ME 466 Introduction to AI
Fall 2021
Take-home Final Exam
Submitted on: 19 January 2022

Name: Korkut Emre ARSLANTÜRK

Student ID:250206039

Grade:

1	
2	
3	
4	
5	
6	
7	
Σ	

I hereby declare that the paper I am submitting under this cover is product of my own efforts only. Even if I worked on some of the problems together with my classmates, I prepared this paper on my own, without looking at any other classmate's paper. I am knowledgeable about everything that is written under this cover, and I am prepared to explain any scientific/technical content written here if a short oral examination about this paper is conducted by the instructor. I am aware of the serious consequences of cheating.

Signature:

BODY OF THE REPORT

Question 1

Prove the following statements:

a) Let $a \neq 1$ be a complex number. Show that $\sum_{n=0}^{N-1} a^n = \frac{1-a^N}{1-a}$.

Using hint, let $a^n = \begin{bmatrix} a^0 \\ a^1 \\ \vdots \\ a^{N-1} \end{bmatrix}$ be one vectors in N-dimensional complex space C^N**.**

The fact is that geometric series can be calculated like that:

$$R_n = \sum_{n=0}^{N-1} a^n = 1 + a^1 + a^2 + \cdots + a^{N-1} \quad (1.1)$$

Let's say,

$$X = \sum_{n=0}^{N-1} a^n \quad (1.2)$$

Then, it can be seen:

$$a.\boldsymbol{X} = \sum_{n=0}^{N} a^n \quad (1.3)$$

Thus, it can be observed:

$$\boldsymbol{X} - a.\boldsymbol{X} = (1 + a^1 + a^2 + a^3 + \cdots + a^{N-1}) - (a^1 + a^2 + a^3 + a^4 + \cdots + a^N)$$

By simplifying,

$$\boldsymbol{X}(1-a) = 1 - a^N \quad (1.4)$$

Then,

$$\boldsymbol{X} = \frac{1-a^N}{1-a} \quad (1.5)$$

As a result,

$$\boldsymbol{X} = \sum_{n=0}^{N-1} a^n = \frac{1-a^N}{1-a}$$

b) Let m be an integer. Show that $\boldsymbol{e}^{\frac{\boldsymbol{j2\pi m}}{\boldsymbol{N}}} = \boldsymbol{e}^{\boldsymbol{j}\frac{\boldsymbol{2\pi(m \bmod N)}}{\boldsymbol{N}}}$.

Let's say, A=m(mod N).

So, it can be written,

A=x.N+m, for x ∈ IR

Using that equation,

$$e^{\frac{j2\pi A}{N}} = e^{\frac{j2\pi(yN+m)}{N}} = e^{\frac{j2\pi m}{N}}\, e^{\frac{j2\pi yN}{N}} \tag{1.6}$$

To simplify,

$$e^{\frac{j2\pi yN}{N}} = e^{j2\pi y} \tag{1.7}$$

Using the Euler Transform,

$$e^{j2\pi y} = \cos(2\pi y) + jsin(2\pi y).$$

Also, it is the fact that, y is an integer. Therefore, using trigonometric formulas,

$$\cos(2\pi y) = 1$$

$$jsin(2\pi y)=0$$

As a result,

$$e^{\frac{j2\pi yN}{N}}=e^{j2\pi y} = 1 \tag{1.8}$$

Finally,

$$e^{\frac{j2\pi A}{N}} = e^{j\frac{2\pi m}{N}}$$

So it is verified the desired equation since it is true for every A, where A = m(mod N).

c) Let m be an integer between 0 and N-1. Show that $\sum_{n=0}^{N-1} e^{j\frac{2\pi m}{N}n}=\begin{cases} N, & if\ m = 0 \\ 0, if\ m = 1,2,..,N-1 \end{cases}$

There are 2 cases should be considered.

Case 1(For m=0):

$$\sum_{n=0}^{N-1} e^{\frac{j2\pi mn}{N}} = \sum_{n=0}^{N-1} e^{0}$$

$$= \sum_{n=0}^{N-1} 1 = (1 + 1 + \cdots + 1) = N$$

Case 2(For m=1, 2,…,N-1):

$$\sum_{n=0}^{N-1} e^{\frac{j2\pi mn}{N}}$$

$$= 1 + e^{\frac{j2\pi m}{N}} + \cdots + e^{\frac{j2\pi(N-1)m}{N}}$$

There is a geometric serie. Thus, it can be shown like that,

$$\sum_{n=0}^{N-1} x^n = \frac{1-x^n}{1-x}, \text{ where } x = e^{\frac{j2\pi nm}{N}} \quad and \quad x^n = e^{j2\pi nm}$$

$$\sum_{n=0}^{N-1} e^{\frac{j2\pi mn}{N}} = \frac{1-e^{j2\pi mn}}{1-e^{\frac{j2\pi nm}{N}}}$$

If $e^{\frac{j2\pi nm}{N}} \neq 1$, equation may be undefined. Thus, it cannot be. Moreover, using the Euler transform,

$$e^{j2\pi nm} = \cos(2\pi nm) + jsin(2\pi nm)$$

Using the trigonometric properties,

$$\cos(2\pi nm) = 1$$

$$jsin(2\pi nm) = 0$$

Finally,

$$e^{j2\pi nm} = 1$$

Using that equation that can be written,

$$\sum_{n=0}^{N-1} e^{\frac{j2\pi mn}{N}} = \frac{1-1}{1-e^{\frac{j2\pi nm}{N}}} = 0 \quad \text{where, m = 1, 2, 3…, N-1}$$

Thus, it is verified,

$$\sum_{n=0}^{N-1} e^{\frac{j2\pi mn}{N}} = \begin{cases} N, & if\ m = 0 \\ 0, if\ m = 1,2,3,..,N-1 \end{cases}.$$

Question 2

Show that $q_k.q_l = q_l^H q_k = \sum_{n=0}^{N-1} e^{\frac{j(2\pi k)}{N}n} e^{-\frac{j(2\pi l)}{N}n} = \begin{cases} N \; if \; k = l \\ 0 \; if \; k \neq l \end{cases}$ where k and l are two integers between 0 and N-1. This means that q_k and q_l are orthogonal if k ≠l.

To confirm that the two equations are equivalent, the complex conjugate transpose of the latter of the vectors must be produced. As a result, the vector's position will be changed.

$$q_k.q_l = q_l^H q_k = \sum_{n=0}^{N-1} e^{\frac{j(2\pi k)}{N}n} e^{-\frac{j(2\pi l)}{N}n}$$

To satisfy that condition, two conditions should be studied separately.

Case 1:

Let's first analyze that condition k=l.

Thus,

$$e^{\frac{j(2\pi kn)}{N}} e^{-\frac{j(2\pi kn)}{N}} = e^0 = 1 \tag{2.1}$$

Subsequently,

$$\sum_{n=0}^{N-1} e^{\frac{j(2\pi kn)}{N}} e^{-\frac{j(2\pi ln)}{N}} = \sum_{n=0}^{N-1} 1 + 1 + \cdots + 1 = N \tag{2.2}$$

Hence, it can be seen, if k equals l:

$$\sum_{n=0}^{N-1} e^{\frac{j(2\pi kn)}{N}} e^{-\frac{j(2\pi ln)}{N}} = N$$

Case= 2

Let's analyze that condition k≠l.

In that condition both of k and l are integers between 0 to N-1. Also, k-l is not equals 0. So range of k-l can be determined like that:

$$N - 1 > k - l > -(N-1), k - l \neq 0 \begin{cases} N - 1 > k > 0 \\ 0 > -l > -(N-1) \end{cases}$$

$$\sum_{n=0}^{N-1} e^{\frac{j(2\pi kn)}{N}} e^{-\frac{j(2\pi ln)}{N}} = \sum_{n=0}^{N-1} e^{\frac{j(2\pi(k-1)n)}{N}} = 1 + e^{\frac{j(2\pi(k-l))}{N}} + \cdots + e^{\frac{j(2(N-1)\pi(k-l))}{N}} \tag{2.3}$$

So, there is a geometric serie and it can be shown like that:

$$\sum_{n=0}^{N-1} x^n = \frac{1-x^n}{1-x} \text{ , where X= } e^{\frac{j2\pi(k-l)}{N}} \text{ and } X^n = e^{j2\pi n(k-l)} \tag{2.4}$$

Thus, it can be observed,

$$\sum_{n=0}^{N-1} e^{\frac{j(2\pi kn)}{N}} e^{-\frac{j(2\pi ln)}{N}} = \sum_{n=0}^{N-1} e^{\frac{j(2\pi(k-1)n)}{N}} = \frac{1-e^{j2\pi n(k-l)}}{1-e^{\frac{j2\pi n(k-l)}{N}}}$$

Also, $e^{\frac{j2\pi n(k-l)}{N}} \neq 1$, as expresses, because it makes the geometric series expression of the equation undefined.

Additionally,

$$\left(e^{\frac{j(2\pi(k-l))}{N}}\right)^{N} = e^{\frac{j(2\pi(k-l)N)}{N}} = e^{2\pi j(k-l)} \tag{2.5}$$

Moreover, using the Euler transform,

$$e^{j2\pi n(k-l)} = \cos(2\pi n(k-l)) + j\sin(2\pi n(k-l)) \tag{2.6}$$

Using the trigonometric properties,

$$\cos(2\pi n(k-l))=1$$

$$j\sin(2\pi n(k-l))=0$$

Finally,

$$e^{j2\pi n(k-l)} = 1+j0=1$$

Putting the values we found in place,

$$\sum_{n=0}^{N-1} e^{\frac{j2\pi(k-l)n}{N}} = \frac{1-1}{1-e^{\frac{j2\pi n(k-l)}{N}}} = 0 \; for \; k \neq l$$

When k does not equal l, those vectors orthogonal since the dot product of the two vectors 0.

Finally, it is verified,

$$q_k.q_l = q_l^H q_k = \sum_{n=0}^{N-1} e^{\frac{j(2\pi k)}{N}n} e^{-\frac{j(2\pi l)}{N}n} = \begin{cases} N \; if \; k = l \\ 0 \; if \; k \neq l \end{cases}$$

Question 3

Determine the DFT matrices for N = 2, N = 4, N = 6, N = 8. (Note that $e^{\frac{j(\pi)}{2}}$ = j, $e^{j\pi}$ = −1, etc.)

A Discrete Fourier Transform is defined as a transformation matrix. It, use for signals by matrix multiplication. It is shown like that:

$$X[k] = \sum_{n=0}^{N-1} x[n]\, w_n^{n_k} \text{ where } w_n^{n_k} = \text{Twiddle Factor} = e^{-\frac{j2\pi nk}{N}}$$

Thus,

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{j2\pi nk}{N}} \quad (3.1)$$

$$X[k] = \left[w_n^{n_k}\right][x_n] \quad (3.2)$$

For N=2:

Using general expression,

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{j2\pi nk}{N}} \text{ where N=2}$$

$$X[k] = \sum_{n=0}^{1} x[n] e^{-\frac{j2\pi nk}{N}}.$$

It can be written as,

$$X[k] = \begin{bmatrix} w_2^0 & w_2^0 \\ w_2^0 & w_2^1 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \end{bmatrix}$$

Using the Euler transform,

$$w_2^0 = e^{-\frac{j2\pi(0)}{2}} = 1\,, w_2^1 = e^{-\frac{j2\pi(1)}{2}} = \cos(\pi) - j\sin(\pi) = -1$$

$$X[k] = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \end{bmatrix}$$

That equation can be obtained. Also, to show general formula of matrices:

$$X[k] = x[0] w_2^{(0)k} + x[1] w_2^{(1)k}$$

$$X[0] = x[0] w_2^0 + x[1] w_2^0$$

$$X[1] = x[0] w_2^0 + x[1] w_2^1$$

For N=4:

Using general expression,

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{j2\pi nk}{N}} \text{ where N=4}$$

$$\boldsymbol{X}[k] = \sum_{n=0}^{3} x[n] e^{-\frac{j2\pi nk}{N}}$$

It can be written as,

$$X[k] = \begin{bmatrix} w_4^0 & w_4^0 & w_4^0 & w_4^0 \\ w_4^0 & w_4^1 & w_4^2 & w_4^3 \\ w_4^0 & w_4^2 & w_4^4 & w_4^6 \\ w_4^0 & w_4^3 & w_4^6 & w_4^9 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

$$w_4^0 = e^{-\frac{j2\pi(0)}{4}} = 1$$

$$w_4^1 = e^{-\frac{j2\pi(1)}{4}} = \cos\left(\frac{\pi}{2}\right) - jsin\left(\frac{\pi}{2}\right) = -j$$

$$w_4^2 = e^{-\frac{j2\pi(2)}{4}} = \cos(\pi) - jsin(\pi) = -1$$

$$w_4^3 = e^{-\frac{j2\pi(3)}{4}} = \cos\left(\frac{3\pi}{2}\right) - jsin\left(\frac{3\pi}{2}\right) = j$$

$$w_4^4 = e^{-\frac{j2\pi(4)}{4}} = \cos(2\pi) - jsin(2\pi) = 1$$

$$w_4^6 = e^{-\frac{j2\pi(6)}{4}} = \cos(3\pi) - jsin(3\pi) = -1$$

$$w_4^9 = e^{-\frac{j2\pi(9)}{4}} = \cos\left(\frac{9\pi}{2}\right) - jsin\left(\frac{9\pi}{2}\right) = j$$

Finally,

$$X[k] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & j \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

For N=6:

Using general expression,

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-\frac{j2\pi nk}{N}} \text{ where N=6}$$

$$X[k] = \sum_{n=0}^{5} x[n]e^{-\frac{j2\pi nk}{N}}$$

It can be written as,

$$X[k] = \begin{bmatrix} w_6^0 & w_6^0 & w_6^0 & w_6^0 & w_6^0 & w_6^0 \\ w_6^0 & w_6^1 & w_6^2 & w_6^3 & w_6^4 & w_6^5 \\ w_6^0 & w_6^2 & w_6^4 & w_6^6 & w_6^8 & w_6^{10} \\ w_6^0 & w_6^3 & w_6^6 & w_6^9 & w_6^{12} & w_6^{15} \\ w_6^0 & w_6^4 & w_6^8 & w_6^{12} & w_6^{16} & w_6^{20} \\ w_6^0 & w_6^5 & w_6^{10} & w_6^{15} & w_6^{20} & w_6^{25} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \end{bmatrix}$$

$$w_6^0 = e^{-\frac{j2\pi(0)}{6}} = 1$$
$$w_6^1 = e^{-\frac{j2\pi(1)}{6}} = \cos\left(\frac{\pi}{3}\right) - jsin\left(\frac{\pi}{3}\right) = 0.5 - 0.866j$$
$$w_6^2 = e^{-\frac{j2\pi(2)}{6}} = \cos\left(\frac{2\pi}{3}\right) - jsin\left(\frac{2\pi}{3}\right) = -0.5 - 0.866j$$
$$w_6^3 = e^{-\frac{j2\pi(3)}{6}} = \cos(\pi) - jsin(\pi) = -1$$
$$w_6^4 = e^{-\frac{j2\pi(4)}{6}} = \cos\left(\frac{4\pi}{3}\right) - jsin\left(\frac{4\pi}{3}\right) = -0.5 + 0.886j$$
$$w_6^5 = e^{-\frac{j2\pi(5)}{6}} = \cos\left(\frac{5\pi}{3}\right) - jsin\left(\frac{5\pi}{3}\right) = 0.5 + 0.886j$$
$$w_6^6 = e^{-\frac{j2\pi(6)}{6}} = \cos(2\pi) - jsin(2\pi) = 1$$
$$w_6^8 = e^{-\frac{j2\pi(8)}{6}} = \cos\left(\frac{8\pi}{3}\right) - jsin\left(\frac{8\pi}{3}\right) = -0.5 - 0.866j$$
$$w_6^9 = e^{-\frac{j2\pi(9)}{6}} = \cos(3\pi) - jsin(3\pi) = -1$$
$$w_6^{10} = e^{-\frac{j2\pi(10)}{6}} = \cos\left(\frac{10\pi}{3}\right) - jsin\left(\frac{10\pi}{3}\right) = -0.5 + 0.866j$$
$$w_6^{12} = e^{-\frac{j2\pi(12)}{6}} = \cos(4\pi) - jsin(4\pi) = 1$$
$$w_6^{15} = e^{-\frac{j2\pi(15)}{6}} = \cos(5\pi) - jsin(5\pi) = -1$$
$$w_6^{16} = e^{-\frac{j2\pi(16)}{6}} = \cos\left(\frac{16\pi}{3}\right) - jsin\left(\frac{16\pi}{3}\right) = -0.5 + 0.866j$$
$$w_6^{20} = e^{-\frac{j2\pi(20)}{6}} = \cos\left(\frac{20\pi}{3}\right) - jsin\left(\frac{20\pi}{3}\right) = -0.5 - 0.866j$$
$$w_6^{25} = e^{-\frac{j2\pi(25)}{6}} = \cos\left(\frac{25\pi}{3}\right) - jsin\left(\frac{25\pi}{3}\right) = 0.5 - 0.866j$$

Finally,

$$X[k] = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0.5 - 0.866j & -0.5 - 0.866j & -1 & -0.5 + 0.886j & 0.5 + 0.886j \\ 1 & -0.5 - 0.866j & -0.5 + 0.886j & 1 & -0.5 - 0.866j & -0.5 + 0.866j \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -0.5 + 0.886j & -0.5 - 0.866j & 1 & -0.5 + 0.866j & -0.5 - 0.866j \\ 1 & 0.5 + 0.886j & -0.5 + 0.866j & -1 & -0.5 - 0.866j & 0.5 - 0.866j \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \end{bmatrix}$$

For N=8:

Using general expression,

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{j2\pi nk}{N}} \text{ where N=8}$$

$$X[k] = \sum_{n=0}^{7} x[n]e^{-\frac{j2\pi nk}{N}}$$

It can be written as,

$$\boldsymbol{X}[\boldsymbol{k}] = \begin{bmatrix} w_8^0 & w_8^0 & w_8^0 & w_8^0 & w_8^0 & w_8^0 & w_8^0 & w_8^0 \\ w_8^0 & w_8^1 & w_8^2 & w_8^3 & w_8^4 & w_8^5 & w_8^6 & w_8^7 \\ w_8^0 & w_8^2 & w_8^4 & w_8^6 & w_8^8 & w_8^{10} & w_8^{12} & w_8^{14} \\ w_8^0 & w_8^3 & w_8^6 & w_8^9 & w_8^{12} & w_8^{15} & w_8^{18} & w_8^{21} \\ w_8^0 & w_8^4 & w_8^8 & w_8^{12} & w_8^{16} & w_8^{20} & w_8^{24} & w_8^{28} \\ w_8^0 & w_8^5 & w_8^{10} & w_8^{15} & w_8^{20} & w_8^{25} & w_8^{30} & w_8^{35} \\ w_8^0 & w_8^6 & w_8^{12} & w_8^{18} & w_8^{24} & w_8^{30} & w_8^{36} & w_8^{42} \\ w_8^0 & w_8^7 & w_8^{14} & w_8^{21} & w_8^{28} & w_8^{35} & w_8^{42} & w_8^{49} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix}$$

$$w_8^0 = e^{-\frac{j2\pi(0)}{6}} = 1$$

$$w_8^1 = e^{-\frac{j2\pi(1)}{8}} = \cos\left(\frac{\pi}{4}\right) - jsin\left(\frac{\pi}{4}\right) = 0.707 - 0.707j$$

$$w_8^2 = e^{-\frac{j2\pi(2)}{8}} = \cos\left(\frac{\pi}{2}\right) - jsin\left(\frac{\pi}{2}\right) = -j$$

$$w_8^3 = e^{-\frac{j2\pi(3)}{8}} = \cos\left(\frac{3\pi}{4}\right) - jsin\left(\frac{3\pi}{4}\right) = -0.707 - 0.707j$$

$$w_8^4 = e^{-\frac{j2\pi(4)}{8}} = \cos(\pi) - jsin(\pi) = -1$$

$$w_8^5 = e^{-\frac{j2\pi(5)}{8}} = \cos\left(\frac{5\pi}{4}\right) - jsin\left(\frac{5\pi}{4}\right) = -0.707 + 0.707j$$

$$w_8^6 = e^{-\frac{j2\pi(6)}{8}} = \cos\left(\frac{3\pi}{2}\right) - jsin\left(\frac{3\pi}{2}\right) = j$$

$$w_8^7 = e^{-\frac{j2\pi(7)}{8}} = \cos\left(\frac{7\pi}{4}\right) - jsin\left(\frac{7\pi}{4}\right) = 0.707 + j0.707$$

$$w_8^8 = e^{-\frac{j2\pi(8)}{8}} = \cos(2\pi) - jsin(2\pi) = 1$$

$$w_8^0 = w_8^8 = w_8^{16} = w_8^{24} = w_8^{32} = w_8^{40} = 1$$

$$w_8^1 = w_8^9 = w_8^{17} = w_8^{25} = w_8^{33} = w_8^{41} = w_8^{49} = 0.707 - 0.707j$$

$$w_8^2 = w_8^{10} = w_8^{18} = w_8^{26} = w_8^{34} = w_8^{42} = -j$$

$$w_8^3 = w_8^{11} = w_8^{19} = w_8^{27} = w_8^{35} = w_8^{43} = -0.707 - 0.707j$$

$$w_8^4 = w_8^{12} = w_8^{20} = w_8^{28} = w_8^{36} = w_8^{44} = -1$$

$$w_8^5 = w_8^{13} = w_8^{21} = w_8^{29} = w_8^{37} = w_8^{45} = -0.707 + 0.707j$$

$$w_8^6 = w_8^{14} = w_8^{22} = w_8^{30} = w_8^{38} = w_8^{46} = j$$

$$w_8^7 = w_8^{15} = w_8^{23} = w_8^{31} = w_8^{39} = w_8^{47} = 0.707 + j0.707$$

$$X[k] = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0.707-0.707j & -j & -0.707-0.707j & -1 & -0.707+0.707j & j & 0.707+j0.707 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & -0.707-0.707j & j & 0.707-0.707j & -1 & 0.707+j0.707 & -j & -0.707+0.707j \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -0.707+0.707j & -j & 0.707+j0.707 & -1 & 0.707-0.707j & j & -0.707-0.707j \\ 1 & j & -1 & -j & 1 & j & -1 & -\mathrm{j} \\ 1 & 0.707+j0.707 & j & -0.707+0.707j & -1 & -0.707-0.707j & -j & 0.707-0.707j \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix}$$

Question 4

In that question, there are two types of coins, w0 and w1. The w0-type coins have a probability p=1/2 of landing heads and they are fair coins. W1-type coins have a probability p=1/4 of landing heads and they are biased coins. Also, fair coins are three times more frequent in the world than biased coins. Someone hands you a coin and wants you to tell them if it is fair or biased.

a) Before making any experiment with the coin, what is your decision? Which category is the coin more likely to belong to? What is your probability of error?

It is the fact that, w0-type(fair) coins are three times more frequent in the world than biased coins. Thus, we can say that, if amount of w1-type is k, amount of w0-type is 3k.

$$\mathrm{P}(w_0) = 0.75 \qquad \mathrm{P}(w_1) = 0.25$$

So, without any experiment, the selected coin is more likely to be a fair coin according to their prior probabilities. Thus, my decision is fair coin.

According to the Bayes Decision Theory,

$$P(error|x) \begin{cases} \mathrm{p}(w_0|\mathrm{x}), \text{if we decide } w_1 \\ \mathrm{p}(w_1|\mathrm{x}), \text{if we decide } w_0 \end{cases}$$

In problem $\mathrm{P}(w_0)$ = 0.75 and $\mathrm{P}(w_1)$ = 0.25. Thus, it can be said that, probability of error equals $\mathrm{p}(w_0|\mathrm{x})$= $\mathrm{p}(w_1)$ =0.25=%25 which equals prior probability of fake coins.

b) To find that, decision boundary should be specified.

Prior probabilities are given:

$$P(w0) = 0.75 \qquad P(w1) = 0.25$$

Also, probability of landing heads is given for each coin:

$$Pw0 = 0.50 \qquad Pw1 = 0.25$$

Using the binomial distribution, the probability of observing k heads in n tosses given in hint as:

$$P(x|w_n)=\binom{n}{k}p^k(1-p)^{n-k} \qquad (4.1)$$

Thus, decision boundary can be specified as,

$$\frac{P(x|W_0).p(w_0)}{P(x)} = \frac{P(x|W_1).p(w_1)}{P(x)} \qquad (4.2)$$

In there, $P(x|w_1)$= Probability of having w_1 and $P(x|w_0)$= Probability of having w_0.

Also, $P(w_1)$= prior probability of w_1 and $P(w_0)$= prior probability of w_0.

By using that equation and hint, that relation can be obtained,

$$\binom{n}{k} p_{w_0}^k (1-p_{w_0})^{n-k} p(w_0) = \binom{n}{k} p_{w_1}^k (1-p_{w_1})^{n-k} p(w_1)$$

By simplifying,

$$p_{w_0}^k (1-p_{w_0})^{n-k} p(w_0) = p_{w_1}^k (1-p_{w_1})^{n-k} p(w_1)$$

Then, substituting the values into this equation,

$$\left(\frac{1}{2}\right)^k x \left(1-\frac{1}{2}\right)^{n-k} x \left(\frac{3}{4}\right) = \left(\frac{1}{4}\right)^k x \left(1-\frac{1}{4}\right)^{n-k} x \left(\frac{1}{4}\right)$$

To obtain an equation that depends on K and N, using ln function,

$$\mathrm{k.}\ln(3) + \ln(3) = n.\,ln(3) - n.\ln(2)$$

By simplifying,

$$k = \frac{n.\,[\ln(3) - \ln(2)] - \ln(3)}{\ln(3)}$$

By calculating ln function values,

k=0.369n-1

Finally, if k<(0.369n)-1 , it means that the coin is fake, on the other hand the coin is fair.

Question 5

In that question, a neural network is designed with two inputs, one hidden layer with two neurons and one output. In that example, bias values and their weights are chosen to be +1. The hyperbolic tangent function is used for both the hidden layer and the output layer. Let $a_1^{(1)} = 0.6$, $a_2^{(1)} = 0.3$, $w_1 = 0.7$, $w_2 = 0.1$, $w_3 = 0.7$, $w_4 = 0.35$, $w_5 = 0.3$, $w_6 = 0.25$.

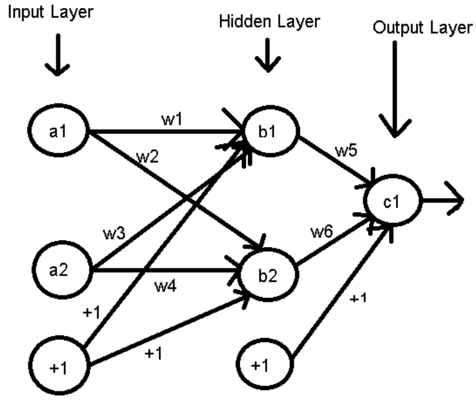

Figure 1: General Neural Network Design

If this diagram is adapted according to the values in the question,

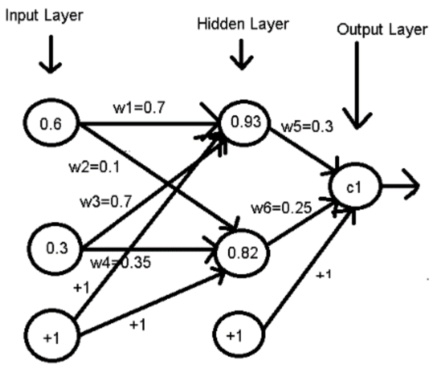

Figure 2: Neural Network Design According to Given Parameters

(a) Perform the forward pass and determine the output $a_1^{(3)}$.

Computation of the output layer values from the input data procedure known as the forward pass. Between the first and the last layer, it traverses through all neurons. In the input layer, the $a_1^{(1)}$, $a_2^{(1)}$ and weights are specified. Thus, in the hidden layer, the neurons will be computed.

$$a_1^{(2)} = tanh(a_1^{(1)}w_1 + a_2^{(1)}w_3 + 1) = \text{tanh } [(0.6\text{x}0.7) + (0.3\text{x}0.7) + 1] = \text{tanh } (1.63) = 0.926$$

$$a_2^{(2)} = tanh(a_1^{(1)}w_2 + a_2^{(1)}w_4 + 1) = \text{tanh } [(0.6\text{x}0.1) + (0.3\text{x}0.35) + 1] = \text{tanh } (1.165) = 0.822$$

As a result, the $\boldsymbol{a_1^{(3)}}$ neuron in the output layer is able to be calculated.

$a_1^{(3)}$= $tanh(a_1^{(2)}w_5 + a_2^{(2)}w_6 + 1)$= tanh [(0.926x0.3) +(0.822x0.25) +1] = tanh (1.48) = 0.902

Finally, it can be shown like that:

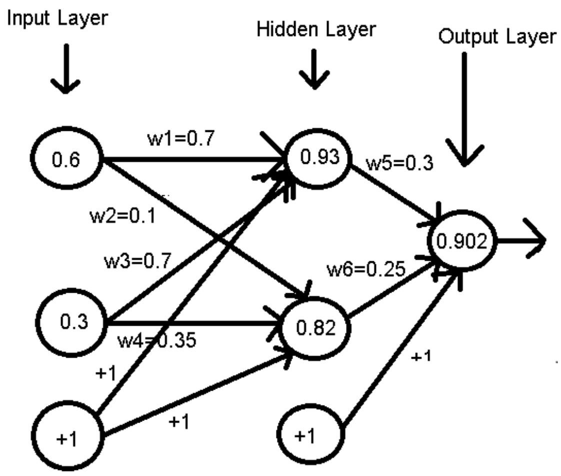

Figure 3: Designed Neural Network

b) The Error defined as E=$\frac{1}{2}(t - a_1^{(3)})^2$ where t is the target output. Also, t=0.7 and earning rate η = 1.

The Chain Rule is utilized to effectively train a neural network using the backpropagation algorithm.

Backpropagation algorithm is used to effectively train a neural network through a method called chain rule. Backpropagation performs a backward pass across a network after each forward pass while modifying the model's parameters like weights and biases.

Using known variables,

$$E = \frac{1}{2}\left(t - a_1^{(3)}\right)^2 = \frac{1}{2} * (0.7 - 0.902)^2 = 0.0204$$

Using Chain Rule.

$$\frac{\partial E}{\partial w_2} = \left(\frac{\partial E}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial a_2^{(2)}}\right)\left(\frac{\partial a_2^{(2)}}{\partial w_2}\right) \qquad (6.1)$$

$$= \left(a_1^{(3)} - t\right) * tanh'\left(a_1^{(2)}w_5 + a_2^{(2)}w_6 + 1\right) * w_6 * \ tanh'\left(a_1^{(1)}w_2 + a_2^{(1)}w_4 + 1\right) * a_1^{(1)}$$

Firstly, w_6 should be found. To find it using the Chain Rule,

$$\frac{\partial E}{\partial w_6} = \left(\frac{\partial E}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_2^{(2)}}{\partial w_6}\right) = \left(a_1^{(3)} - t\right) * tanh'\left(a_1^{(2)} w_5 + a_2^{(2)} w_6 + 1\right) * a_2^{(2)}$$

$$= (0.902 - 0.7) * tanh'\left((0.93x0.3) + (0.82x0.25) + 1\right) * 0.82 = 0.031$$

Now, updated w_6 can be found,

$$w_{6,updated} = w_6 - \eta \frac{\partial E}{\partial w_6} = 0.25 - 1 * 0.031 = 0.219$$

Using Chain Rule,

$$\frac{\partial E}{\partial w_2} = \left(\frac{\partial E}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial a_2^{(2)}}\right)\left(\frac{\partial a_2^{(2)}}{\partial w_2}\right)$$

$$= (0.902 - 0.7) tanh'\left((0.93x0.3) + (0.82x0.25) + 1\right) x\ 0.219\ \ x\ tanh'\left((0.6x0.1) + (0.3x0.35) + 1\right) x\ 0.6 \quad = 0.002$$

Finally, updated w_2 can be found..

$$w_{2,updated} = w_2 - \eta \frac{\partial E}{\partial w_2} = 0.1 - 1 * 0.002 = 0.098$$

Question 6

A neuron with two inputs x_1 and x_2, bias term θ, and output o. A “super-neuron” with the following input-output relation:

$\mathrm{o} = w_1 x_1 + w_{12} x_1 x_2 + w_2 x_2 = \theta$, where w_1, w_2, w_{12} are weight values like an ordinary neuron.

a) The Error defined as $E = \frac{1}{2}(t - \mathrm{o})^2$ where t is the target output. Derive the gradient descent learning equations for this neuron.

First, f(v)= o. Then, the inputs and weight values are determined in matrix form.

$$w = \begin{bmatrix} w_1 \\ w_{12} \\ w_2 \\ \theta \end{bmatrix} \text{ and } x = \begin{bmatrix} x_1 \\ x_{12} \\ x_2 \\ -1 \end{bmatrix}$$

These equations may be used to get updated weights from the gradient descent method.

$$w' = w + \Delta w \qquad (6.1)$$

Then, using the Chain Rule,

$$\Delta w = -\eta \frac{\partial E}{\partial w} = -\eta \left(\frac{\partial E}{\partial \mathrm{o}}\right)\left(\frac{\partial \mathrm{o}}{\partial v}\right)\left(\frac{\partial v}{\partial w}\right) \qquad (6.2)$$

Finally,

$$\Delta \mathrm{w} = -\eta \frac{\partial E}{\partial w} = -\eta(\mathrm{o}-\mathrm{t})x = \begin{bmatrix} x_1 \\ x_1 x_2 \\ x_2 \\ -1 \end{bmatrix}$$

As a result, updated w can be found.

$$w' = w + \Delta \mathrm{w} = \begin{bmatrix} w_1 \\ w_{12} \\ w_2 \\ \theta \end{bmatrix} - \eta(\mathrm{o}-\mathrm{t}) \begin{bmatrix} x_1 \\ x_1 x_2 \\ x_2 \\ -1 \end{bmatrix}$$

b)

Recall that a single ordinary neuron cannot realize the XOR operation since the patterns are not linearly seperable. But this "super-neuron" can. Show how this can be done.

Hint: Consider how ($w_1 x_1 + w_{12} x_1 x_2 + w_2 x_2 - \theta \lesseqgtr 0$) regions look like in the $x_1 - x_2$ plane.)

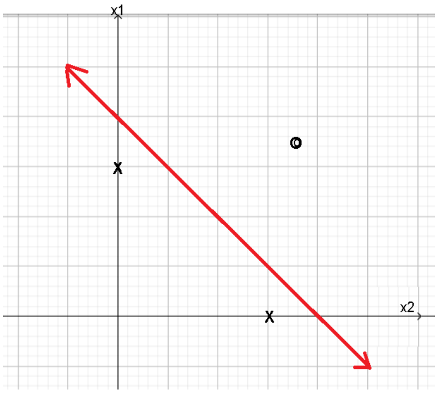

Figure 4: Linear Line Graph

Classes can be separated from each other with a linear line by using one normal neuron as can be seen in Figure 4. In that figures X mark show class 1 which equals 1, and the O mark shows class 2, which equals 0.

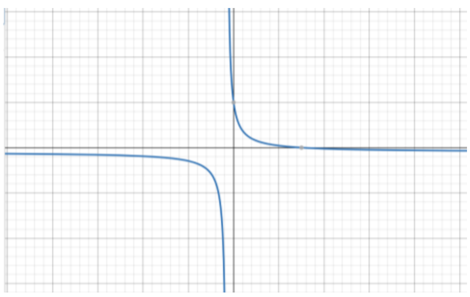

Figure 5: Graph of Equation according to random variables (15=2x+4xy+3y)

On the other hand, a linear line is not useful to use for XOR operation. Using super neuron, (where $w_1 = 2, w_{12} = 4, w_2 = 3$) relation of input and output is drawn. As a result, symmetrical and curved lines are obtained like Figure 5.

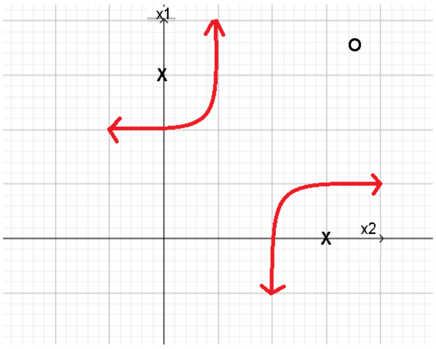

Figure 6: Possible Created Decision Boundary

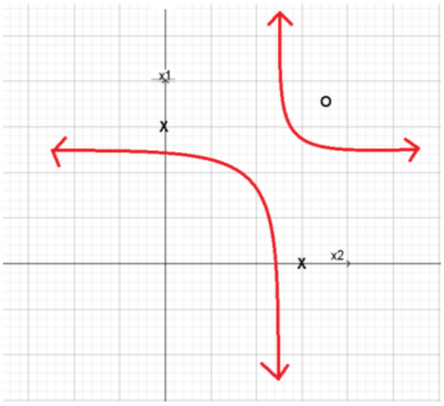

Figure 7: Possible Created Decision Boundary

If it is possible to select weights properly, decision lines can be set like Figure 6 and 7. The reason of those lines are plotted in same colour is that they are from a super neuron.

Question 7

The dataset contains walking data of 7 different people of different genders. Matrix data's each row matches a 5-second recording of the acceleration data, which is sampled at 25 Hertz, as the participant is walking. The participant number is kept in the variable participants for each row. Also, the gender of the participant is kept in the variable gender in the corresponding row. The aim of this study is to design a neural network algorithm to find the user and gender of the participants from the frequency information in the walking data.

It was asked to be created as a vector whose labels have 7 elements and all the elements are -1, which person belongs to that element is 1. For this purpose, the data was requested to be obtained by DFT.

First, the data set was loaded, and the labels were converted to the desired format.

```
load finalq7.mat
Targets=-1*ones(7,420);
for k=1:420
    Targets(participants(k),k)=1;
end
```

a) Then, for gravitational acceleration, the mean was asked to be removed. The mean of each row is subtracted.

```
for j = 1:420
    data(j,:) = data(j,:) - mean(data(j,:));
end
```

b) Then, the calculation of the DFT of each row was performed. Each row consists of 125 features. The FFT function was used to perform the Fourier transform. Since the amplitude information will be needed, the abs function was used and the absolute value was taken. Then, since the negative and positive components on the frequency domain are symmetrical (have the same value as can be seen in Figure 8), only their frequencies on one side were taken. That is, the first 63 elements of the 125-element vector were taken and assigned to the row of the other matrix as can be seen in Figure 9. Thus, each row had its own Fourier transform assigned to the row of another matrix.

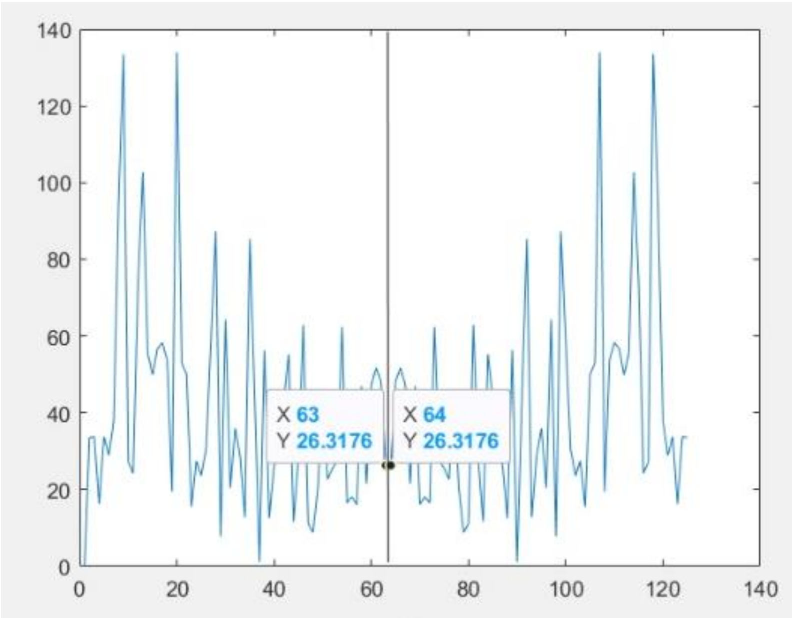

Figure 8: Samples in Frequency Domain

```
DFT_mat = zeros(420,63);
for j = 1:420
    DFT = abs(fft(data(j,:),125));
    DFT_mat(j,:) = DFT(1:63);
end
```

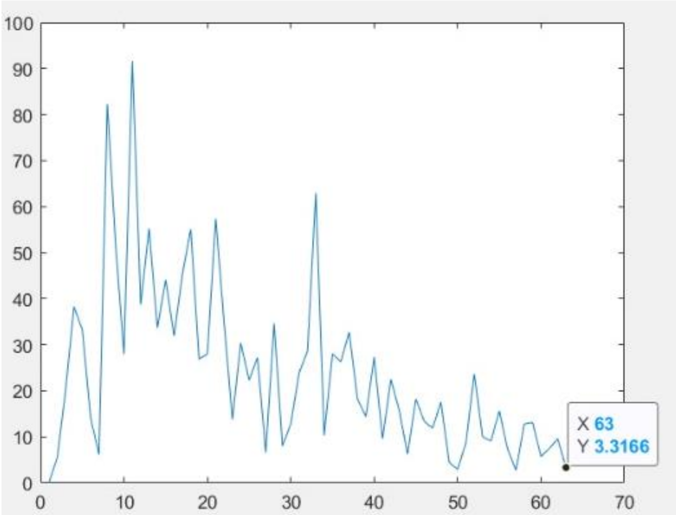

Figure 9: Output of Sample Selection in Frequency Domain

c) After that, a standard normalization process was applied. All the matrix features were decompressed between 0 and 1. The controlled value was subtracted from the minimum value of the row under consideration. The result was divided by the difference between max and min. Thus, all values are normalized between 0 and 1.

```
for i=1:63
    DFT_mat(i,:) = (DFT_mat(i,:) - min(DFT_mat(i,:))) / (max(DFT_mat(i,:)) -
min(DFT_mat(i,:)));
end
```

d)

i)To be able to select random, the random permutation of the instances between 1 and 420 were taken. Then, 250 instances were defined for training, 50 instances for validation, and the remaining 120 instances for testing.

```
idx = randperm(420);
training_idx = idx(1:250);
val_idx = idx(251:300);
test_idx = idx(301:end);
```

Then it is asked to develop a neural network algorithm. Since the feature number is 63, a layer consisting of 63 neurons was defined for the Input layer. Also, because of there are 7 participants, the output layer consists of 7 neurons. For the model to converge faster, the hidden layer was set to a high (150). Also, the learning rate was specified. Then, the Wih and Woh were determined randomly.

In there, multiplied by 0.02 while subtracting 0.5 to get a value between (-0.5, 0.5) . Subsequently, Training, TrainingTargets,Validation, ValidationTargets, Test, TestTargets defined. Also, number of training, validation, test is determined.

```
num_input = 63;
num_hidden = 150;
num_output = 7;
eta=0.0001;

Wih=0.02*(rand(num_input+1,num_hidden)-0.5);
Who=0.02*(rand(num_hidden+1,num_output)-0.5);

f=DFT_mat;
Training= f(training_idx,:)';
TrainingTargets = Targets(:,training_idx);
Validation = f(val_idx,:)';
ValidationTargets = Targets(:,val_idx);
Test = f(test_idx,:)';
TestTargets = Targets(:,test_idx);
num_training=250;
num_validation=50;
num_test=120;
relE=inf;
prevE=inf;
trErrorhistory=[];
valErrorhistory=[];
epoch=0;
```

The validation loss is getting smaller and smaller. Training will stop when the loss starts to increase. If the previous error is less than the current error, it means that the error is increased. Therefore, the condition for continuing while was determined by subtracting the current error from the previous error. For the process does not to take too long time to converge, the equation does was not written with ‘0’ because after values such as 10^{-8}, it starts not to converge. In while loop, allocated data for the train process is specified by their weight and their output. Values of v and o are estimated for the hidden layer. Subsequently, it goes to the output layer from tanh. Also, the error is calculated. The weights are updated with the weight update equations because the desired targets are not output. The derivative of the neuron function is calculated using the tangent hyperbola. Finally, the test is carried out in a single line for validation.

```
while (relE>0.01)
    trE=0;
    for i=1:num_training
        v=Wih'*[Training(:,i);-1];
        o=tanh(v);

        vv=Who'*[o;-1];
        oo=tanh(vv);

        trE=trE+sum((oo-TrainingTargets(:,i)).^2);

        deltao=(oo-TrainingTargets(:,i)).*(1-oo.^2);
        Who=Who+(-eta*[o;-1]*deltao');
```

```
        deltah=(Who*deltao).*(1-[o;-1].^2);
        Wih=Wih+(-eta*[Training(:,i);-1]*deltah(1:end-1)');
    end
    epoch=epoch+1;
    trE=trE/num_training;
    val=tanh(Who'*[tanh(Wih'*[Validation;-1*ones(1,num_validation)]);-
1*ones(1,num_validation)]);
    E=sum(sum((val-ValidationTargets).^2))/num_validation;
    relE=(prevE-E)/E;
    prevE=E;
    trErrorhistory=[trErrorhistory,trE];
    valErrorhistory=[valErrorhistory,E];
    plot(trErrorhistory)
    hold on;
    plot(valErrorhistory)
    hold off;
    pause(0.1)
    title("The Graph of a Loss as a function of Iteration for Participant Estimation")
    xlabel('Iteration')
    ylabel('Loss')
    legend({'Training Loss','Validation Loss'})
end

est=tanh(Who'*[tanh(Wih'*[Test;-1*ones(1,num_test)]);-1*ones(1,num_test)]);
```

Finding the maximum value of each column in the generated data determines the activity. A confusion matrix has been created. Classification accuracy was found by considering the elements in the main diagonal of the confusion matrix.

```
conf=zeros(7);

for k=1:num_test
    [~,I]=max(est(:,k));
    [~,J]=max(TestTargets(:,k));
    conf(J,I)=conf(J,I)+1;
end
conf
temp = 0;
for i = 1:7
    temp = temp + conf(i,i);
end
```

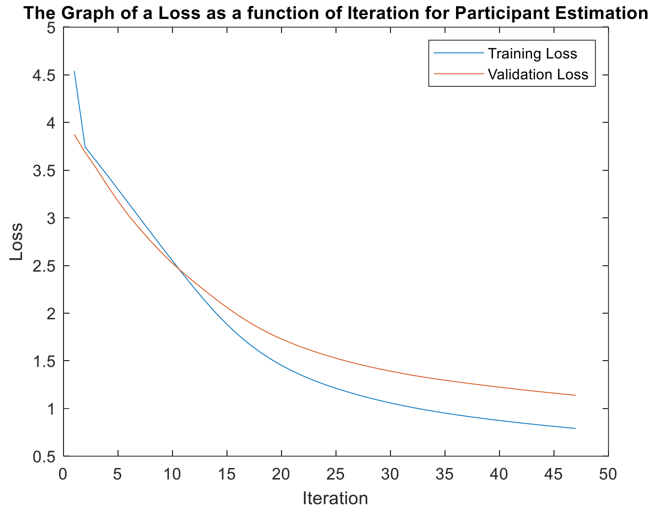

Figure 10: Loss Graph for Participant Estimation

Comparison of validation loss and training loss for participant estimation can be seen in Figure 10. As expected, the validation loss is higher than the training loss since validation loss is specified on all samples. The validation loss is determined over the full validation dataset while the training loss is specified over the entire training data. The size of training set is almost 5 times the validation. Initially, the fact that the training loss is more than the validation loss is due to randomness.

conf =

19	0	0	0	0	0	0
1	16	1	0	0	0	0
0	0	14	0	0	0	1
0	0	0	18	0	0	0
0	0	0	0	13	0	1
0	0	0	0	2	17	0
0	0	0	0	0	0	17

Figure 11: Confusion Matrix for Participant Estimation

The performance of a classification algorithm is described by the confusion matrix. A confusion matrix gives information about right points and types of error of classification model. The participants' activities are shown in Figure 11. It can be seen, the decisions appear to have been accurately made according to numbers in each row.

ii)

In this part, neural network design is desired to estimate the gender of the participants by using 63 features as input. Since many operations are the same as the previous part, not all decode, but the differences between them will be explained. The labels of the previous part were changed and a 2-element label was determined because there are 2 genders. For the male it is -1, for the female, it is 1.

```
Targets=-1*ones(2,420);
```

When the input neuron does not change, the output neuron becomes 2 because of the 2 genders. Accordingly, the model was trained.

```
num_input = 63;
num_hidden = 150;
num_output = 2;
eta=0.0001;
```

Also, the confusion matrix is changed. The confusion matrix is being made 2x2.

```
conf=zeros(2);

for k=1:num_test
    [~,I]=max(est(:,k));
    [~,J]=max(TestTargets(:,k));
    conf(J,I)=conf(J,I)+1;
end
conf
temp = 0;
for i = 1:2
    temp = temp + conf(i,i);
end
```

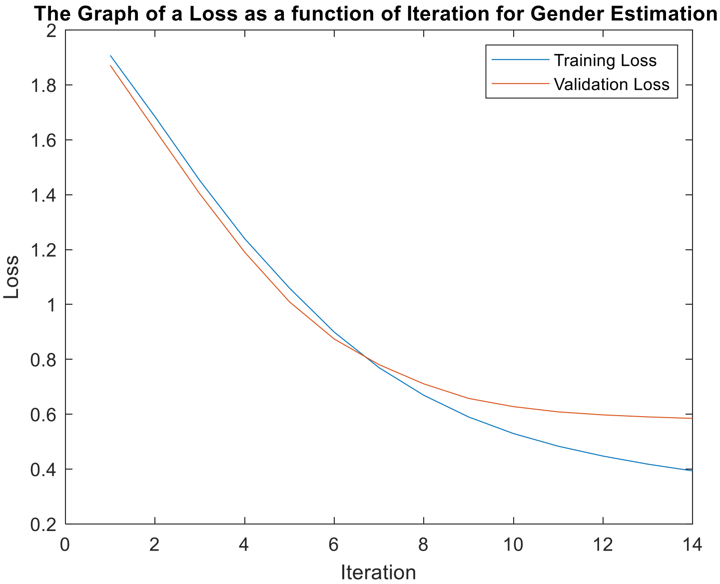

Figure 12: Loss Graph for Gender Estimation

Comparison of validation loss and training loss for gender estimation can be seen in Figure 12. As expected, the validation loss is higher than the training loss since validation loss is specified on all samples. It can be seen, when iteration continues, the validation loss and the training loss converge. Thus, the model will be fit more, when the training loss becomes to be fewer than the validation loss. Initially, the fact that the training loss is more than the validation loss is due to randomness.

```
conf =

    65     9
     3    43
```

Figure 13: Confusion Matrix for Participant Estimation

The performance of a classification algorithm is described by the confusion matrix. The participants' gender type distinction are shown in Figure 13. It can be seen, the decisions appear to have been accurately made according to numbers in each row.

(e)

In this part, a train with a k-nearest neighbor classifier is requested.

First, the data set was loaded. Then, for gravitational acceleration, the mean was asked to be removed. The mean of each row is subtracted. After that, a standard normalization process was applied. All the matrix features were decompressed between 0 and 1. The controlled value was subtracted from the minimum value of the row under consideration. The result was divided by the difference between max and min. Thus, all values are normalized between 0 and 1.

```
load finalq7.mat
for j = 1:420
    data(j,:) = data(j,:) - mean(data(j,:));
end

DFT_mat = zeros(420,63);
for j = 1:420
    DFT = abs(fft(data(j,:),125));
    DFT_mat(j,:) = DFT(1:63);
end

for i=1:63
    DFT_mat(i,:) = (DFT_mat(i,:) - min(DFT_mat(i,:))) / (max(DFT_mat(i,:)) -
min(DFT_mat(i,:)));
end
```

Then, as requested in the question, 300 of the 420 instances were randomly allocated for training, while the rest were used for testing. After that, the labels were kept in true_class.

```
idx = randperm(420);
training_idx = idx(1:300);
test_idx = idx(301:end);
true_class = participants(test_idx);
```

The training data, labels, and K value were given as 5 by calling the "fitcknn" function and entering it. The classifier is returned by the function. Then, the resulting classifier is used in the predict function. Then the confusion matrix has been created. Classification accuracy was found by considering the elements in the main diagonal of the confusion matrix.

```
KNN = fitcknn(DFT_mat(training_idx,:),participants(training_idx)','NumNeighbors',5);
label = predict(KNN,DFT_mat(test_idx,:));
conf = zeros(7);
for i = 1:120
    conf(label(i),true_class(i)) = conf(label(i),true_class(i)) + 1;
end
conf
temp = 0;
for i = 1:7
    temp = temp + conf(i,i);
end
```

```
conf =

    22     1     0     0     0     0     0
     0    13     0     0     0     1     2
     0     0    16     0     0     0     0
     0     0     1    18     0     0     0
     0     0     0     0    13     1     0
     0     2     0     0     0    14     4
     0     0     1     0     0     0    11
```

Figure 14: Confusion Matrix for K-nearest Neighbor Classifier

The performance of a classification algorithm is described by the confusion matrix. The amount of true and false predictions are shown in Figure 14. It can be seen, the choices appear to have been accurately made according to numbers in each row.