“Parkinson’s Disease Detection using Speech and Image Data”

A dissertation report submitted in partial fulfilment of the requirements for the degree

Of

BACHELOR OF COMPUTER APPLICATION (BCA)

Academic Session (2018 – 21)

By,

Sauvik Nath

(BCA1833035)

Project Guide,

Dr. Himanish Shekhar Das

Department of Computer Science & Information Technology,
Cotton University (2018-2021)

Department of
Computer Science and IT
Cotton University

Guwahati–781001
Assam, India

Prof. Jiban Jyoti Das
Head of the Department

Office: 0361 - 2608074
Email: hodcsit@cottonuniversity.ac.in

CERTIFICATE

This is to certify that **Sauvik Nath** bearing Enrolment No. **BCA1833035** has prepared the dissertation entitled *"**Parkinson's Disease Detection using Speech and Image Data***" under the guidance of **Dr. Himanish Shekhar Das**, Dept. of CS & IT, Cotton University.

The project is the result of his effort and endeavours. The project is found worthy of acceptance for the award of Bachelor of Computer Application Sixth (6^{th}) Semester. No part of this report has been reproduced for any other degree or diploma.

Prof. Jiban Jyoti Das

HOD (CS&IT Department)

Cotton University

Department of
Computer Science and IT
Cotton University

Guwahati–781001
Assam, India

CERTIFICATE

This is to certify that the report titled **"*Parkinson's Disease Detection using Speech and Image Data*"** submitted to Cotton University in the Department of Computer Science and IT, in partial fulfilment for the award of the degree of Bachelor of Computer Application, is a record of the project work carried out by **Sauvik Nath** under my supervision and guidance.

No part of this report has been submitted elsewhere for award of any other degree or diploma.

Date:

Place:

Dr. Himanish Shekhar Das

(Project Guide)

CERTIFICATE

This is to certify that the project report entitled "***Parkinson's Disease Detection using Speech and Image Data***" is an approved work done by **Sauvik Nath**, Enrolment No. – **BCA1833035**, a student of **Sixth (6th)** Semester from Cotton University, Pan Bazar, Guwahati under my supervision and guidance. This work has been prepared as a partial fulfilment of the requirement for the award of the degree of Bachelor of Computer Application (BCA) under Cotton University, Guwahati from Department of Computer Science and IT, Cotton University, Guwahati.

I also certify that the reported work has not been submitted as a part of any other project work.

The conduct and behaviour of the student at Cotton University has been good throughout. I wish him a bright and prosperous future ahead.

Date:

Place:

Dr. Himanish Shekhar Das

(Project Guide)

DECLARATION

I hereby declare that the project report entitled *"**Parkinson's Disease Detection using Speech and Image Data**", submitted* to the Department of Computer Science & IT, Cotton University is prepared by me under the guidance of **Dr. Himanish Shekhar Das** at Cotton University, Pan Bazar, Guwahati, Department of CS & IT, Cotton University.

I also declare that no part of this report has been submitted in any form for the award of any other degree or diploma of any other institution or university.

Date: __/__/2021

Place:

Sauvik Nath

(BCA1833035)

Department of CS & IT,
Cotton University

ACKNOWLEDGEMENT

I like to share my sincere gratitude to all those who helped me in completion of this project. During the work I faced many challenges due to my lack of knowledge and experience but these people helped me to get over from all the difficulties and in final compilation of my idea to a shaped sculpture.

I would like to thank my project guide Dr. Himanish Shekhar Das sir for his governance and guidance, because of which I was able to learn the minute aspects of a project work. His daily motivation and encouragement glided me towards the completion of this project, because of which I could gather my thoughts and implement it in the code to make the software better.

I would like to thank the management of Department of Computer Science and Information Technology, Cotton University for providing us such an opportunity to learn from these experiences

I am thankful to all the Faculties and Staff of Department of Computer Science and Information Technology, CU for their help and support towards this project.

I am also thankful to our whole class and most of all to our parents who have inspired us to face all the challenges and win all the hurdles in life.

Thank you,

Sauvik Nath

(BCA1833035)

Department of CS & IT, Cotton University

ABSTRACT

Identification of the correct biomarkers with respect to particular health issues and detection of the same is of paramount importance for the development of clinical decision support systems. For the patients suffering from Parkinson's disease (PD), it has been duly observed that impairment in the handwriting is directly proportional to the severity of the disease. Also, More than 90% of the Parkinson Disease (PD) patients suffer from vocal cord disorders. Speech impairment is an early indicator of PD. Therefore, correctly identifying such biomarkers accurately and precisely at the onset of the disease will lead to a better clinical diagnosis.

In this project, a system design is proposed through deep learning and Image classification for analysing Spiral drawing patterns and wave drawing patterns in patients suffering from Parkinson's disease and healthy subject through the speed and pen pressure applied while sketching or writing something that is found much lower in patients suffering from Parkinson's disease and also analysing voiceprint data or speech features for early detection of Parkinson's disease.

The system developed in the project leverages multiple convolutional neural networks (CNN), for analysing the speech features of a PD patient and also uses Transfer learning and fine tuning to train a pre trained model on image data. The complete image model was trained on the data of 55 patients and has achieved an overall accuracy of 87.5%, average recall of 84%, average precision of 83.5% and average f1 score of 83.94%. The speech models gained a highest accuracy of 86.7% with the best being on the XGBBoost Classifier which is a gradient boosting algorithm for feature selection. Other models such as Random Forest, Support Vector Machine gained 78% to 80% accuracy respectively.

CONTENTS

INTRODUCTION

1.1. About the Project

Parkinson's disease (PD) is a progressive nervous system disorder that affects movement. Symptoms start gradually, sometimes starting with a barely noticeable tremor in just one hand. Tremors are common, but the disorder also commonly causes stiffness or slowing of movement. In the early stages of Parkinson's disease, your face may show little or no expression. Your arms may not swing when you walk. Your speech may become soft or slurred. Parkinson's disease symptoms worsen as your condition progresses over time. In Parkinson's disease, certain nerve cells (neurons) in the brain gradually break down or die. Many of the symptoms are due to a loss of neurons that produce a chemical messenger in your brain called dopamine. When dopamine levels decrease, it causes abnormal brain activity, leading to impaired movement and other symptoms of Parkinson's disease.

To prevent the major negative impact on PD patient’s it is necessary to detect the PD at the early stage. One of the most common effects that are easily noticeable among the PD patients and used most commonly in the early stage of diagnosis is finding the difference in handwriting and sketching abilities. The non-invasive measures such as sketching of a shape such as spiral, waves, and other handwritten texts could be easily distinguished from one person to another person as well as a person with PD and a person without PD.

Also, Vocal characteristics are considered to be one of the earliest signs of this disease. At the early stage, the subtle anomalies of the sound are imperceptible to the listener, but the recorded speech signals can be acoustically analysed for objective evaluation. The existing PD tests use PET-CT imaging equipment to detect whether the dopaminergic neurons are reduced. But such tests are expensive and have high radiation levels, thereby reducing the accuracy of the diagnosis. Thus, a high accuracy, convenient, non-intrusive and inexpensive diagnostic method is required.

So, in this project, an attempt has been made to develop an automated system through deep learning and computer vision, that are trained with the features extracted from the different sketches performed by the healthy group of patients as well as PD patients to assess the severity of the PD disease among different stages as well as between the healthy groups of patients. The investigation performed in this study to differentiate the healthy subjects from PD subjects based on the spiral sketches by extracting features from the images sketched by the healthy subjects and PD subjects. The structure of the paper is organized in the following way.

Also, a study is done on speech data of Parkinson's patient to find the most important features that are necessary to detect PD on an early stage and monitor the patient effectively. An attempt to train a Deep Learning model with speech features of Parkinson's patient has been made and we have received some fruitful result out of the study which will really be useful and certainly helpful.

1.2. Understanding of a Neural Network

Neural Networks is one of the most popular machine learning algorithms at present. It outpaces many other peer algorithms in solving complex problems with speed and accuracy.

1.2.1 Basic Building Block of Neural Networks

"Neuron" From the biological definition, a neuron is the basic working unit of the brain that takes sensory input from the external world and transmits the response to other nerve cells, muscle or gland cells to perform the respective action. In simple terms, it receives an input signal, processes it and delivers an output signal. Similarly, a neuron in machine learning is defined as a placeholder for a mathematical function that takes input and performs the given function on them to provide an output. The function used in neuron is generally termed as activation function. Figure 1 describes X1, X2, X3 as inputs fed to a neuron. It delivers a result output after applying the function on the inputs given.

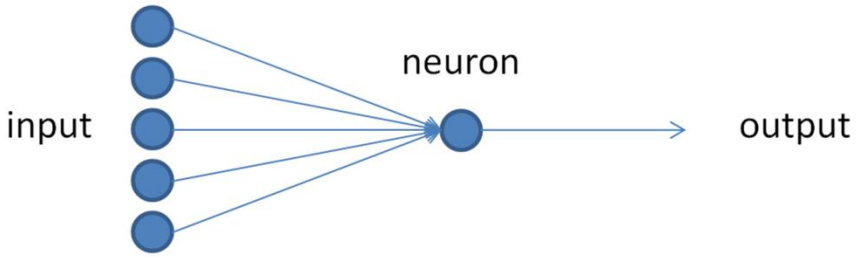


Figure 1: Basic Neuron

1.2.2. Neural Network

The fundamental blocks in neural networks are called layers. A layer is a collection of neurons that takes an input and provides an output. Each neuron has its own function and the inputs are processed layer by layer. Figure 2 describes the leftmost layer as the input layer, rightmost layer is the output layer and the middle layer is called hidden layer. The number and type of hidden layers differ according to the problem. If the neural network has more than 3 or 4 hidden layers, it is called as Deep Neural Network.

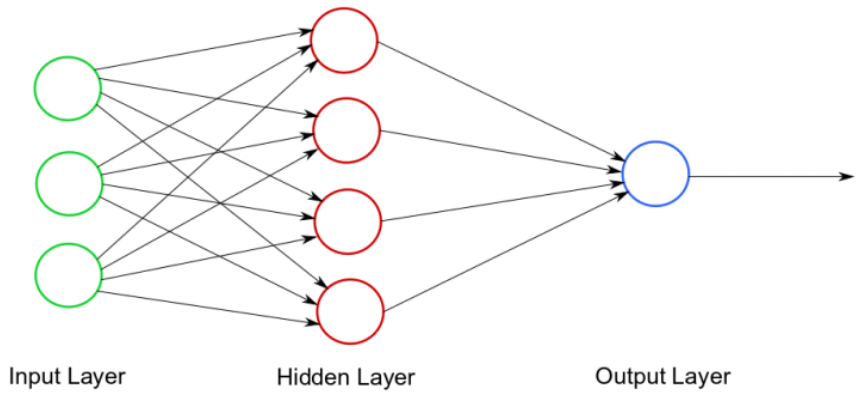


Figure 2: Simple Neural Network

1.2.3. Composition of a Convolution Neural Network

The general activation functions used in most neural networks are step, sigmoid, tanh, ReLU and leaky ReLU. Instead of these activation functions, if we use convolution and pooling functions as activation functions in the hidden layers of the neural network, then it is called as Convolution Neural Network. The hidden layers that use convolution or pooling function as activation function is called convolution layer or pooling layer respectively. Figure 3 describes the convolution layers, pooling layers and fully connected layer together that forms a Convolution Neural Network.

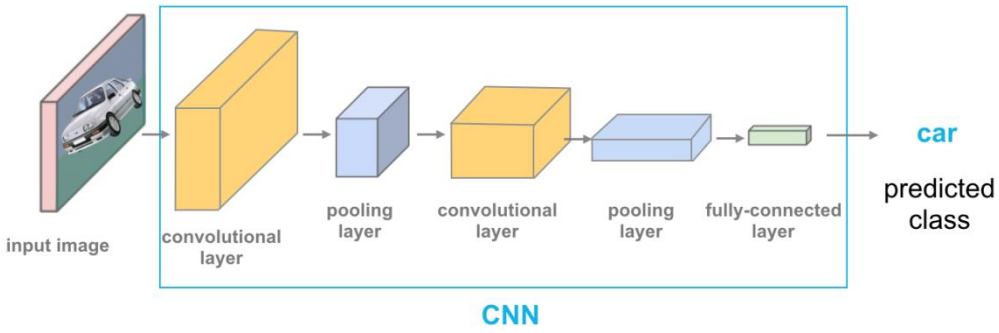


Figure 3: Convolution Neural Network Layers

1.3. Transfer Learning

Using the conventional model for Image classification, we need lots of data to train the network. Although, there is the ImageNet dataset with millions of data available, we may not find the necessary amount of data for all domains. In this scenario, we can use the Transfer Learning technique, which uses pre-trained Neural Networks and uses the obtained weights on new data. The idea of Transfer Learning is that, if we need to classify Task1 images but do not have enough data to train the neural network, then we

find a related Task2 that has vast data and train the neural network on it . Then, we use the whole model or few layers as desired from the Task 2 to solve Task 1. Figure 4 describes the earlier layers of ConvNet that deals with generic features such as edge detectors, texture detectors, patterns, etc. The later layers to the end of the network become more specific to the details of the image dataset. Here come the advantages of Transfer Learning. We will fix/freeze the earlier layers of the pre-trained network and retrain the rest of the layers with fine tuning of back propagation. Instead of training our network from the first layer, we just transfer the weights already learned by a pre-trained network such as VGG16 and save time and efforts.

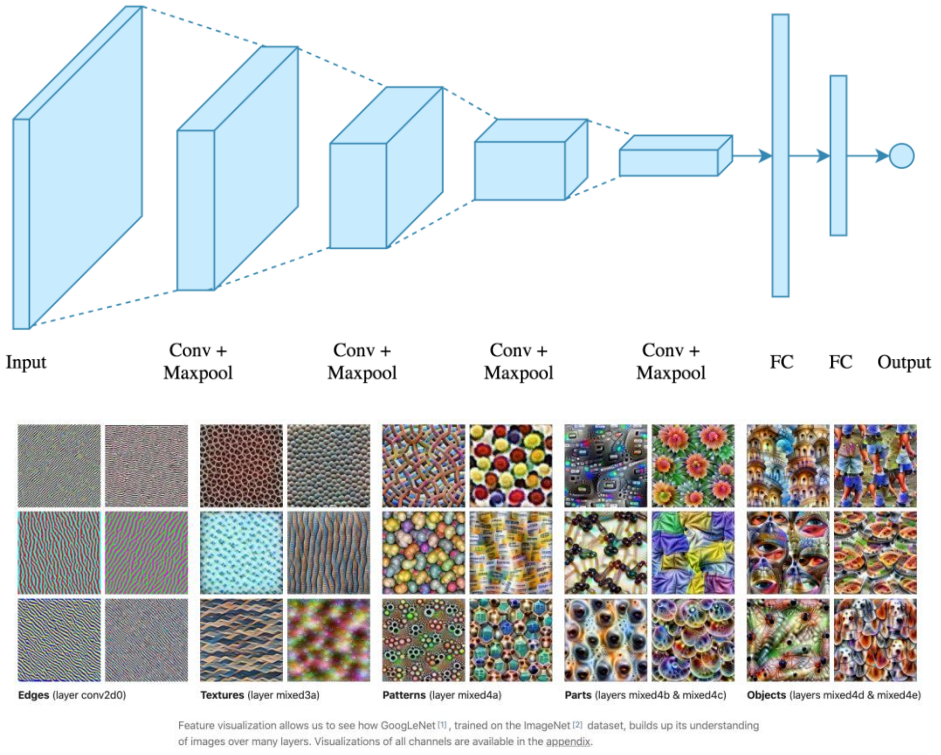


Figure 4: Difference between Initial Layers and Later Layers

LITERATURE SURVEY

2.1. Speech

During recent years, several papers have been published regarding detection of Parkinson's disease using speech and voice data. An author proposed a model where Neural Networks, Decision Tree, Regression and DMneural were used for detection of PD and comparative analysis was made. Zhijing Xu, Juan Wang, Ying Zhang, Xiangjian He, proposed a method based on Deep Neural Network (DNN) recognition and classification combined with Mini-Batch Gradient Descent (MBGD) for the classification of PD. A paper used (LOSO) validation technique on MFCC voice recording samples and SVM classifier to discern between PD patients and healthy people. Another paper describes a two-staged attribute selection and classification model for detection of PD. Ali proposed an early predictive model for PD detection using two-dimensional simultaneous sample and feature selection method. Another author used ensemble learning techniques combining different classifiers for PD detection with accuracy of 86% . There are also some other papers having proposed models with very high accuracy on detection of PD. For instance, a paper combined weighted clustering with Complex Valued Artificial Neural Network (CVANN) in their proposed model which achieved an accuracy of 99.5%. However, despite having high accuracy these experiments show biased results. The reported data-sets used in the experiments had small data points and each subject had multiple voice recording samples. Despite having several papers published on PD detection, improvement is still necessary to provide models with better accuracy, robustness and efficiency.

2.2. Image

Some of the studies related to the implementation of the machine learning techniques for the development of an automated system using different datasets related to Parkinson's disease have been discussed here to support our studies. Zham et al. proposed a study that used two criteria such as speed, and pen-pressure while performing the sketches to distinguish PD subjects at different stages. They have extracted features

from the sketches and proposed a method that can provide a correlation factor between the features and severity level of the PD. Finally, they performed the Mann-Whitney test to validate the study that these methods can be used to distinguish different stages of the PD. They observed that there was a significant difference in the correlation factor at different stages of PD. Kotsavasiloglou et al., presented an investigation based on the trajectory of the tip of the pen on the surface of the pad while drawing simple horizontal lines by the healthy subjects and PD subjects. They extracted features from the simple drawings and trained the machine learning algorithms using those features to distinguish the PD subjects from the healthy subjects. They have used different classifiers such as Naïve Bayes, AdaBoost, Logistic Regression, J48, Support Vector Machine (SVM), and Random Forest classifiers to train the features for developing an automated system. The performance metrics used in that study were accuracy, Area under the curve (AUC), True positives (TP), and True negatives (TN). They achieved an accuracy of 91%, and TP of 0.88 and TN of 0.95. Memedi et al. proposed a study based on the spiral data collected using telemetry touch screen devices in the home environments to distinguish off episodes and peak dose dyskinesia using machine learning algorithms. Several features were extracted from the data and used as input to the machine learning classifiers. They have used Support Vector Machine, Logistic Regression, Random Forest, and Multi-Layer Perceptron (MLP) to train the features for the development of automated systems and found MLP performed well among all the classifiers with an accuracy of 84%. Aich et al. proposed a study that used a voice dataset to distinguish PD patients from others. They have used different feature selection techniques to find the best features that can be used to train different classifiers. The feature selection techniques used for that study is principal component analysis. Finally, a performance comparison study was performed using two groups of datasets such as original feature sets and PCA based feature sets using nonlinear decision tree-based classifiers. It was found that the random forest classifier (RFC) was the best classifier among them and PCA based feature set performance is better than the original feature sets. The best accuracy of 96.83% was found with RF classifier and PCA based feature sets.

METHODOLOGY

3.1 Image

For the Image Part we have used a method of Deep Learning and computer vision for Image classification of two target labels called healthy and Parkinson's. We have used Data augmentation and transfer learning and fine tuning for the image classification part on various pre trained models such as MobileNet, VGGNet etc... Since we had a very limited dataset of 30 Images for validation and 72 images for training purpose for both spiral and wave drawings.

3.1.1 Data Acquisition and Collection

The data for the study has been collected from Kaggle's data repository which was posted by K Scott Mader. The data acquisition process was performed at the RMIT, University, Melbourne, Australia. For the data acquisition procedure, 55 subjects were recruited were 28 subjects belonged to the Control Group and 27 subjects from the Parkinson Group. All the subjects were asked to take two tests namely the Spiral drawing test and the Wave drawing test. The drawing of the spiral was captured using an A3 size commercially used a tablet, where an A3 paper was placed over the tablet and an ink pen was used to sketch the spiral and the wave. Figure 1 shows a sample of images that were obtained from the data acquisition process.

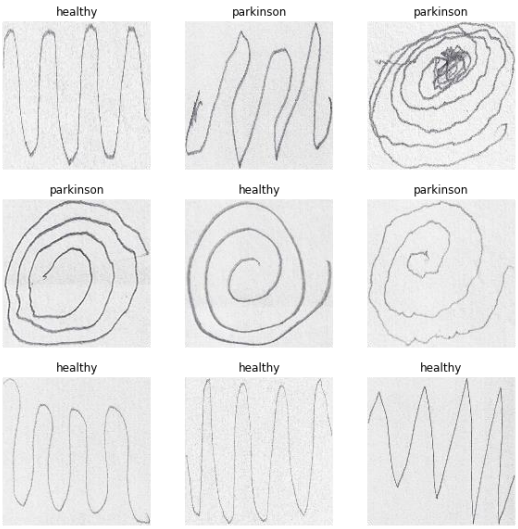


Figure 5: Sample Images for both the sketches and of both the classes.

3.1.2 Data Pre-processing and Augmentation

The images used in the study were subjected to resizing and histogram equalization. The images from the spiral sketches were resized to 256 px width and 256 px height, whereas the wave sketches were resized to the width of 256 px and height of 512 px. In the images that have been collected for the study, it can be observed that the images lack contrast and brightness and overall clarity. Therefore, contrast enhancement and adjustment were performed over all the images using Histogram equalization. For the preparation of images for training, all the images were subjected to a static policy-based augmentation. As the number of images is quite less in the dataset, therefore, the application of Deep Learning algorithms such as CNN's becomes quite difficult. Therefore to ingest some synthetic samples in the training dataset and also to increase the diversity in the dataset image augmentations were performed. Table 1 below show the different image augmentation parameters which were applied to the training data for wave and spiral sketched respectively. Moreover, Figure 3 shows the histogram equalized and augmented images of the training data set from both wave and spiral sketches

Augmentation Parameters	Settings
Horizontal Flip	True
Vertical Flip	True
Width Shift Range	0.1
Height Shift Range	0.1
Brightness Range	(0.5, 1.5)
Shear Range	0.2
Zoom Range	0.2
Max Crop Percentage	0.2
Crop Probability	0.3
Rotation Range	360

Table 1: Data Augmentation Parameters for spiral and wave images

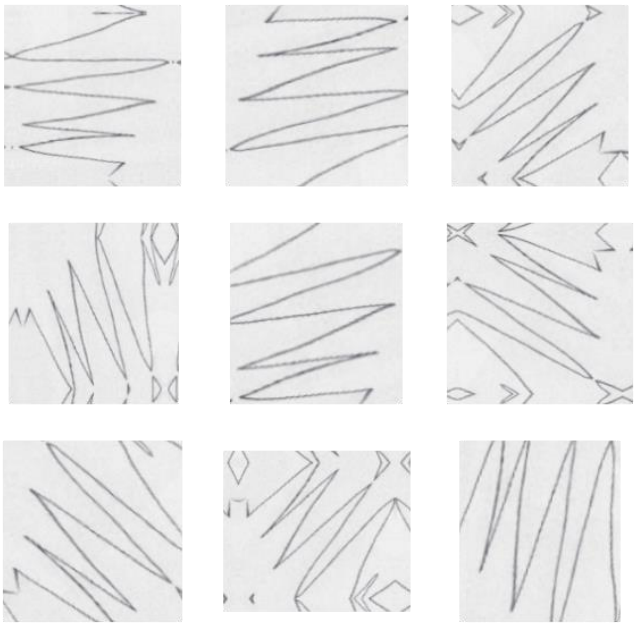


Fig 6: Sample Image after Data Augmentation and Pre processing

3.2 Speech

For the Speech part we have used speech recognition technologies via deep learning and by training a machine learning model for prediction of two target class namely healthy and Parkinson's. First of all we dropped the highly correlated features and then we have used feature selection algorithms. After that we have trained a gradient boosting classifier known as XGBoost for gradient boosting and feature selection and trained the model to get an accuracy of 87% on the validation dataset.

What is correlation or Highly Correlated Features?

Correlation simply means a mutual relationship between two or more things. Consider data points (x_i , y_i), i = 1,2,…n in a dataset. The objective of correlation is to see if large values of "x" are paired with large values of "y" and small values of "x" are paired with small values of "y". If not, check if small values of "x" are paired with large values of "y" and vice versa.

In statistics, the above phenomenon is measured by a fitting function called correlation coefficient. The formula to measure correlation is

$$\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

Correlation coefficient formula

$\bar{x}$ and $\bar{y}$ are means of x and y respectively. When correlation coefficient is < 0, we say that x and y are negatively correlated. If it is > 0, they are positively correlated. Correlation coefficient varies between -1 and 1.

Most important point to note is, correlation measures only the association between the two variables and does not measure causation. i.e., large values of "y" is **not caused** by large values of "x" or vice versa, rather it so happens that such data pairs just exist in the dataset.

Why exclude highly correlated features?

Regression is all about learning the weight vector from the training data and using it to make predictions. The formula for obtaining the weight vector is

$$w_{\mathrm{LS}} = (X^T X)^{-1} X^T y.$$

We have a probabilistic view of regression where it is assumed that the dependent variable "y" is normally distributed with variance σ^2. Under this assumption, it can be mathematically shown that the variance of the above weight vector W_{ls} is

$$\sigma^2(X^TX)^{-1}$$

Variance of W_{ls}

For the model to be stable enough, the above variance should be low. If the variance of the weights is high, it means that the model is very sensitive to data. The weights differ largely with training data if the variance is high. It means that the model might not perform well with test data. So, the natural question would now be,

When will the variance of W_{ls} be large?

Now you should have guessed that when we have highly correlated features, the variance of W_{ls} will be large. Yes, the guess is right !! But let us see how this is correct mathematically. Any n x d matrix can be decomposed as

$$X = USV^T$$

Singular Value Decomposition

The above decomposition is called "Singular Value Decomposition". The "S" matrix in the above equation is a non-negative diagonal matrix. Using this decomposition, the variance of W_{ls} can be re-written as

$$\text{Var}[w_{\text{LS}}] = \sigma^2(X^TX)^{-1} = \sigma^2 VS^{-2}V^T$$

When we have highly correlated features in the dataset, the values in "S" matrix will be small. So inverse square of "S" matrix (S^-2 in the above equation) will be large which makes the variance of W_{ls} large.

So, it is advised that we keep only one feature in the dataset if two features are highly correlated.

Feature Selection

A fundamental problem of machine learning is to approximate the functional relationship f *( )* between an input $X = \{x1, x2... xM\}$ and an output Y, based on a memory of data points, $\{Xi, Yi\}$, $i = 1... N$, usually the Xi's are vectors of real and the Yi's are real numbers. Sometimes the output Y is not determined by the complete set of the input features $\{x1, x2... xM\}$, instead, it is decided only by a subset of them $\{x(1), x(2)... x(m)\}$, where $m < M$. With sufficient data and time, it is fine to use all the input features, including those irrelevant features, to approximate the underlying function between the input and the output. But in practice, there are two problems which may be evoked by the irrelevant features involved in the learning process.

1. The irrelevant input features will induce greater computational cost. For example, using cached *k*d-trees as we discussed in last chapter, locally weighted linear regression's computational expense is $O(m3 + m2 \log N)$ for doing a single prediction, where N is the number of data points in memory and m is the number of features used. Apparently, with more features, the computational cost for predictions will increase polynomially, especially when there are a large number of such predictions, the computational cost will increase immensely.

2. The irrelevant input features may lead to over fitting. For example, in the domain of medical diagnosis, our purpose is to infer the relationship between the symptoms and their corresponding diagnosis. If by mistake we include the patient ID number as one input feature, an over-tuned machine learning process may come to the conclusion that the illness is determined by the ID number.

Another motivation for feature selection is that, since our goal is to approximate the underlying function between the input and the output, it is reasonable and important to ignore those input features with little effect on the output, so as to keep the size of the approximator model small.

Top reasons to use feature selection are:

- It enables the machine learning algorithm to train faster.
- It reduces the complexity of a model and makes it easier to interpret.
- It improves the accuracy of a model if the right subset is chosen.
- It reduces over fitting.

Feature Selection Algorithms

Some popular techniques of feature selection in machine learning are:

- Filter methods
- Wrapper methods
- Embedded methods

3.2.1 Data Acquisition and Collection

The data for the study has been collected from Kaggle's data repository which was posted by Dipayan Biswas. The data used in this study were gathered from 188 patients with PD (107 men and 81 women) with ages ranging from 33 to 87 (65.1Â±10.9) at the Department of Neurology in CerrahpaÅŸa Faculty of Medicine, Istanbul University. The control group consists of 64 healthy individuals (23 men and 41 women) with ages varying between 41 and 82 (61.1Â±8.9). During the data collection process, the microphone is set to 44.1 KHz and following the physicianâ€™s examination, the sustained phonation of the vowel /a/ was collected from each subject with three repetitions. Various speech signal processing algorithms including Time Frequency Features, Mel Frequency Cepstral Coefficients (MFCCs), Wavelet Transform based Features; Vocal Fold Features and TWQT features have been applied to the speech recordings of Parkinson's disease (PD) patients to extract clinically useful information for PD assessment.

```
predictors = dataset.drop(['class'], axis = 1)
criterion = dataset["class"]
predictors.head()
```

	id	gender	PPE	DFA	RPDE	numPulses	numPeriodsPulses	meanPeriodPulses	stdDevPeriodPulses	locPctJitter	locAbsJitter	rapJitter	ppq5Jitter	ddpJitter	locShimmer	locD
0	0	1	0.85247	0.71826	0.57227	240	239	0.008064	0.000087	0.00218	0.000018	0.00067	0.00129	0.00200	0.05883	
1	0	1	0.76686	0.69481	0.53966	234	233	0.008258	0.000073	0.00195	0.000016	0.00052	0.00112	0.00157	0.05516	
2	0	1	0.85083	0.67604	0.58982	232	231	0.008340	0.000060	0.00176	0.000015	0.00057	0.00111	0.00171	0.09902	
3	1	0	0.41121	0.79672	0.59257	178	177	0.010858	0.000183	0.00419	0.000046	0.00149	0.00268	0.00446	0.05451	
4	1	0	0.32790	0.79782	0.53028	236	235	0.008162	0.002669	0.00535	0.000044	0.00166	0.00227	0.00499	0.05610	

5 rows × 754 columns

Fig 7: Data Visualization of 5 rows after dropping the target class of 0 and 1

3.2.2 Data Pre-processing

From the 755 features (including the target class) we can visualize the highly correlated features as follows:

```
{'tqwt_entropy_log_dec_5', 'tqwt_entropy_log_dec_34',
'app_entropy_log_2_coef', 'tqwt_maxValue_dec_13',
'tqwt_stdValue_dec_1', 'det_LT_TKEO_std_9_coef',
'tqwt_TKEO_mean_dec_14', 'tqwt_skewnessValue_dec_2',
'det_LT_entropy_log_5_coef', 'tqwt_maxValue_dec_30',
'tqwt_minValue_dec_6', 'tqwt_entropy_log_dec_4', 'rapJitter',
'det_entropy_log_8_coef', 'ppq5Jitter', 'tqwt_entropy_log_dec_9',
'tqwt_maxValue_dec_27', 'app_LT_entropy_shannon_9_coef',
'tqwt_minValue_dec_4', 'app_LT_TKEO_std_1_coef',
'app_LT_TKEO_mean_6_coef', 'tqwt_minValue_dec_3',
'tqwt_TKEO_std_dec_33', 'tqwt_maxValue_dec_6',
'tqwt_skewnessValue_dec_33', 'tqwt_maxValue_dec_4',
'tqwt_minValue_dec_20', 'tqwt_stdValue_dec_29',
'det_entropy_shannon_3_coef', 'det_TKEO_std_1_coef',
'app_LT_entropy_log_9_coef', 'tqwt_kurtosisValue_dec_18',
'tqwt_maxValue_dec_18', 'tqwt_minValue_dec_28',
'tqwt_skewnessValue_dec_36', 'std_9th_delta_delta',
'tqwt_TKEO_std_dec_29', 'tqwt_kurtosisValue_dec_15',
'tqwt_entropy_log_dec_31', 'VFER_entropy', 'tqwt_TKEO_mean_dec_32',
'tqwt_skewnessValue_dec_35', 'app_entropy_log_8_coef',
'app_LT_TKEO_mean_2_coef', 'det_LT_TKEO_std_6_coef',
'tqwt_TKEO_std_dec_13', 'det_LT_TKEO_mean_6_coef',
'tqwt_kurtosisValue_dec_32', 'tqwt_entropy_shannon_dec_13',
'tqwt_entropy_log_dec_35', 'app_entropy_shannon_8_coef',
'std_1st_delta_delta', 'tqwt_entropy_log_dec_10',
'det_LT_TKEO_mean_10_coef', 'app_LT_TKEO_std_8_coef', 'Ed2_2_coef',
'std_delta_delta_log_energy', 'Ed2_5_coef', 'tqwt_TKEO_std_dec_36',
'app_entropy_shannon_4_coef', 'app_LT_TKEO_mean_5_coef',
'app_LT_TKEO_std_4_coef', 'app_det_TKEO_mean_10_coef',
'tqwt_TKEO_std_dec_30', 'tqwt_TKEO_mean_dec_2',
```

```
'app_LT_entropy_shannon_4_coef', 'tqwt_stdValue_dec_7',
'std_10th_delta_delta', 'tqwt_minValue_dec_23',
'tqwt_TKEO_mean_dec_13', 'app_LT_entropy_log_10_coef',
'tqwt_entropy_log_dec_7', 'tqwt_kurtosisValue_dec_9',
'tqwt_kurtosisValue_dec_35', 'tqwt_entropy_shannon_dec_6',
'apq5Shimmer', 'app_TKEO_std_10_coef', 'app_TKEO_std_8_coef',
'tqwt_TKEO_std_dec_26', 'app_LT_TKEO_mean_7_coef',
'tqwt_TKEO_std_dec_32', 'app_TKEO_std_6_coef', 'tqwt_TKEO_mean_dec_35',
'IMF_NSR_entropy', 'tqwt_TKEO_std_dec_35', 'ddaShimmer',
'app_LT_TKEO_std_5_coef', 'det_TKEO_std_3_coef',
'app_det_TKEO_mean_8_coef', 'det_entropy_log_5_coef',
'det_entropy_log_10_coef', 'tqwt_entropy_log_dec_17',
'tqwt_TKEO_mean_dec_7', 'tqwt_TKEO_mean_dec_10', 'tqwt_stdValue_dec_6',
'app_entropy_shannon_3_coef', 'tqwt_entropy_shannon_dec_8',
'tqwt_stdValue_dec_22', 'tqwt_minValue_dec_19',
'tqwt_TKEO_mean_dec_11', 'app_TKEO_std_4_coef', 'std_4th_delta_delta',
'det_LT_TKEO_mean_2_coef', 'tqwt_entropy_log_dec_16',
'tqwt_TKEO_std_dec_5', 'std_3rd_delta_delta', 'tqwt_TKEO_std_dec_15',
'app_LT_entropy_shannon_5_coef', 'tqwt_minValue_dec_13',
'tqwt_stdValue_dec_13', 'tqwt_TKEO_mean_dec_5', 'tqwt_stdValue_dec_25',
'tqwt_entropy_log_dec_19', 'apq3Shimmer', 'std_7th_delta_delta',
'det_TKEO_mean_5_coef', 'app_entropy_shannon_10_coef',
'tqwt_TKEO_std_dec_7', 'app_LT_entropy_log_8_coef',
'det_LT_TKEO_mean_8_coef', 'app_entropy_log_4_coef',
'det_LT_TKEO_mean_3_coef', 'app_LT_entropy_log_7_coef',
'det_TKEO_std_6_coef', 'app_entropy_shannon_6_coef',
'app_det_TKEO_mean_5_coef', 'det_LT_entropy_log_6_coef',
'tqwt_entropy_log_dec_14', 'app_entropy_log_5_coef',
'tqwt_kurtosisValue_dec_31', 'det_TKEO_mean_4_coef',
'det_LT_entropy_shannon_7_coef', 'det_LT_entropy_log_1_coef',
'det_LT_entropy_log_9_coef', 'det_TKEO_std_2_coef',
'tqwt_maxValue_dec_34', 'app_entropy_log_6_coef',
'tqwt_entropy_log_dec_32', 'tqwt_energy_dec_3',
'app_LT_entropy_log_4_coef', 'tqwt_maxValue_dec_25',
'tqwt_entropy_log_dec_11', 'tqwt_stdValue_dec_16',
'tqwt_minValue_dec_26', 'Ed2_1_coef', 'det_LT_TKEO_std_8_coef',
'tqwt_maxValue_dec_14', 'tqwt_TKEO_mean_dec_1', 'tqwt_maxValue_dec_12',
'tqwt_stdValue_dec_33', 'tqwt_stdValue_dec_10',
'det_LT_TKEO_std_10_coef', 'det_LT_TKEO_std_4_coef',
'app_det_TKEO_mean_4_coef', 'tqwt_maxValue_dec_20',
'tqwt_maxValue_dec_3', 'det_TKEO_mean_6_coef', 'tqwt_maxValue_dec_9',
'app_det_TKEO_mean_3_coef', 'tqwt_energy_dec_8',
'app_LT_entropy_log_2_coef', 'tqwt_TKEO_mean_dec_8', 'GNE_NSR_TKEO',
'app_LT_TKEO_mean_10_coef', 'app_LT_entropy_shannon_6_coef',
'app_LT_TKEO_std_10_coef', 'app_entropy_shannon_2_coef',
'app_det_TKEO_mean_2_coef', 'tqwt_minValue_dec_15',
'app_LT_TKEO_mean_9_coef', 'tqwt_kurtosisValue_dec_33',
'tqwt_maxValue_dec_17', 'tqwt_entropy_log_dec_3',
'tqwt_maxValue_dec_8', 'det_LT_TKEO_mean_9_coef', 'ddpJitter',
'det_entropy_shannon_2_coef', 'tqwt_maxValue_dec_36',
'tqwt_entropy_shannon_dec_16', 'tqwt_kurtosisValue_dec_5',
```

```
'tqwt_minValue_dec_31', 'det_LT_entropy_log_8_coef',
'det_TKEO_mean_3_coef', 'tqwt_maxValue_dec_35', 'Ed_2_coef',
'tqwt_maxValue_dec_5', 'tqwt_stdValue_dec_4', 'tqwt_stdValue_dec_2',
'tqwt_stdValue_dec_14', 'tqwt_entropy_log_dec_6',
'tqwt_minValue_dec_33', 'tqwt_TKEO_std_dec_14',
'tqwt_kurtosisValue_dec_2', 'locDbShimmer', 'std_1st_delta',
'tqwt_stdValue_dec_5', 'det_LT_entropy_shannon_10_coef',
'tqwt_stdValue_dec_9', 'tqwt_stdValue_dec_35', 'tqwt_maxValue_dec_24',
'tqwt_TKEO_std_dec_2', 'std_2nd_delta_delta', 'tqwt_stdValue_dec_19',
'app_LT_entropy_shannon_3_coef', 'det_TKEO_std_4_coef',
'det_TKEO_std_7_coef', 'Ed2_8_coef', 'app_entropy_shannon_7_coef',
'det_LT_entropy_log_2_coef', 'tqwt_TKEO_mean_dec_33',
'tqwt_energy_dec_33', 'det_LT_entropy_log_7_coef',
'tqwt_maxValue_dec_23', 'det_LT_TKEO_std_5_coef',
'det_LT_TKEO_mean_7_coef', 'tqwt_entropy_log_dec_18',
'app_LT_entropy_log_3_coef', 'std_5th_delta_delta',
'tqwt_stdValue_dec_11', 'app_entropy_log_7_coef',
'tqwt_TKEO_std_dec_25', 'tqwt_stdValue_dec_26', 'tqwt_maxValue_dec_7',
'tqwt_minValue_dec_16', 'det_entropy_log_9_coef',
'tqwt_TKEO_mean_dec_27', 'det_TKEO_std_8_coef',
'app_LT_entropy_log_1_coef', 'tqwt_maxValue_dec_33',
'app_LT_entropy_shannon_8_coef', 'tqwt_maxValue_dec_22',
'tqwt_maxValue_dec_26', 'tqwt_maxValue_dec_11', 'tqwt_minValue_dec_21',
'tqwt_entropy_log_dec_2', 'tqwt_TKEO_std_dec_27',
'det_LT_TKEO_mean_5_coef', 'tqwt_minValue_dec_32',
'tqwt_TKEO_std_dec_6', 'tqwt_maxValue_dec_15',
'app_LT_entropy_shannon_2_coef', 'tqwt_minValue_dec_18',
'tqwt_maxValue_dec_16', 'tqwt_TKEO_std_dec_11', 'det_TKEO_mean_1_coef',
'app_TKEO_std_1_coef', 'det_entropy_log_2_coef',
'app_entropy_log_10_coef', 'app_LT_TKEO_std_9_coef',
'det_TKEO_mean_2_coef', 'app_entropy_shannon_5_coef',
'tqwt_entropy_log_dec_8', 'tqwt_maxValue_dec_32',
'tqwt_entropy_shannon_dec_33', 'det_LT_TKEO_std_1_coef', 'Ed2_10_coef',
'tqwt_stdValue_dec_8', 'det_LT_TKEO_std_7_coef',
'app_LT_TKEO_std_2_coef', 'det_LT_entropy_shannon_9_coef',
'det_TKEO_mean_9_coef', 'tqwt_maxValue_dec_19', 'tqwt_maxValue_dec_2',
'det_TKEO_std_5_coef', 'tqwt_maxValue_dec_29',
'tqwt_entropy_log_dec_33', 'app_entropy_shannon_9_coef',
'det_LT_TKEO_std_3_coef', 'app_TKEO_std_5_coef',
'tqwt_TKEO_std_dec_16', 'tqwt_minValue_dec_14', 'det_TKEO_std_9_coef',
'tqwt_entropy_log_dec_13', 'app_LT_TKEO_mean_3_coef',
'app_entropy_log_3_coef', 'app_LT_TKEO_mean_4_coef',
'tqwt_skewnessValue_dec_4', 'tqwt_minValue_dec_9',
'tqwt_TKEO_mean_dec_34', 'tqwt_maxValue_dec_28',
'det_LT_entropy_log_10_coef', 'std_6th_delta_delta',
'app_LT_entropy_shannon_10_coef', 'det_LT_TKEO_std_2_coef',
'tqwt_minValue_dec_17', 'tqwt_minValue_dec_34', 'tqwt_TKEO_std_dec_3',
'det_TKEO_mean_7_coef', 'app_TKEO_std_7_coef', 'tqwt_TKEO_std_dec_31',
'tqwt_TKEO_std_dec_10', 'app_TKEO_std_9_coef',
'app_LT_entropy_shannon_1_coef', 'tqwt_maxValue_dec_21',
'tqwt_minValue_dec_27', 'tqwt_TKEO_mean_dec_9', 'tqwt_stdValue_dec_15',
```

```
'tqwt_TKEO_mean_dec_6', 'locAbsJitter', 'tqwt_stdValue_dec_17',
'app_LT_TKEO_std_6_coef', 'app_TKEO_std_2_coef',
'det_TKEO_mean_10_coef', 'std_11th_delta_delta', 'app_TKEO_std_3_coef',
'app_LT_TKEO_std_3_coef', 'tqwt_minValue_dec_29',
'tqwt_minValue_dec_22', 'tqwt_stdValue_dec_34', 'tqwt_minValue_dec_5',
'meanIntensity', 'tqwt_stdValue_dec_12', 'tqwt_maxValue_dec_10',
'det_entropy_shannon_1_coef', 'app_det_TKEO_mean_7_coef',
'det_LT_TKEO_mean_4_coef', 'tqwt_energy_dec_2', 'std_8th_delta_delta',
'det_LT_entropy_log_3_coef', 'tqwt_minValue_dec_25', 'apq11Shimmer',
'det_TKEO_mean_8_coef', 'tqwt_minValue_dec_36', 'std_12th_delta_delta',
'tqwt_TKEO_mean_dec_4', 'tqwt_entropy_log_dec_15',
'tqwt_skewnessValue_dec_32', 'det_LT_entropy_log_4_coef',
'det_LT_TKEO_mean_1_coef', 'tqwt_stdValue_dec_3',
'app_LT_entropy_log_5_coef', 'app_LT_entropy_log_6_coef',
'tqwt_TKEO_std_dec_12', 'tqwt_kurtosisValue_dec_3',
'meanNoiseToHarmHarmonicity', 'tqwt_TKEO_std_dec_9', 'Ed2_3_coef',
'tqwt_minValue_dec_30', 'app_det_TKEO_mean_9_coef',
'tqwt_kurtosisValue_dec_4', 'tqwt_TKEO_mean_dec_15', 'Ed2_7_coef',
'tqwt_entropy_shannon_dec_3', 'tqwt_minValue_dec_24',
'tqwt_TKEO_std_dec_4', 'app_LT_TKEO_mean_8_coef',
'tqwt_minValue_dec_35', 'app_LT_entropy_shannon_7_coef',
'app_entropy_log_9_coef', 'app_LT_TKEO_std_7_coef',
'tqwt_maxValue_dec_31', 'numPeriodsPulses', 'tqwt_TKEO_std_dec_34',
'det_TKEO_std_10_coef', 'tqwt_TKEO_std_dec_8',
'tqwt_kurtosisValue_dec_6', 'tqwt_TKEO_mean_dec_12',
'app_det_TKEO_mean_6_coef', 'tqwt_stdValue_dec_18',
'tqwt_TKEO_mean_dec_3', 'det_LT_entropy_shannon_8_coef',
'tqwt_TKEO_std_dec_24', 'tqwt_TKEO_mean_dec_29',
'tqwt_TKEO_mean_dec_30'}
```

So dropping all of the 364 highly correlated features from the dataset, we are left with a clean and precise amount of data for a healthy model and data representation.

```
[11] trainX_clean.head()
```

	id	gender	PPE	DFA	RPDE	numPulses	meanPeriodPulses	stdDevPeriodPulses	locPctJitter	locShimmer	meanAutoCorrHarmonicity	meanHarmToNoiseHarmonicity	minIntensity
52	17	0	0.83406	0.57905	0.45109	411	0.004702	0.000085	0.00119	0.09534	0.982524	18.418	74.788918
44	14	0	0.86478	0.76570	0.37052	400	0.004827	0.000054	0.00140	0.03993	0.990283	20.530	68.227558
247	82	0	0.86082	0.81274	0.46932	382	0.005050	0.000036	0.00117	0.14523	0.966982	15.247	40.382830
185	61	1	0.58956	0.67838	0.60454	276	0.007003	0.000119	0.00187	0.08761	0.942403	16.196	62.865438
434	144	0	0.80333	0.73467	0.42786	510	0.003785	0.000069	0.00108	0.04745	0.988650	20.390	64.157555

5 rows × 390 columns

```
testX_clean.head()
```

	id	gender	PPE	DFA	RPDE	numPulses	meanPeriodPulses	stdDevPeriodPulses	locPctJitter	locShimmer	meanAutoCorrHarmonicity	meanHarmToNoiseHarmonicity	minIntensity
747	249	1	0.85141	0.78231	0.42890	343	0.005628	0.000034	0.00077	0.09892	0.993051	22.038	77.317124
615	205	0	0.77498	0.72593	0.25021	426	0.004529	0.000020	0.00052	0.04403	0.994181	23.274	73.100015
546	182	0	0.84838	0.74531	0.41387	320	0.006034	0.000063	0.00073	0.06872	0.988311	19.986	64.027069
536	178	0	0.42102	0.59000	0.58835	406	0.004634	0.001896	0.00775	0.07081	0.940790	15.912	74.929042
717	239	1	0.81783	0.76591	0.67471	208	0.009279	0.000124	0.00310	0.06793	0.938384	14.452	67.109816

5 rows × 390 columns

Fig 8: Train and test split after cleaning and removing highly correlated data (5 rows x 360 columns

EXPERIMENT AND RESULT

4.1 Image

With the recent development in the field of machine intelligence after the advent of deep learning has plotted a paradigm shift in all the sectors. More specifically, in the healthcare sector, the emergence of deep learning and sophisticated computer-aided detection systems has levied to more precise and accurate diagnosis efficiently. In terms of Parkinson's disease, there are multiple biomarkers that need to be analysed for determining the clinical condition of a particular patient. One of the widely observed impairments that are caused by Parkinson's disease is hand tremors and diffused motor flexibility in hands which leads to a decrease in the ability to sketch and write. Therefore, determining the ability measure of writing and sketching can be considered as a very important biomarker for determining the clinical progression and detection of Parkinson's disease.

In this project, a system is developed to analyse the sketching pattern of spirals and waves in patients suffering from Parkinson's disease and also the detection of Parkinson's disease. For the development of the system pre trained convolutional neural networks and voting-ensemble classifiers were used to analyse the sketching pattern and detection of Parkinson's disease. In the study, a total of 102 spirals sketched and 102 wave sketches were used to train and validate the system.

4.1.1 Transfer Learning and Fine Tuning

a) Base Model from Pre-Trained ConvNets

After Data Pre-processing and Augmentation we further proceed to train a pre trained Image Classification model. In this case we are using a Model previously known as ImageNet by Google but now known as MobileNetV2.

This is pre-trained on the ImageNet dataset, a large dataset consisting of 1.4M images and 1000 classes. ImageNet is a research training dataset with

a wide variety of categories like jackfruit and syringe. This base of knowledge will help us classify cats and dogs from our specific dataset.

First, you need to pick which layer of MobileNet V2 you will use for feature extraction. The very last classification layer (on "top", as most diagrams of machine learning models go from bottom to top) is not very useful. Instead, you will follow the common practice to depend on the very last layer before the flatten operation. This layer is called the "bottleneck layer". The bottleneck layer features retain more generality as compared to the final/top layer.

First, instantiate a MobileNet V2 model pre-loaded with weights trained on ImageNet. By specifying the **include_top=False** argument, you load a network that doesn't include the classification layers at the top, which is ideal for feature extraction.

Figure 1 shows the complete summary of the base model.

```
Model: "mobilenetv2_1.00_160"
_________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
==================================================================
input_1 (InputLayer)         [(None, 160, 160, 3) 0
_________________________________________________________________
Conv1 (Conv2D)               (None, 80, 80, 32)   864         input_1[0][0]
_________________________________________________________________
bn_Conv1 (BatchNormalization)   (None, 80, 80, 32)   128         Conv1[0][0]
_________________________________________________________________
Conv1_relu (ReLU)            (None, 80, 80, 32)   0           bn_Conv1[0][0]
_________________________________________________________________
expanded_conv_depthwise (Depthw (None, 80, 80, 32)   288         Conv1_relu[0][0]
_________________________________________________________________
expanded_conv_depthwise_BN (Bat (None, 80, 80, 32)   128         expanded_conv_depthwise[0][0]
_________________________________________________________________
expanded_conv_depthwise_relu (R (None, 80, 80, 32)   0           expanded_conv_depthwise_BN[0][0]
_________________________________________________________________
expanded_conv_project (Conv2D)  (None, 80, 80, 16)   512         expanded_conv_depthwise_relu[0][0
_________________________________________________________________
expanded_conv_project_BN (Batch (None, 80, 80, 16)   64          expanded_conv_project[0][0]
_________________________________________________________________
block_1_expand (Conv2D)         (None, 80, 80, 96)   1536        expanded_conv_project_BN[0][0]
_________________________________________________________________
block_1_expand_BN (BatchNormali (None, 80, 80, 96)   384         block_1_expand[0][0]
_________________________________________________________________
block_1_expand_relu (ReLU)      (None, 80, 80, 96)   0           block_1_expand_BN[0][0]
_________________________________________________________________
block_1_pad (ZeroPadding2D)     (None, 81, 81, 96)   0           block_1_expand_relu[0][0]
_________________________________________________________________
```

.
.
.

```
.
.
.
.
block_16_expand (Conv2D)       (None, 5, 5, 960)    153600     block_15_add[0][0]
__________________________________________________________________________________________________
block_16_expand_BN (BatchNormal (None, 5, 5, 960)    3840       block_16_expand[0][0]
__________________________________________________________________________________________________
block_16_expand_relu (ReLU)     (None, 5, 5, 960)    0          block_16_expand_BN[0][0]
__________________________________________________________________________________________________
block_16_depthwise (DepthwiseCo (None, 5, 5, 960)    8640       block_16_expand_relu[0][0]
__________________________________________________________________________________________________
block_16_depthwise_BN (BatchNor (None, 5, 5, 960)    3840       block_16_depthwise[0][0]
__________________________________________________________________________________________________
block_16_depthwise_relu (ReLU)  (None, 5, 5, 960)    0          block_16_depthwise_BN[0][0]
__________________________________________________________________________________________________
block_16_project (Conv2D)       (None, 5, 5, 320)    307200     block_16_depthwise_relu[0][0]
__________________________________________________________________________________________________
block_16_project_BN (BatchNorma (None, 5, 5, 320)    1280       block_16_project[0][0]
__________________________________________________________________________________________________
Conv_1 (Conv2D)                 (None, 5, 5, 1280)   409600     block_16_project_BN[0][0]
__________________________________________________________________________________________________
Conv_1_bn (BatchNormalization)  (None, 5, 5, 1280)   5120       Conv_1[0][0]
__________________________________________________________________________________________________
out_relu (ReLU)                 (None, 5, 5, 1280)   0          Conv_1_bn[0][0]
==================================================================================================
========
Total params: 2,257,984
Trainable params: 0
Non-trainable params: 2,257,984
__________________________________________________________________________________________________
```

Fig 9: Base Model MobileNetV2 Summary

b) Feature Extraction

In this step, we will freeze the convolutional base created from the previous step and to use as a feature extractor. Additionally, you add a classifier on top of it and train the top-level classifier.

c) Freeze the convolutional base

It is important to freeze the convolutional base before you compile and train the model. Freezing (by setting layer.trainable = False) prevents the weights in a given layer from being updated during training. MobileNet V2 has many layers, so setting the entire model's trainable flag to False will freeze all of them. The 2.5M parameters in MobileNet are frozen, but there

are 1.2K *trainable* parameters in the Dense layer. These are divided between two tf.Variable objects, the weights and biases.

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 160, 160, 3)]     0
_________________________________________________________________
sequential (Sequential)      (None, 160, 160, 3)       0
_________________________________________________________________
tf.math.truediv (TFOpLambda) (None, 160, 160, 3)       0
_________________________________________________________________
tf.math.subtract (TFOpLambda (None, 160, 160, 3)       0
_________________________________________________________________
mobilenetv2_1.00_160 (Functi (None, 5, 5, 1280)        2257984
_________________________________________________________________
global_average_pooling2d (Gl (None, 1280)              0
_________________________________________________________________
dropout (Dropout)            (None, 1280)              0
_________________________________________________________________
dense (Dense)                (None, 1)                 1281
=================================================================
Total params: 2,259,265
Trainable params: 1,281
Non-trainable params: 2,257,984
_________________________________________________________________
```

Fig 10: Base Model Summary after freezing

d) Learning Curves

Let's take a look at the learning curves of the training and validation accuracy/loss when using the MobileNet V2 base model as a fixed feature extractor.

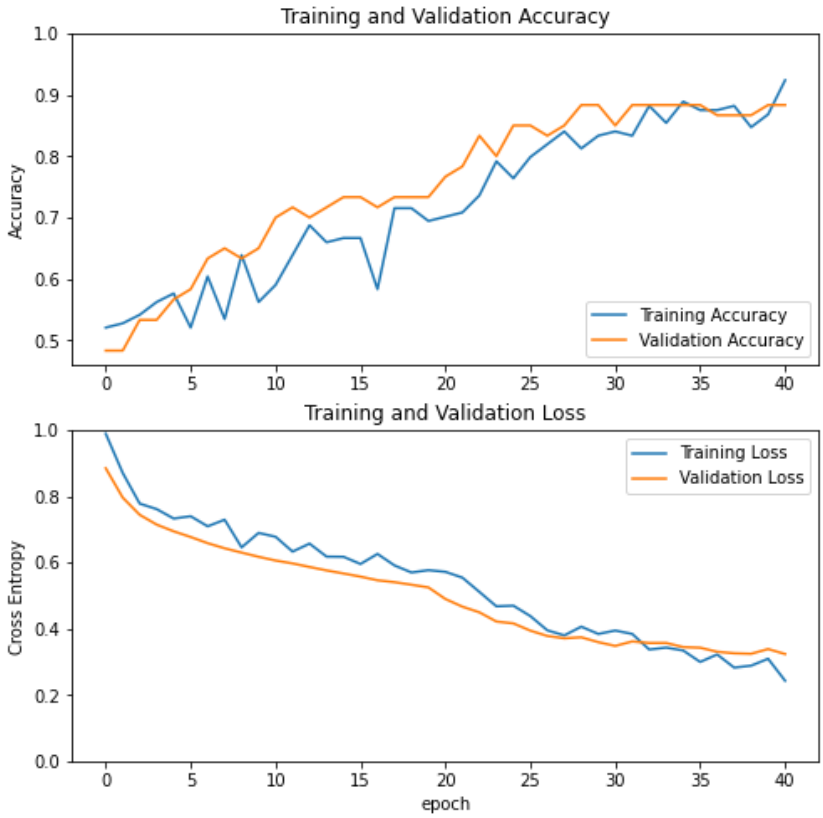


Fig 11: Learning Curve

e) Fine Tuning

In the feature extraction experiment, you were only training a few layers on top of an MobileNet V2 base model. The weights of the pre-trained network were **not** updated during training.

One way to increase performance even further is to train (or "fine-tune") the weights of the top layers of the pre-trained model alongside the training of the classifier you added. The training process will force the weights to be tuned from generic feature maps to features associated specifically with the dataset.

f) Un-freeze the top layers of the model

All you need to do is unfreeze the base_model and set the bottom layers to be un-trainable. Then, you should recompile the model (necessary for these changes to take effect), and resume training.

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
```

```
=================================================================
input_2 (InputLayer)         [(None, 160, 160, 3)]     0
_________________________________________________________________
sequential (Sequential)      (None, 160, 160, 3)       0
_________________________________________________________________
tf.math.truediv (TFOpLambda) (None, 160, 160, 3)       0
_________________________________________________________________
tf.math.subtract (TFOpLambda (None, 160, 160, 3)       0
_________________________________________________________________
mobilenetv2_1.00_160 (Functi (None, 5, 5, 1280)        2257984
_________________________________________________________________
global_average_pooling2d (Gl (None, 1280)              0
_________________________________________________________________
dropout (Dropout)            (None, 1280)              0
_________________________________________________________________
dense (Dense)                (None, 1)                 1281
=================================================================
Total params: 2,259,265
Trainable params: 1,862,721
Non-trainable params: 396,544
_________________________________________________________________
```

Fig 12: Base model after unfreezing the top layers

g) Learning Curve after Fine Tuning

Let's take a look at the learning curves of the training and validation accuracy/loss when fine-tuning the last few layers of the MobileNet V2 base model and training the classifier on top of it.

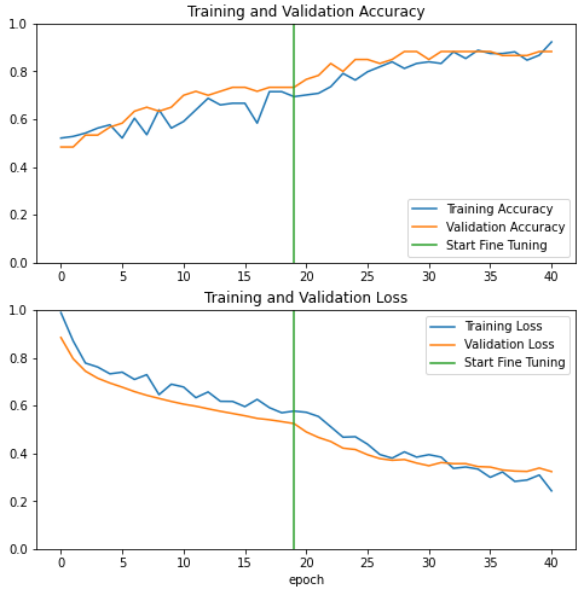


Fig 13: Learning Curve after Fine Tuning

h) Model Testing / Prediction on batch data

After fine tuning, the model nearly reaches 88% accuracy on the validation set. And now you are all set to use this model to predict if our patient has Parkinson's or not.

```
7/7 [==============================] - 1s 131ms/step - loss: 0.3236 -
accuracy: 0.8833
Test accuracy: 0.8833333253860474
```

Test 1:

```
Predictions:
 [1 0 0 1 1 1 1 0 1]
Labels:
 [1 0 0 1 1 1 1 0 1]
```

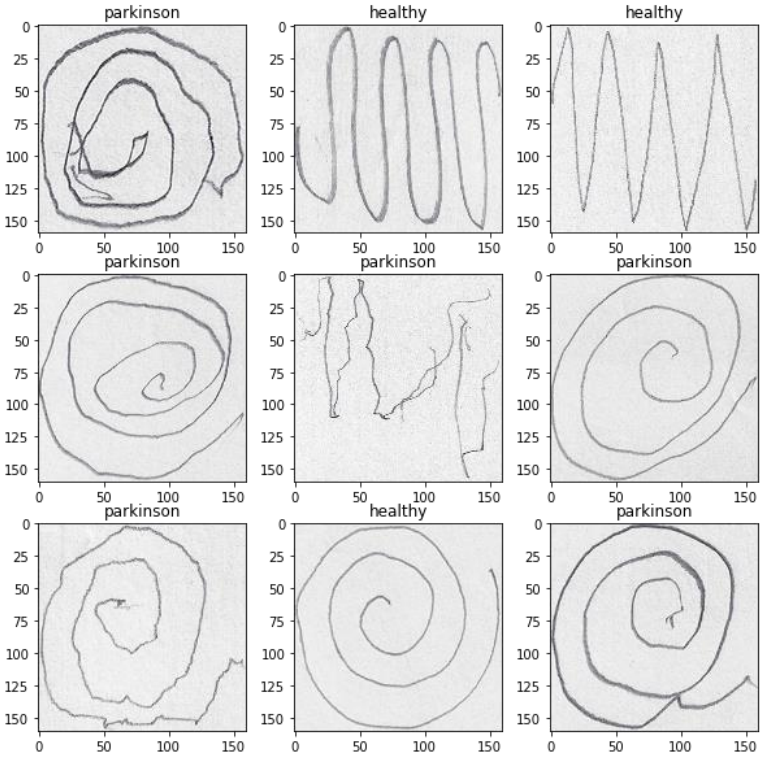


Fig 14: Predicted Labels

Test 2:

```
Predictions:
 [0 0 1 0 0 1 0 1 1]
Labels:
 [0 0 1 0 0 1 0 0 1]
```

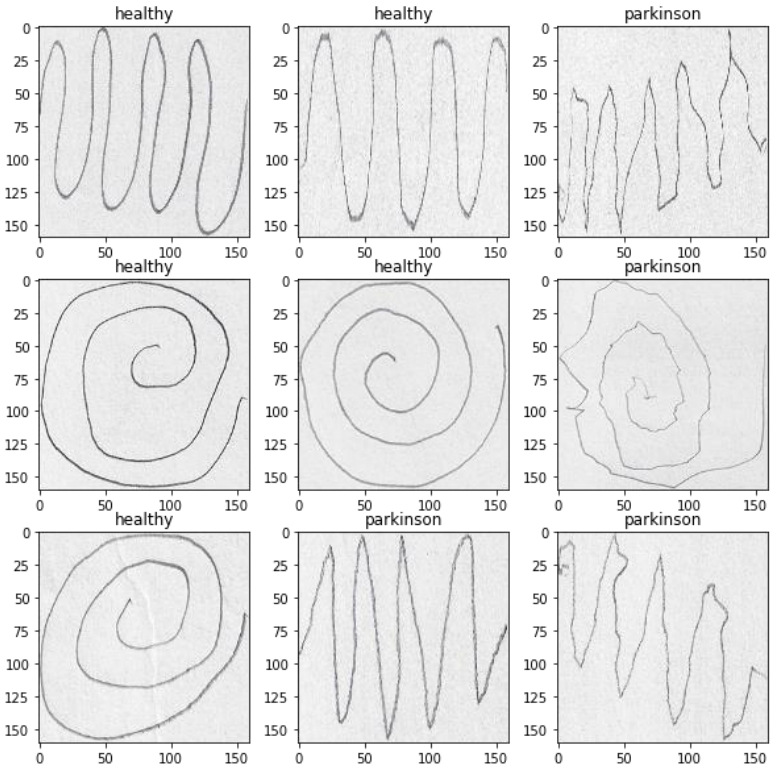


Fig 15: Predicted Labels

i) Confusion Matrix

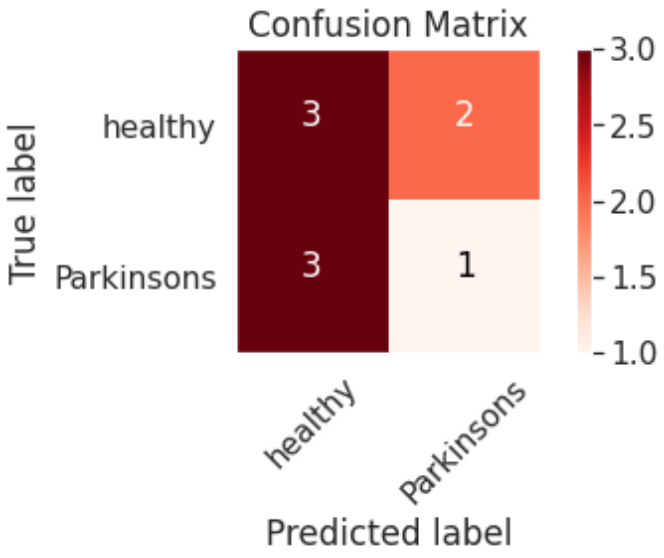


Fig 16: Confusion Matrix

From the confusion matrix we can conclude that the model shows a good consistency to decide the label on whether the patient is healthy or not. The model could decide **5 out of 6 times** whether the patient is healthy and **4 out of 3 times** whether the patient is suffering from Parkinson's disease which is a pretty good result in itself.

j) Table for Classification Report

Transfer Learning and Fine Tuning	Healthy	0.87	0.63	0.71	5
	Parkinson's	0.85	0.89	0.90	4
	Accuracy			0.83	3
	Micro Avg	0.78	0.77	0.77	3
	Weighted Avg	0.87	0.88	0.87	3

4.2 Speech

One of the points to remember about data pre-processing for regression analysis is multicollinearity. Our csv file consists of 755 instances and 700 plus columns of feature set. We cannot use this immensely big data to train our machine learning model since there is a very high chance of over fitting. So in order to reduce some of the unnecessary features we first of all found out which are the most highly correlated features by plotting a heat map of the data and we found around 364 of them to be highly correlated. All of the 364 features were dropped from the original data and after that feature selection algorithm was used.

4.2.1 Dropping Highly Correlated Features

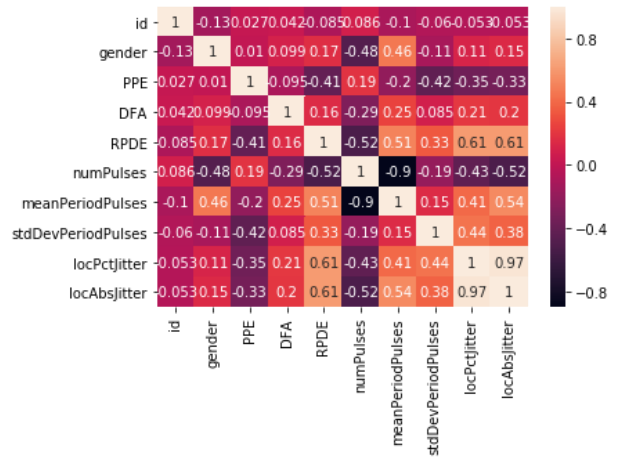


Fig 17: Heat map of Correlated Features

This is just a part of the heatmap and from this we can see that locPctjitter and locAbsjitter are highly correlated. So we drop one of the features.

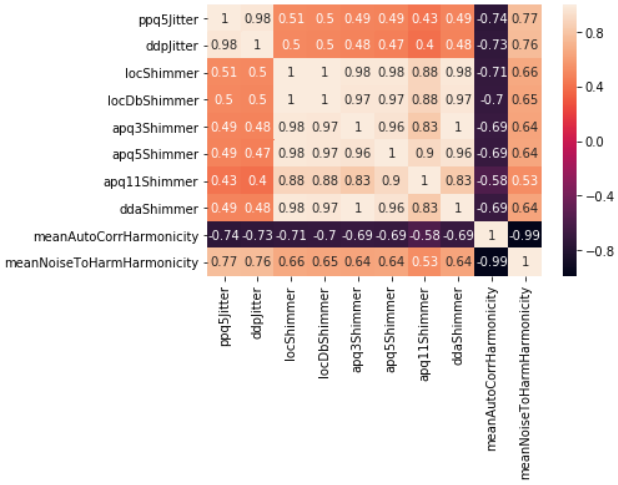


Fig 18: Heat map of Correlated Features

Another heatmap shows that ppq5jitter and ddpjitter is highly correlated.I OCShimmer and IOCDBShimmer is also highly correlated. If features have si milar name they correlated.

4.2.2 Feature Selection and Feature Importance

A benefit of using ensembles of decision tree methods like gradient boostin g is that they can automatically provide estimates of feature importance fro m a trained predictive model. Generally, importance provides a score that i ndicates how useful or valuable each feature was in the construction of the

boosted decision trees within the model. The more an attribute is used to m ake key decisions with decision trees, the higher its relative importance.

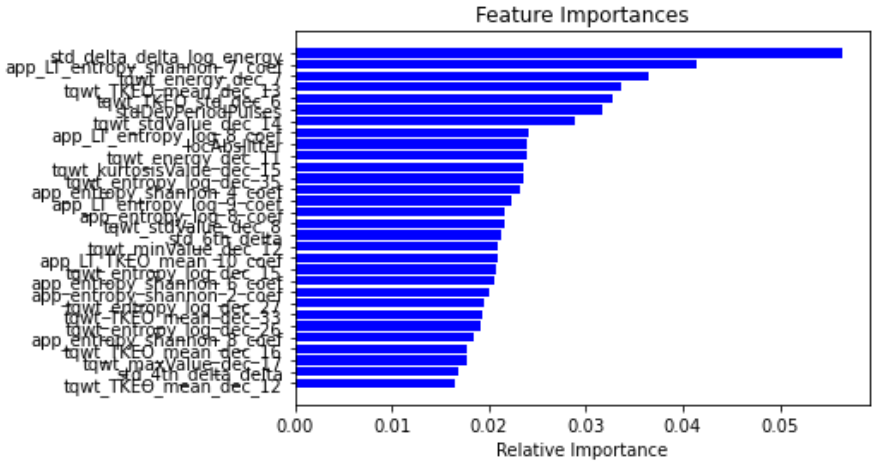


Fig 19: Feature Importance of Some Features

4.2.3 Classification

Now for the Classification Part we have used multi model classification for comparison of various models.

1. <u>XGBoost Classification</u>

This is a Gradient boosting classifier. It provides parallel boosting trees alg orithm that can solve Machine Learning tasks.

The result after using XGBoost Classifier is as follows:-

```
XGBoost Gradient Boosting:
Accuracy: 89.47%
```

a) Learning Curve

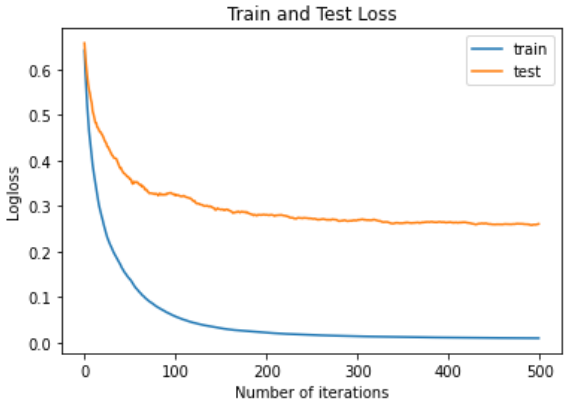
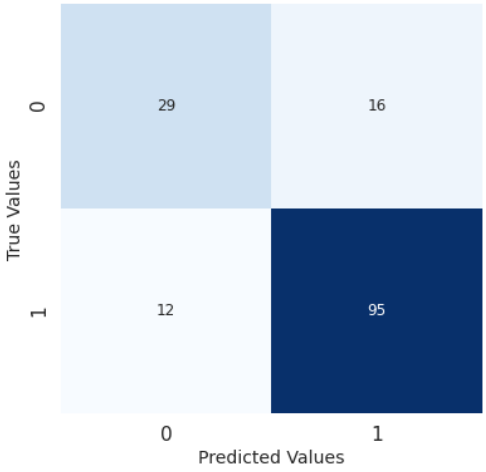


Fig 20: Learning Curve

b) Confusion Matrix

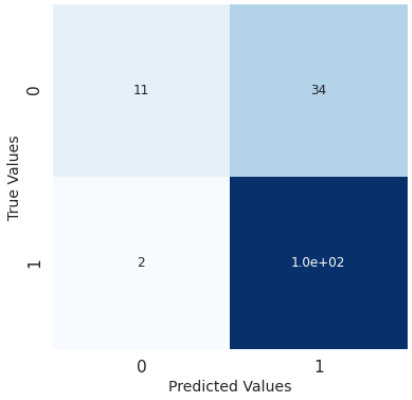


2. Random Forest

Random forests or **random decision forests** are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance. The result after using Random Forest Classifier is as follows:-

```
Random Forest Classification:
Accuracy: 80.26%
```

a) Confusion Matrix



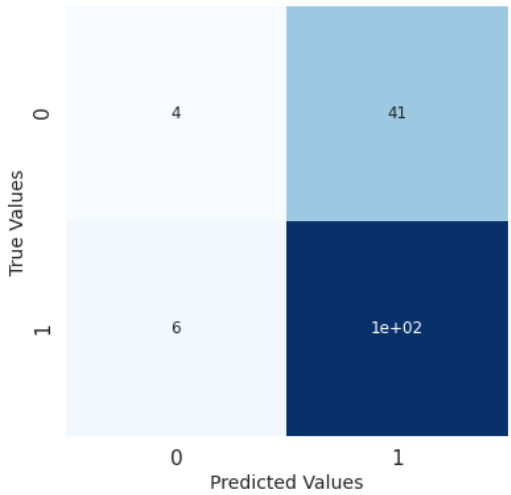
3. Support Vector Machine

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

The result after using Support Vector Machine Classifier is as follows:-

```
Support Vector Machine:
Accuracy Score: 73.03%
```

a) Confusion Matrix



4. Decision Tree

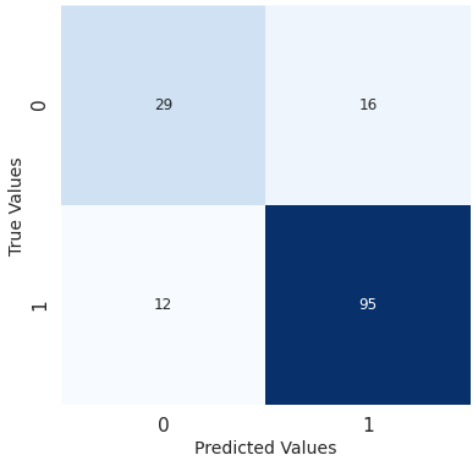
Decision Tree is a **supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.** In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and

have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The result after using Decision Tree Classifier is as follows:-

```
Decision Tree Classifier:
Accuracy Score: 82.89%
```

a) Confusion Matrix



4.2.4. Table for Classification Report of all models

Models		PRECISION	RECALL	F1-SCORE	SUPPORT
XGBoost	Healthy	0.87	0.60	0.71	45
	Parkinson's	0.85	0.96	0.90	107
	Accuracy			0.86	152
	Micro Average	0.86	0.78	0.81	152
	Weighted Average	0.86	0.86	0.85	152
Random Forest	Healthy	0.85	0.24	0.38	45
	Parkinson's	0.76	0.98	0.85	107
	Accuracy			0.76	152
	Micro Average	0.80	0.61	0.62	152
	Weighted Average	0.78	0.76	0.71	152
Support Vector Machine	Healthy	0.40	0.09	0.15	45
	Parkinson's	0.71	0.94	0.81	107
	Accuracy			0.69	152
	Micro Average	0.56	0.52	0.48	152
	Weighted Average	0.62	0.69	0.61	152
Decision Tree	Healthy	0.71	0.64	0.67	45
	Parkinson's	0.86	0.89	0.87	107
	Accuracy			0.82	152
	Micro Average	0.78	0.77	0.77	152
	Weighted Average	0.81	0.82	0.81	152

This table consists of the summary of classification results of all the models used for analysis. The table gives a better visualization on accuracy and other classification metrics. There are four ways to check if the predictions are right or wrong:

1. **TN / True Negative**: the case was negative and predicted negative
2. **TP / True Positive**: the case was positive and predicted positive
3. **FN / False Negative**: the case was positive but predicted negative
4. **FP / False Positive**: the case was negative but predicted positive

Precision — *What percentage of your predictions were correct?*

Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class, it is defined as the ratio of true positives to the sum of a true positive and false positive.

Precision:- Accuracy of positive predictions.

Precision = TP/(TP + FP)

Recall — *What percentage of the positive cases did you catch?*

Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives.

Recall:- Fraction of positives that were correctly identified.

Recall = TP/ (TP+FN)

F1 score — *What percentage of positive predictions were correct?*

The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Support

Support is the number of actual occurrences of the class in the specified dataset. Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or rebalancing. Support doesn't change between models but instead diagnoses the evaluation process.

DISCUSSION

This biomarker for image classification and speech classification is very interesting because it is obtained in a non-invasive way: the patient only must draw certain figures on a tablet and speak for some minutes to have them tested for PD. Based on this biomarker, medical-decision support tools can be developed for PD detection and patient supervision (after a positive diagnosis). Nowadays, PD diagnosis is very difficult and requires the combination of biomarkers based on symptoms like tremor, bradykinesia, and rigidity in order to increase the diagnosis accuracy. In this context, biomarkers can help to improve the health care by patient screening. This way, physicians can focus on the most probable patients reducing the diagnosis time. An earlier diagnosis would allow the development of specific treatment strategies for PD patients.

The supervision of patients is important for monitoring the progression of PD. PD provokes symptoms that can be alleviated with medication. A long-term use of this medication can cause side effects, such as dyskinesia (involuntary muscle movements). In order to reduce these side effects, the physician must periodically adjust the minimum dosage to manage the symptoms according to the disease progression. A widely method used by physicians to rate the current state of PD is the Unified Parkinson's Disease Rating Scale. One important limitation of this method is that the patient must visit the physician periodically. This is a problem for a patient with motion difficulties. A second limitation is that the information obtained by the physician is limited to a short session every few months. Non-invasive automatic biomarkers for supervising PD symptoms can provide objective and long-term supervision data to support the physician decision.

The sound damage of PD patients does not suddenly appear. This is a slow process and the symptoms are at the early stages which may be overlooked. In order to improve the evaluation of Parkinson's disease, the article uses the WMFCC to calculate the entropy method and take the average value to extract the participants' voiceprint characteristics, and then obtain the parameters of PD patients' dysphonia detection. Compared with the previous RASTA-PLP method, WMFCC has some improvements in

comprehensive performance and frame classification. RASTA-PLP characterizes the speech signal by doing short-time spectral analysis. The speech spectrum of RASTA-PLP takes into account the auditory characteristics of the human ear, because the input speech signal is processed by the auditory model, and it facilitates the extraction of anti-noise speech features. However, the cepstrum coefficients extracted by RASTA-PLP contain many frames, which consume lots of processing time in the classification process and hinder accurate diagnosis. HFCC readjusts these cepstral coefficients to quite similar amplitudes by liftering the cepstral coefficients, but the calculation process is complicated and takes a long time. WMFCC has the same anti-noise performance, and can extract more high frequency components from speech signals, so the extracted features have stronger representation ability. Then the samples in the PD database are trained, and 28 patients with PD are tested and judged. The results show that the characteristics of the sixth Mel frequency cepstral coefficient extracted from the CNN classification method achieves the highest classification accuracy of 89.5%. The accuracy is higher than those with the RBF, Linear, Polynomial and MLP kernel functions of the SVM and PLDA. That is to say the vowel /u/ speech samples contain more discriminative information than other speech samples. The traditional SVM method has better robustness in solving high dimensional problems, but there is no universal solution to nonlinear problems. The problem lies in the choice of kernel functions and the difficulty in implementing large-scale samples. PLDA is a channel compensation technique base on i-vector. When applied to classification, it is hard to distinguish information in speaker and channel. Compared with the above two classifiers, DNN adopts the layer-by-layer pre-training method based on the restricted Boltzmann machine for the unsupervised pre-training process, and then carries out supervised tuning training. The efficiency of the training is greatly improved, and the problem of local optimum is well improved. When using 28 independent test sets for PD patients, the maximum classification accuracy is always the data from DNN. Studies have shown that DNN can make a better improvement in the classification performance between PD patients and healthy people. It also shows that the DNN network structure is used to enhance the classification ability of voiceprint features.

Future Works

We can combine both the models (speech and image) for a combined accuracy for prediction on validation set after cross validation for much higher accuracy. This system design can be further implemented on a large scale if possible after collecting more data to perfect the model to 100% accuracy and this can leverage the use of traditional testing and reduce cost of various machineries and equipment required. We could further test the system on various ranges of patients starting from poor, mild and highly affected patients by PD and study the data on various data points and outliers that could be an increasing factor in the higher effectiveness of the model to detect PD. Also Hyperparameter tuning can be done on all the classification models specifically Gradient Boosting Algorithms to achieve even higher accuracy.

Histogram of Oriented Gradient (HOG) features of Spiral and Wave Diagrams of a PD Patient can be used to be fit to an Image classification model to provide higher accuracy and precision.

We can also use the 29 major Mel Frequency Cepstrum Coefficient (MFCC) and Weighted Mel Frequency Cepstrum Coefficient (WMFCC) features on the voice data analysis and use a classification model to detect PD. MFCC is a promising feature and a big contribution to Computer Science as it can prepare a model with fewer features and yield a greater accuracy and precision to detect a label or a class.

Further studies are required to determine whether similar results can be obtained from records of normal conversation or phone calls. This approach could be used to screen large patient populations at different stages of Parkinson's disease. The value of this approach to identify early prodromal PD remains to be determined.

CONCLUSION

In this project, we investigated Parkinson's disease recognition by means of deep learning. We explored many deep learning algorithms for two Datasets, **Image Classification** (Random Forest Classifier, Transfer Learning (VGGNet, MobileNet etc), Fine tuning) and **Speech Classification and Analysis** (Gradient Boosting (XGBoost), Feature Selection, Dropping Highly Correlated Features, Random Forest Classier, Decision Tree (ID3, CART) and more)

The main contribution has been the proposal of the spectrum as inputs to a CNN for PD detection from spiral drawing movements. The CNN includes convolution layers (features learning) and fully connected layers (for PD detection). We evaluated the detection capability of different directions during drawing movements obtaining the best results for X and Y directions. Using a public dataset, Parkinson Disease Spiral and Wave Drawings, the best results obtained in this work showed an accuracy of 88.3%, a F1-score of 87.7% and an area under the curve of 89.2%. These results validate the use of drawings movements to develop medical-decision support tools for PD detection (patient screening) and long-term patient supervision.

Gradient boosting algorithms can be used to identify patients with Parkinson's disease using a simple non-invasive speech test. We achieved an 89.4 5 accuracy with Gradient Boosting via XGB Classifier and other classification models achieved an average accuracy of 85% without hyperparameter tuning and fine tuning. We can see a good result on Speech and voiceprint analysis and Image Data Classification and can further use this system design to use other Machine Learning Algorithms to test the model to a higher accuracy.

REFERENCES

1. This dataset is collected from UCI Machine Learning Repository used for Speech classification of PD patients.
https://archive.ics.uci.edu/ml/datasets/Parkinson%27s+Disease+Classification#

2. This dataset consist of Spiral and Wave drawings of Healthy and PD patient.
https://www.kaggle.com/kmader/parkinsons-drawings

3. Hahne, U.; Schild, J.; Elstner, S.; Alexa, M. Multi-touch focus+ context sketch-based interaction. In Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling, New Orleans, LA, USA, 1–2 August 2009; ACM: New York, NY, USA, 2009; pp. 77–83.

4. Saunders-Pullman, R.; Derby, C.; Stanley, K.; Floyd, A.; Bressman, S.; Lipton, R.B.; Deligtisch, A.; Severt, L.; Yu, Q.; Kurtis, M.; et al. Validity of spiral analysis in early Parkinson's disease. Mov. Disord. 2008, 23, 531–537.
https://movementdisorders.onlinelibrary.wiley.com/doi/10.1002/mds.21874

5. Chui, K.T.; Lytras, M.D. A Novel MOGA-SVM Multinomial Classification for Organ Inflammation Detection. Appl. Sci. 2019, 9, 2284.
https://www.mdpi.com/2076-3417/9/11/2284

6. Ammenwerth, E.; Nykanen, P.; Rigby, M.; de Keizer, N. Clinical decision support systems: Need for evidence, need for evaluation. Artif. Intell. Med. 2013, 59, 1–3.
https://www.sciencedirect.com/science/article/pii/S0933365713000845?via%3Dihub

7. Dreiseitl, S.; Binder, M. Do physicians value decision support? A look at the effect of decision support systems on physician opinion. Artif. Intell. Med. 2005, 33, 25–30.
https://www.sciencedirect.com/science/article/pii/S0933365704001071?via%3Dihub

8. Mattison, H.A.; Stewart, T.; Zhang, J. Applying bioinformatics to proteomics: Is machine learning the answer to biomarker discovery for PD and MSA? Mov. Disord. 2012, 27, 1595–1597.
https://movementdisorders.onlinelibrary.wiley.com/doi/10.1002/mds.25189

9. Lahmiri, S.; Shmuel, A. Detection of Parkinson's disease based on voice patterns ranking and optimized support vector machine. Biomed. Signal Process. Control 2018, 49, 427–433. https://www.sciencedirect.com/science/article/abs/pii/S1746809418302271?via%3Dihub

10. Gómez-García, J.A.; Moro-Velázquez, L.; Godino-Llorente, J.I. On the design of automatic voice condition analysis systems. Part I: Review of concepts and an insight to the state of the art. Biomed. Signal Process. Control 2019, 51, 181–199. https://www.sciencedirect.com/science/article/abs/pii/S1746809418303239?via%3Dihub

11. Gómez-García, J.A.; Moro-Velázquez, L.; Godino-Llorente, J.I. On the design of automatic voice condition analysis systems. Part II: Review of speaker recognition techniques and study on the effects of different variability factors. Biomed. Signal Process. Control 2019, 48, 128–143. https://www.sciencedirect.com/science/article/abs/pii/S1746809418302416?via%3Dihub

12. Viteckova, S.; Kutilek, P.; Svoboda, Z.; Krupicka, R.; Kauler, J.; Szabo, Z. Gait symmetry measures: A review of current and prospective methods. Biomed. Signal Process. Control 2018, 42, 89–100. https://www.sciencedirect.com/science/article/abs/pii/S1746809418300193?via%3Dihub

13. San-Segundo, R.; Navarro-Hellín, H.; Torres-Sánchez, R.; Hodgins, J.; De la Torre, F. Increasing Robustness in the Detection of Freezing of Gait in Parkinson's Disease. Electronics 2019, 8, 119. https://www.mdpi.com/2079-9292/8/2/119

14. McLennan, J.E.; Nakano, K.; Tyler, H.R.; Schwab, R.S. Micrographia in Parkinson's disease. J Neurol. Sci. 1972, 15, 141–152. https://www.jns-journal.com/article/0022-510X(72)90002-0/pdf

15. Zham, P.; Arjunan, S.P.; Raghav, S.; Kumar, D.K. Efficacy of Guided Spiral Drawing in the Classification of Parkinson's Disease. IEEE J. Biomed. Health Inform. 2018, 22, 1648–1652. https://ieeexplore.ieee.org/document/8064621

16. Potgieser, A.R.; Roosma, E.; Beudel, M.; de Jong, B.M. The effect of visual feedback on writing size in parkinson's disease. Parkinsons Dis. 2015, 2015, 857041.
https://www.hindawi.com/journals/pd/2015/857041/

17. Drotár, P.; Mekyska, J.; Rektorová, I.; Masarová, L.; Smékal, Z.; Faundez-Zanuy, M. Evaluation of handwriting kinematics and pressure for differential diagnosis of Parkinson's disease. Artif. Intell. Med. 2016, 67, 39–46.
https://www.sciencedirect.com/science/article/pii/S0933365716000063?via%3Dihub

18. Letanneux, A.; Danna, J.; Velay, J.L.; Viallet, F.; Pinto, S. From micrographia to Parkinson's disease dysgraphia. Mov. Disord. 2014, 29, 1467–1475.
https://movementdisorders.onlinelibrary.wiley.com/doi/10.1002/mds.25990

19. Thomas, M.; Lenka, A.; Kumar Pal, P. Handwriting Analysis in Parkinson's Disease: Current Status and Future Directions. Mov. Disord. Clin. Pract. 2017, 4, 806–818.
https://movementdisorders.onlinelibrary.wiley.com/doi/10.1002/mdc3.12552

20. Rosenblum, S.; Samuel, M.; Zlotnik, S.; Erikh, I.; Schlesinger, I. Handwriting as an objective tool for Parkinson's disease diagnosis. J. Neurol. 2013, 260, 2357–2361.
https://link.springer.com/article/10.1007%2Fs00415-013-6996-x

21. Impedovo, D.; Pirlo, G. Dynamic Handwriting Analysis for the Assessment of Neurodegenerative Diseases: A Pattern Recognition Perspective. IEEE Rev. Biomed. 2019, 12, 209–220.
https://ieeexplore.ieee.org/document/8365754

22. Impedovo, D.; Pirlo, G. Chapter 7: Online Handwriting Analysis for the Assessment of Alzheimer's Disease and Parkinson's Disease: Overview and Experimental Investigation. In Series on Language Processing, Pattern Recognition, and Intelligent Systems. Pattern Recognition and Artificial Intelligence; World Scientific Publishing: Singapore, 2019; pp. 113–128.

23. van Drempt, N.; McCluskey, A.; Lannin, N.A. A review of factors that influence adult handwriting performance. Aust. Occup. Ther. J. 2011, 58, 321–328.
https://onlinelibrary.wiley.com/doi/10.1111/j.1440-1630.2011.00960.x

24. Kotsavasiloglou, C.; Kostikis, N.; Hristu-Varsakelis, D.; Arnaoutoglou, M. Machine learning-based classification of simple drawing movements in Parkinson's disease. Biomed. Signal Process. Control 2017, 31, 174–180.

https://www.sciencedirect.com/science/article/abs/pii/S1746809416300982?via%3Dihub

25. Gallicchio, C.; Micheli, A.; Pedrelli, L. Deep Echo State Networks for Diagnosis of Parkinson's Disease. arXiv 2018, arXiv:1802.06708.