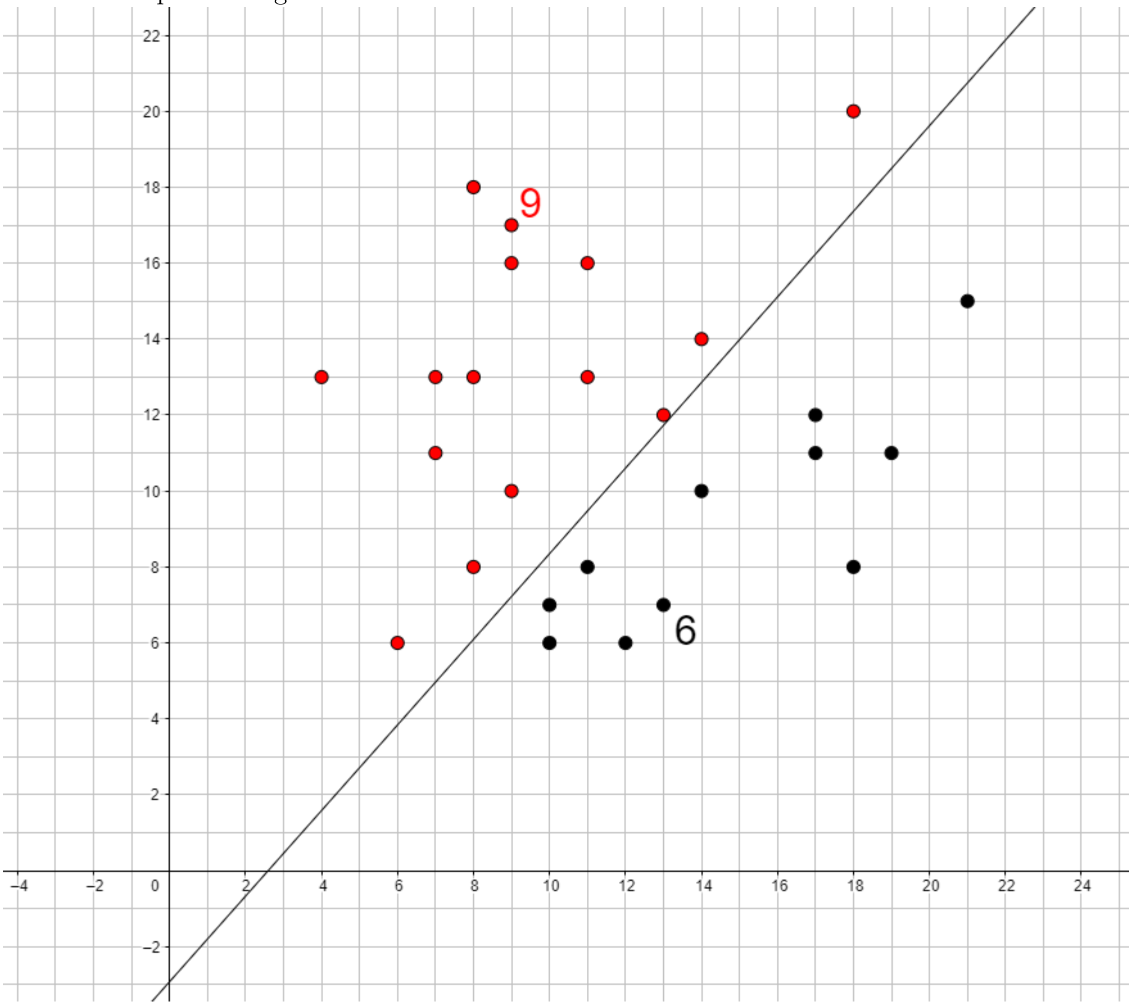
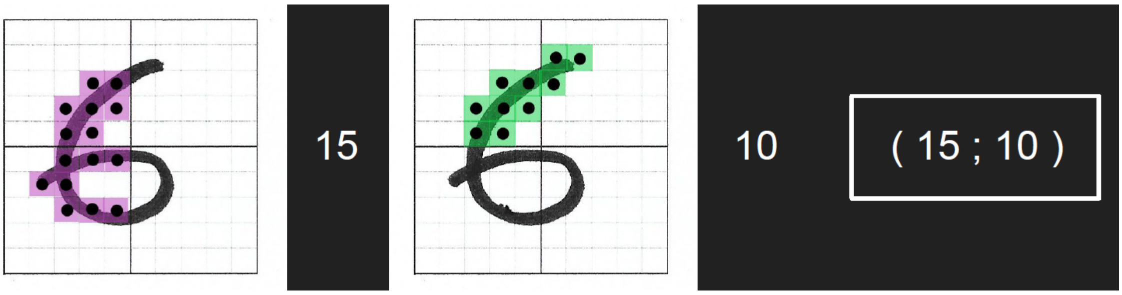
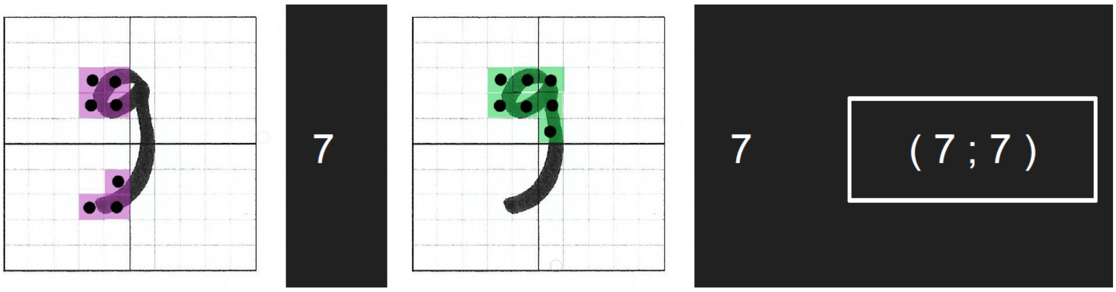
Précédemment, on a vu qu'une IA très simple constituée d'un seul neurone permet de différencier un 6 d'un 9.

Si un neurone a pu apprendre grâce au jeu de vingt-six données ci-dessous, constitué de quinze chiffres 9 et de onze chiffres 6, alors on s'attend à ce qu'il distingue un 6 d'un 9.





Ainsi, le nombre 6 écrit ci-dessus correspond à l'abscisse $x = 15$ (le nombre de points dans la partie gauche de la figure) et l'ordonnée $y = 10$ (le nombre de points dans la partie supérieure de la figure). Le neurone identifie un 6.



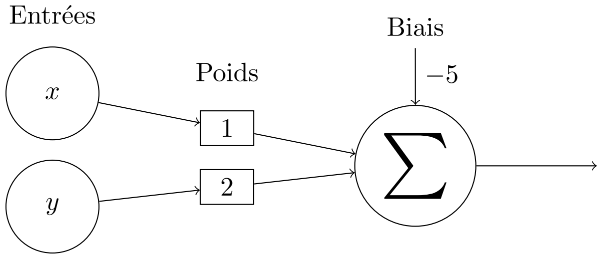
De même, le nombre 9 écrit ci-dessus correspond à l'abscisse $x = 7$ et l'ordonnée $y = 7$. Le neurone identifie un 9.

- Qu'est ce qu'un neurone informatique ?
- Comment fonctionne-t-il ?
- Comment peut-il apprendre à partir d'un jeu de données ?

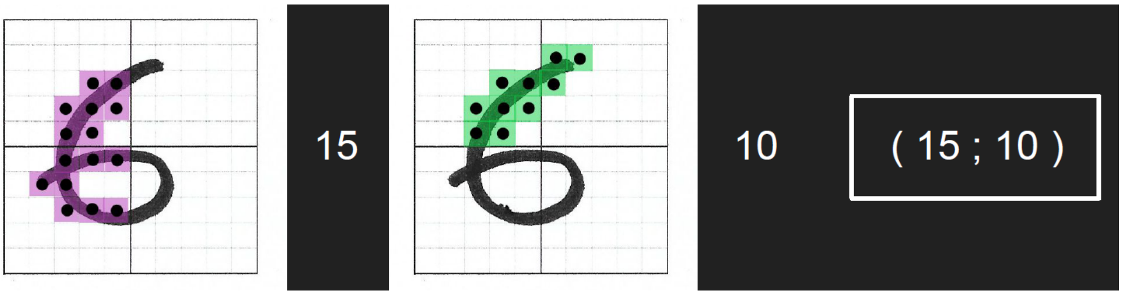
1 Qu'est ce qu'un neurone informatique ? Comment fonctionne-t-il ?

Un neurone :

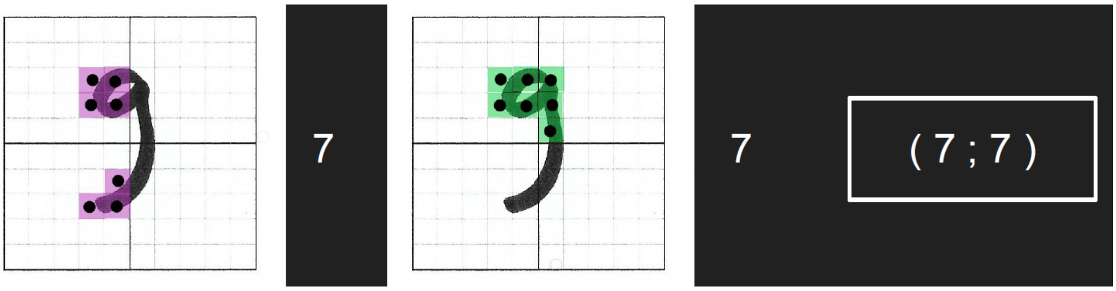
- reçoit des entrées. Dans cet exemple on a deux entrées : x (le nombre de points dans la partie gauche de la figure) et y (le nombre de points dans la partie supérieure de la figure).
- applique des poids synaptiques et ajoute un biais. On prendra, par exemple, les poids respectifs 1 et 2, et le biais -5.
- effectue le calcul : $1 \times x + 2 \times y + (-5)$.



Ainsi, avec le nombre 6 écrit ci-dessous correspondant à $x = 15$ et $y = 10$, le neurone renvoie $1 \times 15 + 2 \times 10 - 5 = 30$.



De même, le nombre 9 écrit ci-dessous correspond à $x = 7$ et $y = 7$: le neurone renvoie alors $1 \times 7 + 2 \times 7 - 5 = 16$.

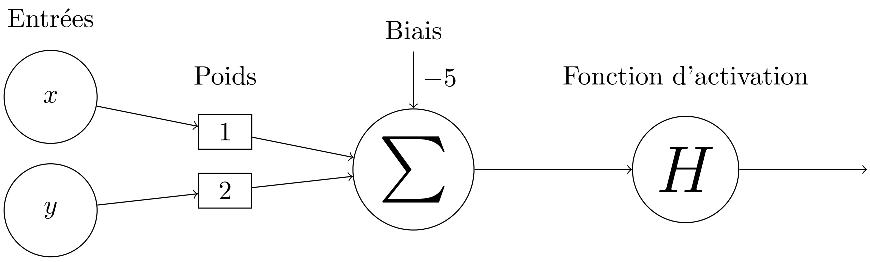


Difficile pour l'instant de voir pourquoi 30 permettrait d'identifier le chiffre 6, ou pourquoi 16 permettrait d'identifier le chiffre 9...

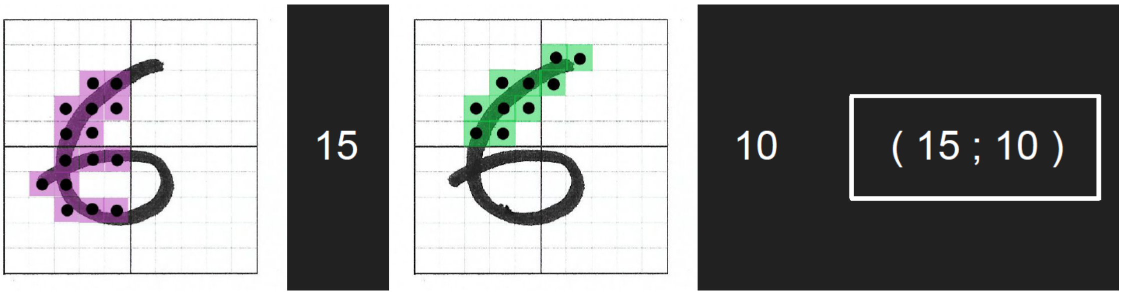
On a besoin d'une fonction d'activation du neurone. On choisit la fonction suivante, notée H :

$$H(x) = \begin{cases} 1 \text{ si } x > 0, \\ 0 \text{ si } x \leq 0. \end{cases}$$

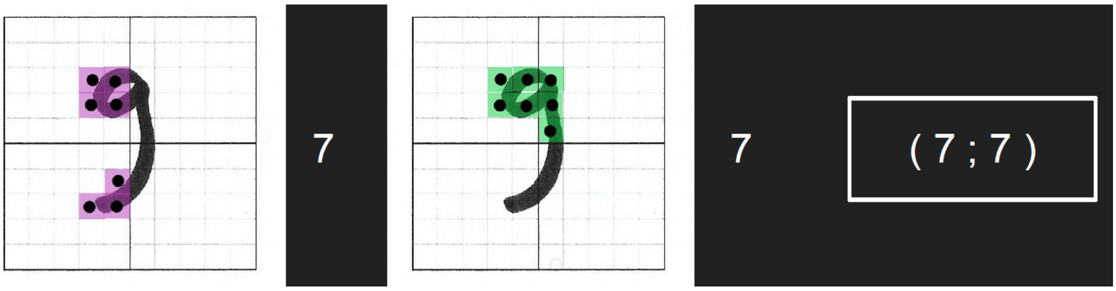
On applique cette fonction après le calcul vu précédemment. L'intérêt est que le neurone ne renvoie plus que deux valeurs : 0 et 1. On peut convenir que le 0 correspond au chiffre 6 et que le 1 correspond au chiffre 9.



Désormais, avec le nombre 6 écrit ci-dessous correspondant à $x = 15$ et $y = 10$, le neurone calcule $1 \times 15 + 2 \times 10 - 5 = 30$ et renvoie 1 car $30 > 0$. Le neurone identifie un 9...

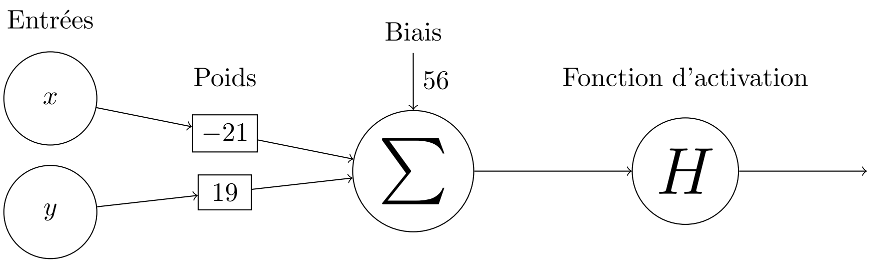


De même, le nombre 9 écrit ci-dessous correspond à $x = 7$ et $y = 7$, le neurone calcule alors $1 \times 7 + 2 \times 7 - 5 = 16$ et renvoie 1 car $16 > 0$. Le neurone identifie un 9 !



Cela ne fonctionne pas très bien. Les poids synaptiques 1 et 2, et le biais -5, ont été choisis de manière arbitraire et ne permettent pas d'obtenir les résultats espérés.

On choisit désormais les poids synaptiques -21 et 19, et le biais 56.

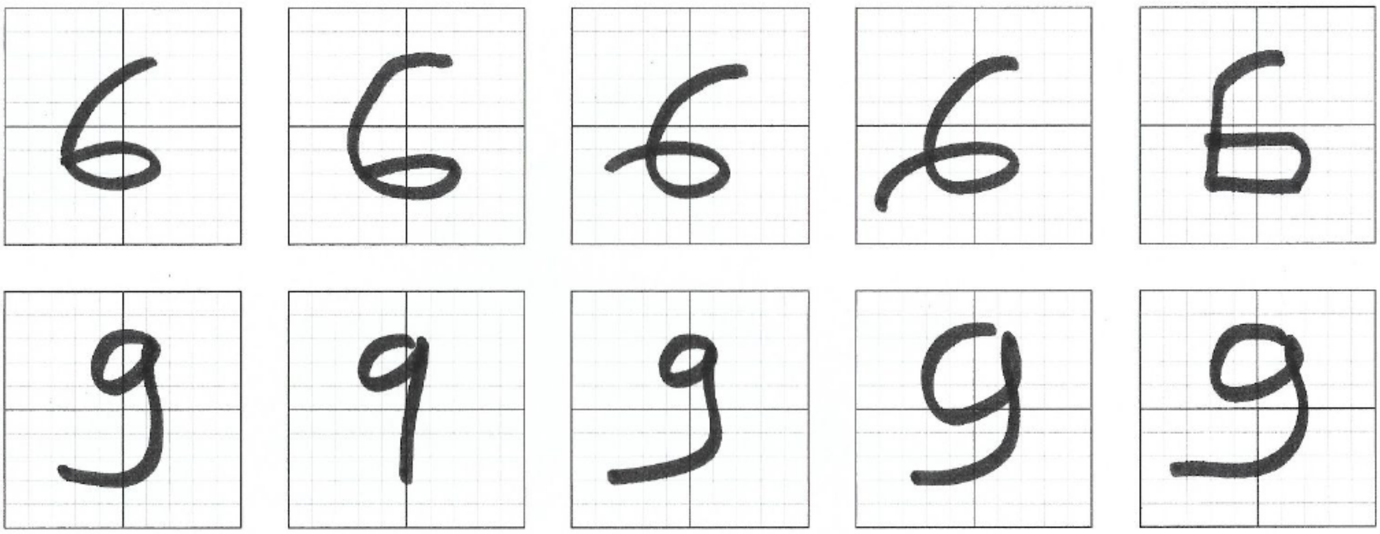


Exercice 1 :

Reprendre les calculs précédents et vérifier si le neurone permet maintenant d'identifier le bon chiffre dans chacun des deux exemples étudiés.

Exercice 2 :

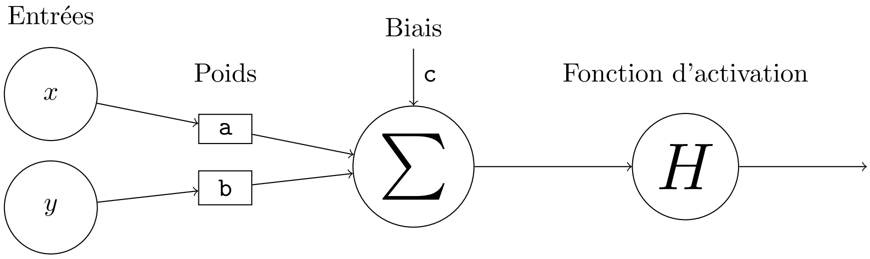
Pour chacun des chiffres écrits à la main ci-dessous, vérifier si le neurone permet d'identifier le bon chiffre.



Il semble que le neurone, avec les poids synaptiques -21 et 19, et le biais 56, permet plutôt bien de distinguer un 6 d'un 9. Comment ces poids synaptiques et ce biais ont-il été déterminés ?

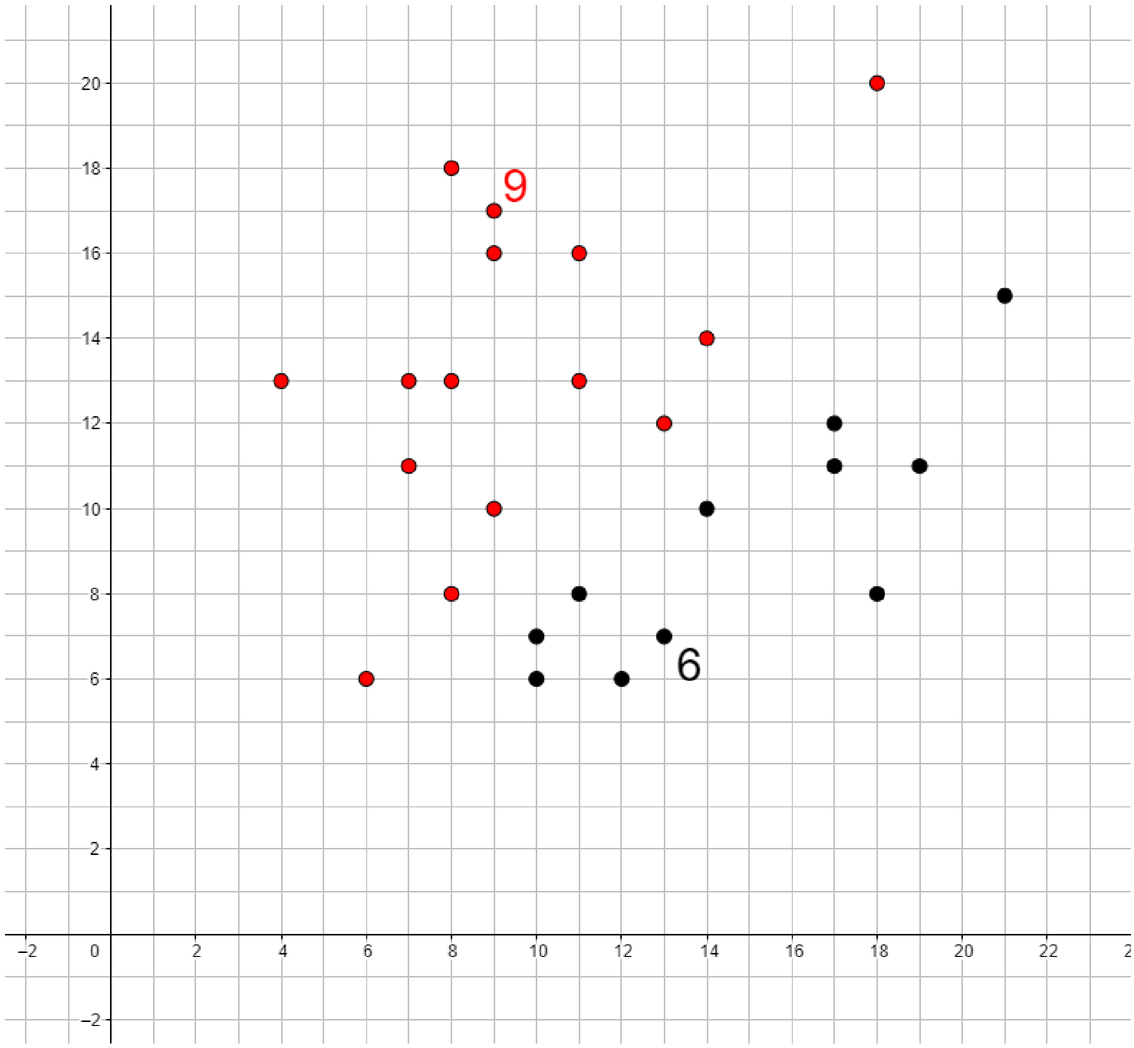
2 Comment un neurone peut-il apprendre à partir d'un jeu de données ?

On considère un neurone dont les poids synaptiques, notés `a` et `b`, et le biais, noté `c`, sont inconnus.



On appelle $\texttt{neurone}(x, y, \texttt{a}, \texttt{b}, \texttt{c})$ le résultat renvoyé par le neurone quand on lui fournit les entrées x et y en appliquant les poids synaptiques `a` et `b` et en ajoutant le biais `c`.

On reprend le jeu de données suivant :



Pour chacun des vingt-six points, on connait son abscisse x, son ordonnée y et le résultat attendu : 1 pour un chiffre 9 et 0 pour un chiffre 6. On va donc pouvoir utiliser ces vingt-six points pour entraîner le neurone.

Par exemple, on sait que $\texttt{neurone}(6, 6, \texttt{a}, \texttt{b}, \texttt{c})$ doit renvoyer 1 : si ce n'est pas le cas, les valeurs de `a`, `b` et `c` ne conviennent pas et il faut les modifier. De la même manière, on sait que $\texttt{neurone}(10, 7, \texttt{a}, \texttt{b}, \texttt{c})$ doit renvoyer 0.

On note (E) l'ensemble des vingt-six points de coordonnées $(x; y)$ dont chacun a un résultat attendu : 1 pour un chiffre 9 et 0 pour un chiffre 6.

L'ensemble (E) contient donc quinze points dont le résultat attendu est 1, ils ont pour coordonnées :
(4;13) ; (6;6) ; (7;11) ; (7;13) ; (8;8) ; (8;13) ; (8;18) ; (9;10) ; (9;16) ; (9;17) ; (11;13) ; (11;16) ; (13;12) ; (14;14) ; (18;20)

et onze points dont le résultat attendu est 0, qui ont pour coordonnées :
(10;6) ; (10;7) ; (12;6) ; (11;8) ; (13,7) ; (14;10) ; (17;11) ; (17;12) ; (18;8) ; (19;11) ; (21;15)

On initialise les poids synaptiques `a` et `b` et le biais `c` du neurone à 0 et on donne l'algorithme suivant, appelé algorithme d'apprentissage par correction d'erreur :

Tant que le neurone ne renvoie pas la bonne valeur de chaque point de (E)
Choisir un point de (E) de coordoonées (`x`, `y`) et dont le résultat attendu est noté `attendu`.
`p` $\longleftarrow$ `neurone(x, y, a, b, c)`
`a` $\longleftarrow$ `a + (attendu - p)` $\times$ `x`
`b` $\longleftarrow$ `b + (attendu - p)` $\times$ `y`
`c` $\longleftarrow$ `c + (attendu - p)`
Fin Tant que

On choisit, par exemple, le point de coordoonnées (10;6) : `attendu` vaut donc 0, ce qui correspond à un 6.
D'autre part `neurone(10, 6, 0, 0, 0)` vaut 0 : en effet $0 \times 10 + 0 \times 6 + 0 = 0$ et $0 \leq 0$.
`attendu - p` vaut donc 0 et les valeurs de `a`, `b` et `c` ne sont pas modifiées.

On prend ensuite le point de coordoonnées (6;6) : `attendu` vaut donc 1.
D'autre part `neurone(10, 6, 0, 0, 0)` vaut 0 : en effet $0 \times 10 + 0 \times 6 + 0 = 0$ et $0 \leq 0$.
`attendu - p` vaut donc 1 et les valeurs de `a`, `b` et `c` sont modifiées :
`a` $= 0 + 1 \times 10 = 10$
`b` $= 0 + 1 \times 6 = 6$
`c` $= 0 + 1 = 1$

Il reste à choisir successivement les 24 points restants.

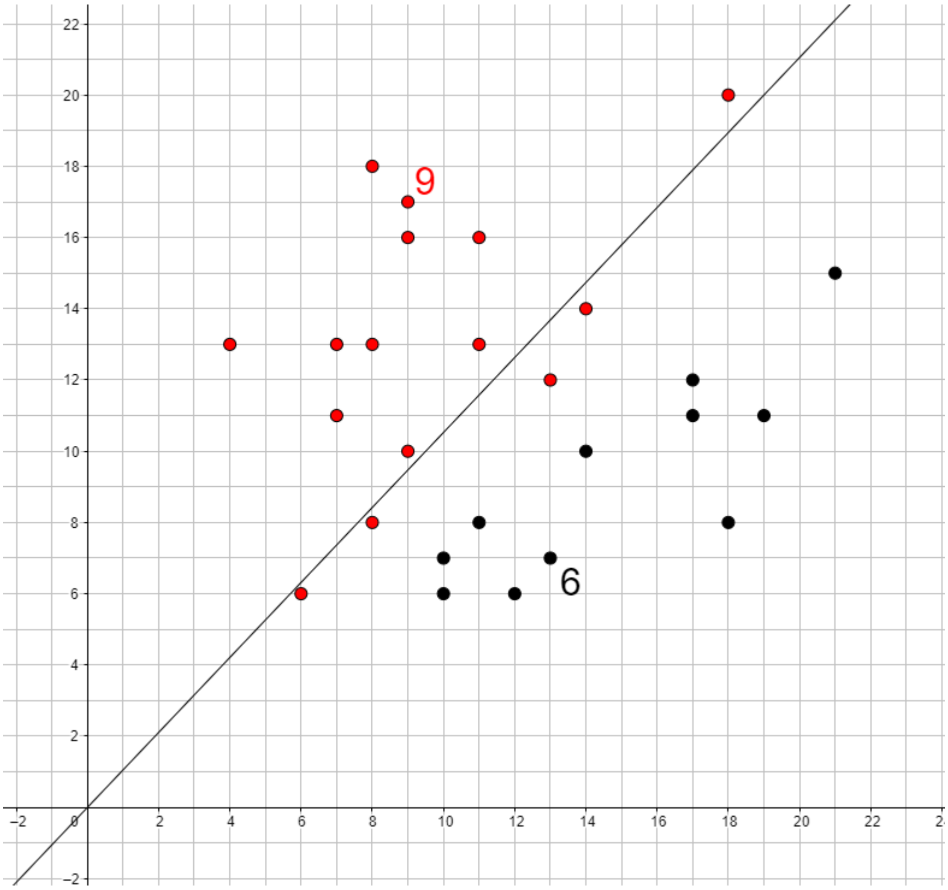
Exercice 3 :

Compléter les étapes 6, 7 et 8 du tableau suivant.

Etape	a	b	c	x	y	attendu	p	attendu = p ?	a	b	c
0									0	0	0
1	0	0	0	10	6	0	0	OUI	-	-	-
2	0	0	0	6	6	1	0	NON	+6	+6	+1
3	6	6	1	8	8	1	1	OUI	-	-	-
4	6	6	1	9	10	1	1	OUI	-	-	-
5	6	6	1	10	7	0	1	NON	-10	-7	-1
6	-4	-1	0	7	11	1					
7				4	13	1					
8				12	6	0					
9	-9	4	0	11	8	0	0	OUI	-	-	-
10	-9	4	0	13	7	0	0	OUI	-	-	-
11	-9	4	0	7	13	1	0	NON	+7	+13	+1
12	-2	17	1	8	13	1	1	OUI	-	-	-
13	-2	17	1	8	18	1	1	OUI	-	-	-
14	-2	17	1	9	16	1	1	OUI	-	-	-
15	-2	17	1	9	17	1	1	OUI	-	-	-
16	-2	17	1	11	13	1	1	OUI	-	-	-
17	-2	17	1	11	16	1	1	OUI	-	-	-
18	-2	17	1	13	12	1	1	OUI	-	-	-
19	-2	17	1	14	10	0	1	NON	-14	-10	-1
20	-16	7	0	14	14	1	0	NON	+14	+14	+1
21	-2	21	1	17	11	0	1	NON	-17	-11	-1
22	-19	10	0	17	12	0	0	OUI	-	-	-
23	-19	10	0	18	8	0	0	OUI	-	-	-
24	-19	10	0	18	20	1	0	NON	+18	+20	+1
25	-1	30	1	19	11	0	1	NON	-19	-11	-1
26	-20	19	0	21	17	0	0	OUI	-	-	-
	Apprentissage non terminé !										

L'apprentissage n'est pas terminé : le résultat n'est pas celui attendu pour chacun des vingt-six points. Il faut continuer et tester encore au moins une fois l'ensemble des vingt-six points en ajustant les poids synaptiques.

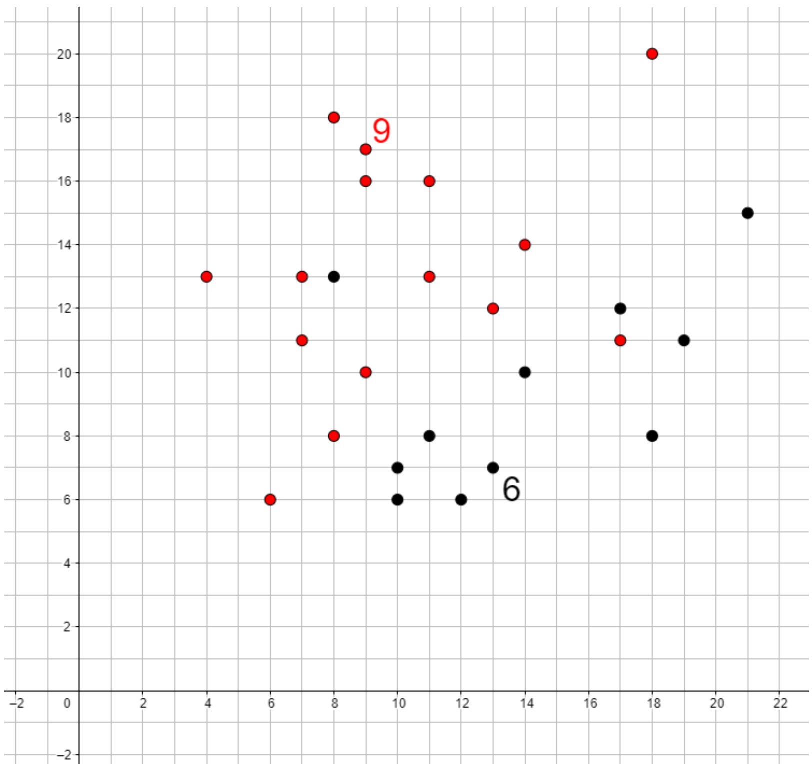
On obtient toutefois un résultat qui est déjà plutôt satisfaisant :



Une équation cartésienne de la droite tracée est $-20x + 19y = 0$, de la forme $ax + by + c = 0$ avec $a = -20$, $b = 19$ et $c = 0$.

♪ Remarque(s) :

- L'algorithme ne s'arrête que si le jeu de données constitue un ensemble linéairement séparable.



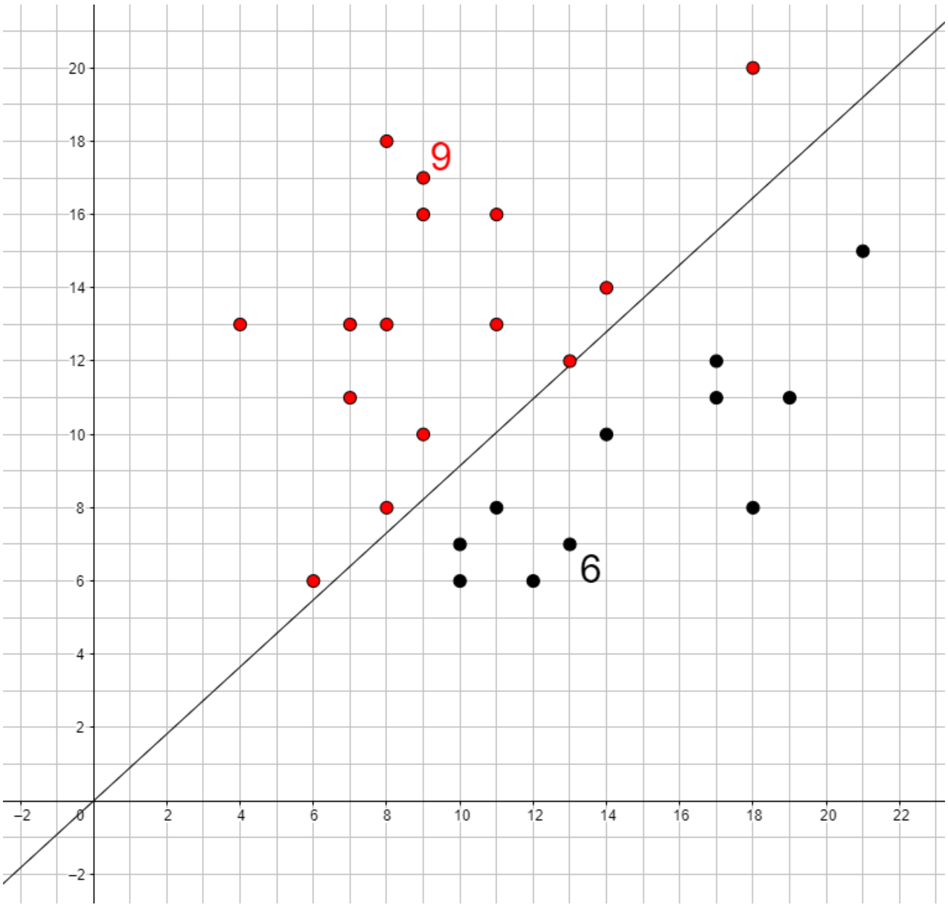
- Dans le cas contraire, il existe d'autres algorithmes pour déterminer les poids synaptiques.

Exercice 4 :

Compléter le tableau suivant. L'apprentissage est-il terminé ?

Etape	a	b	c	x	y	attendu	p	attendu = p ?	a	b	c
27	-20	19	0	10	6	0					
28				6	6	1					
29				8	8	1					
30				9	10	1					
31				10	7	0					
32				7	11	1					
33				4	13	1					
34				12	6	0					
35				11	8	0					
36				13	7	0					
37				7	13	1					
38				8	13	1					
39				8	18	1					
40				9	16	1					
41				9	17	1					
42				11	13	1					
43				11	16	1					
44				13	12	1					
45				14	10	0					
46				14	14	1					
47				17	11	0					
48				17	12	0					
49				18	8	0					
50				18	20	1					
51				19	11	0					
52				21	17	0					
	Apprentissage terminé ?										

Lorsque l'apprentissage est terminé on obtient :



Une équation cartésienne de la droite tracée est $-32x+35y=0$, de la forme $ax+by+c=0$ avec $a=-32$, $b=35$ et $c=0$.

Pour visualiser l'apprentissage complet : `http://ia.dellasantina.corsica/apprentissage-perceptron`