**R PROJECTS**

**Problem Statement:-**

The main aim of the data is to discriminate healthy people from those with PD, according to "status" column which is set to 0 for healthy and 1 for PD.The data is used to separate healthy people from people with Parkinson’s disease.(classification)

**Data Collection:-**

The data is in ASCII CSV format. The rows of the CSV file contain an instance corresponding to one voice recording. There are around six

recordings per patient, the name of the patient is identified in the first column

Parkinson is a nervous system disorder that affects movement. The dataset contains 195 records of people with 23 different attributes which contain biomedical measurements.

Source:

The dataset was created by Max Little of the University of Oxford, in collaboration with the National Centre for Voice and Speech, Denver, Colorado, who recorded the speech signals. The original study published the feature extraction methods for general voice disorders.

Attribute Information:

Matrix column entries (attributes):

name - ASCII subject name and recording number

MDVP:Fo(Hz) - Average vocal fundamental frequency

MDVP:Fhi(Hz) - Maximum vocal fundamental frequency

MDVP:Flo(Hz) - Minimum vocal fundamental frequency

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 ratio of noise to tonal components in the voice

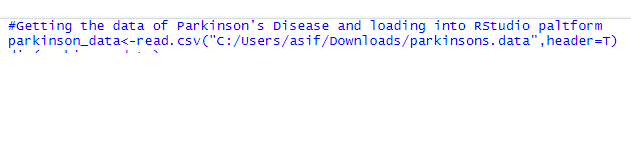
status - Health status of the subject (one) - Parkinson's, (zero) - healthy

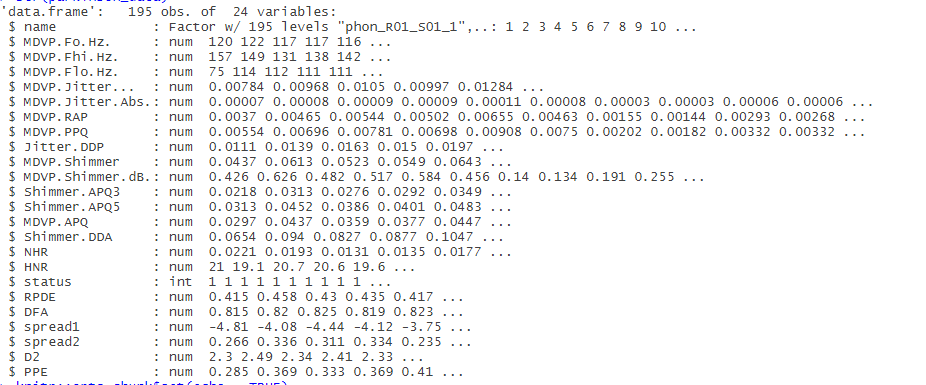
RPDE,D2 - Two nonlinear dynamical complexity measures

DFA - Signal fractal scaling exponent

spread1,spread2,PPE - Three nonlinear measures of fundamental frequency variation

Loading the data into RStudio platform:-





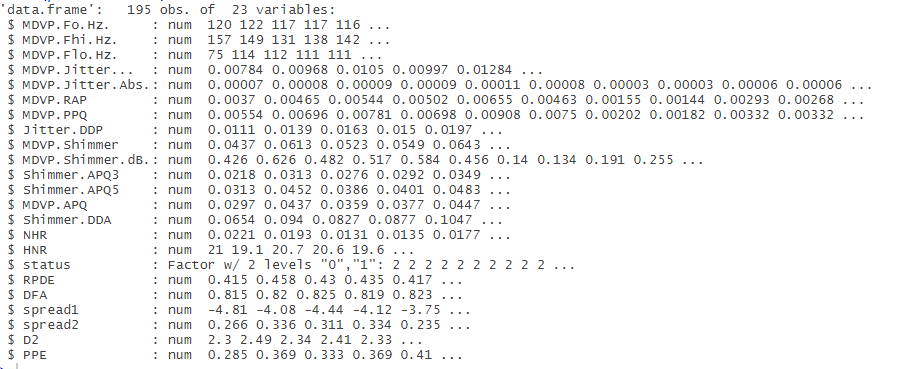
**Data Transformation:-**

As The dependent variable is status(categorical variable) and we want to classify the status as healthy people and people with Parkinson's diseases.So we need to convert that our status column from num into factor



**Exploratory Data Analysis:-**

As we explore our data and analysis it we see that there are almost all numeric varaible except status which is categorical variable so as per our study the model which will best suite our data at the moment is KNN Algorithm because it support numeric variable very efficiently but we will try to use also other models to evaluate our process.



**Model Building:-**

Now the data is cleaned and prepared,So it's Time to build the model.We will split our model into training and testing data because it should get all variety of data and it should be unbiased and accurate

Code:-

#Getting the data of Parkinson's Disease and loading into RStudio paltform

parkinson\_data<-read.csv("C:/Users/asif/Downloads/parkinsons.data",header=T)

#checking the dimensions of the dataset

dim(parkinson\_data)

#checking structure of the data

str(parkinson\_data)

#Checking if the dataset has any missing values

colSums(is.na(parkinson\_data))

summary(parkinson\_data)

summary(parkinson\_data$status)

# Clenaing the data: Data Collection

parkinson\_data$name<-NULL

parkinson\_data$status<-as.factor(parkinson\_data$status)

parkinson\_data1<-parkinson\_data[,-17]

str(parkinson\_data1)

# model building and dividing it into training and testing data

set.seed(123)

index3<-sample(nrow(parkinson\_data1),0.75\*nrow(parkinson\_data1))

training\_parkinson<-parkinson\_data1[index3,]

testing\_parkinson<-parkinson\_data1[-index3,]

dim(training\_parkinson)

dim(testing\_parkinson)

ytrain<-parkinson\_data$status[index3] # vector of only dependent variables

ytest<-parkinson\_data$status[-index3]

length(ytest)

length(ytrain)

library(class)

#Now the data is cleaned and prepared for model building,so it is time to choose the best ML Algorithms

#for the dataset

#As the dataset has almost numeric so it is advisable to go for KNN Algorithms as it support numeric value

#Here the depedent variable is status[0 for healthy and 1 for Parkison's Disease]

knn\_parkinson\_model<-knn(training\_parkinson,testing\_parkinson,k=sqrt(nrow(training\_parkinson)),cl=ytrain)

table(predict=knn\_parkinson\_model,actual=ytest)

(6+34)/49 # accuracy

34/(34+4) # sensitivity

6/(6+5) # specificity

# Decision Trees

library(rpart)

library(rpart.plot)

#Building a Decision Trees

parkison\_tree<-rpart(status~.,data=parkinson\_data)

plot.new()

rpart.plot(parkison\_tree)

parkinson\_pred<-predict(parkison\_tree,parkinson\_data,type="class")

table(predicted=parkinson\_pred,actual=parkinson\_data$status)

(141+38)/195 # accuracy

141/(141+6) # sensitivity

38/(38+10) # specificity

**Output:-**

|  |
| --- |
| > #Getting the data of Parkinson's Disease and loading into RStudio paltform  > parkinson\_data<-read.csv("C:/Users/asif/Downloads/parkinsons.data",header=T)  > #checking the dimensions of the dataset  > dim(parkinson\_data)  [1] 195 24  > #checking structure of the data  > str(parkinson\_data)  'data.frame': 195 obs. of 24 variables:  $ name : Factor w/ 195 levels "phon\_R01\_S01\_1",..: 1 2 3 4 5 6 7 8 9 10 ...  $ MDVP.Fo.Hz. : num 120 122 117 117 116 ...  $ MDVP.Fhi.Hz. : num 157 149 131 138 142 ...  $ MDVP.Flo.Hz. : num 75 114 112 111 111 ...  $ MDVP.Jitter... : num 0.00784 0.00968 0.0105 0.00997 0.01284 ...  $ MDVP.Jitter.Abs.: num 0.00007 0.00008 0.00009 0.00009 0.00011 0.00008 0.00003 0.00003 0.00006 0.00006 ...  $ MDVP.RAP : num 0.0037 0.00465 0.00544 0.00502 0.00655 0.00463 0.00155 0.00144 0.00293 0.00268 ...  $ MDVP.PPQ : num 0.00554 0.00696 0.00781 0.00698 0.00908 0.0075 0.00202 0.00182 0.00332 0.00332 ...  $ Jitter.DDP : num 0.0111 0.0139 0.0163 0.015 0.0197 ...  $ MDVP.Shimmer : num 0.0437 0.0613 0.0523 0.0549 0.0643 ...  $ MDVP.Shimmer.dB.: num 0.426 0.626 0.482 0.517 0.584 0.456 0.14 0.134 0.191 0.255 ...  $ Shimmer.APQ3 : num 0.0218 0.0313 0.0276 0.0292 0.0349 ...  $ Shimmer.APQ5 : num 0.0313 0.0452 0.0386 0.0401 0.0483 ...  $ MDVP.APQ : num 0.0297 0.0437 0.0359 0.0377 0.0447 ...  $ Shimmer.DDA : num 0.0654 0.094 0.0827 0.0877 0.1047 ...  $ NHR : num 0.0221 0.0193 0.0131 0.0135 0.0177 ...  $ HNR : num 21 19.1 20.7 20.6 19.6 ...  $ status : int 1 1 1 1 1 1 1 1 1 1 ...  $ RPDE : num 0.415 0.458 0.43 0.435 0.417 ...  $ DFA : num 0.815 0.82 0.825 0.819 0.823 ...  $ spread1 : num -4.81 -4.08 -4.44 -4.12 -3.75 ...  $ spread2 : num 0.266 0.336 0.311 0.334 0.235 ...  $ D2 : num 2.3 2.49 2.34 2.41 2.33 ...  $ PPE : num 0.285 0.369 0.333 0.369 0.41 ...  > #Checking if the dataset has any missing values  > colSums(is.na(parkinson\_data))  name MDVP.Fo.Hz. MDVP.Fhi.Hz.  0 0 0  MDVP.Flo.Hz. MDVP.Jitter... MDVP.Jitter.Abs.  0 0 0  MDVP.RAP MDVP.PPQ Jitter.DDP  0 0 0  MDVP.Shimmer MDVP.Shimmer.dB. Shimmer.APQ3  0 0 0  Shimmer.APQ5 MDVP.APQ Shimmer.DDA  0 0 0  NHR HNR status  0 0 0  RPDE DFA spread1  0 0 0  spread2 D2 PPE  0 0 0  > summary(parkinson\_data)  name MDVP.Fo.Hz.  phon\_R01\_S01\_1: 1 Min. : 88.33  phon\_R01\_S01\_2: 1 1st Qu.:117.57  phon\_R01\_S01\_3: 1 Median :148.79  phon\_R01\_S01\_4: 1 Mean :154.23  phon\_R01\_S01\_5: 1 3rd Qu.:182.77  phon\_R01\_S01\_6: 1 Max. :260.11  (Other) :189  MDVP.Fhi.Hz. MDVP.Flo.Hz.  Min. :102.1 Min. : 65.48  1st Qu.:134.9 1st Qu.: 84.29  Median :175.8 Median :104.31  Mean :197.1 Mean :116.32  3rd Qu.:224.2 3rd Qu.:140.02  Max. :592.0 Max. :239.17    MDVP.Jitter... MDVP.Jitter.Abs.  Min. :0.001680 Min. :7.000e-06  1st Qu.:0.003460 1st Qu.:2.000e-05  Median :0.004940 Median :3.000e-05  Mean :0.006220 Mean :4.396e-05  3rd Qu.:0.007365 3rd Qu.:6.000e-05  Max. :0.033160 Max. :2.600e-04    MDVP.RAP MDVP.PPQ  Min. :0.000680 Min. :0.000920  1st Qu.:0.001660 1st Qu.:0.001860  Median :0.002500 Median :0.002690  Mean :0.003306 Mean :0.003446  3rd Qu.:0.003835 3rd Qu.:0.003955  Max. :0.021440 Max. :0.019580    Jitter.DDP MDVP.Shimmer  Min. :0.002040 Min. :0.00954  1st Qu.:0.004985 1st Qu.:0.01650  Median :0.007490 Median :0.02297  Mean :0.009920 Mean :0.02971  3rd Qu.:0.011505 3rd Qu.:0.03789  Max. :0.064330 Max. :0.11908    MDVP.Shimmer.dB. Shimmer.APQ3  Min. :0.0850 Min. :0.004550  1st Qu.:0.1485 1st Qu.:0.008245  Median :0.2210 Median :0.012790  Mean :0.2823 Mean :0.015664  3rd Qu.:0.3500 3rd Qu.:0.020265  Max. :1.3020 Max. :0.056470    Shimmer.APQ5 MDVP.APQ  Min. :0.00570 Min. :0.00719  1st Qu.:0.00958 1st Qu.:0.01308  Median :0.01347 Median :0.01826  Mean :0.01788 Mean :0.02408  3rd Qu.:0.02238 3rd Qu.:0.02940  Max. :0.07940 Max. :0.13778    Shimmer.DDA NHR  Min. :0.01364 Min. :0.000650  1st Qu.:0.02474 1st Qu.:0.005925  Median :0.03836 Median :0.011660  Mean :0.04699 Mean :0.024847  3rd Qu.:0.06080 3rd Qu.:0.025640  Max. :0.16942 Max. :0.314820    HNR status  Min. : 8.441 Min. :0.0000  1st Qu.:19.198 1st Qu.:1.0000  Median :22.085 Median :1.0000  Mean :21.886 Mean :0.7538  3rd Qu.:25.076 3rd Qu.:1.0000  Max. :33.047 Max. :1.0000    RPDE DFA  Min. :0.2566 Min. :0.5743  1st Qu.:0.4213 1st Qu.:0.6748  Median :0.4960 Median :0.7223  Mean :0.4985 Mean :0.7181  3rd Qu.:0.5876 3rd Qu.:0.7619  Max. :0.6852 Max. :0.8253    spread1 spread2  Min. :-7.965 Min. :0.006274  1st Qu.:-6.450 1st Qu.:0.174350  Median :-5.721 Median :0.218885  Mean :-5.684 Mean :0.226510  3rd Qu.:-5.046 3rd Qu.:0.279234  Max. :-2.434 Max. :0.450493    D2 PPE  Min. :1.423 Min. :0.04454  1st Qu.:2.099 1st Qu.:0.13745  Median :2.362 Median :0.19405  Mean :2.382 Mean :0.20655  3rd Qu.:2.636 3rd Qu.:0.25298  Max. :3.671 Max. :0.52737    > summary(parkinson\_data$status)  Min. 1st Qu. Median Mean 3rd Qu. Max.  0.0000 1.0000 1.0000 0.7538 1.0000 1.0000  > # Clenaing the data: Data Collection  > parkinson\_data$name<-NULL  > parkinson\_data$status<-as.factor(parkinson\_data$status)  > parkinson\_data1<-parkinson\_data[,-17]  > str(parkinson\_data1)  'data.frame': 195 obs. of 22 variables:  $ MDVP.Fo.Hz. : num 120 122 117 117 116 ...  $ MDVP.Fhi.Hz. : num 157 149 131 138 142 ...  $ MDVP.Flo.Hz. : num 75 114 112 111 111 ...  $ MDVP.Jitter... : num 0.00784 0.00968 0.0105 0.00997 0.01284 ...  $ MDVP.Jitter.Abs.: num 0.00007 0.00008 0.00009 0.00009 0.00011 0.00008 0.00003 0.00003 0.00006 0.00006 ...  $ MDVP.RAP : num 0.0037 0.00465 0.00544 0.00502 0.00655 0.00463 0.00155 0.00144 0.00293 0.00268 ...  $ MDVP.PPQ : num 0.00554 0.00696 0.00781 0.00698 0.00908 0.0075 0.00202 0.00182 0.00332 0.00332 ...  $ Jitter.DDP : num 0.0111 0.0139 0.0163 0.015 0.0197 ...  $ MDVP.Shimmer : num 0.0437 0.0613 0.0523 0.0549 0.0643 ...  $ MDVP.Shimmer.dB.: num 0.426 0.626 0.482 0.517 0.584 0.456 0.14 0.134 0.191 0.255 ...  $ Shimmer.APQ3 : num 0.0218 0.0313 0.0276 0.0292 0.0349 ...  $ Shimmer.APQ5 : num 0.0313 0.0452 0.0386 0.0401 0.0483 ...  $ MDVP.APQ : num 0.0297 0.0437 0.0359 0.0377 0.0447 ...  $ Shimmer.DDA : num 0.0654 0.094 0.0827 0.0877 0.1047 ...  $ NHR : num 0.0221 0.0193 0.0131 0.0135 0.0177 ...  $ HNR : num 21 19.1 20.7 20.6 19.6 ...  $ RPDE : num 0.415 0.458 0.43 0.435 0.417 ...  $ DFA : num 0.815 0.82 0.825 0.819 0.823 ...  $ spread1 : num -4.81 -4.08 -4.44 -4.12 -3.75 ...  $ spread2 : num 0.266 0.336 0.311 0.334 0.235 ...  $ D2 : num 2.3 2.49 2.34 2.41 2.33 ...  $ PPE : num 0.285 0.369 0.333 0.369 0.41 ...  > # model building and dividing it into training and testing data  > set.seed(123)  > index3<-sample(nrow(parkinson\_data1),0.75\*nrow(parkinson\_data1))  > training\_parkinson<-parkinson\_data1[index3,]  > testing\_parkinson<-parkinson\_data1[-index3,]  > dim(training\_parkinson)  [1] 146 22  > dim(testing\_parkinson)  [1] 49 22  > ytrain<-parkinson\_data$status[index3] # vector of only dependent variables  > ytest<-parkinson\_data$status[-index3]  > length(ytest)  [1] 49  > length(ytrain)  [1] 146  > library(class)  > #Now the data is cleaned and prepared for model building,so it is time to choose the best ML Algorithms  > #for the dataset  > #As the dataset has almost numeric so it is advisable to go for KNN Algorithms as it support numeric value  > #Here the depedent variable is status[0 for healthy and 1 for Parkison's Disease]  > knn\_parkinson\_model<-knn(training\_parkinson,testing\_parkinson,k=sqrt(nrow(training\_parkinson)),cl=ytrain)  > table(predict=knn\_parkinson\_model,actual=ytest)  actual  predict 0 1  0 6 4  1 5 34  > (6+34)/49 # accuracy  [1] 0.8163265  > 34/(34+4) # sensitivity  [1] 0.8947368  > 6/(6+5) # specificity  [1] 0.5454545 |
|  |
| |  | | --- | | > | |

Conclusion:-

We have run KNN Algorithms and Decision Trees both but The Decision Trees perform better than KNN Algorithm .