

MINI PROJET : LE PERCEPTRON

Description :

Ce projet peut être traité seul ou par un groupe de deux personnes.

Il sera rendu sur MBN.

Le nom du fichier sera du type "Nom1-Nom2.py" et les deux noms des élèves devront figurer au début du fichier.

Ce projet porte sur l'**Intelligence Artificielle** et plus précisément sur le **perceptron**, ancêtre du **neurone**, constituant de base des **Réseau de neruones** (**Deep Learning**).

- On fournit à la machine des **données**, on choisit un **modèle**, on utilise un **algorithme d'optimisation** pour ajuster le **modèle** et minimiser les erreurs.
- Un **neurone** apprend tout seul, mais le **Data Scientist** lui fournit des **hyperparamètres** qui permettent de régler l'apprentissage :
 - Le **pas d'apprentissage** α .
 - Le **biais** θ .
- Le **but du projet** est de créer un **perceptron** qui nous permettra de répondre à différentes questions :
 - Séparer linéairement **deux classes**.
 - Estimer si une couleur est **rouge**.
 - Savoir si une **personne survit au naufrage du Titanic**.

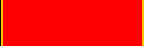



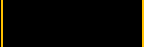




Production : Détection de la couleur rouge

Présentation

Le but est de, toujours modestement, apprendre à notre **perceptron** à reconnaître la couleur **Rouge**.

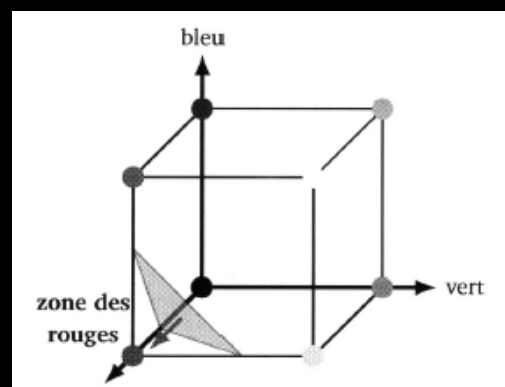
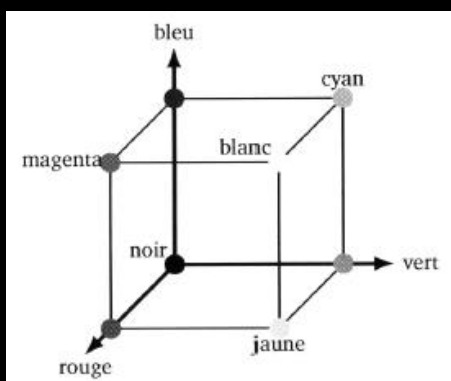
- Pour cela, nous allons utiliser le **système RGB** pour coder une couleur.
- Le principe consiste à coder une couleur selon ses niveaux de **Rouge** – **Vert** – **Bleu** (synthèse additive) : chaque niveau est codé sur un **octet** (de 0 à 255).
- Ce [site](#) permet de voir l'impact des niveaux sur la couleur finale.

Voici quelques exemples :

Couleur	Nom	Niveau Rouge	Niveau Vert	Niveau Bleu	Hexadécimal
	Rouge	255	0	0	#FF0000
	Vert	0	255	0	#00FF00
	Bleu	0	0	255	#0000FF
	Blanc	255	255	255	#FFFFFF
	Noir	0	0	0	#000000
	Orange	255	128	0	#FF8000
	Gris	128	128	128	#808080
	Jaune	255	255	0	#FFFF00
	Nuance Rouge	204	50	25	#CC3219
















On peut considérer l'espace de toutes les couleurs comme un cube : axe des abscisses pour le **rouge** variant de 0 à 255, axe des ordonnées pour le **vert** et axe des cotes pour le **bleu**.

Le **rouge** et ses nuances correspondent à une zone au voisinage du point **(255, 0, 0)**.



Mise en œuvre Python du perceptron

On dispose du fichier *detecterouge_train.csv* correspondant à une partie du jeu d'entraînement ci-dessous.






Test	Couleur	Niveau Rouge	Niveau Vert	Niveau Bleu	Hexadécimal	Objectif
1		255	0	0	#FF0000	1
2		255	255	0	#FFFF00	0
3		0	255	255	#00FFFF	0
4		255	0	50	#FF0032	1
5		255	0	255	#FF00FF	0
6		0	0	0	#000000	0
7		180	0	0	#B40000	1
8		128	128	128	#808080	0
9		230	50	0	#E63200	1
10		255	255	255	#FFFFFF	0
11		230	0	0	#E60000	1
12		50	255	0	#32FF00	0
13		204	0	25	#CC0019	1
14		204	50	0	#CC3200	1
15		204	25	25	#CC1919	1
...						

- Créer une instance `perceptron1` de la classe `Perceptron` avec un `biais` de 1 et un `alpha` de 0.02 et 3 entrées.
- Créer une méthode `chargement_rouge(nom_fichier)` de la classe `Perceptron` pour créer une variable `data_set` de type `dictionnaire` correspond aux entrées et objectifs `y` pour l'entraînement du neurone :

```
data_set = {id_data1: [(rouge, vert, bleu), y],
            id_data2: [(rouge, vert, bleu), y],
            ...}
```

- Entraîner le `perceptron1` avec le `train set` et tester la performance du modèle avec le `test set`.

- Interrogation du **perceptron1** :
 - On peut maintenant vérifier que notre **perceptron1** dans son état final sait détecter correctement le rouge : la fonction **reponse()** renvoie 1 uniquement pour une couleur **Rouge** proposée en entrée.
 - Voici quelques tests :

Couleur	Code RGB	Hexadécimal	Sortie S	Rouge
	(240, 30, 0)	#F01E00		
	(255, 50, 50)	#FF3232		
	(0, 0, 255)	#0000FF		
	(255, 128, 0)	#FF8000		
	(180, 128, 100)	#B48064		

- Essayer en recommençant avec différentes valeurs de **pas d'apprentissage** α et de biais θ en regardant les résultats de la somme pondérée $e_1 \cdot w_1 + e_2 \cdot w_2 + \dots + e_n \cdot w_n$.

Production : Naufrage du Titanic

Présentation

Le but de cette partie est d'utiliser le **perceptron** dans le but de savoir si une personne aurait survécu au **naufrage du Titanic**.



On utilisera une **BDD** donnant les caractéristiques des **passagers du Titanic**.

On fournit en ressources le **fichier *titanic.csv*** dont voici les informations concernant les 10 premiers passagers :

IdPassager	Survivant	Classe	Nom	Sexe	Age	NbFreres_Soeurs/Epoux	NbParents/NbEnfants	Ticket	Tarif	Cabine	Embarquement
1	0	3	Braund Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cummings Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikinen Miss. Laina	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	Futrelle Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen Mr. William Henry	male	35	0	0	373450	8.05		S
6	0	3	Moran Mr. James	male		0	0	330877	8.4583		Q
7	0	1	McCarthy Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
8	0	3	Paisson Master. Gosta Leonard	male	2	3	1	349909	21.075		S
9	1	3	Johnson Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2	347742	11.1333		S
10	1	2	Nasser Mrs. Nicholas (Adele Achem)	female	14	1	0	237736	30.0708		C
11	1	3	Sandstrom Miss. Marguerite Rut	female	4	1	1	PP 9549	16.7	G6	S

- On dispose de **12 informations** sur chaque passager (si tous les champs sont remplis), qui doivent être **numériques** être comprises par notre modèle de **perceptron**.
- On fait de choix de ne retenir que les en-têtes **Classe**, **Sexe** et **Age** en entrées du **perceptron**. Les **objectifs** seront donnés par l'en-tête **Survivant** (0 signifie que le passager est **mort**, 1 que le passager a **survécu**).
- Comme on peut le voir sur la capture d'écran, l'âge de certains passagers n'a pas été renseigné. Il ne faudra donc pas le charger dans la base d'entraînement.

Plus généralement, si un champ est absent, il faudra ignorer cette entrée.

Mise en œuvre Python du perceptron

- Créer une méthode `chargement_titanic(nom_fichier)` qui prend en paramètre un nom de fichier `nom_fichier` et retourne un dictionnaire :

```
data_set = {id_data1: [(classe, sexe, age), survivant],  
            id_data2: [(classe, sexe, age), survivant],  
            ...}
```

Attention : Le perceptron ne sachant traiter que des entrées numériques : lorsque le sexe du passager est (female/male), il sera remplacé par (2/0).

- Fabriquer le train set et le test set à l'aide des fichiers `titanic_train.csv` et `titanic_test.csv`.
- Créer une instance `perceptron2` de la classe `Perceptron` et l'entraîner avec le train set et tester la performance du modèle avec le test set.
- Demander alors au `perceptron2` en lui donnant votre âge, votre sexe et la classe dans laquelle vous imagineriez voyager pour savoir si vous auriez survécu au voyage du Titanic.
- Essayer en recommençant avec différentes valeurs de pas d'apprentissage α et de biais θ en regardant les résultats de la somme pondérée $e_1 \cdot w_1 + e_2 \cdot w_2 + \dots + e_n \cdot w_n$.
- Rajouter les en-têtes `tarif` et le `port d'embarquement` comme entrées du `perceptron2`.

Améliorations possibles :

- ...