Problem 2: Simple speech recognition system:

In this problem I decided to divide the code into 4 parts:

- Initialize the variables and matrices.
- Read the train data then extract the features from it.
- Train the data by KNN.
- Testing data.

Here I have 40 training audio files 10 for each word (move, left, stop and right) when I read the audio I extended it by zeroes till (2)^16 cause when you enter a 2 power number it make it faster while extracting the features.

Then I extracting the features with MFCC that has 14 output features (energy in each frame, log power, DFT….etc.), and pitch which is a perceived tone frequency of a sound in comparison with the perceptively best match with a pure sinusoid, They take the audio and divided it into windows or frames, then calculate the features for this frame then shifted it to calculate the features of the next window.

After this I have 15 features with 15 columns so I used a built in function called 'horzcat' to join them in one column the transpose it to have one row vector for each word, after this block of code I have finally 40 training feature row vectors, that would be the training data that I will use in the next code segment.

The third part of my code is training the data, I tried to train it using KNN, KNN is a famous machine learning technic, it takes the training data and plot them, then when you get the testing data and want to predict them it plots them and takes each point of testing data and measures the distance between the testing data and a number of neighbours from training data points you set before, and count the number and type of each word of the neighbours, then vote between them to have the resulted word type which takes the maximum number of voting, as example I vote from 10 neighbours 4=>move 3=>stop 2=>left and 1=>right, so the test word is move.

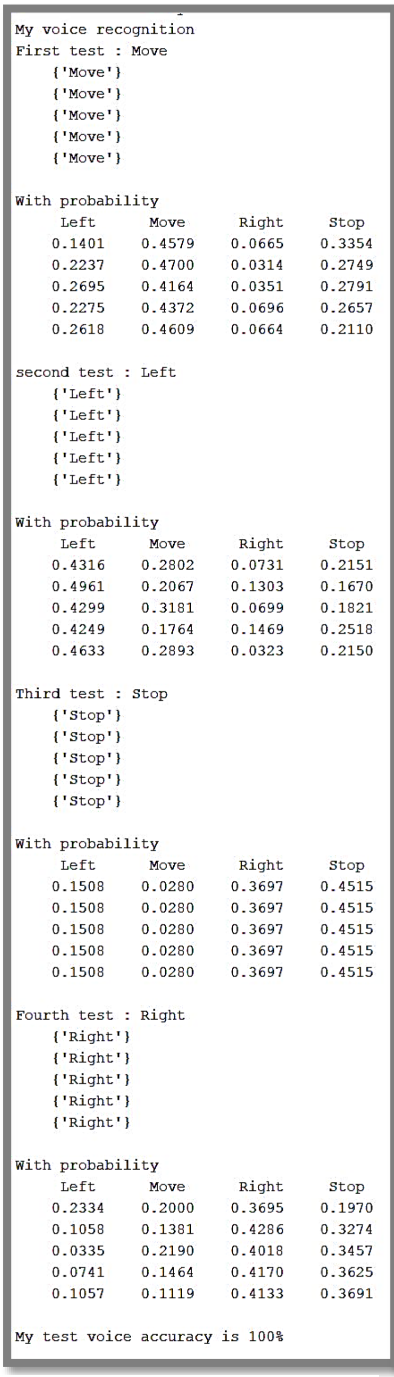
In our code I made the number of neighbourhood is 25 and measure the distance by a method called 'spearman' with this specifications I get the maximum accuracy.

The last part in my code is the testing part, I repeat the second part again, get the audio and make it 2^16 then extract features and finally get the feature vectors one for each testing word, then use the built in function ‘predict’ it returns an array of strings in there is the final prediction of each test word and a matrix in it the probability of being one of any word (move, left, right, stop).

In the test data I use three voices my voice and two of my friends (Ahmed Mahmoud Abusaif and Ahmed Amr), they said every word five times, and I get an accuracy 100% in my test voice and 85% in my friend’s voice ‘Abusaif’ and 75% in my other friend’s voice ‘Amr’, and I want to mansion that I tested my dad voice, too, and get an accuracy 50%.

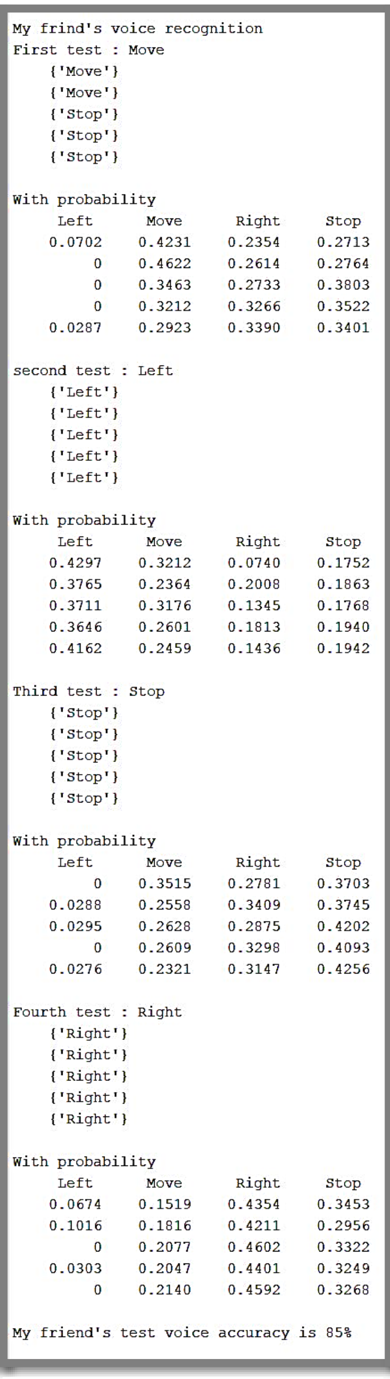
From the result above, I understand that to improve the model accuracy we need to get more voices for males and females with different ages, my dad has a harder voice than me which translated to the lowest accuracy, I tried to improve the accuracy by saying the testing words in different accents as example in move I said it with stressed on ‘m’ then stressed on ‘v’ then said as fast as possible then with a slow rate and raised my voice then make it with a lower tone, to cover all possible ways to say move, then made the same thing in the other words.

I will show the result in the three figures blow.



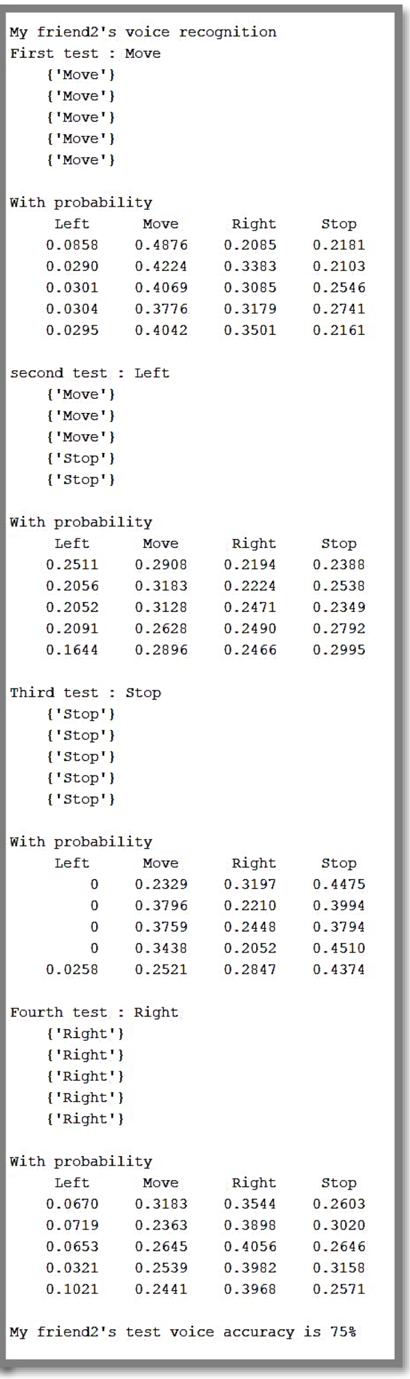
```
My voice recognition
First test : Move
    {'Move'}
    {'Move'}
    {'Move'}
    {'Move'}
    {'Move'}

With probability
     Left      Move      Right     Stop
    0.1401    0.4579    0.0665    0.3354
    0.2237    0.4700    0.0314    0.2749
    0.2695    0.4164    0.0351    0.2791
    0.2275    0.4372    0.0696    0.2657
    0.2618    0.4609    0.0664    0.2110

second test : Left
    {'Left'}
    {'Left'}
    {'Left'}
    {'Left'}
    {'Left'}

With probability
     Left      Move      Right     Stop
    0.4316    0.2802    0.0731    0.2151
    0.4961    0.2067    0.1303    0.1670
    0.4299    0.3181    0.0699    0.1821
    0.4249    0.1764    0.1469    0.2518
    0.4633    0.2893    0.0323    0.2150

Third test : Stop
    {'Stop'}
    {'Stop'}
    {'Stop'}
    {'Stop'}
    {'Stop'}

With probability
     Left      Move      Right     Stop
    0.1508    0.0280    0.3697    0.4515
    0.1508    0.0280    0.3697    0.4515
    0.1508    0.0280    0.3697    0.4515
    0.1508    0.0280    0.3697    0.4515
    0.1508    0.0280    0.3697    0.4515

Fourth test : Right
    {'Right'}
    {'Right'}
    {'Right'}
    {'Right'}
    {'Right'}

With probability
     Left      Move      Right     Stop
    0.2334    0.2000    0.3695    0.1970
    0.1058    0.1381    0.4286    0.3274
    0.0335    0.2190    0.4018    0.3457
    0.0741    0.1464    0.4170    0.3625
    0.1057    0.1119    0.4133    0.3691

My test voice accuracy is 100%
```

Figure 1: My voice recognition



```
My frind's voice recognition
First test : Move
    {'Move'}
    {'Move'}
    {'Stop'}
    {'Stop'}
    {'Stop'}

With probability
     Left      Move      Right     Stop
    0.0702    0.4231    0.2354    0.2713
         0    0.4622    0.2614    0.2764
         0    0.3463    0.2733    0.3803
         0    0.3212    0.3266    0.3522
    0.0287    0.2923    0.3390    0.3401

second test : Left
    {'Left'}
    {'Left'}
    {'Left'}
    {'Left'}
    {'Left'}

With probability
     Left      Move      Right     Stop
    0.4297    0.3212    0.0740    0.1752
    0.3765    0.2364    0.2008    0.1863
    0.3711    0.3176    0.1345    0.1768
    0.3646    0.2601    0.1813    0.1940
    0.4162    0.2459    0.1436    0.1942

Third test : Stop
    {'Stop'}
    {'Stop'}
    {'Stop'}
    {'Stop'}
    {'Stop'}

With probability
     Left      Move      Right     Stop
         0    0.3515    0.2781    0.3703
    0.0288    0.2558    0.3409    0.3745
    0.0295    0.2628    0.2875    0.4202
         0    0.2609    0.3298    0.4093
    0.0276    0.2321    0.3147    0.4256

Fourth test : Right
    {'Right'}
    {'Right'}
    {'Right'}
    {'Right'}
    {'Right'}

With probability
     Left      Move      Right     Stop
    0.0674    0.1519    0.4354    0.3453
    0.1016    0.1816    0.4211    0.2956
         0    0.2077    0.4602    0.3322
    0.0303    0.2047    0.4401    0.3249
         0    0.2140    0.4592    0.3268

My friend's test voice accuracy is 85%
```

Figure 2: Abusaif voice recognition



```
My friend2's voice recognition
First test : Move
    {'Move'}
    {'Move'}
    {'Move'}
    {'Move'}
    {'Move'}

With probability
     Left      Move      Right     Stop
    0.0858    0.4876    0.2085    0.2181
    0.0290    0.4224    0.3383    0.2103
    0.0301    0.4069    0.3085    0.2546
    0.0304    0.3776    0.3179    0.2741
    0.0295    0.4042    0.3501    0.2161

second test : Left
    {'Move'}
    {'Move'}
    {'Move'}
    {'Stop'}
    {'Stop'}

With probability
     Left      Move      Right     Stop
    0.2511    0.2908    0.2194    0.2388
    0.2056    0.3183    0.2224    0.2538
    0.2052    0.3128    0.2471    0.2349
    0.2091    0.2628    0.2490    0.2792
    0.1644    0.2896    0.2466    0.2995

Third test : Stop
    {'Stop'}
    {'Stop'}
    {'Stop'}
    {'Stop'}
    {'Stop'}

With probability
     Left      Move      Right     Stop
         0    0.2329    0.3197    0.4475
         0    0.3796    0.2210    0.3994
         0    0.3759    0.2448    0.3794
         0    0.3438    0.2052    0.4510
    0.0258    0.2521    0.2847    0.4374

Fourth test : Right
    {'Right'}
    {'Right'}
    {'Right'}
    {'Right'}
    {'Right'}

With probability
     Left      Move      Right     Stop
    0.0670    0.3183    0.3544    0.2603
    0.0719    0.2363    0.3898    0.3020
    0.0653    0.2645    0.4056    0.2646
    0.0321    0.2539    0.3982    0.3158
    0.1021    0.2441    0.3968    0.2571

My friend2's test voice accuracy is 75%
```

Figure 3: Amr voice recognition