Prédiction des espèces d'iris

Christophe Viroulaud

Première - NSI

Algo 06

En 1936, le biologiste *Ronald Fisher* a rassemblé les mesures de trois espèces d'iris.

Iris setosa

Iris versicolor

Iris virginica

Comment prédire une information nouvelle à partir de données brutes ?

Sommaire

Étude des données - Données étiquetées

petal_length	petal_width	species
4.5	1.7	virginica
4.6	1.5	versicolor
4.6	1.3	versicolor
4.6	1.4	versicolor
4.7	1.4	versicolor
4.7	1.6	versicolor
4.7	1.4	versicolor
4.7	1.2	versicolor
4.7	1.5	versicolor
4.8	1.8	versicolor
4.8	1.4	versicolor
4.8	1.8	virginica
4.8	1.8	virginica

FIGURE 1 – La mesure de chaque fleur a été **étiquetée** : la variété de l'iris a été déterminée.

Sommaire

Présentation graphique

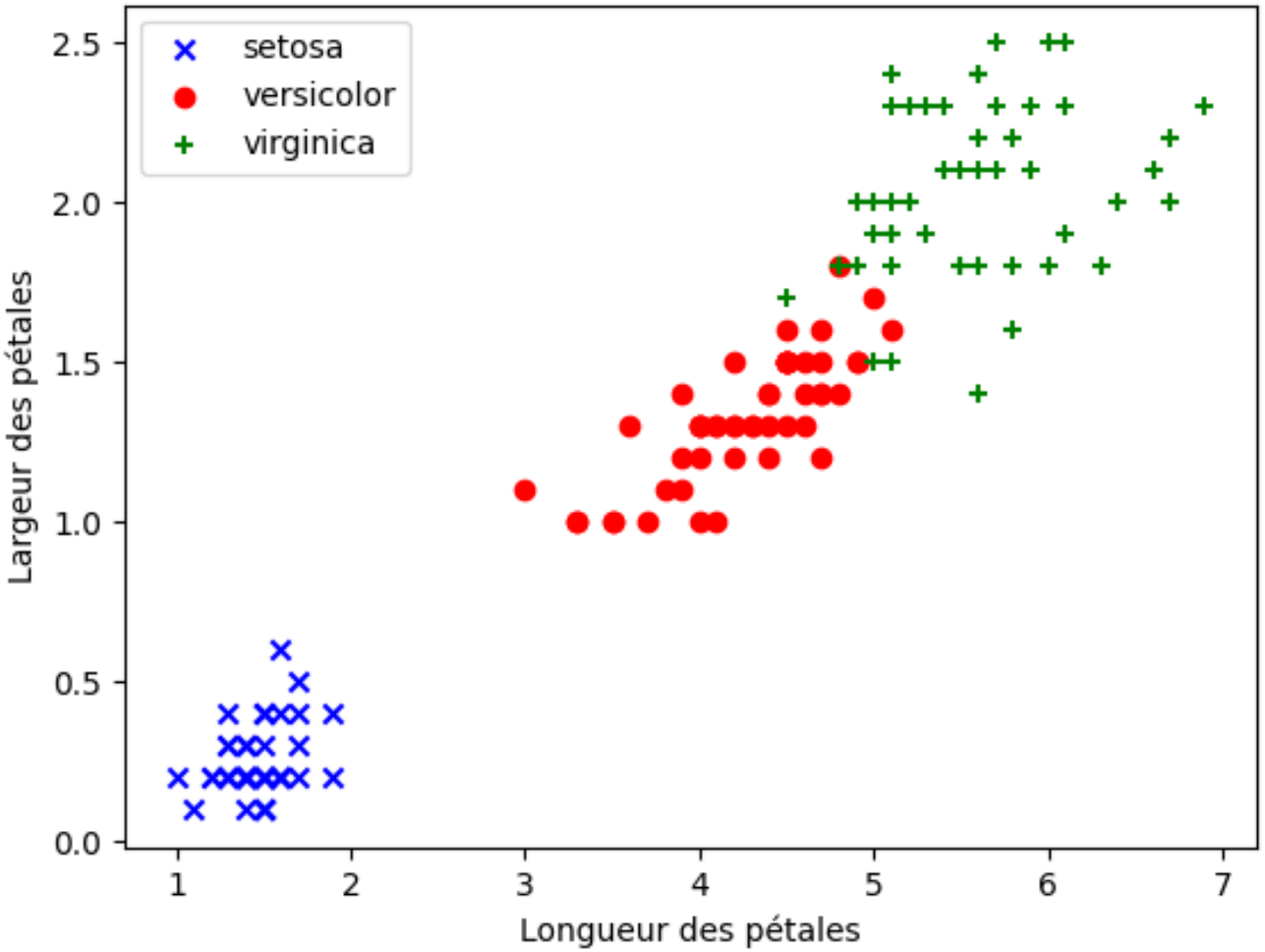


FIGURE 2 – Variétés d'iris en fonction de leurs mesures : Les mesures permettent de différencier les iris.

Sommaire

Utiliser les données

Activité 1 :

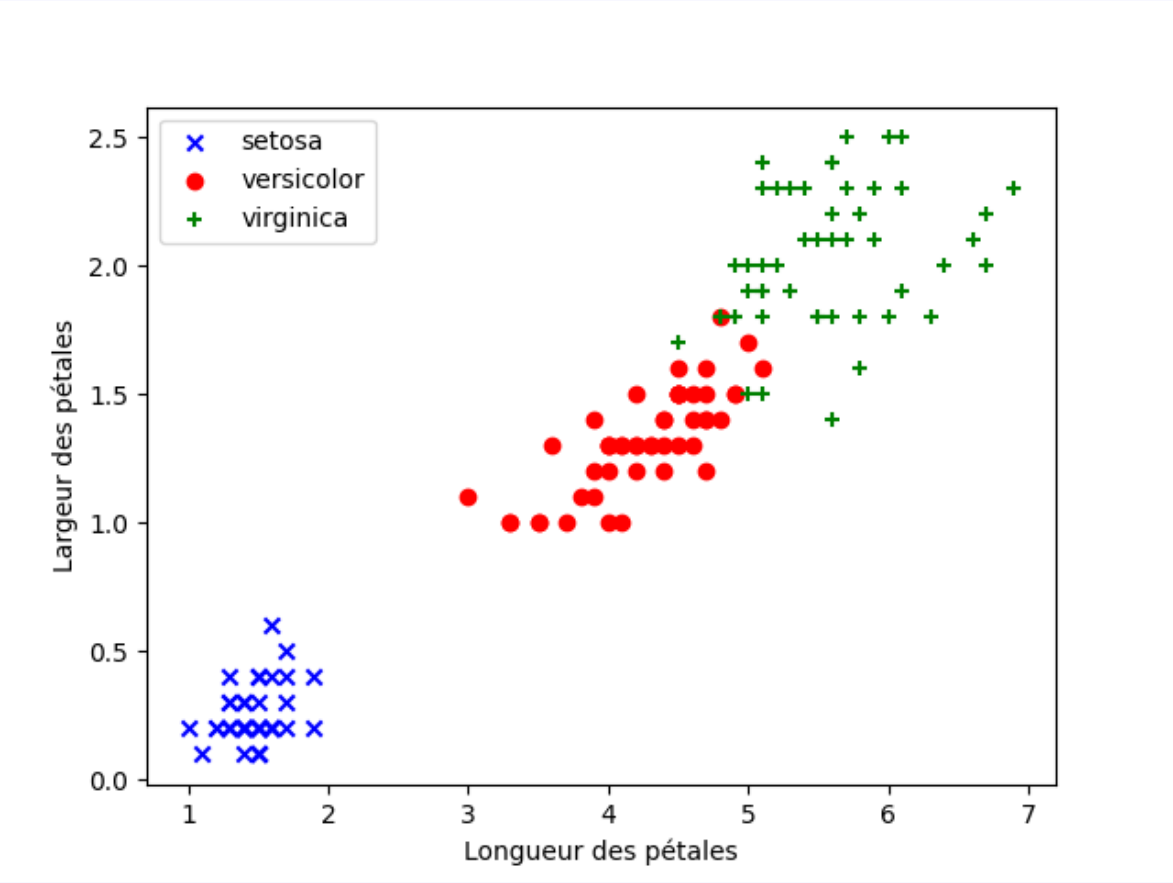


FIGURE 3 – Déterminer la variété des iris suivants :

longueur	1	6	5.1	2.5
largeur	0.5	2.5	1.55	0.85

Correction

longueur	1	6	5.1	2.5
largeur	0.5	2.5	1.55	0.85
variété	setosa	virginica	ambigu	ambigu

Observation

Pour certaines mesures, il est difficile de déterminer l'espèce de l'iris.

Sommaire

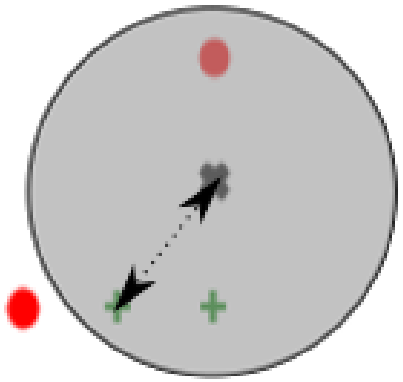
Algorithme kNN - Présentation

À retenir

L'algorithme **k Nearest Neighbors** (K plus proches voisins) détermine la variété de l'iris inconnu à partir de celles des **k** voisins les plus ressemblants.
C'est un algorithme d'apprentissage machine **supervisé** : les données initiales sont étiquetées.

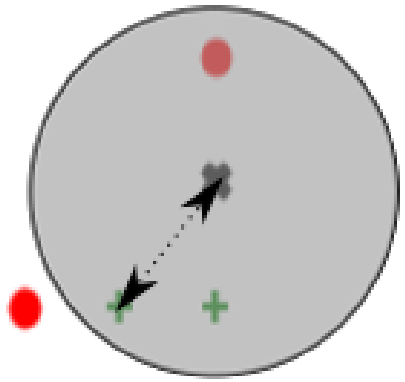
Pour déterminer la variété d'un iris inconnu :

- regarder la variété d'un nombre k de voisins,



Pour déterminer la variété d'un iris inconnu :

- regarder la variété d'un nombre k de voisins,



- attribuer à la fleur inconnue, la variété la plus présente parmi ses k voisins.

Sommaire

Choix du k

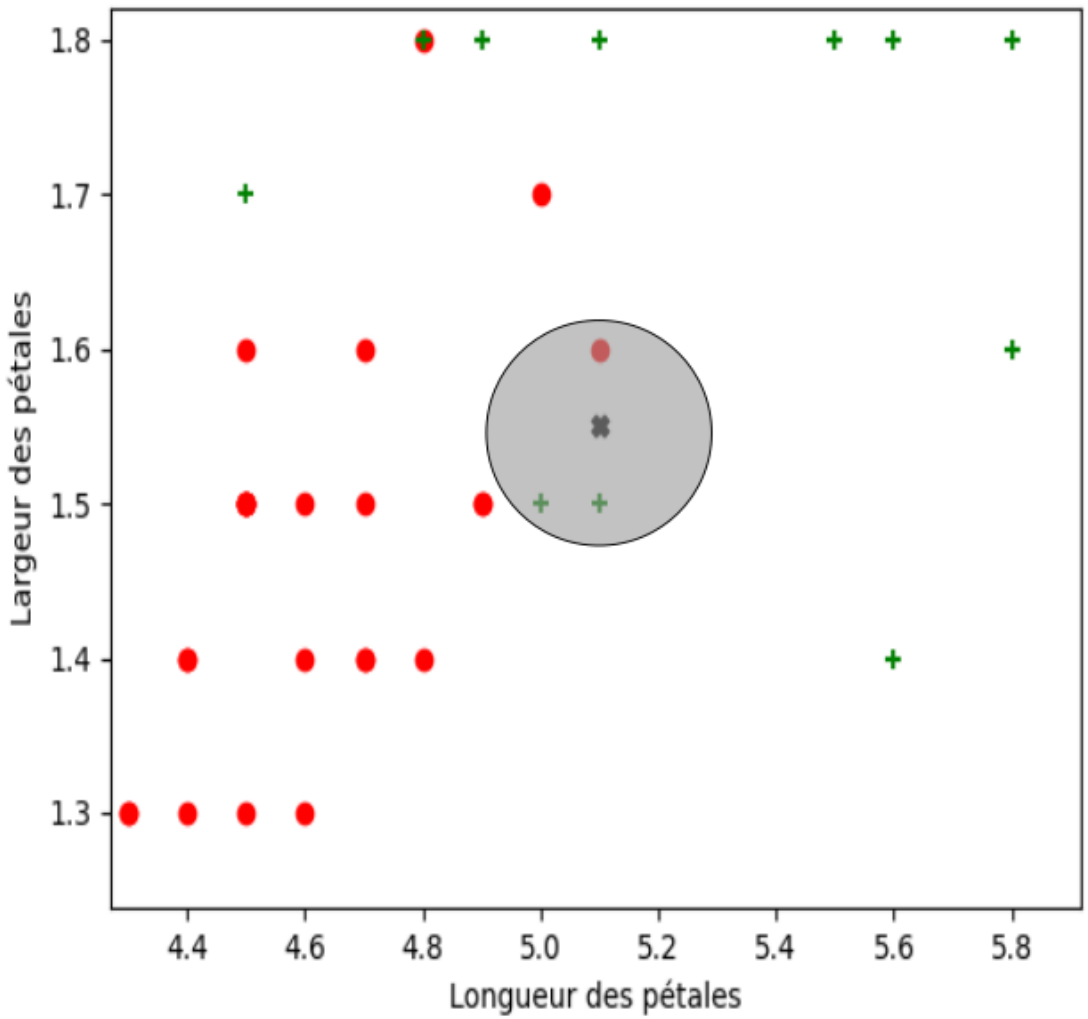


FIGURE 4 – Détermination de l'iris (5.05, 1.5) pour $k = 3$

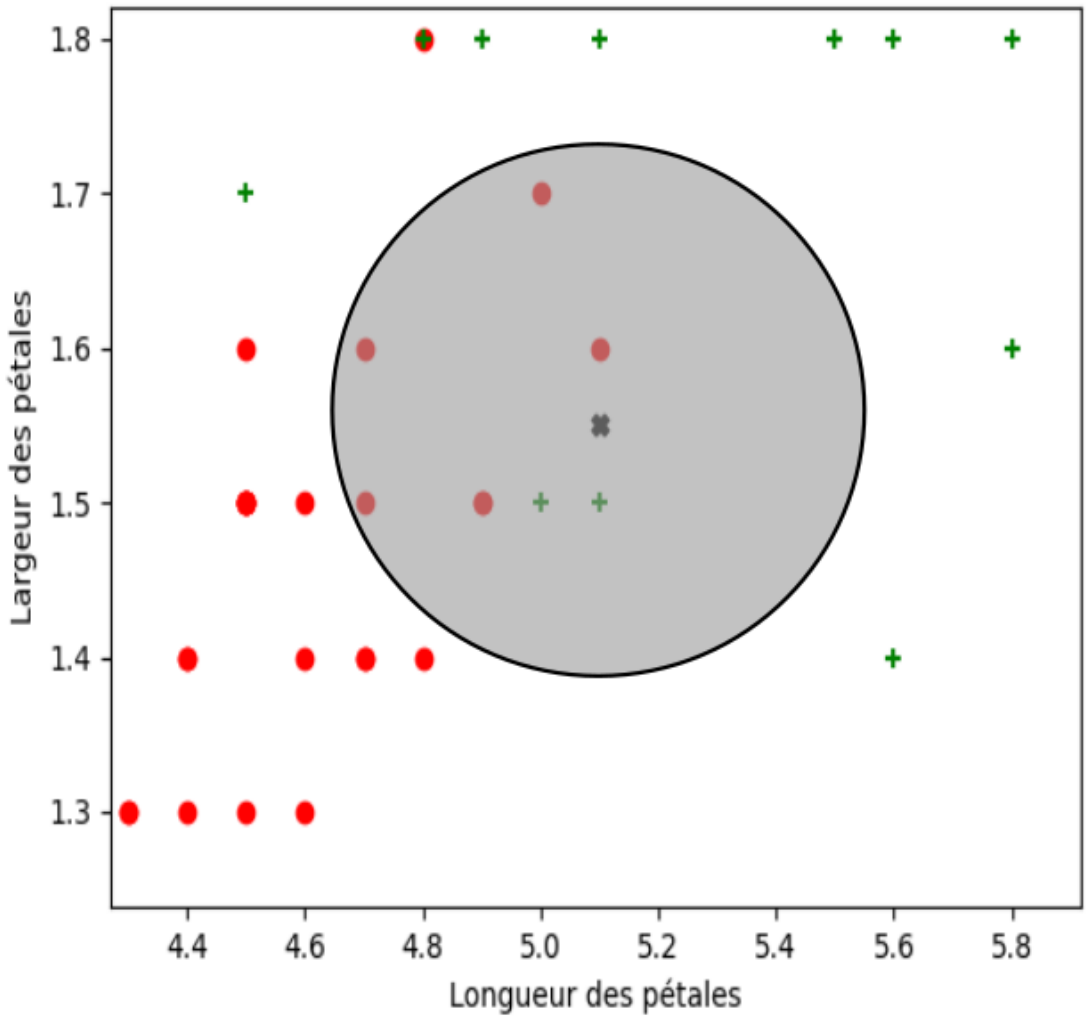


FIGURE 5 – Détermination de l'iris (5.05, 1.5) pour $k = 7$

À retenir

Un bon choix de la valeur **k** est difficile a priori. Plusieurs tests permettent de déterminer la valeur la plus adaptée à l'étude en cours.

Remarque

En pratique on partage les données en deux parties :

- les données d'entraînement,
- les données tests.

On teste différentes valeurs de k avec les données tests et on choisit la plus adaptée.

Sommaire

Calcul de la distance

À retenir

Il existe plusieurs méthodes pour mesurer la distance entre l'élément étudié et son voisin :

- distance euclidienne,
- distance de Manhattan.

$$d = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}$$

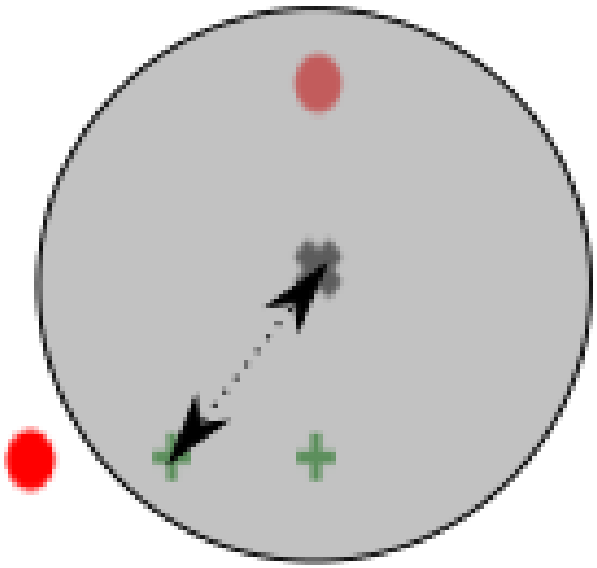


FIGURE 6 – distance euclidienne

$$d = |x_A - x_B| + |y_A - y_B|$$

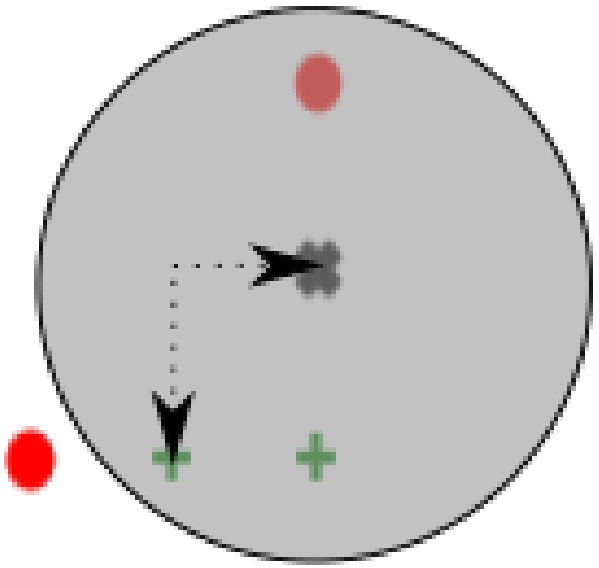


FIGURE 7 – distance de Manhattan

Sommaire

Implémentation

L'algorithme kNN peut s'écrire :

- Charger les données dans le programme.
- Choisir k.
- Stocker les mesures de la fleur inconnue.
- Calculer la distance euclidienne entre la fleur inconnue et tous les autres iris.
- Sélectionner les k plus proches iris (en distance) de la fleur inconnue.
- Affecter la variété majoritaire des k plus proches iris (en distance) à la fleur inconnue.

Activité 2 :

1. Télécharger et extraire le dossier compressé **`iris-eleve.zip`** depuis le site https://cviroulaud.github.io
2. Ouvrir le fichier **`data-iris.csv`** avec un tableur pour observer les données.
3. Ouvrir le fichier **`iris-eleve.py`**

Correction

petal_length	petal_width	species
1.4	0.2	setosa
1.4	0.2	setosa
1.3	0.2	setosa

Activité 3 :

4. Compléter la fonction **`charger_donnees`** en utilisant les informations du fichier **`csv`**.
5. Compléter la fonction **`distance`** qui calcule le carré de la distance euclidienne entre deux points du plan.
6. Compléter la fonction **`calculer_distances`**.
7. Compléter enfin la fonction **`trouver_variete`**. Le dictionnaire **`compteur_voisins`** compte le nombre d'apparitions de chaque variété parmi les **`k`** premiers voisins.

Correction


```
1  def charger_donnees(nom_fichier: str) -> list:
2      fichier = open(nom_fichier, encoding="utf8")
3      data_iris = csv.DictReader(fichier, delimiter=",")
4      tab_iris = []
5      # Pour chaque ligne de données
6      for iris in data_iris:
7          tab_iris.append(
8              {"espece": iris["species"],
9               "longueur": float(iris["petal_length"]),
10              "largeur": float(iris["petal_width"])})
11
12     fichier.close()
13     return tab_iris
```

Correction


```
1 def distance(connu: dict, inconnu: dict) -> float:
2     return (connu["longueur"]-inconnu["longueur"])**2 + \
3         (connu["largeur"]-inconnu["largeur"])**2
```

Correction


```python
def calculer_distances(donnees: list, inconnu: dict) -> list:
    distances = []
    for iris in donnees:
        # iris est un dictionnaire
        d = distance(iris, inconnu)
        # stocke la distance pour cet iris
        distances.append((iris["espece"], d))

    # trie les iris en fonction de la distance
    distances.sort(key=lambda fleur: fleur[1])
    return distances
```

Correction


```
1  def trouver_variete(k: int, distances: list) -> str:
2      # compte le nombre d'occurences de chaque variété
3      compteur_voisins = {}
4      for i in range(k):
5          # espèce de l'iris de rang i
6          nom = distances[i][0]
7          # vérifie si l'espèce a déjà été référencée
8          if nom in compteur_voisins:
9              compteur_voisins[nom] += 1
10         else:
11             compteur_voisins[nom] = 1
```

Code 1 – Début de la fonction **trouver_variete**

```
1      # recherche la variété avec la plus grande valeur
       dans compteur_voisins
2      maxi = 0
3      nom_maxi = 0
4      for nom, quantite in compteur_voisins.items():
5          if quantite > maxi:
6              maxi = quantite
7              nom_maxi = nom
8
9      return nom_maxi
```

Code 2 – Fin de la fonction **trouver_variete**

Activité 4 : Tester la fonction avec $k = 3$ puis $k = 7$, pour l'iris inconnu de mesures :

- longueur : 5,1
- largeur : 1,55

```
1  k = 3
2  inconnu = {"espece": "inconnu",
3             "longueur": 5.1,
4             "largeur": 1.55}
5
6  varietes = charger_donnees("data-iris.csv")
7  distances_cible = calculer_distances(varietes, inconnu)
8  variete = trouver_variete(k, distances_cible)
9
10 print("La variété est", variete)
```

Code complet

Le code complet est accessible ici.