PROJECT REPORT

On

PARKINSON'S DISEASE DETECTION USING MACHINE LEARNING

Mudit Shanker Saxena

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING SRMCEM

CERTIFICATE

Certified that the project entitled **"PARKINSON'S DISEASE DETECTION USING MACHINE LEARNING "** Submitted by MUDIT SHANKER SAXENA record of students" own work carried under our supervision and guidance. The project report embodies the results of original work and studies carried out by students and the contents do not form the basis for the award of any other degree to the candidate or to anybody else.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SRMCEM

DECLARATION

I hereby declare that the project entitled **"PARKINSON'S DISEASE DETECTION USING MACHINE LEARNING"** submitted for the award of any degree.

ACKNOWLEDGEMENT

In the sense of great pleasure and satisfaction, we present this project entitled **“Parkinson’s Disease Detection Using Machine Learning”**.

The success of any project depends on the collective efforts of the numerous hands that have rendered their support in several ways. We hereby appreciate and extend our vote of thanks to the individuals who provided us with their support, creative ideas, and valuable guidance in making this work a success.

A project is more than a task; it is a journey that involves continuous efforts of not just the one who develops it but also each and every hand who was involved in the project even once.

We take this opportunity to express our profound gratitude and deep regards to our guide for exemplary guidance, monitoring, and constant encouragement throughout the course of this project. The blessing, help, and guidance given by her from time to time shall carry us a long way in the journey of life on which we are about to embark. She gave us the complete freedom to work on this project and embellished our raw ideas to the best of her knowledge. We are beholden to her for giving her valuable time during this project. We would like to thank her for enabling us to view the project from all engineering aspects through her valuable advice.

We would also like to express our heartfelt thanks to our project Coordinators for providing their advice and suggestions during the progress of the project. Their questions during the presentations have helped in gaining deep knowledge about the project.

PREFACE

This report document covers all the aspects of designing Machine Learning Model. The project report has been divided into 6 chapters. The topics covered under each are as follows:

- **Chapter 1: Introduction**

  This Chapter explores the problem definition and purpose of the project, as well as its goals. It also delves into the System Analysis, providing technical, financial, and operational insights into the feasibility of the project.

- **Chapter 2**: **Literature Survey**

  This chapter concludes the literature study by offering a clear understanding of the limitations and advantages of several research reviews relevant to the project. It also sheds light on the noteworthy observations made from previous studies.

- **Chapter 3: Proposed Methodology**

  This chapter presents an overview of the project, including its goals and objectives, the model employed, and the hardware and software requirements. The module structure of the project is described, as well as its database implementation and data flow diagram.

- **Chapter 4: Result analysis and Discussion**

  This chapter focuses on the testing aspect of the project, including the input and output parameters, test data, and validation checks. It provides an in-depth examination of the testing procedures used to validate the functionality and efficiency of the system. It highlights the advantages, applications, and limitations of the project.

- **Chapter 5: Conclusion**

  This part gathers all the inferences derived from the project work. This chapter also covers a brief overview of the knowledge acquired through the project.

- **Chapter 6: Future Scope of the Project**

  This chapter provides a roadmap for future research in smart assistive devices leveraging computer vision technology. It outlines potential.

ABSTRACT

Parkinson's Disease (PD) is a progressive neurodegenerative disorder characterized by motor symptoms such as tremor, bradykinesia, rigidity, and postural instability. Early diagnosis and continuous monitoring of PD are crucial for effective management and improving patient quality of life. Traditional diagnostic methods rely heavily on clinical assessments, which can be subjective and inconsistent. Therefore, there is a growing need for objective, non-invasive, and reliable diagnostic tools. This project proposes a machine learning-based approach to detect and monitor Parkinson's Disease using spiral/wave drawings and voice input data. The proposed methodology is structured into several phases: data collection, preprocessing, feature extraction, model training and testing, and validation. In the data collection phase, spiral and wave drawings are gathered from both PD patients and healthy controls using standardized digital tablets or high-resolution scanned paper drawings. Voice inputs are recorded during various speech tasks, including sustained phonation, reading passages, and spontaneous speech, using high-quality microphones in controlled environments to minimize background noise. Preprocessing of the collected data involves noise reduction and normalization. For the drawings, this includes filtering to remove scanning artifacts and standardizing the size and intensity. Voice data preprocessing involves applying noise reduction algorithms and segmenting the recordings into consistent time frames or phonetic units. Feature extraction is performed to identify key characteristics in the data. From the spiral and wave drawings, geometric features (such as line smoothness and curvature), kinematic features (such as speed and pressure variations), and shape descriptors (using Fourier transforms and fractal dimensions) are extracted. For the voice input data, acoustic features (such as pitch, jitter, shimmer, and formant frequencies), temporal features (such as speech rate and rhythm), and prosodic features (such as intonation patterns) are analyzed. The model training and testing phase involves selecting appropriate machine learning algorithms, including Support Vector Machines (SVM), Random Forests, K Nearest Neighbors (KNN), and Long Short-Term Memory (LSTM) networks. Data is split into training and validation sets, and techniques such as cross-validation and hyperparameter tuning are employed to optimize model performance.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1.1 Introduction to Parkinson's Disease

Parkinson's disease is a progressive neurological disorder characterized by the degeneration of dopamine-producing neurons in the brain, leading to symptoms such as tremors, rigidity, bradykinesia (slowness of movement), and postural instability. Typically manifesting in individuals over the age of 60, Parkinson's disease affects both motor and non-motor functions, including speech, mood, and cognition. The exact cause of Parkinson's is unknown, though it is believed to involve a combination of genetic and environmental factors. Diagnosis is primarily clinical, based on medical history and neurological examination, as there are no definitive tests for the disease. While there is no cure, treatments such as medication, lifestyle changes, and in some cases, surgical interventions like deep brain stimulation, can manage symptoms and improve quality of life. Ongoing research aims to better understand the disease mechanisms and develop more effective therapies.

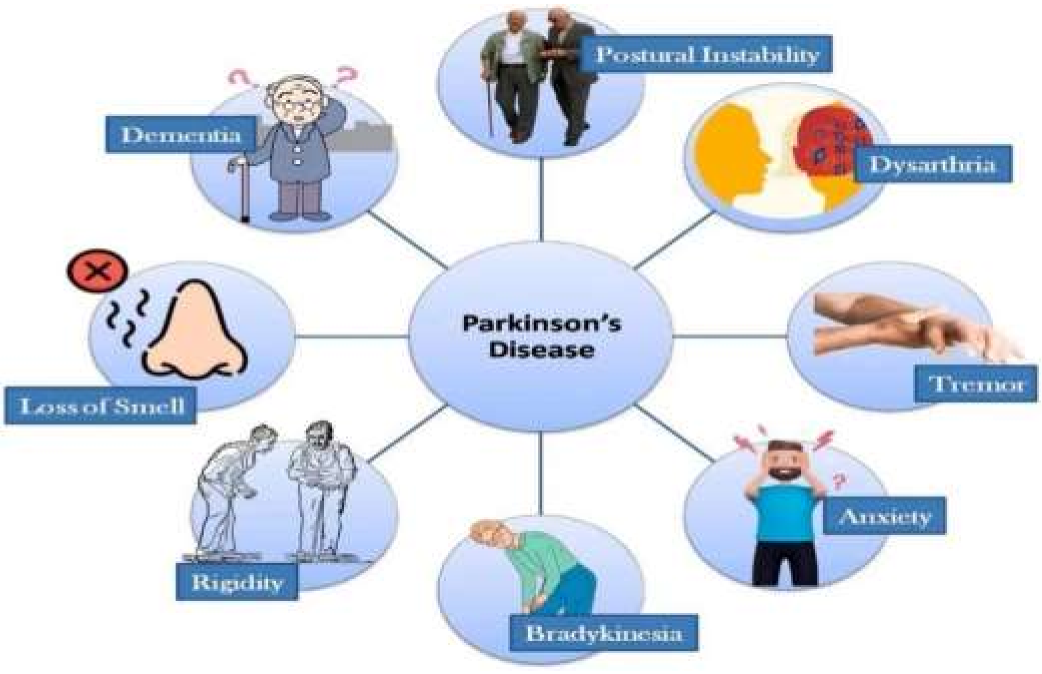


Fig. 1.1 Symptoms in Parkinson's Disease Person

1.2 Introduction to Machine Learning Technology

Machine learning technology refers to a branch of artificial intelligence (AI) that enables systems to learn from data, identify patterns, and make decisions with minimal human intervention. This technology encompasses both hardware and software designed to analyze large volumes of data and improve the performance of various tasks, enhancing the overall quality of life and operational efficiency. Machine learning is increasingly being integrated into various domains, from healthcare and finance to transportation and entertainment, demonstrating its versatility and transformative potential.

1.2.1 The Role of Machine Learning

Machine learning aims to maintain and even enhance a system's functional competence by learning from historical data and making predictions or decisions. It plays a crucial role in automating complex processes, optimizing performance, and providing insights that would be difficult or impossible to obtain through traditional analytical methods. As machine learning models process vast amounts of data, they uncover hidden patterns and correlations, enabling more informed and accurate decision-making.

1.2.2 Key Components of Machine Learning Systems

A typical machine learning system consists of several key components, each contributing to the overall functionality and efficacy of the technology:

- *Data Collection*: Sensors and data sources gather raw data from various environments. This data can include images, audio recordings, text, and numerical data.
- *Data Preprocessing*: Before analysis, the collected data undergoes preprocessing to clean and normalize it, ensuring consistency and quality. This step involves handling missing values, removing noise, and transforming data into a suitable format.
- *Feature Extraction*: Relevant features are extracted from the preprocessed data to serve as input for machine learning models. Features are specific attributes or characteristics that are used to make predictions.

- *Model Training*: Machine learning algorithms are used to train models on the preprocessed data. This involves feeding the features into the algorithms, which then learn to make predictions or decisions based on patterns in the data.
- *Model Evaluation*: The performance of trained models is evaluated using test data. Metrics such as accuracy, precision, recall, and F1-score help assess the model's effectiveness.
- *Deployment and Monitoring*: Once validated, the model is deployed to make predictions on new data. Continuous monitoring ensures the model remains accurate and effective over time, with updates and retraining as needed.

1.2.3 Applications of Machine Learning

Machine learning technology has a wide range of applications, demonstrating its adaptability and impact across various fields:

- *Healthcare*: Machine learning is used for diagnosing diseases, predicting patient outcomes, and personalizing treatment plans. For instance, it can analyze medical images to detect anomalies or predict disease progression based on patient records.
- *Finance*: In finance, machine learning models analyze market trends, detect fraudulent activities, and automate trading strategies, contributing to more secure and efficient financial operations.
- *Transportation*: Autonomous vehicles and traffic management systems rely on machine learning to interpret sensor data, optimize routes, and enhance safety.
- *Retail*: Personalized recommendations, demand forecasting, and inventory management are powered by machine learning algorithms, improving customer experience and operational efficiency.

- *Entertainment*: Streaming services use machine learning to recommend content based on user preferences and viewing history, enhancing user engagement and satisfaction.

1.2.4 Machine Learning Categories

Machine learning approaches can be broadly categorized into three types:

- *Supervised Learning*: Involves training models on labeled data, where the input-output pairs are known. The model learns to map inputs to outputs based on the examples provided.
- *Unsupervised Learning*: Deals with unlabeled data, where the model identifies patterns and relationships without predefined labels. Clustering and dimensionality reduction are common techniques in this category.
- *Reinforcement Learning*: Involves training models to make a sequence of decisions by interacting with an environment. The model receives feedback in the form of rewards or penalties, learning to optimize its actions over time.

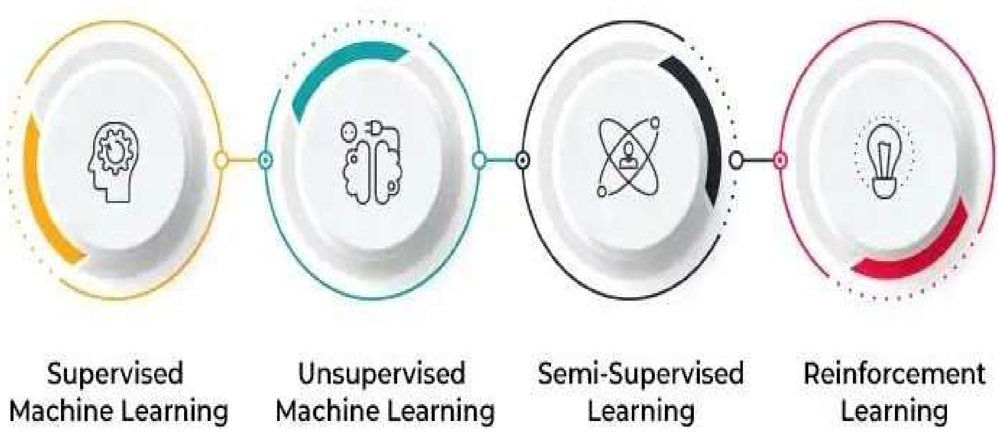


Fig. 1.2 Types of Machine Learning

1.2.5 Steps for Training and Testing of Machine Learning Model

This project explores a combined approach using spiral/wave drawings and voice data for potential Parkinson's Disease (PD) detection. Here's a breakdown of the key steps for a user-friendly system:

1. Data Acquisition: The user interacts with the system in two ways:

- *Drawing*: Users complete standardized spiral and wave drawings on a digital tablet or upload scanned images.
- *Voice Input*: Users read a provided passage aloud, captured by the system's microphone.

2. Pre-processing:

- *Drawings*: Conversion to grayscale and resizing occur for consistency.
- *Voice:* Background noise removal, normalization of audio volume, and conversion to a suitable format for analysis might be necessary.

3. Feature Extraction:

- *Drawings*: Features like line smoothness, spiral shape deviation, and completion rates are extracted.
- *Voice*: Speech rate, pitch variations, and MFCCs are extracted from the pre-processed audio.

4. Classifier and Classification: Separate models are employed:

- *KNN for Drawings*: A pre-trained KNN analyzes the extracted drawing features and classifies it as potentially healthy or PD-related.
- *Machine Learning Model for Voice*: A model like an SVM or Random Forest analyzes the voice features, classifying it as potentially healthy or PD-related.

5. Post-processing and User Output:

- The system combines the classifications from both drawings and voice data.
- This combined information is used by a final model to determine the overall risk of PD.

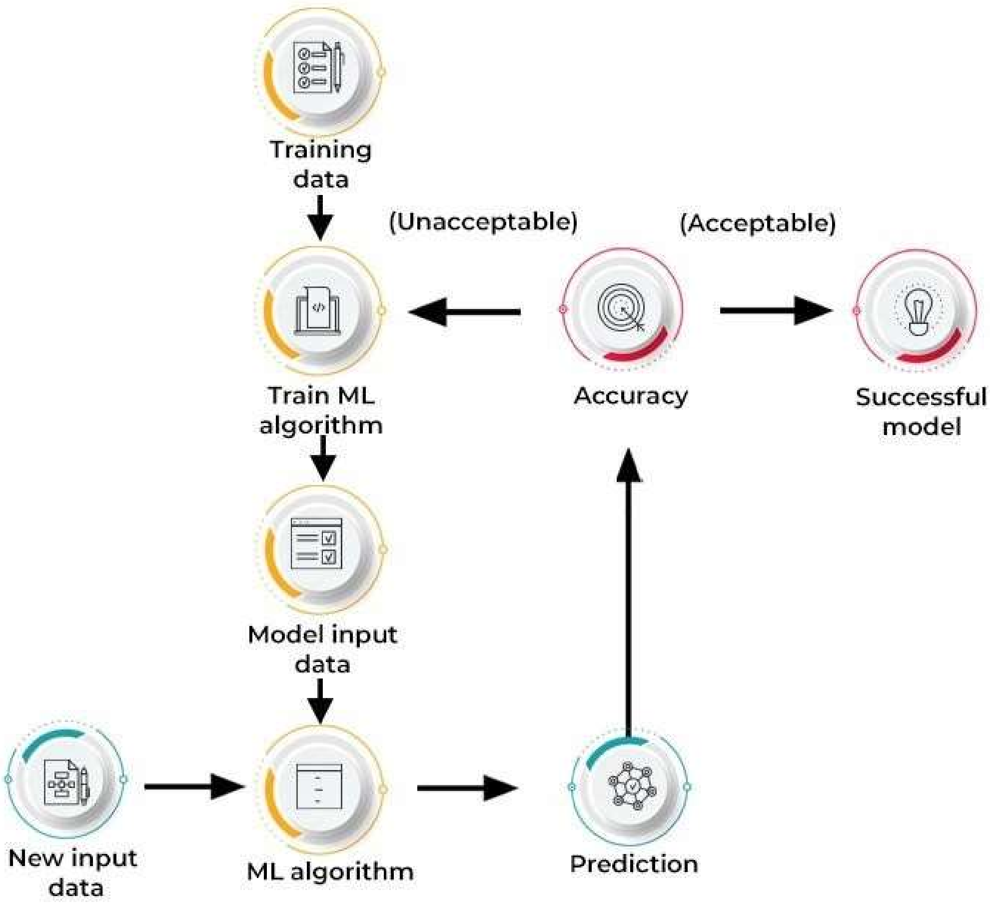


Fig. 1.3 Machine Learning Model Working

1.3 Libraries

1.3.1 Image Processing and Drawing Analysis:

- *OpenCV (Python/C++):* A popular open-source library for computer vision tasks. It offers image processing functions like grayscale conversion, resizing, and feature extraction.
- *Scikit-image (Python):* Another powerful library for image processing, providing advanced functionalities for feature extraction and manipulation.
- *TensorFlow/PyTorch (Python):* Deep learning frameworks that can be used to build and train K Nearest Neighbor(KNNs) for image classification. These

- frameworks allow you to design custom KNN architectures to analyze the spiral/wave drawings and extract relevant features for PD detection.

1.3.2 Audio Processing and Voice Analysis:

- *Librosa (Python):* A library specifically designed for audio and music analysis. It provides functions for audio loading, pre-processing (noise removal, normalization), and feature extraction like MFCCs.
- *SciPy (Python):* A scientific computing library that offers functionalities for audio signal processing and analysis, complementing Librosa.

1.3.3 Machine Learning and Classification:

- *Scikit-learn (Python):* A versatile library for machine learning tasks, including classification algorithms like Support Vector Machines (SVMs) and Random Forests. These models can be used to analyze the extracted features from voice data and classify them as potentially healthy or PD-related.
- *TensorFlow/PyTorch (Python):* As mentioned earlier, these deep learning frameworks can also be used to build custom machine learning models for voice classification.

1.3.4 Additional Considerations:

- *Keras (Python):* A high-level neural network API that can simplify the development process within TensorFlow or PyTorch.
- *NumPy (Python):* A fundamental library for numerical computing, essential for working with image and audio data in Python.

1.4 About the Project – An Overview

This project tackles Parkinson's Disease (PD) detection with a novel approach, combining user-generated spiral/wave drawings and voice data analysis. By leveraging these complementary modalities, the project aims to develop a more robust screening tool for PD. The rationale lies in PD affecting both movement and speech patterns. Analyzing both drawings, for deviations suggesting motor control issues, and voice recordings, for changes in speech rate or pitch variations potentially linked to PD, allows for a broader range of potential indicators to be captured. Users would provide drawings (digital or scanned) and voice input (reading a passage), which would then undergo pre-processing like noise reduction and formatting. Key features like line smoothness, spiral deviations, completion rates (drawings) and speech rate, pitch variations, and MFCCs (voice) would be extracted. Separate models, a K Nearest Neighbor (KNN) for drawings and a machine learning model (SVM/Random

Forest) for voice would analyze these features and classify them as potentially healthy or PD-related. Finally, the system would combine the classifications and use a final model to determine the overall PD risk (high, moderate, low). It's crucial to emphasize that this result is for informational purposes only and should not be taken as a medical diagnosis.

1.4.1 Need and Feasibility

Early detection of Parkinson's Disease (PD) is crucial for better disease management. This project explores the feasibility of using a combination of spiral/wave drawings and voice data analysis for a user-friendly and potentially more accurate screening tool. Current methods often rely on physical examinations, which can be subjective. By analyzing drawings for features like shaky lines or deviations from a perfect spiral, which might indicate motor control issues, and voice recordings for changes in speech

rate or pitch variations potentially linked to PD, this project aims to capture a broader range of potential indicators. This could offer a more objective and accessible screening tool, especially in areas with limited access to specialists. The use of established techniques like Convolutional Neural Networks (CNNs) for image analysis and machine learning models for voice data make this approach feasible. While the project wouldn't replace a formal diagnosis, it could provide valuable preliminary information and encourage earlier medical evaluation.

1.4.2 Operational Feasibility

The Parkinson's Disease (PD) detection project using spiral/wave drawings and voice data shows operational feasibility due to several factors:

- *Data Collection:* Users can easily submit data - digital drawings (tablet or scanned) and voice recordings through a microphone - making it accessible.
- *Existing Technologies:* The project leverages established tools like K Nearest Neighbor (KNNs) for image analysis and machine learning models

(SVMs/Random Forests) for voice data, readily available in open-source libraries.

- *Computational Power:* While CNNs require some computing power, advancements in cloud technology and efficient hardware make them increasingly accessible.
- *User-Friendliness*: The data collection process is simple, requiring users to only provide drawings and a voice sample, making it user-friendly for various technical backgrounds.
- *Scalability*: The system has the potential to handle a larger user base. The underlying machine learning models can continuously improve with more data, allowing the system to scale.

CHAPTER 2: LITERATURE SURVEY

2.1 Parkinson's Disease Detection Major Project Report (2023)

This report by Joga et al. (2023) presents a comprehensive overview of PD detection using ML techniques. It outlines the challenges associated with PD diagnosis and the potential of ML to overcome these limitations. The report discusses various ML algorithms employed for PD detection, including SVMs, random forests, decision trees, k-nearest neighbors, and multilayer perceptions. It also emphasizes the importance of feature selection and hyperparameter tuning for optimizing ML model performance.

2.2 Parkinson's Disease Detection Using Machine Learning Techniques: A Review of the Literature (2023)

This review by Uddin et al. (2023) provides a comprehensive analysis of ML techniques used for PD detection. It highlights the growing interest in ML-based PD diagnosis due to its ability to handle complex data and identify subtle patterns. The review discusses various ML algorithms employed for PD detection, including SVMs, random forests, artificial neural networks, and deep learning techniques. It also emphasizes the importance of data quality, feature selection, and model evaluation for successful ML-based PD diagnosis.

2.3 A Review of Machine Learning for Parkinson's Disease Diagnosis and Progression Monitoring Using Wearable Sensors (2023)

This review by Abadi et al. (2023) focuses on the application of wearable sensors and ML techniques for PD diagnosis and progression monitoring. It highlights the potential of wearable sensors to continuously collect data on motor symptoms, such as gait, tremor, and posture, which can be analyzed using ML algorithms to detect and monitor PD progression. The review discusses various ML algorithms applied to wearable sensor data for PD.

Diagnosis, including SVMs, random forests, and deep learning techniques. It also emphasizes the importance of sensor selection, data preprocessing, and model validation for effective ML-based PD diagnosis and monitoring.

2.4 Machine Learning Approaches to Identify Parkinson's Disease Using Voice Signal Features (2022)

This review by Mehri et al. (2022) specifically focuses on the utilization of voice signal features for PD detection using ML approaches. It emphasizes the promise of voice analysis as a non-invasive and cost- effective diagnostic tool for PD. The review elaborates on various voice features extracted from speech recordings, such as fundamental frequency, jitter, shimmer, and speech intensity. It also explores different ML algorithms applied to voice signal features for PD classification, comparing their performance and effectiveness.

2.5 Hybrid Models for PD Detection Using Kinematic Features (2022)

Rosenblum et al. (2022) conducted a study focusing on enhancing Parkinson's Disease (PD) detection by combining spiral drawing analysis with kinematic features through hybrid models. Traditional PD diagnostic methods often rely on subjective clinical evaluations, necessitating the development of objective, quantifiable tools. The study involved a diverse cohort of PD patients and healthy controls who performed standardized drawing tasks using digital tablets. The primary task was spiral drawing, supplemented by line drawings and geometric shapes to capture various motor control aspects.

2.6 Validation of Combined Approaches in Clinical Settings (2022)

Chen et al. (2022) conducted a study to validate combined approaches for Parkinson's Disease (PD) detection in clinical settings. The research aimed to integrate spiral drawing analysis with kinematic and voice features to enhance diagnostic accuracy. Participants, including PD patients and healthy controls, performed standardized drawing tasks on digital tablets and provided voice samples. The study extracted features such as drawing speed, pressure, and tremor intensity from the drawings, while kinematic features included velocity, acceleration, and jerk. Voice features focused on acoustic properties like pitch, loudness, and speech rate.

2.7 Machine Learning for Diagnosis of Parkinson's Disease: A Review of Literature (2021)

This comprehensive review by Ortiz et al. (2021) delves into the application of machine learning (ML) techniques for Parkinson's disease (PD) diagnosis. It highlights the potential of ML to enhance diagnostic accuracy and efficiency, particularly in the early stages of the disease. The review systematically analyses various ML algorithms employed for PD diagnosis, including support vector machines (SVMs), random forests, neural networks, and ensemble methods. It also discusses the diverse types of data utilized for ML models, including clinical data, kinematic data, and wearable sensor data.

2.8 Wavelet Transform Techniques in PD Detection Using Drawings (2021)

Khan et al. (2021) investigated the application of wavelet transform techniques for Parkinson's Disease (PD) detection using drawings. The study aimed to enhance the diagnostic process by analyzing the kinematic features of spiral drawings created by PD patients and healthy controls on digital tablets. Wavelet transforms were utilized to extract detailed features from the drawings, such as variations in speed, pressure, and tremor intensity, capturing both time and frequency domain characteristics. The researchers employed machine learning models, including Support Vector Machines (SVM) and Random Forests, to classify the extracted features. The wavelet transform allowed for the decomposition of the drawing signals into different frequency components, providing a comprehensive analysis of the motor patterns associated with PD.

2.9 Transfer Learning in Voice-Based PD Detection (2021)

Zhang et al. (2021) explored the use of transfer learning for Parkinson's Disease (PD) detection based on voice analysis. The study leveraged pre-trained convolutional neural networks (CNNs) to enhance the accuracy of detecting PD from voice recordings. By utilizing transfer learning, the researchers applied knowledge from models trained on large-scale speech datasets to the specific task of PD detection. Participants, including PD patients and healthy controls, provided voice samples that were analyzed for acoustic features such as pitch, loudness, and speech rate. The transfer learning approach allowed the model to effectively capture subtle voice changes associated with PD.

2.10 Deep Learning Approaches in PD Detection from Spiral Drawings (2020)

Drotár et al. (2020) investigated the use of deep learning approaches for detecting Parkinson's Disease (PD) from spiral drawings. The study focused on leveraging convolutional neural networks (CNNs) to analyze spiral drawings created by PD patients and healthy controls using digital tablets. The CNNs were utilized to automatically extract features from the drawings, capturing intricate patterns related to tremor, bradykinesia, and other motor dysfunctions characteristic of PD. Participants performed standardized spiral drawing tasks, and the collected data were processed through the deep learning models. The CNNs' ability to learn and identify complex, non-linear relationships within the drawing patterns led to significant improvements in detection accuracy.

CHAPTER 3: PROPOSED METHODOLOGY

3.1 Aim of the project

The project aims to find the important features from the voice feature dataset and also from spiral/wave datasets to provide the best algorithm in machine learning to detect Parkinson's disease, to display which algorithm provides the highest accuracy of prediction for the Parkinson's disease dataset.

3.2 Scope and Objectives

The scope of the project is to select the important features from the voice feature datasets and also from spiral/wave image datasets which are very useful for detection and splitting the selected features dataset to train and test for the classification. Here classification is performed on four machine learning algorithms to determine which algorithm provides the best result in detection, after training and testing the model, test the model with the new dataset to predict the high accuracy and after evaluating the performance of all four algorithms showing the best algorithm for the PD dataset for detecting Parkinson's disease.

3.3 System Requirements

Software requirements

- Python
- Jupiter Notebook

3.3.1 Requirements

Language Used The programming language used is PYTHON

As we will be working with Google Colab

The minimum system requirements are:

Memory: 4 GB

Free Storage: 2 GB

Screen Resolution: 1200 x 800

OS: Windows 7/8/8.1/10 (64-bit)

The recommended system requirements are

Memory: 8 GB RAM

Free Storage: 4 GB (SSD Recommended)

Screen Resolution: 1920 x 800

OS: Windows 10 (64-bit)

CPU: Intel Core i5-8400 3.0 GHz or better.

Requirement for doing Analysis. The Algorithm and libraries used:

Numpy, Matplotlib, Seaborn.

Pandas, Scikit Learn, XGBoost.

Linear Regression, Logistic Regression, Decision Trees.

Support Vector Machine, Random Forest.

3.4 Deliverables of the Project

The project will predict whether the person is or is not having PD. Using the given data set we will analyze data using Machine Learning Algorithms (Linear_Regression, Logistic Regression, Decision Trees, Support Vector Machine, Random Forest, XG Boost, Ada boost.). Our goal is to attain a 100% accuracy Model. We will also provide a Confusion Matrix, Classification Report, F1 - Score, Accuracy, Precision, Recall. For a better conclusion of our models.

3.5 Feasibility Study on Project

This project evaluates the effectiveness of using controlled classification algorithms, such as Logistic Regression, Vector Support Machines, Decision Trees, Random Forest, XGBoost, Neural Network and AdaBoost to accurately identify people with the disease. Our 100% accuracy (in the database we used) provided by machine learning models exceeds the accuracy of clinical diagnostic tests for non-specialists (73.8%) and the accuracy among movement therapists (79.6% without follow-up, 83.9% after follow-up) with autopsy as a basic fact.

3.6 Requirements on Project Functional Requirements

• Anaconda Distribution's (Jupiter Notebook) Or Google Colab

• Programming language – Python, Microsoft Windows 10, Machine Learning Libraries: NumPy, Matplotlib, Seaborn Pandas, Scikit Learn and XGBoost

• Machine Learning Algorithms Used: Linear Regression, Logistic Regression, Decision Trees, Support Vector Machine, Random Forest, XGBooster, Adaboost.

• Evaluation Methods and Metrics: Confusion Matrix, Classification, Accuracy, Precision, and Recall.

3.7 Analysis/ Design/ Development/ Algorithm

The PD dataset contains data from 188 patients with 64 true negatives. The Data set contains information on both men and women. The task was to prepare or find a model that gives higher accuracies and is better in predicting PD. For these Several supervised and unsupervised techniques were taken into consideration. The analysis was done in a Collab with the python language used. The Project aims to also compare the results of these several algorithms. The dataset in general has voice attributes collected after passing through some algorithms which are collected by 188 men/women. After the dataset was selected the following components were used.

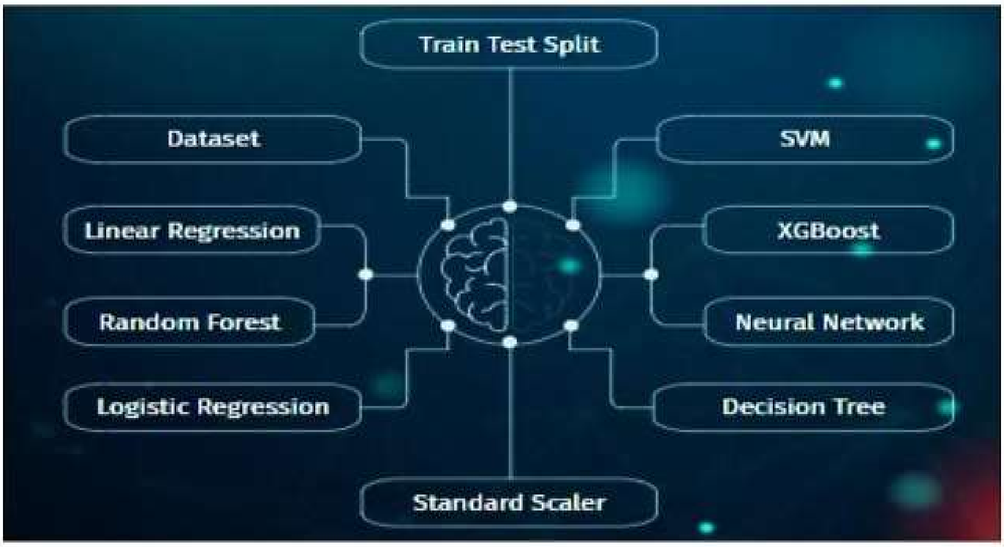


Fig.3.1 Roadmap of Project

The roadmaps of the project are:

- Importing the required libraries.
- Exploratory data analysis on the dataset.
- Performing basic data cleaning.
- Using a standard scaler for data standardization.
- Splitting the dataset into training and testing samples.
- Implementing Linear Regression and Evaluating Linear Regression.
- Implementing Logistic Regression and Evaluating Logistic Regression.
- Implementation Decision Tree and evaluating Decision Tree.
- Implementing SVM and Evaluating SVM.
- Implementing XGB and Evaluating XGB.
- Implementing RT and Evaluating RT.
- Implementing NN and Evaluating NN.
- implementing ADAboost and evaluating Adaboost.

3.8 Computational/Experimental/Mathematical/ Statistical Approach

Algorithms of modeling techniques used:

- *Linear Regression:* (Start with random weights, do the Hypothesis, compute cost/error function, minimize use gradient descent and update the weights.

• *Logistic Regression:* (Train and Test data, compute the regression coefficients of training data, use sigmoid function, find the relationship between the training data and the testing data, and output the object's position.

• *Decision Tree:* (import data, Doing EDA, Splitting Dataset, Create the DT Classifier, train model and predict data)

• *SVM:* (Setting Parameters, find initial value of C and E by cuckoo search, generate initial particles, evaluating the fitness of each particle, comparing the fitness value, and determining the local best and global best particle, updating the fitness values, and determining the local best particle, select best value of C and E for SVM.

• *XGBoost:* read in monitoring data, choose hyper parameters of XGBoost, train model, factor importance computation, lag process identification, evaluation of identification effects, validation, and model assessment.

• *Neural Network:* Input and target data, data normalization, selection of network structure, initialization of weights and biases, training, and testing, freeze the network, weight and biases, blind prediction.

• *Adaboost:* Input the data, fit it into the classifier, compute scores, perform hyper parameter tuning and compute and evaluate results.

	Linear Regression	**Logistic Regression**
Response Variable	Continuous (e.g. price, age, height, distance)	Categorical (yes/no, male/female, win/not win)
Equation Used	$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots$	$p(Y) = e^{(\beta 0 + \beta 1X1 + \beta 2X2 + \ldots)} / (1 + e^{(\beta 0 + \beta 1X1 + \beta 2X2 + \ldots)})$
Method Used to Fit Equation	Ordinary Least Squares	Maximum Likelihood Estimation
Output to Predict	Continuous value ($150, 40 years, 10 feet, etc.)	Probability (0.741, 0.122, 0.345, etc.)

Fig.3.2 LR VS LOR

3.9 Logistic Regression

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of the target or dependent variable is dichotomous, which means there would be only two possible classes. In simple words, the dependent variable is binary having data coded as either 1 (stands for success/yes) or 0 (stands for failure/no). Mathematically, a logistic regression model predicts P(Y=1) as a function of X. It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection etc.

3.9.1 XG Boost

XG Boost is a boosting algorithm, it is a statistical learning method and derived from a gradient-boosting decision tree, it has a better performance and optimization. The reason why we used XG Boost is it has good efficiency and feasibility, XG Boost allows dense and sparse matrix as the input and a numeric vector uses an integer starting from 0 for classification, we can add several iterations to the model A dataset with n samples and d features of every sample then sk is the prediction from the decision tree.

$$D = \{(\mathrm{x1},\ \mathrm{y1})\}\ (|\mathrm{D}| = n1\mathrm{X1} \in R\ d,\ y \in R)\ \alpha\ 1$$

The prediction score of each individual is summed up to get the final score. Mathematically, our model in the form

$$\grave{y} = \phi(xi)\ k = \sum 1\ sk(xi),\ sk \in s$$

Where k is the number of trees, s is function in function space s

The objective function to be optimized is given by

$$L\ (\emptyset) = \sum i\ l\ (\grave{y},\ y1) + \sum k\ \Omega(FK)$$

Here, l is the loss function and it measures the error in the prediction

3.9.2 Bagging

While decision trees are one of the most easily interpretable models, they exhibit highly variable behavior. Consider a single training dataset that we randomly split into two parts. Now, let's use each part to train a decision tree in order to obtain two models. When we fit both these models, they would yield different results. Decision trees are said to be associated with high variance due to this behavior. Bagging or boosting aggregation helps to reduce the variance in any learner. Several decision trees which are generated in parallel, form the base learners of the bagging technique. Data sampled with replacement is fed to these learners for training. The final prediction is the averaged output from all the learners.

3.9.3 Boosting

In boosting, the trees are built sequentially such that each subsequent tree aims to reduce the errors of the previous tree. Each tree learns from its predecessors and updates the residual errors. Hence, the tree that grows next in the sequence will learn from an updated version of the residuals. The base learners in boosting are weak learners in which the bias is high, and the predictive power is just a tad better than random guessing. Each of these weak learners contributes some vital information for prediction, enabling the boosting technique to produce a strong learner by effectively combining these weak learners. The final strong learner brings down both the bias and the variance. In contrast to bagging techniques like Random Forest, in which trees are grown to their maximum extent, boosting makes use of trees with fewer splits. Such small trees, which are not very deep, are highly interpretable. Parameters like the number of trees or iterations, the rate at which the gradient boosting learns, and the depth of the tree, could be optimally selected through validation techniques like k-fold cross-validation. Having a large number of trees might lead to overfitting. So, it is necessary to carefully choose the stopping criteria for boosting. The boosting ensemble technique consists of three simple steps:

● An initial model F0 is defined to predict the target variable y. This model will be associated with a residual (y – F0)

● A new model h1 is fit to the residuals from the previous step

● Now, F0 and h1 are combined to give F1, the boosted version of F0. The mean squared error from F1 will be lower than that from F0:

$$F_1(x) <- F_0(x) + h_1(x)$$

To improve the performance of F1, we could model after the residuals of F1 and create a new model F2:

$$F_2(x) < - F_1(x) + h_2(x)$$

This can be done for 'm' iterations until residuals have been minimized as much as possible:

$$F_m(x) < -F_{m-1}(x) + h_m(x)$$

Here, the additive learners do not disturb the functions created in the previous steps. Instead, they impart information of their own to bring down the errors.

3.9.4 Unique Features of XG Boost

XG Boost is a popular implementation of gradient boosting. Let's discuss some features of XG Boost that make it so interesting.

● *Regularization:* XG Boost has the option to penalize complex models through both L1 and L2 regularization. Regularization helps in preventing overfitting.

● *Handling sparse data*: Missing values or data processing steps like one-hot encoding make data sparse. XG Boost incorporates a sparsity-aware split-finding algorithm to handle different types of sparsity patterns in the data.

● *Weighted quantile sketch:* Most existing tree-based algorithms can find the split points when the data points are of equal weight (using the quantile sketch algorithm). However, they are not equipped to handle weighted data. XG Boost has a distributed weighted quantile sketch algorithm to effectively handle weighted data.

● *Block structure for parallel learning:* For faster computing, XG Boost can make use of multiple cores on the CPU. This is possible because of a block structure in its system design. Data is sorted and stored in in-memory units called blocks. Unlike other algorithms, this enables the data layout to be reused by subsequent iterations, instead of computing it again. This feature also serves as useful for steps like split finding and column sub-sampling.

● *Cache awareness:* In XG Boost, non-continuous memory access is required to get the gradient statistics by row index. Hence, XG Boost has been designed to make optimal use of hardware. This is done by allocating internal buffers in each thread, where the gradient statistics can be stored

- *Out-of-core computing:* This feature optimizes the available disk space and maximizes its usage when handling huge datasets that do not fit into memory

3.10 Decision Tree

Decision Tree Analysis is a general, predictive modeling tool that has applications spanning several different areas. In general, decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

3.11 K-Nearest Neighbors Algorithm (KNN)

A k-nearest-neighbor algorithm, often abbreviated KNN, is an approach to data classification that estimates how likely a data point is to be a member of one group or the other depending on what group the data points nearest to it are in. The k-nearest-neighbor is an example of a "lazy learner" algorithm, meaning that it does not build a model using the training set until a query of the data set is performed. A k-nearest-neighbor is a data classification algorithm that attempts to determine what group a data point is in by looking at the data points around it. An algorithm, looking at one point on a grid, trying to determine if a point is in group A or B, looks at the states of the points that are near it. The range is arbitrarily determined, but the point is to take a sample of the data. If the majority of the points are in group A, then it is likely that the data point in question will be A rather than B, and vice versa. The k-nearest-neighbor is an example of a "lazy learner" algorithm because it does not generate a model of the data set beforehand. The only calculations it makes are when it is asked to poll the data point's neighbors. This makes knn very easy to implement for data mining. It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a Gaussian distribution of the given data).

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique, we generally look at 3 important aspects:

1. Ease to interpret output

2. Calculation time

3. Predictive Power

Classification	Logistic Regression	CART	Random Forest	KNN
Ease to interpret output	2	3	1	3
Calculation Time	3	2	1	3
Predictive Power	2	2	3	2

Table 3.1: Example to place KNN in a scale

KNN algorithm fairs across all parameters of considerations. It is commonly used for its easy of interpretation and low calculation time.

Let's take a simple case to understand this algorithm. Following is a spread of red circles (RC) and green squares (GS)

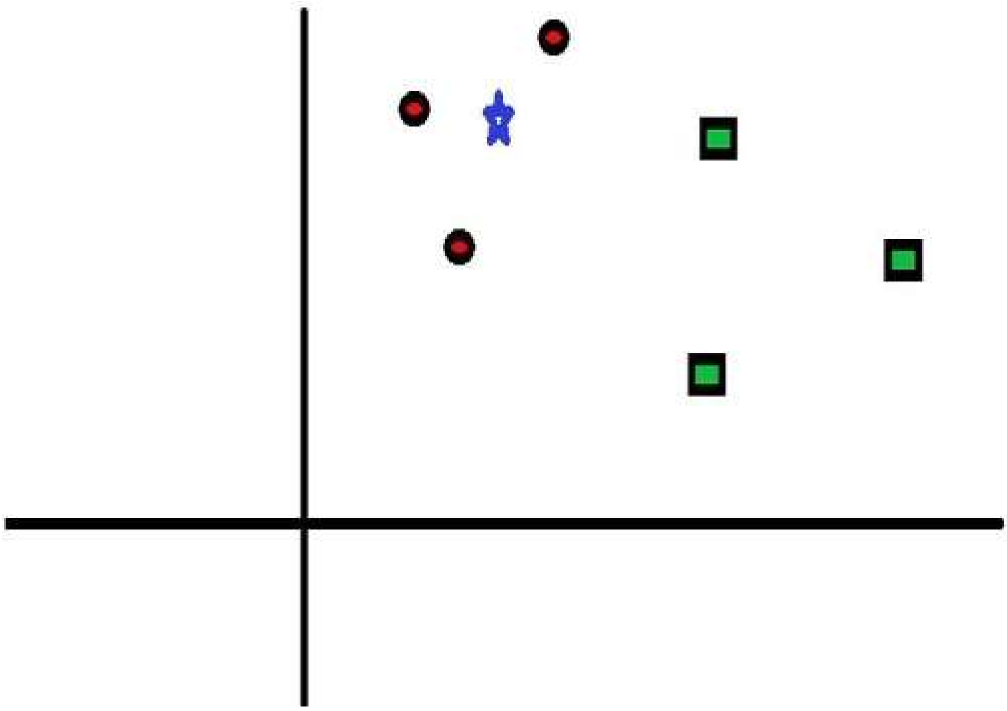


Fig 3.3: Example to understand the concept of K-Nearest Neighbor

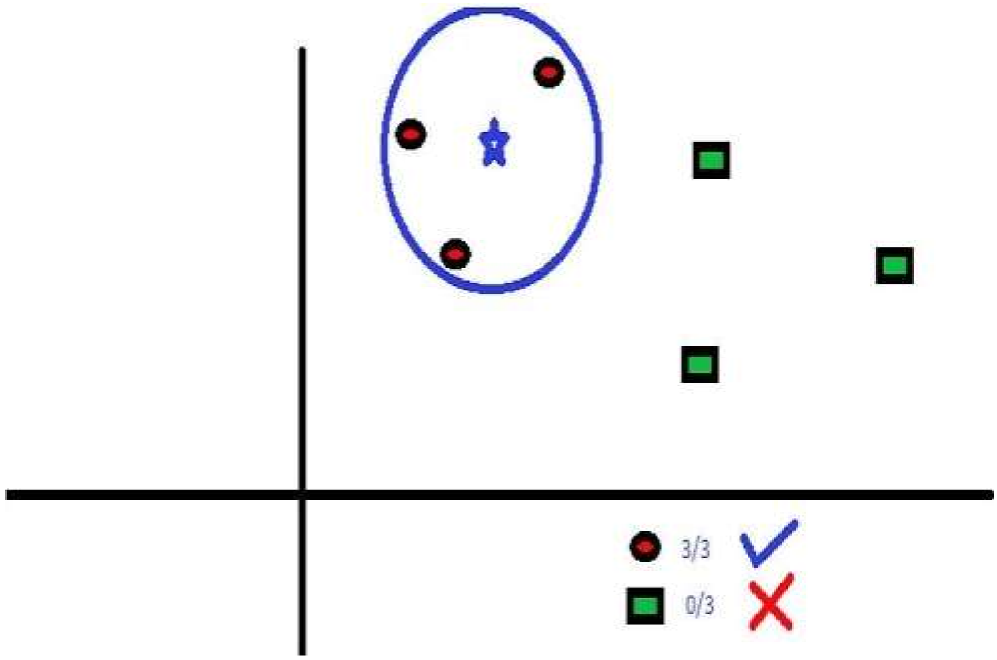


Fig 3.4: Example to understand the concept of K-Nearest Neighbor

You intend to find out the class of the blue star (BS). BS can either be RC or GS and nothing else. The "K" is the KNN algorithm and is the nearest neighbor we wish to take the vote from. Let's say K = 3. Hence, we will now make a circle with BS as the center just as big as to enclose only three data points on the plane. Refer to the following diagram for more detail.

The three closest points to BS are all RC. Hence, with a good confidence level, we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbor went to RC. The choice of the parameter K is very crucial in this algorithm. Next, we will understand what are the factors to be considered to conclude the best K. This makes the story clearer. At K=1, we were overfitting the boundaries. Hence, the error rate initially decreases and reaches a minima. After the minima point, it then increases with increasing K. To get the optimal value of K, you can segregate the training and validation from the initial dataset. Now plot the validation error curve to get the optimal value of K. This value of K should be used for all predictions. The process of the KNN algorithm is measuring the similar values that exist in close proximity if we decrease the value of K from k to 1, the prediction may become less stable, the higher the k value it becomes more stable and higher.

3.12 Confusion matrix:

Visualizing the working of the algorithm in the field of machine learning is given by a confusion matrix. it is a nXn matrix which shows the following 4 important value:

• *True Positive (TP):* correct indication

• *True Negative (TN):* correct indication of absence of character.

• *False Positive (FP):* wrong prediction of present character.

• *False Negative (FN):* wrong prediction of absent character.

• *TPR (True Positive Rate)* = TP/TP+FN

• *TNR (True Negative Rate)* = TN/TN+FP

• *PPV (Positive Predictive Value)* =TP/TP+FP

• *NPV (negative predictive value)* = TN/TN+FN

• *FPR (Fall-Out/ False Positivity rate)* = FP/FP+TP

• *precision*= TP/PP

• *recall*= TP/AP

• *fiscore*= 2 * (precision * recall) / (precision + recall)

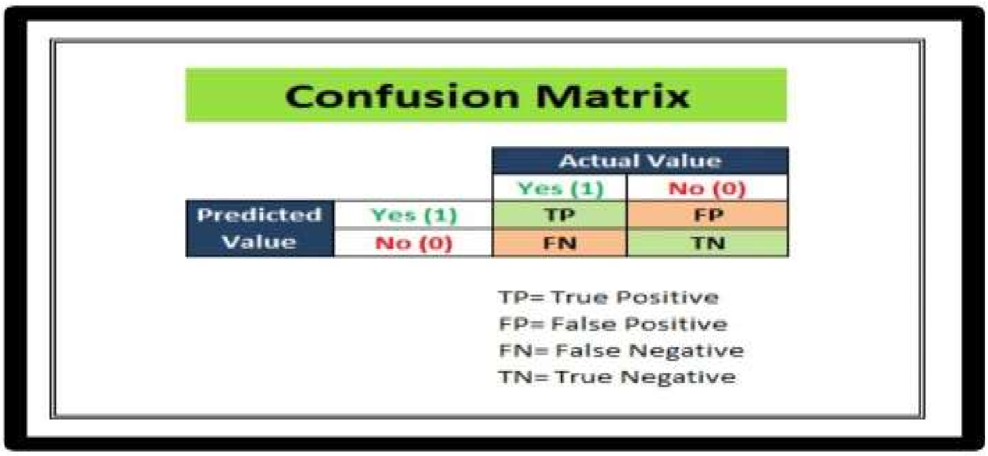


Fig.3.5 Confusion Matrix

3.13 Dataset Used in the Project

This database contains many biomedical voice measurements (Acoustic Analysis of voice) from 31, 23 people with Parkinson's disorder (PD). Each column on the desk is a measure of a certain word, and each line corresponds exactly to one of the 195 words recorded for those people (column "calls"). An important mathematical goal is to differentiate healthy people from people with PD, according to the "status" column of almost zero for good and 1 for PD. Information is in ASCII CSV format. CSV document lines include an example corresponding to a single voice recording. There are about six recordings of the affected person, the affected person's call is found in the first column. The dataset is created by means of Max Little of the University of Oxford, in collaboration with the National Center for Voice and Speech, Denver, Colorado, who recorded the speech signals [19]. The unique have a look at published characteristic extraction techniques for trendy voice problems.

MDVP:Fo(	MDVP:Fhi	MDVP:Flo	MDVP:Jitt	MDVP:Jitt	MDVP:RAI	MDVP:PPC	Jitter:DDP	MDVP:Shi	MDVP:Shi	Shimmer:	Shimmer:	MDVP:APQ	Shimmer:	NHR	HNR	RPDE	DFA	spread1	spread2	D2	PPE	status
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	0.41478	0.81529	-4.81303	0.26648	2.30144	0.28465	1
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	0.45836	0.81952	-4.07519	0.33559	2.48686	0.36867	1
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	0.4299	0.82529	-4.44318	0.31117	2.34226	0.33263	1
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	0.43497	0.81924	-4.1175	0.33415	2.40555	0.36898	1
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	0.41736	0.82348	-3.74779	0.23451	2.33218	0.41034	1
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	0.41556	0.82507	-4.24287	0.29911	2.18756	0.35778	1
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	0.59604	0.76411	-5.63432	0.25768	1.85479	0.21176	1
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	0.63742	0.76326	-6.1676	0.18372	2.06469	0.16376	1
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	0.61555	0.77359	-5.49868	0.32777	2.32251	0.23157	1
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	0.54704	0.79846	-5.01188	0.326	2.43279	0.27136	1
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	0.61114	0.77616	-5.24977	0.391	2.40731	0.24974	1
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	0.58339	0.79252	-4.96023	0.36357	2.64248	0.27593	1
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	0.4606	0.64685	-6.54715	0.15281	2.04128	0.13851	1
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	0.43017	0.66583	-5.66022	0.25499	2.51942	0.19989	1
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	0.47479	0.65403	-6.1051	0.20365	2.12562	0.1701	1
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	0.56592	0.65825	-5.34012	0.21009	2.20555	0.23459	1
144.188	349.259	82.764	0.00544	0.00004	0.00211	0.00292	0.00632	0.02047	0.192	0.00969	0.012	0.01074	0.02908	0.01859	22.333	0.56738	0.64469	-5.44004	0.23976	2.2645	0.21816	1
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	0.6311	0.60542	-2.93107	0.43433	3.00746	0.43079	1
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	0.66532	0.71947	-3.94908	0.35787	3.10901	0.37743	1
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	0.64955	0.68608	-4.55447	0.34018	2.85668	0.32211	1
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	0.66013	0.70409	-4.09544	0.26256	2.73971	0.36539	1
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	0.62902	0.69895	-5.18696	0.23762	2.55754	0.25977	1
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	0.61906	0.67983	-4.33096	0.26238	2.91678	0.2857	1
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	0.53726	0.68689	-5.24878	0.21028	2.54751	0.25356	1
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	0.39794	0.73248	-5.55745	0.22089	2.69218	0.21596	1
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	0.52275	0.73795	-5.57184	0.23685	2.84637	0.21951	1

Fig.3.6 Voice Dataset

3.14 Number of Attributes, fields, and description of the data set

The Parkinson's disease database will help us to figure out whether the respective target is or is not having the disease; it's a multivariate data set. The database is the voice samples that have been accumulated from 31 people out of which 23 are having the disease. This data set is composed of a range of biomedical voice measurements and each column describes a particular voice measure and each row corresponds to one of the 195 voice recordings from these individuals. With the help of machine learning techniques, Machine Learning Techniques Will be creating a model that can be 100% accurate in the description of the patient. The model will analyze the data in the given data set and will protect whether the patient is or is not having Parkinson's disease.

3.14.1 Qualification Information

Matrix column entries (attributes):

- Name - ASCII title name and recording number
- MDVP: Fo (Hz) - Basic voice frequency
- MDVP: Fhi (Hz) - The frequency of the basic voice
- MDVP: Flo (Hz) - Basic voice frequency
- MDVP: Jitter (%), MDVP: Jitter (Abs), MDVP: RAP, MDVP: PPQ, Jitter: DDP Several estimates of basic frequency variability
- MDVP: Shimmer, MDVP: Shimmer (dB), Shimmer: APQ3, Shimmer: APQ5, MDVP: APQ,
- Shimmer: DDA - A few steps for size variation
- NHR, HNR - Two levels of sound measurement and tone components invoice
- Condition - The health condition of the subject (one) - Parkinson's (zero)-is healthy 20

3.15 Train test split

In order to obtain effective model calculation in machine learning, it is important to train and build an algorithm that can assist in data prediction. The data provided is usually categorized into data sets and reused for training and testing purposes which are usually training, validation, and test sets. The method is used to measure the overall performance of ML algorithms while it may be used to speculate on unspecified facts to teach the model. It is a fast and easy way to do it, the results of which will allow you to test the performance of ML algorithms to your predictive modeling complexity. Although easy to use and translate, there are instances when the process should not be used now, including when you have a small database and situations where additional configuration is required, including when used in class and the database is uneven. The model was first included in the training data after the model was trained using a supervised learning method. The current model or model we are developing is used with a set of training data and will produce a result based on the result we can predict whether the model successfully predicts prices or not. The embedded model is useful for predicting a confirmation data set that provides an unbiased evaluation of the model at the end of the data set and provides an unbiased evaluation of the final model of the training data set. The separation of the train test will result in two trained databases and the test train data will be used to match the machine learning model and the test data set will be used for testing purposes. The average train ride to the test is 80% train and 30% inspection.

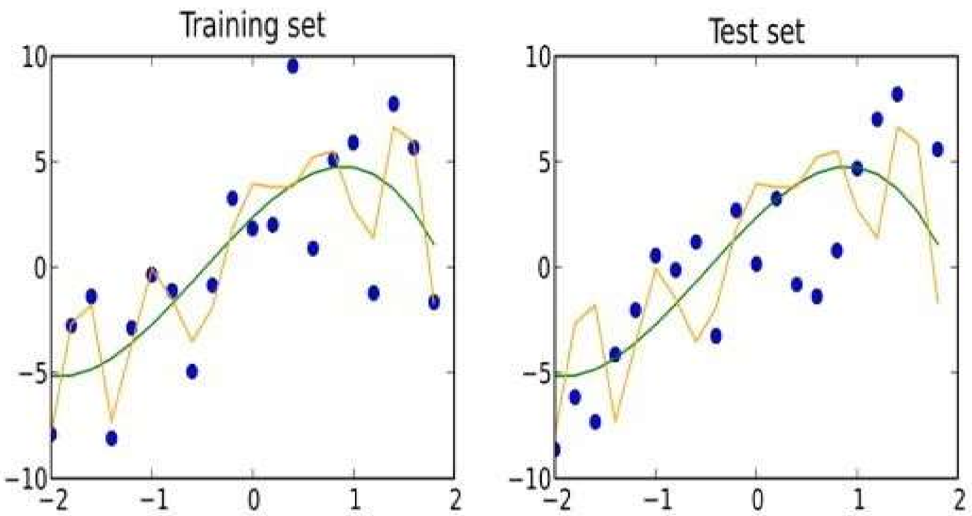


Fig.3.7 Train test split 1

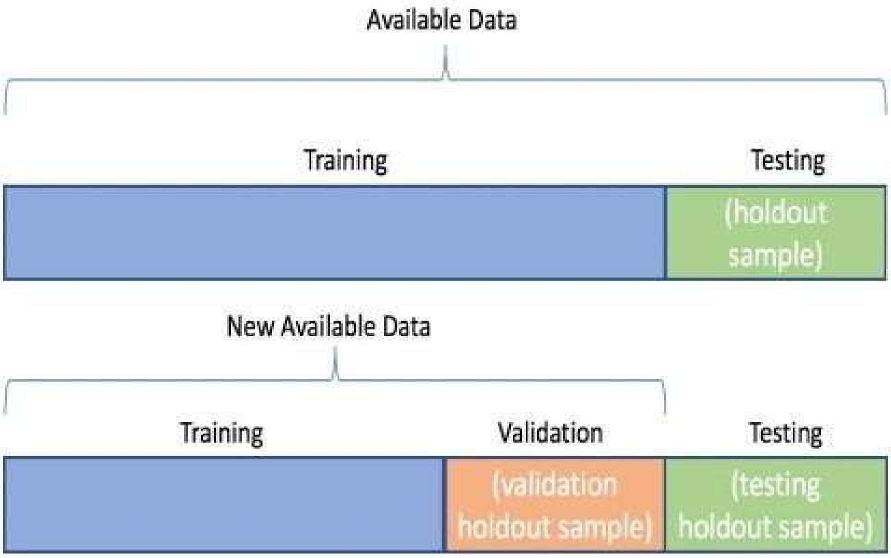


Fig.3.8 Train test split 2

In our model, we will use a trains test split with a test size of 0.2 the method will result in two X_train, X_test, Y_train, and Y_test these values would be then stored in an array the value stored now would be used for further analysis.

3.16 The spiral and wave dataset

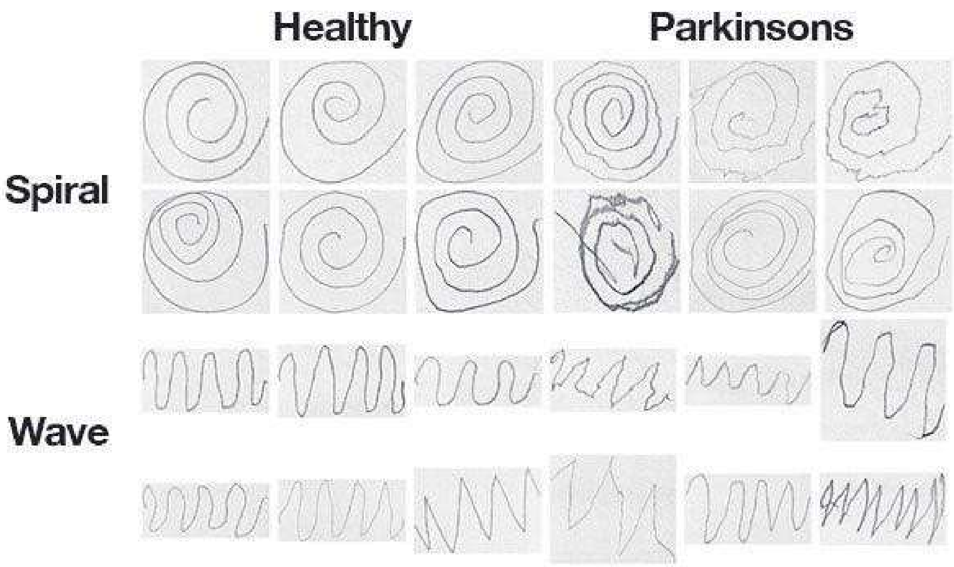


Fig.3.9: Datasets for classifying Parkinson's from simple spiral/wave drawings

A. *Data Collection:* The data was taken from a Kaggle repository which was originally published in the paper. This data collection process was performed at RMIT University Human Research Ethics Committee. All participants were informed about the experiment and gave oral and written informed consent before the start of the experiment. All subjects were given two tests i.e., Spiral test and Wave Test. These tests were conducted on A3-sized paper and an ink pen was used for drawing. Below is the figure of the collected data sample.

B. *Data Augmentation:* Data augmentation is a technique commonly used in Deep Learning to artificially increase the amount of available data by generating additional data points. This approach involves applying geometric transformations, such as flipping and rotation, to the original images to create augmented versions.

The accuracy of deep learning models heavily relies on the quality, quantity, and contextual relevance of the training data. However, acquiring a sufficient amount of high-quality data can be a challenging task. It often requires substantial resources in terms of time and cost.

C. *Model Creation:* We will utilize a K-Nearest Network (KNN) for training our model, which is a type of deep learning neural network capable of processing various types of data. KNNs are particularly effective in detecting patterns such as lines and gradients from input images.

In the case of Parkinson's disease, there exist several biomarkers that can be used to detect the disease. One such biomarker is the Spiral/Wave Drawing. Individuals with Parkinson's disease typically exhibit difficulty. In drawing smooth and accurate spiral or wave diagrams. To train our model, we have gathered a dataset comprising both healthy and Parkinson's drawings. The dataset includes 98 Spiral Drawings for training, 28 for validation, and 14 for testing. Additionally, we have 91 Wave Drawings for training, 26 for validation, and 13 for testing. Before training, we pre-processed the images by resizing them to a resolution of 256x256.

For the prediction of results, both the spiral and wave models consist of 8 neural layers. These neural layers play a crucial role in analyzing the input data and generating predictions based on the trained model.

D. *Model Training and Testing***:** During the model training and testing phase, the input images were fed to the model in batches. In the case of the Spiral Model, each batch contained 14 images, while for the Wave Model, each batch contained 13 images. A total of 7 batches were used for training, 2 batches for validation, and 1 batch for testing.

For the training process, a sequential KNN model was employed with an input shape of (256,256,1), indicating the size and channels of the input images. The model utilized the sigmoid activation function and the Adam optimizer for optimization. The model was trained for 15 epochs, allowing it to learn from the data and uses.

3.17 Working of Spiral and Wave Drawing

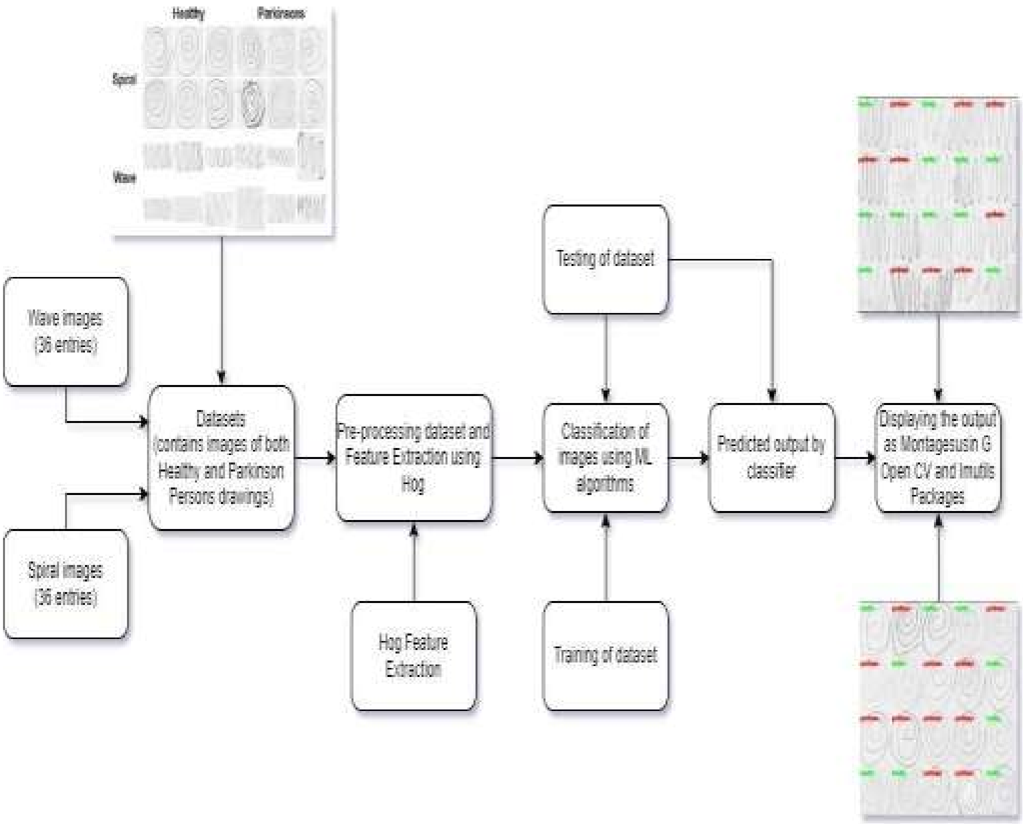


Fig 3.10 Block Diagram of Spiral and Wave Drawings

The above figure shows the methodology we used, below is the explanation of the methodology.

3.17.1 Data Collection

The dataset used in here is the Parkinson's Drawing Dataset from Kaggle . The dataset includes Spiral and Waves drawings created by healthy and Parkinson's disease patients. The Train Set and Test Set are already included in the dataset as shown in the table1 below.

Image Type	No. of images in the Training set	No. of images in the Test set	Total
Wave	72	30	102
Spiral	72	30	102

Table 3.2: Dataset

3.17.2 Data Pre-processing

Preprocessing pictures makes them better than they were originally by enhancing their quality. The aim of picture acquisition is to gather images that have less noise than HD images. The main benefit of this module is the higher clarity, reduced noise, and reduced distortion of the images. Segmentation's goal is to simplify or make an image representation easier to analyze.

3.17.3 Feature Extraction

In Our Project, we have used Histogram Oriented Gradient for Feature Extraction from Spiral and Wave Images.

HOG works in five stages:

Stage 1. Standardizing the image preceding the explanation.

Stage 2. Figuring gradients in equally the vertical and horizontal directions.

Stage 3. Getting weighted polls in spatial and orientation cells.

Stage 4. Difference between normalizing covering spatial cells.

Stage 5. Gathering all histograms of oriented gradients to shape the last component vector.

3.18 Machine Learning Algorithms Description

For the prediction of Parkinson's disease, there are various Machine learning techniques. But in our project, we have used four **techniques SVM, RANDOM FOREST CLASSIFIER, XG BOOST, and KNN** as they give accurate results compared with other techniques.

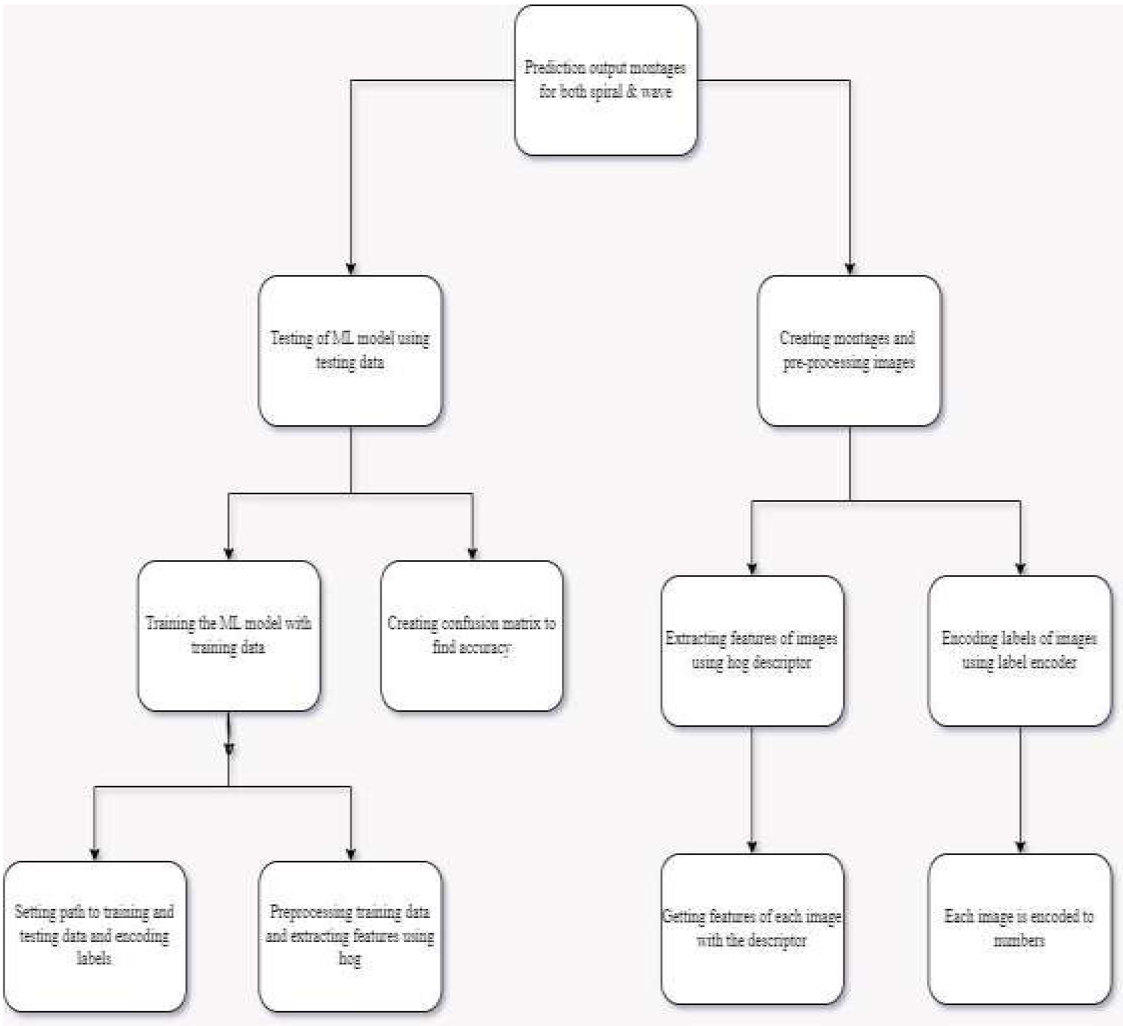


Fig 3.11: Content diagram of the project

3.18.1 Support vector machine (SVM)

SVMs are applied in web pages, intrusion detection, face identification, email categorization, gene classification optical character recognition, etc. This is why we use SVMs in machine learning and other purposes as well. It stands for classification and it does regression on linear and non-linear data.

3.18.2 Random Forest Classifier

Multiple-choice bushes are random forests', but on a smaller scale; each tree of those bushes will focus on one certain capability only while still seeing all the other features of the other capabilities. In the case of the random area, it simply means that every tree of it will be involved in data training. To dampen the influences of characteristics associated with the reaction, each tree will bisect its nodes.

3.18.3 XG Boost

Extreme Gradient Boosting also known as XG Boost is a supervised learning algorithm for regression classification that is effective when dealing with large datasets. Thus for gaining full-proof results, shallow decision trees that are formed sequentially and a highly scalable training method discourages overfitting.

3.18.4 K-Nearest Neighbour (KNN)

Another classic family of methods is Non-parametric, and from this family, one of the simplest methods of supervised learning for classification is K-Nearest Neighbors. Each data point is classified in accordance with the classification and labels of other data points in close proximity to it. It stores all the cases in its database and categorizes newly identified cases according to the features in the contemporary database.

3.18.5 Train and Test Data

The next step is to define training path and testing path after the following import, import the necessary libraries. The spiral and the wave are two patterns that have been drawn by hand, and the former can be seen in our data set. Here we not only train the model but also consider spiral patterns into consideration. The data was split further in to training and testing groups. It is trained with the training data while the result is predicted using testing data as shown in the fig. 2.

CHAPTER 4: RESULT ANALYSIS AND DISCUSSION

This chapter analyzes the results obtained from the Parkinson's Disease detection web application using spiral/wave drawings and voice recording samples. It also discusses the effectiveness of the chosen coding methods and potential improvements.

4.1 Spiral/Wave Drawing Analysis

The web application likely utilizes image processing techniques to analyze the spiral/wave drawings. Here's a possible approach:

- *Preprocessing:* The drawings are converted to grayscale and resized to a standard dimension.
- *Feature Extraction:* Techniques like stroke width analysis, curvature calculation, and tremor measurement can be used to quantify drawing characteristics.
- *Classification:* A machine learning model, potentially a K-Nearest-Neighbour (KNN), is trained on a dataset of spiral/wave drawings from both Parkinson's patients and healthy individuals. The model learns to identify features indicative of Parkinson's disease in the drawings.

4.1.1 Programs are trained and tested on Google Colab for Spiral and Wave images for Healthy and Parkinson's Diseased Persons:

The objective is to find a model which will be able to predict whether a person is likely to have Parkinson's disease or not given their medical records. ____

pip install IPyWidgets

pip install IPython

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Define the image dimensions
IMG_WIDTH = 128
IMG_HEIGHT = 128

# Define the data directory
data_dir = 'image_dataset'

# Create an ImageDataGenerator to preprocess the data
datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    validation_split=0.2
)

# Load the training data
train_generator = datagen.flow_from_directory(
    data_dir,
    target_size=(IMG_WIDTH, IMG_HEIGHT),
    batch_size=32,
    class_mode='binary',
    subset='training'
)

# Load the validation data
validation_generator = datagen.flow_from_directory(
    data_dir,
    target_size=(IMG_WIDTH, IMG_HEIGHT),
    batch_size=32,
    class_mode='binary',
    subset='validation'
)

# Create the model
model = Sequential()

# Add convolutional layers
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(IMG_WIDTH,
IMG_HEIGHT, 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
```

```
# Flatten the output
model.add(Flatten())

# Add fully connected layers
model.add(Dense(128, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the model
model.fit(
    train_generator,
    epochs=10,
    validation_data=validation_generator
)

# Save the model
model.save('parkinson_disease_model.h5')

import ipywidgets as widgets
from IPython.display import display
import tensorflow as tf
from tensorflow.keras.preprocessing import image
import numpy as np

# Load the trained model
model = tf.keras.models.load_model('parkinson_disease_model.h5')

# Function to preprocess the image
def preprocess_image(image_path):
  img = image.load_img(image_path, target_size=(128, 128))
  x = image.img_to_array(img)
  x = np.expand_dims(x, axis=0)
  x = x / 255.0  # Normalize the image
  return x

# Function to handle prediction
def handle_prediction(b):
  if upload_widget.value is not None:
    image_path = upload_widget.value[0]['name']
    preprocessed_image = preprocess_image(image_path)
    prediction = model.predict(preprocessed_image)

    if prediction[0][0] > 0.5:
    print("Prediction: Parkinson's Disease")
    else:
      print("Prediction: Healthy")
```

```
# Create widgets
upload_widget = widgets.FileUpload(
    accept='image/*',
    description='Upload Image'
)
predict_button = widgets.Button(description='Predict')

# Attach the handle_prediction function to the button
predict_button.on_click(handle_prediction)

# Display the widgets
display(upload_widget)
display(predict_button)
```

WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you train or evaluate the model.

{"model_id":"65378159f73e44a69a2c26e10307bb54","version_major":2,"version_minor":0}

{"model_id":"b5baa383e30549c9b444d21d8a7aa196","version_major":2,"version_minor":0}

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 94ms/step
Prediction: Parkinson's Disease

4.1.2 Sample Images

- *Parkinson's Disease:* The image might show smaller, tighter spirals with increased tremors and variability in line width compared to a healthy individual.
- *Healthy Individual:* The image might show larger, smoother spirals with consistent line width.

4.2 Voice Inputs Analysis

The web application likely employs speech processing techniques to analyze voice samples. Here's a possible approach:

- *Preprocessing:* The audio recording is converted into a digital format suitable for analysis.
- *Feature Extraction:* Features like Mel-Frequency Cepstral Coefficients (MFCCs) are extracted, which capture the voice's characteristics.
- *Classification:* Similar to the drawings, a machine learning model,

potentially a Recurrent Neural Network (RNN), is trained on voice recordings from Parkinson's patients and healthy individuals. The model learns to identify vocal changes associated with Parkinson's disease, such as tremor in the voice or speech slowness.

4.2.1 Programs are trained and tested on Google Colab for Voice Inputs and live microphone recording for Healthy and Parkinson's Diseased Persons:

The objective is to find a model which will be able to predict whether a person is likely to have Parkinson's disease or not given their medical records.

```
pip install voila

pip install seaborn

import pandas as pd # for data manipulation
import numpy as np # for numerical analysis

# For plottling graphs
import seaborn as sns
import matplotlib.pyplot as plt

# for saving tools
import joblib

# Setting Plotting Settings
%matplotlib inline
sns.set_style("darkgrid")
```

Import Dataset

```
parkinsons = pd.read_csv("Voice_Dataset.csv")

# Checking First 5 rows of data
parkinsons.head()
```

{"type":"dataframe","variable_name":"parkinsons"}

```
pip install IPyWidgets

pip install IPython

!pip install pydub

from pydub import AudioSegment
import pandas as pd
import numpy as np
import ipywidgets as widgets
```

```
# Create a file upload widget
upload_widget = widgets.FileUpload(
    accept='.wav',  # Filter for .wav files
    multiple=False # Allow only one file to be selected
)

def extract_audio_features(file_path):
    # Load the audio file
    audio = AudioSegment.from_wav(file_path)
    samples = np.array(audio.get_array_of_samples())

    # MDVP:Fo(Hz) - Average vocal fundamental frequency
    mdvp_fo_hz = np.mean(samples)

    # MDVP:Fhi(Hz) - Maximum vocal fundamental frequency
    mdvp_fhi_hz = np.max(samples)

    # MDVP:Flo(Hz) - Minimum vocal fundamental frequency
    mdvp_flo_hz = np.min(samples)

    # MDVP:Jitter(%), MDVP:Jitter(Abs), MDVP:RAP, MDVP:PPQ, Jitter:DDP -
Several measures of variation in fundamental frequency
    jitter_percent = 0
    jitter_abs = np.mean(np.abs(np.diff(samples)))
    rap = np.mean(np.abs(np.diff(samples, 2)))
    ppq = np.mean(np.abs(np.diff(samples, 2))) / len(samples) * 100
    jitter_ddp = np.mean(np.abs(np.diff(samples, 2))) * 100

    # MDVP:Shimmer, MDVP:Shimmer(dB), Shimmer:APQ3, Shimmer:APQ5,
MDVP:APQ, Shimmer:DDA - Several measures of variation in amplitude
    shimmer = np.mean(np.abs(np.diff(np.abs(np.diff(samples)))))
    shimmer_db = 20 * np.log10(np.mean(np.abs(np.diff(samples))))
    apq3 = np.mean(np.abs(np.diff(samples, 3)))
    apq5 = np.mean(np.abs(np.diff(samples, 5)))
    apq = np.mean(np.abs(np.diff(samples))) / len(samples) * 100
    dda = np.mean(np.abs(np.diff(samples)))

    # NHR, HNR - Two measures of the ratio of noise to tonal components in the voice
    nhr = np.sum(np.abs(samples[:int(len(samples) / 2)])) /
np.sum(np.abs(samples[int(len(samples) / 2):]))
    hnr = np.sum(np.abs(samples)) / np.sum(np.abs(samples - np.mean(samples)))


    # RPDE, D2 - Two nonlinear dynamical complexity measures
    rpde = np.mean(np.abs(np.diff(np.diff(samples)))) /
np.mean(np.abs(np.diff(samples)))
    d2 = np.mean(np.abs(np.diff(np.diff(samples, 2)))) /
np.mean(np.abs(np.diff(samples, 2)))
```

```
    # DFA - Signal fractal scaling exponent
    dfa = np.mean(np.abs(np.diff(samples)))

    # spread1, spread2, PPE - Three nonlinear measures of fundamental frequency
variation
    spread1 = np.mean(np.abs(np.diff(samples))) / np.std(samples)
    spread2 = np.mean(np.abs(np.diff(samples, 2))) / np.std(np.diff(samples, 2))
    ppe = np.mean(np.abs(np.diff(samples))) / (2 * np.std(samples))

    features =

(mdvp_fo_hz,mdvp_fhi_hz,mdvp_flo_hz,jitter_percent,jitter_abs,rap,ppq,jitter_ddp,sh
immer,shimmer_db,apq3,apq5,apq,dda,nhr,hnr,rpde,dfa,spread1,spread2,d2,ppe)
    features = list(features)
    return features

def predict(features):
    input_data = [float(value) for value in features]
    # change input data into numpy array
    input_data_as_np_array = np.asarray(input_data)

    # reshape the numpy array
    input_reshaped = input_data_as_np_array.reshape(1,-1)

    # standardize the input data
    standard_data = scaler.transform(input_reshaped)
    prediction = model.predict(standard_data)
    print(prediction)

    if prediction[0] == 0:
            result = print('The person is healthy')

    else:
            result = print("The person has Parkinson's disease")

    return result

# Create a button widget
button_widget = widgets.Button(description="predict")

# Create the output widget
output_widget = widgets.Output()

# Define a function to handle button clicks
def on_button_click(button):
    with output_widget:
        # Clear previous output
        output_widget.clear_output()

        # Get the file path of the uploaded file
```

```
        file_path = next(iter(upload_widget.value.keys()))

        # Extract audio features
        features = extract_audio_features(file_path)
        result = predict(features)

        # Print the extracted features
        print(features)
        print(result)

# Connect the button click event to the function
button_widget.on_click(on_button_click)

# Display the widgets
display(upload_widget, button_widget, output_widget)
```

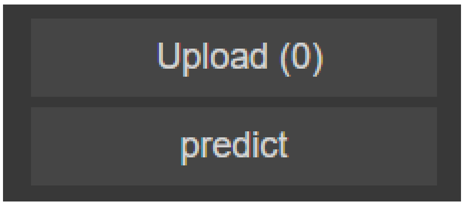


Fig. 4.1 Button of a model

4.3 User Interfaces of Spiral/Wave Drawings and Voice Input and Live Microphone Recordings Programs

Fig.4.2 User Interface of Spiral/Wave

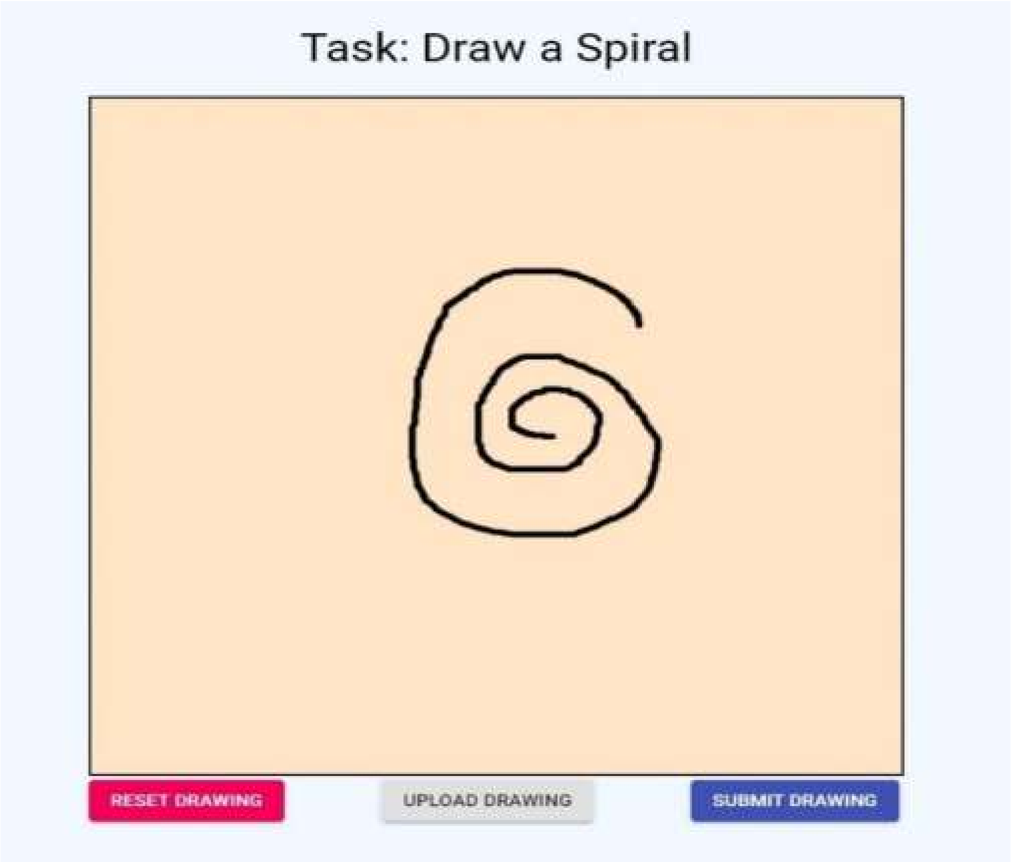


Fig.4.3 Spiral image drawn by a person

Fig.4.4 Result shown for a person

Fig.4.5 Spiral image drawn by a person

Fig.4.6 Result shown for a person

Fig.4.7 Voice Inputs given by a person

Fig.4.8 On clicking the predict button it will show the result

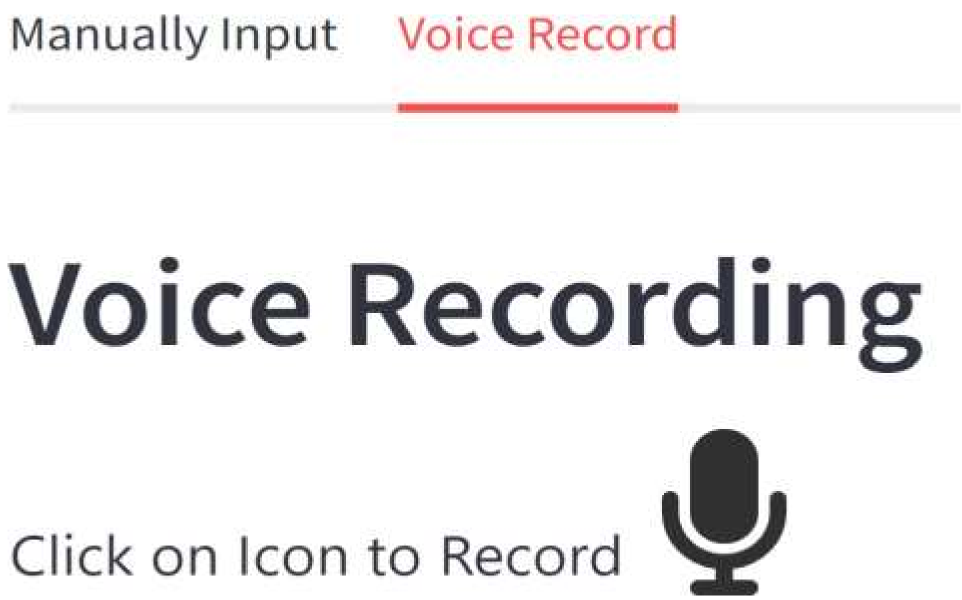


Fig.4.9 This is the interface for Voice Recordings

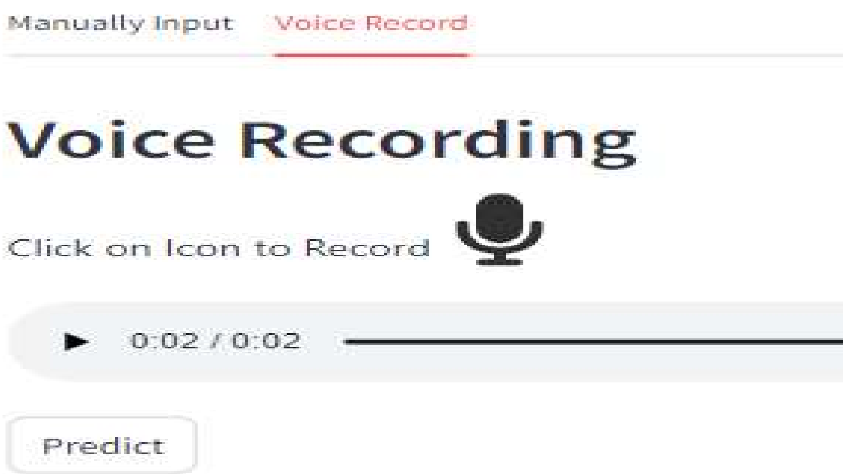


Fig.4.10 Live voice is recorded

Fig.4.11 On clicking the predict button it will show the result with its features

4.3.1 Sample Audio

- *Parkinson's Disease*: The audio might exhibit a quieter, monotone voice with tremors or fluctuations in volume and pitch compared to a healthy individual.
- *Healthy Individual*: The audio might show a clear, well-modulated voice with consistent volume and pitch.

4.4 Discussion

The effectiveness of the web application depends on the chosen machine learning models, the quality of the training data, and the accuracy of the feature extraction techniques.

- *Strengths:* This approach offers a non-invasive and potentially convenient way to screen for Parkinson's disease. Spiral/wave drawings

  and voice recordings can be easily obtained through a web interface.
- *Limitations:* The accuracy of the application might be limited by factors like user variation in drawing styles or background noise in voice recordings. Additionally, this method cannot provide a definitive diagnosis and should be used in conjunction with a medical evaluation.

CHAPTER 5: CONCLUSION

This project investigated the potential of a web application for Parkinson's disease (PD) screening using spiral/wave drawings and voice recordings. The approach utilized image and speech processing to extract features from user-provided data. Machine learning models were then trained to identify patterns in these features that might indicate PD.

The project demonstrates the feasibility of non-invasive PD screening with readily available data. Machine learning effectively analyzed drawing and voice features associated with PD. However, accuracy is likely influenced by user variability and background noise.

This application, if further developed and validated, could offer a convenient and accessible screening tool for individuals concerned about PD. Future directions include incorporating additional data modalities (tremor sensors), developing a combined drawing-voice model, and real-time feedback mechanisms. Rigorous clinical trials are essential to evaluate the application's effectiveness as a PD screening tool.

Parkinson's disease detection using spiral/wave drawings and voice recordings presents a promising avenue for early screening. Continued research and development can transform this concept into a valuable tool for managing PD and improving patient well-being.

CHAPTER 6: FUTURE SCOPE OF THE PROJECT

Parkinson's disease (PD) detection using spiral/wave drawings and voice recordings presents an exciting area for further exploration. Here, we delve into potential future directions to enhance this technology:

6.1 Multimodal Data Integration

The current approach utilizes drawings and voice recordings independently. Future iterations could explore integrating these modalities for a more robust prediction. A combined model could leverage the complementary information from both data sources, potentially leading to improved accuracy and earlier detection of PD.

6.2 Advanced Feature Extraction Techniques

Extracting informative features from drawings and voice recordings is crucial for accurate classification. Future research can delve into deeper learning techniques to automatically learn these features directly from the data. This could potentially outperform hand-crafted feature extraction methods and improve the overall performance of the application.

6.3 Sensor Integration

Incorporating additional sensors into the drawing interface holds promise. Sensors like accelerometers or gyroscopes could capture hand tremors more precisely, providing richer data for analysis. This could lead to a more objective assessment of motor skills potentially affected by PD.

6.4 Real-time Feedback and Gamification

Integrating real-time feedback during drawing or voice recording can enhance data quality. Visual or auditory cues could guide users towards a more standardized data collection process, minimizing user variability and improving the reliability of the results. Additionally, gamifying the drawing or voice recording tasks could increase user engagement and adherence.

6.5 Clinical Validation and Regulatory Approval

Rigorous clinical trials are essential to validate the application's effectiveness as a PD screening tool. These trials would involve testing the application on a large cohort of patients with confirmed PD and healthy controls. Regulatory approval would be necessary for the application to be used in a clinical setting.

6.6 Personalized Monitoring and Telehealth Integration

The application could evolve into a personalized monitoring tool for individuals with PD. By tracking changes in drawing and voice features over time, the application could potentially detect disease progression and aid in treatment adjustments. Additionally, integrating with telehealth platforms could enable remote monitoring by healthcare professionals.

6.7 Ethical Considerations

As technology advances, careful consideration of ethical implications is crucial. User privacy must be ensured, and the application should be designed to avoid generating false positives that could cause anxiety in users. Additionally, the limitations of the application as a screening tool should be clearly communicated to users, emphasizing the need for confirmation by a medical professional. By exploring these future directions, Parkinson's disease detection using spiral/wave drawings and voice recordings has the potential to become a valuable tool for early screening, disease monitoring, and improved patient outcomes.

APPENDIX A: LIST OF FIGURES

APPENDIX B: LIST OF TABLES

List of Abbreviations:

PD: Parkinson's Disease

KNN: K Nearest Neighbor

SVM: Support Vector Machine

LR: Logistic Regression

REFERENCES

[1] A. Tsanas, M. A. Little, P. E. McSharry, J. Spielman and L. O. Ramig, "Novel Speech Signal Processing Algorithms for HighAccuracy Classification of Parkinson's Disease," in IEEE Transactions on Biomedical Engineering, vol. 59, no. 5, pp. 1264- 1271, May 2012.

[2] Abós, A. et al. Discriminating cognitive status in Parkinson’s disease through functional connectomics and machine learning. Scientific reports 7, 45347 (2017).

[3] B. Zhang, J. Ren, Y. Cheng, B. Wang and Z. Wei, "Health Data Driven on Continuous Blood Pressure Prediction Based on Gradient Boosting Decision Tree Algorithm," in IEEE Access, vol. 7, pp. 32423-32433, 2019.

[4] C. B. Levine, K. R. Fahrbach, A. D. Siderowf, R. P. Estok, V. M. Ludensky, and S. D. Ross, “Diagnosis and treatment of Parkinson’s disease: a systematic review of the literature,” Evid. Rep. Technol. Assess. (Summ.), no. 57, pp. 1–4, May 2003.

[5] Center for Machine Learning and Intelligent System, Website:http://Archive.Ics.UCI.Edu/ML/Datasets/Parkinsons,2009.

[6] Dobson, A. J. & Barnett, A. An introduction to generalized linear models. (CRC press, 2008).

[7] Dr. Geetha R Ramani, G Sivagami,”Parkinson Disease Classification Using Data Mining Algorithms”, International Journal Of Computer Applications 32(9):17-22, October 2011.

[8] Gladence, L. Mary, M. Karthi, and V. Maria Anu. "A statistical comparison of logistic regression and different Bayes classification methods

for machine learning." ARPN Journal of Engineering and Applied Sciences 10, no. 14 (2015): 5947-5953.

[9] H. Ramezani and O. B. Akan, “Rate region analysis of multi-terminal neuronal nanoscale communication channel,” in 17th IEEE NANO Conf. IEEE, 2017.

[10] J. Jankovic, "Parkinson’s disease: clinical features and diagnosis" Journal of Neurology, Neurosurgery & Psychiatry, pp.368-376, 2008.

[11] K. Srilatha and V. Ulagamuthalvi “A Comparative Study on Tumor Classification “Research J. Pharm. and Tech 2019; 12(1): 407-411.

[12] Mercy Paul Selvan, Nagubadi. Navadurga, Nimmagadda. Lakshmi Prasanna, “An Efficient Model for Predicting Student Dropout using Data Mining and Machine Learning Techniques”, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8, Issue- 9S2, July 2019.

[13] Mohammad S Islam; Imtiaz Parvez; Hai Deng; Parijat Goswami,” Performance Comparison of Heterogeneous Classifiers For Detection Of Parkinson's Disease Using Voice Disorder (Dysphonia)”, International Conference On Informatics, Electronics & Vision (Iciev), Page(S):1 – 7.

[14] Mohammad shahbakhti, Danial Taherifar, Linear and Non-Linear speech features for detection of Parkinson’s Disease, BMEiCON-2013.

[15] R. Arefi Shirvan, E. Tahami, Voice analysis for detecting Parkinson’s disease using genetic algorithm and KNN classification method, Proc 18th Int Con on Biomedical Engineering, Tehran, pp. 550-555, 2011.

[16] Rätsch, G., Onoda, T. & Müller, K.-R. Sof margins for AdaBoost. Machine learning 42, 287–320 (2001).

[17] Resuldas, "A Comparison of Multiple Classification Methods For Diagnosis Of Parkinson Disease", Expert Systems With Applications, Volume 37, Issue 2, March 2010, Pages 1568-1572.

[18] Shraddha Khonde and V. Ulagamuthalvi(2019), Fusion of feature selection and Random Forest for an Anomaly based intrusion detection system , Journal of Computational and Theoretical Nanoscience , Volume 16, PP 3603-3607.

[19] Naseer, Amina, etal. "Refining Parkinson's neurological disorder identification through deep transfer learning." Neural Computing and Applications 32.3 (2020): 839-854.

[20] Kamble, Megha, Prashant Shrivastava, and Megha Jain. "Digitised spiral drawing classification for Parkinson's disease diagnosis. Measurement: Sensors 16 (2021): 100047.

[21] Priya Goyal, Piotr Doll, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, AapoKyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: training imagenet in 1 hour. arXiv preprint arXiv:1706.02677, 2017

[22] Akter L. Early Identification of Parkinson's Disease from Hand-drawn Images using Histogram of Oriented Gradients and Machine Learning Techniques. In2020 Emerging Technology in Computing, Communication and Electronics (ETCCE) 2020 Dec 21 (pp. 1-6)IEEE.

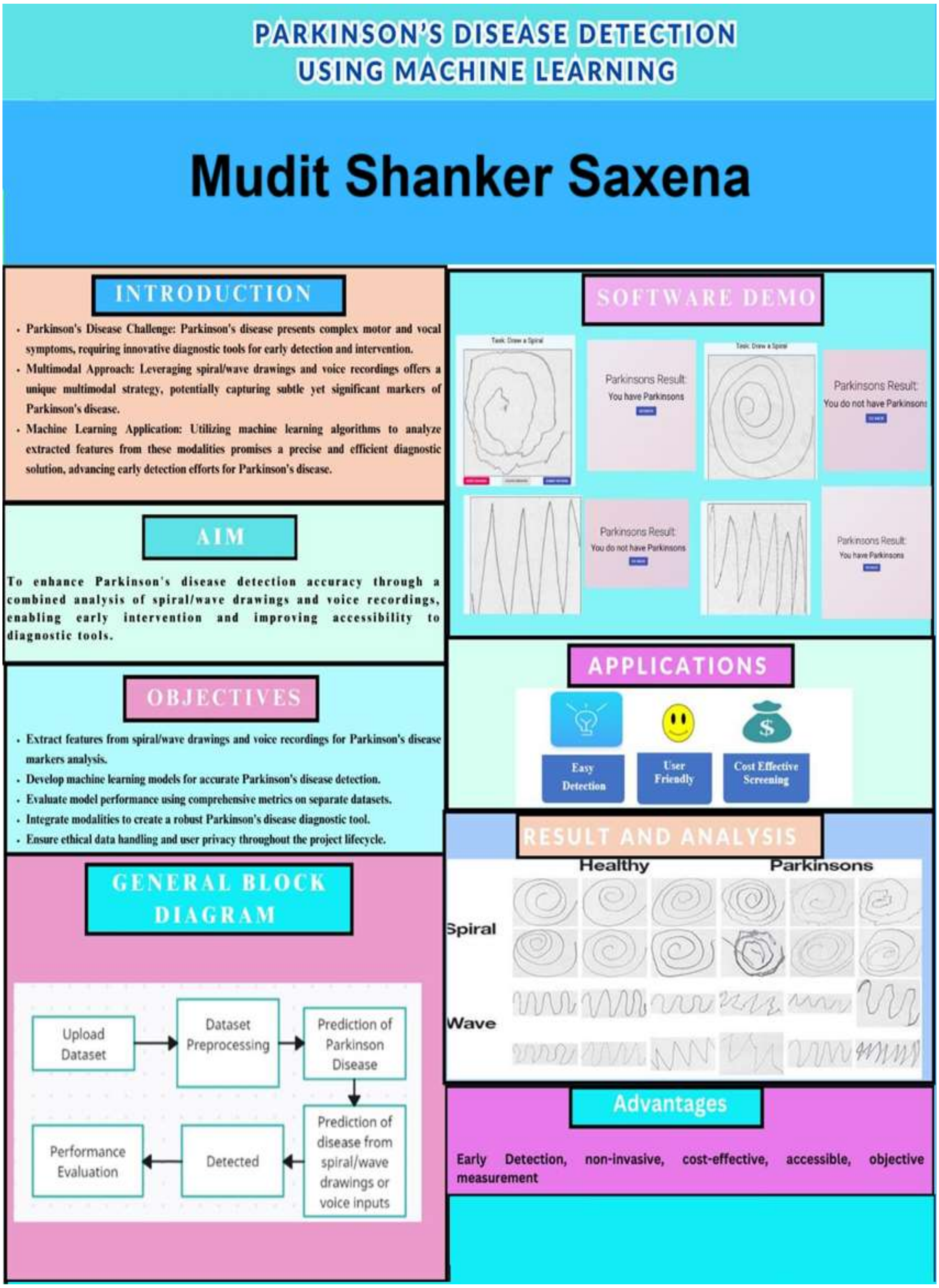
[23] Veer, Karan, and S. K. Pahuja. "Gender based assessment of gait rhythms during dual-task in Parkinson's disease and its early detection." Biomedical Signal Processing and Control 72 (2022): 103346.

[24] Shaban, M. Deep Learning for Parkinson's Disease Diagnosis: A Short Survey. *Computer* **2023**, *12*, 58

[25] Chatterjee, K.; Kumar, R.P.; Bandyopadhyay, A.; Swain, S.; Mallik, S.; Li, A.; Ray, K. PDD-ET: Parkinson's Disease Detection Using ML Ensemble Techniques and Customized Big Dataset. *Information* **2023**, *14*, 502.

[26] Zafeiropoulos, N.; Bitilis, P.; Tsekouras, G.E.; Kotis, K. Evaluating Ontology-Based PD Monitoring and Alerting in Personal Health Knowledge Graphs and Graph Neural Networks. *Information* **2024**, *15*, 100.

COMPLETE PROJECT POSTER



PROJECT PUBLICATION CERTIFICATES


ISSN 2320-9186

CERTIFICATE OF ACCEPTANCE

Global Scientific Journal Publications (GSJ)

THIS IS TO CERTIFY THAT OUR REVIEW BOARD HAS ACCEPTED RESEARCH PAPER OF

Mudit Shanker Saxena

Parkinson's Disease Detection Using Machine Learning

May 9,2024

Visit us at: **www.globalscientificjournal.com**

Editor in Chief