

IB Mathematics High Level

Internal Assessment:

---

Mathematics in Perceptron: Distinguishing the  
sounds of two alphabetical characters' voice with  
Perceptron

---

## Contents:

Rationale:.....	1
Introduction: What is Perceptron.....	2
How to train Perceptron:.....	6
Application of Perceptron with two inputs on a toy:.....	10
Application of Perceptron on Audio Recognition:.....	13
What is MATLAB:.....	14
The Application Process:.....	15
Training & Testing Process:.....	17
Conclusion:.....	19
Reflection:.....	19
Bibliography:.....	20

## **Rationale:**

I have chosen my topic for this exploration to understand how significant mathematics is in perceptron (single-layer neural network). Before that, I was planning to explore on multiple neural networked systems which are far more complex than what I am trying to learn for the first time of my academic IB years. The reason why I changed my plan was because multi-layer neural networked systems most commonly known as "deep neural networks" require the understanding of its origin which is the single-layer neural network. Basically, in order to understand deep-neural networks, I must perfectly understand the perceptron. Why I am not going to explore multi-layer neural networks in this investigation is that, it is far above my level which is taught in deep learning and machine learning in Universities. Another reason was that I liked studying with vectors during my IB-HL lessons. So, choosing perceptron will benefit me through not only my IB but in my future career as well. But the big question is, why do I need to understand perceptron when even people can imitate its process. Being growing in a time where the significance of automation is increasing even more is our duty to recognize and help develop that part of mathematics and science. I have always imagined that the future will be mostly dependent on automation so learning perceptron will help me to improve my knowledge in programming and mathematics. To keep this trend going or changing, it is essential to understand a neural network. By these motivations I want to understand its core that is perceptron and explain it in this investigation.

Therefore, I had to search for perceptron's real-life usage which was a difficult task for me since it is only taught in university in deep or machine learning courses and it was always meant to be a simple example of multi-layer neural networks. So, what does perceptron do again? Unlike its more complex counterpart, it functions linearly.

Therefore, a small change of an input in the data set can affect the output as a whole.

This is basically the process of making the simplest decision making ever. In this investigation I will examine how comparably the most pitched Turkish character "İ" differs from "A" in voice when it is digitized in terms of mathematics.

## **Introduction: What is Perceptron**

Perceptron is a method to understand how inputs affect the outcome. The inputs ( $x$ ) that are to be given are multiplied with their corresponding weights ( $w$ ) and their sum has been added with sigma function with also a biased term ( $b$ ) which is any number. The output ( $y$ ) goes through an activation function called "Signum ". The Signum function takes the output and gives out a value that is either a 0 or 1. Since there are only two outcomes, perceptron can only solve linear problems for example: yes or no questions. This in general is called the perceptron. The model of perceptron is given as follows.

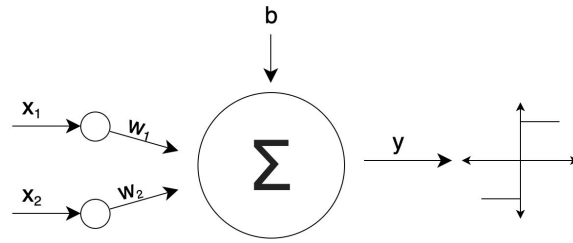


Figure 1: Model of perceptron.

The equation of (y) is as follows where b is a real number.

$$y = b + x_1w_1 + x_2w_2$$

This can also be written as a function.

$$y = f(x_1, x_2; w_1, w_2, b)$$

The inputs x and weights w are defined as vectors. An example for perceptron with two inputs is shown:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad y = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

With using the dot product notation, y can be expressed as follows.

$$y = x^T w + b$$

The (T) is a transpose operation which makes a column into a row and vice versa.

In order to simplify the notation,  $b$  is now assigned to be a new weight ( $w_0$ ) and its corresponding input is ( $x_0$ ) set to be 1. The new model of perceptron is as follows.

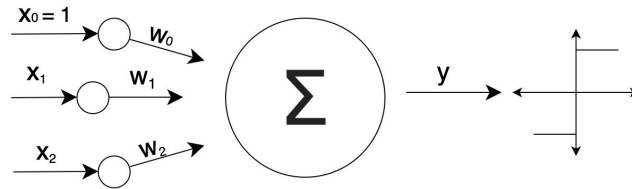


Figure 2: New model of perceptron, the multiplication of corresponding  $x$ s and weights are summed within Sigma producing  $y$ .

The new  $w$  and  $x$  vectors are shown as follows.

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} \quad w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

The new function has no  $b$  in it anymore, it is more simplified.

$$y = x^T w$$

So far,  $w_0$ ,  $w_1$  and  $w_2$  are unknown variables that will be determined. In order to understand the next section which is to understand how to train a perceptron, we must first consider that we have  $m$  number of samples of numerical data. This perceptron is an example with two inputs. Superscript represents the sample index, for example,  $x^{(1)}$  is the first recording which is a column vector. Subscript means the element index of sample recording like the magnitude of an audio wave. For instance,  $x_3^{(5)}$  means the 3rd element of the fifth recording. The outputs are shown in the  $y$  vector.

$$x^{(1)} = \begin{bmatrix} x_0^{(1)} \\ x_1^{(1)} \\ x_2^{(1)} \end{bmatrix}, x^{(2)} = \begin{bmatrix} x_0^{(2)} \\ x_1^{(2)} \\ x_2^{(2)} \end{bmatrix}, \dots, x^{(m)} = \begin{bmatrix} x_0^{(m)} \\ x_1^{(m)} \\ x_2^{(m)} \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 \\ \cdot \\ \cdot \\ \cdot \\ y_m \end{bmatrix}$$

Such that:

$$y_1 = x^{(1)T} w$$

$$y_2 = x^{(2)T} w$$

.

.

.

$$y_m = x^{(m)T} w$$

These  $m$  number of equations can be written in a matrix  $[A]_{m \times 3}$  times the vector  $[w]_{3 \times 1}$

which equals to vector  $[y]_{m \times 1}$ .

$$A = \begin{bmatrix} x^{(1)T} \\ x^{(2)T} \\ \cdot \\ \cdot \\ x^{(m)T} \end{bmatrix}, w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}, y = \begin{bmatrix} y_1 \\ \cdot \\ \cdot \\ \cdot \\ y_m \end{bmatrix}$$

Therefore:

$$y = Aw$$

Matrix A times vector w result in a column vector called y. The matrix (A) contains the samples. For example, the samples in voice recognition are audio recordings. The y vector contains the labels. In voice recognition, the labels would be “0” or “1”. From this point, the main focus becomes how to find w.

## How to train Perceptron:

$$y = Aw$$

The equation above is the first step to train a perceptron. The y variable in this equation, which is a vector, is equal to the dot product of matrix A-which has the inputs inside- and w, which is the vector involving unknown weights. To train a perceptron is the same as solving the above equation for weights. However, depending on what the given A and y are, it can be impossible to even solve for w, so the new question is “Find w such that Aw is close to y”. Since Aw and y are vectors, to understand how close the two vectors are, distance needs to be calculated. The first step to define the distance between two vectors is to take the difference as follows.

$$e = Aw - y$$

The y in the above function is the expected output whereas the Aw is the calculated output. It is logical to call their difference as error. So, we are currently interested in how to make the amount of error smaller, therefore to make a vector smaller, firstly we need to identify its magnitude. The magnitude of a vector is found with the following formula;

$$|x - y| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



Minimizing the norm makes it harder to solve the problem. So instead of norm, norm squared will be easier in the calculations. Instead, the derivative of  $|e|^2$  with respect to  $w$  is going to be taken in order to minimize the error. Since  $|e|^2 = |Aw - y|^2$ , the

derivative that is  $\frac{d|e|^2}{dw}$  can also be shown as follows.

$$\frac{d|e|^2}{dw} = \frac{d|Aw - y|^2}{dw}$$

To find the derivative, the following formula must be noted which explains that squared norm of a vector is equal to the product of the vector and the transpose of the vector.

The order of  $x^T$  and  $x$  in product is important otherwise the output matrix would not be a scalar.

$$|x|^2 = x^T x$$

Therefore, the calculations are as follows.

$$|e|^2 = (Aw - y)^T (Aw - y)$$

Now the transposition will be distributed over  $Aw$  and  $y$ .

$$|e|^2 = ((Aw)^T - y^T)(Aw - y)$$

The following formulae are essential to know.

$$(Ax)^T = x^T A^T$$

Using the above formulae, transpose is distributed inside the parentheses of  $Ax$  so that the order of  $x$  and  $A^T$  changes.

$$|e|^2 = (w^T A^T - y^T)(Aw - y)$$

The expression is expanded as the following:

$$|e|^2 = w^T A^T Aw - w^T A^T y - y^T Aw + y^T y$$

The terms  $w^T A^T y$  and  $y^T Aw$  are scalar and identical that's why the equation can be written as the following:

$$|e|^2 = w^T A^T Aw - 2w^T A^T y + y^T y$$

Before taking the derivative with respect to  $w$  to find the minimum points, these formulas must be known and applied where the resulting derivative will be a vector.

$$\frac{d}{dw} w^T A^T Aw = 2A^T Aw$$

$$\frac{d}{dw} w^T A^T y = A^T y$$

$$\frac{d}{dw} y^T y = 0$$

Now the derivative of the function is taken with respect to  $w$ ;

$$\frac{d|e|^2}{dw} = 2A^T Aw - 2A^T y$$

The derivative of the equation is set to be zero in order to find the minimum points.

$$2A^T Aw - 2A^T y = 0$$

The 2's cancel out.

$$A^T Aw - A^T y = 0$$

$$A^T Aw = A^T y$$

The inverse of the matrix is needed to be taken because matrix division is not defined.

$$w = (A^T A)^{-1} A^T y$$

Now that the “best”  $w$  is calculated, the weight can be placed accordingly to their corresponding inputs and be ready for simple decisions to be made. More specifically, any input ( $x$ ) can be given to perceptron and then the corresponding output is calculated by using the following formula:

$$y = x^T w$$

The calculated  $y$  values according to the above function can result in any real number.

But the output is wanted to either be a 0 or 1. So a piecewise function is needed to convert the raw output to either 0 or 1. The easiest way is placing a threshold between 0 and 1 which is 0.5. Real numbers above 0.5 will be converted to 1 whereas real

numbers under 0.5 will be converted to 0. This piecewise function can be expressed as the following.

$$y = \begin{cases} 0, & \text{if } x^T w < 0.5 \\ 1, & \text{if } x^T w \geq 0.5 \end{cases}$$

## Application of Perceptron with two inputs on a toy:

There are going to be two different classes for this toy example inside a two-dimensional space. The first class will be labeled as “blue”, the second will be “red”. These classes are points that have two coordinates therefore two inputs but  $x_0 = 1$  is an exception that is not going to be considered as an input. The following figure 3 has two variables. The  $x_1$  axis is the horizontal axis, the  $x_2$  variable is the vertical axis. The outputs are going to be the class labels. Knowing that there will be mathematical operations the label of the classes “blue” and “red” aren’t useful. Instead the “blue” is going to be identified as 0, the “red” is going to be 1. Every point in figure 3 is a sample (input) and there are 14 points totally.

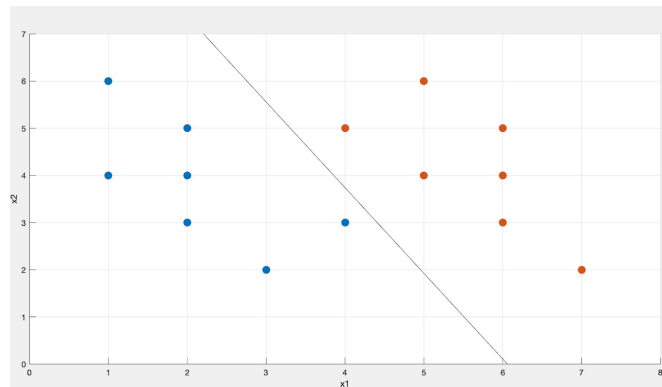


Figure 3: Graph of the toy example in two-dimensional space showing a separating line crossing in between the two different classes.

$$A = \begin{bmatrix} 1 & 1 & 6 \\ 1 & 1 & 4 \\ 1 & 2 & 3 \\ 1 & 2 & 4 \\ 1 & 2 & 5 \\ 1 & 3 & 2 \\ 1 & 4 & 3 \\ 1 & 7 & 2 \\ 1 & 6 & 3 \\ 1 & 6 & 4 \\ 1 & 6 & 5 \\ 1 & 5 & 4 \\ 1 & 5 & 6 \\ 1 & 4 & 5 \end{bmatrix}, w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}, y = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

The outputs are shown in the vector  $y$  as 0's and 1's. The current focus is to show these as  $y = Aw$ . The  $w$  is going to be solved with the method explained in the previous section. The formula  $w = (A^T A)^{-1} A^T y$  is implemented inside MATLAB as shown in the following figure 4.

```
A = [1 1 6; 1 1 4; 1 2 3; 1 2 4; 1 2 5; 1 3 2; 1 4 3; 1 7 2; 1 6 3; 1 6 4; 1 6 5; 1 5 4; 1 5 6; 1 4 5]
y = [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1]'
w = inv(A'*A) * A'*y;
```

Figure 4: This shows how matrix  $A$  and the vector  $y$  are defined. The code simply runs the formula. The highlighted ones are equal signs, the underlined “inv” is taking the inverse of the  $A$  transpose times  $A$  inside of the parenthesis.

With MATLAB,  $w$  is calculated as follows in figure 5.

```
>> w

w =

-0.9744
 0.2434
 0.1339
```

Figure 5: The vector MATLAB has calculated is this.  $w_0$   $w_1$   $w_2$  are these.

As shown in figure 3, there is a best separating line. The following is the equation for the best fit line.

$$x_0w_0 + x_1w_1 + x_2w_2 = x^Tw$$

$x_0$ ,  $x_1$  and  $x_2$  inputs will be given and the final output will be either a zero or one. To further explain a reminder is needed as follows.

$$y = \begin{cases} 0, & \text{if } x^Tw < 0.5 \\ 1, & \text{if } x^Tw \geq 0.5 \end{cases}$$

Accordingly, to the function above, if  $x^Tw$  is smaller than 0.5, sample will be classified as 0, else condition where  $x^Tw$  is bigger or equal to 0.5, the class will be 1. Let's say  $w_0 = 1$ ,  $w_1 = 1$  and  $w_2 = 1$ . If the 1's are plugged inside what will it mean for the result to be over 0.5. It will mean a two-dimensional line. The calculations are shown as follows.

$$1 + x_1 + x_2 > 0.5$$

$$x_1 + x_2 > -0.5$$

In short, the function of the line that separates the classes is as follows.

$$x^Tw = 0.5$$

If the  $w_0$   $w_1$   $w_2$  from MATLAB is substituted to the vector  $w$  in the above and  $x_0$  is set to be 1, the above equation turns into an equation with variables  $x_1$  and  $x_2$  which means a line or  $x_1$  versus  $x_2$  coordinate plane. This line is the line in Figure 3.

If the right-hand side of the  $x^T w = 0.5$  equation is replaced with a more flexible variable  $y$ , the new function will depend on 3 variables:  $x_1$ ,  $x_2$  and  $y$ . This function, when graphed, shows a plane.

The following is the graph of  $x^T w = y$ . The  $x_1$  and  $x_2$  are the axis at the bottom and each point of the plane has coordinates on this vertical  $y$  axis. The outcome is a hyperplane. Hyperplane is a subspace whose dimensions are one less than its ambient space.

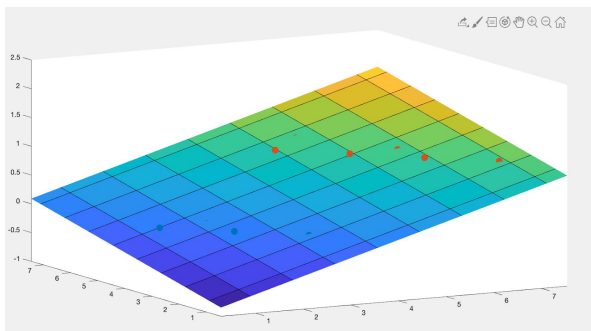


Figure 6: Three-dimensional plane/hyperplane is shown as previously explained in toy example with two inputs.

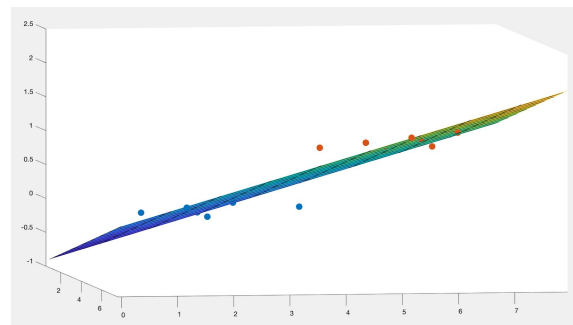


Figure 7: Looking to figure 6 from a different angle. The plane is the best plane of being closest to the labels.

## Application of Perceptron on Audio Recognition:

### Data Preparation:

In order to correctly apply perceptron to my aim, I have firstly made a dataset of my own that consists of 18 sound recordings with their output identified correctly. This dataset is

my perceptron's training set and I will identify the voice records of when I said "A" with 0 and when I said "İ" with 1 which is comparably the most pitched Turkish alphabetical character. A sound is formed of a periodic wave that has sine and cosine functions inside it. There is a detail that is essential to be mentioned. As I am going to record my voice, I will have my inputs through mono-tech channels so the sound waves will be only delivered from one record sensor for simplicity. To make the audio even clearer and precisely processable, the audio recorder will sample 44100 times a second. The inputs which are the coordinates of the points inside the records will be organized in a matrix that will be called (A). The structure of (A) is similar to the previously mentioned (A). The equation will be  $y = Aw$  and it will be solved for w which is a 44101x1 vector. When w is found, it will be placed in the perceptron to test new inputs later on.

### **What is MATLAB:**

MATLAB is a programming platform designed specifically for engineers and scientists. The platform uses MATLAB language which is a matrix-based language allowing a natural expression of computational mathematics. MATLAB can be used to mainly analyze data, develop algorithms or create models and applications<sup>1</sup>. So, the data is collected by using MATLAB and it is shown as follows.

---

<sup>1</sup> "What Is MATLAB?" MATLAB & Simulink, [www.mathworks.com/discovery/what-is-matlab.html](http://www.mathworks.com/discovery/what-is-matlab.html).



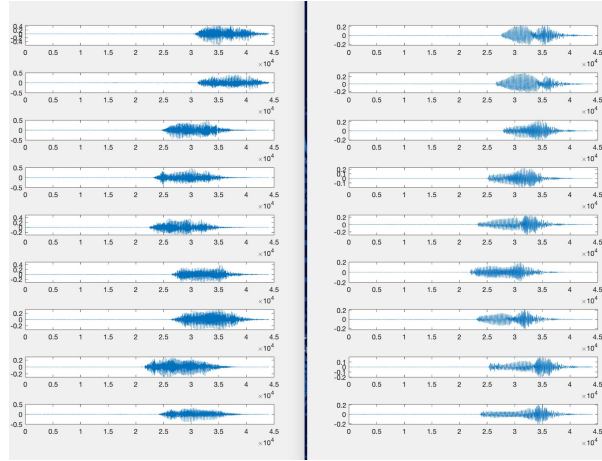


Figure 8: Voice recordings of “A” on the left, Voice recordings of “I” is on the right. This is the training data.

### The Application Process:

Before directly starting to explain how  $y = Aw$  will be solved in an under-determined system where there are more unknowns than there are equations available, it is essential to remember that the  $w$  isn't going to be precise, but it will be the “best approached value”.

The model of the perceptron that will be used is as follows.

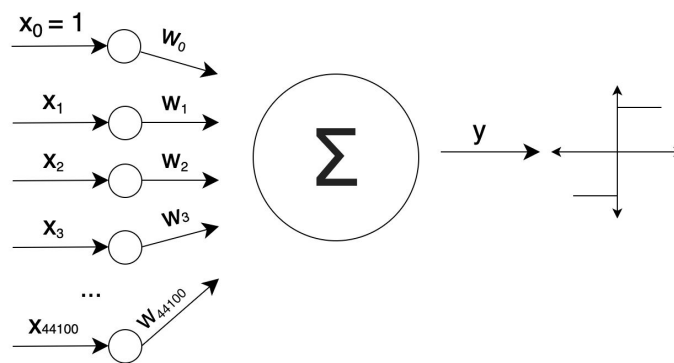


Figure 9: Shows the perceptron with a total of 44101 different weights since the recorder is recording 44100 times per second.

The number of weights is 44101, the number of samples (m) is 18 and the number of outputs is the same. So, the current problem is the number of unknown weights are almost 2500 times higher than the available number of equations to solve for w.

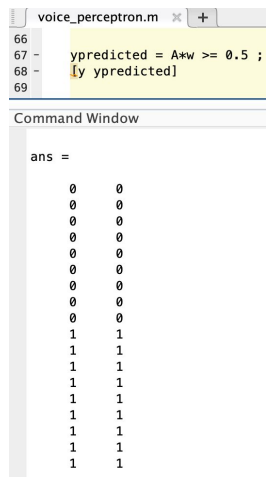
$$\begin{bmatrix} x^{(1)T} \\ x^{(2)T} \\ x^{(3)T} \\ x^{(4)T} \\ x^{(5)T} \\ \dots \\ \dots \\ \dots \\ \dots \\ x^{(18)T} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ \dots \\ \dots \\ \dots \\ \dots \\ w_{44100} \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ y^{(3)} \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ y^{(18)} \end{bmatrix}$$

And again, the equation for finding the weights that has been mentioned several times is essential.

$$w = (A^T A)^{-1} A^T y$$

From this point MATLAB will do the calculations and find the “best” w for the system of equations and answer if the newer inputs which I will make are either an “A” or “I”. As the calculations are true, anywhere above a success rate of 50% is needed to be achieved in order to show the validity of the perceptron.

## Training & Testing Process:

A screenshot of a MATLAB Command Window. The window title is 'voice\_perceptron.m'. The command history shows lines 66 to 69: line 66 is empty, line 67 is 'ypredicted = A\*w >= 0.5 ;', line 68 is '[y ypredicted]', and line 69 is empty. The output shows 'ans =' followed by a 2x15 matrix of 0s and 1s. The first column contains 10 zeros and 5 ones. The second column contains 10 zeros and 5 ones, matching the first column exactly.

```
voice_perceptron.m x +
66
67 - ypredicted = A*w >= 0.5 ;
68 - [y ypredicted]
69

Command Window

ans =

     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     1     1
     1     1
     1     1
     1     1
     1     1
     1     1
     1     1
     1     1
     1     1
```

I have run the formula to find the  $w$ , then used the  $w$  to predict the labels of the recordings that I already have. This is called the training process. The vectors on the left are the true labels of recordings, on the right are the predicted labels of the recording. In summary, 100% training accuracy is achieved.

Figure 10: side by side comparison of the predicted and the true labels of training.

The next thing to do is testing my perceptron. To test it, I recorded 5 audios of me saying “A” and 5 others of me saying “i”. I then compared the outputs that are in a vector at the left side with their true labels of the recordings located at the left side in a vector. On line 81 of my code, I shrunk the number of elements of the test recordings inside of  $A$  to increase my success rate of the test because the part I cut was not providing any information and all of them were almost zero.

```

voice_perceptron.m  createtesting.m  visualizesounds.m  +
78
79 - testA=[1 testx1'; 1 testx2'; 1 testx3'; 1 testx4'; 1 testx5'; 1 testx6';
80 -       1 testx7'; 1 testx8'; 1 testx9'; 1 testx10'];
81 - testA = testA(:, [1 20000:44101]);
82
83 - ytestpredicted = testA*w >= 0.5 ;
84 - ytesttrue=[0 0 0 0 1 1 1 1 1 1]';
85 - [ytesttrue ytestpredicted]
86
87 - ypredicted = A*w >= 0.5 ;
88 - [y ypredicted]

```

Command Window

```

ans =
     0     0
     0     0
     0     1
     0     1
     0     0
     1     0
     1     1
     1     1
     1     1
     1     1

```

Figure 11: Side by side comparison of the true labels and the input recording labels.

The Results showed that only three predictions didn't match out of 10 meaning the perceptron has a 70% success rate.

**Confusion Matrix**

		A	i
Predicted	A	3	1
	i	2	4
		True	

Figure 12: Confusion matrix table shows successes and failures of the perceptron testing process results.

The confusion matrix above is a table that shows what the perceptron couldn't get it right. As shown, 3 of the "A" recordings were predicted correctly out of 5 and 4 of the "i" recordings were predicted correctly out of 5. Confusion matrix is another way of visualizing the success rate of a perceptron. So, it is seen that the perceptron has a 70% success rate and 30% failure rate in general.

**Conclusion:**

In conclusion, how strong mathematics and machine learning are integrated with each other at its simplest form is shown. Linear algebra combined with calculus was used to solve the  $y = Aw$ . Even though it was an under-determined situation. The weights formula was found to be  $w = (A^T A)^{-1} A^T y$ . Throughout the investigation, I used the properties of matrices and integrated with linear algebra. In order to find the minimum error, the square of the norm had to be taken in order to get rid of the square root. Later its derivative was taken and was made equal to zero. The error was found as well as most importantly the formula of weights. It was observed that  $y = Aw$ , whatever the dimension mathematics the operations are taken in, the method of solving for  $w$  is always similar.

**Reflection:**

In overall, I have added and practiced many of the mathematical skills like calculus and linear algebra. These are the mathematical skills that I got in the High-Level Mathematics course. Since I have been very interested in neural networks, machine learning, it motivated me really well throughout the investigation. I was not so afraid when I chose this topic since I was really interested in its entirety. I surely knew there are out of the curriculum mathematical topics but I was still willing to explore.

As I worked on the investigation, I have encountered more new knowledge than I predicted. Out of most of the mathematics used, linear algebra with matrices was the

most interesting and once understood, the favorite. I learned many more than I thought I would initially. I have been seeing the images of neural networks, automation developments, new technologies, before this investigation and haven't been understanding them perfectly in its entirety and maybe I will not be able to still after this. However, I am certain that I have understood its core knowledge well enough to continue on fulfilling my aims. And after the investigation, I realized that I have chosen the best topic for me that I could have ever chosen. It was also interesting that even though for dimensions to work with mathematics, each axis must be perpendicular to each other and it seemed that even though we can't imagine a 44101-dimensional world, the mathematics works out. In addition, when I tested my perceptron after I trained it, giving me a 70% success rate was also interesting to see my perceptron working well. One of the limitations I had was using the useful audio data from raw data which I collected myself. There are smart algorithms to sort out the useful data from the data of when I was silent but it would have been way harder to implement it to my data, so I cut and used the useful data manually. It must be remembered that studying perceptron is very limitative as one of the only possible real-life applications I found is distinguishing the sounds of two alphabetical characters' voices.

## **Bibliography:**

1. "MATLAB." MATLAB Documentation, [www.mathworks.com/help/matlab/](http://www.mathworks.com/help/matlab/).
2. "What Is MATLAB?" MATLAB & Simulink, [www.mathworks.com/discovery/what-is-matlab.html](http://www.mathworks.com/discovery/what-is-matlab.html).
3. Online LaTeX Equation Editor - Create, Integrate and Download, [www.codecogs.com/latex/eqneditor.php](http://www.codecogs.com/latex/eqneditor.php).
4. "Diagrams.net - Free Flowchart Maker and Diagrams Online." Flowchart Maker & Online Diagram Software, [www.draw.io/#G1RxYpXapyk4fxo36PFYFVclWX2I7t8OTn](http://www.draw.io/#G1RxYpXapyk4fxo36PFYFVclWX2I7t8OTn).

