# Mawdoo3 AI Task (Solution)

## Specify the required data; size; type; resources:

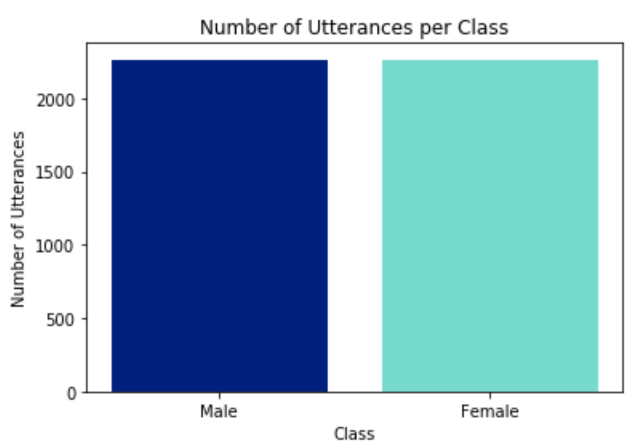
The data must contain speech features that differs between males and females.
The size of the data must be at least 2,000 utterances, or 2 hours.
Resources: Data can be collected from several free resources like:

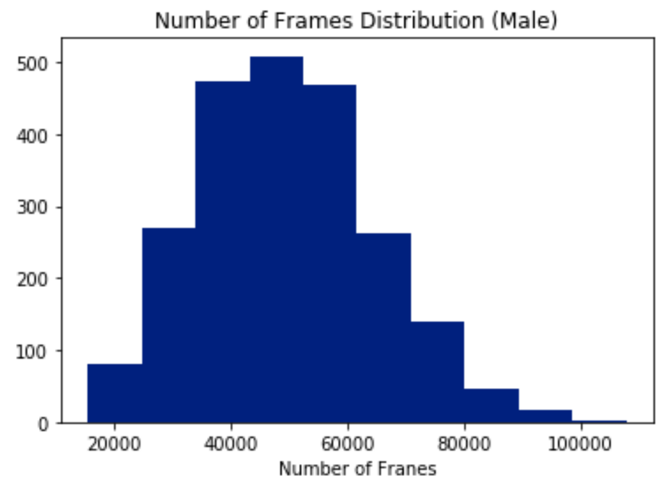
1. Kaggle.
2. CMU_ARCTIC databases.
3. VoxForge repository.

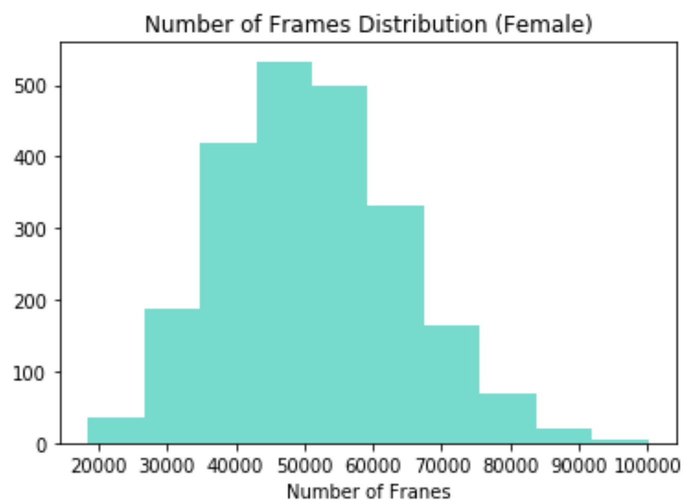
## Crawl the speech data from any open source resources:

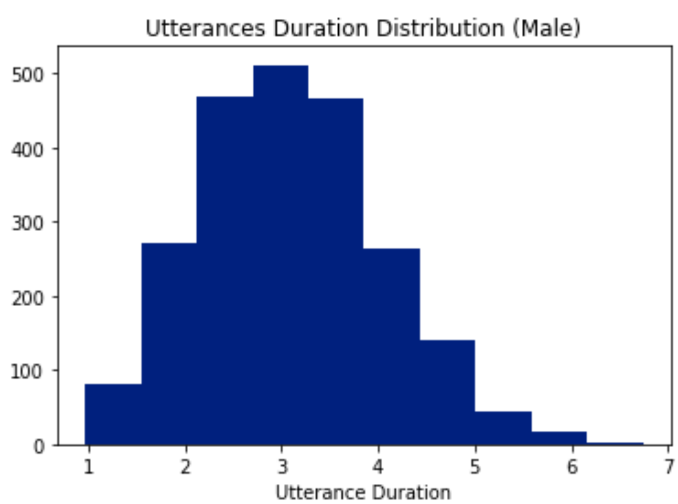
In my solution, I will use CMU_ARCTIC databases ([http://festvox.org/cmu_arctic](http://festvox.org/cmu_arctic)):
4 databases were extracted from this website (2 for male, and 2 for female), each database contains 1132 utterances in english.
File Name: Crawl_Data.py.

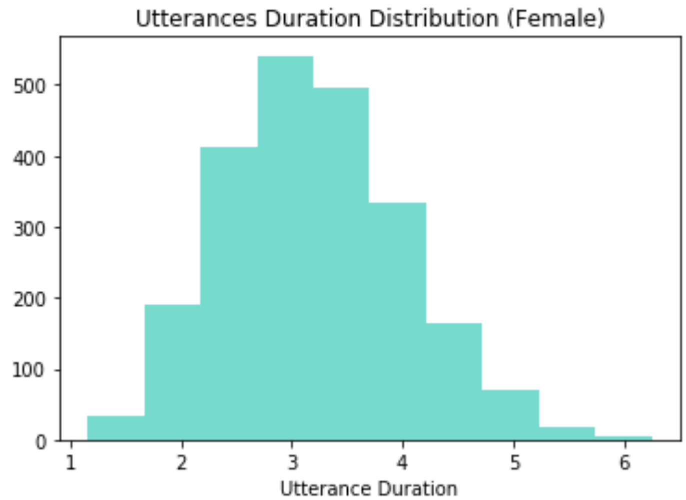
## Provide some exploratory analysis on the data with visualizations:




Number of Frames Distribution (Male)
500
400
300
200
100
0
20000
40000
60000
80000
100000
Number of Franes


Number of Frames Distribution (Female)
500
400
300
200
100
0
20000
30000
40000
50000
60000
70000
80000
90000
100000
Number of Franes


Utterances Duration Distribution (Male)
500
400
300
200
100
0
1
2
3
4
5
6
7
Utterance Duration




## Extract features from the crawled data:

File name: Features_Extraction.py.
Features are saved in all_features.csv.

## Build multiple models/solutions to detect the gender of the speaker:

1. **Recurrent Neural Network (RNN)**

   RNN is known to have better performance than other types of neural networks with speech use cases.

2. **Multilayer Perceptrons (MLPs)**

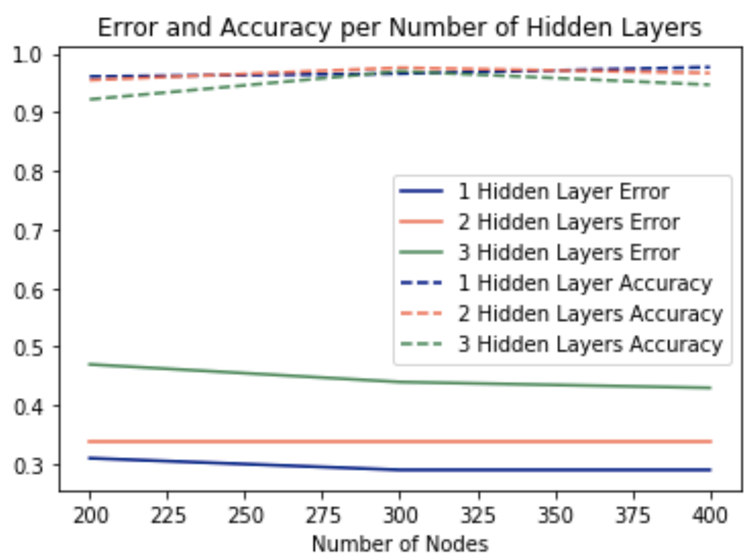
   Several published researches concludes that MLPs have good performance compared to other types of Neural Networks in gender detection use case.

## Compare the implemented models/solutions:

For both networks I specified the following:
Activation Function:
Hidden Layers: tanh.
Output Layer: sigmoid.
Optimizer: Gradient Descent.
Loss Function: Binary Cross Entropy.

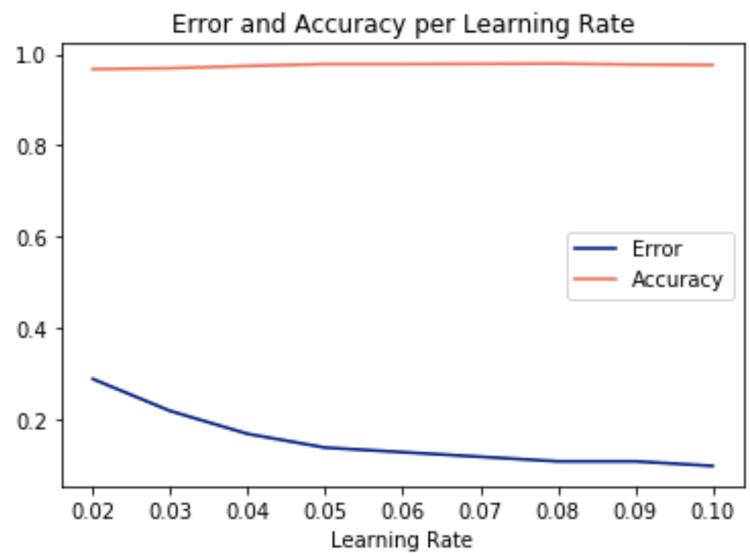
### 1. Recurrent Neural Network (RNN)

As RNN needed time to converge, epochs number was assigned to 100.
First, I specified the learning rate to 0.02 in order to compare the performance for different number of hidden layers with different number of nodes. The results are shown in the figure below.




As shown the model best performance was with 1 hidden layer and 400 nodes which achieved 0.29 Error, and 96.7% Accuracy.

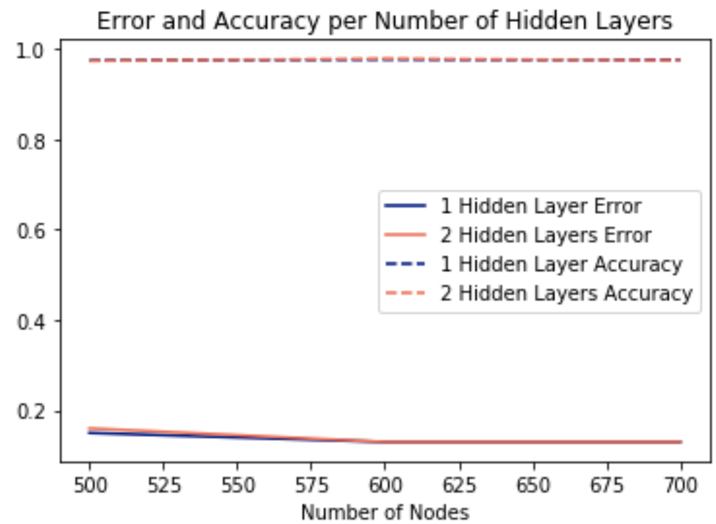
Therefore, I tested the model behavior with different learning rates as shown in the figure below.



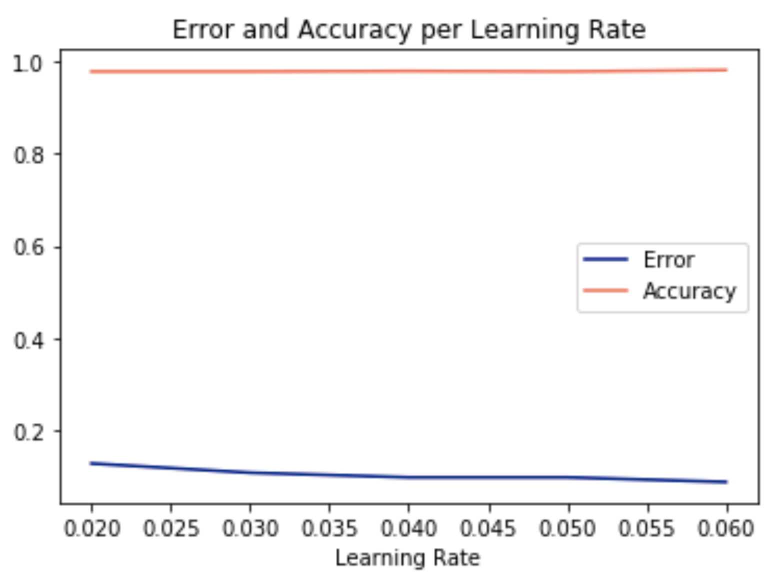

The best model was with 0.08 learning rate which achieved 0.11 error and 97.9 % Accuracy.
Model File Name: Recurrent_model.py
Training File Name: Recurrent_training.py

## 2. Multilayer Perceptrons (MLPs)

MLPs needed less time to converge, so epochs number was assigned to 50.
As in RNN experiments at first, I specified the learning rate to 0.02 in order to compare the performance for different number of hidden layers with different number of nodes. The results are shown in the figure below.




Models with different number of layers have almost the same results. Since training the model with 1 hidden layer needs less time and computation power, I will investigate the model further with 600 nodes, and different learning rates, as shown in the figure below.




The best model was with 0.06 learning rate was which achieved 0.09 error and 98% Accuracy.
Model File Name: MSPs_model.py
Trining File Name: MSPs_training.py

## Serve the resulted model in a Rest API

After tunning both models MLPs had better error rate and accuracy than RNN, therefore MLPs model will be wrapped with REST API using flask-restfull, the trained model is saved in ‘Gender_Detection_MLPs.pth’.
File Name: api.py

In terminal type the following commands to run the app:
python api.py
curl http://localhost:5000/predict -d "data=-0.0620, 0.6383, -0.7371, -0.8406, 0.0942, -0.1965, -0.4943, -1.2131, 0.0594, -0.9031, -1.0989, -0.9497" -X PUT
curl http://localhost:5000/predict

## Create a Docker Image for your application

Docker File Name: Dockerfile.