CSL2050 : Pattern Recognition and Machine Learning
Minor Project Report

Project 5 : Detecting Parkinson's Disease

Submitted in partial fulfillment of the requirements of the Major Project for the Course CSL2050 : Pattern Recognition and Machine Learning

by

Harsh Tomar	B21AI049	PCA and MLP
Pranav Chakravarthy	B21EE050	SFS and Boosting
Krishna Gaurang	B21EE086	LDA and KNN

Indian Institute of Technology Jodhpur
NH 62, Surpura Bypass Rd, Karwar, Rajasthan 342030
May 2023

ABSTRACT

The goal of this study was to develop a supervised learning model for classifying the Medical Subjects into two classes based on their status of Parkinson's disease. The dataset contains various audio parameters of the voice recordings of the patients. The dataset is biased with Recording containing 23 Positive patients out of the total 31. Hence, we have used both accuracy and F1 score as the measures. We have applied dimensionality reduction and feature selection methods and have later trained several models on them.

For this project, we adhered to the entire Supervised Machine Learning pipeline.

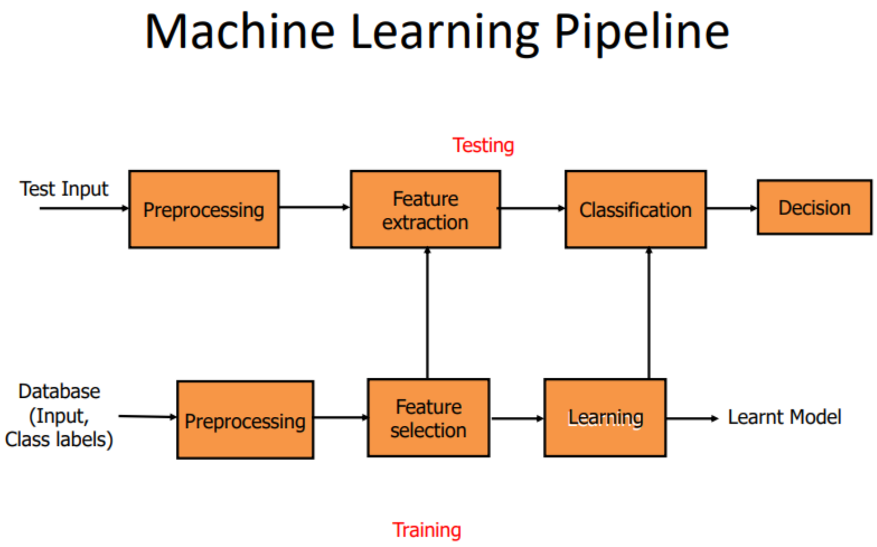


Figure: Machine Learning Pipeline

INTRODUCTION

In this study, we made an effort to classify the patients into healthy and sick using various supervised learning algorithms. At the very beginning, we have tried Linear Discriminant analysis to find out if the data is linearly separable. Then, we went ahead with Independent Component analysis along with Naive Bayes Classification to see whether this method works.

We have then proceeded to try the Sequential Forward Feature selection algorithm with Decision Tree Classifier as the base model. We have also applied PCA on the dataset to reduce the dimensionality.

The various Classification Models that we have used are:

1) LDA Classifier
2) Naive Bayes Classifier
3) Random Decision Forest (Bagging)
4) Boosting (XGBoost)
5) KNN Classifier
6) Support Vector Machine
7) Multi-Layer Perceptron

We have measured the performance of these models by their accuracy and F1 Score since it is a dataset of a very rare disease (Parkinson's disease).

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

$$\text{F1 Score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

Since the dataset is biased with the recordings containing 23 Positive patients out of the total 31, with a total of 195 recordings, we have used stratified train-test splits to maintain the ratio of positive to negative class distribution in the training and test sets.

PRELIMINARY ANALYSIS OF THE DATASET

In this step, we have performed the pre-processing. We see that there are no null values in the dataset. We have used the additional data given on Kaggle webpage to ascertain what each of the columns in the dataset represent:

Column Name	Description
name	ASCII subject name and recording number
MDVP:Fo(Hz)	Average vocal fundamental frequency
MDVP:Fhi(Hz)	Exports of goods and services per capita. Given as %age of the GDP per capita
MDVP:Flo(Hz)	Total health spending per capita. Given as %age of GDP per capita
MDVP:Jitter(%), MDVP:Jitter(Abs), MDVP:RAP, MDVP:PPQ, Jitter:DDP	Several measures of variation in fundamental frequency
MDVP:Shimmer,MDVP:Shimmer(dB),Shimmer:APQ3,Shimmer:APQ5,MDVP:APQ,Shimmer:DDA	Several measures of variation in amplitude
NHR, HNR	Two measures of the ratio of noise to tonal components in the voice
status	The health status of the subject (one) - Parkinson's, (zero) - healthy
RPDE, D2	Two nonlinear dynamical complexity measures
DFA	Signal fractal scaling exponent
spread1,spread2,PPE	Three nonlinear measures of fundamental frequency variation

Table: Column Name and their description in the dataset

We can see that all of the columns contain relevant information to the task at hand except the name column. Hence, we have dropped the name column.

Visualizing the dataset using histogram, we have:

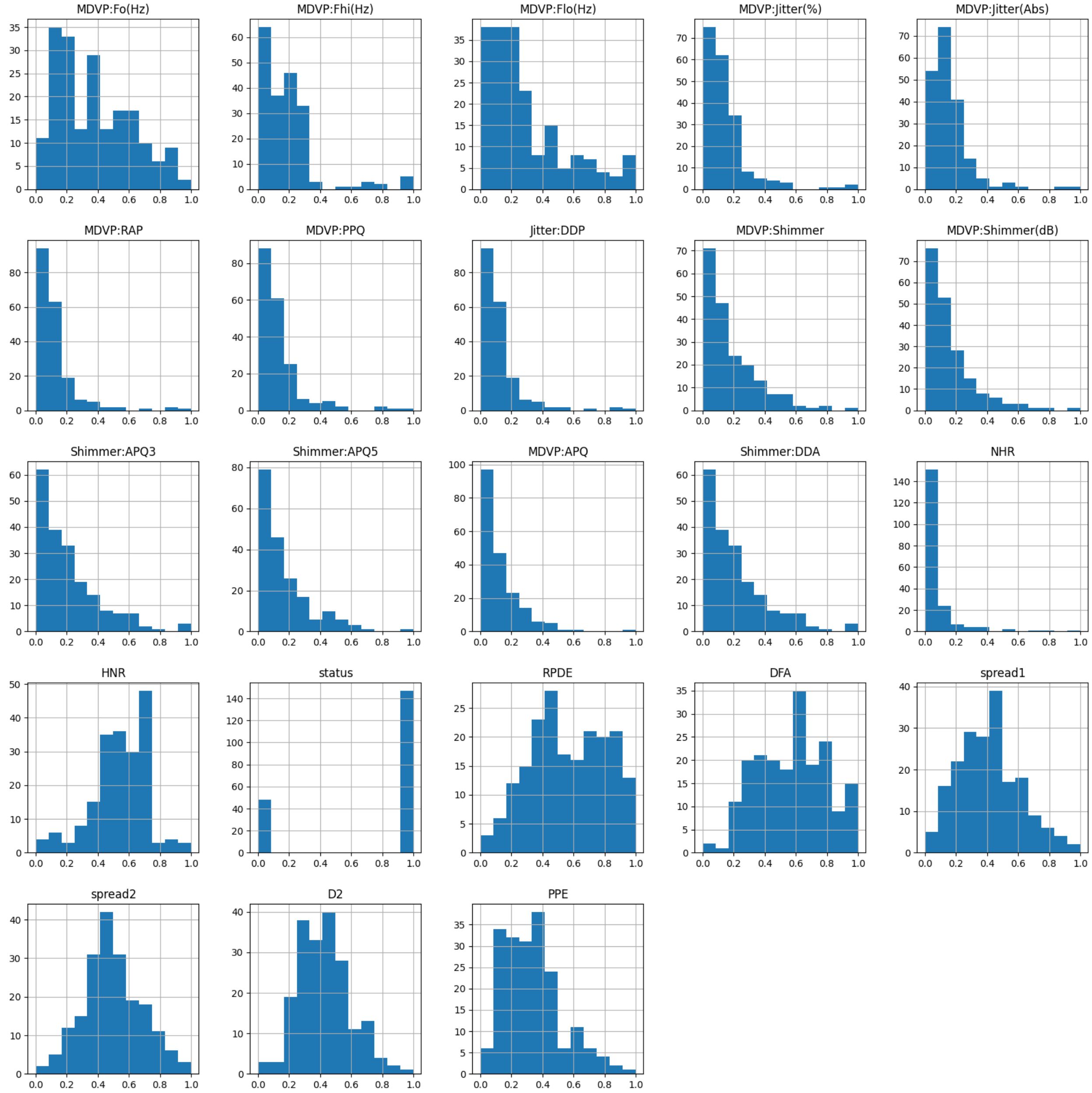


Figure: Histograms for visualizing the dataset

We then visualized the covariance matrix by using the heatmap function of seaborn library.

Visualizing the covariance matrix of features of the dataset using heatmap we have:-

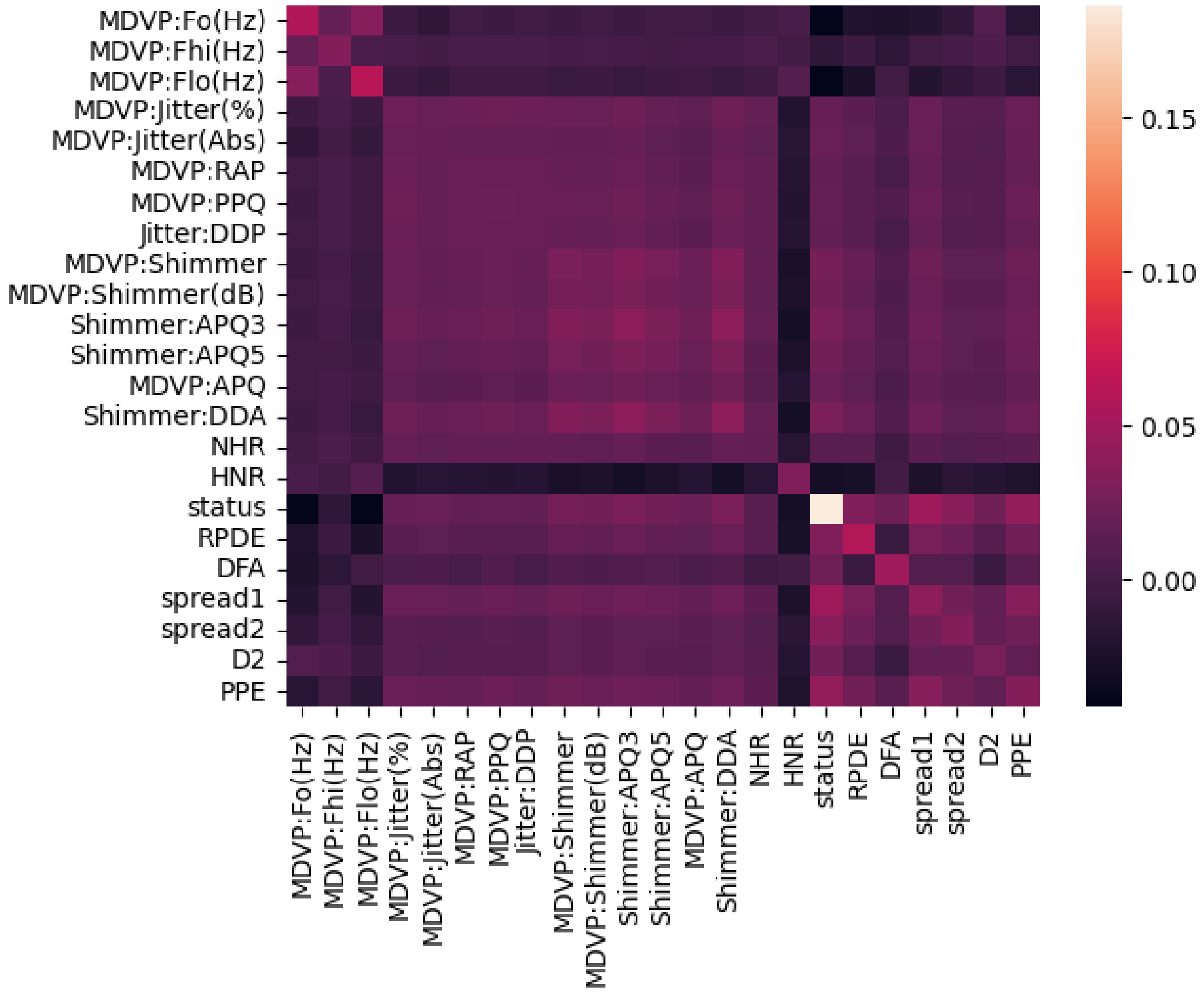


Figure: Heatmap for visualizing the covariance matrix of features

We can see that the covariance matrix shows that most of the features are almost statistically independent.

Dimensionality Reduction

Using LDA classifier

Since it is a 2 class classification problem, we can use the LDA Classifier to obtain a dimension/axis along which the data points are linearly separable. And the accuracy score we got using the LDA classifier is **91.79%.**

Then we plotted data points along this dimension(on Y-axis we took the classes) and the scatter plot we got is this:-

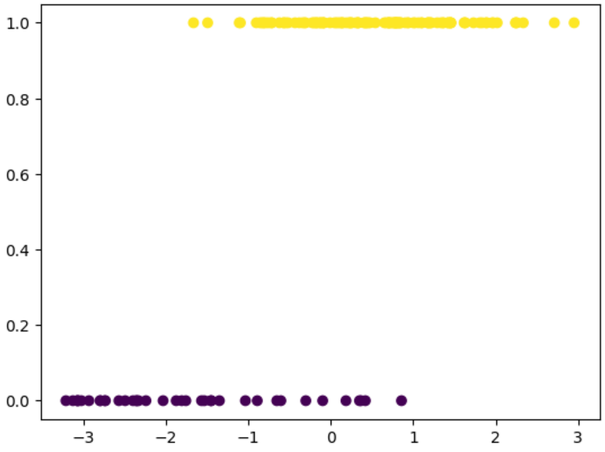


Figure: Scatter plot of data points along the axis obtained from LDA(X-axis)

From the above figure, we find that the dataset is not completely linearly separable, Thus we also expect that a support vector machine(SVM) with a linear kernel will not give accurate results for this dataset.

Using PCA on the Dataset

Now we reduced dimensions using PCA to 4, and the explained variance ratio for the 4 principal components is: [50.26840744 16.3374157   9.38505212 5.47492598].

With **4 principal components**, we can get a **total explained variance ratio** of **80%**. This is good enough for classification and to avoid overfitting and to preserve the information in the data. Plotting them using 3-D plots, we have:

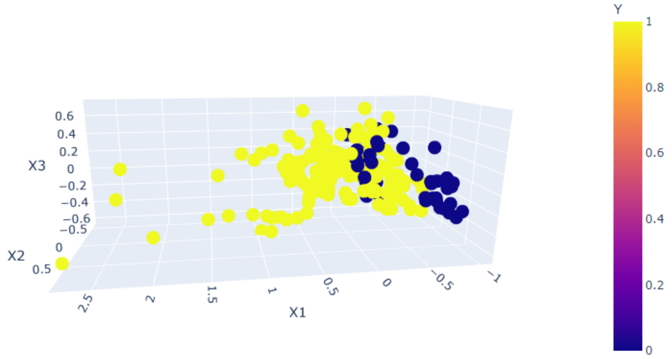


Figure: 3D plot of data points with first three principal components

With the above 3D plot, we can see that there is separability in the data. Also, we can spot a few outliers. However, since all the outliers belong to the same class, we will not remove them.

However, we have also learnt that PCA is not the best dimensionality reduction technique in all cases. Hence, we will also be using **ICA** with **Naive Bayes Classifier** and exploring Feature Selection.

Independent Component Analysis with Naive Bayes

By using ICA, we are transforming the data to get 7 independent components. And plotted the covariance matrix' heatmap for these 7 components.

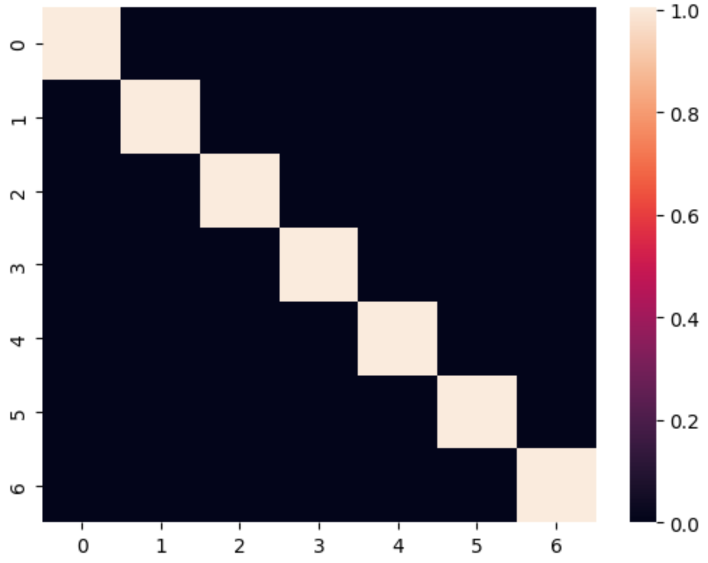


Figure: Heatmap for visualizing the covariance matrix for these 7 components

With this heatmap of the covariance matrix, it becomes safe to say that the features are statistically independent.

Since most of the features were continuous, we can use Gaussian Naive Bayes as the features represented Natural Phenomena (Acoustic) which are known to follow Gaussian Distribution for n>=30 from Central Limit Theorem.

The accuracy obtained for the **Gaussian Naive Bayes classifier** on the **ICA** dataset is **74.36%.**

We obtain a very bad accuracy for the Gaussian Naive Bayes Classification using ICA. Thus, we can say that **Gaussian Naive Bayes along with ICA** is **not useful** in the classification of this dataset. We could have also observed this from the histogram of the features noting that most of them did not follow a Gaussian Normal distribution.

Sequential Forward Feature Selection

We are also exploring the method of feature selection here. For the Forward Feature Selection, we are using the Classifier Model as Decision Tree with max_depth=7 to avoid overfitting. We are also using the stopping/scoring parameter as accuracy as that is the most important in the given classification task. There are 22 features in total.

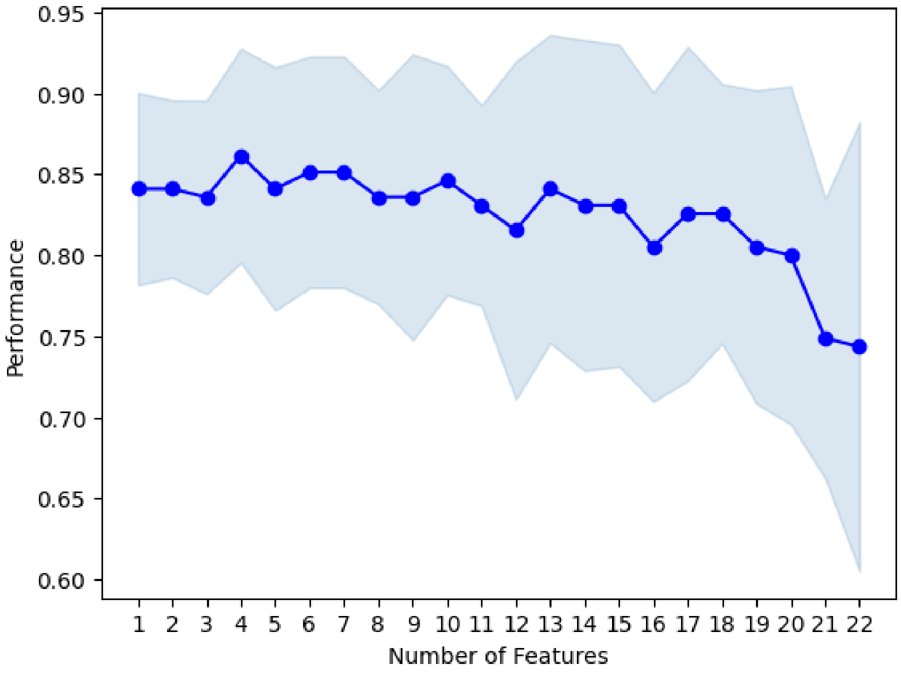


Figure: Performance of SFS with Number of Features

From the above, we can see that the optimum number of features to select would be 4. Hence we select the 4 best features.

These 4 best features are:

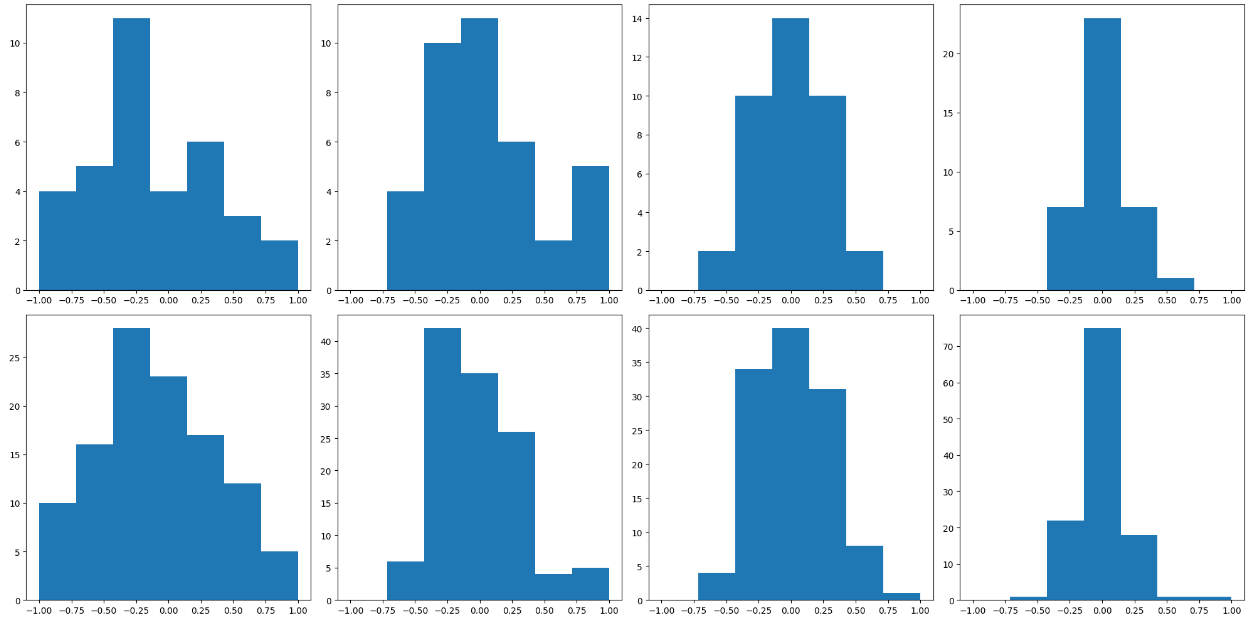
1) MDVP:Fo(Hz)
2) MDVP:Fhi(Hz)
3) MDVP:Flo(Hz)
4) MDVP:Jitter(%)

Learning

Naive Bayes Classifier on the PCA dataset

Here, we applied the Naive Bayes Classifier on the implemented dataset. The accuracy we obtained for this model is: **69.23%.**

Then we explored/visualized the distribution of for this case:-

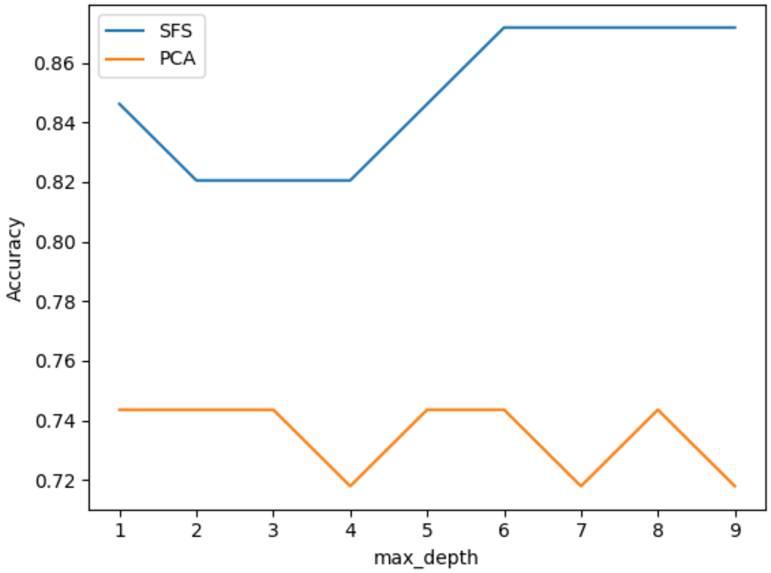


The above plots in row 1 indicate the negative class while those in row 2 indicate the positive class.

With these histograms, we can see that the distribution of PCA implemented dataset has 3 new features which follows a Gaussian (Bell-Shaped) distribution which is leptokurtic.

Ensemble Learning

Since the dataset is highly biased, we now make use of Ensemble Learning in the form of a Random Decision Forest Classifier. Tuning the hyperparameter max_depth of this classifier, we get:



Maximum accuracy obtained from ensemble learning: **87.18%.**

From this, we note that increasing the max_depth of the decision trees improves the accuracy as it can accommodate more complex decision boundaries.We also do notice is that we get a considerably higher accuracy than in the case of other models in which bagging is not being used.

Boosting

Boosting allows for combining many weak learners into a strong learner and to get a complex decision boundary. A weak learner is a learner which gives greater than 0.5 accuracy.

We've used XGBoost on the SFS & PCA dataset for applying Boosting.

The accuracy for XGBoost on SFS is: **87.18%**
The F1_Score for XGBoost on SFS is: **0.92**
The accuracy for XGBoost on PCA is: **64.1%**

Here, we see that with Boosting, we obtain a very high F1 score as well as a high accuracy with the SFS implemented data. Thus, XGBoost performs very well with SFS on the given dataset.

K-Nearest Neighbors

Here, we implemented K-Nearest Neighbors on the SFS & PCA dataset & analyzed it’s performance:-

The accuracy of KNN with SFS is: **87.18%**
The accuracy of KNN with PCA is: **97.44%**
The F1 score of KNN with PCA is: **0.983050847457627**

We note here that KNN performs very well in the given dataset especially with PCA because of the low intra-class distances that we have observed from the 3-D plot of the PCA implemented data points. KNN is also a good method in this case as it is a rare disease and we have very little training data that needs to be carried with the learnt model hence not much memory is needed. We also obtain a high f1 score of **0.98** in this case which is very good considering it is a disease classification task.

SVM

Now, we implemented SVM on the SFS & PCA dataset with different kernels. On analyzing their performance:-

The accuracy for Kernel linear SVM with PCA is: **82.05%**
The accuracy for Kernel sigmoid SVM with PCA is: **69.23%**
The accuracy for Kernel poly SVM with PCA is: **84.62%**
The accuracy for Kernel rbf SVM with PCA is: **82.05%**
The accuracy for Kernel linear SVM with SFS is: **74.36%**
The accuracy for Kernel sigmoid SVM with SFS is: **71.7%**
The accuracy for Kernel poly SVM with SFS is: **82.05%**
The accuracy for Kernel rbf SVM with SFS is: **82.05%**

As we can see from above, none of the SVM kernels are able to provide as good a classification accuracy either with PCA or SFS.

Multi-Layer Perceptron

We are applying MLP on the SFS & PCA dataset because of its versatility due to the fact that a 3 layered MLP can approximate any arbitrary continuous decision boundary. On analyzing both of the models' performance we've:-

The accuracy for MLP on SFS data is: **74.36%**
The F1_Score for MLP on SFS data is: **0.834**

The accuracy for MLP on PCA data is: **89.74%**
The F1_Score for MLP on PCA data is: **0.934**

We are unable to obtain a very high accuracy with MLP because of the fact that there are not many points in the dataset which can be used for training.

REFERENCES

1) www.wikipedia.com
2) Pattern Classification Second Edition by Duda et. al.
3) Lecture Slides, CSL:2050 Spring Term 2023, Dr. Richa Singh, IIT Jodhpur