Detecting Parkinson's Disease using KNN and Naive Bayes' algorithms

Graduate project report

Submitted to

Lamar University

In partial fulfillment of the requirements for the

Masters in computer science

Submitted By:

Team members

Rajasekhar Reddy Peram

Indu Pasham

Naga Jahnavi Gaddam

Submitted to:

Graduate Committee

Dr. Timothy Roden

Dr. Kami Makki

Dr. Bo Sun

Spring 2024

ACKNOWLEDGEMENT

Firstly, we would like to convey our gratitude to the Department of Computer Science for giving us such an opportunity to present our skills and ability and we have tried our best to complete our project successfully.

We would like to thank our project supervisor Dr. Timothy Roden for his regular guidance and valuable suggestions.

Besides our supervisor, we are grateful to other committee members, Dr. Kami Makki and Dr. Bo Sun for their insightful comments.

Finally, we would like to thank our families and friends for their support and encouragement during the course of our work.

ABSTRACT

Parkinson’s disease is a movement disorder of the nervous system that worsens over time. As nerve cells (neurons) in parts of the brain weaken or are damaged or die, people may begin to notice problems with movement, tremors, stiffness in the limbs or the trunk of the body, or impaired balance. As these symptoms become more obvious, people may have difficulty walking, talking, or completing other simple tasks. Not everyone with one or more of these symptoms has Parkinson’s Disease, as the symptoms appear in other diseases as well

This detailed report investigates the use of K-Nearest Neighbors (KNN) and Naive Bayes algorithms for detecting Parkinson’s disease using voice features. By analyzing vocal recordings, this study assesses the algorithms' efficiency, accuracy, and viability in medical diagnostics.

The findings suggest that with optimal parameter tuning, both algorithms can significantly enhance early diagnosis capabilities, which is crucial for timely treatment and management of Parkinson's disease.

The project involves a comprehensive study and implementation of Data Mining algorithms to enhance the detection of Parkinson's disease using vocal biomarkers. The primary technologies and methodologies employed include Python for data manipulation and model implementation, and libraries such as scikit-learn for Data Mining algorithms.

This approach leverages the K-Nearest Neighbors (KNN) and Naive Bayes algorithms to analyze and classify voice recording data from individuals, potentially identifying early signs of Parkinson's disease. Data preprocessing, crucial for achieving optimal performance from the Data Mining models, involves techniques such as feature scaling, normalization, and dimensionality reduction. The project not only focuses on the accuracy of the detection algorithms but also on their efficiency and scalability to handle real-world dataset.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Parkinson's disease is a progressive neurological disorder characterized by the degradation of motor functions due to the decline in dopamine-producing neurons in the substantia nigra part of the brain. Early signs include difficulty with movement, which worsens over time. Currently, over 90,000 new cases are identified annually in the United States alone, though the actual number might be higher due to frequent misdiagnoses.

Data mining, defined as the process of extracting valuable information from large datasets, plays a crucial role in various scientific domains, particularly in medical research. By employing sophisticated algorithms to analyze patterns and predict future events, data mining facilitates a deeper understanding of diseases like Parkinson's and enhances the strategies to combat them.

In this project, we apply data mining techniques—specifically the Naive Bayes and K-Nearest Neighbor (KNN) algorithms—to the UCI ML Parkinson's dataset with the aim of predicting the onset of Parkinson's disease. Naive Bayes is known for its efficacy with large datasets, whereas KNN is celebrated for its simplicity and effectiveness in classification tasks. However, selecting the optimal parameters for these models is crucial to avoid overfitting or underfitting, thus ensuring the precision and reliability of predictions.

This report outlines the objectives of our study, which focuses on the predictive power of these algorithms within the context of medical data analytics. We aim to demonstrate how data mining can significantly contribute to early disease detection, which is paramount in improving patient outcomes.

The subsequent chapters of this report will delve into the literature review, detailing previous work in the field, followed by a comprehensive methodology section that describes our data collection, preprocessing, and analytical techniques. Results will be analyzed, discussed, and finally, the report will conclude with reflections on the findings and suggestions for future research.

CHAPTER 2: LITERATURE SURVEY

1. "Parkinson's Disease: A Complete Guide for Patients and Families" by William J. Weiner, Lisa M. Shulman, and Anthony E. Lang

This book provides comprehensive information on Parkinson's disease, covering symptoms, diagnosis, and treatment options, which helped to understand the clinical aspects that could be relevant when applying data mining in predicting disease onset.

This book provides a deep understanding of Parkinson's disease, which has been an essential part for framing data mining project. Knowledge about the progression, symptoms, and impact of the disease can help in selecting relevant features from the dataset and in interpreting the results of the data mining process. It also guided us in the formulation of research questions and objectives that are aligned with clinical realities.

2. "Machine Learning for Absolute Beginners" by Oliver Theobald

This book breaks down the basics of machine learning algorithms including Naive Bayes and KNN, these three are common machine learning algorithms, each with distinct characteristics and applications.

Naive Bayes is a probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions between the features. It is particularly suited for large datasets and can handle many input variables.

For Parkinson's disease, Naive Bayes can be applied to classify patients as likely or unlikely to develop the disease based on a set of predictors (such as voice measurements, tremor frequencies, etc.). Due to its probabilistic nature, it can handle uncertainties in diagnosis and can provide probabilities of having the disease rather than just classifications.

K-Nearest Neighbors (KNN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether KNN is used for classification (the most common class among the k-nearest neighbors) or regression (the average of the values of the k nearest neighbors).

KNN is used to predict whether a patient has Parkinson's based on the similarity of their symptoms and test results to known cases. It's particularly useful where the prediction needs to be made based on closely resembling examples, thus providing highly contextual results based on the input data

3. "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems" by Aurélien Géron

This book provides a practical introduction to machine learning with Python, focusing on the use of Scikit-learn for various machine learning tasks. It covers fundamental machine learning principles and provides step-by-step guides to implementing machine learning models, including preprocessing data, choosing the right algorithm, and tuning your models. Here's how it could be used in the project:

Comprehensive Coverage: It covers a wide range of topics from basic machine learning to more advanced techniques, all using Scikit-learn. This includes discussions on preprocessing data, feature selection, regression, classification, clustering, and dimensionality reduction.

Model Evaluation and Tuning: It provides guidance on evaluating model performance and fine-tuning models using cross-validation and grid search, which are crucial for optimizing models in medical applications.

4. "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Python" by Wes McKinney

Numpy and pandas is used in below ways in our project:

1. Data Cleaning and Preparation

In medical research, including studies on Parkinson's disease, data often comes from multiple sources and can contain inconsistencies, missing values, or errors. Using Pandas, we can:

Handle Missing Data: Implement various techniques to fill or ignore missing values, depending on what's appropriate for your analysis.

Remove Outliers: Detect and exclude data points that represent measurement errors or are not representative of the general population.

2. Data Transformation

Transforming data is crucial to align it with the needs of machine learning algorithms or specific analysis methods:

Normalization and Scaling: Use Pandas alongside Scikit-learn to scale numerical data, ensuring that features contribute equally to model training without bias due to their scale.

Encoding Categorical Data: Convert categorical data into a format that can be used by machine learning algorithms (e.g., one-hot encoding).

3. Exploratory Data Analysis (EDA)

Before diving into complex analyses or building predictive models, initial exploration of the data is essential:

Statistical Summaries: Use Pandas to generate descriptive statistics to understand the central tendencies, dispersion, and shape of your data distributions.

Correlation Analysis: Identify relationships or associations between different variables that might indicate relevant patterns, such as potential risk factors for Parkinson’s disease.

4. Data Visualization

Pandas integrates well with libraries like **Matplotlib** and **Seaborn** to provide insights through visualizations:

Trend Analysis: Plot time series data to visualize the progression of Parkinson's symptoms over time.

Comparison Charts: Create bar charts or box plots to compare measurements across different patient groups (e.g., patients at different stages of the disease).

CHAPTER 3: AIM AND SCOPE OF THE PROJECT

3.1 AIM OF THE PROJECT

Our project is motivated by the urgent need for advancements in the early detection of neurological disorders, particularly Parkinson's disease. The onset of Parkinson's disease often precedes visible symptoms by many years, making early and accurate diagnosis critical for effective intervention. Current diagnostic methods are largely based on clinical assessments, which may not capture the subtleties of the disease in its early stages. Our research aims to harness the predictive power of machine learning to fill this gap, offering a more objective and quantifiable approach to detecting early signs of Parkinson's disease using vocal biomarkers

The primary objective of this project is to develop and refine a predictive model that utilizes two well-regarded machine learning algorithms: K-Nearest Neighbors (KNN) and Naive Bayes. These algorithms have been chosen for their ability to effectively handle binary classification problems and their suitability for small to medium-sized datasets, which are common in medical research. By applying these algorithms to voice recording data, we aim to identify subtle patterns that may indicate the early stages of Parkinson's disease, long before clinical symptoms become apparent. This project not only seeks to improve diagnostic accuracy but also to provide a non-invasive, cost-effective, and readily deployable method to screen at-risk populations.

Ultimately, this project strives to bridge the gap between current research in neurodegenerative diseases and practical, clinical applications. By developing a robust model that can predict Parkinson's disease from easily obtainable vocal recordings, we aim to facilitate earlier and more personalized treatment strategies, potentially slowing the progression of the disease. This aligns with the broader goal of leveraging cutting-edge data mining techniques to enhance healthcare outcomes and adapt to the evolving needs of a diverse patient population. Our approach promises to revolutionize the way Parkinson's disease is diagnosed, making it more accessible and less subjective, thus aligning with modern healthcare's shift towards precision medicine.

3.2 SCOPE AND OBJECTIVE

This project is dedicated to advancing the capabilities of early Parkinson's disease detection through the application of machine learning algorithms on vocal biomarker data. The scope of our research includes the collection and preprocessing of voice recording data, the implementation and tuning of K-Nearest Neighbors (KNN) and Naive Bayes algorithms, and the validation of these models against established diagnostic criteria. By focusing on non-invasive, audio-based biomarkers, our approach seeks to develop a predictive model that is both accessible and efficient, suitable for early screening in a clinical setting or even remotely.

The primary objective is to accurately classify individuals as 'at risk' of Parkinson's disease or 'not at risk' based on their vocal features. To achieve this, we will:

- Develop a comprehensive dataset by aggregating voice recordings from individuals diagnosed with Parkinson's as well as healthy controls.
- Implement data preprocessing techniques to enhance the quality and efficacy of the machine learning models.
- Apply KNN and Naive Bayes algorithms to analyze the processed data, optimizing parameters to maximize both the sensitivity and specificity of the predictions.
- Evaluate the performance of these models using appropriate statistical measures such as accuracy, precision, recall, and F1 score.

Upon successful development and validation, the predictive model will be proposed for integration into clinical workflows to assist with early diagnosis. The long-term objective includes refining the model through continuous learning and adaptation as new data becomes available, ensuring its relevance and accuracy over time. Additionally, future research may explore the integration of other biomarkers and diagnostic modalities to create a more comprehensive diagnostic tool. The ultimate goal is to provide a non-invasive, cost-effective solution that enhances the early detection and management of Parkinson’s disease, potentially improving outcomes through timely intervention.

3.3 SYSTEM REQUIREMENTS

3.3.1 Hardware Requirements

The selection of hardware is very important in the existence and proper working of any software. When selecting the hardware, the size and requirements are also important to run the software. The minimum hardware requirements are as follows:

- Processor: Intel CORE i3 or higher
- RAM: 4 GB or higher
- Disk space: minimum 256 GB

3.3.2 Software Requirements

- **Operating System:** Windows, MacOS, or Linux
- **Python**
- **Scikit-learn** (for machine learning algorithms)
- **NumPy** (for numerical computing)
- **Pandas** (for data manipulation and analysis)
- **Matplotlib and Seaborn** (for data visualization)
- **Code Editor:** Visual Studio Code, PyCharm, or any preferred IDE that supports Python
- **PIP** (Python package-management system used to install and manage software packages)

3.4 SOFTWARE USED

3.4.1 Python Language

Python language is a high-level, dynamically typed one that is among the most popular general-purpose programming languages. Python is an Interpreted, object-oriented, and high-level programming language. It is called an interpreted language as its source code is compiled to bytecode which is then interpreted. Python's features, among other things, are what make it popular. For instance, it supports dynamic typing and dynamic binding.

3.4.2 Features of Python

Open Source:

Python is completely free, which allows anyone to download and use it for their projects without cost. This open-source nature also means a large community continuously improves the language and its libraries.

Ease of Use:

Python's syntax is clear and intuitive, making it an excellent choice for developers at all levels of experience, from beginners to experts. This readability makes Python particularly appealing for writing and maintaining complex machine learning algorithms.

Extensive Libraries:

Python’s ecosystem includes numerous libraries specifically designed for data science and machine learning, such as Pandas for data manipulation, NumPy for numerical data, Scikit-learn for machine learning algorithms, Matplotlib and Seaborn for data visualization, which are essential for analyzing and interpreting Parkinson’s disease data.

Cross-Platform Compatibility:

Python runs on all major operating systems — Windows, Mac OS, and Linux. This makes it easy to develop and deploy Python applications across different environments, ensuring that your project can be run without modifications on any system.

Integrated Development Environment (IDE) Support:

Python is supported by multiple IDEs, including PyCharm and Visual Studio Code, which offer powerful coding, debugging, and testing tools to improve the development environment for data science and machine learning projects.

3.4.3 Python Libraries

Scikit-learn

Scikit-learn is a powerful open-source library that provides a wide range of tools for machine learning. It is built on top of NumPy and SciPy and offers various classification, regression, clustering algorithms, and utilities for model fitting, data preprocessing, model selection, and evaluation. Scikit-learn is renowned for its robustness and ease of use, making it a standard tool for machine learning tasks in both academia and industry.

NumPy

NumPy, short for Numerical Python, is a fundamental library for high-performance scientific computing and data analysis in Python. It provides support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. Serving as the foundational library for many other Python data science libraries like Pandas and SciPy, NumPy is crucial for numerical computations.

Pandas

Pandas is an essential library for data manipulation and analysis in Python. It provides data structures and operations for manipulating numerical tables and time series. Its primary data structure, the DataFrame, allows for the storage and manipulation of tabular data in rows of observations and columns of variables. Pandas is integral to data cleaning, preparation, and analysis tasks, making it indispensable in data science workflows.

Matplotlib

Matplotlib is a comprehensive library for creating static, interactive, and animated visualizations in Python. It provides an object-oriented API that allows users to embed plots into applications using general-purpose GUI toolkits. Matplotlib is highly customizable and can produce a wide variety of plots and charts, making it a tool of choice for data visualization.

Seaborn

Seaborn is a Python data visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics. It is particularly well-suited for exploring and understanding complex datasets. With Seaborn, users can create heat maps, time series, violin plots, and various other types of charts that make complex data more accessible and interpretable.

3.5 APPLICATION DEVELOPMENT PLATFORM

3.5.1 Visual Studio Code

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.It can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python and C++.

CHAPTER 4: METHODOLOGY

4.1 Data Collection

- The dataset used in this project is sourced from the UCI Machine Learning Repository, specifically the Parkinson's dataset, which can be found Dataset. This dataset is composed of a range of biomedical voice measurements from 31 people, 23 with Parkinson's disease (PD). Each column in the table is a particular voice measure, and each row corresponds one of 195 voice recording from these individuals ("name" column). The main aim of the data is to discriminate healthy people from those with PD, according to "status" column which is set to 0 for healthy and 1 for PD. The data consists of several attributes derived from voice recordings, such as pitch, amplitude, vocal fold, and other vocal signal characteristics that are useful in distinguishing healthy individuals from those with Parkinson's disease.

4.2 Data Preprocessing

Given the nature of the data and the specific requirements of machine learning models, preprocessing is a critical step:

Removal of Highly Correlated Features: In the Naive Bayes approach, there's a preprocessing step where the correlation matrix of the dataset's features is calculated. Features that show a high correlation (greater than 0.90) with other features are removed. This process helps in reducing multicollinearity, which can adversely affect model performance. By removing these highly correlated features, the dataset is simplified, potentially improving the effectiveness of the model without significant loss of information.

Data Splitting: For both KNN and Naive Bayes models, the dataset is divided into training and testing sets. This is a crucial step as it allows the model to learn from one subset of the data (training set) and then validates the learned model on a separate subset (testing set) that was not used during the training phase. This ensures that the performance metrics are reflective of the model's ability to generalize to new data.

In the KNN approach, data is manually partitioned into training and testing subsets based on a percentage split, allowing for flexibility in how much data is used for training versus testing.

For the Naive Bayes model, a function from scikit-learn (train_test_split) is typically used to automate this process, ensuring a random and unbiased division of data.

Feature Extraction and Selection:

In the KNN model, feature extraction involves selecting the relevant columns from the dataset as features, excluding typically the first column (if it contains non-numeric data like IDs or names) and the last column if it represents the label. This process transforms the raw data into a structured format that the algorithm can utilize for distance calculation and nearest neighbor analysis.

In the Naive Bayes model, feature selection not only includes choosing which features to keep but also actively involves dropping features that are not necessary, particularly those that are highly correlated with others. This selection process is crucial to avoid redundancy in the feature set and to enhance the predictive accuracy of the model.

4.3 Implementation of KNN and Naive Bayes

K-Nearest Neighbors (KNN): The K-Nearest Neighbors (KNN) algorithm is a simple, yet powerful machine learning method used for classification and regression. It uses a straightforward approach where the output is class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.

It calculates the distance between points in the feature space. Common distance measures used in KNN include Euclidean, Manhattan, and Hamming distance.

Example for explaining of KNN Algorithm:

Suppose you have a dataset of fruits characterized by their weight and texture. You want to classify whether a new fruit is likely an apple or an orange based on these features.

Dataset Example:

Apple: (150 grams, Smooth)

Apple: (130 grams, Smooth)

Orange: (180 grams, Rough)

Orange: (160 grams, Rough)

You are given a new fruit that weighs 145 grams and has a smooth texture, and you need to decide whether it's an apple or an orange using KNN with k=3.

Steps:

Distance Calculation: Calculate the distance (e.g., using Euclidean distance) between the new fruit and each fruit in the dataset.

Distance to Apple 1 = sqrt((150-145)² + (Smooth-Smooth)²) = 25 + 0 = 25

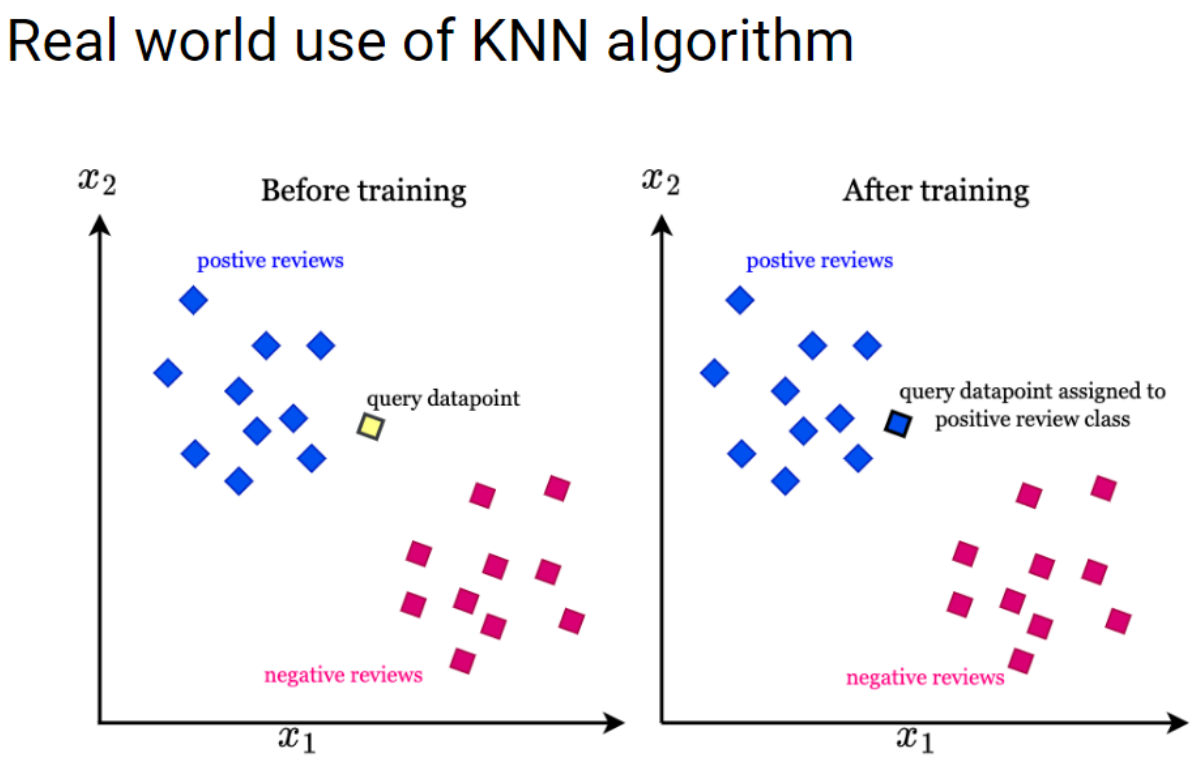
Distance to Apple 2 = sqrt((130-145)² + (Smooth-Smooth)²) = 225 + 0 = 225

Distance to Orange 1 = sqrt((180-145)² + (Rough-Smooth)²) = 1225 + 1 = 1226

Distance to Orange 2 = sqrt((160-145)² + (Rough-Smooth)²) = 225 + 1 = 226

Finding Nearest Neighbors: Sort the fruits by their distance to the new fruit. The closest three, assuming a categorical simplification for texture (where Smooth = 0 and Rough = 1), are probably Apple 1, Orange 2, and Apple 2.

Majority Voting: Among the k=3 nearest neighbors (Apple 1, Orange 2, Apple 2), two are apples and one is an orange. The majority vote is "apple," so the new fruit is classified as an apple



Naive Bayes: The Naive Bayes algorithm is a collection of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of a feature given the value of the class variable. Naive Bayes classifiers rely on Bayes' theorem, which uses prior knowledge of conditions that might be related to an event to predict the likelihood of the event occurring.

It is a probabilistic classifier based on Bayes' theorem, which is one of the simplest yet effective approaches for classification, especially in text classification and medical diagnostic applications. Here's how the formula for Naive Bayes is structured and what each component of the formula represents:

Bayes' Theorem:

P(A|B) = (P(B|A) × P(A)) / P(B)

Where:

- P(A|B) is the **posterior probability**: The probability of hypothesis A given the data B.

- P(B|A) is the **likelihood**: The probability of observing the data B given that the hypothesis A is true.

- P(A) is the **prior probability**: The initial probability of the hypothesis A before observing the data.

- P(B) is the **evidence**: The total probability of observing the data.

Example for explaining of Naive Bayes Algorithm:

Feature Encoding: Since Naive Bayes works with probabilities, we need to handle categorical data (like texture) by encoding it. We can assign 'Smooth' as 0 and 'Rough' as 1.

Calculate Prior Probabilities:

P(Apple) = Number of Apples / Total Number of Fruits = 2/4 = 0.5

P(Orange) = Number of Oranges / Total Number of Fruits = 2/4 = 0.5

Calculate Likelihoods:

For continuous data like weight, assuming a Gaussian distribution:

Mean weight of Apples = (150+130)/2 = 140 grams

Mean weight of Oranges = (180+160)/2 = 170 grams

Calculate variance for weight for both fruits for use in the Gaussian probability density function.

For categorical data like texture, calculate the frequency of each texture within each class:

P(Smooth | Apple) = Number of Smooth Apples / Total Apples = 2/2 = 1

P(Rough | Apple) = 0 (since no apple is rough)

P(Smooth | Orange) = 0 (since no orange is smooth)

P(Rough | Orange) = Number of Rough Oranges / Total Oranges = 2/2 = 1

Calculate Posterior Probabilities for the new fruit (145 grams, Smooth):

For Apple:

P(Weight = 145 | Apple) using Gaussian PDF with mean = 140, variance of Apple weights

P(Texture = Smooth | Apple) = 1

P(Apple | Features) ∝ P(Weight = 145 | Apple) * P(Texture = Smooth | Apple) * P(Apple)

For Orange:

P(Weight = 145 | Orange) using Gaussian PDF with mean = 170, variance of Orange weights

P(Texture = Smooth | Orange) = 0

P(Orange | Features) ∝ P(Weight = 145 | Orange) * P(Texture = Smooth | Orange) * P(Orange)

Decision:

Since P(Texture = Smooth | Orange) = 0, the posterior probability for Orange will be zero, indicating the model predicts the fruit cannot be an orange based solely on the texture being smooth.

Compare P(Apple | Features) with P(Orange | Features). The higher probability determines the class. In this case, it's very likely the prediction would be Apple, since the posterior probability for Orange is zero due to the texture mismatch.

4.4 Optimization Techniques

To enhance the accuracy and precision of the models, optimal parameter selection was crucial.

For the **Naive Bayes algorithm**, the percentage of training data was iteratively adjusted within the range of 70% to 90%. This approach was aimed at finding the training split that maximized the model's performance, balancing the learning from the training data against its generalization to new data.

In the **K-Nearest Neighbors (KNN) algorithm**, the optimal value of k was determined by iterating through values from 1 to 20. This process ensures a balanced model by avoiding overfitting with smaller values of k and underfitting with larger values. Additionally, the training data percentage was varied from 60% to 80% to identify the most effective training split.

CHAPTER 5: RESULTS AND PERFORMANCE ANALYSIS

This chapter details the results obtained from the implementation of the K-Nearest Neighbors (KNN) and Naive Bayes algorithms to detect Parkinson’s disease using voice measurement data.

5.1 Naive Bayes Results

The Naive Bayes model, trained on 75% of the data, provided the following results:

Confusion Matrix:

- True Positives (TP): 32
- True Negatives (TN): 6
- False Positives (FP): 4
- False Negatives (FN): 7

Accuracy: 77.55%

Classification Report:

Precision=TP/(FP+TP))

Recall=TP/(FN+TP)

F1 Score=2×[(Precision*Recall)/(Precision+Recall)]

For Class 0 (presumably the negative class, indicating the absence of Parkinson's):

- Precision: 0.46
- Recall: 0.60
- F1-score: 0.52
- Support: 10 (number of actual instances in this class)

For Class 1 (presumably the positive class, indicating the presence of Parkinson's):

- Precision: 0.89
- Recall: 0.82
- F1-score: 0.85
- Support: 39 (number of actual instances in this class)

Overall:

- **Accuracy:** 0.78
- Macro Average Precision: 0.68
- Macro Average Recall: 0.71
- Macro Average F1-score: 0.69
- Weighted Average Precision: 0.80

- Weighted Average Recall: 0.78
- Weighted Average F1-score: 0.79

```
PS C:\Users\Indu Reddy\OneDrive\Desktop\Graduate Project> python .\Naive_Bayes.py
Confusion Matrix:
[[ 6  4]
 [ 7 32]]

Accuracy: 0.7755102040816326

Classification Report:
              precision    recall  f1-score   support

           0       0.46      0.60      0.52        10
           1       0.89      0.82      0.85        39

    accuracy                           0.78        49
   macro avg       0.68      0.71      0.69        49
weighted avg       0.80      0.78      0.79        49

PS C:\Users\Indu Reddy\OneDrive\Desktop\Graduate Project>
```

Figure 1: Output for Naive Bayes.

Correlation Heatmaps: Illustrating the relationship between different voice measurement features, highlighting which features are most significant for Parkinson's disease detection.

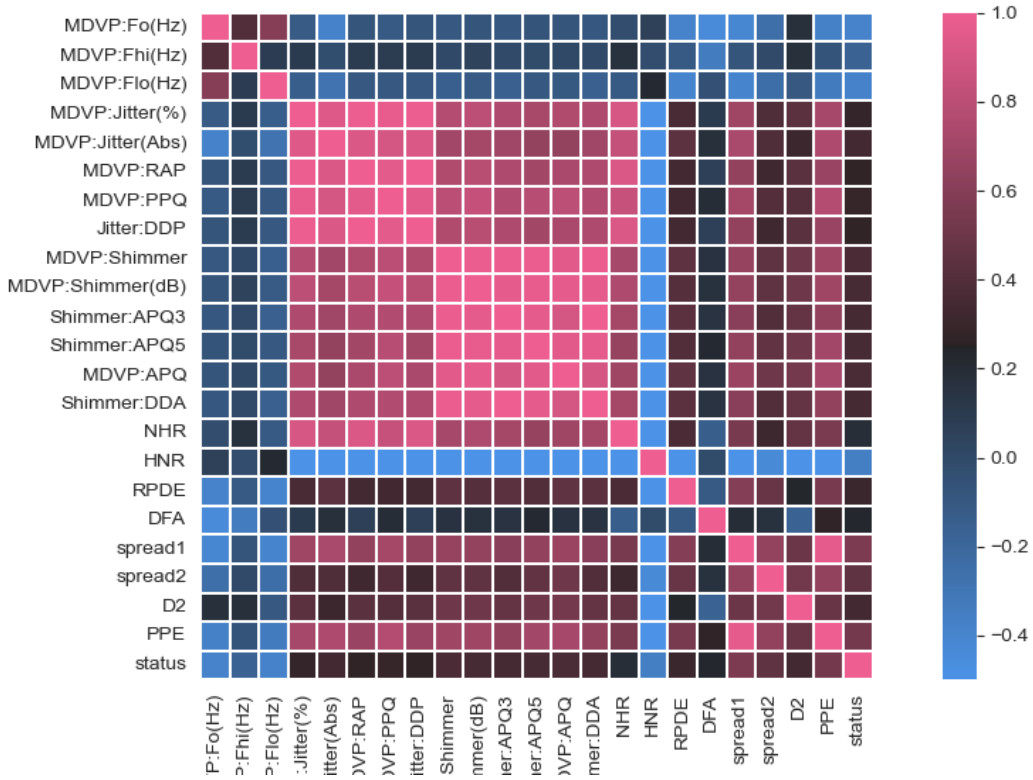


Figure 2: Correlation Heatmap Before Feature Selection.

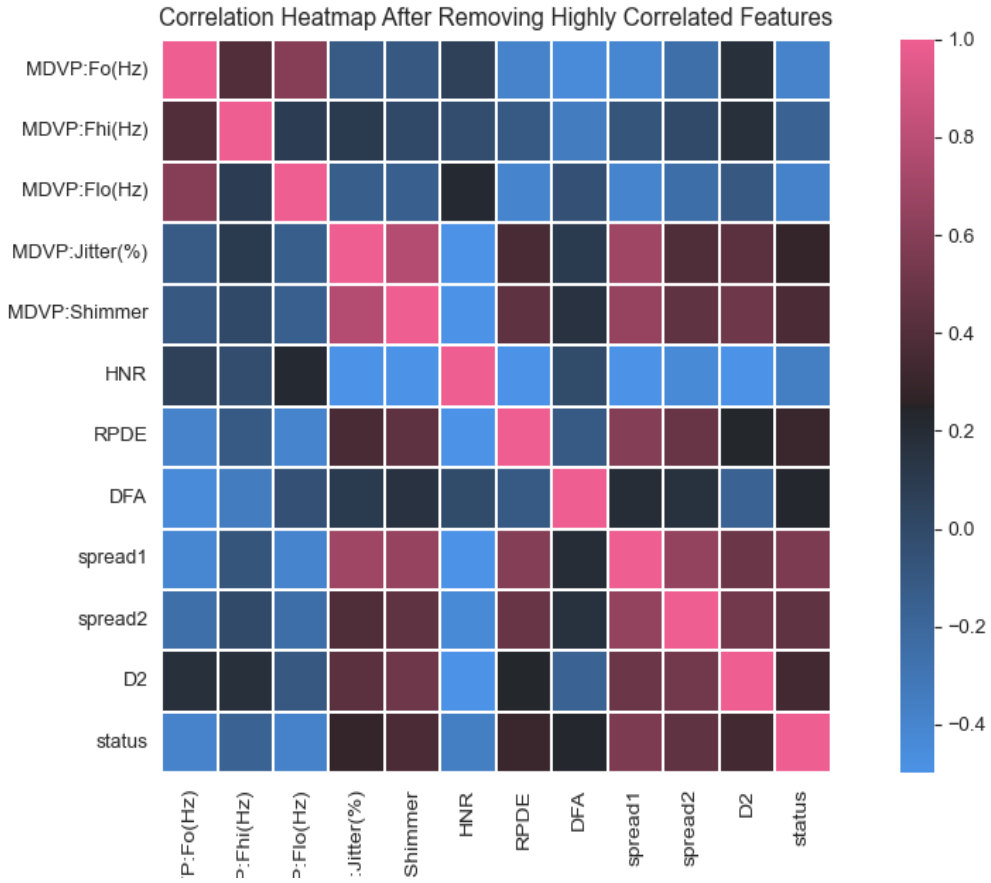


Figure 3: Correlation Heatmap After Removing Highly Correlated Features.

5.2 Optimal Naive Bayes Model Results

To get maximum accuracy, the model is optimized to train data constituted 84% of the dataset when iterated through 70% to 90%

Training Configuration: The Naive Bayes model was trained on 84% of the dataset.

Accuracy: 87.50%.

Confusion Matrix:

- True Positives (TP): 25
- True Negatives (TN): 5
- False Positives (FP): 2
- False Negatives (FN): 2

Classification Report:

- Precision for Class 0 (Negative): 0.71
- Recall for Class 0 (Negative): 0.71
- F1-Score for Class 0 (Negative): 0.71

- Support for Class 0 (Negative): 7
- Precision for Class 1 (Positive): 0.92
- Recall for Class 1 (Positive): 0.92
- F1-Score for Class 1 (Positive): 0.92
- Support for Class 1 (Positive): 25
- Overall Model Accuracy: 0.88
- Macro Average Precision: 0.82
- Macro Average Recall: 0.82
- Macro Average F1-Score: 0.82
- Weighted Average Precision: 0.88
- Weighted Average Recall: 0.88
- Weighted Average F1-Score: 0.88

The model has demonstrated high precision, particularly in identifying positive cases of Parkinson's disease (Class 1). The recall for Class 0 is notably high, which means the model is good at identifying negative cases; however, the precision for this class is lower, indicating a higher false-positive rate. Conversely, the high precision with a slightly lower recall for Class 1 suggests the model is conservative in predicting positive cases, prioritizing correctness over completeness.

```
PS C:\Users\Indu Reddy\OneDrive\Desktop\Graduate Project> python .\optimal_Naive_Bayes.py
Training Percentage: 84.00%
Accuracy: 0.875
Confusion Matrix:
[[ 5  2]
 [ 2 23]]
Classification Report:
              precision    recall  f1-score   support

           0       0.71      0.71      0.71         7
           1       0.92      0.92      0.92        25

    accuracy                           0.88        32
   macro avg       0.82      0.82      0.82        32
weighted avg       0.88      0.88      0.88        32


PS C:\Users\Indu Reddy\OneDrive\Desktop\Graduate Project>
```

Figure 4: Output of Optimal Naive Bayes.

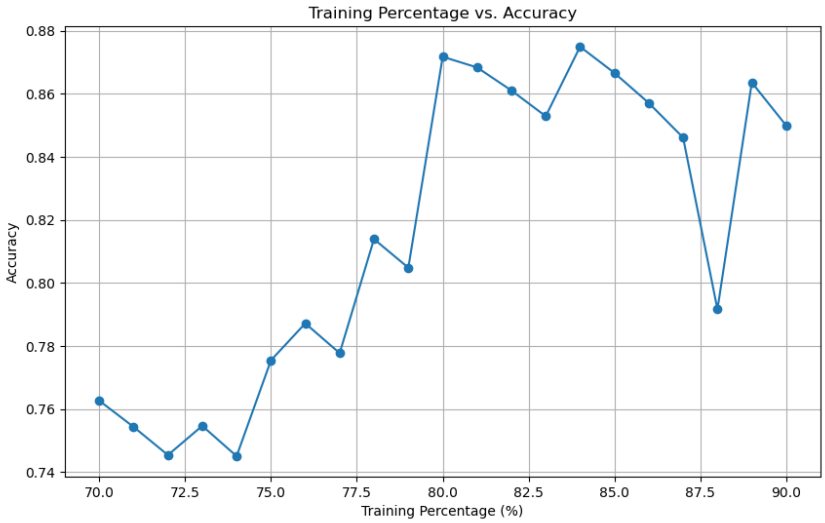


Figure 5: Training Percentage vs. Accuracy for Naive Bayes with Optimal Parameters.

5.3 K-Nearest Neighbors (KNN) Results

Using a KNN model with k set to 5 and trained on 75% of the data, the following performance metrics were achieved:

Confusion Matrix:

- True Positives (TP): 38
- True Negatives (TN): 5
- False Positives (FP): 5
- False Negatives (FN): 1

Accuracy: 87.75%

Classification Report:

- Precision for Class 0 (Negative): 0.83
- Recall for Class 0 (Negative): 0.50
- F1-Score for Class 0 (Negative): 0.62
- Support for Class 0 (Negative): 10
- Precision for Class 1 (Positive): 0.88
- Recall for Class 1 (Positive): 0.97
- F1-Score for Class 1 (Positive): 0.93
- Support for Class 1 (Positive): 39
- Overall Model Accuracy: 0.88
- Macro Average Precision: 0.86

- Macro Average Recall: 0.74
- Macro Average F1-Score: 0.78
- Weighted Average Precision: 0.87
- Weighted Average Recall: 0.88
- Weighted Average F1-Score: 0.87

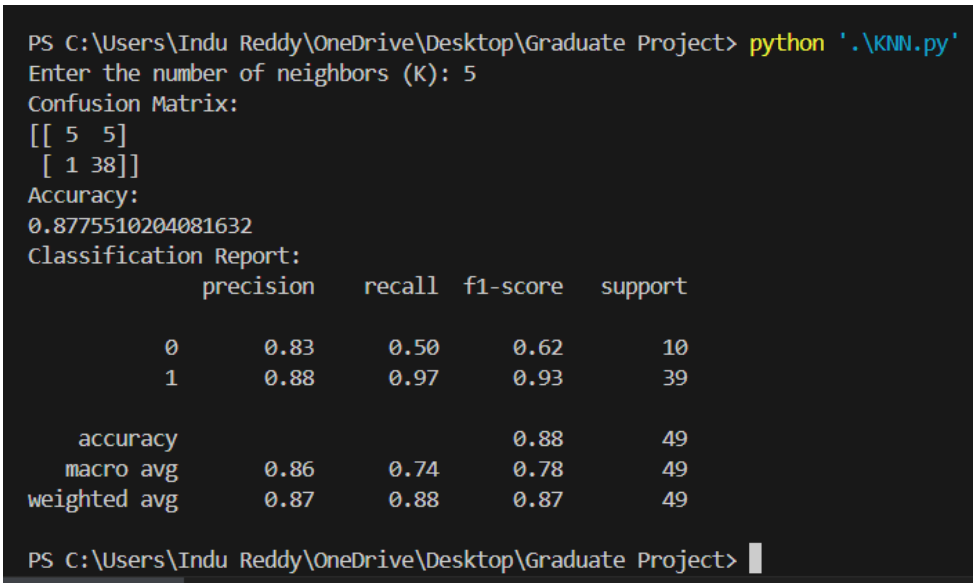


Figure 6: Output of KNN

5.4 Optimal K-Nearest Neighbors (KNN) Model Results

To get maximum accuracy, the model is optimized to train data constituted 73% of the dataset when iterated through 60% to 80% and constituted K=12 when iterated through 1 to 20.

Training Configuration: The model was trained on 73% of the dataset, which was determined to be the optimal training percentage.

K-value: The optimal number of neighbors (k) for the KNN algorithm was found to be 12.

Accuracy: 92.45%.

Confusion Matrix:

- True Positives (TP): 43
- True Negatives (TN): 6
- False Positives (FP): 4
- False Negatives (FN): 0

Classification Report:

- Precision for Class 0 (Negative): 1.00
- Recall for Class 0 (Negative): 0.60

- F1-Score for Class 0 (Negative): 0.75
- Precision for Class 1 (Positive): 0.91
- Recall for Class 1 (Positive): 1.00
- F1-Score for Class 1 (Positive): 0.96
- Overall Model Accuracy: 0.92
- Macro Average Precision: 0.96
- Macro Average Recall: 0.80
- Macro Average F1-Score: 0.85
- Weighted Average Precision: 0.93
- Weighted Average Recall: 0.92
- Weighted Average F1-Score: 0.92

The performance metrics indicate a highly effective model with excellent predictive accuracy for Parkinson's disease detection, based on the voice measurement data. The high precision and recall values, especially for classifying individuals with Parkinson's disease (Class 1), suggest that the model is robust and can be considered reliable for this type of classification task.

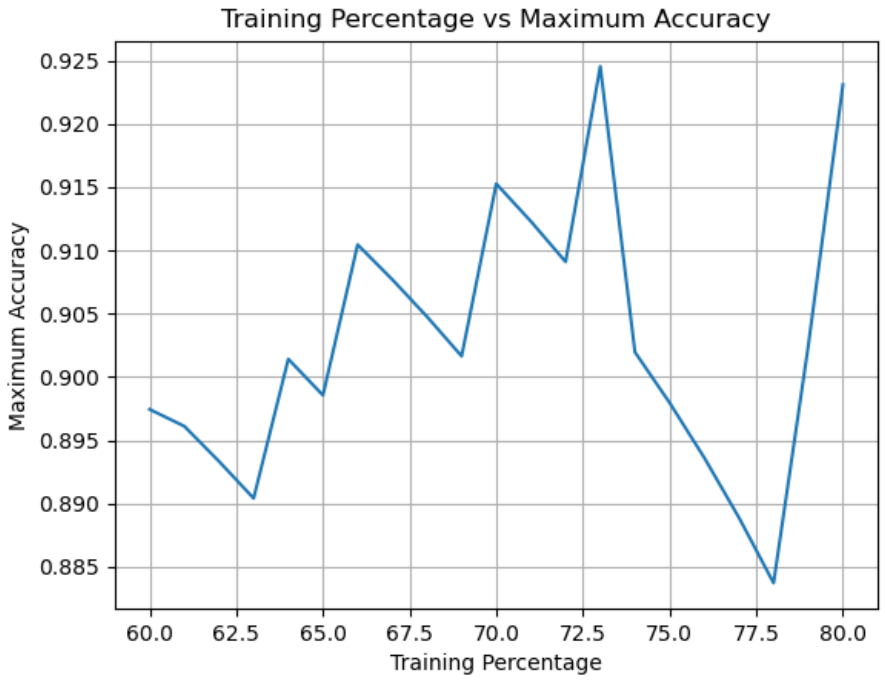


Figure 7: Training Percentage vs. Maximum Accuracy Plot for KNN.

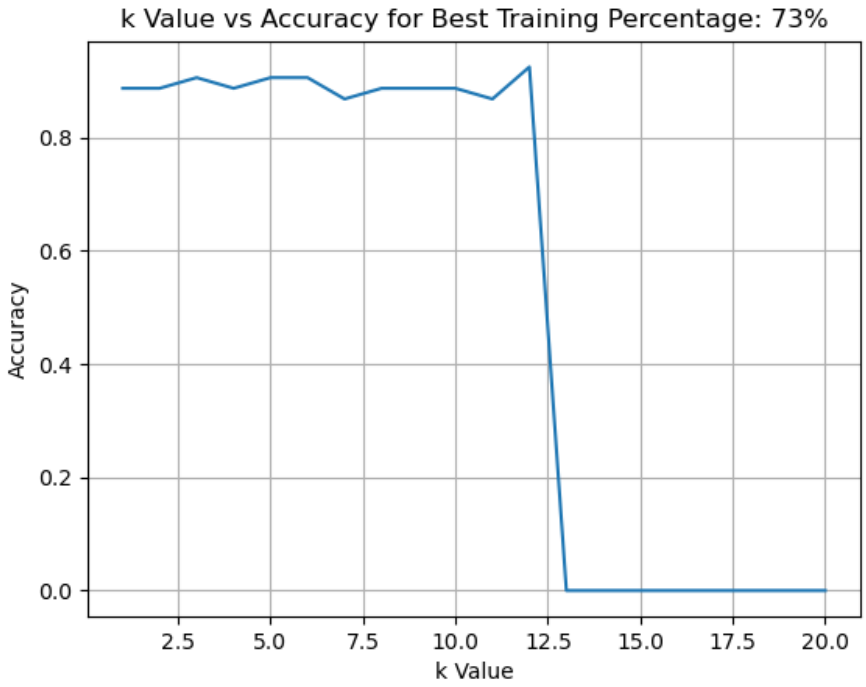


Figure 8: k Value vs. Accuracy Plot for KNN.

```
PS C:\Users\Indu Reddy\OneDrive\Desktop\Graduate Project> python .\optimal_KNN.py
Best training percentage: 73%
Optimal k value: 12
Maximum Accuracy: 0.9245283018867925
Confusion Matrix for Best Model:
[[ 6  4]
 [ 0 43]]
Classification Report for Best Model:
              precision    recall  f1-score   support

           0       1.00      0.60      0.75        10
           1       0.91      1.00      0.96        43

    accuracy                           0.92        53
   macro avg       0.96      0.80      0.85        53
weighted avg       0.93      0.92      0.92        53

PS C:\Users\Indu Reddy\OneDrive\Desktop\Graduate Project>
```

Figure 9: Output of Optimal KNN.

5.5 Comparative Analysis

The performance of the KNN and Naive Bayes models was rigorously evaluated based on a dataset consisting of biomedical voice measurements, with the goal of classifying individuals with and without Parkinson's disease. The analysis centered on the accuracy, precision, recall, and F1-score metrics, providing a multifaceted view of each model's effectiveness.

Naive Bayes Analysis:

- **Performance Metrics:**
  - Accuracy: 87.5% when trained on 84% of the data.
  - Precision: High for positive class (0.92).
  - Recall: Lower for positive class (0.71), indicating potential missed cases.
  - F1-Score: Balanced performance shown by macro-average F1-score (0.82).

K-Nearest Neighbors (KNN) Analysis:

- **Performance Metrics:**
  - Accuracy: Reached up to 92.45% with k=12 when trained on 73% of the dataset.
  - Precision and Recall: Very high for the positive class (0.91 and 1.00, respectively), indicating strong predictability for positive cases.
  - F1-Score: Excellent (0.96 for positive class).

Comparison Summary:

Accuracy and Robustness: KNN outperforms Naive Bayes in terms of accuracy and robustness, particularly in classifying positive cases of Parkinson's disease.

Precision and Recall Trade-offs: Naive Bayes shows high precision but lower recall for positive cases, which might lead to missing true positive cases. KNN, on the other hand, shows excellent precision and recall, suggesting a better balance between identifying true positives and minimizing false negatives.

Model Complexity and Training Data: KNN requires a larger percentage of the dataset for training compared to Naive Bayes, which may affect its scalability and performance on smaller datasets. Naive Bayes is generally faster and less resource-intensive, making it suitable for larger datasets or applications where computational efficiency is a priority.

Practical Use: KNN's requirement for defining the number of neighbors and its sensitivity to the chosen metrics can make it more complex to tune. Naive Bayes is straightforward to implement and can be more intuitive for probabilistic outcomes.

Both algorithms have their merits and are suitable for different aspects of medical diagnostics. KNN's high accuracy and sensitivity make it excellent for applications where the highest possible identification of positive cases is critical, while Naive Bayes offers a good balance of speed and predictability for larger, more complex datasets.

CHAPTER 6: CONCLUSION AND FUTURE WORKS

6.1 Conclusion

In this project, we utilized data mining techniques to develop a predictive model for the detection of Parkinson's disease using vocal biomarkers. By leveraging the capabilities of Python and its libraries, such as Scikit-learn, we implemented two well-known algorithms: K-Nearest Neighbors (KNN) and Naive Bayes.

Effective Classification: The application of KNN and Naive Bayes algorithms proved effective in classifying the presence of Parkinson's disease with considerable accuracy.

Importance of Data Quality: The project underscored the critical importance of quality data collection, preprocessing, and the selection of appropriate features to improve model performance.

Advancing Medical Diagnostics: Our research contributes to the ongoing efforts in the medical field to employ machine learning for early and more accurate diagnosis, which is vital for diseases like Parkinson’s where early intervention can substantially alter patient outcomes.

Algorithmic Foundation: KNN and Naive Bayes were chosen for their simplicity, efficiency, and the interpretability of their results, ensuring a solid foundation for this project.

6.2 Future Works

6.2.1 Algorithm Enhancement:

Incorporate More Algorithms: Test additional machine learning algorithms like Support Vector Machines (SVM), Decision Trees, or Ensemble Methods (e.g., Random Forests) to compare and possibly enhance prediction accuracy.

Advanced Neural Networks: Explore deep learning techniques, such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs), which might capture more complex patterns in voice data associated with Parkinson's disease.

6.2.2 Data Expansion:

Increase Dataset Size: Acquire more voice samples to increase the dataset's diversity and volume, which could help improve the robustness and accuracy of the models.

Multi-Source Data Integration: Combine voice data with other types of medical data (like clinical biometrics or patient genetics) to create a multi-modal prediction model.

6.2.3 AI Predictive Analytics:

Time-Series Analysis: Using AI to predict the future progression of symptoms based on past data.

Risk Assessment Models: Developing AI models that calculate the risk of disease onset based on various biomarkers and patient histories.

6.2.4 Integration with IoT Devices:

Smart Home Assistants: Utilize devices like Amazon Alexa or Google Home to regularly monitor a patient’s vocal patterns and alert for changes that might suggest a need for clinical assessment.

Wearable Technology: Develop AI-powered wearable devices that continuously analyze vocal and motor symptoms to assist in ongoing disease management.

6.2.5 Clinical Testing and Validation:

Clinical Trials: Partner with medical institutions to test the models in clinical settings, helping to validate findings with real-world data.

Regulatory Approval: Work towards getting approval from regulatory bodies to use the system as a diagnostic tool, ensuring it meets all necessary medical and ethical standards.

REFERENCES

- D.Manoj Kumar, R Arthi, Akshay Rajeev, Adithya Ranjith, Ashwin Murali, Arjun K, "Early Detection of Parkinsons Using Machine Learning", 2024 International Conference on Emerging Systems and Intelligent Computing (ESIC), pp.562-565, 2024.
- Santhosh Kumar C, Vishnu Kumar Kaliappan, Rajasekaran Thangaraj, Pandiyan P, "Comparative Study of Classification Algorithms for Early Identification of Parkinson's Disease Based on Baseline Speech Features", Innovations in Information and Communication Technology Series, pp.186, 2020.
- Uddin, S., Haque, I., Lu, H. et al. Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction. Sci Rep 12, 6256 (2022).
- L. Jeancolas et al., "Automatic detection of early stages of Parkinson's disease through acoustic voice analysis with mel-frequency cepstral coefficients," 2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), Fez, Morocco, 2017.
- Das, R. (2010). A comparison of multiple classification methods for diagnosis of parkinson disease. Expert Syst. Appl. 37, 1568–1572