	<pre>import numpy as np import matplotlib.pyplot as plt Excercice 1</pre>
	1- Lecture des données
In [2]:	<pre>from sklearn import datasets iris = datasets.load_iris()</pre>
In [3]:	<pre># Ce n'est pas un DataFrame, mais une sorte de dictionnaire print(type(iris)) <class 'sklearn.utils.bunch'=""></class></pre>
In [4]:	<pre>print(dir(iris)) ['DESCR', 'data', 'data_module', 'feature_names', 'filename', 'frame', 'target', 'target_names']</pre>
In [5]:	# Les noms des varaibles descriptives des différentes observations iris feature_names
Out[5]:	<pre>['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']</pre>
In [6]:	<pre>iris.target_names array(['setosa', 'versicolor', 'virginica'], dtype='<u10')< pre=""></u10')<></pre>
Out[6]: In [7]:	# Les labels associés à chaque observations # [0] étant sétosa, [1] étant versicolor et
	<pre># [2] virginica target = iris.target target</pre>
Out[7]:	array([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
In [8]:	<pre># On limite l'analyse aux deux variables petal length, petal width # On affecte y=1 pour la clsse Setosa = 1 et 0 ailleurs X = iris.data[:, (2, 3)]</pre>
	y = (iris.target == 0).astype(np.int) /var/folders/lg/y19wscx13n7g_vwjxl13gd3m0000gn/T/ipykernel_8761/536534188.py:4: DeprecationWarning: `np.int` is a deprecated alias for the builtin `int`. To silence this warning, use `int` by itself. Doing this will not modify any behavior and is safe. When replacing `int` int`.
	<pre>np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the precision. If you wish to review your current use, check the release note link for additional information. Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations y = (iris.target == 0).astype(np.int)</pre>
In [9]:	2- Dispersion des données plt.figure(figsize=(10, 4))
	<pre>plt.plot(X[y==0, 0], X[y==0, 1], "bs", label="Not Iris-Setosa (0)") plt.plot(X[y==1, 0], X[y==1, 1], "yo", label="Iris-Setosa (1)") plt.xlabel("Petal length", fontsize=14) plt.ylabel("Petal width", fontsize=14) plt.legend(loc="lower right", fontsize=14)</pre>
	plt.legend(loc-lower light, lonesize=14) plt.show()
	ug 15 -
	0.5 Not Iris-Setosa (0) Iris-Setosa (1)
	Petal length 3- Entrainement du perceptron
In [10]:	# Entrainement du perceptron # Les hyperparamètres étant le nombre max d'itérations et # la valeur de la tolérence (critère d'arret) tol=1e-3
	<pre>from sklearn.linear_model import Perceptron per_clf = Perceptron(max_iter=100, tol=1e-3, random_state=42)</pre>
Out[10]:	<pre>per_clf.fit(X, y) Perceptron(max_iter=100, random_state=42)</pre>
In [11]:	4- Paramètres du modèle # Valeur de la constante
Out[11]:	per_clf.intercept_
In [12]:	# Les coefficients de pondérations des caractéristiques # respectivement petal length, petal width per_clf.coef_
ouc[12]:	array([[-1.4, -2.2]]) 4 - Représentation des frontières de décision
In [13]:	<pre>import matplotlib as mpl import matplotlib.pyplot as plt</pre>
	<pre>mpl.rc('axes', labelsize=14) mpl.rc('xtick', labelsize=12) mpl.rc('ytick', labelsize=12) a = -per_clf.coef_[0][0] / per_clf.coef_[0][1]</pre>
	<pre>b = -per_clf.intercept_ / per_clf.coef_[0][1] axes = [0, 7, 0, 3]</pre>
	<pre>x0, x1 = np.meshgrid(</pre>
	<pre>X_new = np.c_[x0.ravel(), x1.ravel()] y_predict = per_clf.predict(X_new) zz = y_predict.reshape(x0.shape) plt.figure(figsize=(10, 4))</pre>
	<pre>plt.ligure(ligs1ze=(10, 4)) plt.plot(X[y==0, 0], X[y==0, 1], "bs", label="Not Iris-Setosa (0)") plt.plot(X[y==1, 0], X[y==1, 1], "yo", label="Iris-Setosa (1)") plt.plot([axes[0], axes[1]], [a * axes[0] + b, a * axes[1] + b], "k-", linewidth=3)</pre>
	<pre>from matplotlib.colors import ListedColormap custom_cmap = ListedColormap(['#9898ff', '#fafab0']) plt.contourf(x0, x1, zz, cmap=custom_cmap)</pre>
	<pre>plt.xlabel("Petal length", fontsize=14) plt.ylabel("Petal width", fontsize=14) plt.legend(loc="lower right", fontsize=14) plt.axis(axes)</pre>
	3.0 plt.show()
	2.5 -
	2.0 - I N T T T T T T T T T T T T T T T T T T
	2.0 1.5 1.0 0.5 Not Iris-Setosa (0) Iris-Setosa (1)
In [14]:	Excercice 2: Perceptron multi-couche import numpy as np import matplotlib.pyplot as plt
In [14]:	Excercice 2: Perceptron multi-couche import numpy as np import matplotlib.pyplot as plt from sklearn.datasets import load_iris from sklearn.model_selection import train_test_split
In [14]:	Excercice 2: Perceptron multi-couche import numpy as np import matplotlib.pyplot as plt from sklearn.datasets import load_iris
In [14]: In [15]:	Excercice 2: Perceptron multi-couche import numpy as np import matplotlib.pyplot as plt from sklearn.datasets import load_iris from sklearn.model_selection import train_test_split 1- Lecture des données
In [14]: In [15]: In [16]:	Excercice 2: Perceptron multi-couche import numpy as np import matplotlib.pyplot as plt from sklearn.model_selection import train_test_split 1- Lecture des données np.random.seed(10) # initialiser le générateur pseudo-aléatoire pour pouvoir répéter l'expérience from sklearn import datasets from sklearn import datasets
In [14]: In [15]: In [16]:	Excercice 2: Perceptron multi-couche import numpy as np import matplotlib.pyplot as plt from sklearn.datasets import load_iris from sklearn.model_selection import train_test_split 1- Lecture des données np.random.seed(10) # initialiser le générateur pseudo-aléatoire pour pouvoir répéter l'expérience from sklearn import datasets iris = datasets.load_iris() # On limite l'analyse aux deux variables petal length, petal width # On afforte y=1 pour la classe Setosa = 1 et 0 alileurs x = iris.datal:, i] y = (ris.sterget == 0).astype(np.int) //var/folders/lg/yl9wscxl3n7g_vwjxll3gd3m0000gn/T/ipykernci_8761/863911844.py;4: DeprecationWarning: `np.int` is a deprecated alias for
In [14]: In [15]: In [16]:	Excercice 2: Perceptron multi-couche import numpy as np import matplocitib, pyplot as plt from sklearn.model_selection import train_test_split 1- Lecture des données np.random.secd(10) # initialiser lo génératour pseudo-aléatoire pour pouvoir répéter l'expérience from sklearn import datasets iris = datasets.load_iris() # On limite l'analyse aux deux variables petal length, petal width # On affecte y-1 pour la clase Setosa = 1 et 0 ailleurs X = iris.data(1, 1) y = (iris.target == 0).astype(np.int)
In [14]: In [15]: In [17]:	Excercice 2: Perceptron multi-couche import numpy as np import matipolitib.pyplot as plt. from sklearn.model_selection import train_test_split 1- Lecture des données np.random.seed(10) # initialiser le générateur pseudo-aléatoirs pour pouvoir répéter l'expérience from sklearn mimport datasets iris = datasets.load_iris() # on ifinite l'analyse aux deux variables petal length, petal width # on affecte y-i pour la classe detosa - i et 0 ailleurs x = iris.data[:; :] y = (frint.strept == 0).astype(np.int) //war/foldersg/lg/y19wexx13n7q_vwjxl13gd3m0000qn/7/jpykernel_8761/8639118da.py.4: DeprecationWarning: `np.int` is a deprecated alias for the builtin' int'. To stlence this warning, use 'int' by itself. Doing this will not modify any behavior and is safe. When replacing 'np.int' you may wish to use e.g. 'np.int64' or 'np.int22' to specify the precision. If you want to review your current use, check the release note link for additional information.
In [14]: In [15]: In [17]:	Excercice 2: Perceptron multi-couche Import numpy as sp Import numpy as numpy as sp Import numpy as numpy as sp Import numpy as sp Import numpy as numpy as sp Import numpy as numpy a
In [14]: In [15]: In [17]:	Excercice 2: Perceptron multi-couche import numpy as mp import mathoritio.mynici as pit from attention.model_selection import train_test_split. 1- Lecture des données mp. vandom.anoid(10) # initializar la génératour pseudo-aléanoire pour pouvoir régéter l'expérience from attention import dotasats initis - distances aloud_irial() # On imitie l'analyse aux deux variables petal length, petal Width # On affecte p-i pour la class étous et ous - 1 et 0 ailleurs # = iris indance; lood_irial() //ar/iolous/luf/loweninining.mynininin //ar/iolous/luf/lowenining.myninininininininininininininininininini
<pre>In [14]: In [15]: In [16]: In [17]:</pre> Out [18]:	Excercice 2: Perceptron multi-couche import numpy as np import numpy
In [14]: In [15]: In [17]: Out [18]:	Excercice 2: Perceptron multi-couche import numpy as no import material individual information. From ablacen import detacets in a second of interest and or interest in a second of interest in a se
In [14]: In [15]: In [17]: In [18]: In [19]:	Excercise 2: Perceptron multi-couche Import manufolia support is application in general section in the section of the section
In [14]: In [15]: In [17]: In [18]: In [19]:	Excercice 2: Perceptron multi-couche Expect range as not provided to the pit provided and the provided and
In [14]: In [15]: In [17]: In [18]: In [19]:	**Recording 2: Perceptron multi-couche Import numpy as up Import nump
In [14]: In [15]: In [17]: Out [18]: In [20]:	Excercice 2: Perceptron multi-couche Import name and in the section (0) I be a section of the section (0) I be a section of the section (0) I could be a section of the section (0) I could be a section of the section of the section of the section (0) I be a section of the section of th
In [14]: In [15]: In [17]: Out [18]: In [20]:	**Not inis Setosa (0) **Inis Setosa (0) **Inis Setosa (0) **Inis Setosa (0) **Inis Setosa (1) **Petal length **Not inis Setosa (1) **Petal length **Not inis Setosa (0) **Inis Setosa (1) **Inis
In [14]: In [15]: In [17]: Out [18]: In [20]:	Excercise 2: Perceptron multi-couche impact mayor in the print may be not impact that the print of the print
In [14]: In [15]: In [17]: Out [18]: In [20]:	Not inicidental (0)
<pre>In [14]: In [15]: In [17]: In [19]: Out [20]: Out [20]:</pre>	Excercise 2: Perceptron multi-couche Expert large or any import individual control of the production
<pre>In [14]: In [15]: In [17]: In [19]: Out [20]: Out [20]:</pre>	Note this Sections (D) Petal length Petal leng
<pre>In [14]: In [15]: In [17]: In [19]: Out [20]: Out [20]:</pre>	Excercise 2: Perceptron multi-couche
In [14]: In [15]: In [16]: In [19]: Out [20]: Out [21]:	Excercise 2: Perceptron multi-couche tagent many as my term and term of the second se
<pre>In [14]: In [15]: In [16]: In [17]: In [19]: Out [20]: Out [21]:</pre>	Exercise 2: Perceptron multi-couche Page Page
<pre>In [14]: In [15]: In [16]: In [17]: In [20]: Out [20]: Out [20]:</pre>	Exercise 2: Perceptron multi-couche Perceptron multi-couche Perce
<pre>In [14]: In [15]: In [16]: In [17]: In [19]: In [20]: In [21]:</pre>	Excercise 2: Perceptron multi-couche Expert augus on any augus of the control of the county of the
<pre>In [14]: In [15]: In [16]: In [17]: In [19]: In [20]: In [21]:</pre>	Exercise 2 - Perception multi-couche despit starty at 15 million and 15 million
In [14]: In [15]: In [17]: In [19]: In [20]: In [21]:	Excercise 2: Perception multi-couche **Green larger to age to be a comment of the country of th
In [14]: In [15]: In [17]: In [19]: In [20]: In [21]:	Exercise 2. Perception multi-couche **Testal league 1. **League 1
In [14]: In [15]: In [17]: In [19]: In [20]: In [21]:	Execution C. 2. Perception multi-couche Execution C. 2. Perception multi-couche Fig. 4 manufacture of the country of the cou
<pre>In [14]: In [15]: In [17]: In [18]: In [20]: In [21]: </pre>	Exercise 2. Perception multi-couche Exercise 2. Perception multi-couche Exercise 3. Perception at 21 Security 1. Perception multi-couche Exercise 3. Perception at 21 Security 1. Perception multi-couche Exercise 3. Perception at 21 Security 1. Perception multi-couche Exercise 3. Perception at 21 Security 1. Perception multi-couche multi-couch
In [14]: In [15]: In [17]: In [19]: In [20]: In [21]:	Excercise 2: Perceptron multi-counts Excercise 2: Perceptron multi-counts Excercise 2: Perceptron multi-counts Figure accept and public and public acceptance of the pub
In [14]: In [15]: In [17]: In [19]: In [20]: In [21]:	Exercise 2: Perception multi-counts Depart language of programs of the prog