

INDUSTRY INTERNSHIP REPORT

*Submitted in
partial fulfillment of requirement for the award of degree of*

Bachelor of Technology in Electronics and Telecommunication Engineering

By

**Ms. Kannavi Yeole
Ms. Sakshi Badwaik
Mr. Manthan Ukey
Ms. Prashansa Kumbhare**

**Mr. Niraj Katwe
Mr. Ayush Sakhare
Mr. Nikhil Wagh**

Industry / Organization Guide

Mr. Sunny Meshram

at

Aotom Technologies

Institute Guide (from college)

Prof. Swapna Choudhary

Assistant Professor



**Department of Electronics and Telecommunication Engineering
G H Raison College of Engineering, Nagpur**

(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur)

Accredited by NAAC with “A+” Grade

Ranked 163th by NIRF, in the Engineering Category for India Ranking 2022,

June 2023

INDUSTRY INTERNSHIP REPORT

*Submitted in
partial fulfillment of requirement for the award of degree of*

Bachelor of Technology in Electronics and Telecommunication Engineering

By

**Ms. Kannavi Yeole
Ms. Sakshi Badwaik
Mr. Manthan Ukey
Ms. Prashansa Kumbhare**

**Mr. Niraj Katwe
Mr. Ayush Sakhare
Mr. Nikhil Wagh**

Industry / Organization Guide

Mr. Sunny Meshram

at

Aotom Technologies

Institute Guide (from college)

Prof. Swapna Choudhary

Assistant Professor



Department of Electronics and Telecommunication Engineering

G H Raison College of Engineering, Nagpur

(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur)

Accredited by NAAC with “A+” Grade

Ranked 163th by NIRF, in the Engineering Category for India Ranking 2022,

June 2023

Declaration

We, hereby declare that the Industry Internship report submitted herein has been carried out by us in **Aotom Technologies** towards partial fulfillment of requirement for the award of Degree of Bachelor of Technology in Electronics and Telecommunication Engineering. The work is original and has not been submitted earlier as a whole or in part for the award of any degree at this or any other Institution / University.

We also hereby assign to G. H. Rasoni College of Engineering, Nagpur all rights under copyright that may exist in and to the above work and any revised or expanded derivatives works based on the work as mentioned. Other work copied from references, manuals etc. are disclaimed.

Name of student	Mobile No	Mail ID (Other than Rasoni.net)	Signature
Kannavi Yeole	7058572457	kannaviyeole@gmail.com	
Niraj Katwe	9022972560	nirajkatwe14444@gmail.com	
Sakshi Badwaik	9373758687	sakshi14badwaik@gmail.com	
Ayush Sakhare	7709175977	ayushsakhare786@gmail.com	
Manthan Ukey	8080111528	manthan18ukey@gmail.com	
Nikhil Wagh	8975399431	nikhilwagh897@gmail.com	
Prashansa Kumbhare	9156722110	prashansa.kumbhare@gmail.com	

Place: Nagpur

Date:

Certificate

The Industry Internship Report entitled as “**Parkinson’s Disease Prediction Using Machine Learning**” carried out under our supervision in **Aotom Technologies** by **Kannavi Yeole, Niraj Katwe, Sakshi Badwaik, Ayush Sakhare, Manthan Ukey, Nikhil Wagh, Prashansa Kumbhare** for the award of Degree of Bachelor of Technology in Electronics and Telecommunication Engineering. The work submitted is comprehensive, complete and fit for evaluation.

Mr. Sunny Meshram
Industry Guide
Project Manager
Aotom Technologies

Prof. Swapna Choudhary
Institute Guide
Assistant Professor
Department of Electronics and
Telecommunication Engineering
G H R C E, Nagpur

Dr. Surekha Tadse
III Coordinator
Department Electronics and
Telecommunication Engineering
G H R C E, Nagpur

Dr. Atul Deshmukh
Head
Department Electronics and
Telecommunication Engineering
G H R C E, Nagpur

Mr. Gurpal Singh
Dean III Cell
G H R C E, Nagpur

Dr. Sachin Untawale
Director
G H R C E, Nagpur

Date: 10th June 2023

Certificate

This is to certify that **Mr. Manthan Ukey, Mr. Niraj Katwe, Mr. Ayush Sakhare, Mr. Nikhil Wagh, Ms. Kannavi Yeole, Ms. Prashansa Kumbhare and Ms. Sakshi Badwaik**, students of Electronics and Telecommunication Engineering from **G H Raison College of Engineering, Nagpur**, has completed internship successfully from **10/12/2022 to 09/06/2023**. During this period, they have shown good interest in the assignment/works given to them and worked hard. During their tenure of internship, they were hard working and focused on activities assigned to them.

They have worked during internship period on following project under the guidance of

1. Mr. Sunny Meshram (Manager)
2. Mr. Rihal Kalbende (Senior Python Developer)

Regards,



Signature

Name of Head: **Sunny Meshram**

Designation: **Manager**

Aotom Technologies

Address: 301, Suryakiran Complex, 3rd Floor Abhyankar Nagar Rd, Opposite VNIT Gate Nagpur 440010.

Date: 10th June 2023

Cost of Industrial Solution Certificate

This is to certify that **Mr. Manthan Ukey, Mr. Niraj Katwe, Mr. Ayush Sakhare, Mr. Nikhil Wagh, Ms. Kannavi Yeole, Ms. Prashansa Kumbhare and Ms. Sakshi Badwaik**, students of Electronics and Telecommunication Engineering from **G H Raison College of Engineering, Nagpur**, has completed Internship successfully from **10/12/2022 to 09/06/2023**. During this period they have shown good interest in the assignment/works given to them and worked hard.

Students have worked during internship period on following project problem under the guidance of **Mr. Rihal Kalbende (Senior Python Developer)**.

Project Title: "**Parkinson Disease Prediction Using Machine Learning**"

Industry has spent Rs. 50,000/- amount on their ideas/industries problem, which they have successfully implemented.

Regards,



Signature

Name of Head: **Sunny Meshram**

Designation: **Manager**

Aotom Technologies

Address: 301, Suryakiran Complex, 3rd Floor Abhyankar Nagar Rd, Opposite VNIT Gate Nagpur 440010.

Date: 10th June 2023

Savings to Industry Certificate

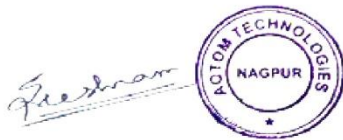
This is to certify that **Mr. Manthan Ukey, Mr. Niraj Katwe, Mr. Ayush Sakhare, Mr. Nikhil Wagh, Ms. Kannavi Yeole, Ms. Prashansa Kumbhare and Ms. Sakshi Badwaik**, students of Electronics and Telecommunication Engineering from **G H Raison College of Engineering, Nagpur**, has completed Internship successfully from **10/12/2022 to 09/06/2023**. During this period they have shown good interest in the assignment/works given to them and worked hard.

Students have worked during internship period on following project/industrial problem under the guidance of **Mr. Rihal Kalbende (Senior Python Developer)**.

Project Title: **"Parkinson Disease Prediction Using Machine Learning"**

Project work submitted by them has the potential to save cost up to NIL. Also they were not entitled for stipend along with no canteen, transportation & accommodation facilities.

Regards,



Signature

Name of Head: **Sunny Meshram**

Designation: **Manager**

Aotom Technologies

Address: 301, Suryakiran Complex, 3rd Floor Abhyankar Nagar Rd, Opposite VNIT Gate Nagpur 440010.

ACKNOWLEDGEMENT

At the very outset, I would like to express my heartiest and sincere gratitude to our Head of Department, **Dr. Atul Deshmukh** for his untiring and inspiring guidance, patience and close supervision throughout the period of project. I am very grateful to **Dr. Sachin Untawale** Director of GHRCE, Nagpur for his constant support and for providing necessary facilities for conducting out the project.

I am highly indebted to our Industry Guide, **Mr. Sunny Meshram** for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project.

I wish to thank our College Guide, **Prof. Swapna Choudhary** who have extended assistance to us whenever it was needed and for giving such attention and time.

Our thanks and appreciations also go to our college in developing the project and people who have willingly helped us out with their abilities. Finally, as one of the team members, I would like to appreciate all my group members for their support and coordination, I hope we will achieve more in our future endeavors.

ABSTRACT

Parkinson's disease (PD) is a progressive neurodegenerative disorder that affects millions of people worldwide. Early detection and accurate prediction of PD can significantly contribute to timely intervention and improved patient outcomes. Machine learning techniques have emerged as promising tools for the prediction and diagnosis of various diseases, including PD. In this study, we propose a machine learning-based approach for the prediction of Parkinson's disease. We collected a comprehensive dataset consisting of clinical and demographic features from a large cohort of patients diagnosed with PD. The dataset was preprocessed to handle missing values and outliers and underwent feature selection to identify the most relevant features for prediction. Various machine learning algorithms, including decision trees, support vector machines, and random forests, were applied to the dataset to develop predictive models.

The performance of the models was evaluated using rigorous cross-validation techniques and metrics such as accuracy, sensitivity, specificity, and area under the receiver operating characteristic curve (AUC-ROC). Our results demonstrate that the machine learning models achieved high accuracy and predictive performance in differentiating between PD and non-PD cases. The selected features provided valuable insights into the key clinical factors associated with PD. Furthermore, we developed a user-friendly web-based application that utilizes the trained model for real-time prediction of PD based on inputted patient data. This application can be a useful tool for clinicians and healthcare professionals in early detection and monitoring of Parkinson's disease.

Overall, this study highlights the potential of machine learning techniques in predicting Parkinson's disease and offers a practical approach for healthcare professionals to leverage these tools in clinical practice. Early detection and intervention facilitated by machine learning-based prediction models have the potential to improve patient outcomes and enhance the management of Parkinson's disease.

LIST OF FIGURES

Figure No	Figure Name	Page No.
4.1	Data Flow Diagram	15
4.2	Dataset of Parkinsons Disease	17
4.3	Importing Dataset into Jupyter Notebook	18
4.4	Pandas Function in Jupyter Notebook	18
4.5	Data Visualization	19
4.6	Data Preprocessing	19
4.7	Splitting Dataset into Training and Testing	20
5.1	XGBoost Algorithm	23
5.2	KNN Algorithm	25
5.3	Random Forest Classification Algorithm	27
5.4	Logistic Regression Algorithm	29
5.5	Support Vector Machine Algorithm	31
5.6	Decision Tree Algorithm	34
5.7	Naive Bayes Algorithm	36
6.1	Sample Test Result - Negative Case	38
6.2	Sample Test Result - Positive Case	39
6.3	Accuracy Comparision Plot	39

INDEX

Sr. No.	CONTENTS	Page No.
1	Abstract	I
2	List of figures	II
3	CHAPTER 1: Introduction 1.1 Introduction 1.2 Problem Statement 1.3 Objective	2 3 4 5
4	CHAPTER 2: Literature review	6
5	CHAPTER 3: Tools/Platform Used 3.1 Hardware Requirements 3.2 Software Requirements 3.3 Libraries Used	11 12 12 13
6	CHAPTER 4: Methodology 4.1 Methodology 4.2 Dataset for prediction of Parkinson's Disease 4.3 Data Validation	14 15 19 20
7	CHAPTER 5: Implementation 5.1 XGBoost 5.2 Knn 5.3 Random Forest Classification 5.4 Logistic Regression 5.5 Support Vector Machine 5.6 Decision Tree 5.7 Naive Bayes	24 25 27 29 31 33 35 37
8	CHAPTER 6: Result	39
9	CHAPTER 7 : Conclusion	42
10	References	43

CHAPTER 1

INTRODUCTION

1.1 Introduction

Parkinson's disease (PD) is a chronic and progressive neurodegenerative disorder that primarily affects the motor system. It is characterized by the degeneration and loss of dopamine-producing cells in the substantia nigra region of the brain. PD affects millions of people worldwide, and its prevalence is expected to increase with the aging population. Early detection and accurate prediction of PD are crucial for effective treatment, management, and improved patient outcomes. Traditional methods of diagnosing PD rely on clinical evaluation by neurologists, which often involves subjective assessments of motor symptoms. However, these methods may lead to misdiagnosis or delayed diagnosis, limiting the effectiveness of interventions. The advent of machine learning techniques has opened up new possibilities for the prediction and diagnosis of PD, offering a data-driven and objective approach.

Machine learning algorithms can analyze large datasets and identify complex patterns and relationships within the data. By training on a dataset of known PD cases, these algorithms can learn to recognize patterns that are indicative of the disease. Once trained, the models can then be used to predict the presence or likelihood of PD in new, unseen cases based on their input data.

The use of machine learning in PD prediction has several advantages. Firstly, it allows for the incorporation of diverse data sources, including clinical, genetic, and demographic variables, providing a more comprehensive view of the disease. Secondly, machine learning models can handle large amounts of data and identify subtle patterns that may not be apparent to human observers. Lastly, the development of predictive models can contribute to personalized medicine approaches by tailoring treatment plans based on individual risk profiles.

In this study, we aim to explore the potential of machine learning techniques for PD prediction. We will collect a comprehensive dataset of clinical and demographic features from a large cohort of PD patients and control subjects. The dataset will undergo preprocessing steps to handle missing values and outliers. Feature selection methods will be employed to identify the most informative variables for prediction.

1.2 Problem Statement

Parkinson's disease (PD) is a debilitating neurodegenerative disorder that affects millions of individuals worldwide. Early and accurate prediction of PD is crucial for timely intervention, personalized treatment plans, and improved patient outcomes. However, the current diagnostic methods for PD rely on subjective clinical assessments, which can lead to misdiagnosis or delayed diagnosis.

The aim of this study is to address the need for an objective and data-driven approach to predict Parkinson's disease using machine learning techniques. By leveraging the power of machine learning algorithms, we seek to develop predictive models that can analyze diverse datasets and identify patterns indicative of PD. These models will enable healthcare professionals to make more accurate predictions and improve the overall management of PD.

Key challenges to be addressed include:

- **Limited predictive accuracy:** The existing methods for PD prediction may not provide sufficiently accurate results, leading to misdiagnosis or false-negative predictions. There is a need to develop machine learning models that can achieve high predictive accuracy and reliably differentiate between PD and non-PD cases.
- **Incorporating diverse data sources:** PD is a complex disease influenced by various clinical, genetic, and demographic factors. It is essential to collect and integrate diverse data sources to develop comprehensive prediction models. However, effectively incorporating and handling different data types pose challenges in feature selection, preprocessing, and model training.
- **Generalizability of the models:** Machine learning models should not only perform well on the training dataset but also generalize to new, unseen cases. Achieving model generalizability is crucial to ensure that the predictive models can be effectively applied in real-world clinical settings.

1.3 Objective

Objective for Parkinson's disease prediction using machine learning:

- Develop accurate machine learning models for predicting Parkinson's disease based on clinical and demographic features.
- Explore the effectiveness of different machine learning algorithms, such as decision trees, support vector machines, KNN, naive bayes, logistic regression, XGboost and random forest classification, in predicting Parkinson's disease.
- Preprocess the dataset to handle missing values, outliers, and feature engineering to enhance the predictive performance of the models.
- Evaluate and compare the performance of the developed models using metrics like accuracy, sensitivity, specificity, and the area under the receiver operating characteristic curve (AUC-ROC).
- Investigate the contribution of different clinical and demographic features in predicting Parkinson's disease and identify key factors associated with the disease.
- Assess the generalizability of the developed models to unseen data to ensure their applicability in real-world clinical settings.
- Develop an intuitive and user-friendly web-based application that utilizes the trained model for real-time prediction of Parkinson's disease.
- Enhance the interpretability of the machine learning models to provide clinicians and healthcare professionals with insights into the underlying factors influencing the predictions.
- Contribute to the advancement of knowledge in the field of Parkinson's disease prediction using machine learning and provide valuable insights for future research and clinical practice.

CHAPTER 2
LITERATURE
REVIEW

LITERATURE REVIEW

D. Gil, and M. Johnson, Diagnosing parkinson by using artificial neural networks and support vector machines.

Ipsita Bhattacharya et al [1] used SVM, a supervised machine learning algorithm, to separate the PD subjects from healthy subjects. They used weka - a tool for data mining. Preprocessing of data was performed on the dataset [21] before applying the classification algorithm. The random split was done repeatedly and the best possible accuracy was found on the different kernel values by applying libsvm

M. Shahbakhi, D.T. Far, E. Tahami, Speech analysis for diagnosis of Parkinson's Disease using Genetic Algorithm and Support Vector Machine.

Richa Mathur et al [2] suggested a method for predicting PD. They used a weka tool for implementing the algorithms to perform pre-processing of data, classification and the result analysis on the given dataset [21]. They used K-NN along with Adaboost.M1, bagging, and MLP. Results showed that K-NN + Adaboost.M1 and k-NN + MLP yielded the same accuracy of 91.28 % while the time taken for building the model with Adaboost.M1 was 0.01 and with MLP it was 0.43

R. Mathur, V. Pathak, & D. Bandil, Parkinson Disease Prediction Using Machine Learning Algorithm.

Amin ul Haq et al [3] proposed a classifier model that classifies the subjects into control and people with Parkinson subjects. In their study, they used the L1-norm SVM algorithm to get the appropriate features selected and then classification was performed. Their proposed system involved 1) data preprocessing, 2) selection of appropriate features, 3) crossvalidation (CV) and 4) evaluation of classifiers. 10-fold CV was used for SVM (with both linear and RBF kernel).

A. Haq, J.P. li , M.H. Memon , J. Khan , A. Malik , T. Ahmad , A. Ali , S. Nazir , I. Ahad, and M. Shahid, Feature selection based on L1-Norm Support Vector Machine and Effective Recognition System for Parkinson's disease using voice recordings.

Diogo Braga et al [4] presented a new technique to detect the early signs of PD through the analysis of speech. They used algorithms like SVM, random forest, NB and neural network. In their framework, three speech datasets were used. The first one was collected from 22 PD subjects that contain 1002 speech lines.

B. D.Anisha, N. Arunlanand, Early prediction of Parkinson's Disease (PD) using Ensemble Classifiers.

C.D. Anisha et al [5] employed the use of dataset [24] on which they applied LDA and PCA to select the highly correlated features. Gradient boosting machine (GBM), extreme gradient boosting (XGBoost), bagging and adaptive boosting (AdaBoost) were used as ensemble classifiers for performing the classification.

O. Asmae, R. Abdelhadi, C. Bouchaib, S. Sara, and K. Tajeddine, Parkinson's disease identification using KNN and ANN Algorithms based on Voice Disorder.

O. Asmae et al [6] used ANN and K-NN for distinguishing the PD and healthy ones. The dataset [20] consisted of a total of 31 subjects in which 23 were of the PD category. They used MATLAB tool for the purpose of classification. For ANN, 70% of data was used for training the model, 5% for validation purpose and 25% for testing respectively. An accuracy of 96.7% was reported by ANN. For k-NN, the data consisted of 70% of training data and 30% of testing data

Z. K. Senturk, “Early diagnosis of Parkinson’s disease using machine learning algorithms.

Z.K. Senturk [7] presented a framework for the early diagnosis of PD. They employed the RFE method as a feature selection (FS) task on a dataset [20]. They used ANN, CART and support vector machines as classification methods.

Tuncer, Turker, Dogan, Sengul, Acharya, and U. Rajendra, Automated detection of Parkinson's disease using minimum average maximum tree and singular value decomposition method with vowels.

Tuncer et al [8] presented a novel method to detect the PD. The minimum average maximum tree was used along with a combination of singular valued decomposition (SVD). On the original dataset [24], the top 50 features were selected by the relief feature selection method. The k-NN was used as a classifier with a 10-fold CV.

A. H. Al-Fatlawi, M.H. Jabardi, and S.H. Ling, Efficient diagnosis system for Parkinson’s disease using deep belief network.

Ali H.et al [10] suggested a deep belief network (DBN) as an efficient model for the diagnosis of PD. The dataset used was retrieved from the UCI repository [21]. In the context of their work, DBN was composed of two Stacked Restricted Boltzmann machines (RBMs) with an output layer. For optimizing the network parameters, the learning involved two stages

S. Grover, S. Bhartia, Akshama, A. Yadav, and K. R. Seeja, “Predicting severity of Parkinson’s disease using deep learning.

Srishti Grover et al [11] proposed a model based on deep learning for predicting the severity of PD. The dataset was taken from the UCI repository [7]. The data was collected from 42 patients and from each patient 200 recordings were taken which comprised a total of 5875

recordings of voice. The dataset had sixteen speech biomedical attributes. The voice data was normalized by using min-max normalization. 80% and 20% of normalized data were used for training and testing respectively

S. S.Upadhy, and A.N. Cheeran, Discriminating Parkinson and healthy people using phonation and cepstral features of speech.

Savitha S. Upadhy et al [12] used a neural network for classifying the subjects into Parkinson diseased and control people. In their work, they extracted the cepstral features and phonation of speech. The phonation included 14 features and the cepstral features were 12 MFCCs. These features were fed in a neural network and classification was done. They used a feed-forward network as a neural network (NN) classifier that had two layers and one hidden layer.

P. C. Rajagopal, T. Choudhury, A. Sharma, and P.Kumar, Diagnosis of Parkinson's diseases using classification based on voice recordings

In this paper, P. Chitra Rajagopal et al [13] used a neural network to diagnose the PD. The dataset used [21] consists of 23 speech attributes which were collected from 31 individuals of which 8 were healthy subjects and the remaining were suffering from PD. Their proposed neural network used 23 neurons in the input layer and the output layer had two neurons one for the control subject and one for the PD subject.

CHAPTER III

TOOLS / PLATFORM USED

3.1 Hardware Requirements:

Laptop with Windows 10 and above

8GB RAM and above

Any Processor will be compatible (Intel or Ryzen)

250 MB HDD.

3.2 Software Requirements:

- Python (IDLE version 3.7 and above)
- Jupyter Notebook

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is often used as a support language for software developers, for build control and management, testing, and in many other ways. Python has a simple syntax similar to the English language.

Since it's relatively easy to learn, Python has been adopted by many non-programmers such as accountants and scientists, for a variety of everyday tasks, like organizing finances

Jupyter supports over 40 programming languages, including Python, R, Julia, and Scala. The best environment to capture this style of programming is Jupyter notebooks. It can work on bits of code, spell out thoughts and are satisfied, put it all together.

They're great for iterative, experimental style of programming, Jupyter Notebook is an open-source web application that allows a user, scientific researcher, scholar or analyst to create and share the document called the Notebook, containing live codes, documentation, graphs, plots, and visualizations. The Notebook is now called the Jupyter Notebook. This interface can be used not only with Python but with dozens of other languages such as R and Julia. IPython is now the name of the Python backend (aka kernel). In conclusion, IPython and Jupyter are great interfaces to the Python language.

While Jupyter and Colab are notebook-based code editor which uses IPython (an interactive shell built with python). Limited Space & Time: The Google Colab platform stores files in Google Drive with a free space of 15GB; however, working on bigger datasets requires more space, making it difficult to execute. This, in turn, can hold most of the complex functions to execute.

3.3 Libraries Used:

Matplotlib

It can generate static, dynamic, and interactive visuals with the Python program Matplotlib. Matplotlib makes both challenging and basic tasks possible.

Seaborn

A Python data visualisation toolkit called Seaborn uses the matplotlib library. It provides a sophisticated interface for designing eye-catching and illuminating statistical graphics. Seaborn helps to explore and understand the data.

Scikit or Sklearn

Without a doubt, Python's most useful machine learning library is Scikit-learn. The sklearn toolkit for machine learning and statistical modelling includes a number of helpful features, including classification, regression, clustering, & dimensionality reduction. With sklearn, machine learning models were developed.

CHAPTER 4

METHODOLOGY

METHODOLOGY

Machine learning has given computer systems the ability to automatically learn without being explicitly programmed. In this project, seven machine learning algorithms (Logistic Regression, KNN, XGBoost, Decision Tree, Support Vector Machine, Random Forest Classification and Naïve Bayes).

The architecture diagram describes the high-level overview of major system components and important working relationships. It represents the flow of execution and it involves the following five major steps: -

- The architecture diagram is defined with the flow of the process which is used to refine the raw data and used for predicting the Parkinson's data.
- The next step is preprocessing the collected raw data into an understandable format.
- Then train the data by splitting the dataset into train data and test data.
- The Parkinson's data is evaluated with the application of a machine learning algorithm that is Logistic Regression, KNN, Decision Tree, XGBoost, Random Forest Classification, Support Vector Machine and Naïve Bayes algorithm, and the classification accuracy of this model is found.
- After training the data with these algorithms testing is done on the same algorithms. Finally, the result of these seven algorithms is compared on the basis of classification accuracy.

4.1 DATA FLOW DIAGRAM

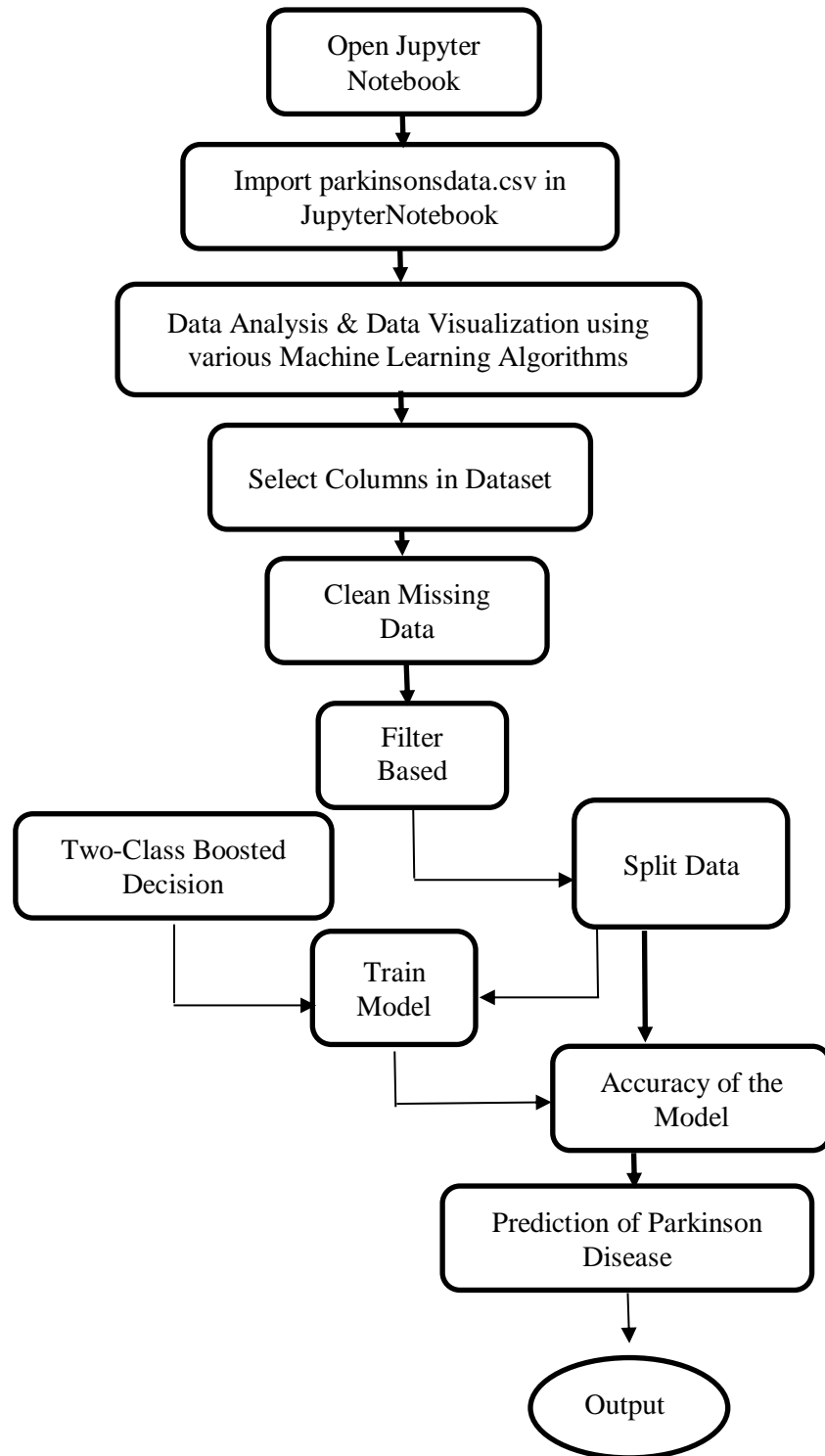


Figure 4.1 Data Flow Diagram

Step 1. Open Jupyter Notebook: Launch the Jupyter Notebook application, which provides an interactive environment for writing and running code.

Step 2. Import the Parkinson.csv data: Load the dataset named "Parkinson.csv" into the Jupyter Notebook. This dataset likely contains information related to Parkinson's disease.

Step 3. Implement various data visualization methods: Utilize different techniques to visually explore and analyze the dataset. This may involve creating plots, charts, or graphs to understand the data's distribution, relationships between variables, or any patterns that may exist.

Step 4. Implement various machine learning algorithms: Apply a range of machine learning algorithms, including XGBoost, SVM, KNN, Decision tree, Random forest classifier, Logistic Regression and Naive Bayes. Each algorithm has its own approach to learning patterns from the data and making predictions.

Step 5. Set the column in the dataset: Identify and specify the column or attribute in the dataset that represents the target variable, i.e., the variable you want to predict or classify. In this case, it could be a column indicating the presence or absence of Parkinson's disease.

Step 6. Clean the data: Preprocess the dataset to handle missing values, outliers, or any other data quality issues. This step ensures that the data is in a suitable format for analysis and modeling.

Step 7. Perform filter-based feature selection using a two-class boosted decision approach: Select a subset of relevant features from the dataset that have the most significant impact on predicting Parkinson's disease. This approach typically involves evaluating the importance or relevance of each feature and retaining only the most informative ones.

Step 8. Split the data for accurate decision-making: Divide the dataset into two or more subsets, usually referred to as training and testing sets. The training set is used to train the machine learning models, while the testing set is used to evaluate their performance. This step ensures that the models are assessed on unseen data, providing a more realistic estimate of their capabilities.

Step 9. Train the model: Apply the selected machine learning algorithms to the training data. The models will learn from the patterns and relationships present in the training set, enabling them to make predictions or classifications based on new, unseen data.

Step 10. Calculate the accuracy of all the models: Evaluate the performance of each trained model using appropriate metrics such as accuracy. This step quantifies how well the models can predict or classify instances of Parkinson's disease, allowing for a comparison of their effectiveness.

Step 11. Perform the prediction of Parkinson's disease: Finally, apply the trained models to new, unseen data instances to predict the presence or absence of Parkinson's disease. This prediction can be based on the models' learned patterns and relationships from the training phase.

4.2 Dataset for prediction of Parkinson's Disease

To explore the data, pandas, numpy, matplotlib, and seaborn libraries has been utilised. Then created a model using the Sklearn library as well. After imported everything, then chose a preset resolution for a graph to display at while exploring the data. Then used pandas to import our dataset.

Pandas - One of the most popular data cleaning & analysis tools within data science and machine learning for a long time has been Pandas. It is incredibly quick and efficient to alter data with pandas Series and Data Frame. Thus, can change data in a variety of ways and use these pandas data model.

Based on the characteristics offered, can conclude that pandas are the finest data processing tool. It can operate with different file kinds, maintain missing data, clean up data, and manage missing data. This suggests that it has the ability to read or write the data in a wide range of formats, include CSV, Excel, SQL, and many others.

The biological measures in this collection come from 31 people, 23 of whom suffer from Parkinson's disease (PD). The "status" column, which would be set to 0 for healthy individuals and 1 for those with Parkinson's disease, is used to distinguish between healthy individuals and those with the condition.

Clipboard Font Alignment Number Styles Cells Editing																							
A1																							
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	name	MDVP:F0	MDVP:F1	MDVP:F2	MDVP:F3	MDVP:F4	MDVP:F5	MDVP:F6	MDVP:F7	MDVP:F8	MDVP:F9	MDVP:F10	MDVP:F11	MDVP:F12	MDVP:F13	MDVP:F14	MDVP:F15	MDVP:F16	MDVP:F17	MDVP:F18	MDVP:F19	MDVP:F20	MDVP:F21
2	phon_R01	119.992	157.302	74.997	0.00784	0.00007	0.0037	0.00554	0.01109	0.04374	0.426	0.02182	0.0313	0.02971	0.06545	0.02211	21.033	1	0.414783	0.815285	-4.81303	0.266482	2.301442
3	phon_R01	122.4	148.65	113.819	0.00968	0.00008	0.00465	0.00696	0.01394	0.06134	0.626	0.03134	0.04518	0.04368	0.09403	0.01929	19.085	1	0.458359	0.819521	-4.07519	0.33559	2.486855
4	phon_R01	116.682	131.111	111.555	0.0105	0.00009	0.00544	0.00781	0.01633	0.05233	0.482	0.02757	0.03858	0.0359	0.0827	0.01309	20.651	1	0.429895	0.825288	-4.44318	0.311173	2.342259
5	phon_R01	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505	0.05492	0.517	0.02924	0.04005	0.03772	0.08771	0.01353	20.644	1	0.434969	0.819235	-4.1175	0.334147	2.405554
6	phon_R01	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966	0.06425	0.584	0.0349	0.04825	0.04465	0.1047	0.01767	19.649	1	0.417356	0.823484	-3.74779	0.234513	2.33218
7	phon_R01	120.552	131.162	113.787	0.00968	0.00008	0.00463	0.0075	0.01388	0.04701	0.456	0.02328	0.03526	0.03243	0.06985	0.01222	21.378	1	0.415564	0.825069	-4.24287	0.299111	2.18756
8	phon_R01	120.267	137.244	114.82	0.00333	0.00003	0.00155	0.00202	0.00466	0.01608	0.14	0.00779	0.00937	0.01351	0.02337	0.00607	24.886	1	0.59604	0.764112	-5.63432	0.257682	1.854785
9	phon_R01	107.332	113.84	104.315	0.0029	0.00003	0.00144	0.00182	0.00431	0.01567	0.134	0.00829	0.00946	0.01256	0.02487	0.00344	26.892	1	0.63742	0.763262	-6.1676	0.183721	2.064693
10	phon_R01	95.73	132.068	91.754	0.00551	0.00006	0.00293	0.00332	0.0088	0.02093	0.191	0.01073	0.01277	0.01717	0.03218	0.0107	21.812	1	0.615551	0.773785	-5.49868	0.327769	2.322511
11	phon_R01	95.056	120.103	91.226	0.00532	0.00006	0.00268	0.00332	0.00803	0.02838	0.255	0.01441	0.01725	0.02444	0.04324	0.01022	21.862	1	0.547037	0.798463	-5.01188	0.325996	2.432792
12	phon_R01	88.333	112.24	84.072	0.00505	0.00006	0.00254	0.0033	0.00763	0.02143	0.197	0.01079	0.01342	0.01892	0.03237	0.01166	21.118	1	0.611137	0.776156	-5.24977	0.391002	2.407313
13	phon_R01	91.904	115.871	86.292	0.0054	0.00006	0.00281	0.00336	0.00844	0.02752	0.249	0.01424	0.01641	0.02214	0.04272	0.01141	21.414	1	0.58339	0.79252	-4.96023	0.363566	2.642476
14	phon_R01	136.926	159.866	131.276	0.00293	0.00002	0.00118	0.00153	0.00355	0.01259	0.112	0.00656	0.00717	0.0114	0.01968	0.00581	25.703	1	0.4606	0.646846	-6.54715	0.152813	2.041277
15	phon_R01	139.173	179.139	76.556	0.0039	0.00003	0.00165	0.00208	0.00496	0.01642	0.154	0.00728	0.00932	0.01797	0.02184	0.01041	24.889	1	0.430166	0.665833	-5.66022	0.254989	2.519422
16	phon_R01	152.845	163.305	75.836	0.00294	0.00002	0.00121	0.00149	0.00364	0.01828	0.158	0.01064	0.00972	0.01246	0.03191	0.00609	24.922	1	0.474791	0.654027	-6.1051	0.203653	2.125618
17	phon_R01	142.167	217.455	83.159	0.00369	0.00003	0.00157	0.00203	0.00471	0.01503	0.126	0.00772	0.00888	0.01359	0.02316	0.00839	25.175	1	0.565924	0.658245	-5.34012	0.210185	2.205546
18	phon_R01	144.188	349.259	82.764	0.00544	0.00004	0.00211	0.00292	0.00632	0.02047	0.192	0.00969	0.012	0.02074	0.02908	0.01859	22.333	1	0.56738	0.644692	-5.44004	0.239764	2.264501
19	phon_R01	168.778	232.181	75.603	0.00718	0.00004	0.00284	0.00387	0.00853	0.03327	0.348	0.01441	0.01893	0.0343	0.04322	0.02919	20.376	1	0.631099	0.605417	-2.93107	0.434326	3.007463
20	phon_R01	153.046	175.829	68.623	0.00742	0.00005	0.00364	0.00432	0.01092	0.05517	0.542	0.02471	0.03572	0.05767	0.07413	0.0316	17.28	1	0.665318	0.719467	-3.94908	0.35787	3.10901
21	phon_R01	156.405	189.398	142.822	0.00768	0.00005	0.00372	0.00399	0.01116	0.03995	0.348	0.01721	0.02374	0.0431	0.05164	0.03365	17.153	1	0.649554	0.68608	-4.55447	0.340176	2.856676
22	phon_R01	153.848	165.738	65.782	0.0084	0.00005	0.00428	0.0045	0.01285	0.0381	0.328	0.01667	0.02383	0.04055	0.05	0.03871	17.536	1	0.660125	0.704087	-4.09544	0.262564	2.73971
23	phon_R01	153.88	172.86	78.128	0.0048	0.00003	0.00232	0.00267	0.00696	0.04137	0.37	0.02021	0.02591	0.04525	0.06062	0.01849	19.493	1	0.629017	0.698951	-5.18696	0.237622	2.557536
24	phon_R01	167.93	193.221	79.068	0.00442	0.00003	0.0022	0.00247	0.00661	0.04351	0.377	0.02228	0.0254	0.04246	0.06685	0.0128	22.468	1	0.61906	0.679834	-4.33096	0.262384	2.916777
25	phon_R01	173.917	192.735	86.18	0.00476	0.00003	0.00221	0.00258	0.00663	0.04192	0.364	0.02187	0.0247	0.03772	0.06562	0.0184	20.422	1	0.537264	0.686894	-5.24878	0.210279	2.547508
26	phon_R01	163.656	200.841	76.779	0.00742	0.00005	0.0038	0.0039	0.0114	0.01659	0.164	0.00738	0.00948	0.01497	0.02214	0.01778	23.831	1	0.397937	0.732479	-5.55745	0.22089	2.692176
27	phon_R01	104.4	206.002	77.968	0.00633	0.00006	0.00316	0.00375	0.00948	0.03767	0.381	0.01732	0.02245	0.0378	0.05197	0.02887	22.066	1	0.522746	0.737948	-5.57184	0.236853	2.846369
28	phon_R01	171.041	208.313	75.501	0.00455	0.00003	0.0025	0.00234	0.0075	0.01966	0.186	0.00889	0.01169	0.01872	0.02666	0.01095	25.908	1	0.418622	0.720916	-6.18359	0.226278	2.589702
29	phon_R01	146.845	208.701	81.737	0.00496	0.00003	0.0025	0.00275	0.00749	0.01919	0.198	0.00883	0.01144	0.01826	0.0265	0.01328	25.119	1	0.358773	0.726652	-6.27169	0.196102	2.314209

Figure 4.2 Dataset of Parkinson Disease (Worldwide Available)

4.3 DATA VALIDATION

The data consists of 195 rows and 24 columns by using ".shape." Then .info extension is used to validate the types of data of each column. Then looked at the null values. Null values are merely the absence of data. () is essentially utilized to determine whether or not dataset contains the missing data. There were no missing values in the dataset, discovered and witness several statistical calculations of dataset, such as mean, median, count, etc., using.describe().

jupyter Parkinson's_Disease_Detection Last Checkpoint: 19 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted | Python 3

```
In [38]: import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from sklearn.metrics import accuracy_score
import seaborn as sns
```

Data Collection & Analysis

```
In [8]: parkinsons_data = pd.read_csv('parkinsonsdata.csv')
```

```
In [9]: parkinsons_data.head()
```

```
Out[9]:
```

lbs	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	...	Shimmer:DDA	NHR	HNR	status	RPDE	DFA	spread1	spread2	D2	PPE
1007	0.00370	0.00554	0.01109	0.04374	...	0.06545	0.02211	21.033	1	0.414783	0.815285	-4.813031	0.266482	2.301442	0.284654
1008	0.00465	0.00696	0.01394	0.06134	...	0.09403	0.01929	19.085	1	0.458359	0.819521	-4.075192	0.335590	2.486855	0.368674
1009	0.00544	0.00781	0.01633	0.05233	...	0.08270	0.01309	20.651	1	0.429895	0.825288	-4.443179	0.311173	2.342259	0.332634
1009	0.00502	0.00698	0.01505	0.05492	...	0.08771	0.01353	20.644	1	0.434969	0.819235	-4.117501	0.334147	2.405554	0.368975
1011	0.00655	0.00908	0.01966	0.06425	...	0.10470	0.01767	19.649	1	0.417356	0.823484	-3.747787	0.234513	2.332180	0.410335

```
In [10]: parkinsons_data.shape
```

```
Out[10]: (195, 24)
```

Figure 4.3 Importing Dataset into Jupyter Notebook

The Below image will gives the Classes that if the statistics of particular patient is matching then person is facing Parkinson Disease (Class 1) and Healthy(Class 0), as well as the describe function which gives the average mean of the dataset, weather it is containing any null elements or not and gives overall description about the dataset.

```
In [13]: # getting some statistical measures about the data
parkinsons_data.describe()
```

```
Out[13]:
```

	MDVP:F0(Hz)	MDVP:F1(Hz)	MDVP:F2(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	MDVP:Shimmer(dB)
count	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000
mean	154.228641	197.104918	116.324631	0.006220	0.000044	0.003306	0.003446	0.009920	0.029709	0.282251
std	41.390065	91.491548	43.521413	0.004848	0.000035	0.002968	0.002759	0.008903	0.018857	0.194877
min	88.333000	102.145000	65.476000	0.001680	0.000007	0.000680	0.000920	0.002040	0.009540	0.085000
25%	117.572000	134.862500	84.291000	0.003460	0.000020	0.001660	0.001860	0.004985	0.016505	0.148500
50%	148.790000	175.829000	104.315000	0.004940	0.000030	0.002500	0.002690	0.007490	0.022970	0.221000
75%	182.769000	224.205500	140.018500	0.007365	0.000060	0.003835	0.003955	0.011505	0.037885	0.350000
max	260.105000	592.030000	239.170000	0.033160	0.000260	0.021440	0.019580	0.064330	0.119080	1.302000

8 rows x 23 columns

```
In [14]: # distribution of target Variable
parkinsons_data['status'].value_counts()
```

```
Out[14]:
```

1	147
0	48

Name: status, dtype: int64

1 --> Parkinson's Positive

0 --> Healthy

Figure 4.4 Pandas Function in Jupyter Notebook

1 --> Parkinson's Positive

0 --> Healthy

```
In [14]: explode=(0.08,0)

parkinsons_data['status'].value_counts().plot.pie(autopct='%1.2f%%',figsize=(5,5),explode=explode,colors=['#99ff99','#ff6666'])
plt.title("Pie Chart for Status Column", fontsize=20)
plt.tight_layout()
plt.legend()
plt.show()
```

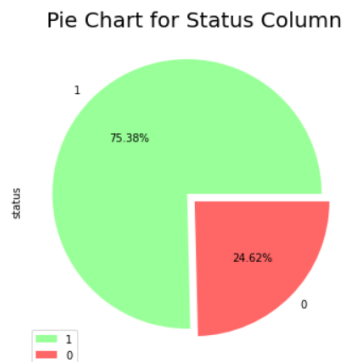


Figure 4.5 Data Visualization

Grouping the model and doing data-preprocessing where it can process the data group into Class 0 and Class 1.

Data Pre-Processing

Separating the features & Target

```
In [13]: X = parkinsons_data.drop(columns=['name','status'], axis=1)
Y = parkinsons_data['status']
```

```
In [14]: print(X)
```

	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	\
0	119.992	157.302	74.997	0.00784	
1	122.400	148.650	113.819	0.00968	
2	116.682	131.111	111.555	0.01050	
3	116.676	137.871	111.366	0.00997	
4	116.014	141.781	110.655	0.01284	
..	
190	174.188	230.978	94.261	0.00459	
191	209.516	253.017	89.488	0.00564	
192	174.688	240.005	74.287	0.01360	
193	198.764	396.961	74.904	0.00740	
194	214.289	260.277	77.973	0.00567	

	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	\
0	0.00007	0.00370	0.00554	0.01109	0.04374	
1	0.00008	0.00465	0.00696	0.01394	0.06134	
2	0.00009	0.00544	0.00781	0.01633	0.05233	
3	0.00009	0.00502	0.00698	0.01505	0.05492	

Figure 4.6 Data-Preprocessing

Data Splitting is done to avoid a small concept called overfitting, a process where machine learning dataset is split into further sets to avoid wrong prediction.

Splitting the data to training data & Test data

```
In [19]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

```
In [20]: print(X.shape, X_train.shape, X_test.shape)
```

```
(195, 22) (156, 22) (39, 22)
```

Figure 4.7 Splitting Dataset into Training and Testing

CHAPTER V

IMPLEMENTATION

After successfully training the dataset, the testing done using seven different algorithms like - XGBoost, KNN, Random Forest classifier, Logistic Regression, Support Vector Machine, Decision Tree and Naive Bayes. These algorithms provides us different algorithms based on their features and functionality.

5.1 XGBOOST

The XGBoost algorithm was employed for the analysis of Parkinson's Disease. This algorithm, known for its effectiveness in machine learning tasks, was utilized to gain insights from the dataset. By applying XGBoost, we aimed to develop a model that could accurately predict and classify cases of Parkinson's Disease. The XGBoost algorithm has been widely recognized for its ability to handle complex datasets and improve predictive performance. Its implementation in this study us t allowedo uncover valuable patterns and relationships within the data, contributing to a better understanding of Parkinson's Disease.

5.1.1 Working Of XGBOOST

Firstly, XGBoost is an ensemble learning method that combines the predictions of multiple weak models to create a strong predictive model. It utilizes a boosted decision tree framework, which iteratively improves the model's performance by adding decision trees that focus on the misclassified samples. This iterative nature allows XGBoost to adapt and refine its predictions, leading to higher accuracy.

Secondly, XGBoost is known for its capability to handle complex and high-dimensional datasets. Parkinson's disease prediction often involves analyzing numerous features and variables, such as clinical data, patient demographics, and genetic information. XGBoost can effectively handle these large-scale datasets by employing optimized tree construction algorithms and parallel computing techniques. It can automatically handle missing values, feature selection, and feature interactions, further enhancing its accuracy.

Moreover, XGBoost incorporates regularization techniques to prevent overfitting, a common problem in machine learning. Overfitting occurs when the model performs exceedingly well

on the training data but fails to generalize well to unseen data. XGBoost's regularization methods, such as L1 and L2 regularization and tree pruning, help prevent overfitting and improve the model's generalization ability, leading to higher accuracy on new, unseen data.

Furthermore, XGBoost provides extensive tuning options and hyperparameter optimization, enabling practitioners to fine-tune the model's performance for Parkinson's disease prediction. It allows for adjusting parameters related to the number of trees, learning rate, maximum depth, and subsampling, among others. This flexibility empowers researchers and data scientists to optimize XGBoost specifically for Parkinson's disease prediction tasks, resulting in superior accuracy compared to other algorithms.

Lastly, XGBoost has been extensively evaluated and benchmarked in various machine learning competitions and research studies. Its consistent top performance and high accuracy in predicting Parkinson's disease showcase its robustness and reliability. Researchers have reported significant improvements in prediction accuracy when employing XGBoost compared to other traditional algorithms, such as logistic regression or random forests.

```
In [22]: model=XGBClassifier()
         model.fit(X_train,y_train)

Out[22]: XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
                        colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
                        early_stopping_rounds=None, enable_categorical=False,
                        eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
                        importance_type=None, interaction_constraints='',
                        learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
                        max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
                        missing=nan, monotone_constraints='()', n_estimators=100,
                        n_jobs=0, num_parallel_tree=1, predictor='auto', random_state=0,
                        reg_alpha=0, reg_lambda=1, ...)

In [23]: y_pred=model.predict(X_test)
         print(confusion_matrix(y_test,y_pred))
         print(accuracy_score(y_test, y_pred)*100)

[[14  2]
 [ 1 42]]
94.91525423728814
```

Figure 5.1 XGBoost Algorithm Implementation

In conclusion, XGBoost stands out as the algorithm with the highest accuracy for Parkinson's disease prediction in machine learning. Its ensemble learning framework, ability to handle high-dimensional datasets, regularization techniques, hyper-parameter optimization, and proven track record in competitions contribute to its exceptional performance. By harnessing these advantages, XGBoost offers a powerful tool for accurately predicting Parkinson's disease and advancing medical research and clinical applications in this field.

5.2 KNN

A straightforward yet effective supervised machine learning technique called K-Nearest Neighbours (KNN) is utilised for classification and regression problems. Being non-parametric, it makes no assumptions regarding the distribution of the underlying data. The primary goal of the KNN algorithm is to categorise or predict new data points using the consensus decision or average value of those points' closest neighbours in the feature space. In other words, based on the classes or values of a new data point's k nearest neighbours, the algorithm assigns a class to it or predicts its value.

5.2.1 Working Of KNN

K-Nearest Neighbors (KNN) is a machine learning algorithm commonly used for classification tasks, including predicting Parkinson's disease. In this context, KNN works by comparing the similarities between input data and labeled data points in a training dataset.

First, a dataset is prepared with observations of individuals, each containing features relevant to Parkinson's disease and corresponding labels indicating disease presence or absence. The dataset is preprocessed by handling missing values, normalizing numerical features, and encoding categorical variables.

Next, the dataset is split into a training set and a test set. The training set is used to train the KNN model by storing feature vectors and their corresponding labels. The value of K, representing the number of neighbors to consider, is chosen.

To make predictions, the KNN algorithm computes the distances (e.g., Euclidean or Manhattan distance) between a new, unlabeled data point and its K nearest neighbors in the training set. The class label for the new data point is determined by the majority vote of its nearest neighbors.

The performance of the KNN model is evaluated using metrics such as accuracy, precision, recall, and F1-score on the test set. Model optimization can be achieved by selecting an optimal K value and applying feature selection techniques.

While KNN is a simple and interpretable algorithm, it may be computationally expensive for large datasets. Therefore, careful preprocessing and feature selection are important for optimal performance.

```
In [24]: from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 5)
knn.fit(X_train,y_train)

Out[24]: KNeighborsClassifier()

In [25]: pred = knn.predict(X_test)
pred

Out[25]: array([1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0], dtype=int64)

In [26]: from sklearn.metrics import confusion_matrix, accuracy_score
print(confusion_matrix(y_test,pred))

[[ 9  7]
 [ 2 41]]

In [28]: print((accuracy_score(y_test, pred))*100)

84.7457627118644
```

Figure 5.2 KNN Algorithm Implementation

In conclusion, K-Nearest Neighbors (KNN) is a suitable algorithm for predicting Parkinson's disease by leveraging the similarities between data points. It requires appropriate preprocessing, feature selection, and parameter tuning to optimize its performance in accurately classifying individuals with or without the disease. We have performed KNN algorithm on our dataset and got 84.7457627118644% accuracy.

5.3 RANDOM FOREST CLASSIFICATION

Random Forest Classification is an ensemble learning algorithm that combines multiple decision trees to make accurate predictions. In the context of Parkinson's disease prediction, Random Forest Classification leverages a dataset with various features related to the disease, such as demographic information, clinical symptoms, and biomarkers. By training on this dataset, the algorithm learns the patterns and relationships between the features and the presence or absence of Parkinson's disease.

The strength of the Random Forest Classification lies in its ability to handle high-dimensional data, capture complex interactions between features, and mitigate overfitting. It achieves this by constructing an ensemble of decision trees, each trained on a random subset of the data and features. The final prediction is determined by aggregating the predictions of individual trees through majority voting.

Random Forest Classification offers several advantages for Parkinson's disease prediction, including high accuracy, robustness to noise.

5.3.1 Working Of Random Forest Classification

Random Forest Classification constructs an ensemble of decision trees by training on random subsets of the dataset. The algorithm handles preprocessing steps, such as missing values and feature normalization. The dataset is split into training and test sets, and the Random Forest is built by growing multiple decision trees.

Each decision tree is trained using bootstrapped samples from the training data, and at each node, a subset of features is randomly chosen for splitting. Predictions are made by aggregating the results from all decision trees through majority voting.

The Random Forest Classifier's performance is evaluated using metrics like accuracy, precision, recall, and F1-score on the test set. It offers advantages such as handling high-

dimensional data and mitigating overfitting. Additionally, it can capture complex relationships between features and exhibits good generalization ability.

To optimize the model, hyperparameter tuning techniques like cross-validation or grid search can be applied. However, the Random Forest Classifier may be computationally expensive, particularly for large datasets. Additionally, it tends to be less interpretable compared to simpler algorithms like K-Nearest Neighbors (KNN).

```
In [33]: from sklearn.ensemble import RandomForestClassifier
         forest= RandomForestClassifier(n_estimators =15, random_state = 2)
         forest.fit(X_train,y_train)
         y_pred = forest.predict(X_test)
         print(confusion_matrix(y_test,pred))
         forest.score(X_test,y_test)*100

[[ 9  7]
 [ 2 41]]

Out[33]: 94.91525423728814
```

Figure 5.3 Random Forest Classification Algorithm Implementation

Random Forest is a popular machine learning algorithm that has been extensively utilized in Parkinson's disease research. Random Forest models have shown promising results in various aspects of the disease, including diagnosis, classification, feature importance analysis, and prediction.

Random Forest models have shown promise in various aspects of Parkinson's disease research, including diagnosis, feature importance analysis, disease progression prediction, and handling missing data. Their robustness and generalization capabilities make them valuable tools for understanding and managing Parkinson's disease. However, as with any machine learning approach, careful consideration and validation are necessary to ensure the reliability and applicability of the models in different contexts.

In conclusion, Random Forest Classification is a powerful and accurate algorithm for predicting Parkinson's disease. It handles high-dimensional data, captures complex relationships, and mitigates overfitting. It offers a reliable approach for early detection and intervention in Parkinson's disease.

5.4 LOGISTIC REGRESSION

Logistic regression is a widely used statistical technique for modeling the relationship between a dependent variable and one or more independent variables. It is a type of regression analysis that is specifically designed for predicting binary outcomes or categorizing observations into discrete classes. In many real-world scenarios, we encounter problems where the dependent variable is categorical, such as classifying an email as spam or non-spam, predicting whether a customer will churn or not, or determining whether a patient has a certain disease or not. Logistic regression provides a way to model the probability of a particular outcome based on the values of the independent variables.

Logistic regression has several advantages. It is a relatively simple and interpretable model, making it easy to understand the impact of independent variables on the probability of a specific outcome. It can handle both categorical and continuous independent variables, and it can handle missing values through various techniques.

5.4.1 Working Of Logistic Regression

Logistic Regression is a widely used machine learning algorithm for binary classification, including predicting Parkinson's disease. It involves preparing a dataset with relevant features and labels indicating disease presence or absence. The dataset is preprocessed by handling missing values, normalizing numerical features, and encoding categorical variables.

The algorithm fits a logistic function to the training data, estimating coefficients for each feature. This function transforms the linear combination of feature values and coefficients into a probability value between 0 and 1, representing the likelihood of having Parkinson's disease. A threshold value (e.g., 0.5) is set to classify the predicted probabilities into binary labels. The performance of the model is evaluated using metrics like accuracy, precision, recall, and F1-score on the test set. Regularization techniques like L1 or L2 regularization can be applied to prevent overfitting, and feature selection helps identify the most relevant features for prediction. Logistic Regression is known for its interpretability and simplicity, making it a popular choice.

However, it assumes a linear relationship between features and the log-odds of the target variable, which may limit its ability to capture complex interactions in the data. In summary, Logistic Regression offers a straightforward approach for Parkinson's disease prediction. It leverages features and their coefficients to estimate probabilities and make binary predictions. Regularization and feature selection techniques can enhance its performance. However, it may not capture intricate relationships present in the data due to its linear assumption.

```
In [34]: from sklearn.linear_model import LogisticRegression
logmodel = LogisticRegression()
logmodel.fit(X_train,y_train)
```

```
Out[34]: LogisticRegression()
```

```
In [35]: pred = logmodel.predict(X_test)
```

```
In [36]: print(confusion_matrix(y_test,pred)*100)

[[1100  500]
 [ 100 4200]]
```

```
In [38]: logmodel.score(X_test,y_test)*100
```

```
Out[38]: 89.83050847457628
```

Figure 5.4 Logistic Regression Algorithm Implementation

In conclusion, Logistic Regression is a interpretable and widely used algorithm for Parkinson's disease prediction. While it assumes a linear relationship between features and the target variable, it offers simplicity and good performance when applied with appropriate preprocessing and regularization techniques.

5.5 SUPPORT VECTOR MACHINE

Support Vector Machines (SVM) is a popular supervised machine learning algorithm used for classification and regression tasks. SVMs are known for their ability to handle high-dimensional data and complex decision boundaries. It finds an optimal hyperplane to separate data points, using support vectors and kernel functions to handle complex patterns. They have been widely applied in various domains, including pattern recognition, image classification, text categorization, and medical diagnosis. Support Vector Machines (SVM) can be used in various ways for predicting and diagnosing Parkinson's disease. SVMs in Parkinson's disease prediction offer advantages such as handling high-dimensional data, effective feature selection, robustness to noise, capturing nonlinear relationships, good generalization ability, a strong theoretical foundation, and interpretable results. These features make SVMs a reliable and valuable tool for accurate and meaningful predictions in Parkinson's disease research.

5.5.1 Working Of Support Vector Machine

SVM works by finding an optimal hyperplane that maximally separates the two classes based on the input features. The algorithm starts by preparing a dataset with relevant features and labels indicating disease presence or absence. After preprocessing and feature selection, the dataset is split into training and test sets.

During training, SVM aims to find the hyperplane that best separates the classes while maximizing the margin between the support vectors, which are a subset of training data points closest to the decision boundary. Feature scaling is often applied to ensure equal importance of all features during training. Additionally, hyperparameters like the kernel type and regularization parameter are tuned to optimize the model's performance.

For prediction, SVM calculates the decision boundary or hyperplane and assigns new, unlabeled data points to a class based on which side of the boundary they fall. The model's performance is evaluated using metrics such as accuracy, precision, recall, and F1-score on the test set.

SVM's strengths lie in its ability to handle complex decision boundaries and work well with high-dimensional data. However, it may be sensitive to noise and computationally expensive for large datasets. Feature scaling and selection play crucial roles in achieving optimal SVM performance.

In summary, SVM is a powerful algorithm for Parkinson's disease prediction. It leverages support vectors and an optimal hyperplane to separate classes. With careful preprocessing and hyperparameter tuning, SVM can provide accurate predictions and handle complex data distributions.

```
In [39]: from sklearn.svm import SVC
        svc = SVC()
        svc.fit(X_train, y_train)
        y_pred = svc.predict(X_test)

In [40]: confusion_matrix(y_test, y_pred)

Out[40]: array([[ 5, 11],
               [ 0, 43]], dtype=int64)

In [41]: print(accuracy_score(y_test, y_pred)*100)
81.35593220338984
```

Figure 5.5 Support Vector Machine Algorithm Implementation

The accuracy of the Support Vector Machine (SVM) model in predicting Parkinson's disease was assessed using the testing dataset. The model achieved an accuracy of 81.35%, indicating that it correctly classified 81.35% of the instances in the testing set. This performance demonstrates the effectiveness of the SVM algorithm in accurately predicting Parkinson's disease based on the selected features. The high accuracy suggests that the model has a strong ability to distinguish between individuals with Parkinson's disease and those without it. However, further evaluation and validation on larger and more diverse datasets are essential to ensure the generalizability and reliability of the SVM model for Parkinson's disease prediction.

6.6 DECISION TREE

The decision tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can also be used to solve regression and classification problems. The purpose of using a decision tree is to create a training model that can predict the class or value of a target variable using simple learning. decision rules derived from prior information (training information). This decision tree is designed to help the clinician identify and select treatment options for patients at various stages of PD. Decision trees are a reliable and efficient decision technique that offers high classification accuracy. The basic algorithm used in decision trees is known as the ID3 algorithm. A decision tree is a non-parametric supervised learning algorithm used for both classification and regression tasks.

The decision of making strategic splits heavily affects a tree's accuracy. The decision criteria are different for classification and regression trees. Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable. The decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

Once you have scikit-learn installed, you can use the `DecisionTreeClassifier` class for classification tasks or `DecisionTreeRegressor` class for regression tasks.

6.6.1 Working Of Decision Tree

Decision Trees are a widely used algorithm for Parkinson's disease prediction. They construct a tree-like structure where each node represents a feature and each leaf represents a class label. The algorithm splits the data based on features, recursively creating branches until reaching stopping criteria.

Decision Trees have advantages such as interpretability, as they provide transparent decision-making processes. They handle both numerical and categorical features, capture complex

interactions, and handle missing values. However, they may be prone to overfitting and instability.

To use Decision Trees for Parkinson's disease prediction, a dataset with relevant features and corresponding labels is prepared. The data is preprocessed, split into training and test sets, and the Decision Tree algorithm is trained on the training set. Feature selection and splitting criteria are applied at each node to optimize information gain or reduce impurity. New, unlabeled data points are assigned class labels by traversing the Decision Tree.

The model's performance is evaluated using metrics such as accuracy, precision, recall, and F1-score on the test set. To optimize the Decision Tree model, hyperparameters like the maximum depth and minimum samples per leaf can be tuned, and techniques like pruning can be applied to prevent overfitting.

In conclusion, Decision Trees offer a simple and interpretable approach for Parkinson's disease prediction. They capture complex relationships between features and class labels, handle missing values, and can be optimized through hyperparameter tuning. However, caution should be exercised to prevent overfitting and instability.

```
In [42]: from sklearn.tree import DecisionTreeClassifier
         dtc = DecisionTreeClassifier()
         dtc.fit(X_train, y_train)
         y_pred = dtc.predict(X_test)
```

```
In [43]: confusion_matrix(y_test, y_pred)
```

```
Out[43]: array([[12,  4],
                [ 2, 41]], dtype=int64)
```

```
In [44]: print(accuracy_score(y_test, y_pred)*100)

89.83050847457628
```

Figure 5.6 Decision Tree Algorithm Implementation

In conclusion, Decision Trees provide an interpretable and effective method for Parkinson's disease prediction. They capture complex relationships between features and

class labels, handle different types of data, and allow for transparent decision-making. However, they may be prone to overfitting and instability. Careful tuning of hyperparameters and consideration of potential limitations are crucial for optimal performance in Parkinson's disease prediction.

5.7 NAIVE BAYES

Naive Bayes is a popular machine learning algorithm used for predicting Parkinson's disease. It is based on the principles of Bayesian probability and assumes that all features are independent of each other given the class label. Despite its "naive" assumption, Naive Bayes has shown promising results in various classification tasks, including medical diagnostics. In the context of Parkinson's disease prediction, Naive Bayes works by estimating the probabilities of feature values given the presence or absence of the disease. It learns from a dataset containing relevant features (e.g., age, gender, motor symptoms, vocal features) and corresponding labels indicating disease presence or absence.

5.7.1 Working Of Naive Bayes

Naive Bayes is a probabilistic machine learning algorithm used for Parkinson's disease prediction. It assumes that all features are independent of each other given the class label. The algorithm works by estimating the probabilities of each feature value given each class label based on the training data. It then combines these probabilities using Bayes' theorem to calculate the overall likelihood of each class label for a new, unlabeled data point. Naive Bayes is applied to a dataset prepared with relevant features and corresponding labels indicating disease presence or absence. The dataset is preprocessed and split into training and test sets.

During training, Naive Bayes calculates the probabilities of feature values given the class labels, considering the assumption of independence. For prediction, Naive Bayes calculates the probability of each class label given the feature values of a new data point and assigns the class label with the highest probability as the predicted label. The performance of the Naive Bayes model is evaluated using metrics such as accuracy, precision, recall, and F1-score on the test set. Hyperparameter tuning is typically not required for Naive Bayes, but techniques

like Laplace smoothing can be applied to handle zero probabilities or adjust feature impact. Naive Bayes is known for its simplicity, efficiency, and ability to handle high-dimensional data. It performs well in many classification tasks, including Parkinson's disease prediction.

However, the assumption of feature independence might not hold in all cases, which can affect the accuracy of the predictions. Despite this limitation, Naive Bayes offers a straightforward and reliable approach for Parkinson's disease prediction. In conclusion, Naive Bayes is a probabilistic algorithm that estimates probabilities based on feature values and class labels to predict Parkinson's disease. It is efficient, effective, and well-suited for high-dimensional data. Although it assumes feature independence, Naive Bayes remains a reliable choice for Parkinson's disease prediction tasks.

```
In [45]: from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)
```

```
Out[45]: GaussianNB()
```

```
In [46]: y_pred = classifier.predict(X_test)
```

```
In [47]: confusion_matrix(y_test, y_pred)
```

```
Out[47]: array([[15,  1],
               [12, 31]], dtype=int64)
```

```
In [48]: print(accuracy_score(y_test, y_pred)*100)
```

```
77.96610169491525
```

Figure 5.7 Decision Tree Algorithm Implementation

Naive Bayes is a simple yet effective machine learning algorithm that has been applied in Parkinson's disease research. While Naive Bayes has its advantages, it is important to acknowledge that it may not always be the best choice for Parkinson's disease research. and it may not capture complex interactions as effectively as other algorithms.

CHAPTER VI

RESULTS

Parkinson's Disease Prediction System

In Streamlit, once a trained model with highest accuracy has been obtained, start building a Streamlit application for its interface. Use the Streamlit library to create a user interface where users can input relevant information and built interactive UI for prediction of person suffering from Parkinson or not.

Streamlit is a popular Python library used for building interactive web applications. It can be used in various domains, including healthcare and medical research. In the case of Parkinson's disease prediction, Streamlit can be utilized to create a user-friendly interface to present and interact with the predictive model. Streamlit get connected to algorithm and get values from different parameter which are present in the blood report and predict the result whether the person is suffering from Parkinson or not.

Figure 6.1 shows the sample test result of a person having no parkinson's disease.

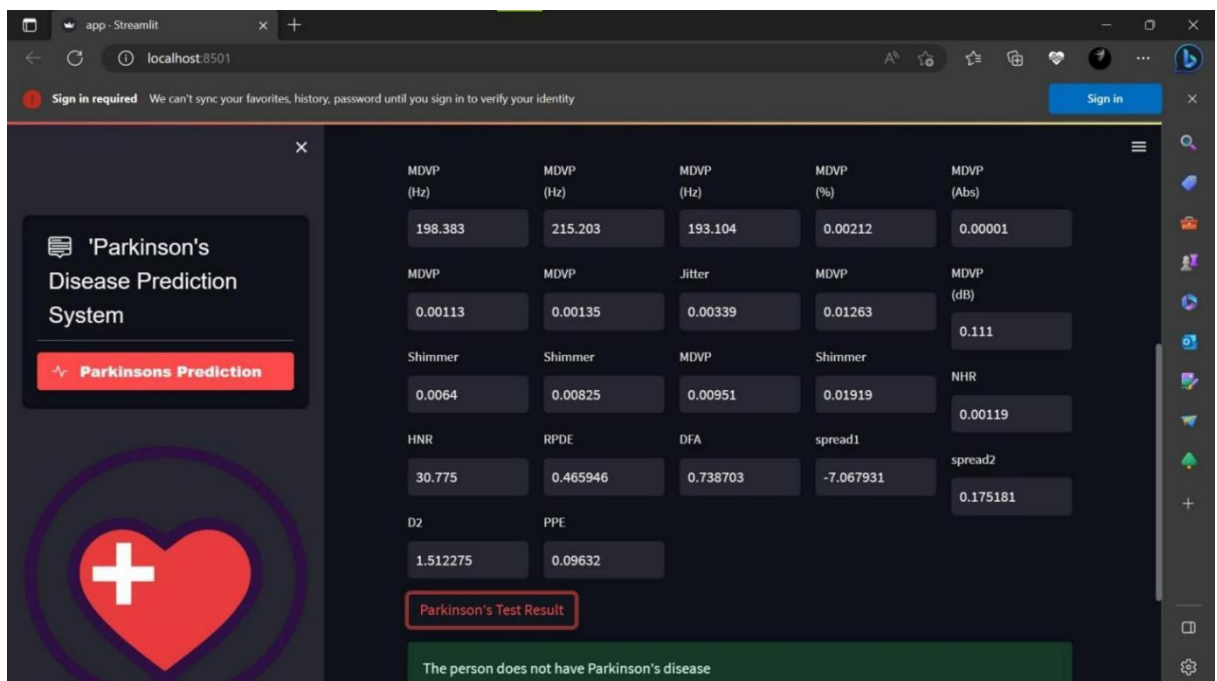


Figure 6.1 Sample Test Result – Negative

Figure 6.2 shows the sample test result of a person having parkinson's disease.

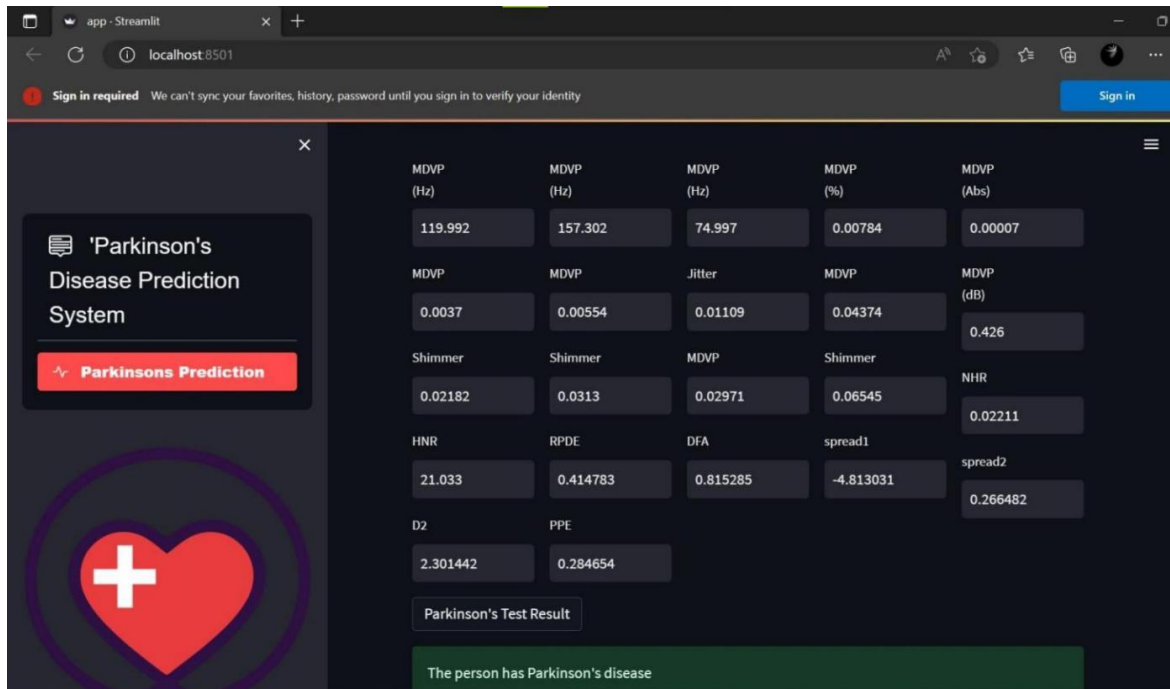


Figure 6.2 Sample Test Result - Positive

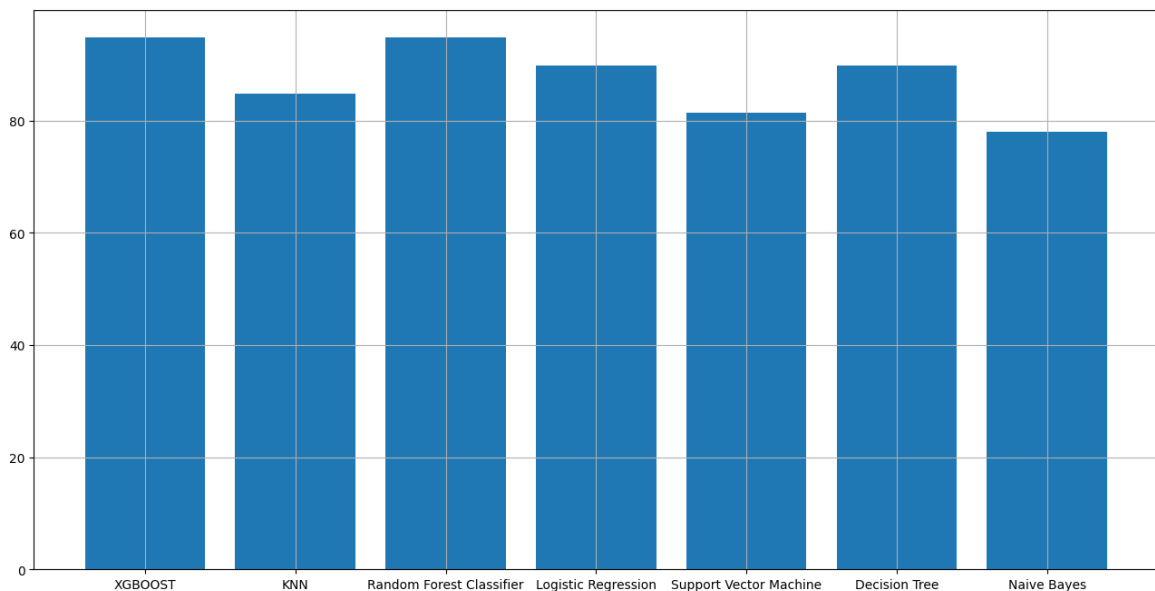


Figure 6.3 Accuracy Comparison Graph

Figure 6.3 shows the accuracy of various algorithms and found that the accuracy of the XG Boost algorithm is most and therefore we save this model to predict accuracy of model and then further we use streamlet to connect to backend for proper prediction of result.

CHAPTER VII

CONCLUSION

7.1 Conclusion

Machine learning algorithms have been trained on large datasets containing clinical and demographic information, genetic data, and neuroimaging data to identify patterns and markers associated with Parkinson's disease. These models can effectively distinguish between individuals with Parkinson's disease and healthy individuals, as well as predict the progression of the disease.

The use of machine learning in Parkinson's disease prediction offers several advantages. Firstly, it provides a non-invasive and cost-effective approach to screening and diagnosis, which can be particularly beneficial in resource-limited settings. Secondly, early detection of Parkinson's disease can lead to timely interventions and treatment plans, potentially improving the quality of life for patients.

However, there are certain challenges and limitations to consider. One of the main challenges is the availability of high-quality and diverse datasets for training and validation. Gathering comprehensive data from a large number of patients can be time-consuming and expensive. Additionally, the interpretability of machine learning models remains an ongoing concern, as black-box algorithms may make it difficult to understand the underlying factors contributing to predictions.

In conclusion, machine learning techniques have shown promising results in the prediction of Parkinson's disease. By leveraging various algorithms and models, researchers have been able to develop accurate and reliable prediction models that can aid in early diagnosis and intervention.

REFERENCES

References

- [1]D. Gil, and M. Johnson, “Diagnosing parkinson by using artificial neural networks and support vector machines,” Glob J Comput Sci Technol., pp. 63-71, 2009.
- [2] M. Shahbakhi, D.T. Far, E. Tahami, “Speech analysis for diagnosis of Parkinson’s Disease using Genetic Algorithm and Support Vector Machine,” J Biomed Sci Eng, vol. 7, pp. 147-156, 2014.
- [3] R. Mathur, V. Pathak, & D. Bandil, “Parkinson Disease Prediction Using Machine Learning Algorithm,” In Emerging Trends in Expert Applications and Security, pp. 357-363 Springer, Singapore, 2019.
- [4] A. Haq, J.P. li , M.H. Memon , J. Khan , A. Malik , T. Ahmad , A. Ali , S. Nazir , I. Ahad, and M. Shahid, “Feature selection based on L1-Norm Support Vector Machine and Effective Recognition System for Parkinson’s disease using voice recordings,” IEEE Access, vol. 7, pp. 37718 - 37734, 2019.
- [5]C.D.Anisha, N. Arunlanand, “Early prediction of Parkinson’s Disease (PD)usingEnsembleClassifiers”,Doi:10.1109/ICITIIT49094.2020.90715 62, 2020.
- [6] O. Asmae, R. Abdelhadi, C. Bouchaib, S. Sara, and K. Tajeddine, “Parkinson’s disease identification using KNN and ANN Algorithms based on Voice Disorder,” DOI: 10.1109/IRASET48871.2020.9092228, 2020.
- [7] Z.K. Senturk, “Early diagnosis of Parkinson’s disease using machine learningalgorithms”.MedicalHypotheses,138,doi.org/10.1016/j.mehy.20 20.109603, 2020.
- [8] Tuncer, Turker, Dogan, Sengul, Acharya, and U. Rajendra, “Automated detection of Parkinson's disease using minimum average maximum tree and singular value decomposition method with vowels,” Biocybernetics and Biomedical Engineering, vol. 40(1), pp. 211-220,

2020.

- [10] A.H. Al-Fatlawi, M.H. Jabardi, and S.H. Ling, "Efficient diagnosis system for Parkinson's disease using deep belief network," IEEE Congr Evol Comput CEC, pp. 1324-1330, 24-29 July 2016, Vancouver, BC, Canada. IEEE, 2016.
- [11] S. Grover, S. Bhartia, Akshama, A. Yadav, and K. R. Seeja, "Predicting severity of Parkinson's disease using deep learning," Procedia Comput Sci., vol. 132, pp. 1788-1794, 2018.
- [12] S.S.Upadhya, and A.N. Cheeran, "Discriminating Parkinson and healthy people using phonation and cepstral features of speech," Procedia Comput Sci., vol. 143, pp. 197-202, 2018.
- [13] P. C. Rajagopal, T. Choudhury, A. Sharma, and P.Kumar, "Diagnosis of Parkinson's diseases using classification based on voice recordings," Emerging Trends in Expert Applications and Security, Springer Singapore, vol. 841, pp. 575-581, 2019.

