eval\_set=[(x\_train, y\_train), (x\_test, y\_test)])

​

​

y\_pred=XG\_model.predict(x\_test)

predictions = [round(value) for value in y\_pred]

ACCXg=accuracy\_score(y\_test.astype('float64'), y\_pred.astype('float64'))\*100

print("ACCURACY SCORE OF XGBOOST: %.f%%"%ACCXg)

results=XG\_model.evals\_result()

epochs=len(results['validation\_0']['error'])

print("Epochs",epochs)

# Calculate the Confusion Matrix for the model.

CM1=confusion\_matrix(y\_test.astype(int),y\_pred.astype(int))

print("Confusion Matrix\n",CM1)

ax = sns.heatmap(CM1, annot=True, cmap='Blues')

ax.set\_xlabel('\nPredicted Values')

ax.set\_ylabel('Actual Values ')

# Displaying the classification report using sklearn

print("Classification Report:\n",classification\_report(y\_test.astype(int),y\_pred.astype(int)))

Validation

from scipy.sparse.construct import random

from sklearn.model\_selection import StratifiedKFold

from sklearn.model\_selection import cross\_val\_score

import xgboost

model = xgboost.XGBClassifier()

kfold = StratifiedKFold(n\_splits=10, random\_state=None)

results = cross\_val\_score(model, x.astype(float), y.astype(int), cv=kfold)

print("Mean Accuracy: %.f%% \nStandard deviation percentage: %.2f%%"%(results.mean()\*100, results.std()\*100))

Random Forest

import numpy as np

import math

class RandomForest():

def \_init\_(self, x, y, n\_trees, n\_features, sample\_sz, depth=10, min\_leaf=2):

np.random.seed(12)

if n\_features == 'sqrt':

self.n\_features = int(np.sqrt(x.shape[1]))

elif n\_features == 'log2':

self.n\_features = int(np.log2(x.shape[1]))

else:

self.n\_features = n\_features

self.x, self.y, self.sample\_sz, self.depth, self.min\_leaf = x, y, sample\_sz, depth, min\_leaf

self.trees = [self.create\_tree() for i in range(n\_trees)]

​

def create\_tree(self):

idxs = np.random.permutation(len(self.y))[:self.sample\_sz]

f\_idxs = np.random.permutation(self.x.shape[1])[:self.n\_features]

return DecisionTree(self.x[idxs], self.y[idxs], self.n\_features, f\_idxs,

idxs=np.array(range(self.sample\_sz)),depth = self.depth, min\_leaf=self.min\_leaf)

def predict(self, x):

return np.mean([t.predict(x) for t in self.trees], axis=0)

​

def std\_agg(cnt, s1, s2): return math.sqrt((s2/cnt) - (s1/cnt)\*\*2)

​

class DecisionTree():

def \_init\_(self, x, y, n\_features, f\_idxs,idxs,depth=10, min\_leaf=5):

self.x, self.y, self.idxs, self.min\_leaf, self.f\_idxs = x, y, idxs, min\_leaf, f\_idxs

self.depth = depth

# print(self.depth)

self.n\_features = n\_features

self.n, self.c = len(idxs), x.shape[1]

self.val = np.mean(y[idxs])

self.score = float('inf')

self.find\_varsplit()

def find\_varsplit(self):

for i in self.f\_idxs: self.find\_better\_split(i)

if self.is\_leaf: return

x = self.split\_col

lhs = np.nonzero(x<=self.split)[0]

rhs = np.nonzero(x>self.split)[0]

lf\_idxs = np.random.permutation(self.x.shape[1])[:self.n\_features]

rf\_idxs = np.random.permutation(self.x.shape[1])[:self.n\_features]

self.lhs = DecisionTree(self.x, self.y, self.n\_features, lf\_idxs, self.idxs[lhs], depth=self.depth-1, min\_leaf=self.min\_leaf)

self.rhs = DecisionTree(self.x, self.y, self.n\_features, rf\_idxs, self.idxs[rhs], depth=self.depth-1, min\_leaf=self.min\_leaf)

​

def find\_better\_split(self, var\_idx):

x, y = self.x[self.idxs,var\_idx], self.y[self.idxs]

sort\_idx = np.argsort(x)

sort\_y,sort\_x = y[sort\_idx], x[sort\_idx]

rhs\_cnt,rhs\_sum,rhs\_sum2 = self.n, sort\_y.sum(), (sort\_y\*\*2).sum()

lhs\_cnt,lhs\_sum,lhs\_sum2 = 0,0.,0.

​

for i in range(0,self.n-self.min\_leaf-1):

xi,yi = sort\_x[i],sort\_y[i]

lhs\_cnt += 1; rhs\_cnt -= 1

lhs\_sum += yi; rhs\_sum -= yi

lhs\_sum2 += yi\*2; rhs\_sum2 -= yi\*2

if i<self.min\_leaf or xi==sort\_x[i+1]:

continue

​

lhs\_std = std\_agg(lhs\_cnt, lhs\_sum, lhs\_sum2)

rhs\_std = std\_agg(rhs\_cnt, rhs\_sum, rhs\_sum2)

curr\_score = lhs\_std\*lhs\_cnt + rhs\_std\*rhs\_cnt

if curr\_score<self.score:

self.var\_idx,self.score,self.split = var\_idx,curr\_score,xi

​

@property

def split\_name(self): return self.x.columns[self.var\_idx]

@property

def split\_col(self): return self.x[self.idxs,self.var\_idx]

​

@property

def is\_leaf(self): return self.score == float('inf') or self.depth <= 0

​

def predict(self, x):

return np.array([self.predict\_row(xi) for xi in x])

​

def predict\_row(self, xi):

if self.is\_leaf: return self.val

t = self.lhs if xi[self.var\_idx]<=self.split else self.rhs

return t.predict\_row(xi)

import pandas as pd

from sklearn.preprocessing import MinMaxScaler

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

​

#Get the features and labels

features=data.loc[:,data.columns!='status'].values[:,1:]

labels=data.loc[:,'status'].values

​

#Scale the features to between -1 and 1

scaler=MinMaxScaler((-1,1))

X=scaler.fit\_transform(features)

y=labels.astype(int)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=1)

print(X\_train.shape, X\_test.shape, y\_train.shape, y\_test.shape)

model2=RandomForest(X\_train,y\_train,10,20,321,10,2)

preds=model2.predict(X\_test)

#Checking the accuracy

ACCRF=accuracy\_score(y\_test.astype('int64'), preds.astype('int64'))\*100

print("ACCURACY Score Of Random Forest : %.f%%"%ACCRF)

Validation

from scipy.sparse.construct import random

from sklearn.model\_selection import StratifiedKFold

from sklearn.model\_selection import cross\_val\_score

from sklearn.ensemble import RandomForestClassifier

model2 = RandomForestClassifier(random\_state=0)

kfold = StratifiedKFold(n\_splits=5)

result = cross\_val\_score(model2, X.astype('float64'), y.astype('int64'), cv=kfold)

print("Mean Accuracy: %.f%% \nStandard deviation classification Accuracy: %.2f%%"%(result.mean()\*100, result.std()\*100))

from sklearn.metrics import classification\_report, confusion\_matrix

import seaborn as sns

CM= confusion\_matrix(y\_test.astype('int64'),preds.astype('int64'))

print("Confusion Matrix of Random Forest\n",CM)

ax = sns.heatmap(CM, annot=True, cmap='Blues')

ax.set\_xlabel('\nPredicted Values')

ax.set\_ylabel('Actual Values ')

print("Classification report of Random Forest: \n",classification\_report(y\_test.astype('int64'),preds.astype('int64')))

KNN

#Importing the required modules

import numpy as np

from scipy.stats import mode

#Euclidean Distance

def eucledian(p1,p2):

dist = np.sqrt(np.sum((p1-p2)\*\*2))

return dist

class KNN:

#Function to calculate KNN

def predict(x\_train,y , x\_input, k):

op\_labels = []

#Loop through the Datapoints to be classified

for item in x\_input:

#Array to store distances

point\_dist = []

#Loop through each training Data

for j in range(len(x\_train)):

distances = eucledian(np.array(x\_train[j,:]) ,item)

#Calculating the distance

point\_dist.append(distances)

point\_dist = np.array(point\_dist)

#Sorting the array while preserving the index

#Keeping the first K datapoints

dist = np.argsort(point\_dist)[:k]

#Labels of the K datapoints from above

labels = y[dist]

#Majority voting

lab = mode(labels)

lab = lab.mode[0]

op\_labels.append(lab)

return op\_labels

#Importing the required modules

#Importing required modules

from sklearn.preprocessing import MinMaxScaler

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

from numpy.random import randint

#Get the features and labels

features=data.loc[:,data.columns!='status'].values[:,1:]

labels=data.loc[:,'status'].values

#Scale the features to between -1 and 1

scaler=MinMaxScaler((-1,1))

x=scaler.fit\_transform(features)

y=labels.astype(int)

#Split the dataset

X\_train,X\_test,y\_train,y\_test=train\_test\_split(x, y, test\_size=0.3,random\_state=5)

model3=KNN()

#Applying our function

y\_pred = KNN.predict(X\_train,y\_train,X\_test,10)

#Checking the accuracy

ACCKNN=accuracy\_score(y\_test, y\_pred)\*100

print("ACCURACY Score Of KNN: %.f%%"%ACCKNN)

from sklearn.metrics import classification\_report, confusion\_matrix

import seaborn as sns

CM= confusion\_matrix(y\_test,y\_pred)

print("Confusion Matrix of KNN\n",CM)

ax = sns.heatmap(CM, annot=True, cmap='Blues')

ax.set\_xlabel('\nPredicted Values')

ax.set\_ylabel('Actual Values ')

print("Classification report of KNN: \n",classification\_report(y\_test,y\_pred))

from scipy.sparse.construct import random

from sklearn.model\_selection import StratifiedKFold

from sklearn.model\_selection import cross\_val\_score

from sklearn.neighbors import KNeighborsClassifier

model3 = KNeighborsClassifier(n\_neighbors=10)

kfold = StratifiedKFold(n\_splits=5)

results = cross\_val\_score(model3, X.astype('float64'), y.astype('int64'), cv=kfold)

print("Mean Accuracy: %.f%% \nStandard deviation classification Accuracy: %.2f%%"%(results.mean()\*100, results.std()\*100))

Accuracy Plot

import matplotlib.pyplot as plt

def AccPlot():

fig=plt.figure(figsize=(7,5))

ax=fig.add\_axes([0,0,1,1])

models= ['XGBOOST','RANDOM FOREST','KNN']

acc=float(format(ACCXg, '.0f'))

acc1=float(format(ACCRF,".0f"))

acc2=float(format(ACCKNN,".0f"))

scores=[acc,acc1,acc2]

ax.set\_ylabel('Accuracy Score(%)',fontsize=12)

ax.set\_xlabel('\n Model',fontsize=12)

ax.set\_title("Models Accuracy Score \n")

for i,v in enumerate(scores):

ax.text(i,v+1,str(v),color='black',fontsize=18)

plt.savefig('barplot\_1.png',dpi=100,format='png',bbox\_inches='tight')

plt.bar(models,scores, color=['firebrick', 'green', 'blue'])

plt.show()

AccPlot()

data\_in=data.drop(['z\_score', 'ewm\_alpha\_1','ewm\_alpha\_3','ewm\_alpha\_6'], axis =1)

#input data and transform into numpy array

in\_data= np.asarray(tuple(map(float,input().rstrip().split(','))))

in\_data=in\_data.reshape(1,-1)

in\_data\_sca = scaler.fit\_transform(in\_data)

​

#print the predicted output for input array

output=XG\_model.predict(in\_data)

if output==1:

print("Does Has Parkinson's Disease")

else:

print("Does not Has Parkinson's Disease")

​

#print("Parkinson's Disease Detected" if XG\_model.predict(in\_data) else "No Parkinson's Disease Detected")

#325,12.666,28.447,34.894,0.00564,0.00003,0.00331,0.0292,0.00994,0.02751,0.263,0.01604,0.01657,0.1879,0.04812,0.0181,19.147,0.431674,0.683244,0.168895

#Parkinson Disease Detected

#9,117.66,33.178,42.904,0.01713,0.000169,0.01021,0.00865,0.03064,0.04686,0.446,0.2312,0.02679,0.3946,0.06935,0.05563,17.387,0.75697,0.77215,0.45606

#No Parkinson Disease Detected

#Parkinson Disease is Detected

#325,152.64,35.509,46.013,0.0136,0.00008,0.00624,0.00564,0.01873,0.02308,0.256,0.01268,0.01365,0.01667,0.03804,0.10715,17.883,0.407567,0.655683,0.131728

​

# No parkinson

#297,117.67,33.178,42.904,0.00742,0.00005,0.00364,0.00432,0.01092,0.05517,0.542,0.2471,0.03572,0.05767,0.07413,0.0316,17.28,0.665318,0.719467,0.377429

**FRONT END**

import tkinter

from tkinter import ttk

from tkinter import \*

from PIL import ImageTk, Image

import pandas as pd

import tkinter as tk

from tkinter import filedialog

from tkinter.filedialog import askopenfile

​

def DATASET():

​

my\_w = tkinter.Tk()

my\_w.geometry("800x800") # Size of the window

my\_w.title('Dataset')

​

my\_font1=('times', 12, 'bold')

l1 = tk.Label(my\_w,text='Read File & create DataFrame',

width=30,font=my\_font1)

l1.grid(row=1,column=1)

b1 = tk.Button(my\_w, text='Browse File',

width=20,command = lambda:upload\_file())

b1.grid(row=2,column=1)

t1=tk.Text(my\_w,width=200,height=200)

t1.grid(row=3,column=1,padx=5)

​

def upload\_file():

f\_types = [('CSV files',".csv"),('All',".\*")]

file = filedialog.askopenfilename(filetypes=f\_types)

#l1.config(text=file) # display the path

df=pd.read\_csv(file) # create DataFrame

#str1="Rows:" + str(df.shape[0])+ "\nColumns:"+str(df.shape[1])

#print(str1)

t1.insert(tk.END, df) # add to Text widget

my\_w.mainloop() # Keep the window open

​

​

​

​

#for i in v:

#print(i)

​

newWindow3 = tkinter.Toplevel(top)

newWindow3.title("Patient Result")

newWindow3['background']='white'

newWindow3.geometry("800x450")

label=tkinter.Label(newWindow3,

text =' Patient Result \t \t', bg='black', fg="white", font=("Serif", 32, 'bold')).place(x = 0,y = 10)

​

d={'MDVP:FO' :[d1] ,'MDVP:FHO' :[d2],' motor\_UPDRS':[d3],'total\_UPDRS':[d4],'Jitter':[d5],'Jitter(Abs)':[d6],'Jitter:RAP':[d7],'Jitter:PPQ5':[d8],

'Jitter:DDP':[d9],'Shimmer':[d10],'Shimmer(dB)':[d11],'Shimmer:APQ3':[d12],'Shimmer:APQ5':[d13],'Shimmer:APQ11':[d14],

'Shimmer:DDA':[d15],'NHR':[d16],'HNR':[d17],'RPDE':[d18],'DFA':[d19],'PPE':[d20]}

#print(type(d1))

#print(type(d3))

df3=pd.DataFrame(d)

#print(df3)

pred = XG\_model.predict(df3)

​

r=0

for i in v:

if i==0:

r+=1

​

if r!=0:

result="Invalid Input"

elif pred==1:

result="Does Have Parkinson Disease"

else:

result="Does not has Parkinson Disease"

​

​

tkinter.Label(newWindow3,text =result,bg='white',font=("Serif", 26,'bold')).place(x = 2,y = 100)

sub=ttk.Button(top,text="Submit",command=Submit).place(x=450,y=650)

Prediction.mainloop()

​

​

global img

from tkinter import \*

from PIL import ImageTk,Image

def ACCURACYPLOT():

root =tkinter.Toplevel()

root.geometry("811x575")

img1=PhotoImage(file='plot.png')

Label(root,image=img1).pack()

​

root.mainloop()

top = tkinter.Toplevel()

top.title("Parkinson's Disease Prediction")

label=tkinter.Label(top, text =" Parkinson's Disease Prediction \t \t", bg='black', fg="white", font=("Serif", 32, 'bold')).place(x = 0,y = 10)

top.geometry('1000x1000')

img = ImageTk.PhotoImage(file="bg.jpg")

label1 = tkinter.Label(

top,

image=img

)

label1.place(x=0, y=0)

btn0 = tkinter.Button(top, text="DATASET", command=DATASET,bg="white", font=("Serif", 11)).place(x=50, y=300)

btn1 = tkinter.Button(top, text="ACCURACY PLOT",command=ACCURACYPLOT,bg="white", font=("Serif", 11)).place(x=200, y=300)

btn2 = tkinter.Button(top, text="ACCURACY",command=Accuracy,bg="white", font=("Serif", 11)).place(x=400, y=300)

btn3 = tkinter.Button(top, text="PREDICTION",command=Prediction, bg="white", font=("Serif", 11)).place(x=600, y=300)

top.mainloop()