USE CASE STUDY REPORT

Group No.: Group 02

Student Names: Gaurav Tople, Shaifali Juneja

Executive Summary:

This case study aims at determining the gender through human voice data. We worked on the voice data and predicted through our model whether it's a male voice or a female. We obtained the dataset from Data.world blog. A database was built using thousands of samples of male and female voices consisting of 3168 rows and 21 columns, each labeled by their gender of male or female. We performed PCA on 20 of our predictors and from the 20 principal components, we chose first 10 to build and train our model. We leveraged 5 different algorithms: KNN, Logistic Regression, Random forest, SVM and Neural Networks. The SVM has the highest accuracy- 97.68% and Random forest has the lowest-96.53%. On the basis of the accuracy and the computation time, we recommend the SVM model.

I. Background and Introduction

Determining a person's gender as male or female, based upon a sample of their voice seems to initially be an easy task. Often, the human ear can easily detect the difference between a male or female voice within the first few spoken words. However, designing a computer program to do this turns out to be a bit trickier.

As the significance of the computers in our daily life is getting popular, the interaction between human and machine is becoming more important day by day. The desire of humans to communicate with machines in a natural way has led to the evolution of natural language processing. As the advancements in this field are hap- penning, it is likely that voice interaction systems will replace the standard keyboards in near the future. Today if we look in the technology market around us, we have some real state of the art technologies like Microsoft Kinect and Google Assistant which performs really well. But every speech system that is available today has its own drawbacks and continuous work is being done to increase the performance of such systems. To increase the performance of speech systems pre-processing like gender recognition and language identification is required.

This study focuses on gender identification system using speech. Identification of gender using the speech of the speaker concerns in detecting that the spoken speech is of male or female speaker. Here we describe the design of a computer program to model acoustic analysis of voices and speech for determining gender. The model is constructed using 3,168 recorded samples of male and female voices, speech, and utterances. The samples are processed using acoustic analysis and then applied to an artificial intelligence/machine learning algorithm to learn gender-specific traits.

Gender identification is one of the significant problems in speech analysis today. Tracing the gender from acoustic data i.e., pitch, median, frequency etc. Machine learning gives promising results for classification problem in all the research domains. There are several performance metrics to evaluate algorithms of an area. Our Comparative model algorithm for evaluating 5 different machine learning algorithms based on eight different metrics in gender classification from acoustic data. Agenda is to identify gender, with five different algorithms: Logistic Regression, K-Nearest Neighbor (KNN), Random Forest (RF), Support Vector Machine (SVM), and Neural Network on basis of eight different metrics. The main parameter in evaluating any algorithms is its performance.

Misclassification rate must be less in classification problems, which says that the accuracy rate must be high. Here with this comparative model algorithm, we are trying to assess the different ML algorithms and find the best fit for gender classification of acoustic data.

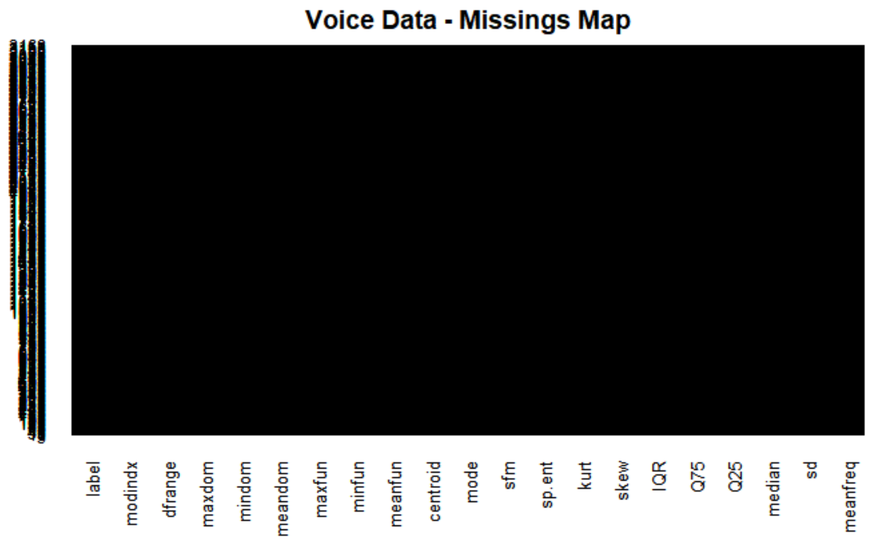
The output from the pre-processed WAV files were saved into a CSV file, containing 3168 rows and 21 columns (20 columns for each feature and one label column for the classification of male or female). Following is the snippet of how the data looks.

```
##     meanfreq         sd     median         Q25        Q75        IQR
## 1 0.05978098 0.06424127 0.03202691 0.015071489 0.09019344 0.07512195
## 2 0.06600874 0.06731003 0.04022873 0.019413867 0.09266619 0.07325232
## 3 0.07731550 0.08382942 0.03671846 0.008701057 0.13190802 0.12320696
## 4 0.15122809 0.07211059 0.15801119 0.096581728 0.20795525 0.11137352
## 5 0.13512039 0.07914610 0.12465623 0.078720218 0.20604493 0.12732471
## 6 0.13278641 0.07955687 0.11908985 0.067957993 0.20959160 0.14163361
##        skew        kurt    sp.ent       sfm       mode   centroid
## 1 12.863462  274.402906 0.8933694 0.4919178 0.00000000 0.05978098
## 2 22.423285  634.613855 0.8921932 0.5137238 0.00000000 0.06600874
## 3 30.757155 1024.927705 0.8463891 0.4789050 0.00000000 0.07731550
## 4  1.232831    4.177296 0.9633225 0.7272318 0.08387819 0.15122809
## 5  1.101174    4.333713 0.9719551 0.7835681 0.10426140 0.13512039
## 6  1.932562    8.308895 0.9631813 0.7383070 0.11255543 0.13278641
##      meanfun     minfun    maxfun     meandom    mindom    maxdom
## 1 0.08427911 0.01570167 0.2758621 0.007812500 0.0078125 0.0078125
## 2 0.10793655 0.01582591 0.2500000 0.009014423 0.0078125 0.0546875
## 3 0.09870626 0.01565558 0.2711864 0.007990057 0.0078125 0.0156250
## 4 0.08896485 0.01779755 0.2500000 0.201497396 0.0078125 0.5625000
## 5 0.10639784 0.01693122 0.2666667 0.712812500 0.0078125 5.4843750
## 6 0.11013192 0.01711230 0.2539683 0.298221983 0.0078125 2.7265625
##     dfrange    modindx label
## 1 0.0000000 0.00000000  male
## 2 0.0468750 0.05263158  male
## 3 0.0078125 0.04651163  male
## 4 0.5546875 0.24711908  male
## 5 5.4765625 0.20827389  male
## 6 2.7187500 0.12515964  male
```

II. Data Exploration and Visualization

The first step in this process is to check if there are any missing values in the data. The following command gives us the required result. It can be seen that there are no missing values in the data.

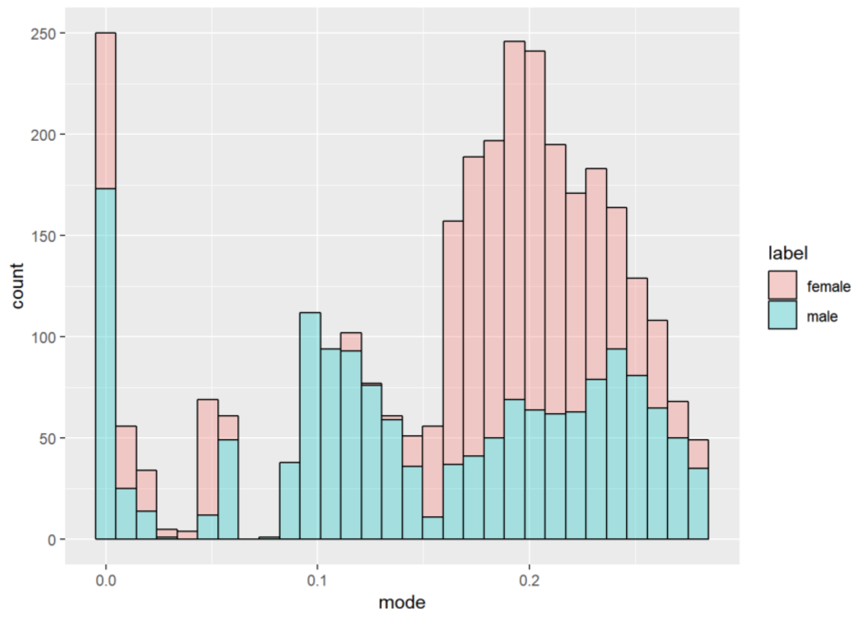
```
any(is.na(voice.df))
## [1] FALSE
```

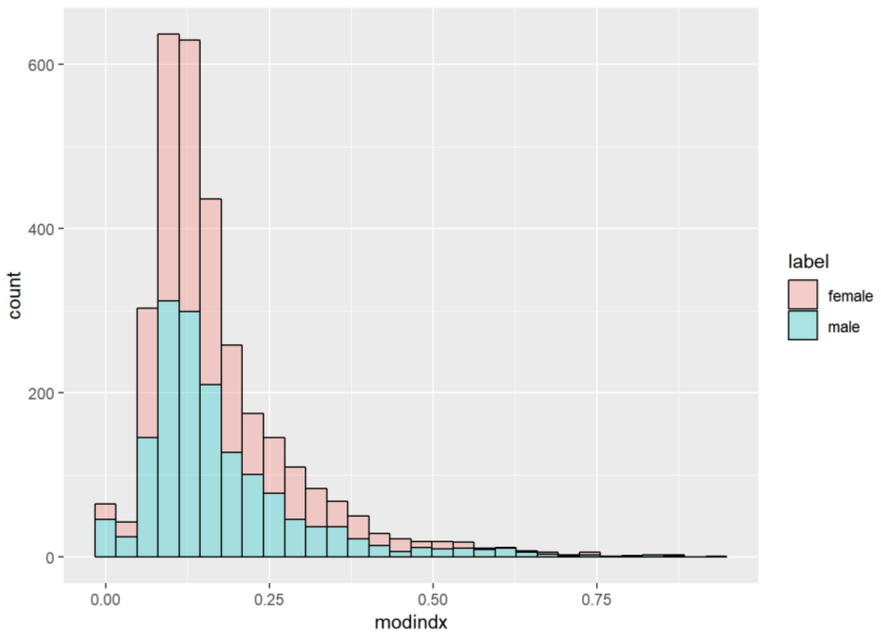


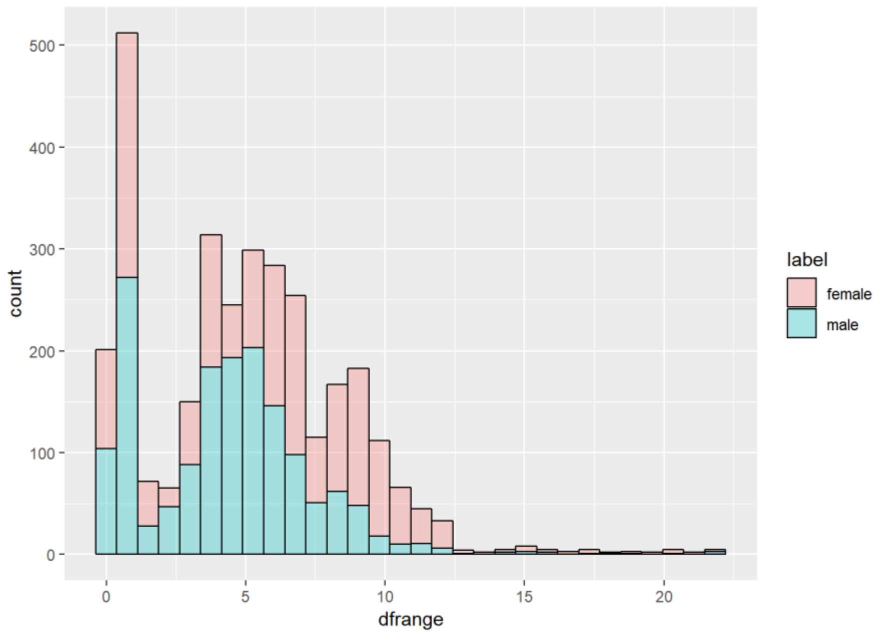
The next step is to find the summary of data. Out of 20 acoustic variables, the summary of the following three variables has a minimum value of 0. Hence, these variables require further investigation.

```
     dfrange             modindx             mode
 Min.    : 0.000     Min.    :0.00000    Min.    :0.0000
 1st Qu.: 2.045      1st Qu.:0.09977     1st Qu.:0.1180
 Median : 4.945      Median :0.13936     Median :0.1866
 Mean   : 4.995      Mean    :0.17375    Mean    :0.1653
 3rd Qu.: 6.992      3rd Qu.:0.20918     3rd Qu.:0.2211
 Max.   :21.844      Max.    :0.93237    Max.    :0.2800
```

We plot a histogram for each of the above variables to find its distribution. It can be depicted from the histograms that the distribution of dfrange and modindx does not seem unusual. However, the mode has a distribution which is not usual in nature as the count of 0 values in this variable is surprisingly larger than the other values. Hence, these 0 values require imputation with the gender-wise mean of the variable.



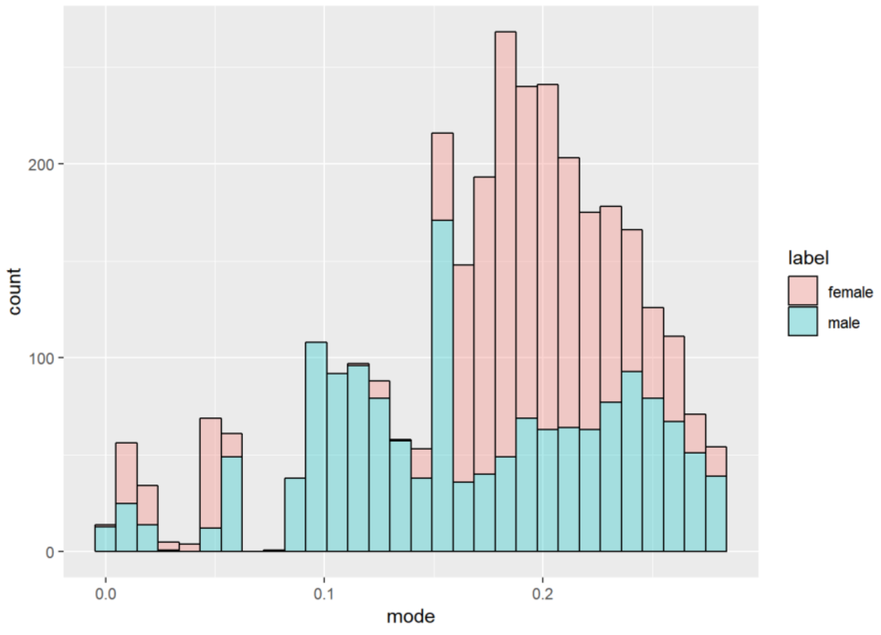




As discussed above, the 0 values will be imputed with the mean by running the following commands in R.

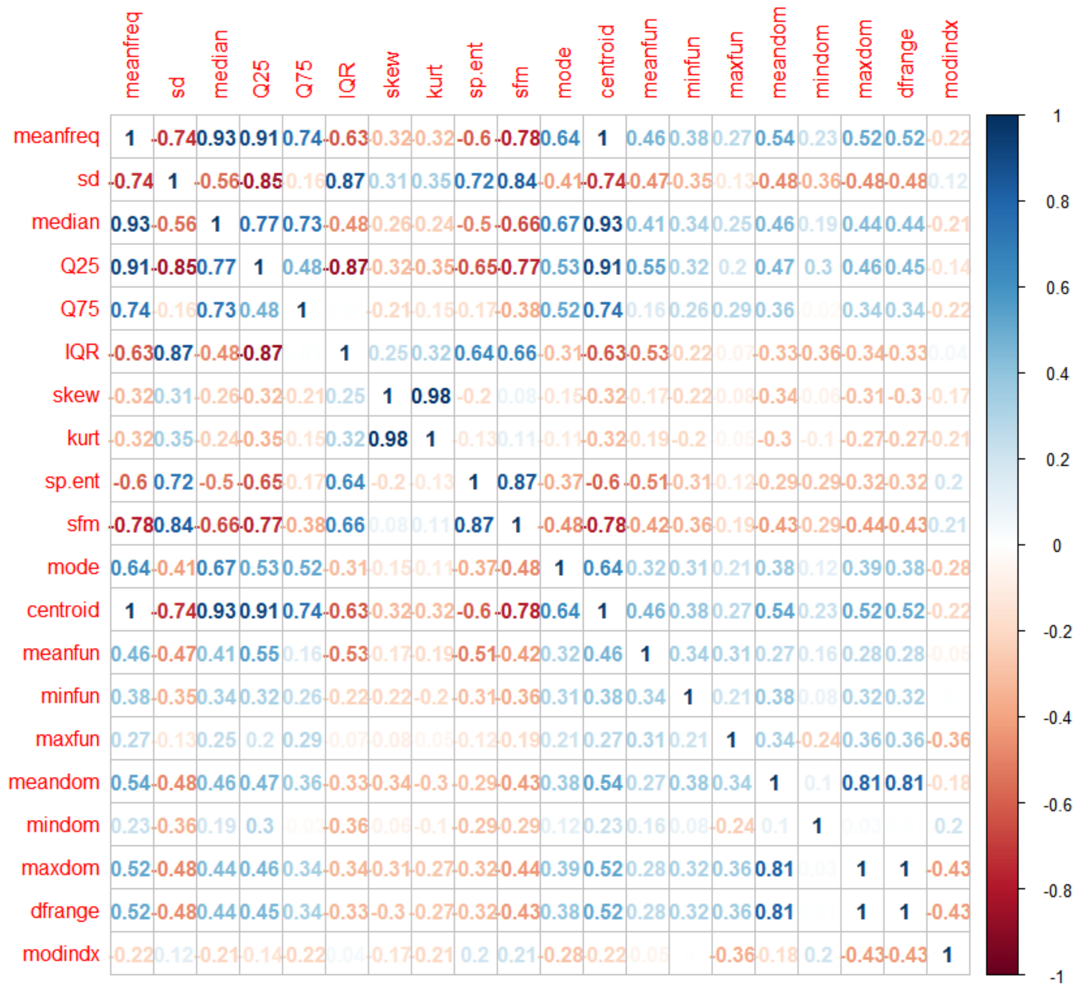
```
smf <- summarise(group_by(voice.df,label),mean(mode))

voice.df[which(voice.df$mode == 0 & voice.df$label == "male"), "mode"]
<- smf$`mean(mode)`[2]

voice.df[which(voice.df$mode == 0 & voice.df$label == "female"),
"mode"] <- smf$`mean(mode)`[1]
```

The distribution of the mode variable after the mean imputation is performed.



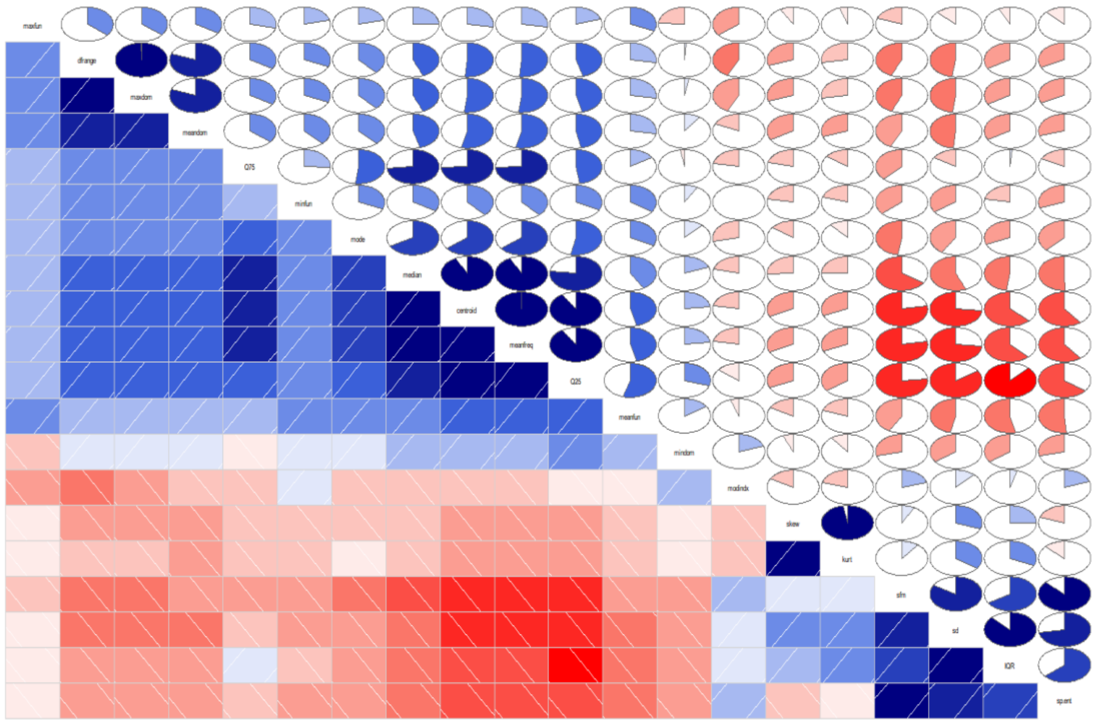
III. Data Preparation and Preprocessing

Now that we have performed Exploratory Data Analysis (EDA) on this dataset by checking if the dataset has any missing values and by performing mean imputation on the values which are unusual when compared with its distribution, the next step would be to check the correlation between each of the 20 acoustic feature variables. From the below correlation plots it can be observed that there are many features which are highly correlated with each other. We are considering a feature highly correlated with another feature when the correlation between them has a value greater than 0.80 or less than -0.80.

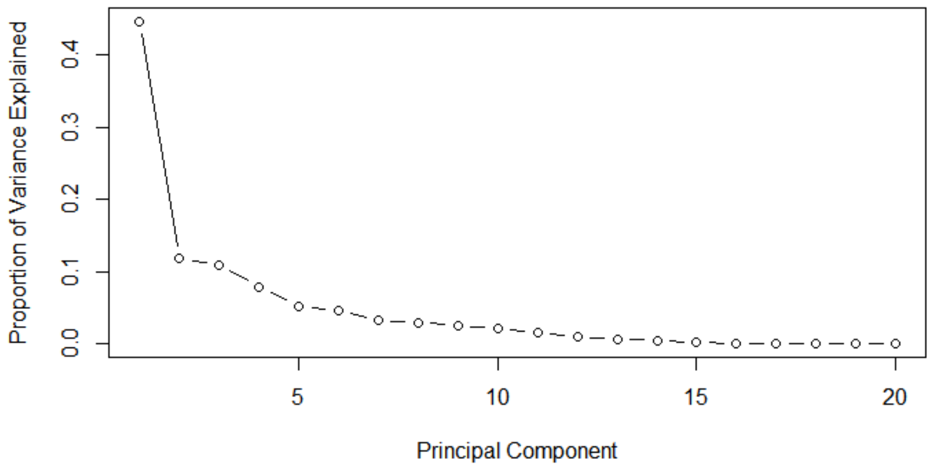


	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	sp.ent	sfm	mode	centroid	meanfun	minfun	maxfun	meandom	mindom	maxdom	dfrange	modindx
meanfreq	1	-0.74	0.93	0.91	0.74	-0.63	-0.32	-0.32	-0.6	-0.78	0.64	1	0.46	0.38	0.27	0.54	0.23	0.52	0.52	-0.22
sd	-0.74	1	-0.56	-0.85	-0.16	0.87	0.31	0.35	0.72	0.84	-0.41	-0.74	-0.47	-0.35	-0.13	-0.48	-0.36	-0.48	-0.48	0.12
median	0.93	-0.56	1	0.77	0.73	-0.48	-0.26	-0.24	-0.5	-0.66	0.67	0.93	0.41	0.34	0.25	0.46	0.19	0.44	0.44	-0.21
Q25	0.91	-0.85	0.77	1	0.48	-0.87	-0.32	-0.35	-0.65	-0.77	0.53	0.91	0.55	0.32	0.2	0.47	0.3	0.46	0.45	-0.14
Q75	0.74	-0.16	0.73	0.48	1	[illegible]	-0.21	-0.15	-0.17	-0.38	0.52	0.74	0.16	0.26	0.29	0.36	-0.02	0.34	0.34	-0.22
IQR	-0.63	0.87	-0.48	-0.87	[illegible]	1	0.25	0.32	0.64	0.66	-0.31	-0.63	-0.53	-0.22	-0.07	-0.33	-0.36	-0.34	-0.33	0.04
skew	-0.32	0.31	-0.26	-0.32	-0.21	0.25	1	0.98	-0.2	0.08	-0.15	-0.32	-0.17	-0.22	-0.08	-0.34	-0.06	-0.31	-0.3	-0.17
kurt	-0.32	0.35	-0.24	-0.35	-0.15	0.32	0.98	1	-0.13	0.11	-0.11	-0.32	-0.19	-0.2	-0.05	-0.3	-0.1	-0.27	-0.27	-0.21
sp.ent	-0.6	0.72	-0.5	-0.65	-0.17	0.64	-0.2	-0.13	1	0.87	-0.37	-0.6	-0.51	-0.31	-0.12	-0.29	-0.29	-0.32	-0.32	0.2
sfm	-0.78	0.84	-0.66	-0.77	-0.38	0.66	0.08	0.11	0.87	1	-0.48	-0.78	-0.42	-0.36	-0.19	-0.43	-0.29	-0.44	-0.43	0.21
mode	0.64	-0.41	0.67	0.53	0.52	-0.31	-0.15	-0.11	-0.37	-0.48	1	0.64	0.32	0.31	0.21	0.38	0.12	0.39	0.38	-0.28
centroid	1	-0.74	0.93	0.91	0.74	-0.63	-0.32	-0.32	-0.6	-0.78	0.64	1	0.46	0.38	0.27	0.54	0.23	0.52	0.52	-0.22
meanfun	0.46	-0.47	0.41	0.55	0.16	-0.53	-0.17	-0.19	-0.51	-0.42	0.32	0.46	1	0.34	0.31	0.27	0.16	0.28	0.28	-0.05
minfun	0.38	-0.35	0.34	0.32	0.26	-0.22	-0.22	-0.2	-0.31	-0.36	0.31	0.38	0.34	1	0.21	0.38	0.08	0.32	0.32	0
maxfun	0.27	-0.13	0.25	0.2	0.29	-0.07	-0.08	-0.05	-0.12	-0.19	0.21	0.27	0.31	0.21	1	0.34	-0.24	0.36	0.36	-0.36
meandom	0.54	-0.48	0.46	0.47	0.36	-0.33	-0.34	-0.3	-0.29	-0.43	0.38	0.54	0.27	0.38	0.34	1	0.1	0.81	0.81	-0.18
mindom	0.23	-0.36	0.19	0.3	-0.02	-0.36	-0.06	-0.1	-0.29	-0.29	0.12	0.23	0.16	0.08	-0.24	0.1	1	0.03	[illegible]	0.2
maxdom	0.52	-0.48	0.44	0.46	0.34	-0.34	-0.31	-0.27	-0.32	-0.44	0.39	0.52	0.28	0.32	0.36	0.81	0.03	1	1	-0.43
dfrange	0.52	-0.48	0.44	0.45	0.34	-0.33	-0.3	-0.27	-0.32	-0.43	0.38	0.52	0.28	0.32	0.36	0.81	[illegible]	1	1	-0.43
modindx	-0.22	0.12	-0.21	-0.14	-0.22	0.04	-0.17	-0.21	0.2	0.21	-0.28	-0.22	-0.05	0	-0.36	-0.18	0.2	-0.43	-0.43	1

The high correlation between the features may hurt the interpretability of the model. Hence, it is necessary to deal with such features.



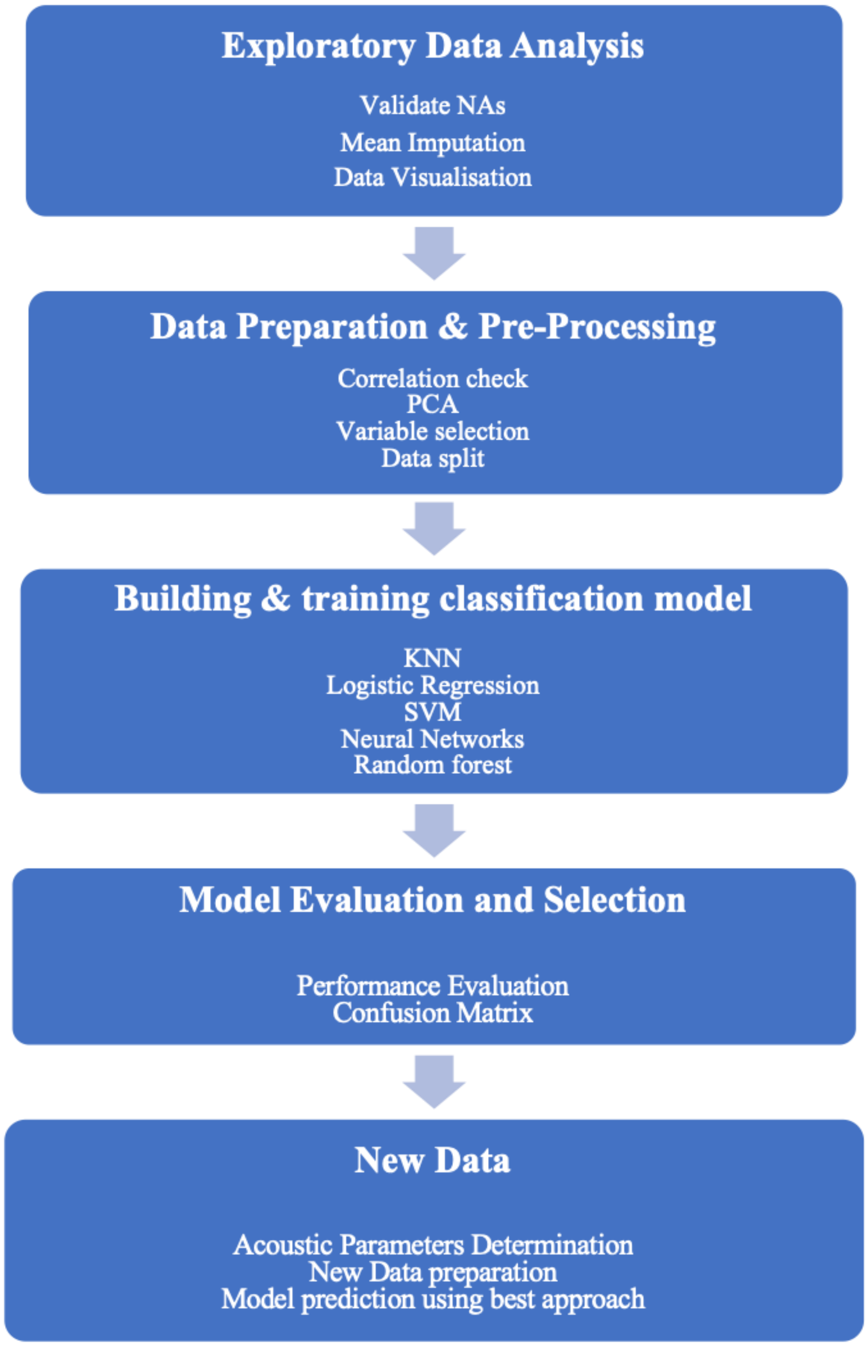
To deal with this issue it is required to perform the Principal Component Analysis (PCA) on the data and then choose the significant component scores that are required to build and train a model. Following is the plot which shows the proportionality of variance for all the 20 principal components generated. Out of 20 components we are choosing the first 10 components to build and train the model as it gives the highest accuracy for training and validation data.



The next step would be to split the dataset into training and validation data. We have split the data into 7:3 ratio, that means we have randomly assigned 70% of data for training and 30% of data for validation.

IV. Data Mining Techniques and Implementation

We worked on five different approach for this classification problem. The final required outcome is a variable named a "Label". The "Label" variable is set to 1 if the predicted gender is male and 0 if the predicted gender is a female.

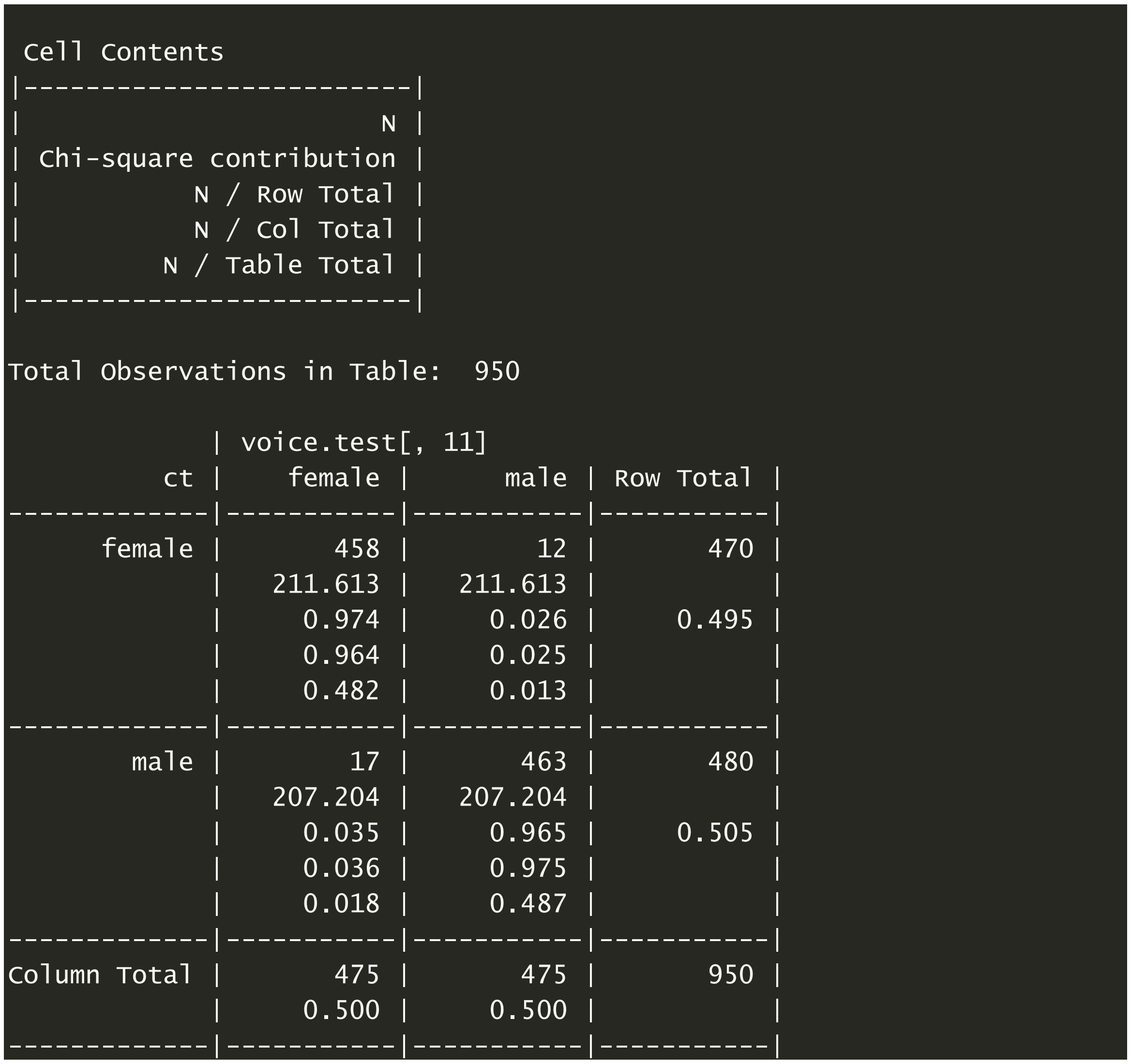


V. Performance Evaluation

Based on the consideration of accuracy and the time of execution, we picked Support Vector Machine model to be our best approach.

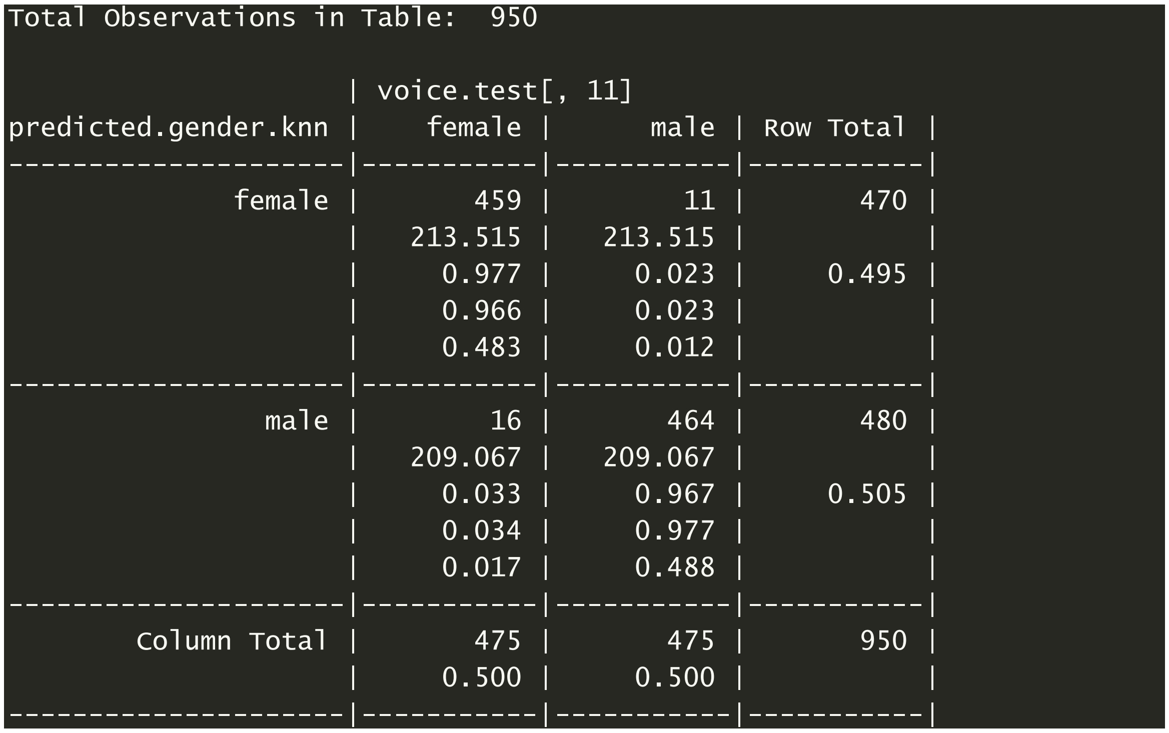
Data Mining Techniques	**Model Accuracy**	**Computation Time**
Logistic Regression	96.95%	0.49 second
KNN	97.16%	8.78 seconds
Random Forest	96.53%	2.28 seconds
SVM	97.68%	0.47 second
Neural Network	97.37%	0.56 second

Classification Matrix for Logistic Regression on Validation Data:



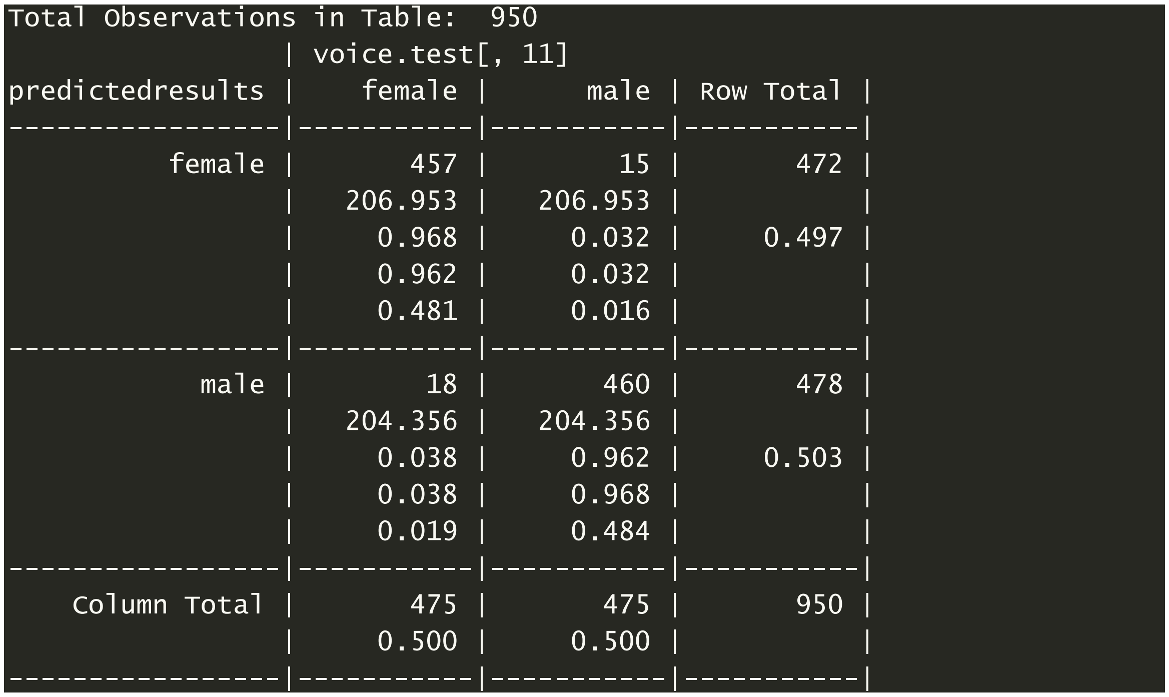
```
 Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  950

             | voice.test[, 11]
          ct |    female |      male | Row Total |
-------------|-----------|-----------|-----------|
      female |       458 |        12 |       470 |
             |   211.613 |   211.613 |           |
             |     0.974 |     0.026 |     0.495 |
             |     0.964 |     0.025 |           |
             |     0.482 |     0.013 |           |
-------------|-----------|-----------|-----------|
        male |        17 |       463 |       480 |
             |   207.204 |   207.204 |           |
             |     0.035 |     0.965 |     0.505 |
             |     0.036 |     0.975 |           |
             |     0.018 |     0.487 |           |
-------------|-----------|-----------|-----------|
Column Total |       475 |       475 |       950 |
             |     0.500 |     0.500 |           |
-------------|-----------|-----------|-----------|
```

Classification Matrix for KNN on Validation Data:



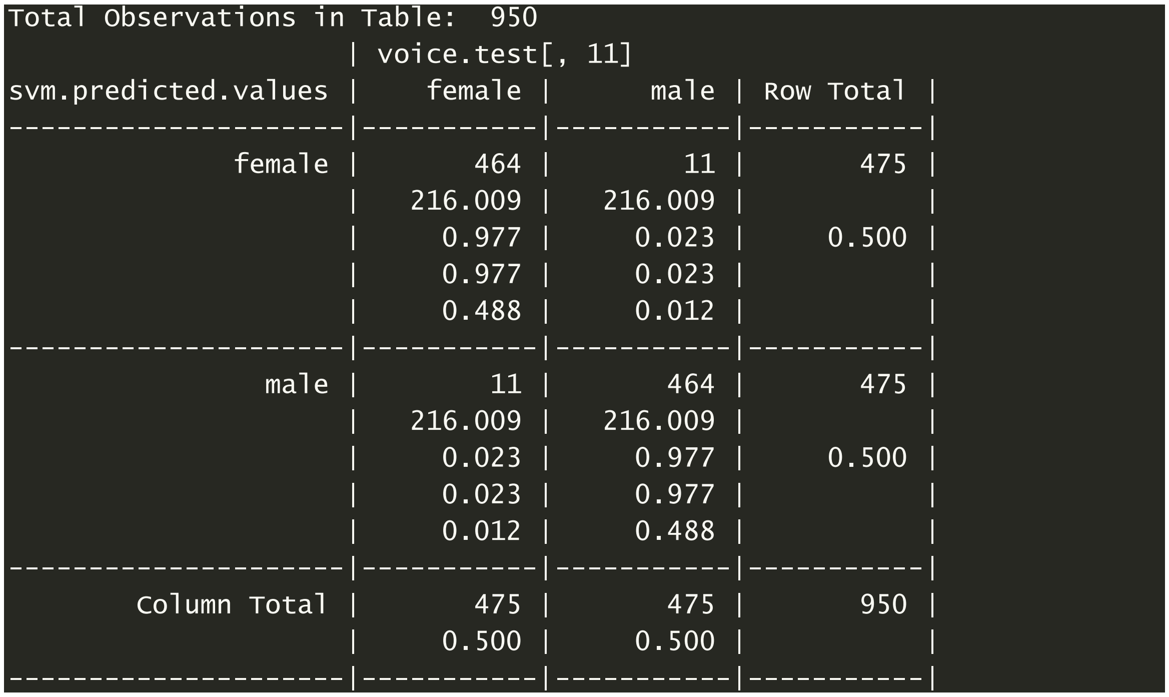
```
Total Observations in Table:  950

                     | voice.test[, 11]
predicted.gender.knn |    female |      male | Row Total |
---------------------|-----------|-----------|-----------|
              female |       459 |        11 |       470 |
                     |   213.515 |   213.515 |           |
                     |     0.977 |     0.023 |     0.495 |
                     |     0.966 |     0.023 |           |
                     |     0.483 |     0.012 |           |
---------------------|-----------|-----------|-----------|
                male |        16 |       464 |       480 |
                     |   209.067 |   209.067 |           |
                     |     0.033 |     0.967 |     0.505 |
                     |     0.034 |     0.977 |           |
                     |     0.017 |     0.488 |           |
---------------------|-----------|-----------|-----------|
        Column Total |       475 |       475 |       950 |
                     |     0.500 |     0.500 |           |
---------------------|-----------|-----------|-----------|
```

Classification Matrix for Random Forest on Validation Data:



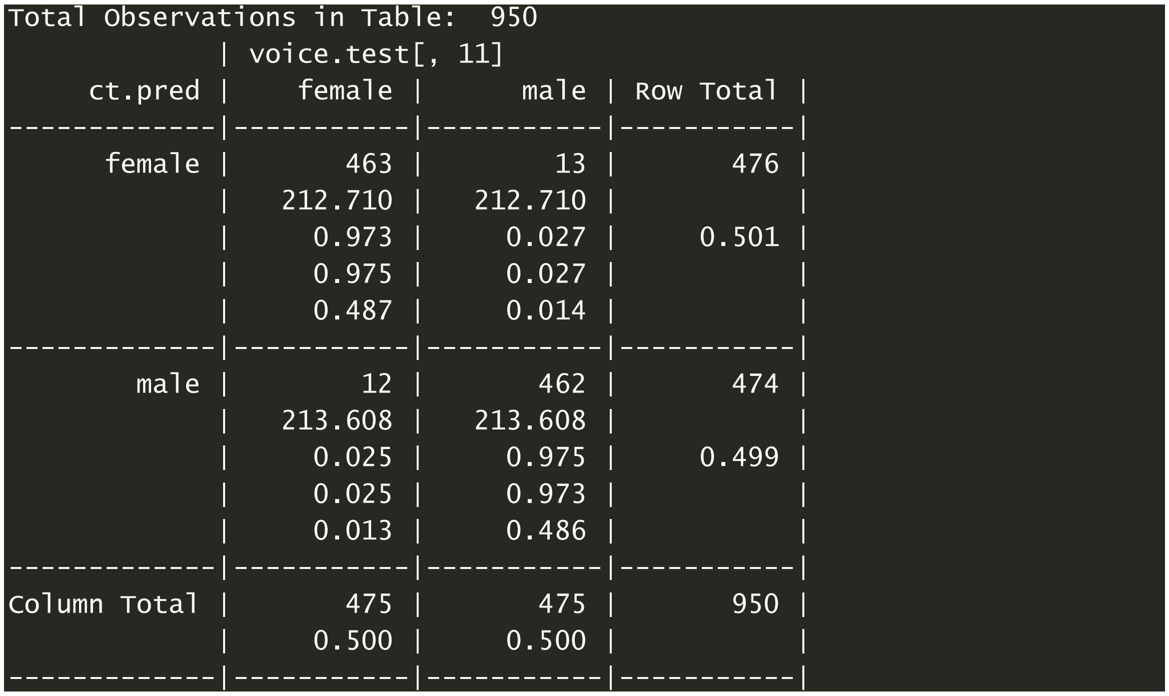
```
Total Observations in Table:  950
                 | voice.test[, 11]
predictedresults |    female |      male | Row Total |
-----------------|-----------|-----------|-----------|
          female |       457 |        15 |       472 |
                 |   206.953 |   206.953 |           |
                 |     0.968 |     0.032 |     0.497 |
                 |     0.962 |     0.032 |           |
                 |     0.481 |     0.016 |           |
-----------------|-----------|-----------|-----------|
            male |        18 |       460 |       478 |
                 |   204.356 |   204.356 |           |
                 |     0.038 |     0.962 |     0.503 |
                 |     0.038 |     0.968 |           |
                 |     0.019 |     0.484 |           |
-----------------|-----------|-----------|-----------|
    Column Total |       475 |       475 |       950 |
                 |     0.500 |     0.500 |           |
-----------------|-----------|-----------|-----------|
```

Classification Matrix for Support Vector Machine (SVM) on Validation Data:



```
Total Observations in Table:  950
                     | voice.test[, 11]
svm.predicted.values |    female |      male | Row Total |
---------------------|-----------|-----------|-----------|
              female |       464 |        11 |       475 |
                     |   216.009 |   216.009 |           |
                     |     0.977 |     0.023 |     0.500 |
                     |     0.977 |     0.023 |           |
                     |     0.488 |     0.012 |           |
---------------------|-----------|-----------|-----------|
                male |        11 |       464 |       475 |
                     |   216.009 |   216.009 |           |
                     |     0.023 |     0.977 |     0.500 |
                     |     0.023 |     0.977 |           |
                     |     0.012 |     0.488 |           |
---------------------|-----------|-----------|-----------|
        Column Total |       475 |       475 |       950 |
                     |     0.500 |     0.500 |           |
---------------------|-----------|-----------|-----------|
```

Classification Matrix for Neural Network on Validation Data:



```
Total Observations in Table:  950
             | voice.test[, 11]
     ct.pred |    female |      male | Row Total |
-------------|-----------|-----------|-----------|
      female |       463 |        13 |       476 |
             |   212.710 |   212.710 |           |
             |     0.973 |     0.027 |     0.501 |
             |     0.975 |     0.027 |           |
             |     0.487 |     0.014 |           |
-------------|-----------|-----------|-----------|
        male |        12 |       462 |       474 |
             |   213.608 |   213.608 |           |
             |     0.025 |     0.975 |     0.499 |
             |     0.025 |     0.973 |           |
             |     0.013 |     0.486 |           |
-------------|-----------|-----------|-----------|
Column Total |       475 |       475 |       950 |
             |     0.500 |     0.500 |           |
-------------|-----------|-----------|-----------|
```

As discussed, we applied the new data to SVM model and predicted its results. Following is the confusion matrix depicting the same.

```
Reference
Prediction male female
    male      3      1
    female    0      2
```

The model only misclassified just one record of 6 new records which shows the high accuracy the SVM model possess.

VI. Discussion and Recommendation

KNN is a simple model which is effective at capturing complex interactions among variables without having to define a statistical model. However, it might be time consuming in case of a large training set as it takes a significant amount time to find distances to all the neighbors and then identify the nearest one. In our case, it took about 9 seconds to run the model and compute the result which was highest when compared to the other four models. The model is computationally expensive and requires high memory as the algorithm store all the training data.

Logistic regression extends the idea of linear regression to a situation where outcome variable is categorical and binary (as in our case). The model needs two steps. First, it estimates of the probabilities of belonging to each class. Then by using a cutoff value on these probabilities to classify each case in one of the classes. We set the cutoff value to 0.5 for this case study. Although it didn’t achieve the highest accuracy rate among five models, it performed well when it comes to the time of execution. The model takes less than a second to run the mode.

Random Forest is an ensemble classifier consisting many decision trees, further providing the output by means of class's output by individual trees outputs of the class. The method couples the bagging idea and the random selection. Each tree is built up using a variant bootstrap data sample. In addition to this random forest changes it also constructs a classification of regression trees. In case of standard trees, the best node is split using its variables. Moreover, each node is divided using the best subset of predictors which are randomly chosen at that node. This method seems to be robust against over fitting. The main disadvantage of Random forests is their complexity. They are much harder and time-consuming to construct than decision trees. They also require more computational resources and are also less intuitive. In our case it took around 3 seconds for the model to run. Moreover, the accuracy was not the best when compared to neural network and SVM algorithm.

Support Vector Machine (SVM) is the machine learning task for deducing functions from a labeled training data that can be occupied for both classification and regression. Support vector machines are a binary classification algorithm. Support vectors are the data points adjacent to the hyperplanes if the dataset is removed it will change the position of the dividing hyperplane. Advantages of model selection by means of both optimal number as well as the location of functions are obtained automatically during training. SVM is especially effective in cases where the number of dimensions is greater. Since we have around 20 dimensions in our model as a predictor, SVM model gave us the highest accuracy and it took less than a second for a model to run.

Neural Network is an information processing paradigm that is inspired by the way biological nervous system process information. An artificial neuron is a device with many inputs and one output. This way we can combine input information in a flexible way that captures complicated relationships among variables. The hidden layer acting as a distillation layer helps distill some of the important patterns from the input and passes it onto the next layer to see. Being one of the most effective approach while designing a model, neural networks are more computationally expensive that the traditional algorithms. The amount of computational power needed for a Neural Network depends on the size of the data but also the complexity of the network.

VII. Summary

Based on the dataset and the prediction of whether the gender is male or a female, we highly recommend using the SVM model. The Random forest has the lowest accuracy - 96.53% and the computation time being slightly higher as compared to others (2.28 seconds). There’s a minute difference between the accuracies for the different approaches, but we would recommend SVM with the lowest computation time (0.47 seconds) and the highest accuracy (97.68%).

Appendix: R Code for use case study

```
# Load the required libraries
library(dplyr)
library(Amelia)
library(ggplot2)
library(corrgram)
library(corrplot)
library(caTools)
library(caret)
library(gains)
library(class)
library(randomForest)
library(e1071)
library(psych)
library(neuralnet)
library(pROC)
library(gmodels)
library(tuneR)
library(psycho)
library(warbleR)

#Load the files
```

```
voice.df <- read.csv("voice.csv")
```

#Exploratory Data Analysis

```
head(voice.df)
str(voice.df)
summ <- summary(voice.df[,-21])
any(is.na(voice.df))
missmap(voice.df, main="Voice Data - Missings Map",col=c("yellow", "black"),
legend=FALSE)
print(summ)
ggplot(voice.df, aes(meanfreq, fill =  label)) + geom_histogram( color="black",alpha=0.3
,bins = 30)
ggplot(voice.df, aes(mode)) + geom_histogram( color= "black",alpha=0.3 ,bins = 30)
smf <- summarise(group_by(voice.df,label),mean(mode))
voice.df[which(voice.df$mode == 0 & voice.df$label == "male"), "mode"] <-
smf$`mean(mode)`[2]
voice.df[which(voice.df$mode == 0 & voice.df$label == "female"), "mode"] <-
smf$`mean(mode)`[1]
ggplot(voice.df, aes(mode, fill =  label)) + geom_histogram( color= "black",alpha=0.3
,bins = 30)
ggplot(voice.df, aes(modindx, fill =  label)) + geom_histogram( color= "black",alpha=0.3
,bins = 30)
ggplot(voice.df, aes(dfrange, fill =  label)) + geom_histogram( color= "black",alpha=0.3
,bins = 30)
```

#Seperating Numerical columns

```
num.cols <- sapply(voice.df, is.numeric)
```

#Plotting Correlation plot

```
corr.data <- cor(voice.df[,num.cols])
corrplot(corr.data,method='number')
corrgram(voice.df,order=TRUE, lower.panel=panel.shade,upper.panel=panel.pie,
text.panel=panel.txt)
```

#Function to Assign 1 to male and 0 to female

```
val <- function(lab){
  temp <- 1:length(lab)
  for (i in 1:length(lab)) {
    if(lab[i] == "male"){
      temp[i] <- 1
    }
    else{
      temp[i] <- 0
    }

  }
  return(temp)}
```

```
#Principal Component Analysis (PCA)
voice.pca<- prcomp(voice.df[,-21],scale. = T)
summary(voice.pca)
pc_var <- (voice.pca$sdev^2)/sum(voice.pca$sdev^2)
plot(pc_var, xlab = "Principal Component", ylab = "Proportion of Variance Explained",
type = "b")
plot(voice.pca, main = "Principal Component Analysis")
voice.pca.imp<-as.data.frame(voice.pca$x[,1:10])
voice.pca.imp$label <- voice.df$label

#Split the data into training and validation data
set.seed(101)
split =  sample.split(voice.pca.imp$label, SplitRatio = 0.7)
voice.train <- subset(voice.pca.imp,split== TRUE)
voice.test <- subset(voice.pca.imp,split== FALSE)

#Train and Build logistic regression model using PCA scores
start_time<-Sys.time()
logmodel <- glm(label ~ ., family = binomial(link = 'logit'),data = voice.train)
summary(logmodel)

# Predict the data
fitted.probability <- predict(logmodel,newdata = voice.test[,-11],type = 'response')
end_time<-Sys.time()
time.taken.logit <-end_time-start_time
time.taken.logit <- round(as.numeric(time.taken.logit),2)
fitted.results <- as.factor(ifelse(fitted.probability > 0.5,1,0))
logit.con <- confusionMatrix(as.factor(ifelse(fitted.results=="1", "male", "female")),
voice.test[,11])
ct <-as.factor(ifelse(fitted.results=="1", "male", "female"))
CrossTable(ct, voice.test[,11])
print(logit.con$table)
accuracy.logit <- round(logit.con$overall[[1]] * 100 ,2)
print(paste("Accuracy :",accuracy.logit,"%"))

# KNN

start_time<-Sys.time()
acc <- 1:100
for(i in 1:100){
  set.seed(101)
  predicted.gender.knn <-  knn(voice.train[,-11],voice.test[,-11],voice.train[,11],k=i)
  c <- confusionMatrix(predicted.gender.knn, voice.test[,11])
  acc[i] <- c$overall[[1]] * 100
}
```

```
acc <- as.data.frame(acc)
acc$knn <- 1:100
acc$err <- 100- acc$acc
ggplot(acc,aes(x=knn,y=err)) + geom_point()+ geom_line(lty="dotted",color='red')
set.seed(101)
predicted.gender.knn <-  knn(voice.train[,-11],voice.test[,-11],voice.train[,11],k=1)
end_time<-Sys.time()
time.taken.knn <-end_time-start_time
time.taken.knn <- round(as.numeric(time.taken.knn),2)
print(time.taken.knn)
conknn <- confusionMatrix(predicted.gender.knn, voice.test[,11])
CrossTable(predicted.gender.knn, voice.test[,11])
print(conknn)
accuracy.knn <- round(conknn$overall[[1]] * 100 ,2)
print(paste("Accuracy :",accuracy.knn,"%"))
```

#Random Forest

```
start_time<-Sys.time()
voice.randf.model <- randomForest(label ~ ., data = voice.train, ntree = 500)
print(voice.randf.model)
voice.randf.model$confusion
voice.randf.model$importance
predictedresults <- predict(voice.randf.model,voice.test[,-11])
end_time<-Sys.time()
time.taken.rdf <-end_time-start_time
time.taken.rdf <- round(as.numeric(time.taken.rdf),2)
print(time.taken.rdf)
plot(voice.randf.model)
conrdf <- confusionMatrix(predictedresults,voice.test[,11])
CrossTable(predictedresults,voice.test[,11])
print(conrdf)
accuracy.rdf <- round(conrdf$overall[[1]] * 100 ,2)
print(paste("Accuracy :",accuracy.rdf,"%"))
```

# SVM algorithm

```
start_time<-Sys.time()
voice.svm.model <- svm(label ~ ., data= voice.train)
summary(voice.svm.model)
svm.predicted.values <- predict(voice.svm.model,voice.test[,-11],type="class")
end_time<-Sys.time()
time.taken.svm <-end_time-start_time
time.taken.svm <- round(as.numeric(time.taken.svm),2)
print(time.taken.svm)
```

```
confsvm <- confusionMatrix(svm.predicted.values,voice.test[,11])
CrossTable(svm.predicted.values,voice.test[,11])
accuracy.svm <- round(confsvm$overall[[1]] * 100 ,2)
print(paste("Accuracy :",accuracy.svm,"%"))

# Neural Network
start_time<-Sys.time()

# Get column names
f <- as.formula(paste("label ~", paste(n[!n %in% "label"], collapse = " + ")))
nn.voice <- neuralnet(f,data= voice.train)
plot(nn.voice, rep = "best")
summary(nn.voice)
pred.voice <- compute(nn.voice,voice.test[,-11])
predicted.class=apply(pred.voice$net.result,1,which.max)-1
predicted.class <- as.factor(predicted.class)
end_time<-Sys.time()
time.taken.nn <-end_time-start_time
time.taken.nn <- round(as.numeric(time.taken.nn),2)
print(time.taken.nn)
confnn <- confusionMatrix(as.factor(ifelse(predicted.class=="1", "male",
"female")),voice.test[,11])
print(confnn)
accuracy.nn <- round(confnn$overall[[1]] * 100 ,2)
print(paste("Accuracy :",accuracy.nn,"%"))

# New Data

#Loading the voice file: male voice
snap <- readWave("Recording.wav")
print(snap)
plot(snap@left[30700:31500], type = "l", main = "Snap",xlab = "Time", ylab =
"Frequency")
summary(snap)
ad <- autodetec(threshold = 5, env = "hil", ssmooth = 300, power=1,bp=c(0,22), xl = 2,
picsize = 2, res = 200, flim= c(1,11), osci = TRUE, wl = 300, ls = FALSE, sxrow = 2,
rows = 4, mindur = 0.1, maxdur = 1, set = TRUE)
c <- specan(ad,bp=c(0,1),pd= F)

#Loading the voice file: female voice

snap2 <- readWave("FemaleRecord.wav")
print(snap2)
plot(snap2@left[30700:31500], type = "l", main = "Snap",xlab = "Time", ylab =
"Frequency")
summary(snap2)
```

```
ad2 <- autodetec(threshold = 5, env = "hil", ssmooth = 300, power=1,bp=c(0,22), xl = 2,
picsize = 2, res = 200, flim= c(1,11), osci = TRUE,wl = 300, ls = FALSE, sxrow = 2,
rows = 4, mindur = 0.1, maxdur = 1, set = TRUE)
c2 <- specan(ad,bp=c(0,1),pd= F)
```

#Consolidating the male and female data

```
newdata <- rbind(c,c2)
```

#adjusting variable names

```
newdata$median <- c$freq.median
newdata$Q25 <- c$freq.Q25
newdata$Q75 <- c$freq.Q75
newdata$IQR <- c$freq.IQR
newdata <- newdata[names(newdata) %in% names(voice.df)]
```

#Mean Imputation of missing data

```
smf <- summarise(group_by(voice.df,label),mean(maxfun))
newdata$maxfun[1:3] <- round(smf[2,2],7)
newdata$maxfun[4:6] <- round(smf[1,2],7)
newdata$maxfun <- as.numeric(newdata$maxfun)
smf <- summarise(group_by(voice.df,label),mean(minfun))
newdata$minfun[1:3] <- round(smf[2,2],7)
newdata$minfun[4:6] <- round(smf[1,2],7)
newdata$minfun <- as.numeric(newdata$minfun)
smf <- summarise(group_by(voice.df,label),mean(meanfun))
newdata$meanfun[1:3] <- round(smf[2,2],7)
newdata$meanfun[4:6] <- round(smf[1,2],7)
newdata$meanfun <- as.numeric(newdata$meanfun)
smf <- summarise(group_by(voice.df,label),mean(centroid))
newdata$centroid[1:3] <- round(smf[2,2],7)
newdata$centroid[4:6] <- round(smf[1,2],7)
newdata$centroid <- as.numeric(newdata$centroid)
smf <- summarise(group_by(voice.df,label),mean(mode))
newdata$mode[1:3] <- round(smf[2,2],7)
newdata$mode[4:6] <- round(smf[1,2],7)
newdata$mode <- as.numeric(newdata$mode)
newdata$label <- factor("male",levels = c("male","female"))
newdata$label[4:6] <- factor("female",levels = c("male","female"))
new.svm.model <- svm(label ~ ., data= voice.df)
```

#Predicting the gender of new data

```
new.predicted.values <- predict(new.svm.model,newdata[,-21],type="class")
```

#Performance evaluation

```
confusionMatrix(as.factor(new.predicted.values), newdata[,21])
```