Parkinson's Disease: Modeling a Binary Classifier for Early Detection

Nayan Arora
Faculty of Science and Technology
University of Canberra
Canberra, Australia
u3249907@uni.canberra.edu.au

***Abstract*— Parkinson's Disease (PD) is a growing health concern on a global scale. It is a disease that has no cure and is passed onto generations genetically in the DNA. It is a gradual worsening of neurological functioning of the brain. In the long-term is leads to a total damage of neural circuits, eventually leading to loss of movement in the body. Various studies have revealed that in the earliest indication of PD is the voice of an individual as their speech starts to deteriorate. Since it is exceptionally hard to diagnose PD, in recent times there has been an exponential increase in using artificially intelligent machine learning models to detect PD at an early stage using their non-evasive in nature on Vocal data. This report builds four different models using the Random Forest classifier, the XGBoost classifier, Logistic Regression and K-Nearest Neighbor classifier to find the best fitting model. Extensive data visualizations are used to form a basis for the pre-processing steps needed to produce accurate results. Using interpretations from the exploratory data analyses, several pre-processing steps are followed – class augmentation, scaling, standardization, winsorization and feature selection using analysis of variance and recursive feature elimination process. After tuning the hyperparameters, the models are built and evaluated using ROC curves, precision, accuracy, f-1, and recall scores as well as confusion matrices. The best overall result achieved is untuned XGB classifier, using all the features or selected features to get a 95% weighted accuracy on unseen data in both cases.**

Keywords*— *binary classification, parkinson's, AI, healthcare, anova, rfe, XGB, voice data.

I. INTRODUCTION

This report covers all aspects of the Pattern Recognition and Machine Learning project implemented through the course of this semester. The problem that is being tackled in this project is to create a machine learning model that can predict if a person is likely to suffer from Parkinson's based on their speech. Parkinson's Disease (PD) is categorized as a neurological disorder that may lead to gradual worsening of muscle stiffness, slowness in movement, impaired balance or coordination including the feeling of tremors [7]. All these symptoms worsen overtime that could lead to overall stopping of normal daily bodily functions. This could further lead to depression, difficulty swallowing and even urinary problems [7]. Till today, we do not really understand the underlying cause of this disease, but it is often associated with impairment or death of nerve cells in the brain's basal ganglia, resulting in reduced dopamine production [7].

The Parkinson's disease usually effects people over the age of 60 or it may also be carried in the genes, resulting in a hereditary cause. The current diagnosis relies on the family medical history along with a neurological examination to find any unusual protein lumps in the brain. There is no blood test that can help in diagnosis. Furthermore, the disease is not treatable and cannot be cured. Only medications can be provided to alleviate the symptoms and the primary therapy available for people suffering from the disease is called levodopa [7]. The overall goal of the therapy is to stimulate the brain deeply by exercising and other methods, while also maintaining a good therapy diet. In summary, because there is no cure to Parkinson's, it is really challenging both emotionally and physically for both the individual suffering as well as their families to cope up with it. Thus, extensive research has been conducted in recent years to analyze and look for methods that could help detect Parkinson's at an early stage.

Motivation

Since the disease gradually worsens, early detection and intervention will help delay the disease progression by following a rigorous care plan. By managing the symptoms early, we can prolong the worst-case scenario. Currently there are no blood or laboratory tests to diagnose non-genetic case of Parkinson's. So, a person might already be in the later stages of the disease before a correct diagnosis. This will have severe negative impact on the health of an individual. The successful execution of this project will result in a model that can be utilized of early detection of the disease. Early detection of the disease can help

healthcare practitioners to develop early treatments plans and therapies as well as providing the patients with appropriate medicines at early stages to keep the patients in good health whilst living with the disease.

I personally have a deep interest in learning and implementing innovative technological solutions in the health sector with an overall goal to improvise the current methods. I have previously worked on an image recognition project to detect the correct surgical tools depending on the type of operation to be conducted. This project helped eliminated the human error involved in delivering surgical tools to operation theatres. My personal motivation to devote my work and further learn and experience the implementation of technological solutions in the healthcare sector has since gone up. By way of this project, I aim to further expand my knowledge base.

II. LITEARURE REVIEW

A. Research Questions

The overall goal of the project is to create an in-depth understanding of the theory and implementation steps needed to develop a machine learning model that can make accurate predictions for a given scenario by recognizing and learning patterns in data. Some of the questions that we aim to answer through this project are:

- What are the methodologies and approaches towards diagnosing PD and how can we improve the current standards and methods of diagnosing PD?
- What types of data are collected in the dataset? Is there any correlation or causation in the data or in general, does the data need preprocessing?
- What types of ML algorithms are suited for a binary classification problem (0-No PD, 1-PD)?
- How to analyze and compare the performance of different ML algorithms like Logistic Regression, Decision Trees?
- What are the regulatory requirements and current limitations to achieve clinical validations for real-world application? And more.

The successful implementation of this model will result in a possible ML model that can aid healthcare practitioners with diagnosing PD. The generated results from the model can provide useful insights to healthcare practitioners that could help decide which patients should have further tests to detect PD. The role of the model is to act as information provider and will not be a basis for making final decisions.

B. Research on Algorithm Selection

One of the main research criterions was to read through research papers that previously implemented the ideologies of machine learning models on Vocal Data. The Voice data is not as like other data that is usually used for classification problems. In voice data, we have different ratio of noises, different amplitudes of voice and various other frequency related observations. This makes it necessary to implement only those models that are known to perform well with such vocal data. Below are the models that were finalized along with a description on the advantages of using them:

Logistic Regression: LR has proven efficient for various classification tasks due to its simplicity, ease of interpretability, and model transparency. LR is a member of the supervised classification algorithm family. The LR algorithm, by definition, measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function. Therefore, the LR algorithm will perform well on the supervised binary classification problem like the Parkinson's classification problem that deals with vocal data [3] [8].

Random Forest Classifier: This classifier is an ensemble learning method that combines the predictions of multiple decision trees which helps in achieving an overall improved performance and more generalized models. It is also robust to overfitting. It also averages the predictions of multiple decision trees, thus eliminating any noise in the data [1]. It has also been widely implemented for classification tasks with voice data [1].

XGBoost Classifier: XGBoost algorithm is a member of the ensemble algorithm family which is used for both classification and prediction. The algorithm creates multiple classifiers that are weak learners and combine them to get better overall performance. The XGBoost is chosen as it is also widely used for supervised classification tasks and a different approach will give us different insights. XGBoost Classifier has also been widely implemented for voice classification tasks and is known to perform well for such problems [18].

K-Nearest Neighbour: KNN is effective when local patterns and instance-based learning are critical. It does not make any assumptions about data and minimal training is required. It is also known as the lazy learner but is widely used for supervised classification problems. Hence it is a good choice. It is chosen based on the research [2].

Further research was done to finalize the above selected models. There are more options for different algorithms that could have been implemented (for ex, Naïve Bayesian, and others) but the above selections were made based on my understanding of the algorithm model, flowchart, and documentation available for the sklearn library as well as the documentation found through scholarly articles that verifies their performance on Voice related data.

C. Reference Solutions

A reference solution was found on Kaggle [9]. This solution claims to have achieved 99.2% overall accuracy after appropriate augmentation and tunings techniques. But I further researched the solution implementation techniques, and it cannot be confirmed that the model is overfitting. Neither has good modeling practices been followed for testing purposes. The author not only use synthetic class augmentation on the training set but also implemented and created 'unseen' synthetic values for the test set [9]. This is not a good practice as the test set should never be altered as it is considered the ground true values. Whereas, when we produce synthetic values for the minority class using the smote method in sklearn library we produce certain synthetic values that may or may not be true in the real world.

So, this report has used this solution as a starting point in brainstorming ideas and methods that could be implemented to achieve a highest fitting and best performing model for detecting and predicting PD.

III. Methodology

A. The Dataset Description

The chosen problem set is to use a Parkinson's data set [6] from the UCI Irvine Machine Learning Repository. This dataset consists of the Vocal Voice recordings of 31 individuals. There is a total of 195 recordings of which a total of 147 recordings have the 'status' as 1 (suffering from PD), and a total of 48 voice measurements have the 'status' as 0 (not suffering from PD).

Several medical studies have been conducted in the past summarized in [17], that have found that the voice of an individual is the earliest indicator of Parkinson's Disease. Therefore, we chose this dataset with voice measurements. A machine learning model will be built using this dataset to extract and learn patterns in the data that will help distinguish individuals suffering from PD from healthy individuals.

The dataset has over 6 voice recordings per patient, and 23 columns in which 22 columns represent different voice measurements of each recording and one column represents the status of either having or not having PD in form of 1s and 0s. So, the overall aim is to build a binary classifier with a high accuracy to make predictions on new unseen data.

B. Initial Analysis

First step implemented in my solution is to perform an initial analysis of various features in the data. There are a total 24 attributes or columns in the data as described below:

- name - ASCII subject name and recording number.
- MDVP:Fo(Hz) - Average vocal frequency.
- MDVP:Fhi(Hz) - Maximum vocal frequency.
- MDVP:Flo(Hz) - Minimum vocal frequency.
- MDVP:Jitter(%) - Measure of variation in frequency.
- MDVP:Jitter(Abs) - Measure of variation in frequency.
- MDVP:RAP - Measure of variation in frequency.
- MDVP:PPQ - Measure of variation in frequency.
- Jitter:DDP - Measure of variation in frequency .
- MDVP:Shimmer - Measures of variation in amplitude.
- MDVP:Shimmer(dB) - Measures of variation in amplitude.
- Shimmer:APQ3 - Measures of variation in amplitude.
- Shimmer:APQ5 - Measures of variation in amplitude.
- MDVP:APQ - Measures of variation in amplitude.
- Shimmer:DDA - Measures of variation in amplitude.
- NHR and HNR - Two measures of ratio of noise to tonal components in the voice.
- status - Health status of the subject (1) - Parkinson's, (0) – healthy.
- RPDE and D2 - Two nonlinear dynamical complexity measures.
- DFA - Signal fractal scaling exponent.
- spread1, spread2, and PPE - Three nonlinear measures of fundamental frequency variation.

Using these attributes, we plot and visualize the data to understand and further develop an understanding of the Pre-Processing steps needed for implementing multiple ML algorithms. In the attached code file various plots have been made that help create various inferences around the steps taken towards building my solution. The most important ones are included in this report.

We first plot the correlation heatmap, to visualize how the data is correlated. A high correlation in the data attributes and features basically means that there are many potential patterns in the data that the algorithm will be able to learn. Moreover, certain correlations can further help in the pre-processing stage by creating a

basis for the type of methods that will be implemented for feature selection and extraction.

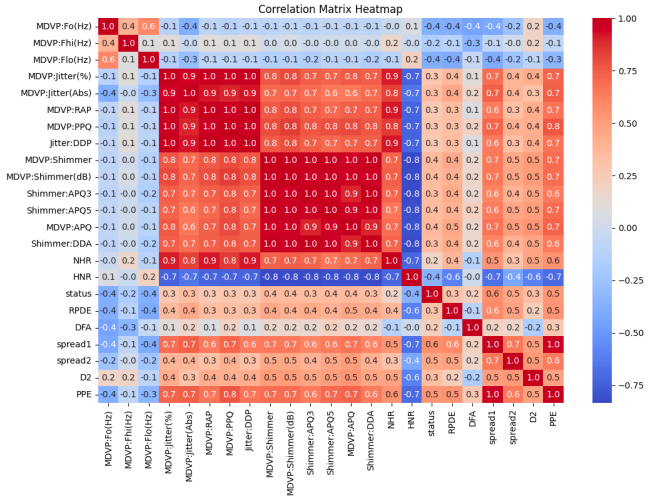

Figure 1: Correlation Heatmap

In Fig. 1 the matrix shows the measure of variation in frequency has multiple attributes (high, low avg, jitter, etc.) are all highly correlated which is a good sign. This means that the model building should be relatively straightforward.

Fig. 2 is a plot for average, minimum and maximum vocal frequencies. This plot helped create the inference that the data when considered as a whole is not skewed and has an almost normal distribution.

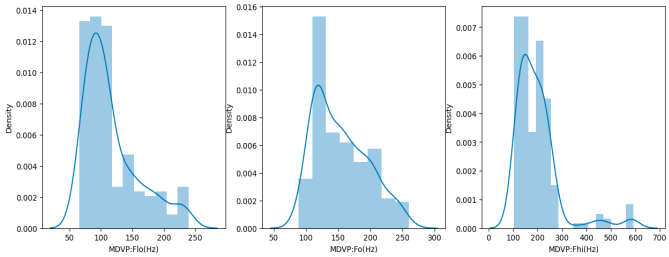

Figure 2: Vocal Frequencies

See the attached html or ipynb file for full scale plots. To further analyze and ensure that the features in the data are normally distributed and it's just the extreme values in the data that is causing the above plot to look skewed, we plot the QQ plots. QQ plots are basically a scatter plot with that represents the two quantities against each other. When both set of quantities come from the normal distribution, we notice an almost straight line. This plot helps ensure that the data attributes are not skewed. The plot has been generated for all attributes but only the one for MDVP:Fo(Hz), that is the average vocal frequency is included below:

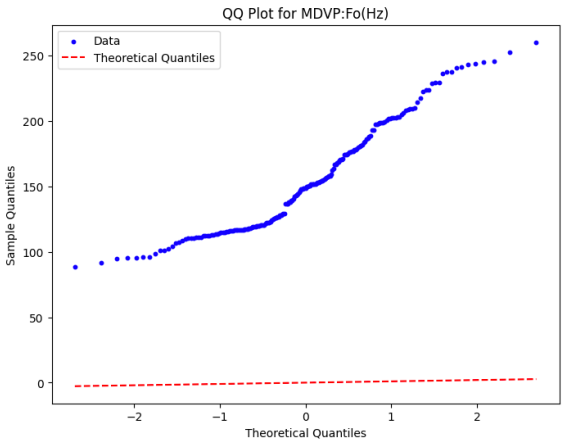

Figure 3: QQ plot for MDVP:Fo(Hz)

Fig. 3 shows an almost straight line in blue with certain extreme values on either ends. These extremities will be handled using Winorization in the pre-processing stage.

C. Data Pre-Processing

The very first step implemented in this stage is to remove any outliers from the data using a masking technique in Z-score analysis. The threshold used for calculating the z-scores was 3 standard deviations on either side of the graph. The rows with any outliers were then removed from the dataset. The initial shape of the data was (195, 22) and after removing the rows with outliers using z-score analysis, the dataset shape was (181, 22).

Next, the MinMax Scaler Standardization was performed. It is always a good practice to perform this step as scaling the values helps us ensure that all data features/attributes contribute equally to the model. This step is very useful and can be considered a requirement when we are using algorithms like Logistic Regression or any other gradient descent-based algorithms that are sensitive to feature scales.

Next, we perform the Standard Scaler optimization step. In this step we standardize all the attributes in our dataset by removing the mean and scaling to unit variance. This helps us scale the data to assume a standard normal distribution for all features which helps in reducing the impact of outliers in the data. This is an additional step that ensures that our data in normally distributed without any skewness that may cause bias.

Now using the Box-Cox transformation we estimate the value of lambda and if the evaluated value is close to 1, the transformation is approximately a natural logarithm [11]. Which it is. Output is generated for one row, but the result can be manually checked for all rows. Furthermore, if the result achieved is not close to 1 then we can perform a natural log transformation using np.log(array) to ensure a normal distribution [11].

The last step used in the pre-processing stage is to Winsorize the data. To winsorize data means to set extreme outliers equal to a specified percentile of the data. This step helps us handle the outliers found in the vocal frequency data. Fig. 4 shows the data scatter plot for maximum vocal frequency. It shows how the data is more normal than earlier.

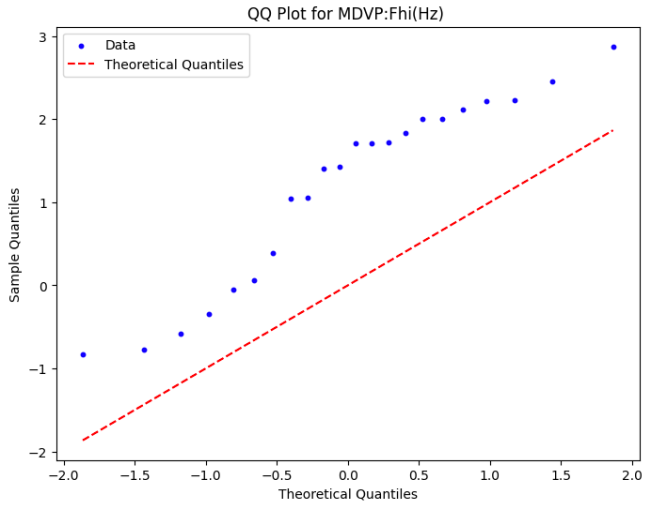

Figure 4: Max Vocal Frequency after Data Pre-Processing

Fig. 5 shows the boxplots of the frequency data after removing all the outliers.

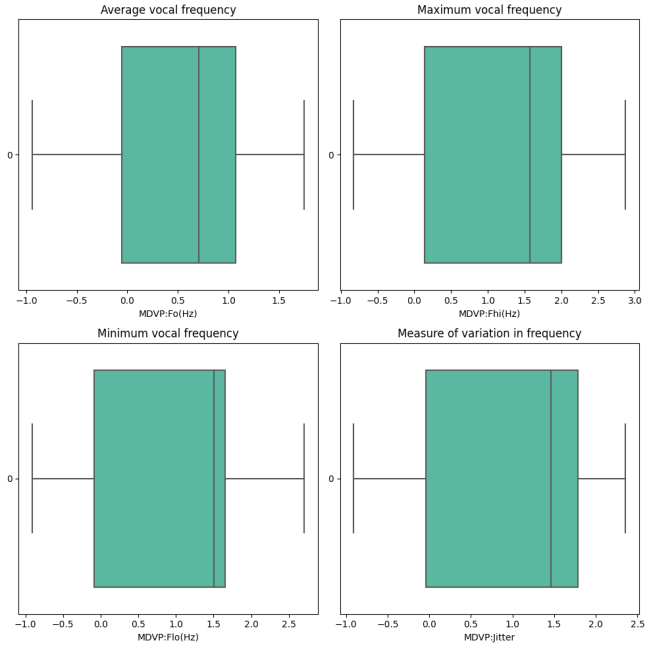

Figure 5: Boxplots after removing Outliers.

One additional pre-processing step used is class augmentation. Since the dataset does not follow the 10 times rule, we use the SMOTE method from the imblearn library under sklearn. Smote uses the ideologies of K-nearest neighbor to interpolate between existing minority class samples to produce new synthetic samples. Thus, it helps us fix the class imbalance and balance the overall class distributions that helps us to get a more generalized ML model.

D. Feature Extraction and Selection

There are various methods that can be used for feature extraction and selection process like Principal Component Analyses and others. The main aim we are trying to achieve in this stage of model development is to perform an analysis on the all the 22 attributes in the dataset to find the subset of attributes that learn the most accurate and generalized patterns and relations in the dataset.

Two methods ANOVA, Analysis of Variance and RFE, Recursive Feature Elimination are used for extracting a set of features with the most accurate predictions produced. ANOVA f-test accounts for one feature at a time and then each feature is examined based on how well it predicted the target or 'status'. This process helps us select only those features that are highly related to the target and will in turn be able to learn the patterns in the data better [12]. SelectKBest method from the sci-kit learn library was used to select the top best k features based on the accuracy score.

The RFE method is implemented as a wrapper method that recursively eliminates the features based on their classification accuracy [13]. It only selects the features that produces the most accurate classification results using a defined classifier without any tuning. Random Forest classifier was used for producing these results.

A 10-fold cross validation strategy was implemented to analyze the accuracy of the selected features produced at each iteration. This cross-validation technique helps ensure that there is no overfitting and the results produced are robust and generalizes the model well. Fig. 6 shows the resultant accuracies achieved for number of features after implementing the ANOVA and RFE feature selection processes.

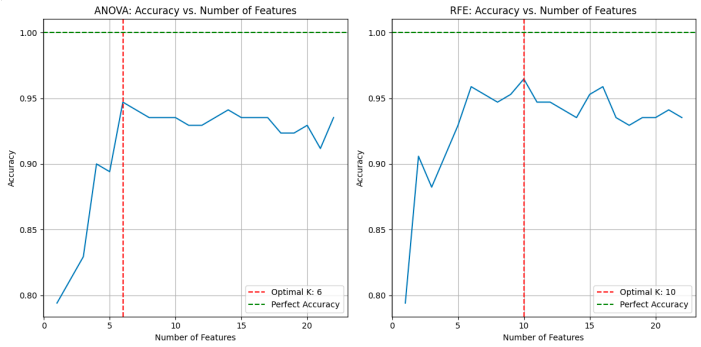

Figure 6: Accuracy vs No of Features.

The optimal ANOVA Feature Count achieved was 6 with a perfect accuracy of 1.0 and the optimal RFE Feature Count was 10 with an Accuracy of 1.0.

E. *Building Models*

The first step towards building a model is to generate the training and testing data. A two-fold approach to is used here to ensure that the final model built is generalized and not overfitting or underfitting. First, the data is split into Train, Test and Validation set. The split is 65-15-20, train-validate-test. After splitting the shapes are- (115, 22) for train, (29, 22) for validate and (37, 22) for test set.

Then, the original data is used again to create a supplementary train and test set with all the training data including all the features for final standalone model creation and testing. The split followed here is an 80-20 split.

The basic approach used for building the model and spot-checking the algorithms are to use the train, test and validate data split to first train the model using four different algorithms – Logistic Regression, Random Forest Classifier, XGBoost Classifier and KNNClassifier. The trained model is then evaluated using the validation set to ensure that there is no overfitting or underfitting cases. Then the best performing algorithms are chosen for further analysis.

The results achieved using this approach are covered in the next section.

IV. Evaluation of Resulsts

A. *Preliminary Analysis*

As explained earlier, we use the train and validate sets of data to evaluate the model performance using the chosen algorithms to find the best fitting model. Note that this step is performed without any hyperparameter tuning. This is to ensure that we are not producing a skew in the algorithm's performances. This step in the overall model development process is to choose the algorithm that produces the best and most generalized solution.

We evaluate the performance of all four algorithms using the ANOVA selected feature set, the RFE selected feature set and the original set of features. The results achieved using these steps are summarized in table 1.

TABLE 1: Summary of Performance

Initial Acc.	Original set		ANOVA set		RFE set	
	Train	Val	Train	Val	Train	Val
RF	0.93	0.90	0.93	0.93	0.92	0.90
XGB	0.93	0.97	0.95	0.93	0.92	0.93
KNN	0.90	0.93	0.94	0.93	0.93	0.93
LR	0.82	0.86	0.81	0.83	0.83	0.83

The training and validation accuracies achieved through this preliminary analysis help choose the algorithms that will be further used for hyperparameter training and further tuning to get the most optimal model. Note that the performance of each model on each type of dataset was done cross-validated using a 10-fold cross validation technique and the final accuracies included in Table 1 are the mean values obtained after performing the cross-validation.

For each algorithm's performance a learning curve was plotted, an example plot, in Fig. 7. The shaded region around both lines is the std-dev. The learning curve is used for calculating and plotting the training and validation scores for different training sizes using the 10-fold validation technique.

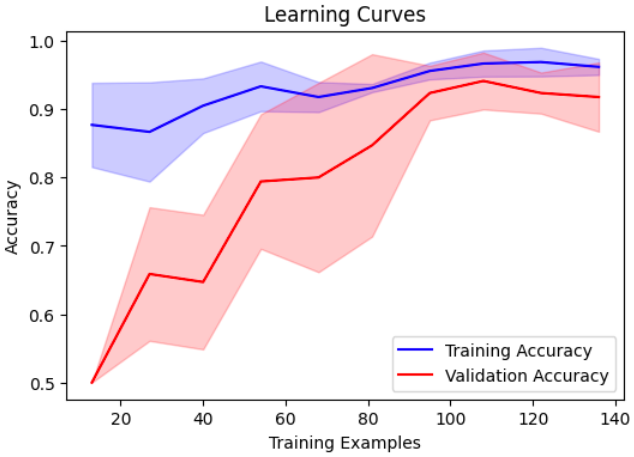

Figure 7: XGB algorithm performance on original feature set.

The learning curve can also help interpret and spot any cases of overfitting and underfitting relatively easily.

A total of 12 plots were generated in the process. Three for each algorithm (ANOVA, RFE, Original). The training and validation accuracies were printed along with a boxplot in Fig. 8, summarizing the performance for each type of feature evaluation.

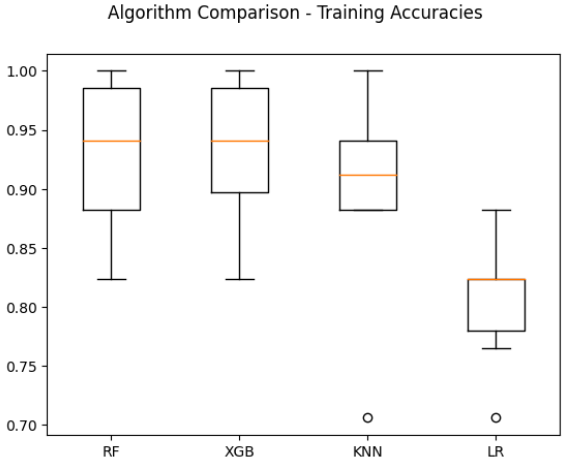

Figure 8: Comparing Preliminary Accuracies

Now, based on this initial analysis, further hyperparameter tuning and evaluation of results was done for Random Forest, KNN and XGB as the three algorithms had similar results. LR's performance being low was left out from any further evaluations.

B. Hyperparameter Tuning and Evaluations

In the final stage of model development, hyperparameter tuning was performed to get the best fitting results on unseen data or the test set. Since we have three training and test sets (original, anova, and rfe) we only choose the one on which we got the best performance from the earlier evaluations. That is, anova sets for all three algorithms. Below a detailed representation of results is included for each algorithm.

Random Forest Classifier: Using the anova selected set of features we train the model and make predictions on the unseen test set. The precision, recall, f-1 scores for which are in Fig. 9 below:

	precision	recall	f1-score	support
0	0.88	0.78	0.82	9
1	0.93	0.96	0.95	28
accuracy			0.92	37
macro avg	0.90	0.87	0.89	37
weighted avg	0.92	0.92	0.92	37

Figure 9: RF report without tuning

We get an overall average accuracy of 92%. Then a 10-fold cross validation Grid-Search is used to find the best fitting hyper-parameters [15] for the model as - {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}.

Note that both Grid-Search and Random-Search techniques were used for all the models but only Grid-Search was kept as the final results because it produced the best results, as expected.

After the Hyperparameter tuning of RF the tuned model was used for making predictions on the test data. The results achieved were the same as shown in Fig. 9. The overall result achieved showed no improvement.

K-Neighbors-Classifier: Using the anova selected features, the model was trained, and predictions were made on the unseen test set. Results shown in Fig. 10.

	precision	recall	f1-score	support
0	0.73	0.89	0.80	9
1	0.96	0.89	0.93	28
accuracy			0.89	37
macro avg	0.84	0.89	0.86	37
weighted avg	0.90	0.89	0.90	37

Figure 10: knn without tuning

A weighted accuracy of 90% was achieved but a better overall result was achieved in terms of the recall score, as less healthy patients were wrongly predicted to have PD. After using the 10-fold Grid-Search the best parameters [16] were {'n_neighbors': 7, 'p': 1, 'weights': 'uniform'}. The results were then revaluated after the hyperparameter tuning shown in Fig. 11.

	precision	recall	f1-score	support
0	0.67	0.89	0.76	9
1	0.96	0.86	0.91	28
accuracy			0.86	37
macro avg	0.81	0.87	0.83	37
weighted avg	0.89	0.86	0.87	37

Figure 11: knn after tuning

Fig. 11 shows how the results worsened after the tuning process, which is not a good sign and could mean that the model is now underfitting due to overtraining or other reasons.

XGBoost Classifier: Lastly, we follow a similar process to produce the results on unseen data after training using the anova set of features. Fig. 12 shows the classification report for this algorithm.

	precision	recall	f1-score	support
0	0.89	0.89	0.89	9
1	0.96	0.96	0.96	28
accuracy			0.95	37
macro avg	0.93	0.93	0.93	37
weighted avg	0.95	0.95	0.95	37

Figure 12: XGB without tuning

We get a weighted average accuracy of 95%, which is the best overall along with the best recall score yet. We now perform 10-fold grid-search evaluation to find the best parameters for the tuning stage [14]. The best parameters are {'colsample_bytree': 0.9, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 300, 'subsample': 0.8}. Now, the results were revaluated to produce a report after the tuning stage as shown in Fig. 13 below.

	precision	recall	f1-score	support
0	0.88	0.78	0.82	9
1	0.93	0.96	0.95	28
accuracy			0.92	37
macro avg	0.90	0.87	0.89	37
weighted avg	0.92	0.92	0.92	37

Figure 13: XGB after tuning

The results worsened which concludes that the best performance was achieved using the untuned XGB Classifier. This interpretation seems straightforward here, but multiple evaluation metrics were used to create this final inference on optimality. The evaluation metrics used and included in the attached code file include a confusion matrix that helps visualize the test cases against the training cases for each model. Furthermore, a bias versus variance using the cost-metrics analyses was plotted to visualize the tradeoff between the false-negative and false-positive predictions. The cost metrics

used was - Bias (Cost of FN): 5 and Variance (Cost of FP): 1. Fig. 14 shows the bias vs variance tradeoff plot for the best performing model, that is the XGB model without tuning.

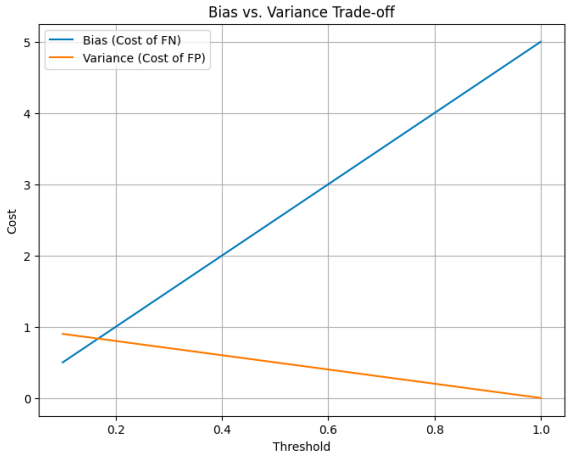

Figure 14: Cost metrics for xgb without tuning

It is summarized in Fig. 14, that the cost of false negative that is predicting a patient suffering from PD as healthy is much higher than predicting a healthy person as one with PD. This is a real-world inference as not being able to correctly predict PD would have serious implications and lack of trust in the model. In the predictions generated by XGB Classifier. The model wrongly predicted one time for both the cases but when implementing the model in a real- world scenario we have different cost related to each wrong prediction.

Overall, a based on the analysis of the precision, f-1, accuracy and recall scores along with the cost metrics, we can conclude that the XGBClassifier without any tuning is the best fitting and most optimal model for making predictions on unseen data.

As a cross-check evaluation metric, a ROC -area under curve (AUC) was plotted in Fig. 15. Here the model 1 is tuned random forest, model 2 is untuned XGB, and model 3 is untuned KNN. This plot basically confirms the results produced are valid and the overall inferences made are correct.

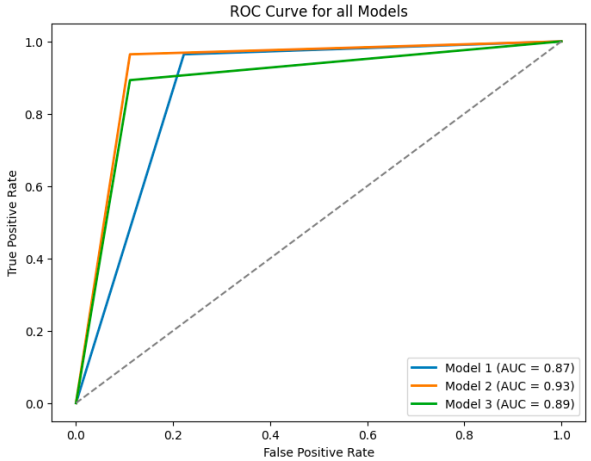

Figure 15: ROC area under curve for all models

C. *Optimal Results*

This section summarizes the evaluations and interpretations made in the previous section on how we chose the most optimal model using the confusion matrix plot.

TABLE 1: Interpreting Confusion Matrix

True values		Predicted values	
		Healthy (0)	PD (1)
0	Healthy	True Negative	False positive
1	PD	False Negative	True positive

Below are the optimal results produced by all the classifiers.

Random Forest Classifier:

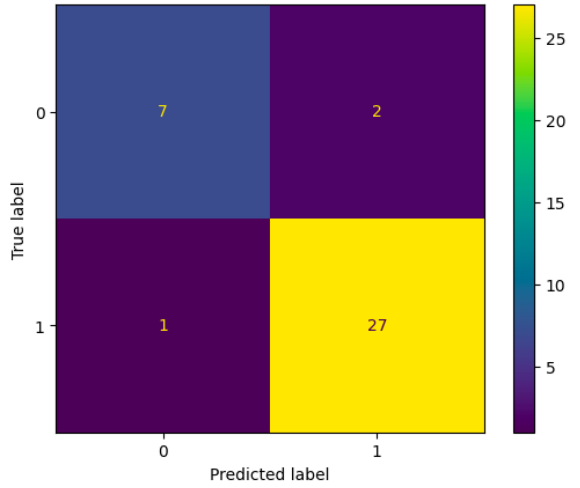

Figure 16: CM for RF

K-Neighbors-Classifier:

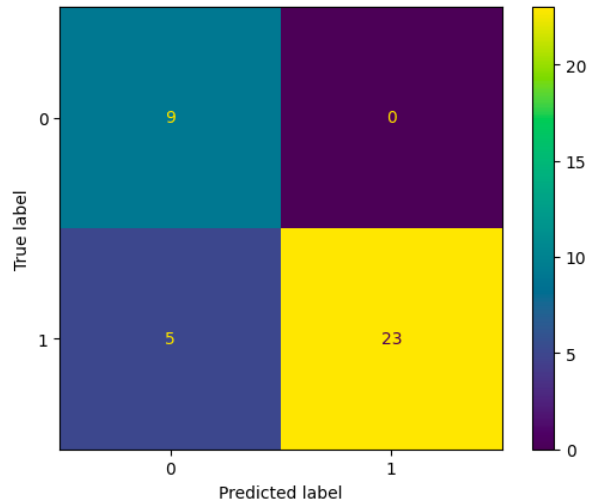

Figure 17: CM for KNN

XGBoost Classifier:

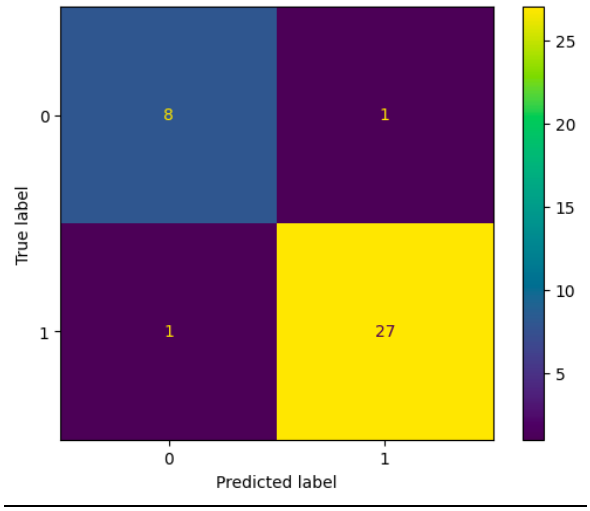

Figure 18: CM for XGB

Now, using our understanding of the cost metrics and the confusion matrices, it is easily inferred that the XGB Classifier model performed the best overall on unseen data as well as on the validation data earlier in the pre-processing stage. It is the ensemble learning strategy implemented in the XGB Classifier where it produces multiple weak learning decision trees that are then combined to get an overall performance that outperforms all the other learning models in this case.

Thus, the XGBClassifier model without any hyperparameter tuning will be used as the final most optimal model for creating a standalone model that can be updated for new test cases in the future.

D. Standalone Model for future work

The steps used for creating a standalone model using the entire training set is relatively straightforward. We use two lines of code to achieve this. First, we define the model as - final_standalone_model = XGBClassifier(). Next, we train the model using the entire training set as - final_standalone_model.fit (my_X_train, my_y_train). Here my_X_train and my_y_train are the entire original dataset on which the XGB model is trained.

E. Save Model for Future use

There are multiple methods we can use for saving our model. I have used the joblib library to achieve this, alternatively we can use the pickle library as well. We use joblib.dump with a defined file name to save the model as shown in Fig. 19.

```
import joblib

# Save the model to a file
model_filename = 'predict_parkinsons.pkl'
joblib.dump(final_standalone_model, model_filename)
```

✓ 0.0s

```
['predict_parkinsons.pkl']
```

Figure 19: Saving the model.

We can load the model again in the future to make predictions on new unseen data or further tune it as per the need of the situation as shown in Fig. 20.

```
# Load the saved model
my_model = joblib.load(model_filename)

new_data = my_X_test
# Make predictions using the loaded model
new_predictions = my_model.predict(new_data)

print(new_predictions)
print(classification_report(my_y_test, new_predictions))
```

✓ 0.0s

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 1 1 1 1 1 0 1 1 0 1
 1 1]
              precision    recall  f1-score   support

           0       1.00      0.71      0.83         7
           1       0.94      1.00      0.97        32

    accuracy                           0.95        39
   macro avg       0.97      0.86      0.90        39
weighted avg       0.95      0.95      0.95        39
```

Figure 20: Final Results

In the figure above we load the model using the joblib library and then make predictions on unseen original test dataset to get the predicted classification results and a weighted accuracy of 95% as was achieved by the anova test set in the validation stages. Thus, confirming the performance of the model is optimum on unseen dataset.

V. Ethical and Privacy Considerations

There are various ethical and privacy related issues that can emerge in the process of building a machine learning model to predict Parkinson's Disease. Below is a list of key issues along with the potential methods we can use to address them.

- Individual's Consent: We need data for training a model and the first step needed to collect any data is to get an individual's consent of having their personal information used. This can be done by using a strict process of first availing a signed consent waiver form before collecting any data.
- Data Privacy and Security: This is always one of the most important aspects. We need to protect data from any unauthorized access or misuse. This can be done by implementing strategies such as data encryption, access controls, etc.
- Bias: Any bias in the data can lead to serious drawbacks or even a total failure of the project. It is important to ensure accurate results without any bias especially for health-related projects. One strategy we can implement to ensure non-biased data collected is to use diverse groups of people across the world who have suffered from PD. This will account for any geographical bias that may be present due to an individual's

upbringing in that area. For example, the implications of their routine, the food they eat, etc. could then be accounted for.

- Model Transparency and Awareness: It is understood from the results achieved that the model built is not perfect and even when we build a perfect model it does not always ensure the correct results in real-world scenarios. We can take the example of Tesla's self-driving cars. The software running the car is near prefect which is the reason behind its public release. But same software has not only prevented accidents but led to accidents due to unfamiliar road conditions. Since an artificially intelligent machine learning model is based on the ideologies of identifying and learning patterns, it is important to spread awareness that if and when a new pattern emerges the model would fail to recognize it unless it is taught to first learn that. Thus, it was stated in the earlier stages of this report that the role of the model is to act as information provider and not be a basis for making final decisions.

VI. Conclusion and Future Work

In conclusion, by implementing various strategies across the board it is found that the XGB model with or without the anova selected feature set has the best overall performance of 95%. By way of this project, we have done a deep analysis of the Parkinson's Classification problem and have identified the optimal results based on multiple approaches with multiple evaluation strategy for each approach. The results achieved also summarizes the exceptional performance of the XGB Classifier on Voice Data.

One of the major limitations of this project has been the size of the dataset. In my inference the small size of the dataset was one of the major reasons behind the performance dropping after hyperparameter tuning for some models. This problem cannot be easily fixed. The personal due diligence performed in this project to use class augmentation and balance the two classes (0,1) that helped produce more synthetic data. But this does not really increase the overall size of the data by a lot. Producing more synthetic data will also not help our case because it will lead to unreal results. Thus, in my future work I aim to look for methods that can be used to tackle such a problem of building a generalized solution using a small dataset. I am sure that with the change and advances in technology, there should be new methods that are able to tackle such problems in real time.

The current implementation summarizes the process that can be used for building a ML model to provide an AI solution in the health sector. Implementation of this project should act as further encouragement towards building more AI solutions for hospitals and doctors.

VII. Acknowledgements

I would also like to take this opportunity to acknowledge and thank the unit convener Mr. Dharmendra Sharma for providing this opportunity to work on an extensive research project. I would also like to thank DR. Ambikesh Jayal and Dr. Asmaa Elsaeidy for their help and guidance throughout this venture. It would not have been possible to achieve any results, without their continuous support. I deeply appreciate it.

References

[1] N. Beheshti, "Random Forest Classification," *Medium*, Jan. 28, 2022. https://towardsdatascience.com/random-forest-classification-678e551462f5

[2] R. Thiruvengatanadhan, "Speech/Music Classification using MFCC and KNN," *International Journal of Computational Intelligence Research*, vol. 13, no. 10, pp. 2449–2452, 2017, Available: https://www.ripublication.com/Openaccess/ijcirv13n10_14.pdf

[3] D. Castillo, "Machine Learning Regression Explained," *Seldon*, Oct. 29, 2021. https://www.seldon.io/machine-learning-regression-explained

[4] Shubham, "Pattern Recognition | Phases and Activities," *GeeksforGeeks*, Sep. 02, 2019. https://www.geeksforgeeks.org/pattern-recognition-phases-and-activities/

[5] Kolamanvitha, "Design Patterns for Machine Learning," *Medium*, Jul. 19, 2021. https://towardsdatascience.com/design-patterns-for-machine-learning-410be845c0db

[6] U. ML Repository, "UCI Machine Learning Repository," *archive.ics.uci.edu*. https://archive.ics.uci.edu/dataset/174/parkinsons

[7] "Parkinson's Disease: Causes, Symptoms, and Treatments," *National Institute on Aging*. https://www.nia.nih.gov/health/parkinsons-disease#:~:text=The%2520main%2520therapy%2520for%2520Parkinson (accessed Nov. 02, 2023).

[8] A. Nikhil, "Implementation of Logistic Regression from scratch," *Medium*, Jun. 20, 2021. https://blog.devgenius.io/step-by-step-implementation-of-logistic-regression-on-gender-voice-classifcation-dataset-from-535671cb281c (accessed Nov. 03, 2023).

[9] "Parkinsons Classification 99.2%," *kaggle.com*. https://www.kaggle.com/code/bahaakhaled97/parkinsons-classification-99-2/comments (accessed Nov. 03, 2023).

[10] Zach, "How to Winsorize Data: Definition & Examples," *Statology*, Jan. 22, 2021. https://www.statology.org/winsorize/

[11] R. Pannell, "The Box-Cox Transformation: What It Is and How to Use It - LeanScape." https://leanscape.io/the-box-cox-transformation-what-it-is-and-how-to-use-it/#:~:text=The%20Box%20Cox%20transformation%20is%20a%20statistical%20technique%20used%20to (accessed Nov. 03, 2023).

[12] sampath kumar gajawada, "ANOVA for Feature Selection in Machine Learning," *Medium*, Oct. 20, 2019. https://towardsdatascience.com/anova-for-feature-selection-in-machine-learning-d9305e228476

[13] E. G. PhD, "Recursive Feature Elimination: A Powerful Technique for Feature Selection in Machine Learning," *Medium*, Sep. 02, 2023. https://medium.com/@evertongomede/recursive-feature-elimination-a-powerful-technique-for-feature-selection-in-machine-learning-89b3c2f3c26a (accessed Nov. 03, 2023).

[14] C. Spark, "Hyperparameter tuning in XGBoost," *Medium*, Dec. 23, 2019. https://blog.cambridgespark.com/hyperparameter-tuning-in-xgboost-4ff9100a3b2f

[15] W. Koehrsen, "Hyperparameter Tuning the Random Forest in Python," *Medium*, Jan. 10, 2018. https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

[16] A. Martulandi, "K-Nearest Neighbors in Python + Hyperparameters Tuning," *Medium*, Oct. 24, 2019. https://medium.datadriveninvestor.com/k-nearest-neighbors-in-python-hyperparameters-tuning-716734bc557f

[17] Diagnosis of parkinson's disease using fuzzy c-means clustering https://www.researchgate.net/publication/283453672_Diagnosis_of_Parkinson's_Disease_using_Fuzzy_C-Means_Clustering_and_Pattern_Recognition (accessed Nov. 3, 2023).

[18] "Identifying male or female based on voice using XGBoost and ML," TechieYan Technologies, https://techieyantechnologies.com/identifying-male-or-female-based-on-voice-using-xgboost-and-ml/ (accessed Nov. 3, 2023).