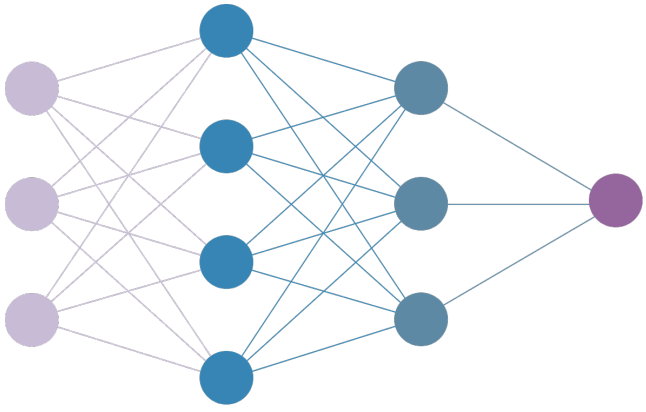
BO - HUB

B-INN-000

A.I. Masterclass

Identifying the Genre of a Song with Neural Networks



Masterclass Explanation:

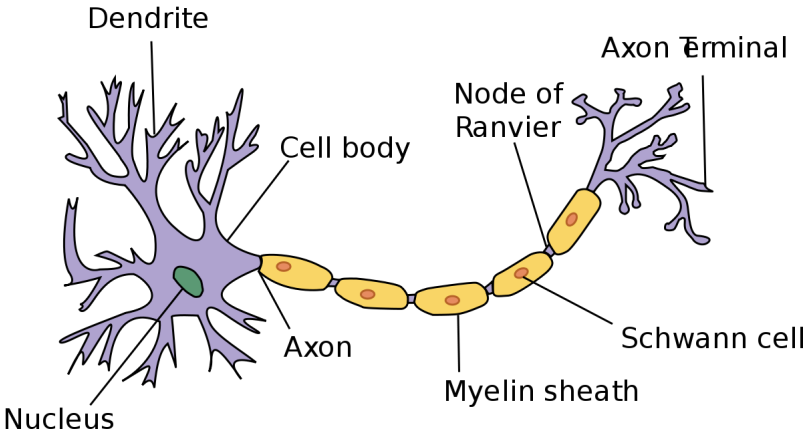
The masterclass is a series of workshop / lessons that will help you to understand how to use neural networks.
The masterclass will last 2 days and will be divided into 3 parts:
- Part 1: Introduction to neural networks and the basics of the machine learning
- Part 2: Developing a neural network
- Part 3: Testing and Upgrading the neural network

You need to be confortable with basic mathematics and have at least Python 3.8 installed on your computer

Part 1: Introduction to N.N.

Part 1.1: Where come from neural networks ?

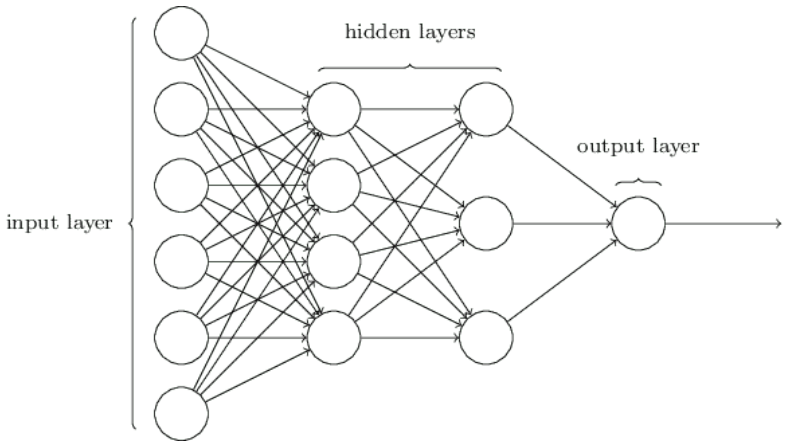
The neural network take the concept of neurone like in the brain.
TODO (Explain the concept of neurone and how the brain works with axios etc)



Part 1.2: Describing the neural network

The neural network is a set of neurons that are connected to each other.
The NN has the particularity that it is a set of layers. 3 types of layers exactly:

- The input layer is the layer that contains the input data.
- The hidden layer is the layer that contains the neurons that process the input data.
- The output layer is the layer that contains the output data.

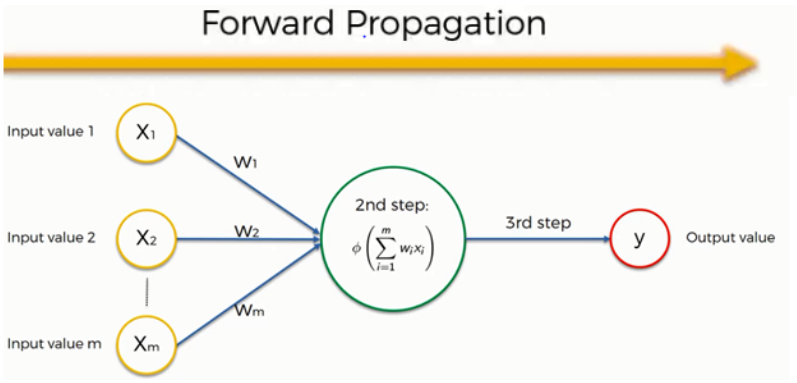


Each layer has his own functionnality and has a number of neurons.
Each neuron has his own weights and biases. We'll see how to use them in the next part.
We'll see that each neuron in the hidden layer has a function called activation function.
The NN can have several units in the output layer in the case of **multiclass classification**.

A **multiclass classification problem** is a problem that has multiple existing solutions.
We can take the example of the object recognition problem. We have a set of images of objects and we want to know which one of them is the object we are looking for. So, the possible solutions are maybe a chair, a table, a sofa, a lamp etc.

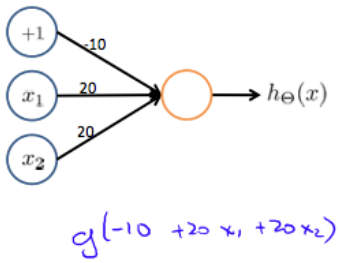
PART 1.3: HOW THE NEURAL NETWORK LEARNS ?

First, the neural network uses the input data and the weights of the hiden layers' units to compute the output data.
This technique is called the **forward propagation**.



To illustrate the forward propagation, we will use the following example:
Let's say we want to compute the OR binary function. We'll use 2 inputs units, 1 hidden layer with 1 unit and 1 output unit. We already have the weights of the hidden layer.

Example: OR function

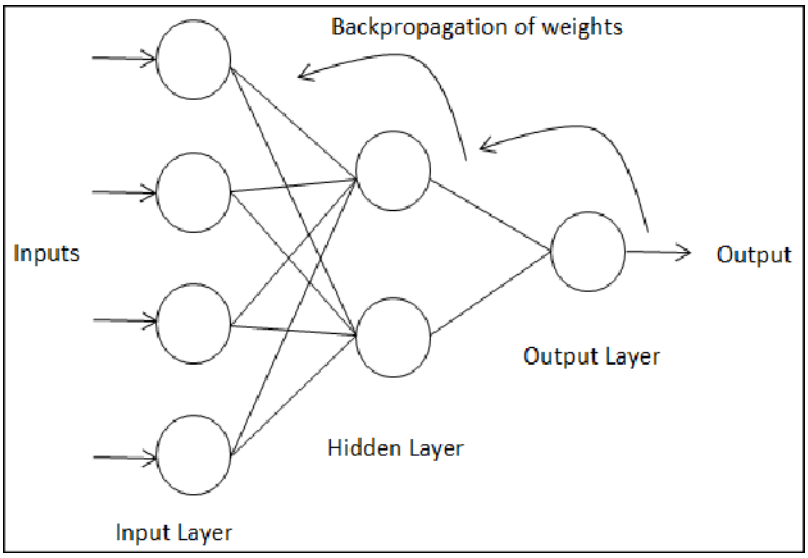


x_1	x_2	$h_\Theta(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	≈ 1
1	1	≈ 1

The activation function **g** will take the following form:
g(x) with g being the sigmoid function and x equal to :
x = (-1 * 10) + 20 * x1 + 20 * x2
We can create the following table with this function:

x1	x2	g(x)
0	0	g(-10) = 0
0	1	g(10) = 1
1	0	g(10) = 1
1	1	g(30) = 1

Then, the neural network tries to improve the output data by changing the weights.
So, the neural network learns the relationship between the input data and the output data.
This technique is called the **back propagation**.



Behind this simple reprensentation, the Neural Network computes the following calculus:

Summary: the equations of backpropagation

$$\delta^L = \nabla_a C \odot \sigma'(z^L) \tag{BP1}$$

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l) \tag{BP2}$$

$$\frac{\partial C}{\partial b^l_j} = \delta^l_j \tag{BP3}$$

$$\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j \tag{BP4}$$

Part 2: Developing a neural network

```
Terminal
~/B-INN-000> mkdir DiscordBotWorkshop
cd DiscordBotWorkshop
npm init
npm install discord.js
npm install axios
```

Ici nous avons crée un projet DiscordBotWorkshop, Ou nous avons installé discord.js et axios.
axios est une lib de node qui permet de faire des requetes a une api.
Par exemple nous pouvons faire ça:

```
const axios = require('axios');

axios.get('http://54.36.183.102:2900/anime/832').then(function (response) {
    console.log(response.data);
}).catch(function (error) {
    console.log(error);
});
```

Ici nous faisons une request a une api ou l'on lui demande d'avoir l'id 832 d'un anime
Maintenant nous allons crée le bots.

Configuration

Tout d'abord nous allons allez sur le Discord Portal: https://discordapp.com/developers/applications/
Puis vous allez crée une application
Maintenant va dans **Bot** and *Add a Bot*.
Une fois le bot crée il faut l'invité sur le serveur.
Allez dans OAuth2 et cliquez sur bot et Admin
Copier le link générer, ouvrer le puis ajouter le bot au serveur

Ensuite nous allons allez sur le site my.epitech.eu
Une fois sur le site l'objectif est de récupérer le token d'authentification avec la console du navigateur (F12)

Création du bot

Maintenant nous allons pouvoir commencer a coder
Tout d'abord crée un fichier json avec le nom config.json:

```
{
    "tokenDiscord": "LE TOKEN",
    "tokenEpitech": "LE TOKEN",
    "prefix": "!"
}
```

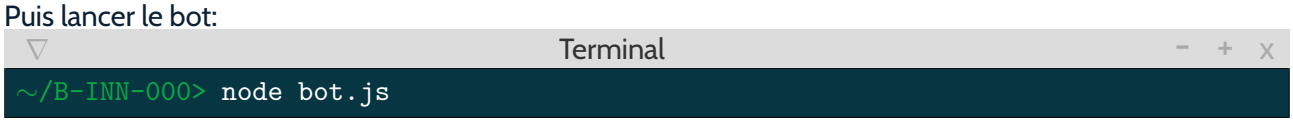
Puis il vous faudra crée un fichier bot.js ou nous allons pouvoir commencer a coder.

```
const { Client, Intents } = require('discord.js');
const config = require('./config.json');
const bot = new Client({ intents: [Intents.FLAGS.GUILDS, Intents.FLAGS.GUILD_MESSAGES
    ]});

bot.on('ready', () => {
    console.log('Ready!');
    bot.user.setActivity('Bot Launch')
});

bot.login(config.tokenDiscord);
```

Puis lancer le bot:



Ce bout de code nous permet de lancer le bot et de faire qu'au moment du lancement nous auront le message "Logged in as {Nom du bot}" et de changer l'activité du bot par "First bot launch".

Exercice time

Pour commencer nous allons devoir faire en sorte que le bot répond au message qu'on lui envoie.
Pour cela nous allons utlisé:

```
bot.on('messageCreate', async message => {
    const prefix = config.prefix;
    if (!message.content.startsWith(prefix) || message.author.bot) return;
    const args = message.content.slice(prefix.length).split(/ +/);
    const command = args.shift().toLowerCase();
    console.log(`${message.author.tag} do the command ${command}`);
});
```

Ici nous disont que quand le bot recevra un message si le message ne contiens pas le prefix defini dans le Json ou bien alors que l'author de ce messages sois un bot il ne devra pas réagir
mais si ce n'est pas le cas il devra parse les arguments et mettre la commande principale dans la valeur command

Ping

Maintenant c'est a vous de jouer faite en sorte que quand vous faite la commande "!ping" le bot vous réponde pong

Embed Message

Maintenant que vous savez répondre autant mettre de la forme faite un message embed de votre commande !ping
Qui contienne :

- En titre le Nom de celui qui envoie le messages
- En description le messages de votre choix
- En Footer le nom de votre bot

Je vous invite a regarde new Discord.MessageEmbed()

Avoir toute les notes de my.epitech.eu

Maintenant que nous savons faire de joli message et nous allons enfin utilisé axios pour faire une commande qui nous sortira toute les dernieres moulinettes.
Pour cela nous allons utilisé l'api du site my.epitech.eu
crée La commande !Notes

```
const response = await axios.get('https://api.epitest.eu/me/${year}', {headers: {
    Authorization: config.tokenEpitech }
});
console.log(response.data);
```

Ici grace au console.log vous allez recevoir toute la data de vos notes a vous de l'utilisé pour en faire un joli message Embed contenant le titre de votre projet ainsi que le nombre de test que vous avez passée.

Avoir une notes spécific d'un projet

Le but pour vous est de faire en sorte que parmis tout les projet disponibles seulement celui que vous avez demande avec la commande !note ${Projet} vous sois sortie avec tout les détails

Avoir une messages réactif

Le but pour vous est de faire en sorte qu'avec des émotes vous puissiez modifié un message et ainsi change le projet que vous voyez sur le moment
par exemple vous etes sur la 1er moulinettes du projet Bistromatique grace au emote vous pouvez allez sur la seconde ou voir meme changer de projet.