Perfomance Investigation of Machine Learning Approaches for Speech-Based Parkison's Disease Diagnosis

MATH5836 Data Mining Assignmet 2

Vasiliki Vamvaka - z5251098
School of Mathematics and Statistics
University of New South Wales,NSW
Australia,2052
v.vamvaka@student.unsw.edu.au

***Abstract*—This report summarizes the evaluation of different classification schemes applied on the diagnosis of Parkinson's disease from speech data. The methods applied were Neural Networks, Decision Trees and Ensemble Learning. The speech data of two separate Parkinson datasets for UCI Machine Learning Repository were used. The performance of the techniques was assessed based on the confusion matrices. More specifically, the AUC, F1 Score, specificity, and sensitivity metrics were used to estimate the accuracy of the classification performance. Different model parameters were investigated for each approach. Overall, Bagging delivered the best results for the first dataset with accuracy 74.32% and AdaBoost for second dataset with 89.33% accuracy.**

Keywords—SGD, ADAM, ANN, Decision Trees Ensemble Learning, PD diagnosis

I. INTRODUCTION

Parkinson's disease (PD) has been identified as the number two most common neurological disorder among the elderly people with symptoms in speech, behavior, mental processing, and motor reflexes [1]. The main cause of PD has not been established so far, making its diagnosis a rather crucial task [2]. There are two main categories of PD diagnosis, namely the invasive and non-invasive methods, with the latter ones being the most popular due to their low cost and ease of implementation [3]. The non-invasive PD diagnosis is mainly based on voice data and classification to diagnose impairments on speech such as dysphonia, hypophonia, dysarthria and monotone [4].

Recently, speech based PD diagnosis has gain momentum as it is estimated that 90% of the patients suffering from PD, might present certain language difficulties signifying early stages of PD [5]. A study reported in [6] investigates the performance of different machine learning methods for PD diagnosis based on dysphonia measures. The performance was evaluated via 10-fold cross validation. More specifically, seven classifiers were used , namely, linear discriminant analysis (LDA), k-nearest neighbors (k-NN), naïve Bayes (NB), regression trees (RT), radial basis function neural networks (RBFNN), support vector machine (SVM) and Mahalanobis distance (MDC). The authors used 22 voice features from a 295 unbalanced speech dataset, achieving the highest average accuracy of 92% with a standard deviation of 0.02 using SVM. On the other hand, the worst performing classifier was found to be MDC with average accuracy of 63% and a standard deviation of 0.13. To further improve the performance of SVM and achieve better precision and F-measure, the authors suggested including the use of feature extraction techniques. Moreover, in [7] SVM was also used as a binary classifier to detect whether a person is healthy or has PD. In this study feature selection was also considered by empolying a maximum-relevance-minimum-redundancy (mRMR) scheme to reduce the dimensionality of the input features. The dataset comprised by a total number of 195 recordings with 22 features per recording. In addition, the proposed leave-one-individual-out validation scheme was used to eliminate any estimation biasing and assess the classification performance of SVM. After optimizing the SVM with mRMR, the highest accuracy of 81.53% and standard deviation of 2.17% was achieved with a minimal subset of only 4 features. The highest classification accuracy of 99.64% ± 0.01 using SVM for PD detection was reported in [8]. The main focus of this research was to investigate the impact of different dysphonia measurement in PD on the classification rate of SVM. It was found that by considering the variability of measurements across healthy and unhealthy subject, SVM's accuracy could be greatly improved.

Apart from the classic SVM classifier, Artificial Neural Networks (ANNs) have also been employed to diagnose PD based on speech data. An attempt to evaluate the suitability of ANNs for PD diagnosis along with several data pre-processing techniques can be found in [9]. The authors used a Multi-Layer Perceptron (MPL) based ANN that was tested against ten-fold cross validation, 0.8 validation split, and 0.7 validation split across 12 independent experimental runs. At the same time, different pre-processing methods were evaluated using the confusion matrix. Discretize + Resample performed better with ten-fold cross validation and 0.8 validation split giving a ROC of 96.7 and 100 respectively, while Resample+SMOTE was proven to give the best accuracy with an 0.7 validation split with an ROC of 95.8.Another approach to speech-based PD detection is by using Artificial Neural Networks (ANNs) [10]. The ANN was fine-tuned using Levenberg-Marquardt (LM) optimization with a randomized 0.8 validation split parameter. The reported classification accuracy of PD was 94.93%, however the authors did not refer to any limitation or biasing of the proposed ANN. In [11] the authors tried to classify Parkinson's disease based on ANNs. More specifically, feed forward ANNs were employed in both one and multi-dimensional configurations with different architectures in terms of hidden layers and number of neurons per layer. The effect of different feature selection techniques such as SOM, PCA, Pearson's and Kendall's correlation coefficient was assessed according to their

accuracy, sensitivity, and specificity. The best performing accuracy of 81.33% was observed using one ANN with 10 neurons and a single hidden layer and Kendall's correlation coefficient as feature selection method. On the other hand, in the case of multiple ANNs topology, two hidden layers with 10 neurons in each was proven to be the most suitable architecture for PD diagnosis. However, the authors suggested that ANN classification performance on PD speech data must be improved in order to be considered for clinical diagnosis. Finally, the performance of parallel ANNs compared to a single ANN was also investigated in [12]. The authors designed an ANN based on LM training algorithm with 3 hidden layers and 20 neurons per layer. This ANN was then expanded into a parallel configuration consisting of either 3,5,7 or 9 ANNs. In addition, a majority voting decision scheme was applied to the output of the parallel architecture to increase the robustness of the prediction. The unlearned data from each individual ANN was also sequentially fed to the next ANN. The classification accuracy of the proposed architecture on an imbalanced PD speech dataset was measured to be 91.20±1.6% in the case of 9 parallel ANNs, indicating a performance improvement of 8.4% against the single ANN architecture.

In addition, Decision Trees and Ensemble Learning have also been used as classification methods for PD diagnosis from speech data. In [13] the authors compared the performance of four different classification techniques, namely Decision Trees, ANNs, DMneural and regression. The classification performance was evaluated on a dataset containing 23 samples for 31 test subjects, with 23 being PD positive. Each patient had six recordings, summing up to 195 voice recordings. The authors, prior to the application of the four methods, applied variable selection techniques, to reduce the number of inputs. Following that, the dataset was split to training 0.65 and test 0.35 sets and then the four classification techniques were applied. The best performing classifier was the ANN with one hidden layer and 10 neurons with an accuracy of 92.9% while the Decision Trees achieved an accuracy of 84.3%. Another implementation of Decision Trees for PD diagnosis was performed in [14]. The authors attempted a non-linear feature selection using PCA where it was found that 11 of the 23 features were enough to explain the dataset. Different classification approaches such as Regression RPART, C4.5, PART, Bagging CART, Random Forest (RF) and Boosted C5.0 were applied on the dataset both initially and after PCA. Accuracy, sensitivity and specificity were used as measures of the performance of the aforementioned techniques with Random Forest classifier yielding the best performance with an accuracy of 96.87% on the dataset after feature selection. Moreover, in [15] CART and Ensemble learning (RF, SVM, Extreme Learning Machine ELM) was used on PD speech data. Initially, CART was used to obtain the optimal samples which are the most helpful in terms of correct classification. Then Ensemble Learning consisting of RF, SVM and Extreme Learning Machine (ELM) classifiers was used to classify the training samples based on the results from the CART algorithm. The results showed that the authors achieved maximum accuracy of 75.5% under Leave-One-Out (LOO) and 90% for Leave-One-Subject-Out (LOSO) technique. A comparison between RF and Ensemble Learning can be found on [16]. The authors introduced the multi-edit-nearest-neighbor algorithm (MENN) to optimize the number of training samples, then applied RF and decorrelated neural network ensembles (DNNE) for the classification of the PD patients. To assess the accuracy the LOSO and LOO cross validation techniques were used. Overall, the MENN algorithm combined with Random Forest and LOSO yielded the best accuracy of 73.37% in the training data. However, the test dataset accuracy was found to be 100%. To verify that, the authors applied the MENN with RF and LOO on an unbalanced PD dataset with a resulting test accuracy of 87.8%.

From the literature review, it is obvious that SVM represents a more robust and popular method for PD diagnosis based on speech data such as dysphonia measurements. However, the potential of ANNs on speech-based PD diagnosis is great and worth investigating as indicated by certain attempts [11, 12]. Furthermore, Decision Trees and Ensemble Learning classifiers are promising candidates classifying PD patients based on voice recording. In certain studies, Decision Trees and Ensemble Learning also presented the lowest classification error as compared to ANNs [17, 18]. Therefore, the purpose of this report is twofold. Firstly, ANNs as classifiers for speech-based PD diagnosis will be investigated. More specifically, the performance of two optimizers will be investigated namely Adam and SGD for different moment and learning rates as well as ANN with different number of neurons and depth. The accuracy will be assessed across 10 independent experimental runs for a validation split of 0.2. Secondly, the implementation of Decision Trees and Ensemble Learning for PD diagnosis will be explored. The performance of Full Trees, Pruned Trees, and Random Forest will be assessed. Additionally, techniques that belong to Ensemble Learning such as Bagging and Boosting; gradient Boosting, XGBoost and AdaBoost will also be studied. The final classification performance comparison will be presented in terms of the confusion matrix for each classifier. The overall performance investigation and comparison is completed on two different datasets containing PD speech data from the UCI database, namely *Dataset1*[4] and *Dataset2*[19]. The paper is organized as follows; in Section II the two datasets are cited and analyzed along with our proposed algorithm. Section III outlines the experimental evaluation on *Dataset1* and presents the obtained results. Similarly, Section IV outlines the experimental evaluation on *Dataset2* and presents the obtained results. In Section V the classification performance of the different classifiers is compared by also considering the characteristics of each dataset. Finally, the paper concludes with Section VI

II. Problem Definition and Algorithm

A. Data Set and Task Definition

The objective of this study can be divided into two parts. The first part is devoted in evaluating the testing and training performance of SGD and Adam optimizer when applying ANNs for PD speech data classification of positive or negative test subjects. Moreover, the second part of the study deals with the performance of DT and Ensemble Learning techniques for speech-based PD diagnosis. Two distinct PD speech data sets are used which are referred to as *Dataset1* and *Dataset2*, respectively.

Dataset1 refers to 40 test subjects with 20 healthy individuals and 20 PD diagnosed ones. For each subject, 26 voice samples were recorded, giving a total of 1040

different voice samples. The voice samples included sustained vowels, numbers, words, and short sentences. Furthermore, 26 handcrafted time-frequency based features were extracted using Praat acoustic analysis software [4]. *Dataset2* contains 80 test subjects where 40 of them were identified with PD. Three vowel recordings were taken per test subject giving a total of 240 voice recordings. Moreover, 446 acoustic features were extracted by using a waveform matching algorithm as outlined in [19]. Exploratory analysis was performed on both datasets, identifying the distribution of the classes and the correlation plots. *Dataset1* and *Dataset2* were then split into train and test randomly with 60% used during training and 40% used during the testing phase.

B. *ANNs and Algorithm Definition*

The inputs to the ANN were the features from the dataset and the output was defined as 0 for no PD diagnosed or 1 for PD diagnose as can be seen in Fig.1. After defining the model for the ANN, two different optimizers were investigated in terms of their test and training performance. Initially, the train accuracy and loss as well as the testing accuracy and loss of SGD and Adam were evaluated for different number of hidden neurons in the range of 5 to 25 for a single hidden layer. Then, the learning rate was varied from 0.1 to 1 and the testing and train performance was recoded. Following that, only the momentum rate of SGD was changed from 0.001 to 0.1 observing the loss and accuracy of the fine-tuned ANN. Moreover, the effect of different number of hidden layers starting from 1 to 4 was also investigated in terms of accuracy and loss. Finally, the overall performance of the ANN under SGD or Adam was examined using a confusion matrix and looking at the ROC, AUC, Precision-recall curve and F1 score. The analysis of the optimized ANN was performed for 10 experimental runs to handle the effect of randomness.

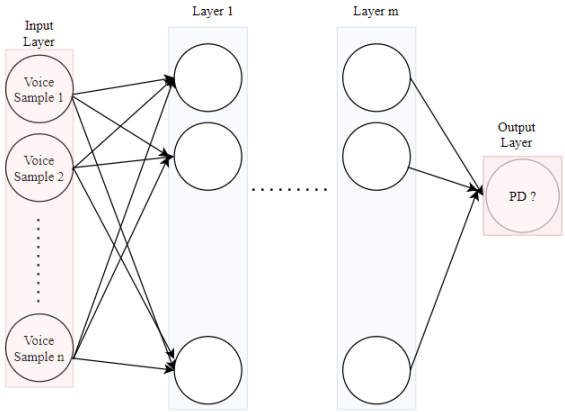


Fig.1. Artificial Neural Network illustration for Parkison's disease classification where Voice Samples represent features

The algorithm that was used to assess the performance of SGD and ADAM on the Parkinson's speech data is based on the waterfall model. The idea of the waterfall approach comes from the area of engineering design in which each activity is a linear sequence of the previous one and "flows" towards one direction, hence the name. The first step was the investigation for a number of neurons in the range of 5 to 25, while fixing the learning rate and momentum rate for each optimizer fixed. After assessing the best model, the learning rate was investigated for ADAM and SGD while holding the rest hyperparameters fixed. Then, by using the "best" model from the previous results, a momentum rate was chosen. Finally, some experiments took place for several hidden layers with various number of neurons. A pseudocode of the waterfall approach is illustrated below where *lr* denotes the learning rate and *mr* denotes the momentum rate:

1. *For neurons in (5,10,15,20,25):*
   - *For i = 1, 2…,10 do*
   - *Fit the NN model with lr = t and mr = w*
2. *For lr in (0.1,0.2,0.3,0.5,1):*
   - *For i = 1, 2…,10 do*
   - *Fit the NN model with neurons = best from step 1 and mr = w*
3. *For mr in (0.01,0.02,0.04,0.07,0.1,0.9):*
   - *For i = 1, 2…,10 do*
   - *Fit NN model with neurons = best from step 1 and lr = best from step 2*

C. *Decission Trees and Ensemble Learning Algorithm Definition*

Decision trees belong to the supervised learning algorithms and make use of *"if and then"* rules for prediction of a target outcome. Decision trees or CART (Classification and Regression Trees) can be used both for classification and regression tasks. CART trees' architecture consists of a root node, branches and terminal nodes or leaves. The general idea of the algorithm is to split the dataset into subgroups constantly in order to have as much homogeneity as possible in each set. The division of the data is measured according to impurity measures such as Gini and entropy. The feature which contributes the largest Gini gain (most information gain) will be assigned to the root node of the tree. On the other hand, Ensemble learning, which belongs again to supervised learning methods, combines multiple machine learning algorithms to achieve better prediction performance. By combining different prediction algorithms, which might have low accuracy, ensemble learning can construct a "strong learner" and highly improve the overall performance of the model.

The synopsis of the procedure applied to *Dataset1* and *Dataset2* is as follows. Firstly, a fully grown Decision Tree is constructed, which is then pruned by using the optimal complexity parameter through 10-fold cross-validation. Random Forests are then constructed and investigated in terms of number of trees, number of variables per split and complexity of the trees Followingly, the most known schemes of Ensemble learning are investigated for both Parkinson speech datasets. These are Bootstrap Aggregating (Bagging), Adaptive Boosting (AdaBoost), Gradient Boosting and Extreme Gradient Boosting (XGBoost).

III. EXPERIMENTAL EVALUATION ON *DATASET1*

A. Exploratory analysis and data pre-proccesing

The goal in exploratory data analysis is to get an understanding of the Parkinson's speech data set. The features of the data set are the subject ID, various speech recordings and the classes of PD diagnosis (1 is for positive PD and 0 is for negative PD). The data set is balanced as there are 520 observations for each class. This is visualized in Fig.2.

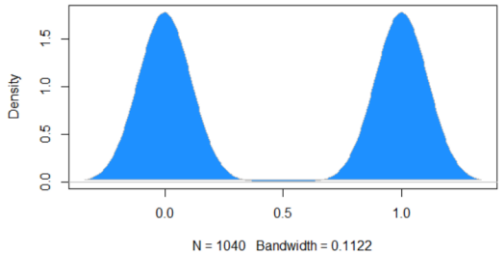


Fig.2. Distribution of the classes for *Dataset1*

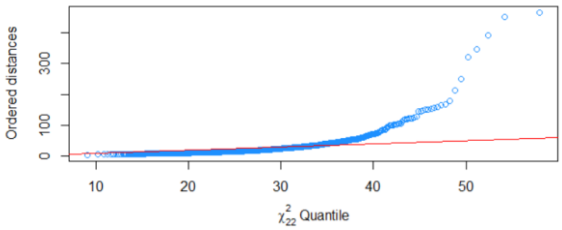


Fig.3. Multivariate normality test for *Dataset1*

Following that, the data set is tested for multivariate normality. As it can be seen from Fig.3, the data set has many outliers and does not fall to the straight line. In terms of correlation the heatmap correlation matrix is illustrated in Fig.4. Each square in the heatmap indicates the correlation between the variables. Values which are closer to zero indicate that there is not much correlation between the variables, whereas 1 is the perfect linear correlation. Generally, the higher the correlation, the darker the color is. Due to the many features of the data set, Principal Component Analysis (PCA) can also be used to better

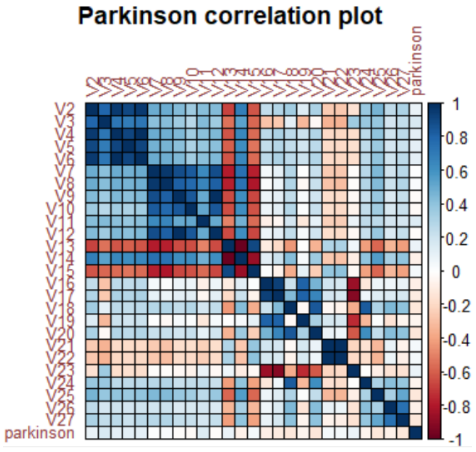


Fig.4. Correlation heat map visualisation for *Dataset1*

handle the data. After the PCA analysis, it was found that

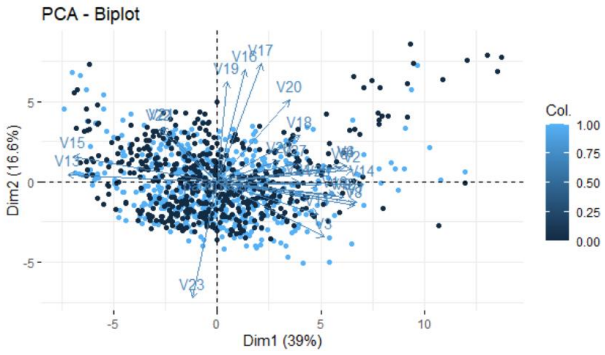


Fig.5. Principal Component Analysis (PCA) visaulization for *Dataset1*

the first 8 components are enough to explain 90% of the variance of the data. Fig. 5 represents an example of the first two components, which explain 55.6% of the data's variance. Also, from the PCA visualization, it is obvious

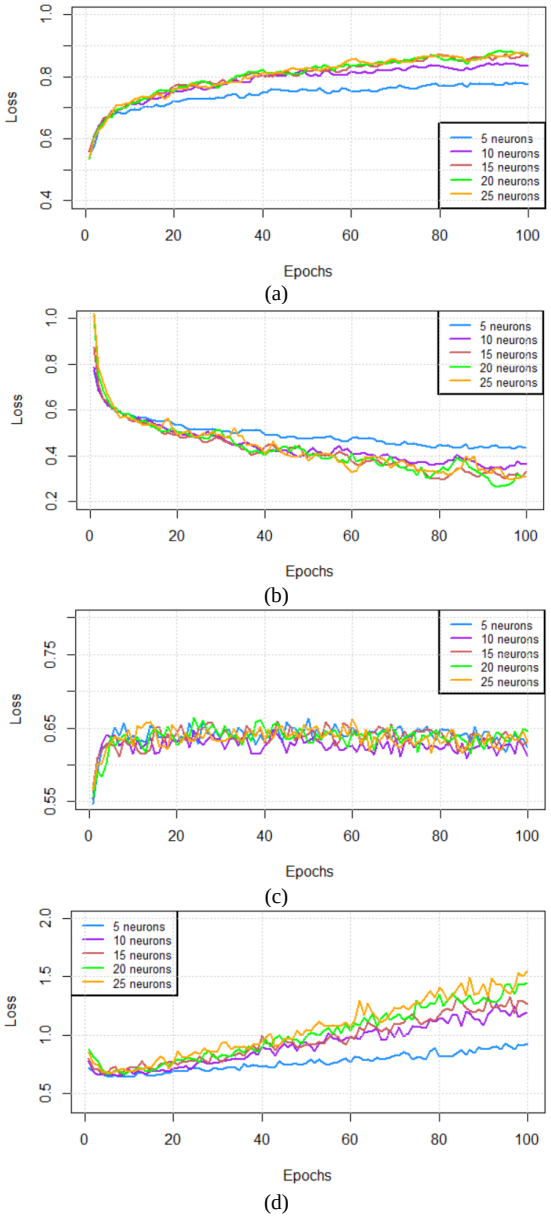


Fig.6. Effect of number of neurons on ADAM's training accuracy (a), trainning loss(b), test accuracy(c) and test loss(d).

that the different classes are clustered together. That could increase the classification difficulty, resulting into poor test accuracy.

The last steps before moving to the neural network specifications is to scale the data set and to apply one-hot-encoding. By scaling the data, the mean and standard deviation is calculated for each vector and then each element of the vector is "scaled" according to these findings, by subtracting the mean and dividing with the standard deviation. Scaling is very important in many data analysis techniques and machine learning models, as it makes it easier to compare components with different units and all the variables contribute equally to the analysis. With one-hot-encoding the variables that contain categories are transformed to binary vectors containing 0s and 1s.

B. Artificial Neural Network Approach

The model for the neural network architecture was created layer-by-layer (sequential model). As an input layer, all the 26 features from the Parkinson's speech data were used. These features after combined with weights, which are drawn randomly from a Uniform distribution, pass to the following layer. The objective is to minimize the binary crossentropy loss function, through the SGD and ADAM optimizers. Binary crossentropy function, as the name suggests, is used as a loss function in binary classification problems. The binary crossentropy loss function works with logarithmic values, hence it needs to be in the range between 0 and 1. To achieve this the use of the softmax activation function in the last layer of the neural network is essential. The softmax activation function is a generalization of the logistic function. By passing an input of real number to the softmax function, the result is normalized between (0,1) and all the components sum to 1, so they can be interpreted as probabilities. For the hidden

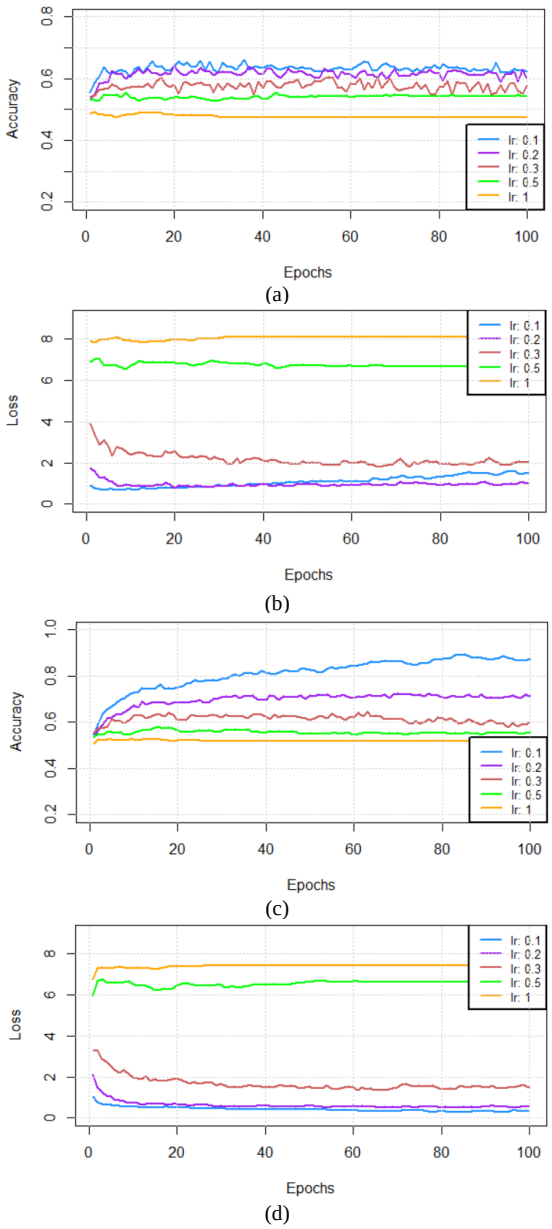


Fig.7. Effect of differnet learning rates on ADAM's training accuracy (a), trainning loss(b), test accuracy(c) and test loss(d).

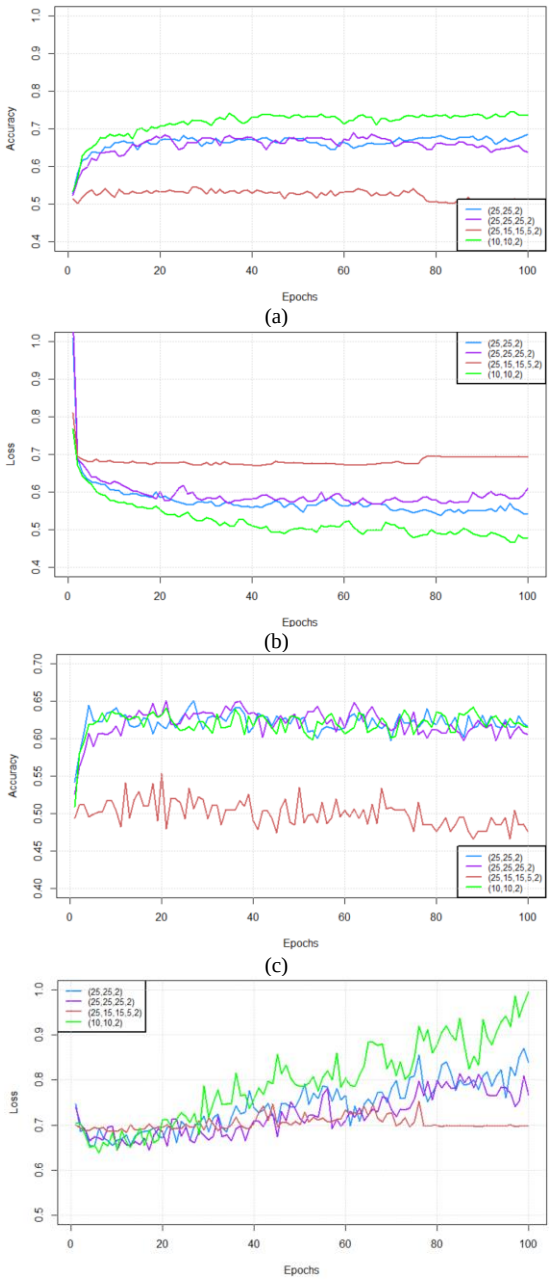


Fig.8. Effect of number of the hidden layers on ADAM's training accuracy (a), trainning loss(b), test accuracy(c) and test loss(d)

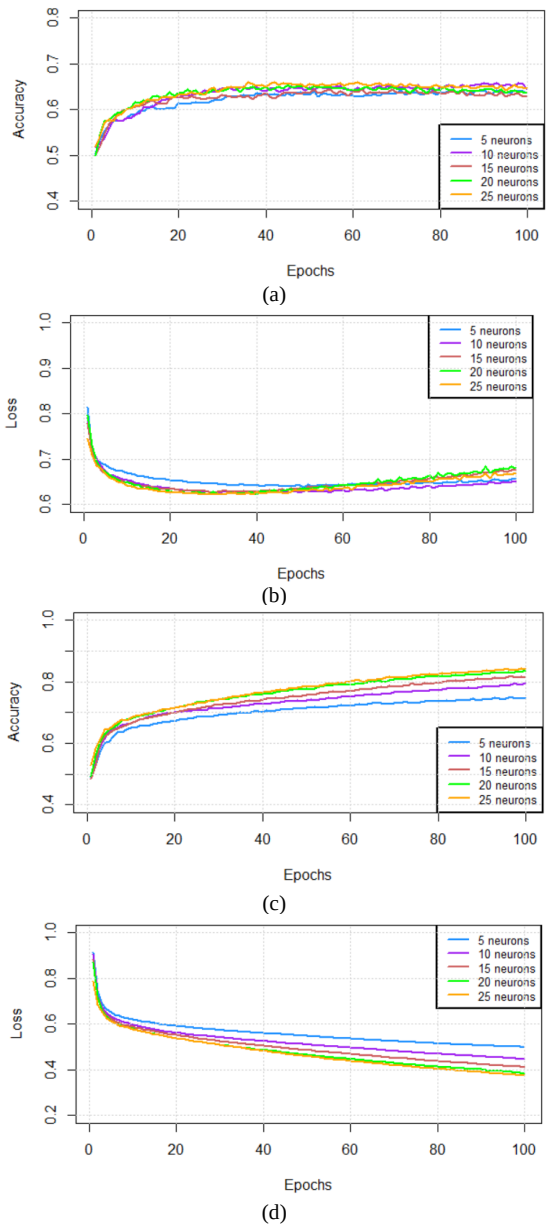


Fig.9. Effect of number of neurons on SGD's training accuracy (a), trainning loss(b), test accuracy(c) and test loss(d)

layers, the activation function that was used is the Rectified Linear Unit (ReLU), which is a linear function for positive values, outputting either its input if it is positive or zero. The optimization algorithm is iterative, which means that the process occurs in multiple steps and in during each step the performance is improved. For the all the models, 100 epochs were used with a validation set of 0.2. That means, 20% of the data set will not be used to train the model and will be provided to evaluate the loss and the accuracy at the end of each epoch. As a batch size, 50 samples were used to process before the model is updated. Finally, before each of the 100 epochs the training data were shuffled. To handle the uncertainty of the results, the algorithm was run 10 times to gather the average performance measures of the neural network. After the multiple experiments, summary statistics are reported based on the final model configuration.

1) Adam Optimizer Investigation

ADAM optimizer is the name for adaptive moment estimation, and it mostly used to train deep neural networks. The algorithm of ADAM is a combination of AdaGrad and RMSProp and the advantage is that it exploits the first and second moments to compute different learning rates for different parameters [20]. ADAM works well for multilayer networks and high dimensional data, as it has less computational cost than other methods. For our problem, ADAM was investigated for different number of neurons for per layer, several learning rates and for 1,2,3 and 4 hidden layers. Ten experimental runs were implemented for each investigated architecture, the mean value and 95% confidence intervals were reported in Table II for each

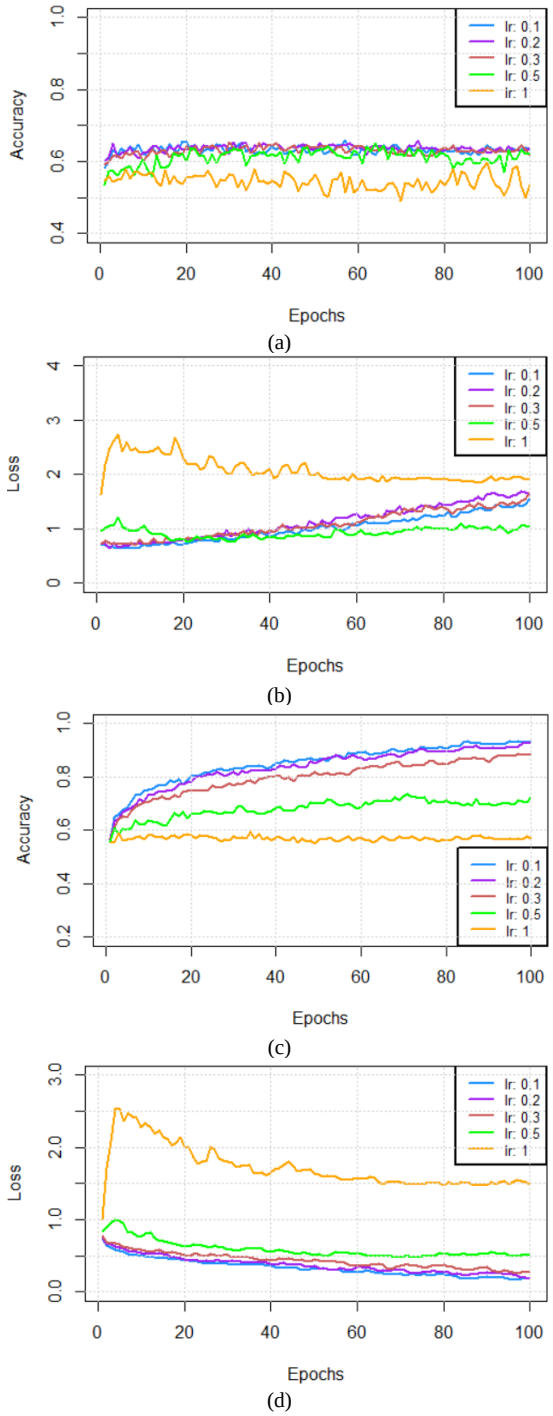


Fig.10. Effect of the learning rate on SGD's training accuracy (a), trainning loss(b), test accuracy(c) and test loss(d)

hyperparameter. In terms of the number of neurons, after experimenting with 5,10,15,20 and 25 the maximum accuracy was achieved with 25 neurons. The mean training accuracy was 81%, although the highest classification accuracy reached 64%. On the other hand, the loss was maximized as more neurons were added to the layer. As it can be seen in Fig.6 (c), test accuracy remains stable for different number of neurons, however the test loss decreases in the case of 5 neurons. The next experiment involved 25 neurons for a single layer model where the learning rate was investigated. As it can be observed from the results in Table II suggest, the best learning rate was 0.1. The neural network achieved the highest accuracy of 63% . As depicted in Fig.7 as the learning rate became larger, the value of the loss function was increased while the accuracy was decreased to almost 50% which means that the classification is random. Finally, for different hidden layers 1 to 4 and different number of neurons per layer as it can be seen in Fig.8, the best outcomes were obtained when using 1 and 2 hidden layers. A classification accuracy of 64% was achieved with 1 hidden layer and 25 neurons while the second-best accuracy was 61% for 2 hidden layers and 25 neurons per layer. As the number of hidden layers was increased data the train accuracy dropped below 50%, resulting into a test accuracy of 50%. In conclusion, the "best" performed model in terms of ADAM optimizer was the model with 1 hidden layer which included 25 neurons and learning rate 0.1.

2) SGD Optimizer Investigation

Stochastic gradient descent (SGD) is an optimizing method to smoothly minimize an objective function. The difference with the classic gradient descent optimization algorithm is that to update the set of parameters, only a random set of the training data is used in each iteration [21]. SGD converges faster the gradient descent, which is helpful in reducing the computational burden especially in high dimensional data sets. A disadvantage of the SGD is that sometimes convergence to the actual minimum is complicated, as it can stay to the local minimum instead of the global minimum. By introducing the idea of momentum, oscillation is reduced. During the first experiment the effect of number of neurons was investigated as shown in Fig.9. By referring to Table I, the best classification accuracy of 64% was achieved with 25 neurons. As for the learning rate, again the value of 0.1 produces the best results with accuracy of 64%. As shown in Fig.10 similar results were obtained for learning rates between 0.1 to 0.5 in terms of accuracy and loss. On the other hand, for learning rate equal to 1, the loss is around 2.7 and the classification performance is less than 60%. With 25 neurons and learning rate 0.1, the momentum rate was assessed. After evaluation of the results, 0.9 was chosen as the optimum momentum giving a test accuracy of 66%. At last, more hidden layers produce worst results and are overfitting the PD data. As a result, the best test accuracy of 65% was found with 1 hidden layer and 25 neurons. The concluded model for the SGD optimizer is a neural network with 25 neurons in one hidden layer, learning rate 0.1 and momentum rate 0.9.

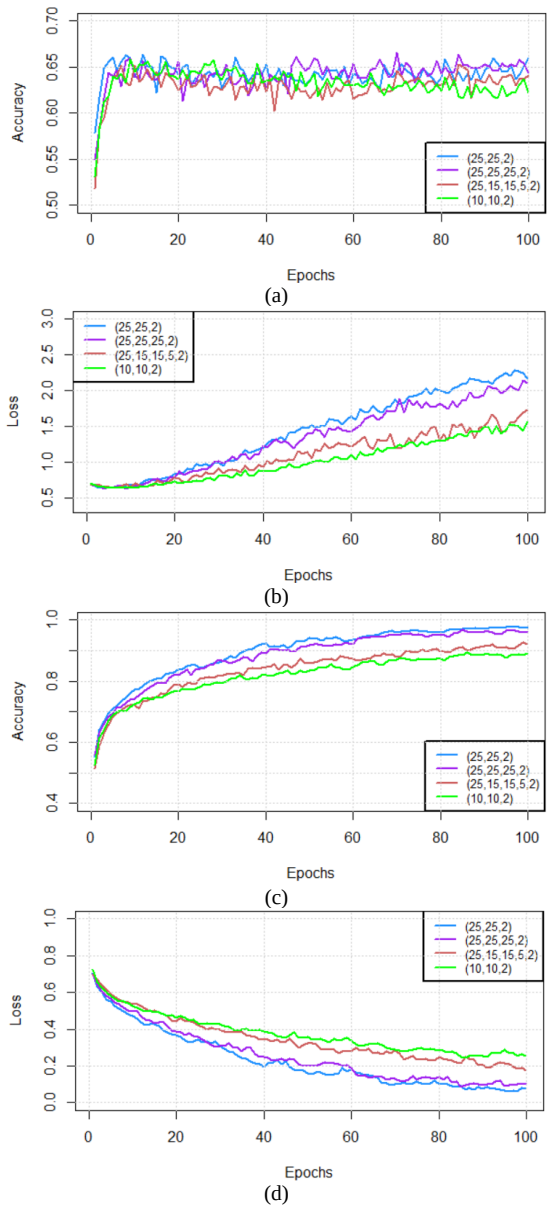


Fig.11. Effect of number of the hidden layers SGD's on training accuracy (a), trainning loss(b), test accuracy(c) and test loss(d)

TABLE I. SGD PERFORMANCE INVESTIGATION FOR DIFFERENT NEURAL NETWORK PARAMETERS FOR N=10 EXPERIMETAL RUNS ON DATASET1

Number of Neurons	Train				Test			
	Loss		Accuracy		Loss		Accuracy	
	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI
5	0.5294617	(0.52,0.54)	0.7264423	(0.72,0.74)	0.7045039	(0.67,0.72)	0.6072115	(0.59,0.62)
10	0.5054703	(0.49,0.52)	0.7493590	(0.73,0.76)	0.7115661	(0.68,0.73)	0.6149038	(0.60,0.62)
15	0.4832278	(0.47,0.49)	0.7780449	(0.76,0.79)	0.7449631	(0.72,0.76)	0.6173077	(0.60,0.62)

20	0.4665043	(0.45,0.47)	0.9074519	(0.88,0.93)	0.7370262	(0.7,0.76)	0.6254808	(0.61,0.63)	
25	**0.4554861**	**(0.44,0.46)**	**0.7977564**	**(0.78,0.80)**	**0.7407682**	**(0.71,0.76)**	**0.6293269**	**(0.62,0.64)**	
0.1	**0.5984530**	**(0.45,0.74)**	**0.8458333**	**(0.82,0.87)**	**1.775965**	**(1.57,1.97)**	**0.6396635**	**(0.61,0.64)**	**Learning Rate**
0.2	0.6958352	(0.6,0.78)	0.8325320	(0.80,0.86)	2.013769	(1.86,2.16)	0.6264423	(0.61,0.64)	
0.3	0.6493984	(0.60,0.79)	0.8105769	(0.79,0.83)	1.746525	(1.64,1.84)	0.6211538	(0.60,0.62)	
0.5	0.5860088	(0.60,0.70)	0.7108974	(0.69,0.72)	1.110327	(1.01,1.20)	0.6105769	(0.59,0.62)	
1	1.5328770	(0.95,2.11)	0.5458333	(0.52,0.56)	2.767513	(1.20,2.77)	0.5355769	(0.52,0.55)	
0.01	0.4702984	(0.57,0.58)	0.7855769	(0.77,0.79)	0.7377985	(0.71,0.75)	0.6250000	(0.61,0.63)	**Momentum Rate**
0.02	0.4750522	(0.56,0.58)	0.7815705	(0.77,0.78)	0.7621442	(0.73,0.79)	0.6189904	(0.61,0.62)	
0.04	0.4630520	(0.57,0.56)	0.7818910	(0.77,0.79)	0.7319940	(0.71,0.75)	0.6239231	(0.61,0.63)	
0.07	0.4816285	(0.56,0.57)	0.7774038	(0.76,0.78)	0.7611058	(0.72,0.80)	0.6108173	(0.59,0.62)	
0.1	0.4620094	(0.45,0.47)	0.7927885	(0.78,0.80)	0.7477439	(0.72,0.76)	0.6281250	(0.62,0.64)	
0.9	**0.5626384**	**(0.55,0.57)**	**0.7757885**	**(0.78,0.80)**	**0.7376639**	**(0.72,0.76)**	**0.6581340**	**(0.63,0.66)**	
1	**0.4554861**	**(0.44,0.46)**	**0.7977564**	**(0.78,0.80)**	**0.7407682**	**(0.71,0.76)**	**0.6293269**	**(0.62,0.64)**	**Number of Hidden Layers**
2	0.6916153	(0.64,0.73)	0.8846154	(0.87,0.89)	2.411596	(2.28,2.53)	0.6507212	(0.63,0.66)	
3	0.7046402	(0.62,0.78)	0.8642628	(0.85,0.87)	2.357691	(2.12,2.58)	0.6375000	(0.62,0.64)	
4	0.6307591	(0.54,0.71)	0.8471154	(0.83,0.85)	1.911961	(1.60,2.21)	0.6324519	(0.62,0.64)	

TABLE II. ADAM PERFORMANCE INVESTIGATION FOR DIFFERENT NEURAL NETWORK PARAMETERS FOR N=10 EXPERIMETAL RUNS ON DATASET1

	Train				**Test**				
	Loss		**Accuracy**		**Loss**		**Accuracy**		
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	
5	0.5779616	(0.54,0.60)	0.7277244	(0.69,0.75)	1.0494362	(0.95,1.14)	0.6036058	(0.59,0.62)	**Number of Neurons**
10	0.5857607	(0.54,0.62)	0.7775641	(0.76,0.79)	1.3287047	(1.22,1.43)	0.6137019	(0.60,0.62)	
15	0.5724933	(0.52,0.62)	0.8223520	(0.72,0.76)	1.5239148	(1.40,1.64)	0.6213942	(0.62,0.65)	
20	0.6112080	(0.57,0.64)	0.8176282	(0.80,0.82)	1.7797209	(1.50,1.78)	0.6312500	(0.60,0.64)	
25	**0.6602387**	**(0.57,0.74)**	**0.8118590**	**(0.79,0.82)**	**1.7149911**	**(1.51,1.91)**	**0.6209135**	**(0.63,0.64)**	
0.1	**0.6492901**	**(0.60,0.69)**	**0.8100962**	**(0.79,0.82)**	**1.722966**	**(1.57,1.80)**	**0.6204327**	**(0.60,0.63)**	**Learning Rate**
0.2	0.6723217	(0.60,0.74)	0.6910256	(0.66,0.71)	1.190320	(1.05,1.32)	0.5954327	(0.57,0.61)	
0.3	1.6273920	(1.13,2.12)	0.6004808	(0.57,0.62)	2.016304	(1.46,2.01)	0.5704327	(0.55,0.58)	
0.5	6.6311290	(5.83,7.43)	0.5504808	(0.51,0.58)	6.813512	(6.20,7.43)	0.5360577	(0.50,0.56)	
1	7.6239525	(7.42,7.82)	0.5041667	(0.49,0.51)	7.711987	(7.49,7.94)	0.4983173	(0.48,0.51)	
1	**0.6602387**	**(0.57,0.74)**	**0.8118590**	**(0.79,0.82)**	**1.7149911**	**(1.51,1.91)**	**0.6209135**	**(0.63,0.64)**	**Number of Hidden**
2	0.621628	(0.58,0.65)	0.6673077	(0.62,0.70)	0.8813420	(0.79,0.97)	0.5824519	(0.56,0.60)	
3	0.6444787	(0.61,0.67)	0.6307692	(0.57,0.68)	0.7765054	(0.65,0.89)	0.5615385	(0.53,0.59)	
4	0.6931598	(0.69,0.70)	0.5027244	(0.48,0.51)	0.6944136	(0.69,0.70)	0.5091346	(0.50,0.51)	

TABLE III. COMPARIOSN BETWEEN SGD AND ADAM FOR ON DATASET1

`	**Accuracy**	**95% CI**	**Sensitivity**	**Specificity**	**F1-score**	**AUC**
SGD	0.65	(0.60,0.69)	0.5634	0.7438	0.6233	0.6536
ADAM	0.62	(0.57,0.67)	0.4554	0.8030	0.5543	0.6292

3) *Comparison between ADAM and SGD*

Based on the hyperparameters investigations from the previous experiments, the best ANN architecture for both optimizers were found to be 25 neurons in one hidden layer, learning rate 0.1 and momentum rate 0.9. To evaluate and compare the performance of ADAM versus SGD, a classification performance comparison is given on Table III. More specifically, SGD achieved an average better accuracy of 65% with a maximum of 69% as compared to an average of 62% with a maximum of 67% in the case of ADAM. Accuracy and loss visualizations for the best performing models are included in Fig.12 and Fig.13. SGD presents a higher validation loss compared to ADAM while it outperforms ADAM in terms of training accuracy.

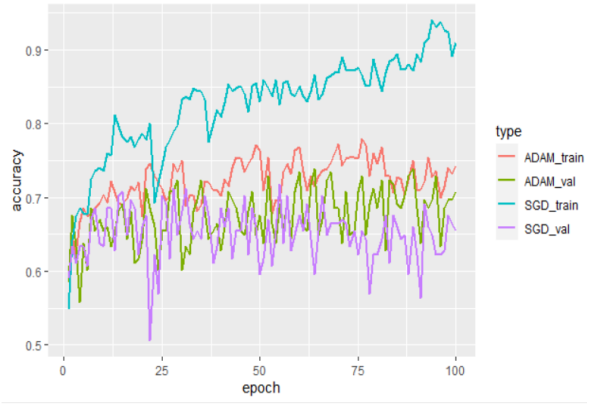


Fig.12. Accuracy comparison between SGD and ADAM for *Dataset1*

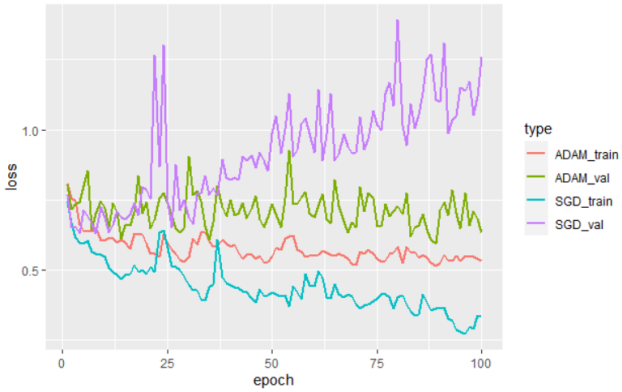


Fig.13. Loss comparison between SGD and ADAM *Dataset1*

C. *Decision Trees and Ensemble Learning Approach*

1) *Full Tree Investigation*

A fully grown tree investigation was applied to *Dataset1* with an overall train accuracy of 83% as shown in Table IV. On the other hand, the accuracy for the test set of *Dataset1* was 63% which could be an indicator of the model overfitting the data. A plot of the fully grown tree for PD *Dataset1* is illustrated in Fig.14. As it can be seen in Fig.14, the first node, which is the root of the tree, is split in two decisions. When the feature Jitter. Rap is greater than or equal to -0.15, the cases fall to the second node which is Standard deviation. If Jitter.rap is less than -0.15, then there

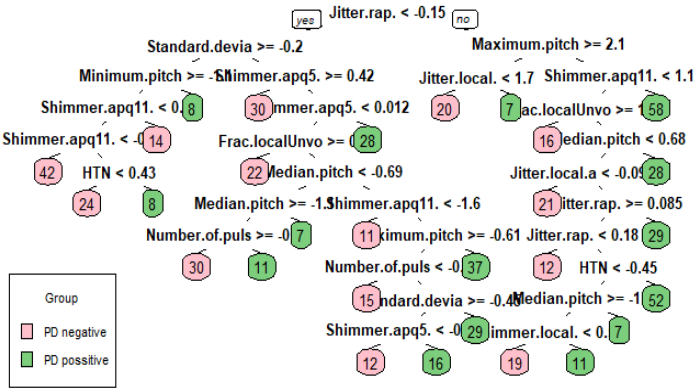


Fig.14. A fully grown tree of *Dataset1*

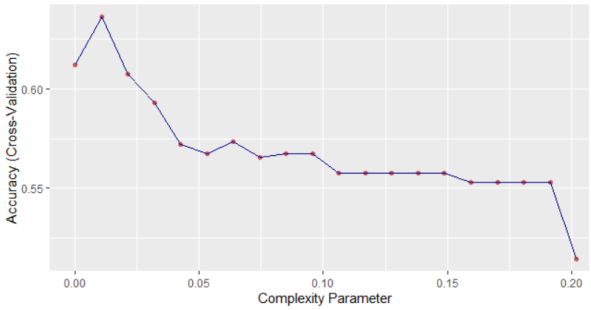


Fig.15. Accuracy against complexity parameter for *Dataset1*

exists another node which is Maximum pitch. By following this pattern, some cases fall into the PD positive group and some to the PD negative group.

2) *Prunned Tree Investigation*

After growing a large and complex tree, an efficient way to improve the performance is pruning. By pruning the decision tree, the cost of the complexity is minimized. After investigating multiple complexity parameters through cross validation, the one that provides the optimal accuracy and lowest error, is used to provide the final model. Fig.15 illustrates how the value of the accuracy of the models change in terms of the complexity parameter. As the complexity parameter becomes larger, the tree is deeper and more complex.

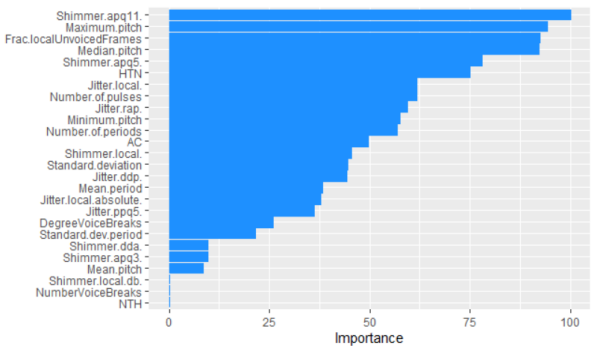


Fig.16. Feature importance of the Pruned Tree for *Dataset1*

For a complexity parameter close to zero (cp=0.011) the model has the maximum accuracy. The results obtained from this technique were better than the full tree in terms of accuracy as observed in Table IV. A classification accuracy of 79% was achieved for the training set and 64% for the validation set. In order to measure the importance of each feature for the classification task, a plot of the 26 features of the dataset is provided in Fig.16. The importance is measured in terms of the minimization of the loss function. In Fig.16 the most important variable is given a value of 100, and the rest of the variables are arranged based on their contribution in the loss reduction. Shimmer (apq11), Maximum pitch, Fraction of locally unvoiced frames and Median pitch seem to be the most important features. The uninformative features are not used in the decision tree.

3) Bagging

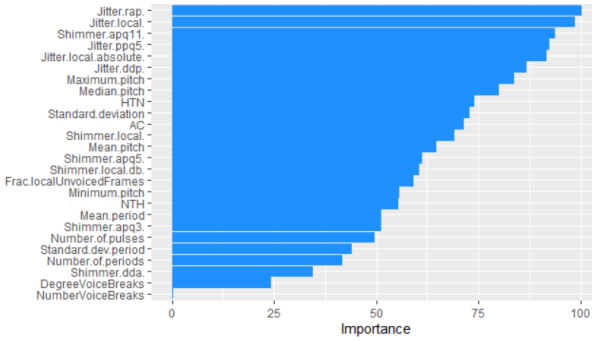
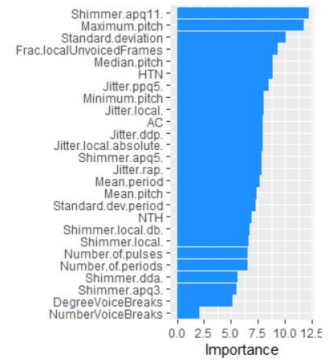


Fig.17. Feature importance of the Bagging algorithm for *Dataset1*

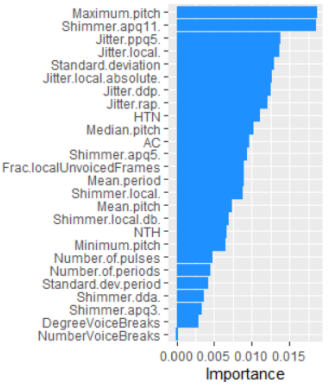
Bagging is a method using bootstrapping techniques to produce ensemble predictions. Bootstrap Aggregating is widely used in machine learning to improve the outcomes of the prediction. During the training procedure, the data are split in subsets, which are sampled uniformly with replacement. Bagging stabilizes the bias-variance-trade-off by minimizing the variance and keeping bias constant. By applying bagging to *Dataset1* for 300 trees and performing a 10-fold cross-validation (CV), an accuracy of 100% and 70% is achieved in the training data and test set respectively as illustrated in Table IV. The variable importance plot in Fig.17 provides the top 26 features. Shimmer (apq11) is still in the top 5 features, although the most important one is Jitter (rap). The results from the Bagging algorithms are more difficult to interpret since the method averages the results from all the trees. Hence, the influence of each feature in the final output averages across all models since the trees are not independent of each other.

4) Random Forest

Random Forests (RF) are considered an Ensemble Learning method able to perform classification and regression tasks. They use multiple trees as decision makers, which are uncorrelated and that helps in preventing overfitting the data. RF are basically a mix of decision trees and bagging. Each tree is built on the bootstrapped samples of the training set and then the decisions are aggregated with bagging across all the trees. The most notable difference from Bagging is that RF introduce randomness during the process of adding trees, which contributes to de-correlation [22]. To tune the hyperparameters of the Random forests, grid search was applied. The hyperparameters that were tuned are the number of trees, the number of variables to use



(a)



(b)

Fig.18. Feature importance in terms of impurity (a) and permutation (b) for *Dataset1*

at each split and the complexity parameter for the trees. The number of trees and the features to consider at the splits are the parameters that are the most critical and have the greatest influence in the accuracy. For *Dataset1*, after tuning through grid search, the overall performance was improved with 2000 trees. Table IV shows that, during training the accuracy reached 100% and during testing 69%. Fig.18 contains the results of the impact of the features for the impurity (a) and permutation (b) of the RF. The two options do not have identical results, however in the top of the illustration there are similar variables. In this case, Shimmer (apq11), Maximum pitch and Standard deviation are the most influential. In contrast, the Number of voice breaks and the Degree of voice breaks seem to have no influence.

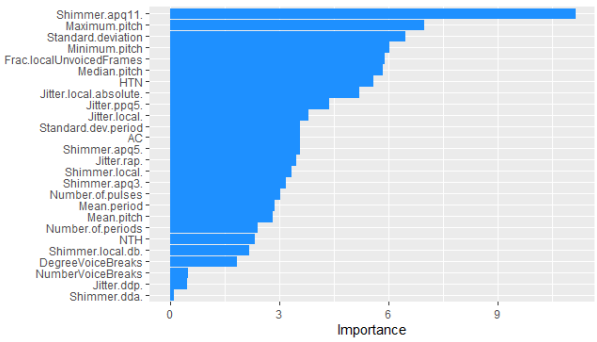


Fig.19. Feature importance for the Gradient Boosting algorithm for *Dataset1*

5) Boosting

Boosting is another set of techniques which belong to Ensemble learning. In general they transform weak classifiers to strong ones by combining them in an overall ensemble. Boosting is mostly effective when applied in high biased models. Each tree that is grown sequentially, provides a better prediction, by learning from the errors of the previous tree. The Boosting methods that are applied in *Dataset1* are Gradient Boosting, AdaBoost and XGBoost. Gradient boosting is a branch of the gradient descent algorithm, which focuses on finding the optimal solutions to problems, through iterative steps. The main hyperparameters worth to investigate in the Gradient Boosting algorithm are the learning rate and the number of trees in the sequence. During tuning, different learning rates were investigated in the values of (0.3, 0.1, 0.05, 0.01, 0.005). For learning rate equal to 0.05, which gave the best results, the number of trees was determined and found that 236 trees produced the optimal results. The previous steps included a 10-fold CV to obtain the best estimates. The results are presented in Table IV where an accuracy of 92% was obtained through training while 66.8% was achieved to unseen data. The ROC curve for the true positive and false positive rates can be seen in Fig.21(a).In the case of feature importance for Gradient Boosting, similar results with the previous techniques were obtained. The features that mostly contributed to the dataset were Shimmer (apq11), Maximum pitch and Standard deviation. A summary of the results is illustrated in Fig.19

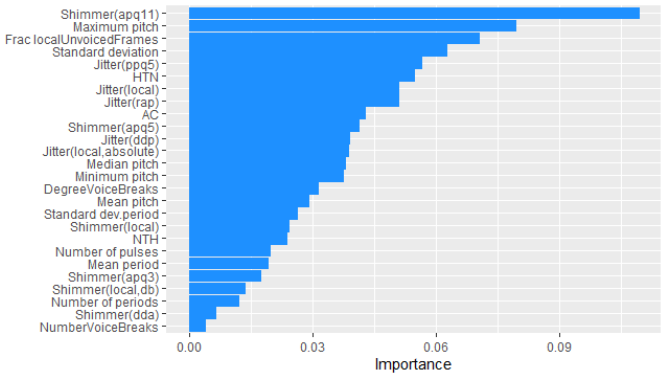


Fig.20. Feature importance for XGBoost for *Dataset1*

Extreme Gradient Boosting (XGBoost) follows the same idea as Gradient Boosting but is optimized to be more efficient. The advantages over Gradient Boosting are that it contains a regularization parameter that minimizes overfitting the data, early stopping techniques that provide an additional assessment of when to stop growing trees if no improvement is offered and flexibility across different user

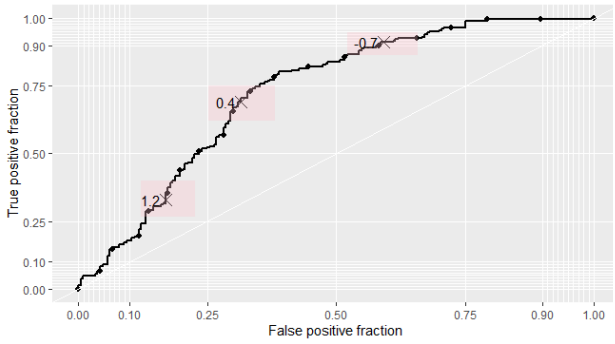


Fig.21.ROC curve for the Gradient Boosting (a), XGBoost (b), Adaboost(c) algorithm for *Dataset1*

defined loss functions. Same processes were applied in exploring the optimal values for different hyperparameters. The additional parameters that were tuned are the regularization and the early stopping. The results that were obtained were 79% and 64.66% classification accuracy for the train and test set, respectively as depicted in Table IV. XGBoost reduced the overfitting that occurred during the training phase as compared to Gradient Boosting. Fig.21(b) illustrates the resulted ROC curve. The contribution of the 26 features is presented in Fig.20. and is similar with the results of the previous algorithms.

TABLE IV. PERFORMANCE INVESTIGATION OF DECISION TREES AND ENSEMBLE LEARNING ON DATASET1

	Train Accuracy		Test Accuracy	
	Mean	**95%CI**	**Mean**	**95%CI**
Full Tree	0.8317	(0.8, 0.8603)	0.6322	(0.5839, 0.6787)
Pruned Tree	0.7949	(0.761, 0.8259)	0.6418	(0.5937, 0.688)
Bagging	1	(0.9941, 1)	0.7065	(0.653, 0.7432)
Random Forest	1	(0.9941, 1)	0.6851	(0.6381, 0.7295)
AdaBoost	0.8397	(0.8086, 0.8677)	0.6562	(0.6084, 0.7018)
Gradient Boosting	0.9215	(0.8975, 0.9413)	0.6683	(0.6207, 0.7134)
XGBoost	0.7917	(0.7577, 0.8229)	0.6466	(0.5986, 0.6926)

The last Boosting method that was applied is Adaboost. Adaboost utilizes the errors of the previous classifiers sequentially during the learning process. The main difference with RF is that there is a dependency among the models and the misclassifications are weighted higher in order to improve the accuracy. The main drawback of Adaboost is that it is highly sensitive to outliers and noisy data. After applying Adaboost to the problem, the mean accuracy during training was 83.97% and during testing 65.62%. A ROC curve can be found in Fig.21(c) and the accuracy performance can be seen in Table IV.

IV. Experimental Evaluation using Dataset 2

A. *Exploratory analysis and data pre-proccesing*

The given dataset *Dataset2* was balanced, meaning that there were 40 patients diagnosed with Parkinson and 40 healthy as seen in Fig.22. The dataset was tested for multivariate normality. As it can be seen from Fig.23, the are many outliers and the tails of the distribution are very heavy. The heatmap correlation matrix is illustrated in Fig.24. There seems to be extremely high correlation for some sets of variables, which can affect the classification performance. PCA can also be used to reduce the high dimensionality due to the 45 features of *Dataset2*. It was found that the first 12 components are enough to explain 90% of the variance of the data. Fig. 25 represents a visualization of the first two principal components, which explain 68.5% of the data's variance. From the PCA plot, it seems that the two classes of *Dataset2* are more separated in comparison to *Dataset1* which could lead to higher classification accuracy.

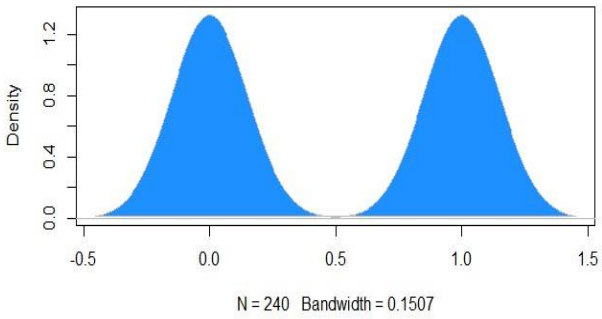


Fig.22. Distribution of the classes for *Dataset2*

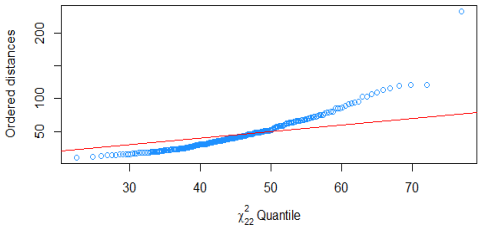


Fig.23. Multivariate normality for *Dataset2*

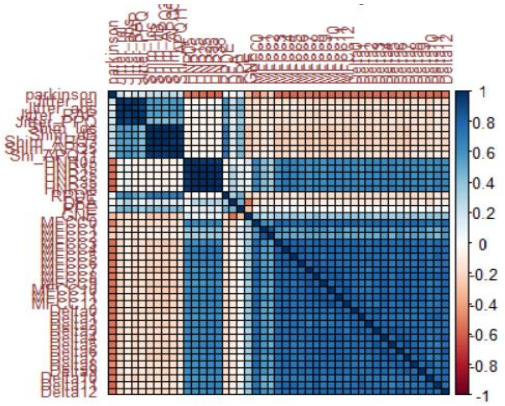


Fig.24. Heatmap correlation plot for *Dataset2*

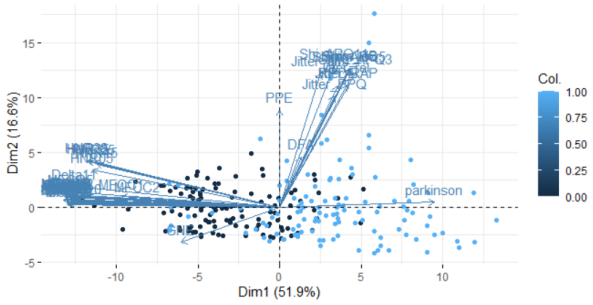


Fig.25. PCA visaulization for *Dataset2*

B. *Articficial Neural Network Approach*

1) *Comparison between ADAM and SGD*

For the purpose of applying ANNs to *Dataset2*, scaling and one-hot-encoding was applied. The architecture of the ANN used is 25 neurons in one hidden layer, learning rate 0.1 and momentum rate 0.9 which was shown to be the best performing one as outlined in Section III for *Dataset1*.

TABLE V. Compariosn between SGD and ADAM Dataset2

`	Accuracy	95% CI	Sensitivity	Specificity	F1-score	AUC
SGD	0.7812	(0.6853, 0.8592)	0.8085	0.7551	0.7835052	0.7818
ADAM	0.7708	(0.6739, 0.8505)	0.8298	0.7143	0.78	0.772

As it is illustrated in Fig.26 and Fig.27, the accuracy both for ADAM and SGD present similar results. During training the classification accuracy reached its maximum value of 100% after some iterations and during the validation time the accuracy was around 78% for both optimizers. In terms of the loss, SGD performed better and had more stability, as it can be seen in Fig.27.Table V summarizes the classification performance of ADAM and SGD on *Dataset2*. SGD performs better than ADAM with a maximum classification accuracy of 85.92%.

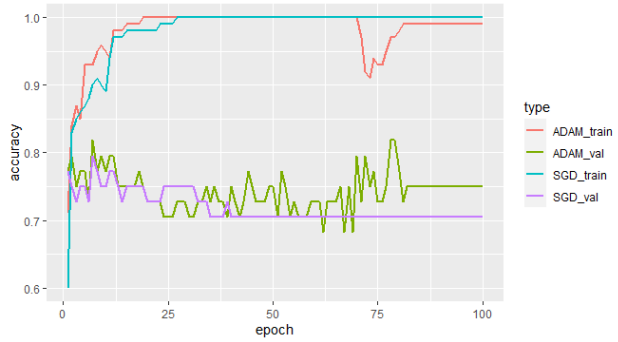


Fig.26. Accuracy comparison between ADAM and SGD *Dataset2*

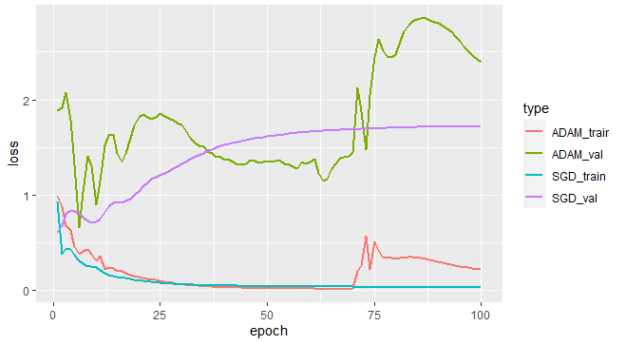


Fig.27. Loss comparison between ADAM and SGD *Dataset2*

C. Decision Trees and Ensemble Learning Approach

1) Full Tree Investigation

During full tree investigation, the model overfitted the data as it gave 100% accuracy during training but only 71.88% accuracy during validating to unseen data. Table VI summarizes the training and test accuracy while Fig.28 illustrates the fully grown tree before pruning.

2) Prunned Tree Investigation

The investigation through cross validation for the value of the complexity parameter to prune the tree gave a result of 0.011 as it is visualized in Fig.29. After pruning the decision tree, the accuracy is minimized to 87.5% for the train set and raised to 75% for the test set. The signs of overfitting the data are reduced after pruning.

3) Bagging

By applying Bagging on *Dataset2* for 300 trees and 44 predictors, the classification precision was 100% for the train data and 79% for the test data as observed in Table VI. A variable importance plot was also produced in order to access the features which are mostly contributing to the analysis and it is illustrated in Fig.30. Mel frequency cepstral coefficient-based spectral measures (MECC) of

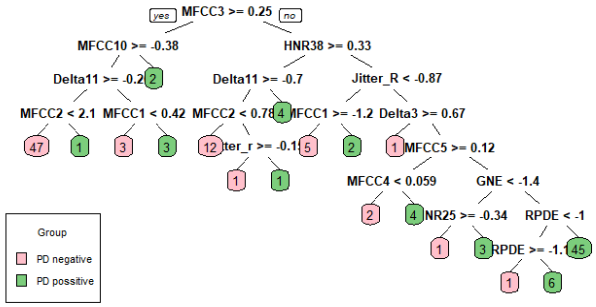


Fig.28. Fully grown tree for *Dataset2*

3,4,5,8 and 7 and some of their derivatives (Delta) overwhelm the top of the plot.

4) Random Forests

For *Dataset2*, after tuning the random forest, the overall performance was improved again for 2000 trees. As it can be seen in Table VI, Random forests had the same results with Bagging during training, although for the test data the classification accuracy improved to 81.25%. Similar results were obtained for the most contributing features. As seen in Fig.31, the feature importance in terms of impurity (a) and on permutation (b) is presented.

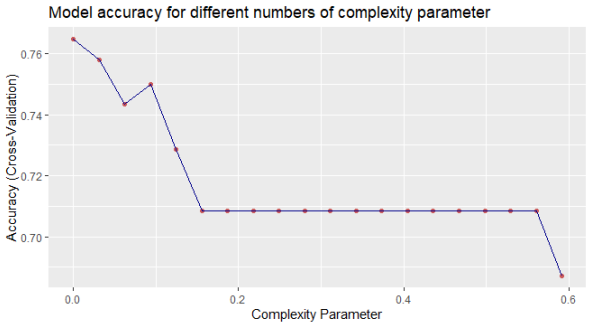


Fig.29. Accuracy against complexity parameter for *Dataset2*

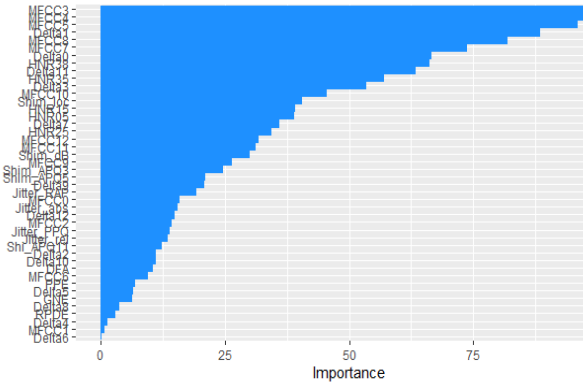
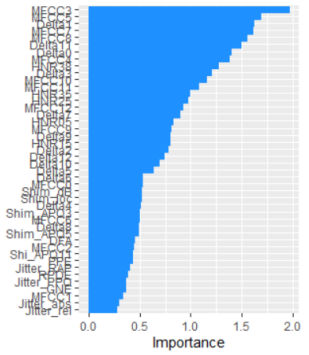


Fig.30. Feature importance of the Bagging algorithm for *Dataset2*

5) Boosting

In terms of Boosting the same methods and tunning strategy as *Dataset1* were applied. Optimal results for Gradient Boosting were achieved for 83 trees and learning rate equal to 0.05. The resulted classification accuracy was 97.92% for the training data and 81.25% for the testing data as set out in Table VI. The ROC curve is visualized in Fig.33(a). Feature importance, as shown in Fig.32 produced similar results with previous techniques, suggesting that MECC features are the most important variables and are the ones which mostly contribute to the classification problem.

When XGBoost was applied on *Dataset2*, the learning accuracy was around 87%. During the testing phase the mean classification prediction was 80% as shown in Table VI. Moreover, the ROC Curve of XGBoost can be seen in Fig.33(b). Finally, Adaboost method produced an 88.89% training and 82.29% testing accuracy as shown in Table VI.



(a)

(b)

Fig.31. Feature importance in terms of impurity (a) and permutation (b) for PD dataset 2

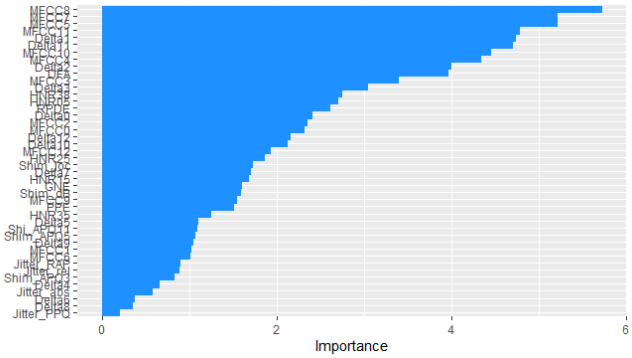
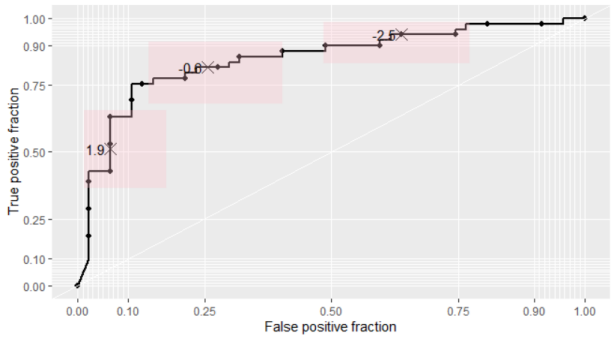
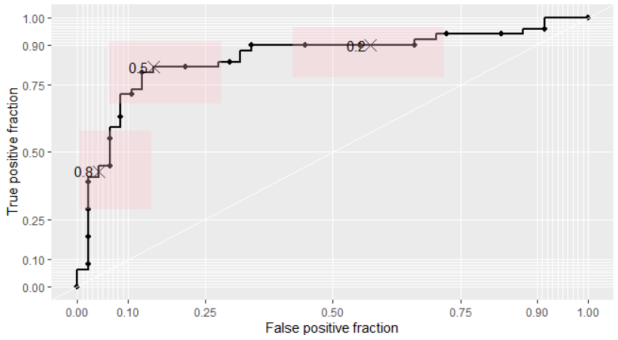


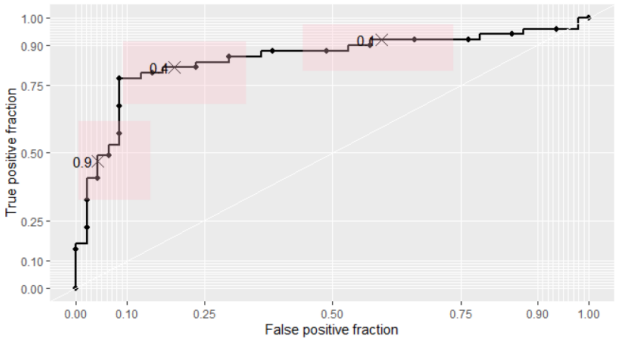
Fig.32. Freature importance for Gradient Boosting for dataset 2



(a)



(b)



(c)

Fig.33.ROC curve for the Gradient Boosting (a), XGBoost (b), Adaboost(c) algorithm for PD dataset 2

TABLEVI. PERFORMANCE INVESTIGATION OF DECISION TREES AND ENSEMBLE LEARNING ON DATASET2

	Train Accuracy		Test Accuracy	
	Mean	**95%CI**	**Mean**	**95%CI**
Full Tree	1	(0.9747, 1)	0.7188	(0.6178, 0.8058)
Pruned Tree	0.875	(0.8097, 0.9242)	0.75	(0.6512, 0.8328)
Bagging	1	(0.9747, 1)	0.7917	(0.6967, 0.8679)
Random Forest	1	(0.9747, 1)	0.8125	(0.72, 0.8849)
AdaBoost	0.8889	(0.8258, 0.9351)	0.8229	(0.7317, 0.8933)
Gradient Boosting	0.9792	(0.9403, 0.9957)	0.8125	(0.72, 0.8849)
XGBoost	0.875	(0.8097, 0.9242)	0.8021	(0.7083, 0.8764)

TABLE VII. CLASSIFICATION PERFORMANCE COMPARISON BETWEEN ARTIFICIAL NEURAL NETWORKS, DECISION TREES AND ENSEMBLE LEARNING ON DATASET1

`	**Accuracy**	**95% CI**	**Sensitivity**	**Specificity**	**F1-score**	**AUC**
SGD	0.65	(0.60,0.69)	0.5634	0.7438	0.6233	0.6536
ADAM	0.62	(0.57,0.67)	0.4554	0.8030	0.5543	0.6292
Full Tree	0.6322	(0.5839,0.6787)	0.5728	0.6946	0.6146096	0.6337
Pruned Tree	0.6418	(0.5937, 0.688)	0.5634	0.7241	0.6169666	0.6438
Bagging	**0.7065**	**(0.653, 0.7432)**	**0.5728**	**0.8325**	**0.6557377**	**0.7004**
Random Forest	0.6851	(0.6381,0.7295)	0.5634	0.8128	0.654561	0.6973
AdaBoost	0.6562	(0.6084,0.7018)	0.7059	0.6220	0.6266319	0.6585
Gradient Boosting	0.6683	(0.6207,0.7134)	0.7042	0.6305	0.6849315	0.6674
XGBoost	0.6466	(0.5986,0.6926)	0.5540	0.7438	0.6036745	0.6394

TABLE VIII. CLASSIFICATION PERFORMANCE COMPARISON BETWEEN ARTIFICIAL NEURAL NETWORKS, DECISION TREES AND ENSEMBLE LEARNING ON DATASET2

`	**Accuracy**	**95% CI**	**Sensitivity**	**Specificity**	**F1-score**	**AUC**
SGD	0.7812	(0.6853, 0.8592)	0.8085	0.7551	0.7835052	0.7818
ADAM	0.7708	(0.6739, 0.8505)	0.8298	0.7143	0.78	0.772
Full Tree	0.7188	(0.6178, 0.8058)	0.7021	0.7959	0.7333333	0.749
Pruned Tree	0.75	(0.6512,0.8328)	0.7021	0.7959	0.7333333	0.749
Bagging	0.7917	(0.6967,0.8679)	0.7872	0.7959	0.787234	0.7916
Random Forest	0.8125	(0.72, 0.8849)	0.8085	0.8163	0.8085106	0.8124
AdaBoost	**0.8229**	**(0.7317,0.8933)**	**0.8163**	**0.8511**	**0.8333333**	**0.8337**
Gradient Boosting	0.8125	(0.72, 0.8849)	0.8723	0.7551	0.82	0.8137
XGBoost	0.8021	(0.7083,0.8764)	0.7872	0.8163	0.787234	0.7916

The results of this method were very accurate despite the outliers identified in *Dataset2* during the exploratory analysis. The ROC curve for AdaBoost is illustrated in Fig.33(c).

V. RESULTS AND COMPARISON BETWEEN ANNS, DECISION TREES AND ENSEMBLE LEARNING

Tables VII and VIII, summarize the results obtained from ANNs, Decision Trees and Ensemble Learning methods for *Dataset1* and *Dataset2*.

A. DATASET1

For *Dataset1*, in terms of accuracy, Bagging produced the best results with mean classification accuracy of 70.65% that reached a maximum value of 74.32%. Specificity, which is the ability of the model to correctly classify the healthy subjects, was 83.25%. Sensitivity on the other hand stayed low with a value of 57.28%. That means, the model was better in classifying the healthy patients, rather than the PD positive ones. In general, Bagging and Ensemble Learning methods performed better than Neural Networks, Decision Trees and Random Forests.

B. DATASET2

For the second data set, AdaBoost performed overall better in terms of accuracy, sensitivity, and specificity with 82.29%, 81.63% and 85.11% respectively. On the other hand, Gradient Boosting produced the highest sensitivity during the investigation of 87.23%, which is important as it is the method which had the lowest error in identifying which patients belonged to the positive class. Regarding the neural networks, SGD and ADAM optimizers had similar results, with ADAM fluctuating more during the testing phase.

VI. CONCLUSION

This report summarizes the investigation of the performance of ADAM and SGD optimizers, Decision trees, Random Forests and Ensemble Learning methods for speech-based Parkinson disease diagnosis for two experimental datasets. The best ANN architecture was chosen based on the experimentation with different hyperparameters such as the number of neurons, number of hidden layers, learning and momentum rate. The best architecture was found to be 25 neurons in one hidden layer, learning rate 0.1 and momentum rate 0.9. The same architecture was applied to both Parkinson datasets, yielding accuracy of 69% and 85.92% for the first and second case, respectively. As for Decision Trees and ensemble learning, Bagging gave the best results for the first dataset with maximum accuracy 74.32%. At the second experimental case, AdaBoost obtained the optimal results with the highest accuracy of 89.33%. In general, all the methods yielded more accurate and steady results in the second experimental case. A reason for this could be the way the classes were clustered and the correlation of the features. As previously stated, the Parkinson classes were clustered together in the first data set, but in the second they were more separable. In summary, for both cases, Ensemble methods outperformed the other techniques. For future work, to achieve higher accuracy and reduce overfitting, feature importance as well as feature selection techniques such as PCA could be considered.

REFERENCES

[1] E. Ronken and G. J. M. v. Scharrenburg, "Parkinson's Disease", *Solvay Pharmaceuticals Conferences*. Amsterdam: IOS Press, 2002.

[2] J. Jankovic, "Parkinson's disease: clinical features and diagnosis," *J. Neurol. Neurosurgery Psychiatry*, vol. 79, no. 4, pp. 368-376, 2007.

[3] M. A. Little, P. E. McSharry, S. J. Roberts, D. A. E. Costello, and I. M. Moroz, "Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection," *BioMedical Engineering OnLine*, vol. 6, no. 1, p. 23, 2007/06/26 2007.

[4] B. E. Sakar *et al.*, "Collection and Analysis of a Parkinson Speech Dataset With Multiple Types of Sound Recordings," *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 4, pp. 828-834, 2013.

[5] A. B. New *et al.*, "The intrinsic resting state voice network in Parkinson's disease," *Human brain mapping*, vol. 36(5), no. 1097-0193 (Electronic), pp. 1951-62, 2015.

[6] S. Lahmiri, D. A. Dawson, and A. Shmuel, "Performance of machine learning methods in diagnosing Parkinson's disease based on dysphonia measures," *Biomedical Engineering Letters*, vol. 8, no. 1, pp. 29-39, 2018/02/01 2018.

[7] C. O. Sakar and O. Kursun, "Telediagnosis of Parkinson's Disease Using Measurements of Dysphonia," *Journal of Medical Systems*, vol. 34, no. 4, pp. 591-599, 2010/08/01 2010.

[8] S. Lahmiri, "Parkinson's disease detection based on dysphonia measurements," *Physica A: Statistical Mechanics and its Applications*, vol. 471, pp. 98-105, 2017/04/01/ 2017.

[9] S. M, M. Martin, and A. Tripathi, "ANN based Data Mining Analysis of the Parkinson's Disease," *International Journal of Computer Applications*, vol. 168, pp. 1-7, 06/15 2017.

[10] A. Yasar, I. Saritas, M. A. Sahman, and A. C. Cinar, "Classification of Parkinson disease data with artificial neural networks," *IOP Conference Series: Materials Science and Engineering*, vol. 675, p. 012031, 2019/11/15 2019.

[11] L. Berus, S. Klancnik, M. Brezocnik, and M. Ficko, "Classifying Parkinson's Disease Based on Acoustic Measures Using Artificial Neural Networks," *Sensors*, vol. 19, p. 16, 12/20 2018.

[12] F. Åström and R. Koker, "A parallel neural network approach to prediction of Parkinson's Disease," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12470-12474, 2011/09/15/ 2011.

[13] R. Das, "A comparison of multiple classification methods for diagnosis of Parkinson disease," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1568-1572, 2010/03/01/ 2010.

[14] S. Aich, K. Younga, K. Hui, A. Absi, and M. Sain, "A nonlinear decision tree based classification approach to predict the Parkinson's disease using different feature sets of voice data," *21st International Conference on Advanced Communication Technology (ICACT)* 2019, pp. 638-642.

[15] Y. Li and P. Wang, "Classification of Parkinson's Disease by Decision Tree Based Instance Selection and Ensemble Learning Algorithms," *Journal of Medical Imaging and Health Informatics*, vol. 7, 04/02 2017.

[16] H.-H. Zhang *et al.*, "Classification of Parkinson's disease utilizing multi-edit nearest-neighbor and ensemble learning algorithms with speech samples," *BioMedical Engineering OnLine*, vol. 15, no. 1, p. 122, 2016/11/16 2016.

[17] S. Wu and J. Guo, "A Data Mining Analysis of The Parkinson's Disease," *iBusiness*, vol. 03, 01/01 2011.

[18] J. Sujatha and S. P. Rajagopalan, "Performance evaluation of machine learning algorithms in the classification of parkinson disease using voice attributes," *International Journal of Applied Engineering Research*, vol. 12, pp. 10669-10675, 01/01 2017.

[19] L. Naranjo, C. J. Pérez, Y. Campos-Roca, and J. Martín, "Addressing voice recording replications

for Parkinson's disease detection," *Expert Systems with Applications,* vol. 46, pp. 286-292, 2016/03/15/ 2016.

[20] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations,* 12/22 2014.

[21] S. Ruder, "An overview of gradient descent optimization algorithms," 09/15 2016.

[22] P. Probst, B. Bischl, and A.-L. Boulesteix, *Tunability: Importance of Hyperparameters of Machine Learning Algorithms*. 2018.