



The National ICT Up skilling program – Data Science section

CLASSIFY VOICE BASED ON GENDER

PREPERD BY

ANWAR FAIEZ ANEES ABBASS

Table of Contents

Summary.....	4
1 Introduction.....	5
2 Literature review.....	Error! Bookmark not defined.
3 Related work.....	8
3.1 Algorithms.....	8
3.1.1 Random Forest.....	8
3.1.2 Decision Tree.....	9
3.1.3 Perceptron.....	10
3.1.4 Multilayer Perceptron.....	11
3.1.5 Support vector machine.....	11
3.2 Methodology.....	12
5 Result.....	18
5.1 Decision Tree.....	18
5.2 Random Forest.....	19
5.3 Logistic Regression Model.....	20
5.4 Confusion matrix.....	23
5.5 Neural Network.....	25
5.5.1 Perceptron.....	25
5.5.2 Multilayer Perceptron	26
5 Conclusion.....	27
6 References.....	28

LIST OF FIGURES

Figure1: Random Forest

Figure2: Desicion Tree

Figure3: Perceptron

Figure4: Multilayer Perceptron

Figure5: Support victor machine

Figure6: Data information

Figure7: Number of labels

Figure8: Number of labels 2

Figure9: Features correlated

Figure10: Decision tree command with entropy criterion

Figure11: Decision tree command with gini criterion

Figure12: Random forest command with entropy criterion

Figure13: Random forest command with entropy criterion

Figure14: optimizer options

Figure15: Learning rate

Figure16: Batch size

Figure17: Initialization

Figure18: Logistic model

Figure19: Algorithms result

Figure20: Confusion matrix

Figure21: Perceptron accuracy

Figure22: Multilayer perceptron

Summary:

In this research, we used data-set from the Kaggle website. We have used classification algorithms and neural network algorithms to train and test data using python language.

Building the best possible model, so we tried to improve the data and compare the results of the algorithms used, such as the Random Forest algorithm and some algorithms not used in previous research such as Multilayer Perceptron to find the best possible algorithm that gives the best results

This project we worked on was a real challenge for our community, we have never been through this kind of analysis before, which makes it more difficult, but we've done what we can do, and this is going to be our first step in the real world of the voice systems

1. Introduction:

With the great technological development that we are witnessing, it has become possible to talk with machines and understand what we say through modern sound systems. Voice classification systems are gaining popularity due to their wide range of applications used in a variety of fields ranging from student exams, security services, documentation, and content-based information retrieval to criminal investigations, and search engines.

However, sound systems lack many of the characteristics that specialists are still trying to develop and improve. The classification of voice by gender is one of the most important of these characteristics because of its great impact and great importance, which was shown by a group of studies, which showed that gender-based speech recognition models perform much better than sex-independent models. This is because determining sex, for example, in search engines gives better results according to the area of interest for each gender. They are also more able to show satisfactory results due to understanding the concerns of each gender separately.

We aim to give promising results to the classification problem in all fields. This project aims to classify voice by machine learning using classification algorithms and Neural Network Algorithm, then compare results and accuracy, improve the quality of gender voice classification.

2. Literature review:

As people have always been passionate about talking to machines, there was a need for a computer to identify the gender of the speakers to understand the speech and their needs according to the voice features.

Hussam Allah Sheik [1] whose work aims to build a gender-based classification system applied to the GMM (Gaussian Mixture Modeling) algorithm with MFCC and SDC (where SDC fused model gave satisfactory results on Voxforge dataset).

Nevertheless, when they tested on different data sets with different data languages, they were not large, and the accuracy was 80%.

While Jiao [2] investigated whether there is a main effect of speakers' native language (Arabic, Korean, and Mandarin) even when speaking a second language, English, it also investigated a particular speaker-listener relationship, namely the degree of linguistic familiarity. And estimating the age of the speaker, they also found that smokers were older than non-smokers of the same ages, probably because of the effect of smoking.

Acoustic characteristics of a human being, as with age estimates, researchers have found that there is a miscalculation of about ten years, as the age of younger adults has overestimated, while older people have underestimated and Accuracy in total (71%).

The genetic algorithm is also used to identify a speech gender by comparing different approaches like the combination of fuzzy logic and neural network [3].

Automatic speech recognition and speaker verification can be accomplished under rather highly constrained conditions [4] an automatic gender classifier assists the development of improved male and female voice synthesizers [5, 6] in [7] an accent classification method is introduced on

the top of gender classification. The most closely related work to the present one is that of Xiao et al. [8] where gender classification was incorporated in an emotional speech recognition system using a wrapper approach based on back-propagation neural networks with sequential forward selection. An accuracy of 94.65% was reported for gender classification in the Berlin dataset [9], which that research showed back-propagation neural networks for the first time studying the gender classification to express the speech in 5 emotional classes, such as anger, happiness, neutral, sadness, and surprise. The high precision it provided was one of the reasons why our team used the back-propagation algorithm-specific.

This paper concentrates on a comparative study of gender and age [7] gender classification can be performed with an accuracy of 95% approximately using speech signal either from both genders or male and female separately. The accuracy for age classification is about 88%.

Many recent studies have considered the problem of language identification based on various speech production features [10, 11]. A related problem that has not been explored in detail is the issue of foreign accent identification. The accent is also a challenging problem in speech recognition. It is one of the most important factors, aside from gender; it creates undesirable variability in speaker-independent speech recognition schemes.

[12]An initial version of the classification algorithm classed speaker accent from among four different accents with an accuracy of 81.5% in the case of unknown text, and 88.9% assuming known text. Finally, it is shown that as accent sensitive word count increases, the ability to correctly classify accent also increases, achieving an overall classification rate of 92% among four accent classes.

Automatic identification of age and gender of a person from his/her speech has gained increasing importance in recent years because of its necessity in various commercial, medical, and forensic applications our speaking style to the person we are talking to [13]the automatic dialogue system might change the speed of its speech synthesis according to the identified user's age [14]They[15]describes an experiment with using the Gaussian mixture models (GMM) for automatic classification of the speaker age and gender, using MFCC features and support vector machines (SVMs) with a GMM super vector the overall precision of about 75 % achieved.

3. Related Work:

There are many algorithms used to classify voice based on gender, like Linear Discriminate, Classification and Regression Trees, Decision Tree, etc.

So, we used classification algorithms and neural network algorithms, some of them are used before in many types of research like Linear Discriminate, and we tried to improve accuracy by using unused algorithms like Back Propagation (Multilayer Perceptron), we think it will be better in classifying voice than other algorithms that was used before.

3.1. Algorithms

3.1.1. Random Forest:

Random forests are a group-learning method for classification (and regression) that operates by constructing a multitude of decision-making trees at training time and generating a class that is the class-learning mode for individual trees. The random forest-inducing algorithm was

developed by Leo Breiman[16] and Adele Cutler[17], and "Random Forests" is their trademark. The term originated from a random decision on forests that were first proposed by Tin Kam Ho of Bell Labs in 1995. The method combines the 'bagging' idea of Breiman and the random selection of features introduced independently by Ho [18][19] and Amit and Gemanin[20] to construct a collection of decision-making trees with controlled variance. The selection of a random subset of features is an example of a random subspace method which, in Ho's formulation, is a means of implementing the classification proposed by Eugene Kleinberg.[21]

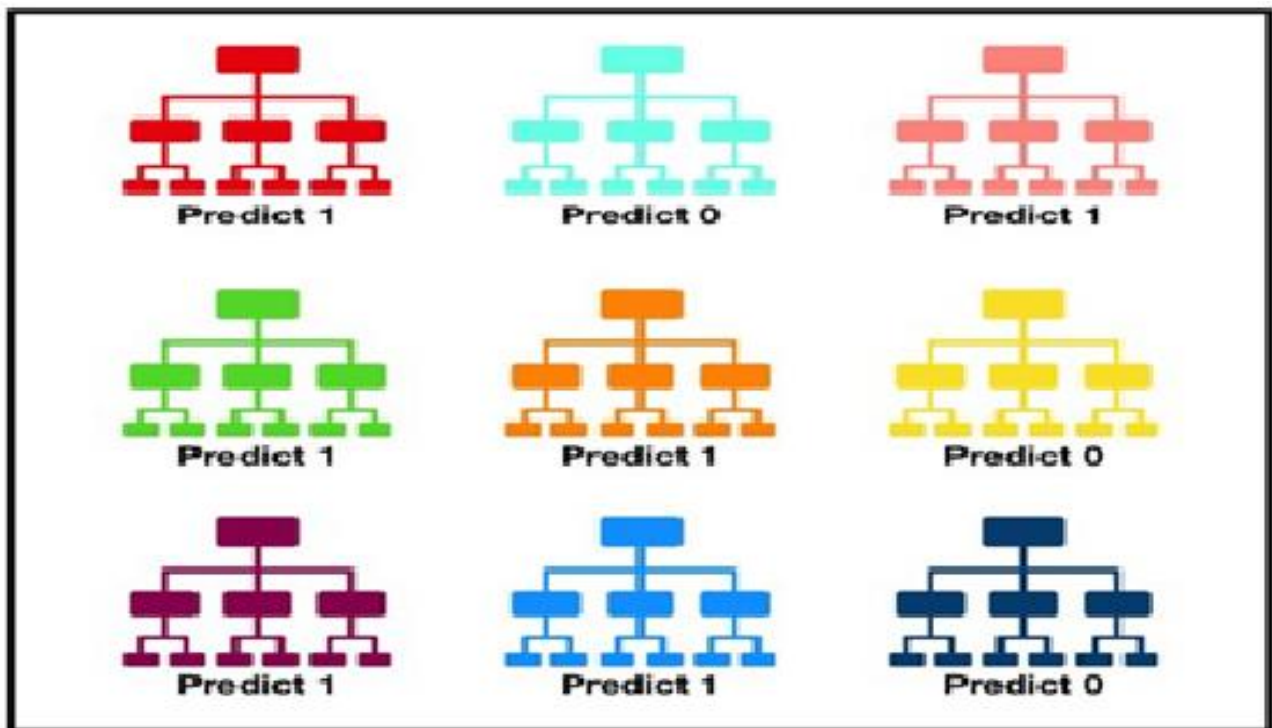


Figure1: Random Forest

3.1.2. Decision Tree (weka.classifiers.rules.DecisionTable):

Decision tables are a concise visual representation of the actions to be carried out depending on the conditions. The information expressed in the decision tables could also be represented as decision trees or as a series of if-then-else and switch-case statements in the programming

language. Each decision corresponds to a variable, relationship, or predicate, the possible values of which should be listed when alternatives to the condition. Each action is a procedure or operation to be carried out, and the entries specify whether (or in what order) the action is to be carried out for a set of condition alternatives to which the entry corresponds. To make them more concise, many decision tables include alternatives that don't care about symbols in their condition. This may be a hyphen [23][24][25] or a blank hyphen[26], although the use of a blank is discouraged, as it may merely indicate that the decision table has not been finalized. [Citation Need] One of the uses of the decision tables is to identify the conditions under which certain input factors are irrelevant to the action to be taken, allowing these input tests to be skipped and thus streamlining decision-making procedures. [27]

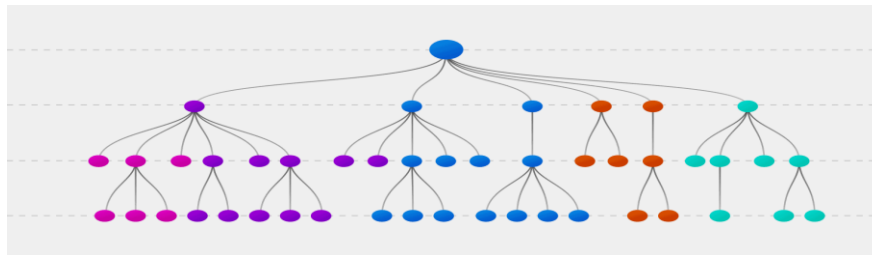


Figure2: Desicion Tree

3.1.3. Perceptron:

Perceptron is a simple binary classification algorithm, suggested by Cornell scientist Frank Rosenblatt. It helps to divide the input signals set into two parts—"yes" and "no." However, unlike many other classification algorithms, the perceptron was modeled on the essential unit of the human brain—the neuron—and has an uncanny ability to learn and solve complex problems.

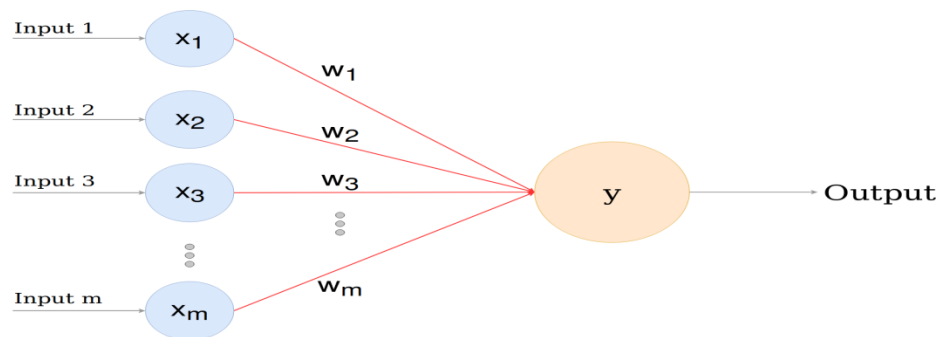


Figure3:Perceptron

3.1.4. Multilayer Perceptron:

A multilayer perceptron (MLP) is a perceptron that is paired with additional perceptron's, stacked in several layers, to solve complex problems. The diagram below shows a three-layer MLP. Also, every perceptron in the first layer to the left (the input layer) sends outputs to all the perceptron's in the second layer (the hidden layer) and all the perceptron's in the second layer send outputs to the final layer to the right (the output layer). Class MLPClassifier implements a multi-layer perceptron (MLP) algorithm that uses Back propagation to train.

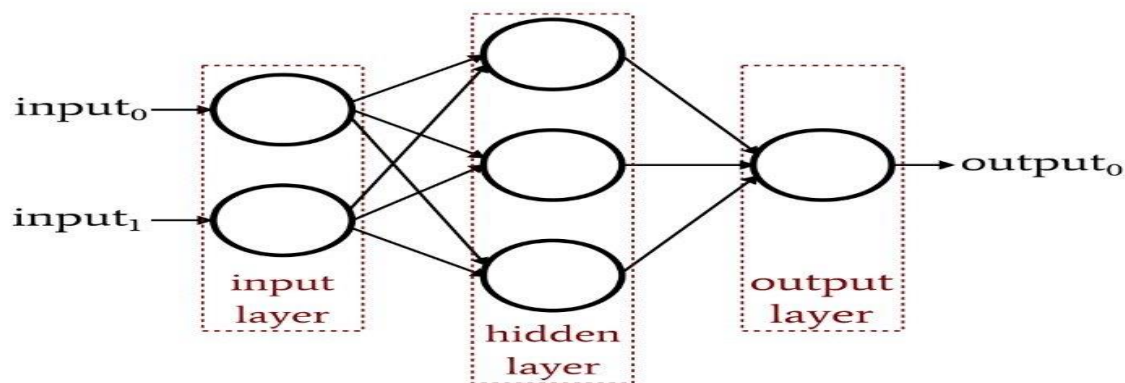


Figure4: Multilayer Perceptron

3.1.5. Support vector machine

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

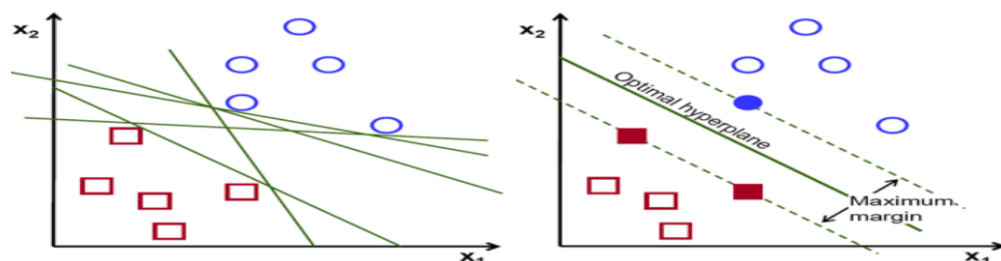


Figure5: Support vector machine

4. Methodology:

Gender recognition by speech and speech analysis

This database was created to identify the voice as male or female, based on the acoustic properties of voice and speech. The data set consists of 3,168 recorded voice samples from male and female speakers.

```
RangeIndex: 3168 entries, 0 to 3167
Data columns (total 21 columns):
#   Column      Non-Null Count  Dtype
---  -
0   meanfreq    3168 non-null   float64
1   sd          3168 non-null   float64
2   median      3168 non-null   float64
3   Q25         3168 non-null   float64
4   Q75         3168 non-null   float64
5   IQR         3168 non-null   float64
6   skew        3168 non-null   float64
7   kurt        3168 non-null   float64
8   sp.ent      3168 non-null   float64
9   sfm         3168 non-null   float64
10  mode        3168 non-null   float64
11  centroid    3168 non-null   float64
12  meanfun     3168 non-null   float64
13  minfun      3168 non-null   float64
14  maxfun      3168 non-null   float64
15  meandom     3168 non-null   float64
16  mindom      3168 non-null   float64
17  maxdom      3168 non-null   float64
18  dfrange     3168 non-null   float64
19  modindx     3168 non-null   float64
20  label       3168 non-null   object
dtypes: float64(20), object(1)
memory usage: 519.9+ KB
```

Figure6: Data information

The previous figure shows some information about the data set, such as the number of records that is 3168, the number of teachers that is 21 with the class column, and 20 of the columns that are float except for the class that is the object.

As for the class, it contains two labels Female or male, and we know from the figure below that the number of males is equal to the number of females.

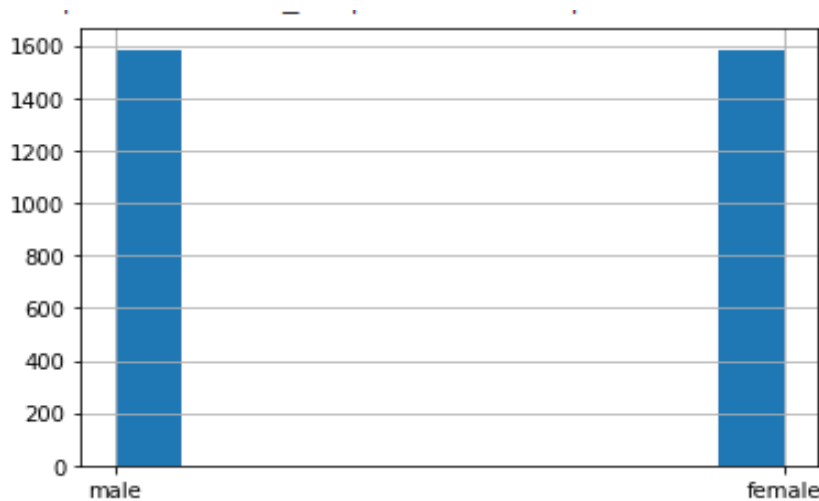


Figure7: Number of labels

Appropriate algorithms and data also help a lot in designing a model or machine learning system capable of distinguishing between different labels, and there are some steps we have to follow in order to show the best results. First, we tidy the data by looking for what might hinder the learning process, such as missing or incorrectly entered data or duplicated records and finally making sure that the class contains only two types of values. Then, pre-processing, we tried to improve the data by showing a percentage of the feature correlation and deleting highly correlated features. After that, we divided the data into training data and testing data and improved the scale of the features by using a standard scaler. Finally, we can start training and testing using classification algorithms and get the best results.

As we explained earlier, there are a number of steps that need to be followed, starting with the tidying the data, and at this stage, we first checked whether there was duplication in the records because duplicate records could affect the results of the algorithms and produce unreal results.

```
[5] voice.duplicated().sum()
```

2

So by 'duplicated().sum()' command we can know how much records are duplicated, and we found two duplicate records so we deleted them via the following command.

```
[6] voice.drop_duplicates(inplace=True, ignore_index=True)
```

Then, based on the information we saw it before in figure6, there are no missing data, but as we see now in there are some zero values that we think are 'nulls', so we've deleted those records that are estimated at 240 records.

mode

0.0

0.0

0.0

0.0

0.0

Then, by showing the class's unique values, we made sure that there were only two labels, male and female.

```
voice['label'].unique()  
array(['male', 'female'], dtype=object)
```

And as a result of the deletion of a group record, there was a slight difference between the number of female records and the number of male records.

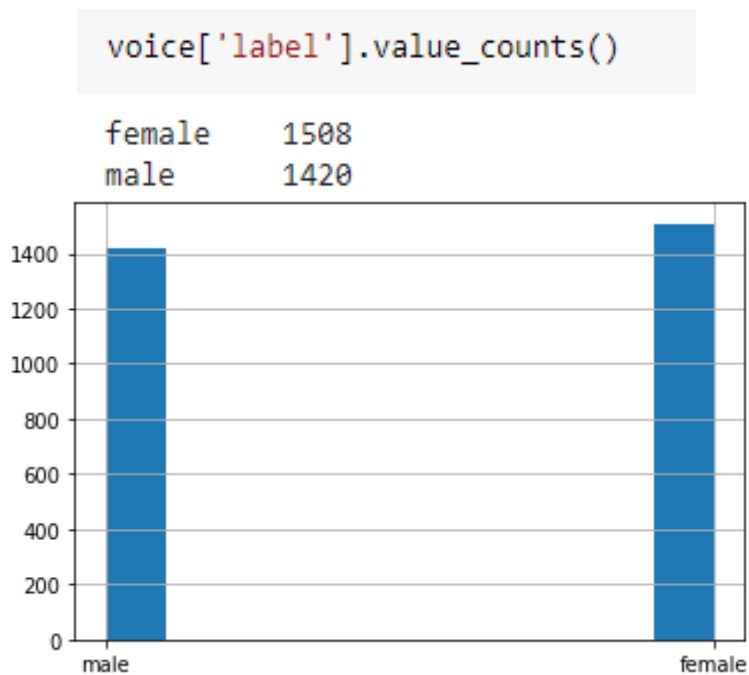


Figure8: Number of labels 2

Also, we gave values for class labels where females are '0' and males are '1' by mapping.

```
mapdec={'male':1,'female':0}  
voice['labelnew']=voice['label'].map(mapdec)
```


Second step, is the pre-processing phase, at this stage, we showed a correlation of features between each other and based on the results, like figure below, there were approximately five features with a high correlation of more than 90%,

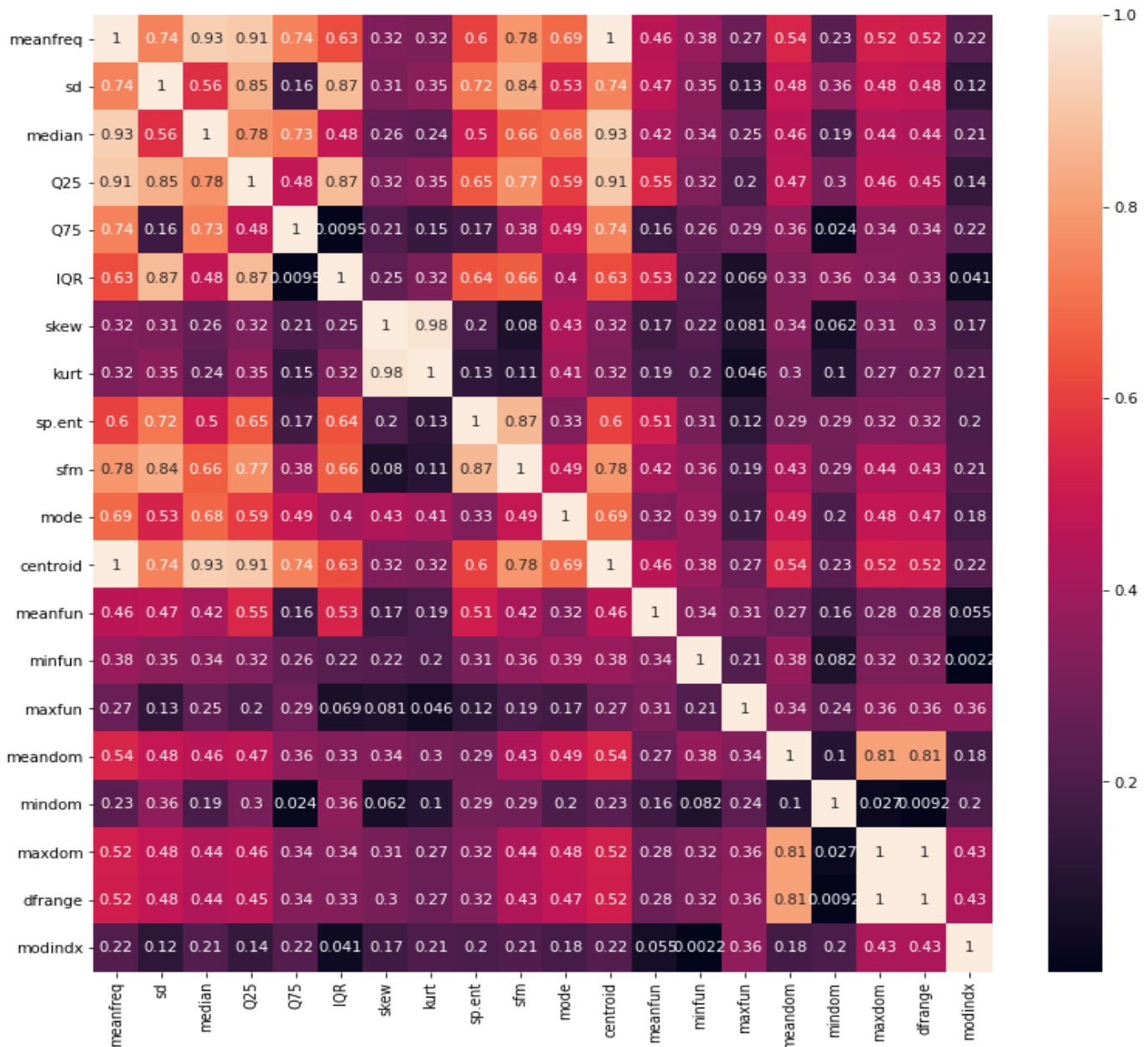


Figure9: features correlated

So we deleted them to make up the total of the features 15.

```
upper_tri = abs(cor_matrix.where(np.triu(np.ones(cor_matrix.shape),k=1).astype(np.bool)))
to_drop = [column for column in upper_tri.columns if any(upper_tri[column] > 0.900000)]
print(20-len(to_drop))
```

15

```
voice1 = clnvoice.drop(clnvoice[to_drop], axis=1)
```

```
voice1.drop('label', axis=1,inplace=True)
```

After that, we divided the data into training and testing data by using (train-test-split) and then applied a standard scale on them

```
all_inputs = voice1[voice1.columns.drop('labelnew')]
all_labels = voice1['labelnew'].values
```

```
(training_inputs,
 testing_inputs,
 training_classes,
 testing_classes) = train_test_split(all_inputs, all_labels, test_size=0.30, random_state=111)
```

```
sc =StandardScaler()
training_inputs = sc.fit_transform(training_inputs)
testing_inputs = sc.transform(testing_inputs)
```

Finally, after the data was ready to apply algorithms to it.

5. Results:

5.1 DecisionTree:

The decision tree has many parameters each one of them has its effect on the result so we need to choose the best choice for every parameter. At first, we started with criterion, what is the criterion? It is the function to measure the quality of a split, and have two choices, first gini where it use for the Gini impurity, and entropy for the information gain.

However, at entropy criterion, we have a decision tree score of 0.972

```
from sklearn.tree import DecisionTreeClassifier

# Create the classifier
decision_tree_classifier = DecisionTreeClassifier( criterion='entropy',class_weight='balanced',random_state=25)
# Train the classifier on the training set
decision_tree_classifier.fit(training_inputs, training_classes)

# Validate the classifier on the testing set using classification accuracy
print('decision_tree_classifier.score =',decision_tree_classifier.score(testing_inputs, testing_classes))

decision_tree_classifier.score = 0.9726315789473684
```

Figure10: Decision tree command with entropy criterion

In the other side, at gini criterion, we have a decision tree score of 0.966

```
from sklearn.tree import DecisionTreeClassifier

# Create the classifier
decision_tree_classifier = DecisionTreeClassifier( criterion='gini',class_weight='balanced',random_state=25)
# Train the classifier on the training set
decision_tree_classifier.fit(training_inputs, training_classes)

# Validate the classifier on the testing set using classification accuracy
print('decision_tree_classifier.score =',decision_tree_classifier.score(testing_inputs, testing_classes))

decision_tree_classifier.score = 0.9663157894736842
```

Figure11: Decision tree command with gini criterion

In the conclusion, entropy criterion is better than gini criterion because it gives a higher score or accuracy.

5.4 RandomForest:

Random forest is optimized for the decision tree so we applied what we did in the decision tree to the random forest.

However, at entropy criterion, we have a random forest score of 0.984.

```
from sklearn.ensemble import RandomForestClassifier

random_forest_classifier = RandomForestClassifier(bootstrap=True, class_weight='balanced',
                                                criterion='entropy', random_state=43 )

random_forest_classifier.fit(training_inputs, training_classes)
random_forest_classifier.score(testing_inputs, testing_classes)
```

Figure12: Random forest command with entropy criterion

In the other hand, at gini criterion, we have a random forest score of 0.983.

```
from sklearn.ensemble import RandomForestClassifier

random_forest_classifier = RandomForestClassifier(bootstrap=True, class_weight='balanced',
                                                criterion='gini', random_state=43)

random_forest_classifier.fit(training_inputs, training_classes)
random_forest_classifier.score(testing_inputs, testing_classes)

0.9831578947368421
```

Figure13: Random forest command with entropy criterion

Finally, as we saw there is no difference between gini and entropy criterion or it was so low.

5.3 Logistic Regression Model

In this algorithm we used grid search to find the best option for each parameter in order to get the best accuracy. So we applied grid search on optimizer to choose best optimizer from this list [SGD,Adam,Adagrad,RMSprop], and the best two were [SGD,Adagrad]

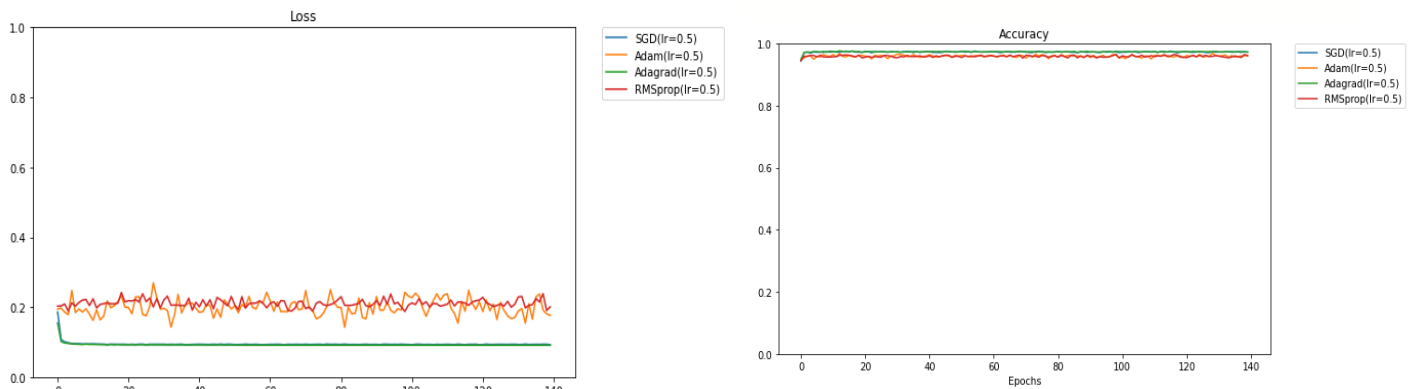


Figure14: optimizer options

The next grid search was on learning rate, the figure below show us the best rate, and the best rates from the list [0.05, 0.01, 0.1, 0.5, 0.8, 0.2] were 0.8 and 0.5.

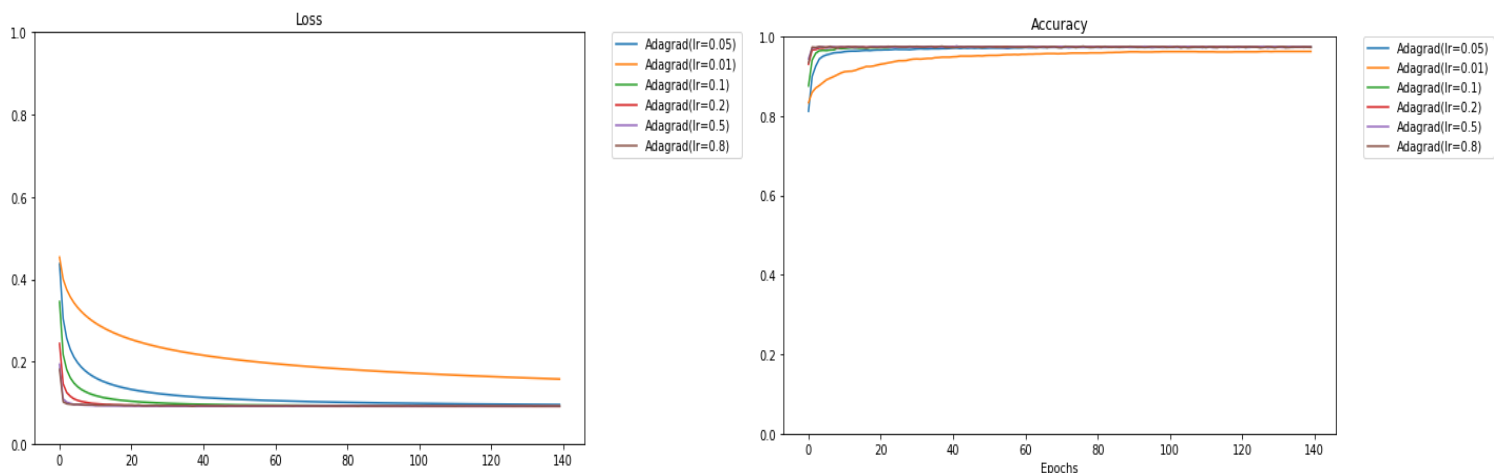


Figure15: learning rate

Then we have the batch size, whose options are as follows [2,3,7,10,11,15,20,25,30] and according to the results below, it was the best batch size from the list 20.

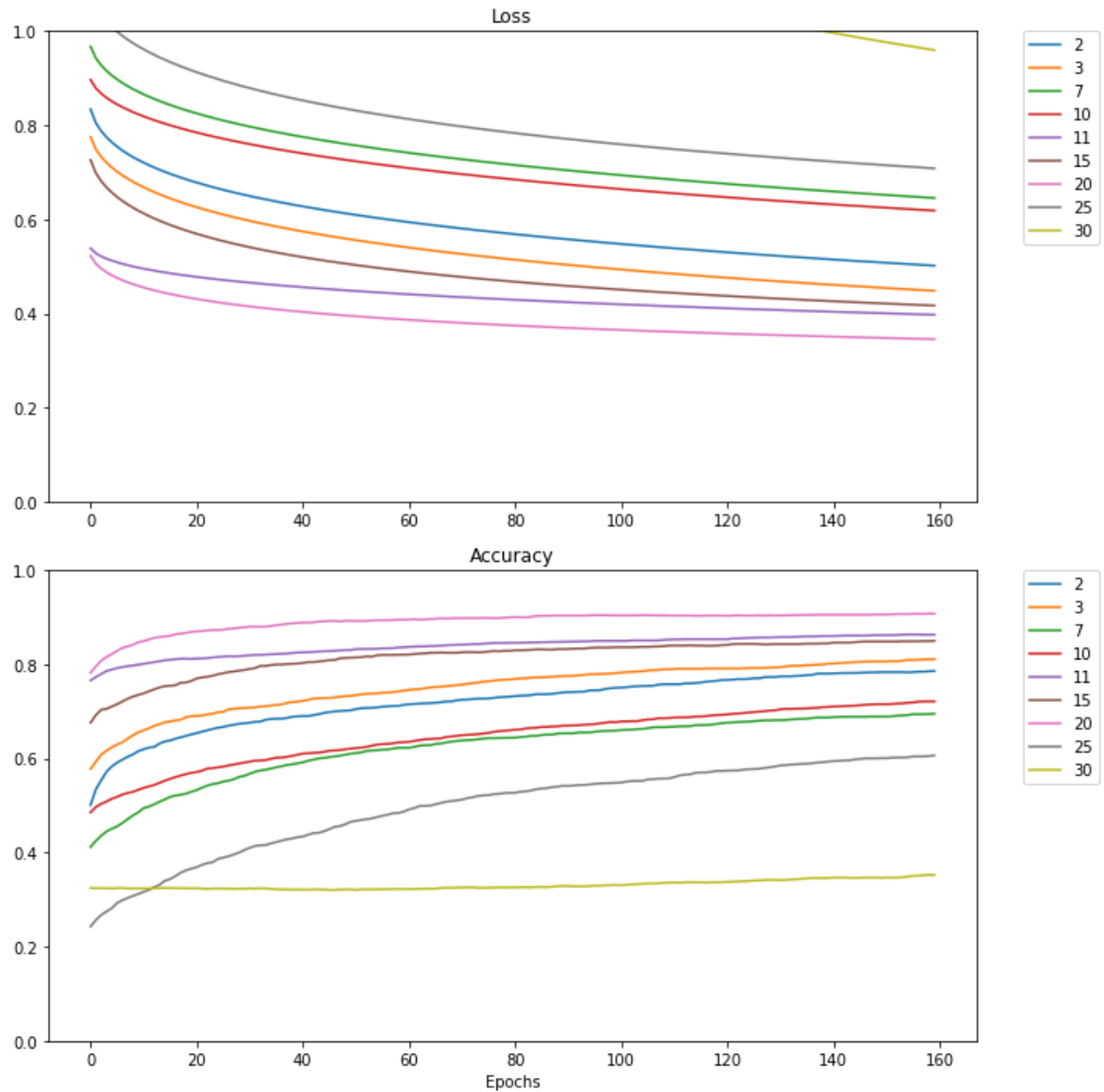


Figure16: Batch size

Finally, it was the last grid search for the initialization which the options were as follows [zeros, uniform, normal, he_normal, lecun_uniform, gloriot_uniform] and according to the results below, the configuration menu was all good.

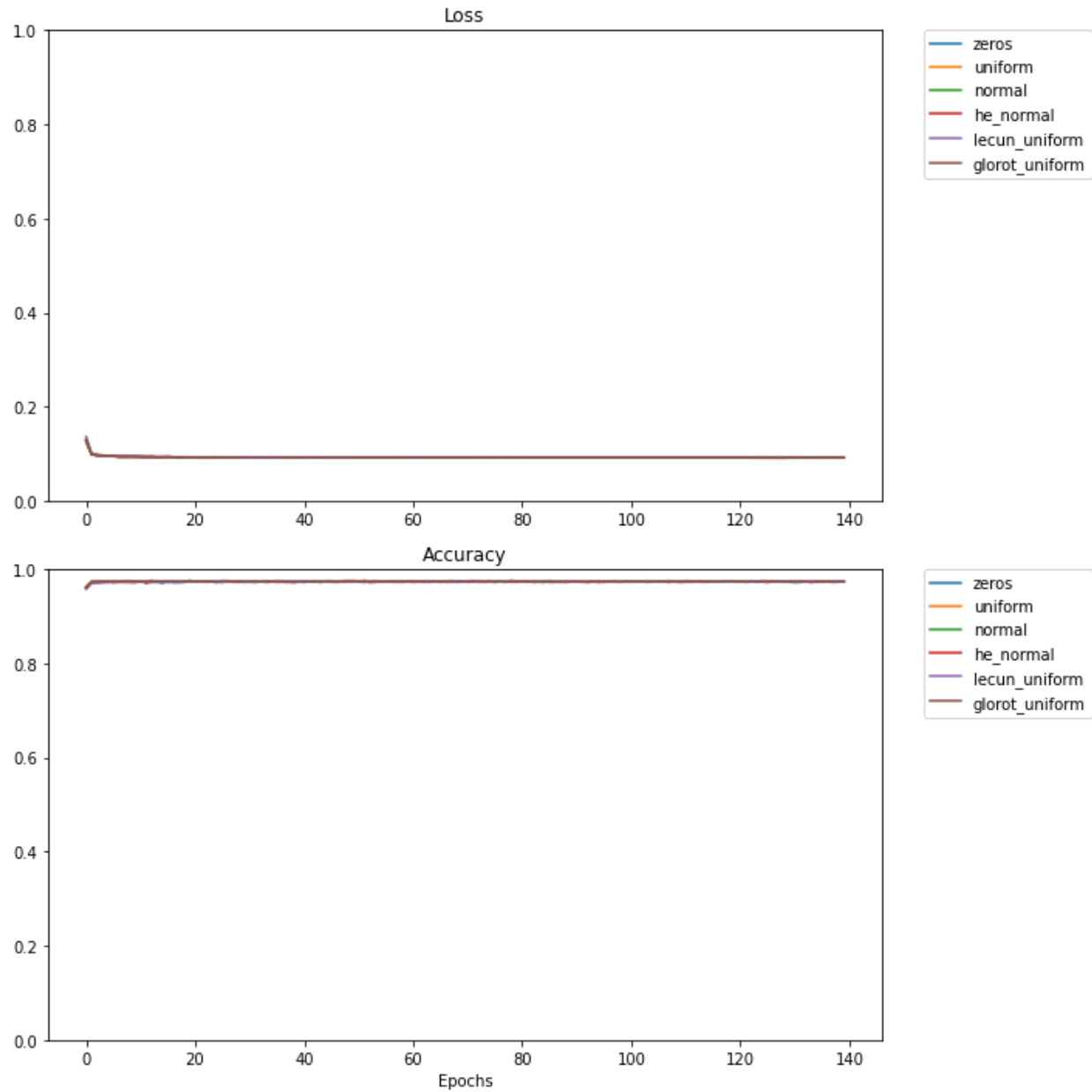


Figure17: initialization

The grid search is useful to choose the best options for parameters like what we've done before, and after we've got the best options, we've used the Logistic Regression Model. Figure below to view the results of the model when entering data

```
del model
model = Sequential()
model.add(Dense(1,kernel_initializer='lecun_uniform' , input_shape=(15,), activation='sigmoid'))
model.add(Dense(1,kernel_initializer='lecun_uniform' , activation='sigmoid'))
model.compile(Adagrad(lr=0.8), 'binary_crossentropy', metrics=['accuracy'])
model.fit(training_inputs, training_classes,batch_size=20, epochs=160, verbose=0,shuffle=False )
model.evaluate(testing_inputs,testing_classes)
```

30/30 [=====] - 0s 1ms/step - loss: 0.0843 - accuracy: 0.9789
[0.08433771133422852, 0.9789473414421082]

Figure18: logistic model

According to the previous figure, the accuracy was 0.9789, and what could also be important as a result of the high accuracy we achieved is that the loss was very low and the value was 0.08.

5.4 Confusion Matrix

A confusion matrix is a technique used to summarize the results of a classification algorithm. If you have an unequal number of observations in each class, or if you have more than two classes in your dataset, the accuracy of classification alone can be misleading. It will give you a better understanding of what your classification model is getting right and what kinds of mistakes it makes by computing a confusion matrix.

Based on the previous definition, we chose a group of algorithms to choose the best algorithm and use a confusion matrix to see more details on the results.

Seven different algorithms were used to compare accuracy and loss, and the algorithms were labeled as follows:

'LR', Logistic Regression

LDA ', Linear Discriminant Analysis

'LRCV', Logistic Regression CV

'KNN', K Neighbors Classifier

'CART', Decision Tree Classifier

'NB', Gaussian NB

'SVM', Support victor machine

```
LR, 0.9724729241877257, 0.011117946746168676))
LDA, 0.9684115523465704, 0.010170963600708155))
LRCV, 0.9724729241877257, 0.010820919502059366))
KNN, 0.9711191335740073, 0.006990944668244427))
CART, 0.970216606498195, 0.010009509482318947))
NB, 0.9237364620938628, 0.023426619769706503))
SVM, 0.9796931407942238, 0.005690216702580535))
```

Figure19: Algorithms result

As we can see from the result figure, the support vector machine gives us the best result with higher accuracy and lower loss, so the confusion matrix has been applied to it.

```
----- Accuracy -----
0.9810526315789474
-----
----- Confusion Matrix -----
[[448   9]
 [  9 484]]
-----
----- Classifcation Report-----
```

	precision	recall	f1-score	support
0	0.98	0.98	0.98	457
1	0.98	0.98	0.98	493
accuracy			0.98	950
macro avg	0.98	0.98	0.98	950
weighted avg	0.98	0.98	0.98	950

```
-----
```

Figure20: Confusion matrix

From the confusion matrix, we can see that there are nine male records classified as female from 457 records, and nine female records classified as male from 493 records. And as expected from the support vector machine algorithm, it has produced great results for the test data as the accuracy is 98 and the loss is estimated to be 0.01.

5.5 Neural Network

A neural network is a series of algorithms that seek to recognize underlying relationships in a data set through a process that mimics the way the human brain operates. Neural networks can adapt to changing inputs; therefore, the network generates the best possible result without the need to redesign the output criteria.

Perceptron and Multilayer Perceptron are some of the neural network algorithms, so we've expected high accuracy, and as known, neural network algorithms are powerful and always give amazing results.

5.5.1 Perceptron

Starting with the Perceptron algorithm and as is expected of it to give promising results, the accuracy of the test data was equivalent to 0.976

```
perceptron= Perceptron(shuffle=True,random_state=43,class_weight='balanced')
perceptron.fit(training_inputs,training_classes)
perceptron.score(training_inputs,training_classes)
```

```
0.973826714801444
```

```
y=perceptron.predict(testing_inputs)
accuracy_score(testing_classes,y)
```

```
0.9768421052631578
```

Figure21: Perceptron accuracy

5.5.2 Multilayer Perceptron

Multilayer Perceptron Algorithm It was developed from the Perceptron algorithm so it was expected that its results would be better than the Perceptron results, where testing data accuracy was 0.981, and because its results were wonderful, we used a confusion matrix to see the results in more detail.

```
mlp= MLPClassifier(hidden_layer_sizes=25, activation='relu', solver='sgd',
                   batch_size='auto', learning_rate='adaptive', learning_rate_init=0.005, power_t='invscaling',
                   shuffle=True, random_state=43, tol=0.0001)
mlp.fit(training_inputs,training_classes)
print(mlp.score(training_inputs,training_classes))
y=mlp.predict(testing_inputs)

print(accuracy_score(testing_classes,y))
```

0.98014440433213

0.9810526315789474

----- Accuracy -----

0.9810526315789474

----- Confusion Matrix -----

```
[[447  10]
 [  8 485]]
```

----- Classification Report-----

	precision	recall	f1-score	support
0	0.98	0.98	0.98	457
1	0.98	0.98	0.98	493
accuracy			0.98	950
macro avg	0.98	0.98	0.98	950
weighted avg	0.98	0.98	0.98	950

Figure22: Multilayer perceptron

6. Conclusion

In this work, we proposed to apply a set of algorithms on datasets based on gender as a class after tremendous studies to make voice recognition clearer and more understanding. The results we achieved varying according to the differences in the working mechanism of the algorithms. All of them were unexpectedly very good. In conclusion, we can choose the best algorithms that give higher accuracy, starting with the Multilayer Perception Algorithm with (98.1) accuracy then the Support Vector Machine algorithm with (98.1) accuracy and the Random Forest Algorithm with (98.4) accuracy. However, we can have higher accuracy if we have a larger dataset with a better feature, larger data give algorithm more record to train, and a good feature makes it easier for algorithms to separate labels by features.

The challenge for us is to find large data with good features to build a conventional system with good algorithms. Human nature is passionate about talking to machines and computers. Because of this very reason, voice classification systems and their improvement will not die soon and we hope to find other powerful algorithms.

7. References

1. Sheikh, H., *Who is speaking? Male or female*. 2013: GRIN Verlag.
2. Jiao, D., et al., *Age estimation in foreign-accented speech by non-native speakers of English*. *Speech Communication*, 2019. **106**: p. 118-126.
3. Adi, K., et al., *Identifying the developmental phase of Plasmodium falciparum in malaria-infected red blood cells using adaptive color segmentation and back propagation neural network*. *Int J Appl Eng Res*, 2016. **11**: p. 8754-9.
4. Wu, K. and D.G. Childers, *Gender recognition from speech. Part I: Coarse analysis*. *The journal of the Acoustical society of America*, 1991. **90**(4): p. 1828-1840.
5. Childers, D., K. Wu, and D. Hicks. *Factors in voice quality: Acoustic features related to gender*. in *ICASSP'87. IEEE International Conference on Acoustics, Speech, and Signal Processing*. 1987. IEEE.
6. Sedaaghi, M., *A comparative study of gender and age classification in speech signals*. *Iranian Journal of Electrical and Electronic Engineering*, 2009. **5**(1): p. 1-12.
7. Lin, X. and S. Simske. *Phoneme-less hierarchical accent classification*. in *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004*. 2004. IEEE.
8. Xiao, Z., et al., *Hierarchical classification of emotional speech*. *IEEE Transactions on Multimedia*, 2007. **37**.
9. Burkhardt, F., et al. *A database of German emotional speech*. in *Ninth European Conference on Speech Communication and Technology*. 2005.
10. House, A.S. and E.P. Neuburg, *Toward automatic identification of the language of an utterance. I. Preliminary methodological considerations*. *The Journal of the Acoustical Society of America*, 1977. **62**(3): p. 708-713.
11. Muthusamy, Y.K. and R.A. Cole. *Automatic segmentation and identification of ten languages using telephone speech*. in *Second International Conference on Spoken Language Processing*. 1992.
12. Hansen, J.H. and L.M. Arslan. *Foreign accent classification using source generator based prosodic features*. in *1995 International Conference on Acoustics, Speech, and Signal Processing*. 1995. IEEE.
13. Bocklet, T., et al. *Age and gender recognition for telephone applications based on gmm supervectors and support vector machines*. in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2008. IEEE.
14. Dobry, G., et al., *Supervector dimension reduction for efficient speaker age estimation based on the acoustic speech signal*. *IEEE Transactions on Audio, Speech, and Language Processing*, 2011. **19**(7): p. 1975-1985.
15. Přibíl, J., A. Přibílová, and J. Matoušek, *GMM-based speaker age and gender classification in Czech and Slovak*. *Journal of Electrical Engineering*, 2017. **68**(1): p. 3-12.
16. Breiman, L., *Random forests*. *Machine learning*, 2001. **45**(1): p. 5-32.

17. Liaw, A., *Documentation for R package randomForest*. PDF). Retrieved, 2013. **15**: p. 191.
18. Ho, Tin Kam (1995). "Random Decision Forest". Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.
- 19.. Ho, Tin Kam (1998). "The Random Subspace Method for Constructing Decision Forests" *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (8): 832– 844. doi:10.1109/34.709601
20. Amit, Yali; Geman, Donald (1997). "Shape quantization and recognition with randomized trees". *Neural Computation* 9 (7): 1545–1588. doi:10.1162/neco.1997.9.7.1545.
21. Kleinberg, Eugene (1996). "An Overtraining-Resistant Stochastic Modeling Method for Pattern Recognition". *Annals of Statistics* 24 (6): 2319–2349. doi:10.1214/aos/1032181157. MR 1425956.
22. Adi, K., et al., *Identifying the developmental phase of Plasmodium falciparum in malaria-infected red blood cells using adaptive color segmentation and back propagation neural network*. *Int J Appl Eng Res*, 2016. **11**: p. 8754-9.
23. LI Jing (1 April 2015). "SEEM 3430 Tutorial: Decision Tables" (PDF). p. 23. Retrieved 11 November 2017.
24. "Creating a Decision Table in Business Rules". Oracle Help Center. 6 August 2017. Retrieved 11 November 2017.
25. Ross, Ronald G. (2005). "Decision Tables, Part 2 ~ The Route to Completeness". *Business Rules Journal*. 6 (8). Retrieved 11 November 2017.
26. Snow, Paul (19 July 2012). "Decision Tables". *DTRules: A Java Based Decision Table Rules Engine*. Retrieved 11 November 2017.
27. LI Jing 2015, p. 24-25.