Nous allons maintenant travailler sur un algorithme d'apprentissage automatique, souvent appelé, même en français, algorithme de machine learning. L'idée est d'utiliser un grand nombre de données afin "d'apprendre à la machine" à résoudre un certain type de problème.

Cette idée d'apprentissage automatique ne date pas d'hier, puisque le terme de machine learning a été utilisé pour la première fois par l'informaticien américain Arthur Samuel en 1959. Pourquoi le machine learning est tant "à la mode" depuis quelques années ? Simplement parce que le nerf de la guerre dans les algorithmes de machine learning est la qualité et la quantité des données (les données qui permettront à la machine d'apprendre à résoudre un problème), or, avec le développement d'internet, il est relativement simple de trouver des données sur n'importe quel sujet (on parle de "big data"). À noter aussi l'importance des stratégies mises en place par les GAFAM (Google, Apple, Facebook, Amazon et Microsoft) afin de récupérer un grand nombre de données concernant leurs clients. Ces données sont très souvent utilisées pour "nourrir" des algorithmes de machine learning.

Nous allons étudier un algorithme d'apprentissage assez simple à appréhender : l'algorithme des "k plus proches voisins" (en anglais "K Nearest Neighbors" : KNN)

Afin de travailler sur un exemple, nous allons utiliser un jeu de données relativement connu dans le monde du machine learning : le jeu de données "iris".

En 1936, Edgar Anderson a collecté des données sur 3 espèces d'iris : "iris setosa", "iris virginica" et "iris versicolor"


iris setosa


iris virginica


iris versicolor

Pour chaque iris étudié, Anderson a mesuré (en cm) :

- la largeur des sépales
- la longueur des sépales
- la largeur des pétales
- la longueur des pétales

Par souci simplification, nous nous intéresserons uniquement à la largeur et à la longueur des pétales. Pour chaque iris mesuré, Anderson a aussi noté l'espèce ("iris setosa", "iris virginica" ou "iris versicolor"). Vous trouverez 15 de ces mesures dans le fichier iris_test.csv

iris_test.csv

```
longueur_petale,largeur_petale,espece
1.4,0.2,0
1.2,0.15,0
1.3,0.2,0
1.5,0.2,0
1.4,0.3,0
4.7,1.4,1
4.5,1.5,1
4.9,1.5,1
4.0,1.3,1
4.6,1.5,1
6.0,2.5,2
5.1,1.9,2
5.9,2.1,2
```

À vous de faire : Importer un fichier csv dans python

Après avoir placé le fichier iris_test.csv dans le même répertoire que votre fichier Python, étudier et tester le code suivant :

```
import pandas
# Traitement csv comma separated value valeurs séparées par des virgules
# iris est un dataframe (trame de données) similaire à un tableau
iris=pandas.read_csv("iris_test.csv")  # iris est le dataframe qui contient les données du *.csv
x=iris.loc[:,"longueur_petale"]        # sélectionne les données par nom de colonne (.loc)
y=iris.loc[:,"largeur_petale"]         # x est une série à laquelle on attribue les valeurs
lab=iris.loc[:,"espece"]               # successives de la colonne 'longueur petale'
print(iris)
```

```
...
*** Console de processus distant Réinitialisée ***
    longueur_petale  largeur_petale  espece
0               1.4            0.20       0
1               1.2            0.15       0
2               1.3            0.20       0
3               1.5            0.20       0
4               1.4            0.30       0
5               4.7            1.40       1
6               4.5            1.50       1
7               4.9            1.50       1
8               4.0            1.30       1
9               4.6            1.50       1
10              6.0            2.50       2
11              5.1            1.90       2
12              5.9            2.10       2
13              5.6            1.80       2
14              5.8            2.20       2
>>>
```

A quoi sert Pandas ?

_ _

_ _

_ _

_ _

_ _

_ _

_ _

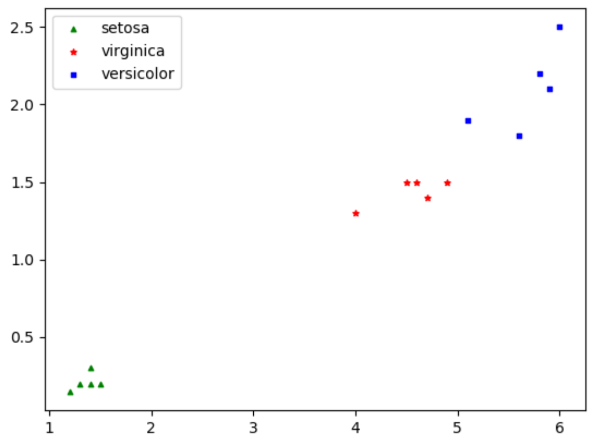
_ _

À vous de faire : Tracer les données du csv sous forme de nuages de points

Vous disposez du méménto *python matplotlib sur le serveur du lycée*

```
import matplotlib.pyplot as plt
# Affichage des données sous forme de nuages de points (x,y) appartenant au label lab
plt.scatter(x[lab==0],y[lab==0], s=10, marker='^', color='g',label='setosa')
_ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _
plt.legend()
plt.show()
```

Faire de même pour les labels 1 : virginica et 2 : versicolor.



Expliquer les lignes :

plt.legend() : _

_ _

plt.show() : _

_ _

plt.scatter(x,y, s=, marker='', color='',label='') : _

_ _

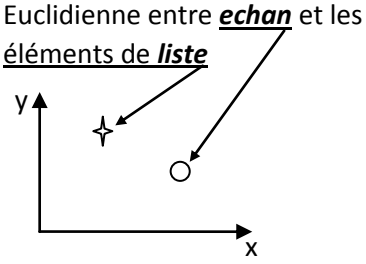
_ _

Positionner un point, sur votre graphique, de coordonnées (5.1,1.65) de taille 40 identifié par une croix de couleur noire.
Ce nouvel iris à quelle espèce appartient-il ?
Peut-on écrire un programme qui le détermine ? bien sur, voici les étapes (→)de ce calcul !

→ Convertir le dataframe en liste de tuples (utiliser iris.csv) ***liste**=list(zip(x,y,lab)*

→ Acquérir un nouvel échantillon de la forme ***echan**=[x,y,espece]* utiliser *input*

→ Calculer la distance Euclidienne entre ***echan*** et les éléments de ***liste***



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

→ Editer la liste des distances

PSEUDO CODE

Définir dist_euclid (echan,liste) :

Initialiser dist=[ ]

Parcourir la liste [i] :

$$\text{distance} = \sqrt{(y_1 - y_2)^2 + (x_1 - x_2)^2}$$

ajouter à dist [[distance, espece]]

retourner dist = [[…,…], […,…], _ _ _ _ _ _, […,…]]

→ Trier, dans le sens croissant, la liste des distances dist.

CE CODE EST CONNU

Tri par insertion

retourner dist_triées = [[…,…], […,…], _ _ _ _ _ _, […,…]]

→ Quelle est l'espèce dominante des k plus proches voisins ?

→ Acquérir k

On cherche la valeur max

PSEUDO CODE

Définir KNN (dist_triées, k) :

Initialiser voisins = [[0,0], [1,0], [2,0]] de le forme [espece,qt] avec qt quantité à définir par la fonction

Parcourir dist_triées de i vers k :

Si espece de dist_triées[i] = setosa :

Incrémenter qt de voisins[[0,+1]

SiSinon espece de dist_triées[i] = virginica :

Incrémenter qt de voisins[[1,+1]

SiSinon espece de dist_triées[i] = versicolor :

Incrémenter qt de voisins[[2,+1]

Chercher la valeur max de qt de voisisns

Attribuer l'espece dominante des k voisisns à ***echan***

Afficher ***echan*** sur le graphe avec les bons paramètres (marker, coulor)