Parkinson's Disease Detection using Machine Learning

Ahaladitha Thamada
BTech, Department of Computer Science and Engineering

***Abstract*: Parkinson's Disease is a special type of neurological condition which has an impact on people across the globe. For efficient treatment of this disease, early detection is crucial. It is a neurodegenerative disorder that affects the motor functions of a human. Research has demonstrated that early signs of the disease include changes in speech such as those in speaking rate and voice quality. This model proposes a brand new approach for the early detection of Parkinson's using the speech alterations. Previously, machine learning has been used to detect the presence of the disease by processing 'spiral' images drawn by the healthy and Parkinson's patients. However, due to minimum accuracy voice and speech alterations have been considered as a reliable parameter for the identification of the disease. The model is trained using various machine learning algorithms from which 80% data is used for training and 20% is used for testing. The model has an average training accuracy of 96.47% and an average testing accuracy of 91.67% making it efficient and accurate when applied to the dataset of speech samples from people with and without the disease.**

***Index Terms:* Parkinson's Disease, early detection, neurological condition, speech alterations, machine learning, voice quality, speech analysis.**

I. INTRODUCTION

Parkinson's is a progressive neurological disorder that affects the central nervous system by targeting the dopamine producing brain cells. Dopamine is a neurotransmitter that is important for controlling and coordinating body movements. Hence, this disorder causes disturbance in movement, causing tremors, stiffness and difficulty with coordination and balance.

Parkinson's is difficult to diagnose and is believed that the disease manifests itself in the body years before the motor related symptoms such as tremors, slowness of hand and leg movement, loss of smelling sense, sleep difficulties and constipation become evident. Moreover, facial abnormalities affect around 90% of PD affected patients. Hence, to reduce the improvement of the disease, many researchers are working towards finding accurate strategies to identify these non-motor symptoms that manifest early in the illness. Recently, machine learning (ML) has become a popular tool for diagnosing diseases in the medical field due to its easy implementation with high accuracy rate.

A. Problem Statement

Being referred to as a long term neurological illness, PD affects more than 1% of the world population annually. Currently, there is no precise diagnostic test for PD which accounts for 25% of misclassification and maltreatment. To overcome this problem, the model proposes to identify the presence of the disease using various voice parameters such as jitter, shimmers, HNR, NHR, RPDE, DFA, PPE, etc.

These parameters help in the accurate identification of the disease making early diagnosis possible and reducing the risk of lifetime medication for the diagnosed.

B. Existing System

Parkinson's disease detection systems leverage both traditional clinical approaches and advanced computational techniques to identify the disease in its early stages. Traditionally, diagnostic methods include physical examinations, medical history assessments, and neuroimaging techniques like MRI, PET scans, or DAT scans. The existing systems analyze non-invasive data such as speech patterns, gait dynamics, or handwriting samples, as PD often presents early symptoms like voice changes, tremors, and motor impairments. ML algorithms such as Support Vector Machines (SVM), Random Forests, and XGBoost process features like pitch, jitter, and shimmer from voice datasets to classify individuals as healthy or affected.

C. Drawbacks in the existing system

Existing diagnostic methods include:

- Neuroimaging (MRI, PET scans): Effective but expensive and inaccessible in resource-limited settings.
- Clinical Evaluations: Subjective and dependent on the clinician's expertise.
- Deep Brain Stimulation: Primarily a treatment option, not diagnostic.

- Data Dependency: ML approaches require substantial high-quality data for effective training and variances in algorithm outcomes can occur.

D. Proposed Model

The proposed system aims to enhance Parkinson's disease detection through a comprehensive approach. It begins with data collection, compiling a diverse dataset that includes clinical data and various biomarkers relevant to the disease. Feature extraction follows, focusing on identifying key attributes from the dataset that distinguish healthy individuals from those with Parkinson's. Using these features, machine learning algorithms are applied to develop predictive models tailored to detect the disease. The system undergoes rigorous validation and testing to ensure the models perform effectively on unseen data. Once validated, the models are integrated into clinical environments, enabling real-time monitoring and early detection of Parkinson's disease. Continuous improvement is prioritized by updating the models with new data, ensuring ongoing refinement and enhanced accuracy over time.

E. Requirements Specification

a. Hardware Specifications:

- Processor (Dual-core 2 GHz or higher)
- Memory (8 GB or higher)
- Storage (20 GB or higher)
- OS

b. Software Specifications:

- Python 3.7 or higher programming language
- Required Libraries (numpy, pandas, scikit-learn, matplotlib, seaborn, pickle)
- Environment Setup (package manager, IDE)
- Dataset

II. DESIGN METHODOLOGY

A. Model Design

Fig1: Model design of the Parkinson's Disease Detection System

As shown in Fig1, the design involves loading and preprocessing the dataset by splitting it into features and target labels, followed by an 80-20 train-test split. Features are standardized to ensure uniform scaling, benefiting algorithms sensitive to feature magnitudes.

Multiple machine learning models, including Random Forest, Gradient Boosting, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN), are trained and optimized for hyperparameter tuning with 5-fold cross-validation. Model performance is evaluated based on training and testing accuracy to select the best classifier. After selecting the best model, the data is evaluated and predicted as 'Normal' or 'Parkinson's'.

B. Model Methodology

1.Data Pre-processing:

The script reads a Parkinson's dataset CSV, performs EDA to assess column types, missing values, and statistics, sets a target variable, and splits features into training/testing sets.

2. Standardization:

The features are standardized using to ensure uniform scaling, improving model performance for algorithms sensitive to feature magnitude.

3. Model Training and Evaluation:

Multiple classifiers (Random Forest, Gradient Boosting, Support Vector Machine (SVM), K-Nearest Neighbors) are trained on the dataset.

Each model undergoes hyperparameter optimization with 5-fold cross-validation to find the best configuration.

Training and testing accuracy for each model is recorded.

4. Prediction:

A sample input is provided, standardized, and passed through the best-performing model for a Parkinson's Disease prediction.

5. Model Persistence:

The final model is saved for reuse, and it can be reloaded later for prediction.

III. IMPLEMENTATION

A. Training

The training phase involves converting the different speech inputs into mel-frequency cepstrum coefficients (MFCC), which comprise the mel-frequency cepstrum. The frequency bands are divided evenly based on the mel band. This response is very similar to that of the human auditory system. Four distinct algorithms are used in the testing phase to identify speech recognition; no pre-processing techniques are needed. The signal is subsequently transformed into MFCC, sent to an SVM for feature extraction, and then sent to various machine learning models, including KNearest Neighbor, SVC, Bagging Classifier, and Logistic Regression, to ascertain whether Parkinson's disease is a possibility.

B. Testing

The testing of data involves evaluating the performance of machine learning models on unseen data (test set) to measure how well the models generalize. This starts with splitting of the dataset into training and testing sets where 20% of the data is reserved for testing.

The data is the standardized or rescaled so that it has a mean of 0 and standard deviation of 1. Each model is trained on the training set with the best hyperparameters. After training, the model's performance is tested on the testing set where the prediction is evaluated and accuracy is calculated.

C. Algorithm Implementation

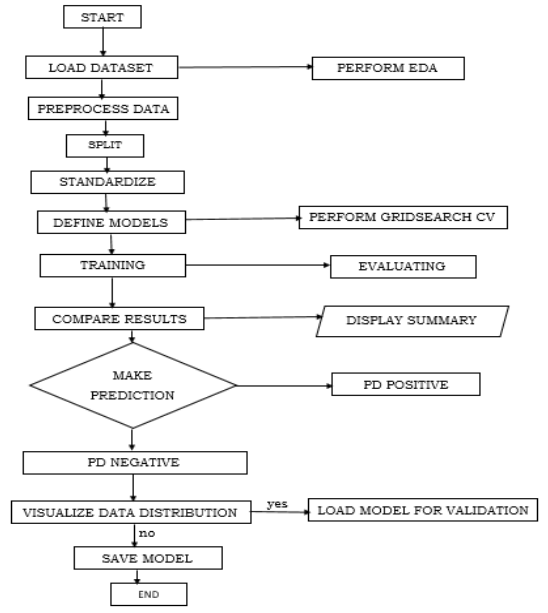

Fig2: Implementation of the algorithm

a. Support Vector Machine

For classification-type tasks, a support vector classifier—a supervised machine learning-based method—is employed. Finding the hyperplane—a line or plane in a high-dimensional space—that best separates the different training data classes is the aim of this SVM model. By choosing one hyperplane that maximizes the distance between the different classes, the margin—the distance between two of the hyperplanes and the closest points of data from each class—was maximized. Because of this, the SVM model is less likely than other classification algorithms to overfit. Apart from identifying the hyperplane, SVC also uses a kernel-style function to convert the input data into a higher-dimensional space, which could make class separation relatively simple.

There are various kinds of kernel machine functions, including radial basis functions (RBF), polynomial type, and linear type. SVM can be used to categorize the provided data into new data points according to which side of the hyperplane they fall on after the hyperplane and kernel function have been established. Referenced from [1][3].

Library used: SVC from sklearn.svm

b. K Nearest Neighbours

For classification and regression problems, a supervised artificial intelligence-based machine learning technique known as KNN can be applied. Since it is a non-parametric algorithm, it does not assume anything regarding the distribution of the underlying data. The KNN algorithm's basic idea is to identify the "K" closest observations (i.e., neighbors) in the training set, then use the majority votes (for classification) or averages (for regression) of their corresponding target values to classify or predict the target variable of a new observation. Stated otherwise, the algorithm identifies the K points that are closest to the new observation and labels them with the majority class of those K points. Referenced from [4].

Library used: KNeighborsClassifier from sklearn.neighbors

c. Gradient Boosting

This is an ensemble learning approach that builds models sequentially, where each subsequent model corrects the errors of its predecessor.

This algorithm builds trees sequentially, where each tree corrects the errors of the previous trees. Referenced from [1][2][3][4].

Library used: GradientBoostingClassifier from sklearn.ensemble

d. Random Forest

This is another ensemble learning technique that builds multiple decision trees and aggregates their outputs via majority voting or averaging. Each tree is trained on a random subset of the data and features to reduce overfitting. Referenced from [1][2][3][4].

Library used: RandomForestClassifier from sklearn.ensemble

e. GridSearch CV

GridSearchCV automates the process of hyperparameter tuning by exhaustively searching through a grid of specified hyperparameters for a model.

For each model, a dictionary of possible hyperparameters is defined using 'param_grid'. The grid search trains the model with all possible combinations of hyperparameters, evaluates them using cross-validation and selects the best set of parameters based on the evaluation metric which is the accuracy of the model.

After completing the search, the best model and its parameters can be accessed. Thus, it ensures the optimal parameters for each model, improving their predictive performance on the given dataset.

IV. RESULTS

A. Importing the dependencies

The various machine learning algorithms are to be imported and this is done on the Google Colab platform.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
```

Fig3: Importing libraries

1. Importing Libraries as shown in Fig3:

- numpy as np:

NumPy is a library for numerical computations, particularly useful for handling arrays and matrices. It provides efficient tools for mathematical operations.

Aliased as np for convenience.

- pandas as pd:

Pandas is a library for data manipulation and analysis. It provides powerful data structures like DataFrame and Series for handling and processing structured data.

Aliased as pd for ease of use.

2. Data Preparation

- from sklearn.model_selection import train_test_split:

This function is used to split datasets into training and testing subsets. It helps evaluate a machine learning model's performance by training it on one subset and testing it on another. Commonly, the dataset is divided into X_train, X_test (features) and y_train, y_test (labels).

3. Model Creation

- from sklearn import svm:

Imports the Support Vector Machines (SVM) module, a supervised learning model used for classification and regression tasks.

SVM finds a hyperplane that best separates data points into different classes.

4. Evaluation Metrics

- from sklearn.metrics import accuracy_score:

This function calculates the accuracy of a model by comparing its predictions with the true labels. Accuracy is the ratio of correctly predicted samples to the total number of samples.

B. Data Collection and Analysis

```
# loading the data from csv file to a Pandas DataFrame
parkinsons_data = pd.read_csv('/content/drive/MyDrive/parkinsons.csv')
```

Fig4: Loading the dataset

Fig4 is a snippet used to load the CSV file which contains the data required for training the model.

Loads the data from the specified CSV file into a Pandas DataFrame, which is a tabular data structure resembling a table (with rows and columns).

Once loaded, this data will allow easy manipulation and analysis of the dataset.

Table 1: The voice measurements used in the experiment

Feature no	Voice measure	MEANING
1	MDVP:Fo (Hz)	Average vocal fundamental frequency
2	MDVP:Fhi (Hz)	Maximum vocal fundamental frequency
3	MDVP:Flo (Hz)	Minimum vocal fundamental frequency
4	MDVP:Jitter (%)	Several measures of variation in
5	MDVP:Jitter (Abs)	fundamental frequency
6	MDVP:RAP	
7	MDVP:PPQ	
8	Jitter:DDP	
9	MDVP:Shimmer	Several measures of variation in amplitude
10	MDVP:Shimmer (dB)	
11	Shimmer:APQ3	
12	Shimmer:APQ5	
13	MDVP:APQ	
14	Shimmer:DDA	
15	NHR	Two measures of ratio of noise to tonal
16	HNR	components in the voice
17	RPDE	Two nonlinear dynamical complexity
18	D2	measures
19	DFA	Signal fractal scaling exponent
20	spread1	Three nonlinear measures of fundamental
21	spread2	frequency variation
22	PPE	
23	status	Health status of the subject: (1) Parkinson's, (0) healthy

```
# printing the first 5 rows of the dataframe
parkinsons_data.head()
```

```
# number of rows and columns in the dataframe
parkinsons_data.shape
```

Fig5: Obtaining the structure of the data

```
# getting more information about the dataset
parkinsons_data.info()
```

Fig6: Obtaining more information about the data

```
# checking for missing values in each column
parkinsons_data.isnull().sum()

# getting some statistical measures about the data
parkinsons_data.describe()

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

Fig7: Structuring and analysing the data

As shown in Fig5, Fig6, Fig7 these are used to print the values available in the dataset, for obtaining the first five rows, for checking of null values, for obtaining the statistical measures, obtaining the total number of rows and columns in the dataset and for the distribution of the target variable.

These snippets are exploratory data analysis (EDA) tools that help understand the structure, quality, and distribution of the dataset. They ensure the data is ready for further preprocessing or modeling.

The results of the above snippets in Fig8, Fig9 would be as follows:

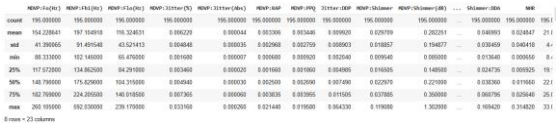

Fig8: The data available in the dataset

Fig9: The data containing values with respect to the disease

C. Model Training and Testing

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

Fig10: Splitting the data

This line of code shown in Fig10 splits a dataset into training and testing subsets, which is an essential step in building and evaluating a machine learning model. For obtaining the accuracy, the Python script outlines a complete machine learning workflow, focusing on comparing multiple classification models for a dataset. The process begins by importing necessary libraries such as pandas for data manipulation and scikit-learn tools for splitting data, scaling features, building models, and tuning hyperparameters. The dataset is loaded and pre processed by separating features (independent variables) from the target variable (status) while excluding irrelevant columns like name. The data is split into training and testing subsets using an 80-20 ratio, ensuring reproducibility with a fixed random state. Feature standardization is performed using StandardScaler to scale all features to a consistent range, optimizing model performance.

The script evaluates four machine learning models: Random Forest, Gradient Boosting, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN). For each model, hyperparameter tuning is conducted using GridSearchCV with 5-fold cross-validation to find the optimal parameters. The best model from the grid search is then evaluated on the training and testing datasets, and the accuracy scores are calculated. The results are stored in a dictionary for comparison.

Finally, the script calculates and prints the training and testing accuracy for each model, along with their averages as shown in Fig11. This enables easy comparison of model performance and helps identify the best classifier for the dataset.

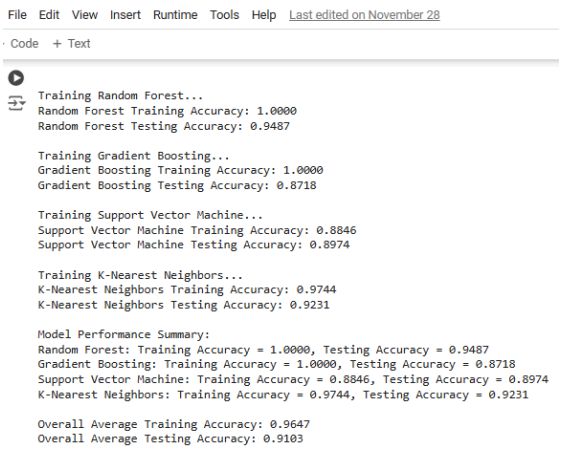

Fig11: The accuracy score of the model

D. Building a Predictive System

A trained machine learning model is used to make a prediction for a new input data point. It starts by defining the input features as a tuple, which represents a single sample with multiple attributes relevant to the Parkinson's disease diagnosis. The input data is converted into a NumPy array for compatibility with machine learning workflows. To ensure the data is in the correct shape for prediction, the array is reshaped into a 2D structure, where the sample forms one row.

The input data is then standardized using the same object that was fitted on the training data earlier, ensuring consistency in feature scaling. This is crucial because the model was trained on scaled data, and any deviation in scaling could affect prediction accuracy. The standardized data is passed to the best_model (obtained from earlier training and tuning) for prediction.

Finally, the model outputs a prediction indicating whether the person has Parkinson's disease (1) or not (0) as shown in Fig12. The result is interpreted and

printed in a human-readable format, providing a clear diagnosis based on the model's prediction. This code highlights the deployment phase of a machine learning pipeline, where the trained model is used to generate actionable insights from new data.

```
[1]
The Person has Parkinson's Disease
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:493: UserWarning: X does not have valid feature
  warnings.warn(
```

Fig12: The output displaying if the patient has the disease or not

E. Visualisation

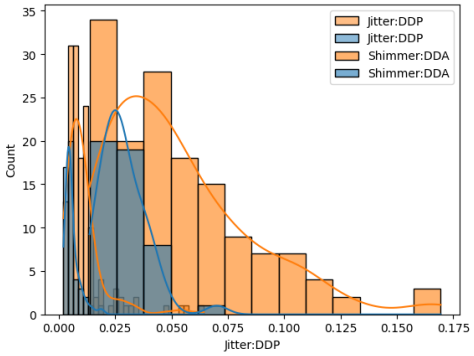

Fig13: Graph visualising the voice parameters between normal and PD patients

As shown in Fig13 the graph visualizes and compares the distributions of two variables, Jitter:DDP and Shimmer:DDA, using histograms and overlaid kernel density estimation (KDE) curves. The histograms display the frequency of data points within specific ranges, with Jitter:DDP represented in orange and Shimmer:DDA in blue. Each bar in the histogram indicates how often data points fall into a particular range, providing a basic view of the data's spread.

The KDE curves, which are the smooth lines overlaid on the histograms, estimate the probability density function for each variable. These curves offer a clearer visualization of the overall shape and spread of the distributions, making it easier to identify trends or differences. For Jitter:DDP, most values are concentrated at lower ranges, showing a positively skewed distribution, while Shimmer:DDA has a similar concentration but potentially a slightly different spread.

This comparison is used in a Parkinson's disease analysis to observe patterns or differences between these voice-related features. Overlapping areas suggest similarities between the variables, while differences in the KDE curves may highlight distinct characteristics.

V. CONCLUSION AND FUTURE SCOPE

A. Conclusion

The study's conclusion demonstrates that a number of variables, including speech and voice traits, can be used to accurately diagnose Parkinson's disease. Better patient outcomes and earlier treatment are two potential benefits of early Parkinson's disease diagnosis and identification. Two further aspects of Parkinson's disease management that could profit from the use of machine learning techniques are tracking the progression of the illness and the efficacy of treatment. However, more research is needed to validate these findings on larger and more diverse datasets. Standardizing data gathering methods and protecting data privacy are two challenges that may arise when using machine learning algorithms in clinical practice.

However, machine learning can be very helpful in both identifying and managing Parkinson's disease, and there is hope that this work will stimulate further research in this area.

B. Future Scope

Future developments in machine learning (ML) for Parkinson's disease detection could focus on several key areas:

- Enhanced Data Diversity: Expanding datasets to include diverse populations and varied speech patterns will improve model generalization and accuracy.
- Integration of Multimodal Data: Incorporating additional data types, such as gait analysis or neuroimaging, alongside speech features can enhance diagnostic precision.
- Real-Time Monitoring: Developing real-time monitoring systems using mobile applications or wearable devices could facilitate continuous assessment and early intervention.
- Explainable AI: Focusing on explainable AI will help clinicians understand model decisions, improving trust and integration into clinical workflows.

REFERENCES

[1] M. Nalini, M. J. Kinol, B. R. M and N. Vijayaraj, "Parkinson's Disease Detection by Machine Learning," *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*, Chennai, India, 2023, pp. 1-5, doi: 10.1109/ICCEBS58601.2023.10449173.

[2] Aditi Govindu, Sushila Palwe, "Early Detection of Parkinson's Disease using Machine Learning". In: Mallick, P.K., Bhoi, A.K., Chae, GS., Kalita, K. (eds) Advances in Electronics, Communication and Computing. ETAEERE 2023. Lecture Notes in Electrical Engineering, vol 709. Springer, Singapore.

[3] Pahuja, G., & Nagabhushan, T. N. (2021). A Comparative Study of Existing Machine Learning Approaches for Parkinson's Disease Detection. *IETE Journal of Research*, *67*(1), 4–14.)

[4] C.K. Gomathy, B. Dheeraj Kumar Reddy, B. Varsha, B. Varshini (2020). The Parkinson's disease detection using Machine Learning techniques. International Research Journal of Engineering and Technology (IRJET).

[5] B. M. Eskofier *et al.*, "Recent machine learning advancements in sensor-based mobility analysis: Deep learning for Parkinson's disease assessment," *2019 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Orlando, FL, USA, 2019, pp. 655-658, doi: 10.1109/EMBC.2016.7590787.

[6] Aich, Satyabrata, et al. "A supervised machine learning approach using different feature selection techniques on voice datasets for prediction of Parkinson's disease." *2019 21st International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2019.

[7] Banita. "Detection of Parkinson's Disease Using Rating Scale." *2019 International Conference on Computational Performance Evaluation (ComPE)*. IEEE, 2019.

[8] Bhalchandra, Noopur A., et al. "Early detection of Parkinson's disease through shape based features from 123 I-Ioflupane SPECT imaging." *2018 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2018.

[9] Gil-Martín, Manuel, Juan Manuel Montero, and Rubén San-Segundo. "Parkinson's disease detection from drawing movements using convolutional neural networks." *Electronics* 8.8 (2016): 907.

[10] Ortiz A, Munilla J, Martínez-Ibañez M, Górriz JM, Ramírez J, Salas-Gonzalez D. Parkinson's Disease Detection Using Isosurfaces-Based Features and Convolutional Neural Networks. Front Neuroinform. 2015 Jul 2;13:48. doi: 10.3389/fninf.2015.00048. PMID: 31312131; PMCID: PMC6614282.

[11] Becker, Georg, et al. "Early diagnosis of Parkinson's disease." *Journal of neurology* 249.Suppl 3 (2002): iii40-iii48.