Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Report
on

"Selection of best classification model to detect Parkinson's disease using Machine Learning"

[Code No: COMP 206]

(For partial fulfillment of II Year/Semester I in Computer Science)

Submitted by

Bigyan Kumar Piya (41)
Sujan Ghimire (15)
Aagaman Bhandari (07)
Sewan Uprety (60)

Submitted to

Mrs. Praynita Karki

Computer Science Coordinator

Department of Computer Science and Engineering

May 15, 2023

Bona fide Certificate

This project work on

“Selection of best classification model to detect Parkinson’s disease using Machine Learning”

is the bona fide work of

“BIGYAN KUMAR PIYA”

“SUJAN GHIMIRE”

“SEWAN UPRETY”

“AAGAMAN BHANDARI”

who carried out the project work under my supervision.

Project Supervisor

____

Mr. Saugat Singh

Teaching Assistant

Department of Computer Science and Engineering

Date:

Abstract

The research-based approach project where we created a program that gave us the basic initiation on which model would be the best choice for uttermost accuracy. Our prime focus and intention was to learn the basic machine learning and its transverse the field as a team. We took data from UCI machine learning repository and used seven different classification models, measured its accuracy parameters to select best model for prediction of the disease.

Keywords: *Parkinson's disease, machine learning, classification*

Table of Contents

List of Figures

List of Tables

Acronyms/Abbreviations

The list of all abbreviations used in the documentation is included in this section. See the example below:

ML	Machine Learning
PD	Parkinson’s disease
ICPR	International Conference on Pattern Recognition
MDVP	Multidimensional voice program
ROC	Receiver Operating Characteristic Curve
AUC	Area Under the Curve
Hz	Hertz
GHz	GigaHertz
IPCR	Input Port Change Register
dB	Decibel
ms	Millisecond
RAM	Random Access Memory
SVC	Support Vector Classifier
PCA	Principal Component Analysis
CSV	Comma Separated Value
EDA	Exploratory Data Analysis
DNN	Deep Neural Network

Chapter 1 Introduction

1.1 Background

Parkinson’s disease is a neurodegenerative disorder that results in unintended or inevitable movements in our body. It is estimated to affect numerous people across the globe and has negatively affected the quality of life of those who are affected. If not treated on time and not given the effective treatment, it will start to hamper the health on worst level. So, a proper treatment in proper time, is a must for Parkinson’s disease. For detection, it is hard to comprehend for a human brain as there are dozens of symptom variables but a computer program can diligently detect the disease with appreciable accuracy and reliability. This is where machine learning comes in hand. A program model can be taught to detect the disease with the help of large dataset and numerous training sets.

Multiple studies in recent years have given a clearance on how machine learning algorithms can be used in the prediction of Parkinson’s disease. An example can be taken from one of the earliest studies performed regarding this field, which was published in 2009. It used decision tree algorithm for the prediction of Parkinson’s disease. This study approached the dataset of patients with Parkinson’s disease and healthy controls, and concluded that this method was successful to obtain the prediction with high accuracy. In more recent years, several studies have proposed the use of deep learning algorithms, such as artificial neural networks and convolutional neural networks, for the prediction of Parkinson’s disease. In this project, we are using classification model to detect Parkinson’s disease with best efficiency.

The speech recording experiments were carried out by Little et al., with all 31 subjects providing their written informed consent reviewed and approved by University of Oxford, United Kingdom, and United States National Center for Voice and Speech, Denver, Colorado. Each subject was requested to pronounce vocal vowels with a head-mounted microphone positioned at 8 cm in front of the lips. The head-mounted microphone was calibrated by using a Class 1 sound level meter. The acquired acoustic signals were sampled at 44.1 kHz with 16-bit resolution per sample. The amplitude of each signal was digitally normalized in order to suppress the effects of individual difference. The Kay Pentax MDVP was used by Little et al. to measure 16 voice perturbation parameters, including the period (jitter) and amplitude (shimmer) perturbations, and harmonics-to-noise (and noise-to-harmonics) ratios. Six additional nonlinear parameters were also computed by Little et al. to characterize the signal complexity degree and fractal dimensions of the dysphonic voice records.

1.2 Objectives

Objectives of the Parkinson's disease predictor program model are:

- To evaluate and validate the best machine learning algorithm that can predict the likelihood of an individual developing Parkinson's disease.
- To evaluate the accuracy and reliability of the predictive model through testing and validation on a diverse dataset.
- To explore the potential of machine learning in health sector and its reliability.

1.3 Motivation and Significance

Our main motto to do this project was to explore the possibilities of artificial intelligence and machine learning in health domain. We were clearly aware that simply through our program we can't fully diagnose the disease and cure it. But still, we were hopeful that if we could pull off the best model selection for peak accuracy, it would, to some extent, help a lot for the detection of the Parkinson disease and we could, further, do the research on the same. And to make our basic idea into reality, we decided to work on our project using Machine learning classification algorithms. This might be the basic thing in context of Parkinson disease as the disease itself is a huge thing, but our intention was clear to focus on the research-based approach of machine learning and to explore more as a group.

Chapter 2 Related Works

2.1 Prediction of Parkinson’s disease using speech signal with Extreme Learning Machine, (2016)

The authors of this paper are Agarwal A, Chandrayan S and Sahu S. This research aims to predict Parkinson’s by using extreme machine learning techniques called as Extreme Learning Machine. With this method, PD can be accurately predicted using various speech samples. This method uses reliable dataset from UCI repository and distinguishes Parkinson’s disease with healthy samples with an accuracy of 90.76% for the training dataset. When tested with an independent dataset, the method gave an accuracy of 81.55%. Extreme learning machine are feedforwarding neural network for classification, regression, clustering, sparse approximation, compression and feature learning with a single layer or multiple layers of hidden nodes, where the parameters of hidden nodes need to be tuned. The major advantage of ELM is that it can significantly reduce the amount of time needed to train a single-hidden nodes due to the random determination of the input weights and hidden biases than the conventional tuning based learning algorithms.

2.2 A deep learning approach for prediction of Parkinson’s disease progression, (2020)

The authors of this paper are Afzal Hussain Shahid and Maheshwari Prasad Singh. This paper purposed a deep neural network (DNN) model using the reduced input feature space of Parkinson’s tele monitoring dataset to predict PD progression. In this research, Principal Component Analysis (PCA) was used with a deep learning paradigm for the assessment of PD progression. PCA addresses a multicollinearity problem in the data and reduces the dimensionality of the feature space that improves the model’s performance. PD is assessed by using the unified Parkinson’s disease rating scale (UPDRS). In this research, the model's performance was evaluated by conducting various experiments and the result was compared with the result of previously developed methods on the same dataset. The model's prediction accuracy was measured by fitness parameters, mean absolute error (MAE) root mean squared error (RMSE), and coefficient of determination (R^2).

2.3 Early Prediction of Parkinson's disease using machine learning and deep learning approaches, (2020)

The authors of this work are: Tiwari H, Shridhar S, Patil P, Sichana K R, Aishwarya G. In this paper, they used various prediction models for the prediction of PD using a reliable dataset from UCI machine learning repository where 195 samples of features from 31 people, 23 with PD and 8 of them were the control group. This research consulted with the application of six classification algorithms such as Logistic Regression, Support Vector Machine (SVM), Decision tree, K- Nearest Neighbor (KNN), and XGBOOST (Extreme Gradient boosting) which are used to predict the outcome whether the person is healthy or diagnosed with PD. According to their report, Accuracy achieved by the seven classifiers lied within the range of 70-100%. It performed with impressive accuracy for the training sets. In this research, some "Ensemble" learning techniques were also used. With this technique, we try to improve the accuracy by combining various models. According to their report, this machine learning model can significantly improve diagnosis method of PD. In this study, it was indicated that the ensemble techniques XGBOOST classification (Extreme gradient boosting) algorithm achieved the high-test accuracy (95%) compared to other classification model.

Chapter 3 Design and Implementation

After setting up the jupyter notebook, we imported required libraries . After importing the libraries, we accessed the dataset as CSV file (Comma Separated values).

3.1.1 Data Preprocessing

As the next step, we proceeded to data preprocessing. Data pre-processing is the process of cleaning and organizing the raw data to make it suitable for building and training ML models. We accessed the data from UCI repository.

The following flow chart shows data processing in the program:

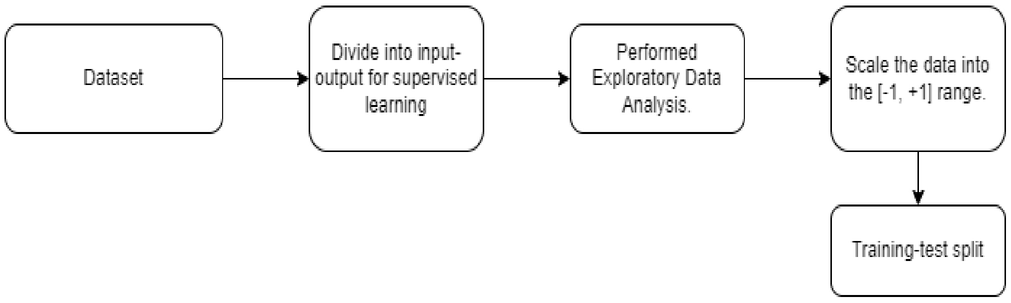
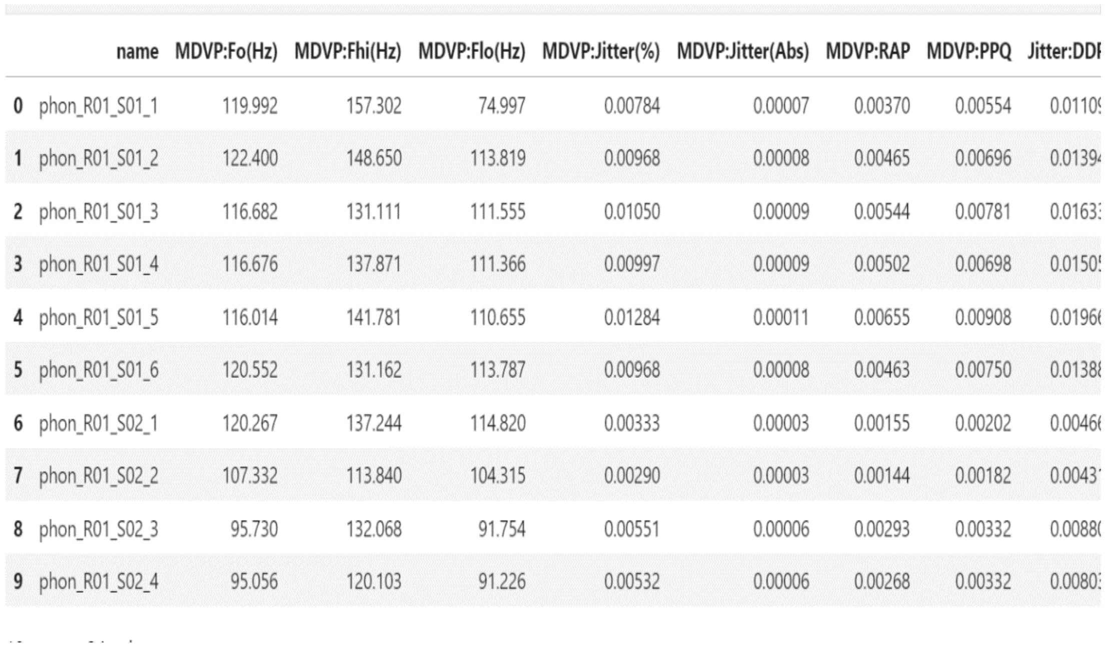

Dataset
Divide into input-output for supervised learning
Performed Exploratory Data Analysis.
Scale the data into the [-1, +1] range.
Training-test split

Figure 3.1.1.1 Flowchart for data preprocessing

3.1.2 Exploratory Data Analysis (EDA)

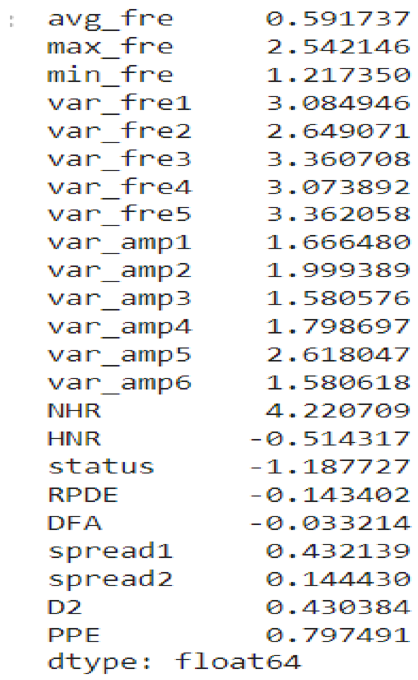
For visualizing the data, we did Exploratory Data Analysis. EDA is the process of acquiring knowledge and insights from the data sets by the help of proper analysis and summarization. Firstly, we started from EDA which helped us gain the proper insight and knowledge about the data through which we were going to work on our project. It focuses on understanding of the pattern, relationships or distributions of the data set and also identifies the irregularities in the sets that might require future study



	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DD
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.0110
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.0139
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.0163
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.0150
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.0196
5	phon_R01_S01_6	120.552	131.162	113.787	0.00968	0.00008	0.00463	0.00750	0.0138
6	phon_R01_S02_1	120.267	137.244	114.820	0.00333	0.00003	0.00155	0.00202	0.0046
7	phon_R01_S02_2	107.332	113.840	104.315	0.00290	0.00003	0.00144	0.00182	0.0043
8	phon_R01_S02_3	95.730	132.068	91.754	0.00551	0.00006	0.00293	0.00332	0.0088
9	phon_R01_S02_4	95.056	120.103	91.226	0.00532	0.00006	0.00268	0.00332	0.0080

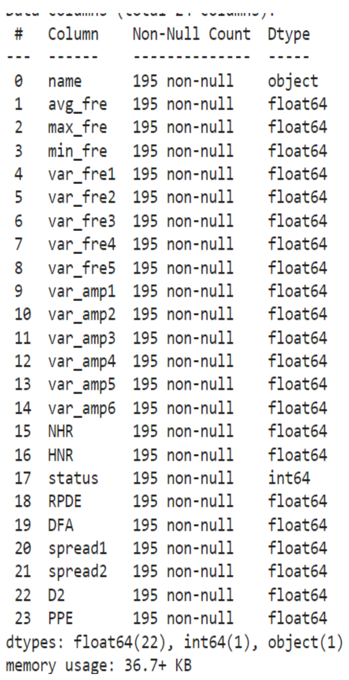
Figure 3.1.2.1 Reading the data

There are 195 rows and 24 columns in our dataset. After that, we dug out the statistical entities of the data like mean, median, mode, correlation and skewness. These statistical entities helped us to get an overview of frequency distribution and biasness of the data and also how much the features are related with one another. We also explored whether the data has any duplicate, null features or not.

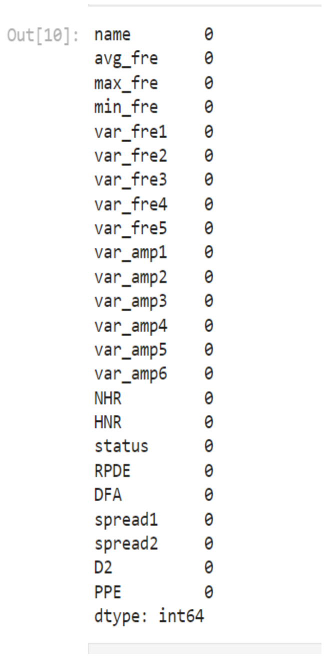


```
:  avg_fre      0.591737
   max_fre      2.542146
   min_fre      1.217350
   var_fre1     3.084946
   var_fre2     2.649071
   var_fre3     3.360708
   var_fre4     3.073892
   var_fre5     3.362058
   var_amp1     1.666480
   var_amp2     1.999389
   var_amp3     1.580576
   var_amp4     1.798697
   var_amp5     2.618047
   var_amp6     1.580618
   NHR          4.220709
   HNR         -0.514317
   status      -1.187727
   RPDE        -0.143402
   DFA         -0.033214
   spread1      0.432139
   spread2      0.144430
   D2           0.430384
   PPE          0.797491
   dtype: float64
```

Figure 3.1.2.2 Calculating skewness



```
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   name      195 non-null    object 
 1   avg_fre   195 non-null    float64
 2   max_fre   195 non-null    float64
 3   min_fre   195 non-null    float64
 4   var_fre1  195 non-null    float64
 5   var_fre2  195 non-null    float64
 6   var_fre3  195 non-null    float64
 7   var_fre4  195 non-null    float64
 8   var_fre5  195 non-null    float64
 9   var_amp1  195 non-null    float64
 10  var_amp2  195 non-null    float64
 11  var_amp3  195 non-null    float64
 12  var_amp4  195 non-null    float64
 13  var_amp5  195 non-null    float64
 14  var_amp6  195 non-null    float64
 15  NHR       195 non-null    float64
 16  HNR       195 non-null    float64
 17  status    195 non-null    int64  
 18  RPDE      195 non-null    float64
 19  DFA       195 non-null    float64
 20  spread1   195 non-null    float64
 21  spread2   195 non-null    float64
 22  D2        195 non-null    float64
 23  PPE       195 non-null    float64
dtypes: float64(22), int64(1), object(1)
memory usage: 36.7+ KB
```



```
Out[10]: name        0
         avg_fre     0
         max_fre     0
         min_fre     0
         var_fre1    0
         var_fre2    0
         var_fre3    0
         var_fre4    0
         var_fre5    0
         var_amp1    0
         var_amp2    0
         var_amp3    0
         var_amp4    0
         var_amp5    0
         var_amp6    0
         NHR         0
         HNR         0
         status      0
         RPDE        0
         DFA         0
         spread1     0
         spread2     0
         D2          0
         PPE         0
         dtype: int64
```

Figure 3.1.2.3 Check for null value

The first table tells the skewness of the dataset. From the above two tables, we came to know that no column is empty in the data set.

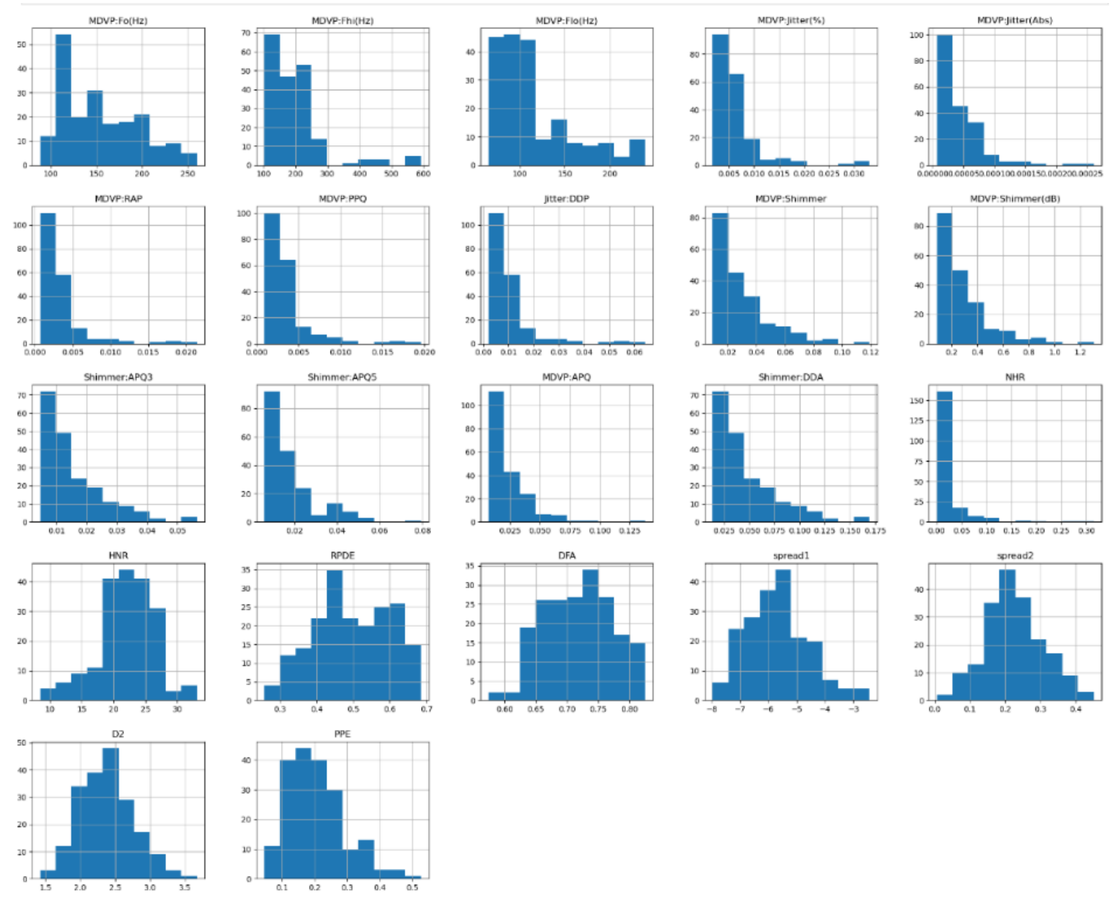


Figure 3.1.2.4 Histogram

- Histogram shows us frequency distribution of the features of the dataset. We found that there were few outliners but we choose not to remove them because it can further increase the biasness.

- Correlation is a statistical entity which tells us how any two variables are linearly related. Heat map gives us intuitive idea showing correlation of the dataset and helps us to understand variables better. It helps in principal component analysis (PCA), data which are highly correlated could be neglected.

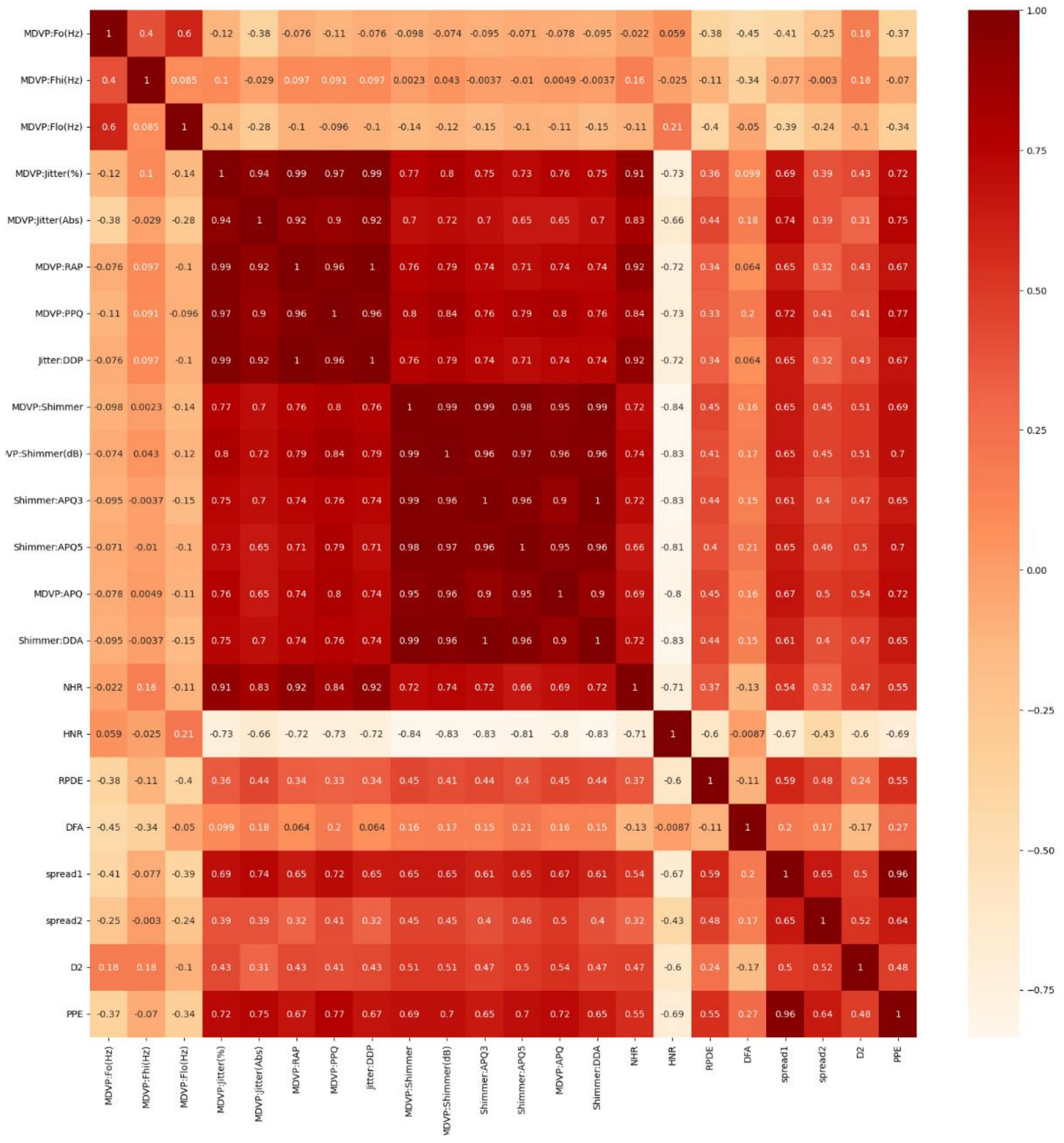


Figure 3.1.2.5 Heatmap

3.1.3 Training and Testing Split

We then split the dataset in the ratio 85:15 of the existing data for training and the remaining for the test with random state=42 which keeps the training set and test set constant.

Dataset
Training Set
Test Set

Figure 3.1.3.1 Train-test split

- Training Data=165
- Test Data=30

3.1.4 Feature Scaling

After this, we moved to feature scaling. Feature scaling is a method used to normalize the range of independent variables or features of data. Feature scaling adjusts the numbers to make it easy to compare the values that are out of each other's scope. It helps in increasing accuracy of the model.

3.1.5 Algorithms

Logistic regression:

Logistic regression is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set. The sigmoid function generates a probability output. By comparing the probability with a pre-defined threshold, the object is assigned to a label accordingly.

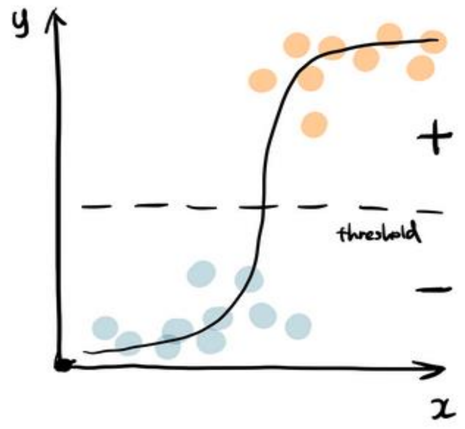


Figure 3.1.5.1 Logistic Regression

Decision Tree:

Decision tree builds tree branches in a hierarchy approach and each branch can be considered as an if-else statement. The branches develop by partitioning the dataset into subsets based on most important features. Final classification happens at the leaves of the decision tree.

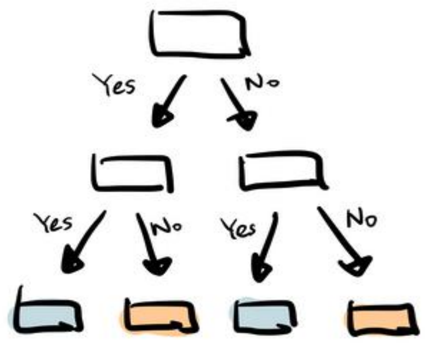


Figure 3.1.5.2 Decision tree

Random Forest:

As the name suggest, random forest is a collection of decision trees. It is a common type of ensemble methods which aggregate results from multiple predictors. Random forest additionally utilizes bagging technique that allows each tree trained on a random sampling of original dataset and takes the majority vote from trees. Compared to decision tree, it has better generalization but less interpretable, because of more layers added to the model.

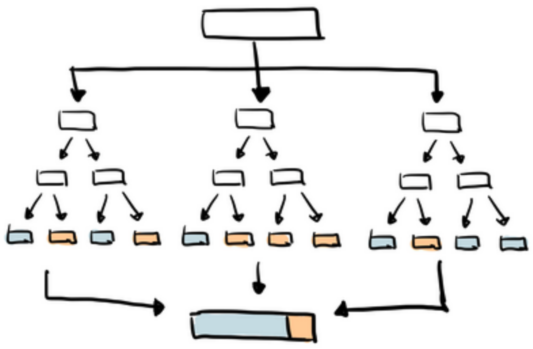


Figure 3.1.5.3 Random Forest

Support Vector Machine (SVM):

Support vector machine finds the best way to classify the data based on the position in relation to a border between positive class and negative class. This border is known as the hyperplane which maximize the distance between data points from different classes. Similar to decision tree and random forest, support vector machine can be used in both classification and regression, SVC (support vector classifier) is for classification problem.

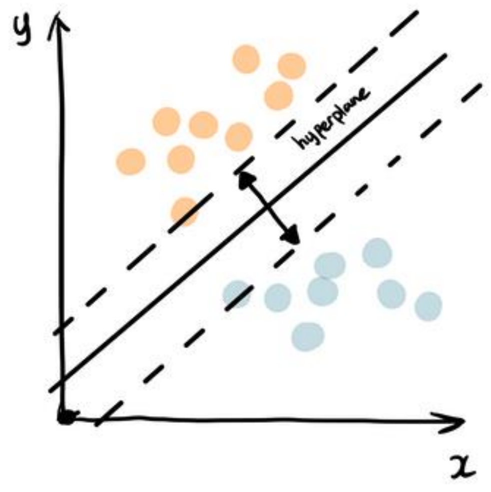


Figure 3.1.5.4 Support Vector Machine

K-Nearest Neighbor (KNN):

K-nearest neighbors (KNN) algorithm uses ‘feature similarity’ to predict the values of new data points which further means that the new data point will be assigned a value based on how closely it matches the points in the training set.

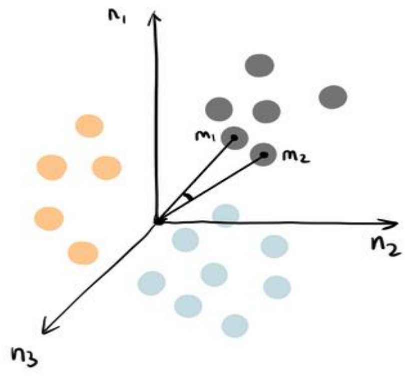


Figure 3.1.5.5 K-Nearest Neighbor

Naive Bayes:

Naive Bayes is based on Bayes’ Theorem — an approach to calculate conditional probability based on prior knowledge, and the naive assumption that each feature is independent to each other. The biggest advantage of Naive Bayes is that, while most machine learning algorithms rely on large amount of training data, it performs relatively well even when the training data size is small.

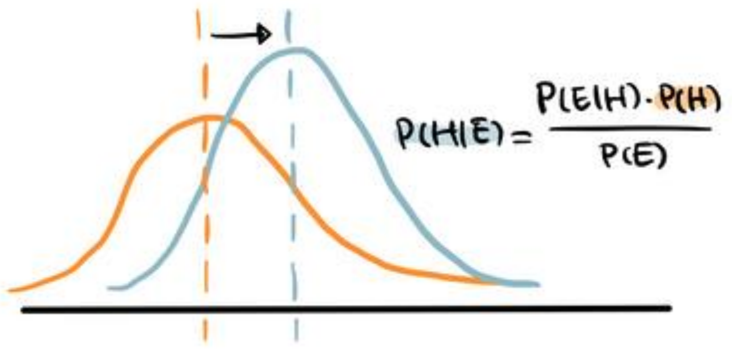


Figure 3.1.5.6 Naive Bayes

XGBoost:

It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems. It's vital to an understanding of XGBoost to first grasp the machine learning concepts and algorithms that XGBoost builds upon: supervised machine learning, decision trees, ensemble learning, and gradient boosting.

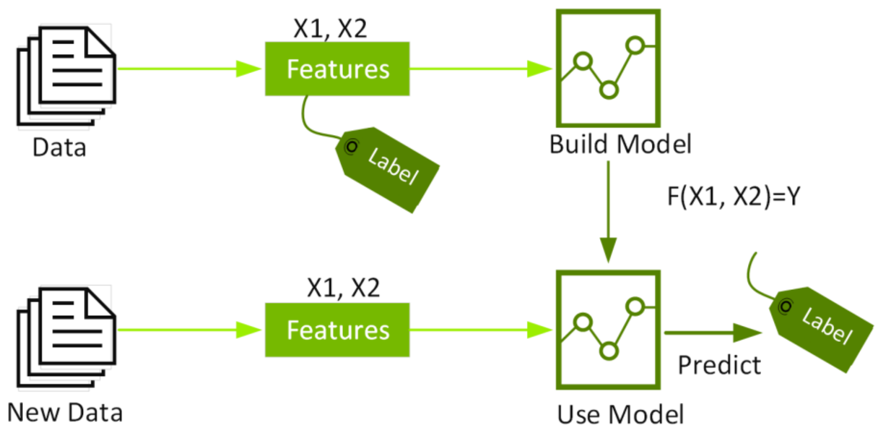


Figure 3.1.5.7 XGBoost algorithm

3.1.6 Prediction of Disease

Various machine learning algorithms like SVC, Naive Bayes, Decision Tree, Random Forest, Logistic Regression, Xg-boost and K-Nearest Neighbor are used for classification. Comparative analysis is performed among algorithms and the algorithm that gives the highest accuracy is used for heart disease prediction.

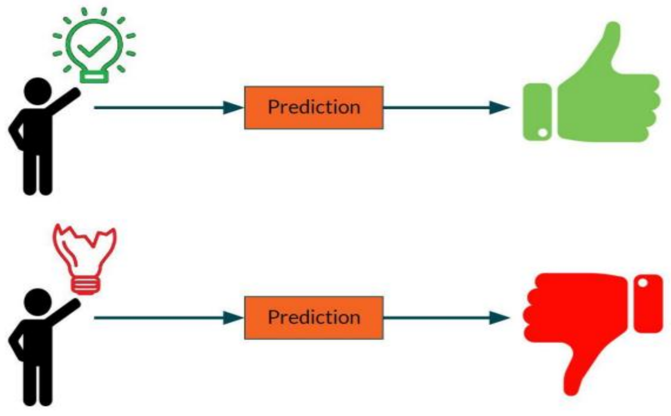


Figure 3.1.6.1 Prediction of Disease

3.1.7 Hyperparameter tuning:

It is the process of selecting the optimal values for the hyperparameters for any machine learning module. It was done to ensure that all possible combinations with different values of hyperparameters would come into implementation and help us select the most appropriate set of hyperparameters. Out of different tuning methods, we used two of them namely grid search and random search.

- Grid Search: Under the grid search, we created a grid of different hyperparameters values and evaluated the performance of modules after training them with each combination of hyperparameters. The hyperparameter combination that yielded the best performance was selected as the final hyperparameters for the model.

- Random Search: This method randomly selects the hyperparameters for any given model. Here we defined a search space as a range of values for each hyperparameters where the hyperparameters are randomly selected. The best set of hyperparameters was selected based on the model's performance on a validation set.

Random forest classifier had its accuracy upsurged from 0.933 to 0.967 whereas the SVC model plummeted from 0.967 to 0.945. Other models had little to no effect while their hyperparameters were tuned. In a nutshell, it gave us a vivid idea on how a model's hyperparameters could be tuned to fit into different scenarios without compromising the accuracy.

3.1.8 Gradio Library:

Gradio is free and open source python library . We can quickly and easily create UI interfaces within our python notebook ,or share with anyone with just few lines of code and demonstrate our finished model results. Gradio helps quickly create customizable UI components within colab , jyupter notebook or scripts and around TensorFlow or PyTorch models, or even arbitrary Python functions.We created an interface to input the features by the user using Gradio library. After all the features are put into the interface, it shows the status whether the data input has PD or not.

3.2 Overall architecture

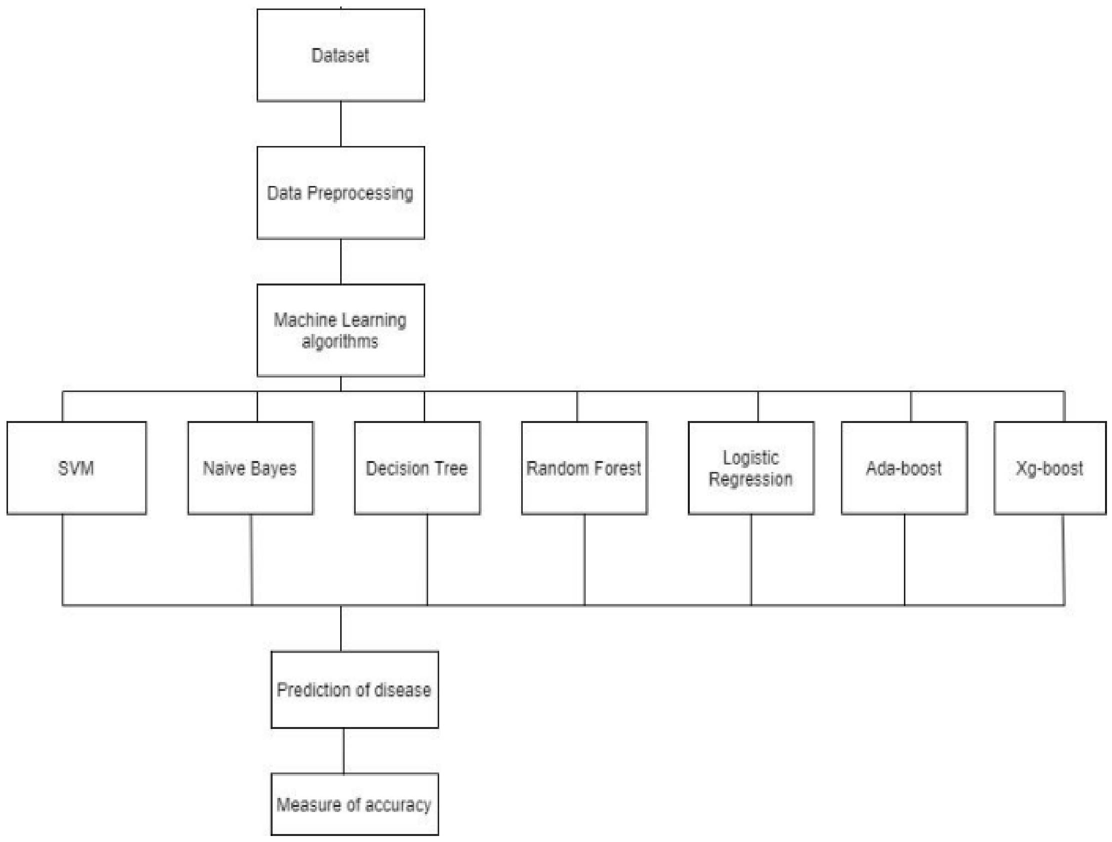


Figure 3.1.8.1 Overall architecture

The flowchart above illustrates the overall architecture of our program. Initially, we imported the dataset from UCI repository. As a part of data preprocessing, our dataset underwent feature scaling and EDA. Then we split our dataset into train-test set (0.85, 0.15) for further processing. Thereafter, we trained our 7 models using the training set and analyzed their performance based on their accuracy, roc, auc curve, precision, recall and f1 score.

3.3 System Requirement Specifications

3.3.1 Software requirements

- Python 3.5 in Jupyter notebook is used for data pre-processing, model training and prediction.
- Operating System: Windows 7 and above or Linux based OS or Mac OS.
- Browser: MS Edge, Google Chrome or similar browser for web basesd results and Jupyter notebook.

3.3.2 Hardware requirements

- Processer : Any Update Processer
- Ram : Min 4GB
- Hard Disk : Min 100GB

3.3.3 Functional Requirements

- The software should accept the Parkinsson_disease.csv dataset as input.
- The software should do verification and processing on input for model training.
- The software uses classification models as main component.
- It evaluates the input data and calculates the accuracy of the classification model.

3.3.4 Non- Functional requirements

- Usability: It defines the UI of the software in terms of simplicity of understanding the user interface of PD prediction software.
- Efficiency: To maintain the possible highest accuracy in the classification of PD in shortest time with available data.
- Performance: It is a quality attribute of the PD classification software that describes the responsiveness to various user interactions.

3.4 Dataset description

Abbreviations	**Feature Description**
MDVP:F0 (Hz)	Average vocal fundamental frequency
MDVP:Fhi (Hz)	Maximum vocal fundamental frequency
MDVP:Flo (Hz)	Minimum vocal fundamental frequency
MDVP:Jitter(%)	MDVP jitter in percentage
MDVP:Jitter(Abs)	MDVP absolute jitter in ms
MDVP:RAP	MDVP relative amplitude perturbation
MDVP:PPQ	MDVP five-point period perturbation quotient
Jitter:DDP	Average absolute difference of differences between jitter cycles
MDVP:Shimmer	MDVP local shimmer
MDVP:Shimmer(dB)	MDVP local shimmer in dB
Shimmer:APQ3	Three-point amplitude perturbation quotient
Shimmer:APQ5	Five-point amplitude perturbation quotient
MDVP:APQ11	MDVP 11-point amplitude perturbation quotient
Shimmer:DDA	Average absolute differences between the amplitudes of consecutive periods
NHR	Noise-to-harmonics ratio
HNR	Harmonics-to-noise ratio

RPDE	Recurrence period density entropy measure
D2	Correlation dimension
DFA	Signal fractal scaling exponent of detrended fluctuation analysis
Spread1	Two nonlinear measures of fundamental
Spread2	Frequency variation
PPE	Pitch period entropy

Table 3.4.1 Dataset description

Chapter 4 Discussion on the achievements

4.1 Performance Analysis

In this project, various machine learning algorithms like SVM, Naive Bayes, Decision Tree, Random Forest, Logistic Regression, XG-boost are used to predict parkinson disease. Parkison Disease UCI dataset, has a total of 23 attributes. The accuracy for individual algorithms has to measure and whichever algorithm is giving the best accuracy,that is considered for the heart disease prediction.

For evaluating the experiment, various evaluation metrics like accuracy, confusion matrix, precision, recall, and f1-score are considered.

4.1.1 Accuracy

Accuracy is the ratio of the number of correct predictions to the total number of inputs in the dataset. It is expressed as:

Accuracy = (TP + TN) /(TP+FP+FN+TN)

4.1.2 ROC and AUC:

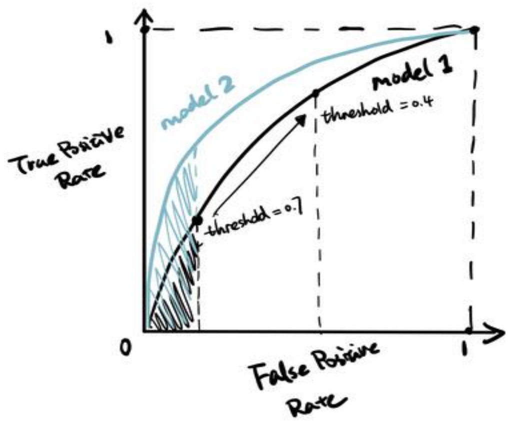


Figure 4.1.2.1 ROC and AUC curve

ROC is the plot of **true positive rate against false positive rate** at various classification threshold. AUC is the area under the ROC curve, and higher AUC indicates better model performance

4.1.3 Precision

It is the ratio of correct positive results to the total number of positive results predicted by the system. It is expressed as:

4.1.4 Confusion Matrix

It gives us a matrix as output and gives the total performance of the system.

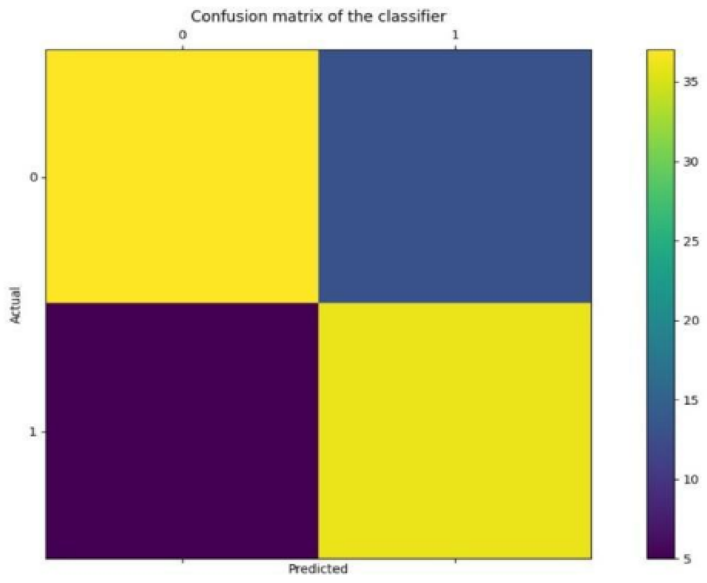


Figure 4.1.4.1 Confusion matrix

Where, TP: True positive

FP: False Positive

FN: False Negative

TN: True Negative 59

4.1.5 Recall

It is the ratio of correct positive results to the total number of positive results predicted by the system. It is expressed as:

F1 Score-It is the harmonic mean of Precision and Recall. It measures the test accuracy. The range of this metric is 0 to 1.

4.2 Achievement

Our main objective was to find the optimal model for our dataset. On course of doing so, we trained seven different models and evaluated accuracy obtained from these respective models. A comparative study of these models is given below:

S.N	Model Name	Accuracy	Auc	f1 Score	Precision	Recall
1.	Logistic Regression	0.90	0.90	0.941	0.96	0.923
2.	XGBoost	0.933	0.95	0.961	0.961	0.961
3.	Random Forest	0.933	0.93	0.961	0.961	0.961
4.	Support Vector Classifier	1.0	1.0	1.0	1.0	1.0
5.	K-Nearest Neighbor	0.9667	0.99	0.981	0.962	1.0
6.	Naïve Byes	0.767	0.75	0.851	0.952	0.769
7.	Decision tree	0.9	0.88	0.941	0.96	0.923

Table 4.2.1 Accuracy and AUC of classification models

Here we are dealing with medical field, so our main objective would be to implement the model with least false negative. Since we cannot diagnose a person with

Parkinson as a healthy individual, it remains our topmost priority to either have the least false negatives or zero false negatives. In order to facilitate this process, we evaluated the accuracy, roc, auc curve, confusion matrix, precision, recall and f1 score for all these seven models.

Here models XG boost, Random forest, SVC and KNN gave us the optimal accuracy given by the performance analysis parameter. Naïve byes had the least accuracy. The SVM classifier gave good results due to its ability to handle non-linearly separable data, imbalanced data sets, and good generalization performance. Also, the performance of the SVM classifier can also depend on the choice of kernel function, regularization parameters, and other hyperparameters, which need to be tuned carefully.

Since every performance analysis in SVC gave the best result we concluded that Support Vector Classifier as best classification model to detect Parkinson's disease based on dataset provided by UCI.

S.N	Performed By:	Model	Accuracy
1.	Agarwal A,Chadrayan S,Sahu S(2016)	Extreme learning	0.81
2.	Afzal Shahid, Maheshwari Singh	DNN	0.83(PIC)
3.	Tiwari H, Shridhar S, Patil P, Sichana K R, Aishwarya G	XGBOOST	0.95
4.	Ours	SVC	1.0

Table 4.2.2 Accuracy comparison

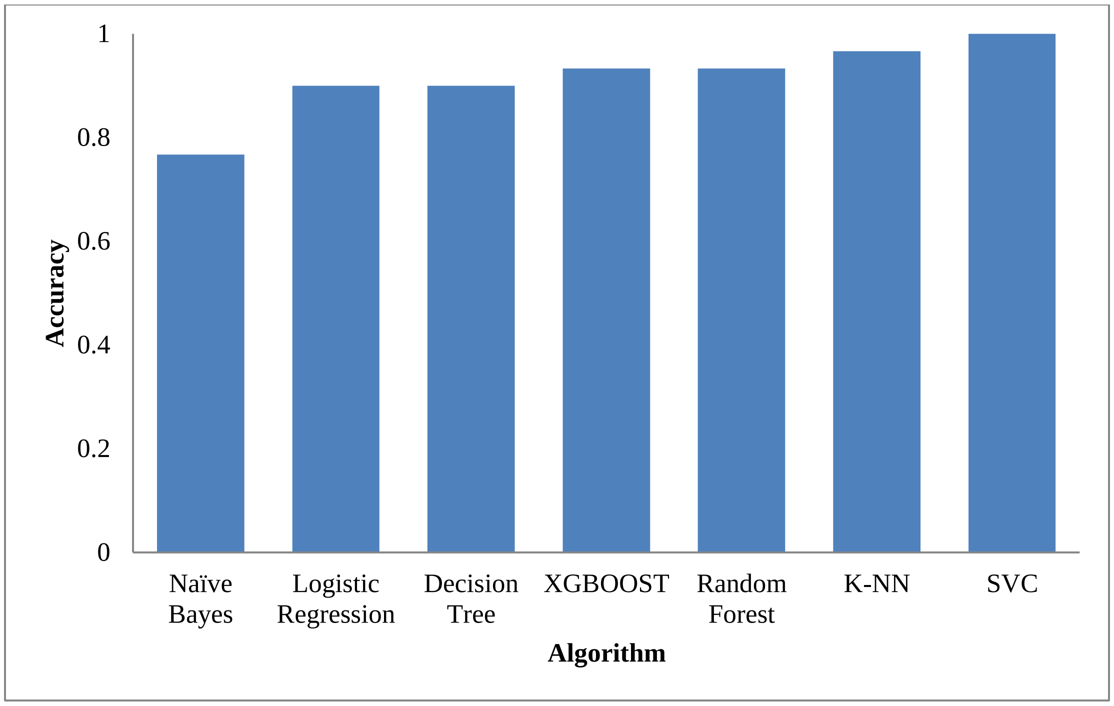


Figure 4.1.5.1 Accuracy- Model name bar graph

The above bar chart shows differences in accuracies of different models we tested on our program.

Chapter 5 Conclusion and Recommendation

Through research based approach we found out that machine learning and artificial intelligence can play a significant role in health domain. We created a program that could predict the likelihood of Pakinson disease with reliable accuracy.

5.1 Limitations

The PD data set from UCI depository consist of 23 features and 195 data which is not sufficient for traning especially in case of health domain. The dataset is comprised of 75% patients having PD (i.e Status = 1) and only 25 % remaining not having PD (i.e Status = 0). This brings biasness to the program model. We have used simple machine learning classification algorithm. Our interface is limited to input and output only.

5.2 Future Enhancement

Instead of simple ML classification algorithm, deep learning algorithm might have been more precise. The interface can be made more interactive. Instead of just predicting the likelihood of disease, we can add more information about the disease and create a platform which could be more helpful and insightful for the user.

References

- Max, A,. , Little, Patrick E,. , McSharry, Eric J,. , Hunter, Lorraine O,. and Ramig (2008),. *Parkinsons Data Set*. Retrieved from https://archive.ics.uci.edu/ml/datasets/Parkinsons?fbclid=IwAR3l-68Ck8qYgbeQMYdkTiiBRLOhoA7RJXefhSTfy8qCLPlH_epnsy78lyA

- Kirill, E,. and Ponteves,H. (2023, February 1). *Machine Learning A-Z™: Python & R in Data Science.* Retrieved February 11, 2023, from https://www.udemy.com/course/machinelearning/
- March, J .(n.d.*). Enthought Python Minimum Hardware Requirements.* Retrieved from https://support.enthought.com/hc/en-us/articles/204273874-Enthought-Python-Minimum-Hardware-Requirements
- Wu, Y,. , Chen, P,. , Yao, Y,. , Ye, X,. , Xiao, Y,. , Liao, L,. , Wu, M,. and Chen, J,. (2017, May 3). *Dysphonic Voice Pattern Analysis of Patients in Parkinson's Disease Using Minimum Interclass Probability Risk Feature Selection and Bagging Ensemble Learning Methods.* Retrieved February 11, 2023, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5434464/?fbclid=IwAR1xzi3aIZRD-Qo1AGwzhdfnFfMGcWXZjb4_4aXGawvFN8HC6c2s_aLDIZ8
- Youtube. (2022, September 26*). Machine Learning for Everybody – Full Course.* Retrieved February 11, 2023, from https://www.youtube.com/watch?v=i_LwzRVP7bg
- Gong, D,. (2022, February 23). *Top 6 Machine Learning Algorithms for Classification.* Retrieved February 11, 2023, from https://towardsdatascience.com/top-machine-learning-algorithms-for-classification-2197870ff501
- Wu, W,. , Lee, J,. , Harrou, F,. and Sun, Y(2020, August). *Early Detection of Parkinson's Disease Using Deep Learning and Machine Learning*. Retrieved March 10, 2023, from https://www.researchgate.net/publication/343625952_Early_Detection_of_Parkinson's_Disease_Using_Deep_Learning_and_Machine_Learning
- Shahid, H, A,. and Singh, P, M(2020, April 16). *A deep learning approach for prediction of Parkinson's disease progression*. Retrieved March 10,

2023, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7235154/?fbclid=IwAR0oC56e7Ro_jItzqiKgRrP4gx0S2zBR90zytgt7wmC6U_BwkYz-H_YYbbM

- Agarwal, A,. , Chandrayan, S and Sahu, S, S(2016, November 24). *Prediction of Parkinson's disease using speech signal with Extreme Learning Machine*. Retrieved March 10, 2023, from https://ieeexplore.ieee.org/document/7755419/authors#authors

APPENDIX

Python

Python is an interpreted, high-level, general purpose programming language created by Guido Van Rossum and first released in 1991, Python's design philosophy emphasizes code Readability with its notable use of significant White space. Its language constructs and object oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

Sckit-learn

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib

Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a statemachine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.

Seaborn

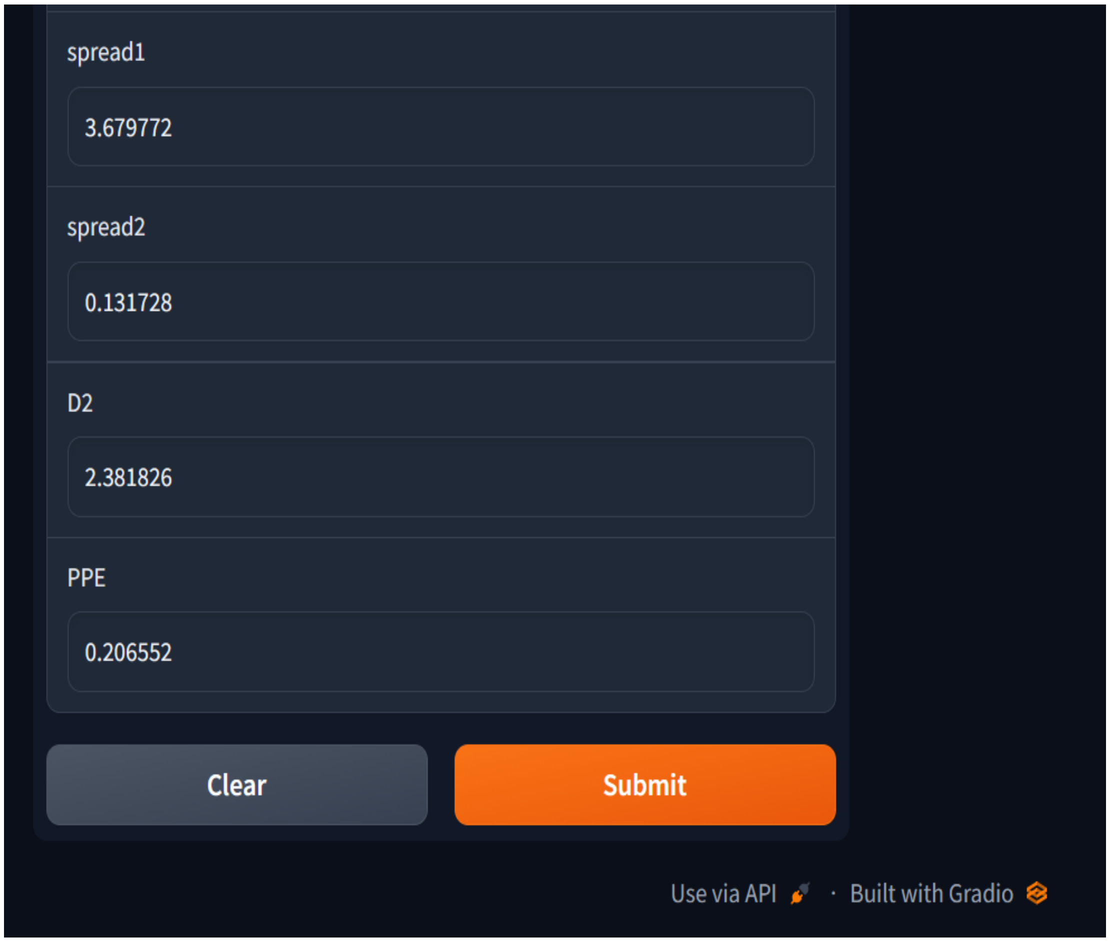
Seaborn is a Python data visualization library based on matplotlib. It provides a highlevel interface for drawing attractive and informative statistical graphics. Seaborn is a library in Python predominantly used for making statistical graphics. Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

Numpy

NumPy is a library for the python programming language, adding support for large, multi- dimensional arrays and matrices, along with a large collection of high level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim with contributions from several other developers. In 2005, Travis created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open source software and has many contributors.

Pandas

Pandas is a popular open-source data manipulation library for Python, designed to make working with structured data (i.e., tables, spreadsheets, databases) easier and more intuitive. It provides tools for reading and writing data from a variety of sources, including CSV, Excel, SQL databases, and more.


spread1

3.679772

spread2

0.131728

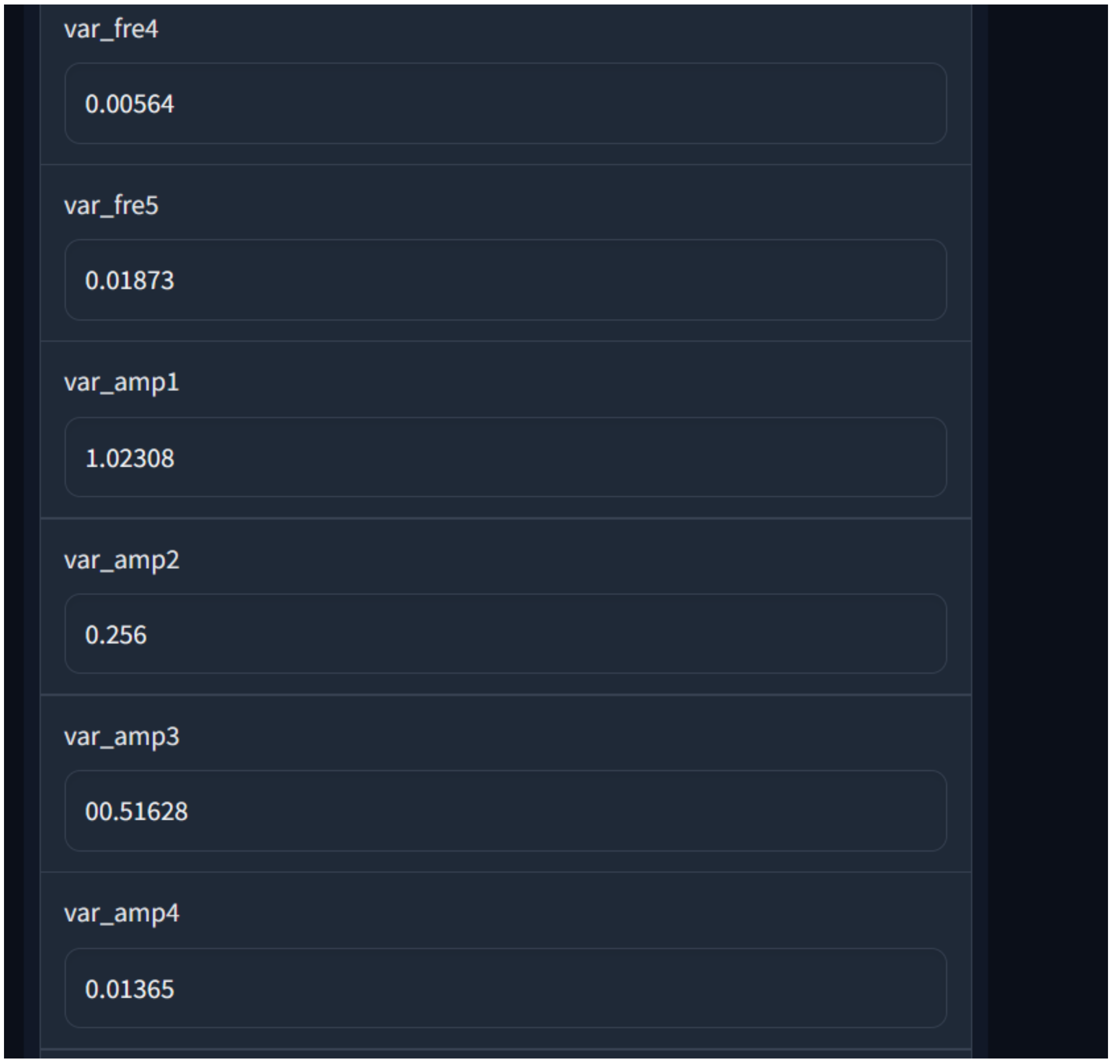
D2

2.381826

PPE

0.206552

Figure 4.1.5 Taking input from user


var_fre4

0.00564

var_fre5

0.01873

var_amp1

1.02308

var_amp2

0.256

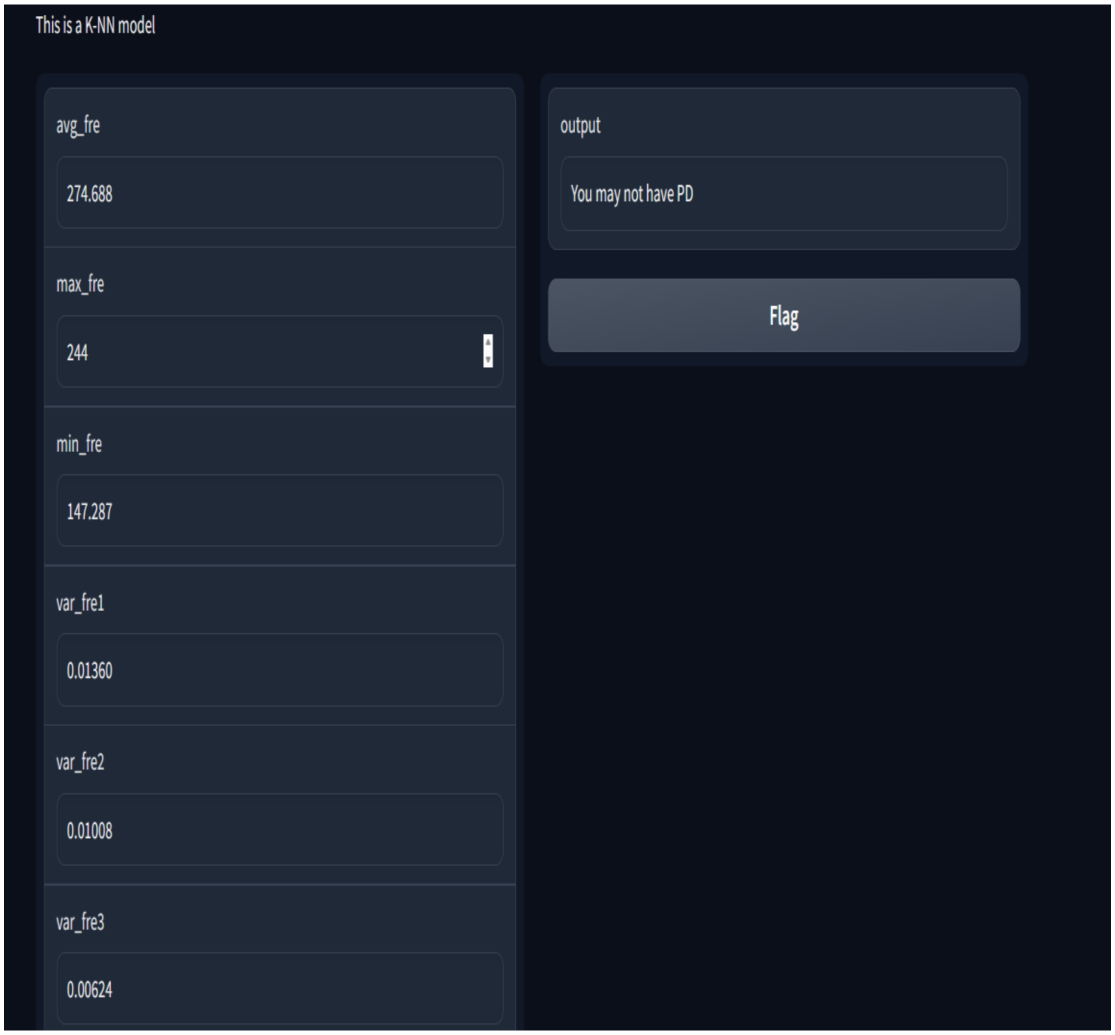
var_amp3

00.51628

var_amp4

0.01365

Figure 4.1.5.1 Taking input from user



This is a K-NN model

avg_fre

274.688

max_fre

244

min_fre

147.287

var_fre1

0.01360

var_fre2

0.01008

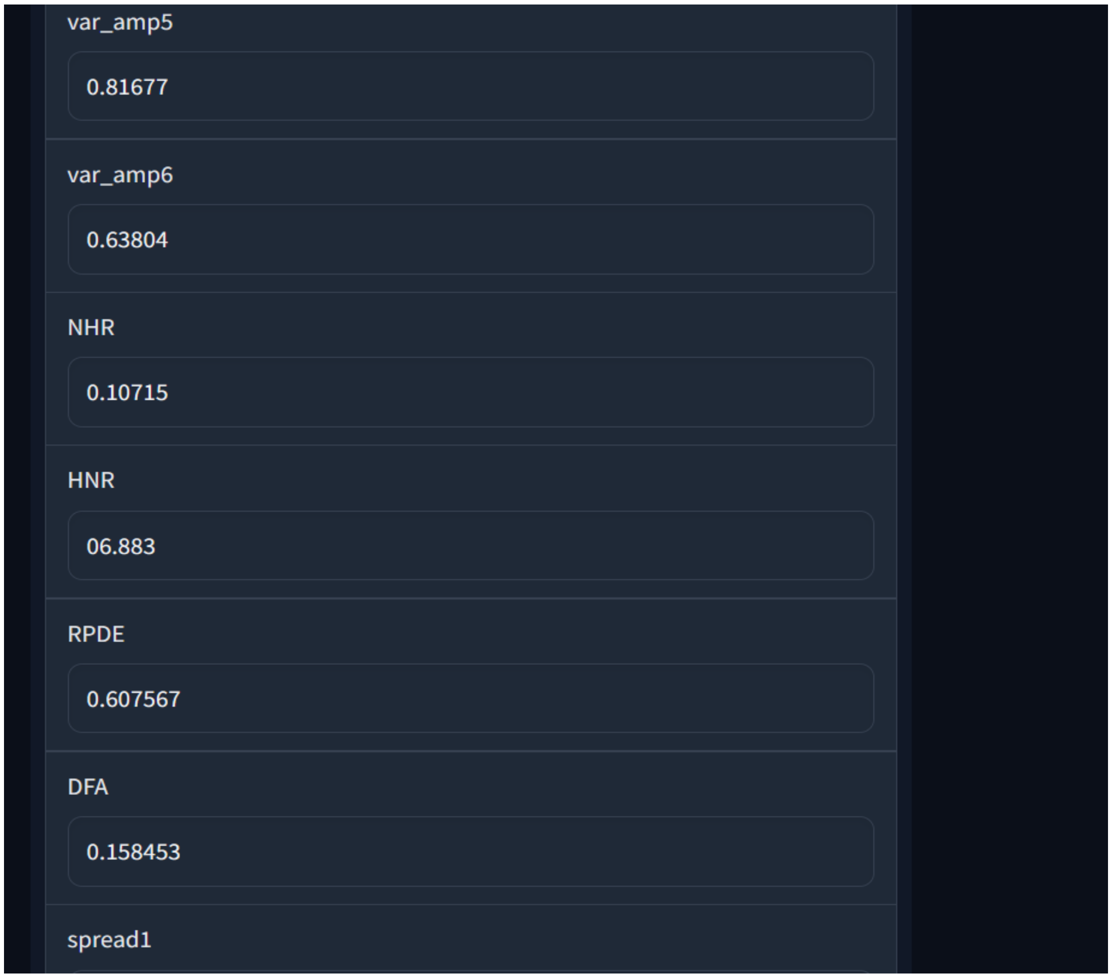
var_fre3

0.00624

output

You may not have PD

Figure 4.1.5.2 Predicting the result after taking inputs


var_amp5

0.81677

var_amp6

0.63804

NHR

0.10715

HNR

06.883

RPDE

0.607567

DFA

0.158453

spread1

Figure 4.1.5.3 Taking input from users

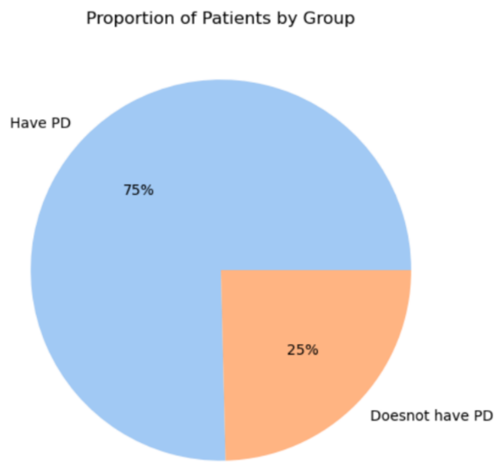


Figure 4.1.5.4 Pie-chart of people having PD and not having PD