UNIVERSITE BADJI MOKHTAR-ANNABA

DEPARTEMENT D'INFORMATIQUE
M1 IATI

Apprendre à Classifier

par

Dr. Samira LAGRINI

Année universitaire:2024/2025

Plan du cours

- ❑Notions de base
- ❑ la classification et ses types
- ❑ Algorithmes utilisés
- ❑ Métriques d’évaluation

Basic concepts

Problème:
Supposons qu'un chef de produit vous dit : « Nous devons être capables de prédire si un client particulier restera avec nous. Voici les logs des interactions des clients avec notre produit sur cinq ans. »

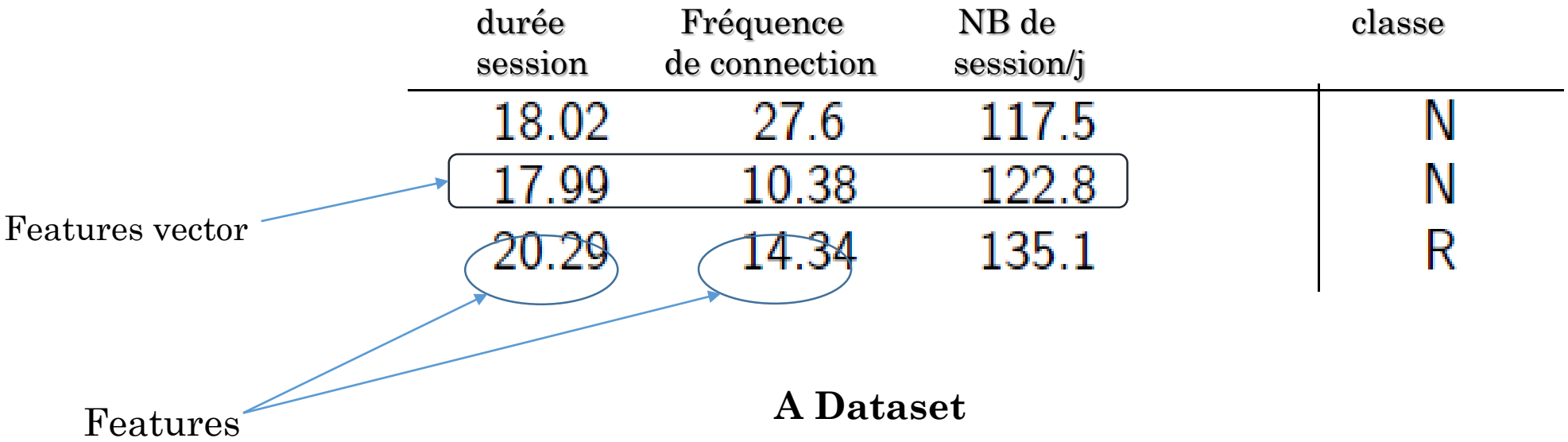
```
66.82.9.16 - - [30/Aug/2004:01:12:27 +0000] "GET /images/main_nor2.gif HTTP/1.1" 200 73797 "http://www.utilities.h12.ru/Fre
66.82.9.16 - - [30/Aug/2004:01:13:56 +0000] "GET /images/main_nor2.gif HTTP/1.1" 200 73797 "http://www.utilities.h12.ru/Fre
66.82.9.16 - - [30/Aug/2004:01:14:39 +0000] "GET /images/main_nor2.gif HTTP/1.1" 200 73797 "http://www.utilities.h12.ru/Fre
80.170.104.145 - - [30/Aug/2004:01:18:13 +0000] "GET /download.htm HTTP/1.1" 200 7901 "http://telecharger.01net.com/windows/
80.170.104.145 - - [30/Aug/2004:01:18:15 +0000] "GET /favicon.ico HTTP/1.1" 200 1406 "-" "Mozilla/5.0 (Windows; U; Windows
80.170.104.145 - - [30/Aug/2004:01:18:15 +0000] "GET /images/search_no.GIF HTTP/1.1" 200 215 "http://www.coolfilesearch.com/
80.170.104.145 - - [30/Aug/2004:01:18:15 +0000] "GET /images/search_hi.GIF HTTP/1.1" 200 215 "http://www.coolfilesearch.com/
80.170.104.145 - - [30/Aug/2004:01:18:16 +0000] "GET /images/search_tri.gif HTTP/1.1" 200 874 "http://www.coolfilesearch.com
80.170.104.145 - - [30/Aug/2004:01:18:36 +0000] "GET /index.html HTTP/1.1" 200 11393 "http://www.coolfilesearch.com/downloa
80.170.104.145 - - [30/Aug/2004:01:18:38 +0000] "GET /images/main3.jpg HTTP/1.1" 200 8721 "http://www.coolfilesearch.com/in
63.238.163.75 - - [30/Aug/2004:01:22:06 +0000] "HEAD / HTTP/1.1" 200 0 "-" "InternetSeer.com" coolfilesearch.com text/html
```

Est-il suffisant de prendre les données, les charger dans une bibliothèque pour obtenir la prédiction???????

Réponse : Non, il faut construire d'abord un **Dataset** (jeu de données)

Basic concepts

- Un **Dataset** est une collection de N **vecteur de caractéristiques** étiquetés.
- Les dimensions dans un vecteur de caractéristiques sont appelées ***features ou* caractéristique**
- **Les étiquetes** dans un **dataset** sont les classes à prédire,



A Dataset

Basic concepts in ML

- Transformer des données **brutes** en un **Dataset** est appelé '**ingénierie des caractéristiques**' ou **feature engineering**

feature engineering

durée session	Fréquence de connection	NB de session/j	classe
18.02	27.6	117.5	N
17.99	10.38	122.8	N
20.29	14.34	135.1	R

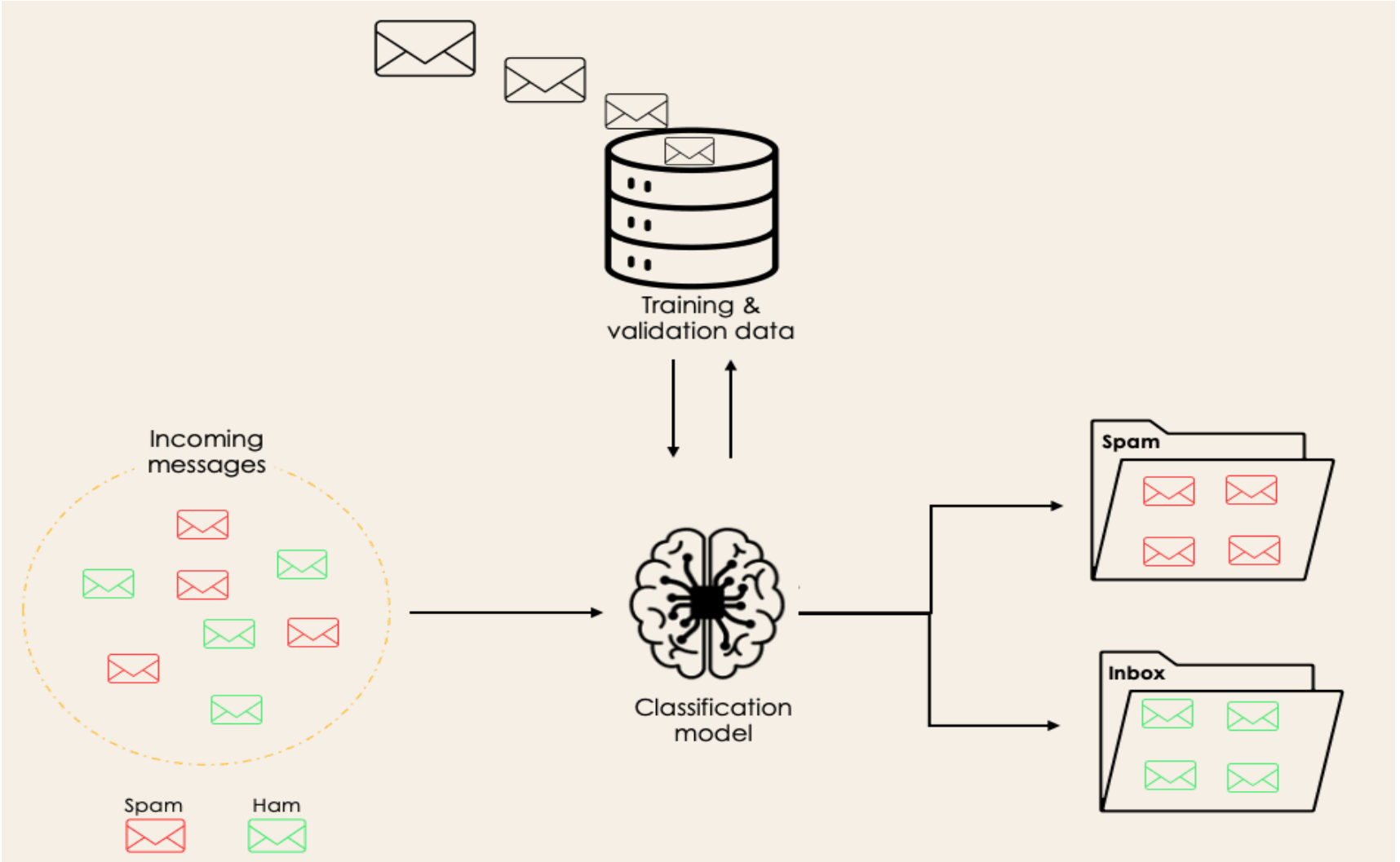
Feature engineering consiste à sélectionner, transformer ou créer des **caractéristiques** à partir des données brutes.

Apprendre à Classifier

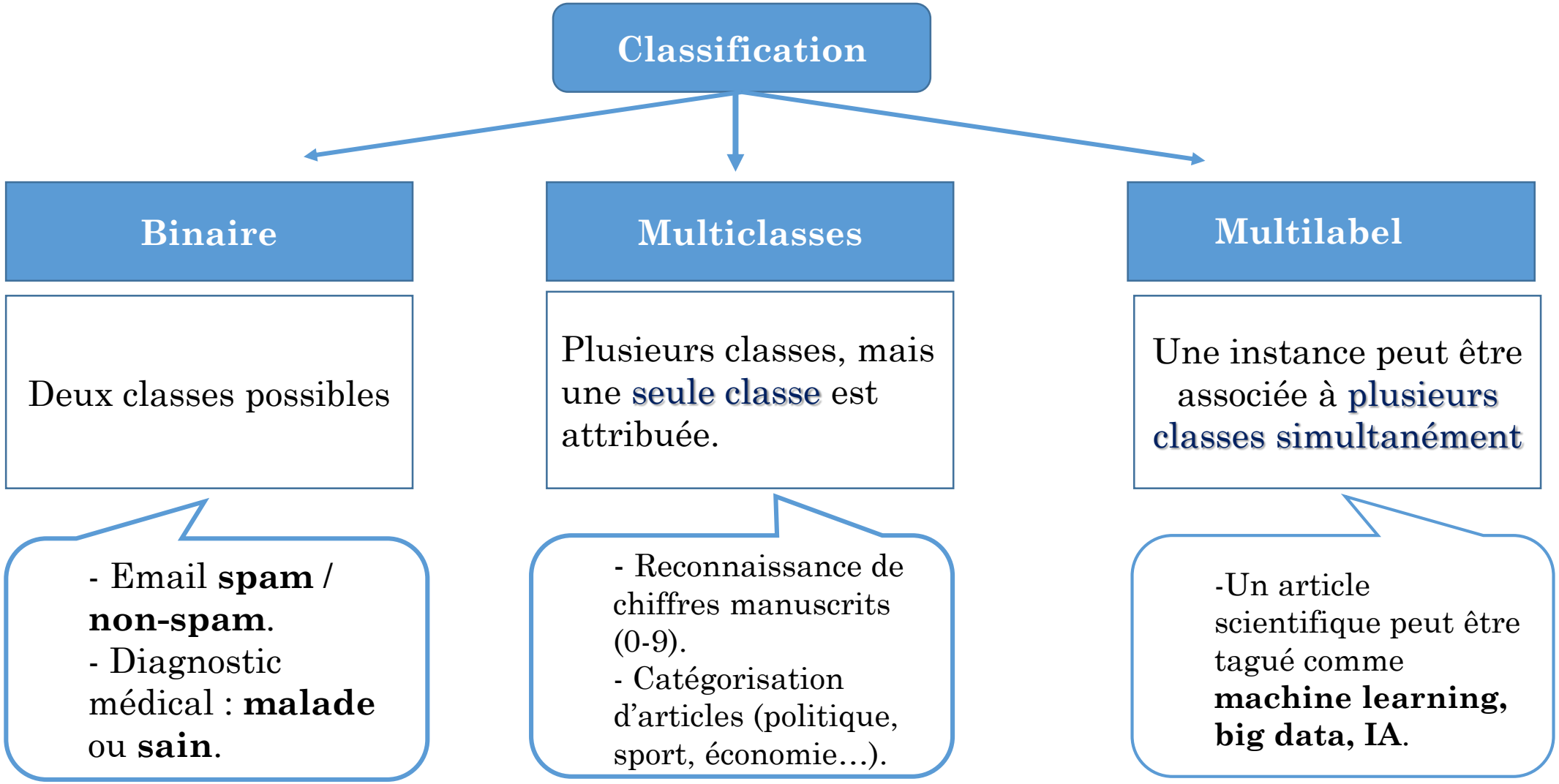
Définition

- La classification est une technique d'apprentissage supervisé où un modèle est entraîné sur des données étiquetées pour attribuer une classe à de nouvelles observations.
- Elle repose sur un ensemble de features et une variable cible catégorique.
- Le but est d'apprendre une fonction de décision qui assigne correctement une classe à une nouvelle donnée.

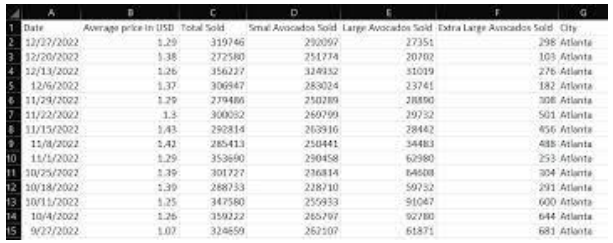
Définition


Training &
validation data
Incoming
messages
Spam
Ham
Classification
model
Spam
Inbox

Types de Classification

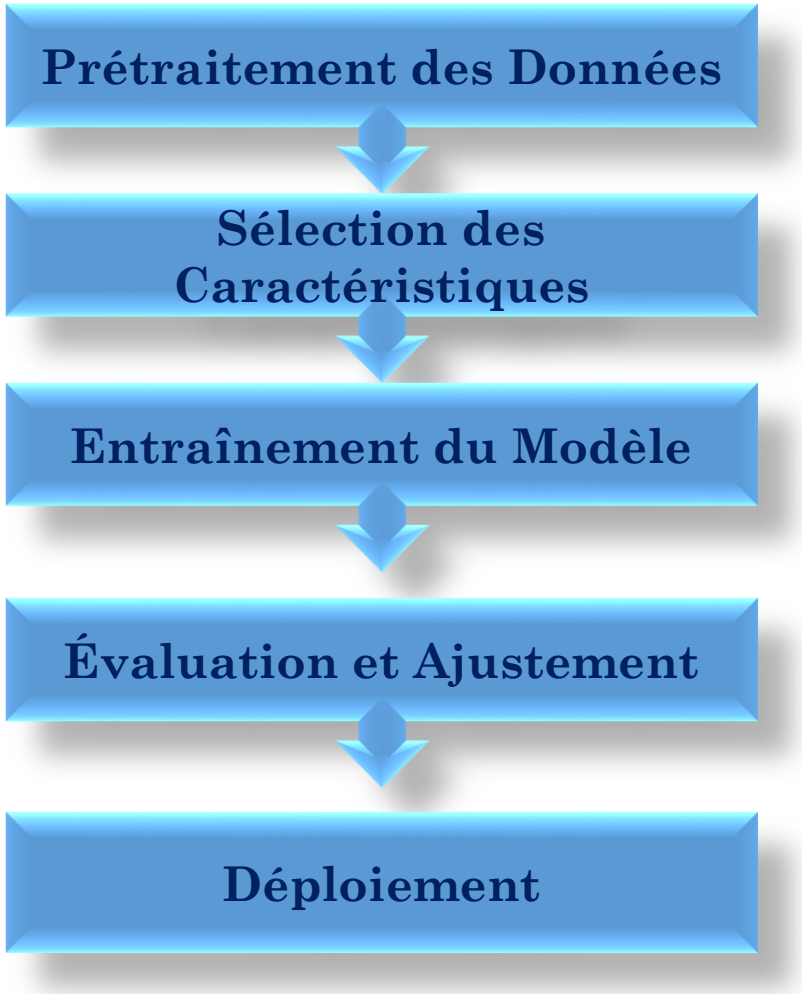


Étapes Pratiques pour construire un modèle de Classification



	A	B	C	D	E	F	G
1	Date	Average price in USD	Total Sold	Small Avocados Sold	Large Avocados Sold	Extra Large Avocados Sold	City
2	12/27/2022	1.29	319746	292097	27351	298	Atlanta
3	12/20/2022	1.38	272580	251774	20702	103	Atlanta
4	12/13/2022	1.26	356227	324932	31019	276	Atlanta
5	12/6/2022	1.37	306947	283024	23741	182	Atlanta
6	11/29/2022	1.29	279486	250289	28890	308	Atlanta
7	11/22/2022	1.3	300082	269799	29732	501	Atlanta
8	11/15/2022	1.43	292814	263916	28442	456	Atlanta
9	11/8/2022	1.42	285413	250441	34483	488	Atlanta
10	11/1/2022	1.29	353690	290458	62980	253	Atlanta
11	10/25/2022	1.39	301727	236814	64608	304	Atlanta
12	10/18/2022	1.39	288733	228710	59732	291	Atlanta
13	10/11/2022	1.25	347580	255933	91047	600	Atlanta
14	10/4/2022	1.26	359222	265797	92780	644	Atlanta
15	9/27/2022	1.07	324859	262107	61871	681	Atlanta

Dataset



Algorithmes courants en classification

K-Nearest Neighbors (KNN)

- Classe une instance en fonction des classes des k instances les plus proches dans l'espace des caractéristiques. Pas besoin d'entraînement explicite.

❖ **Fonctionnement**

1. **Stockage des données d'apprentissage**
2. **Calcul de la distance**
   Lorsqu'un nouvel échantillon x doit être classé, on calcule la distance entre x et tous les points d'entraînement.
3. **Sélection des k plus proches voisins**
4. **Vote majoritaire (Classification)**

- chaque voisin vote pour sa classe, et la classe majoritaire est attribuée à l'échantillon.

-En cas de Régression on prend la **moyenne** des valeurs des voisins.

Les distances courantes :

- **Distance Euclidienne (L2)** (la plus utilisée) :

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

- **Distance de Manhattan (L1)** :

$$d(x,y) = \sum_{i=1}^{n}|x_i - y_i|$$

- **Distance de Minkowski** (généralisation des précédentes) :

$$d(x,y) = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{\frac{1}{p}}$$

- **Distance Cosinus** (souvent utilisée pour des données textuelles) :

$$d(x,y) = 1 - \frac{x \cdot y}{||x||||y||}$$

Support Vector Machines (SVM)

Fonctionnement

- Trouve l'hyperplan optimal qui sépare les classes avec la marge maximale.
- Peut utiliser des noyaux (kernels) pour gérer des problèmes non linéaires.

Utilisation

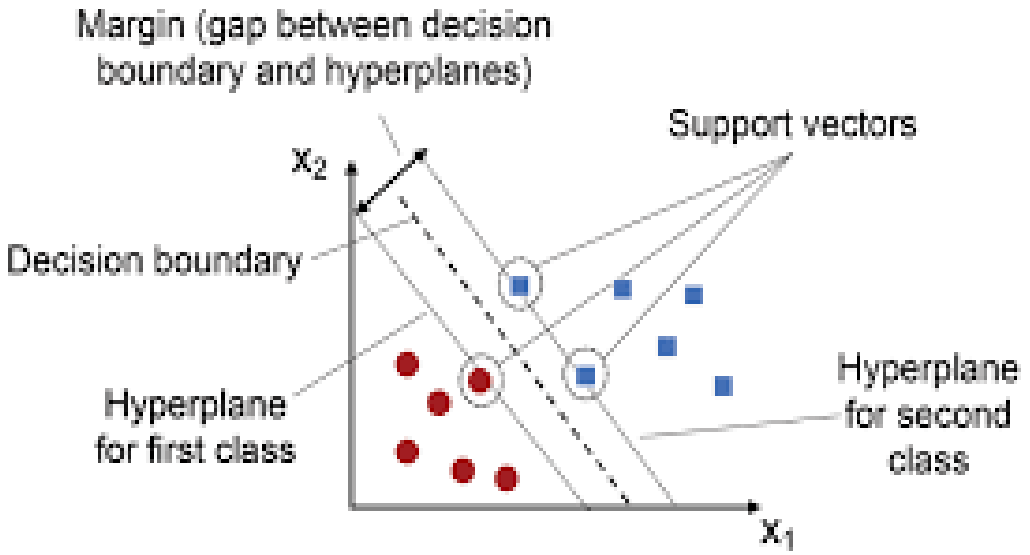
- Classification binaire et multiclasse (avec des adaptations).

- Avantages

Performant pour les petites et moyennes bases de données.

- Inconvénients

Lourd en calcul pour les grandes bases, nécessite un bon choix du noyau et des hyperparamètres.



Arbres de Décision (Decision Trees, DT)

Fonctionnement

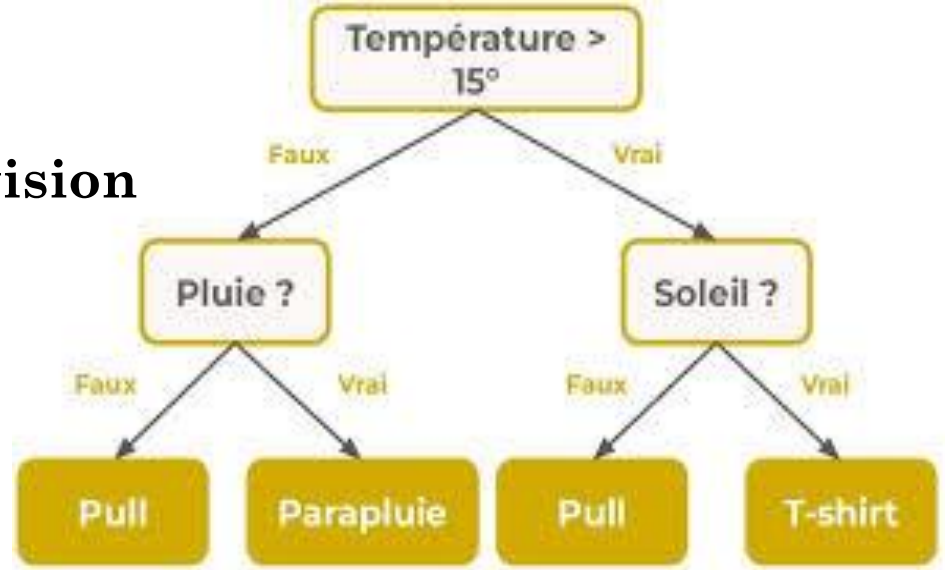
- Divise l'espace des caractéristiques en régions en posant des questions (tests) successives sur les caractéristiques (*ex: "Est-ce que la température > 15 ?"*).
- Chaque feuille de l'arbre représente une classe.

Comment construire l'arbre de décision???

1) **Entrée :** Ensemble d'entraînement
2) **Sélection de la première caractéristique pour la division**

- en se basant sur un critère d'impureté (calculé pour toutes les caractéristiques)
- Le critère d'impureté quantifie **l'hétérogénéité** des classes

  (ex: Gini, Entropie, Gain d'Information)).



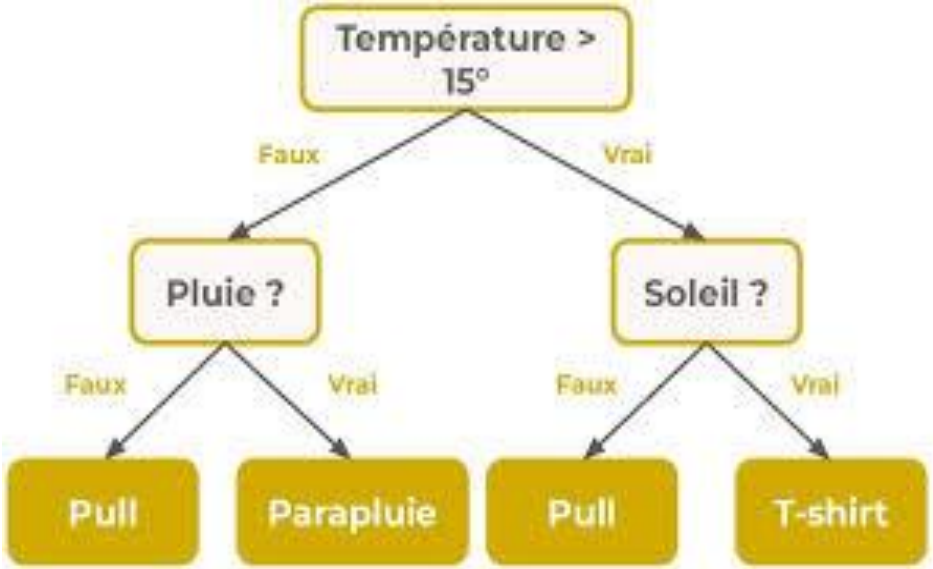
Arbres de Décision (Decision Trees, DT)

3) **Division des données en sous-groupes**, puis sélection d'une nouvelle caractéristique pour chaque sous-groupe.
4) **Arrêt de la division** lorsque les sous-groupes sont homogènes ou selon des critères d'arrêt.
5) **Prédiction** : L'arbre classifie ou prédit la valeur de nouveaux échantillons en suivant les décisions des nœuds jusqu'à atteindre une feuille (la classe).

- **Utilisation** : Classification binaire et multiclasse.
- **Avantages** : Facile à interpréter et Gère bien les données non linéaires.

- **Inconvénients**

Sur-apprentissage (Overfitting) si l'arbre devient trop profond,

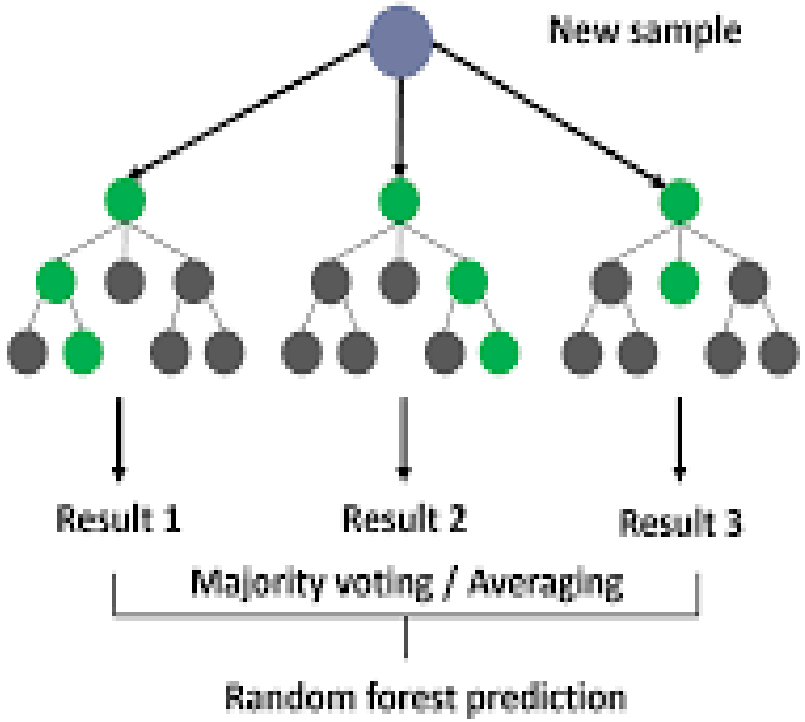


Random Forest (Forêt aléatoire)

- Principe : utilise plusieurs arbres de décision pour effectuer des prédictions (classification et de régression).
- Chaque arbre est construit en utilisant une version différente (aléatoire) des données d'entraînement et des caractéristiques

Cela signifie :

- Chaque arbre est entraîné sur un sous-ensemble aléatoire des données d'entraînement. généré par un **tirage avec remise** → certaines instances peuvent être répétées dans cet échantillon.
- Lors de la construction de chaque arbre, un **sous-ensemble aléatoire de caractéristiques** est choisi. Cela permet d'augmenter la diversité des arbres.



Processus de Construction d'une Random Forest

- **Étape 1** : Tirage aléatoire avec remise des sous-échantillons de données
- **Étape 2 :** Chaque sous-échantillon créé est utilisé pour entraîner un arbre de décision *(Construction des arbres de décision)*
- **Étape 3** : Sélection aléatoire des caractéristiques à chaque nœud lors de la construction de chaque arbre
- **Étape 4 : Prédiction par agrégation**
  - **Classification :** Chaque arbre vote pour une classe, et la classe majoritaire est choisie comme prédiction finale.
  - **Régression :** On prend la moyenne des prédictions de tous les arbres pour obtenir la prédiction finale.

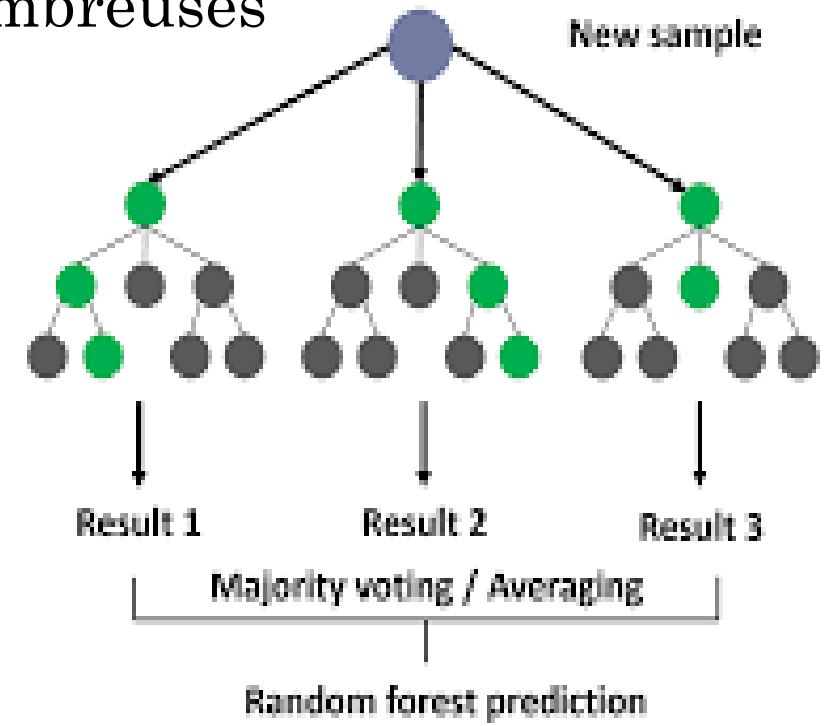
Random Forest (Forêt aléatoire)

Avantages

- Réduction du sur-apprentissage
- Plus Robuste aux données bruitées
- Capacité à traiter des jeux de données avec de nombreuses caractéristiques

Inconvénients

- Temps d'entraînement élevé
- Prédictions lentes



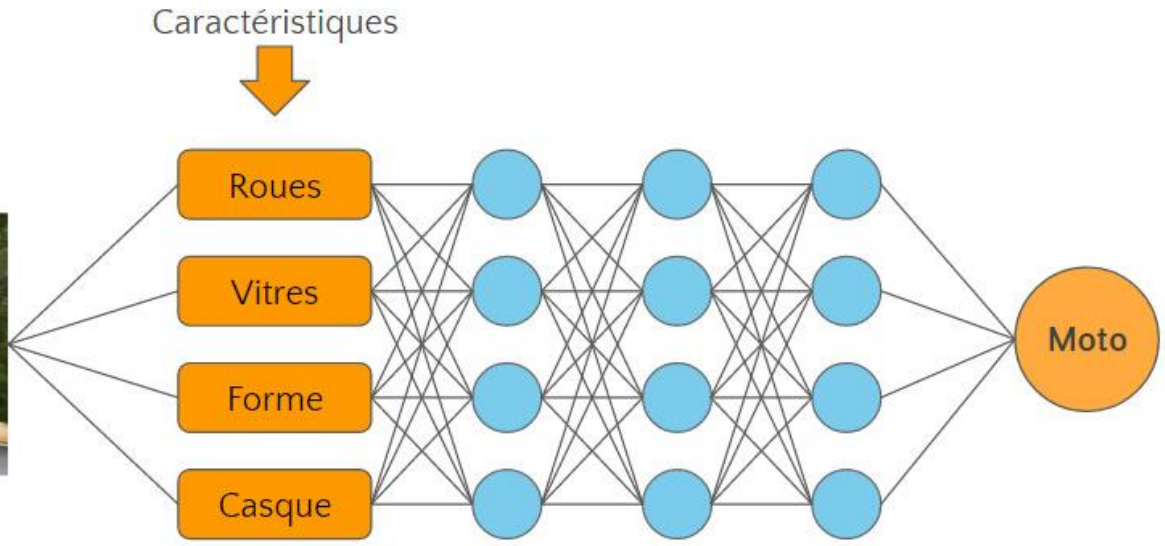
Réseaux de Neurones Artificiels (ANN - Artificial Neural Networks)

- Principe : Modèle inspiré du cerveau humain, composé de couches de neurones interconnectés qui apprennent des représentations complexes.

- Avantages :

  - Capable d'apprendre des relations complexes, utilisé pour les images, le texte…

- Inconvénients :
  - Temps d'entraînement élevé
  - besoin de grandes quantités de données d'entrainement.



Évaluation des performances d'un modèle de classification

Matrice de confusion

		Prédit par le classifier		
		Positif	**Négatif**	
Réalité	**Positif**	TP	FN	Rappel= TP/TP+FN
	Négatif	FP	TN	
		Precision= TP/TP+FP		Accuracy=TP+TN /TP+TN+FP+FN

TP : Nombre de prédictions correctes pour la classe 1.

TN : Nombre de prédictions correctes pour la classe 0.

FP: Nombre de prédictions incorrectes, pour la classe 0

FN : Nombre de prédictions incorrectes pour la classe 1

Indicateurs à partir de la matrice de confusion

- **Accuracy**: mesure la proportion des prédictions correctes par rapport au total des prédictions.

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

La précision : mesure la proportion des prédictions positives correctes parmi toutes les prédictions positives.

$$\text{Précision} = \frac{TP}{TP + FP}$$

Le rappel mesure la proportion des éléments de la classe positive correctement identifiés.

$$\text{Rappel} = \frac{TP}{TP + FN}$$

La F-mesure est une moyenne harmonique de la précision et du rappel, et est particulièrement utile lorsque les classes sont déséquilibrées.

$$F1 = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

Exemple:

- Supposons que vous entraîniez un modèle de classification binaire pour détecter si un e-mail est **spam** ou **non-spam**. Après avoir testé votre modèle, vous obtenez la matrice de confusion suivante :

	Prédit : Spam	Prédit : Non-Spam
Spam	TP=70	FN=30
Non-Spam	FP=10	TN=90

Accuracy :

$$\frac{70+90}{70+90+10+30} = \frac{160}{200} = 0.8 \Rightarrow 80\%$$

Precision :

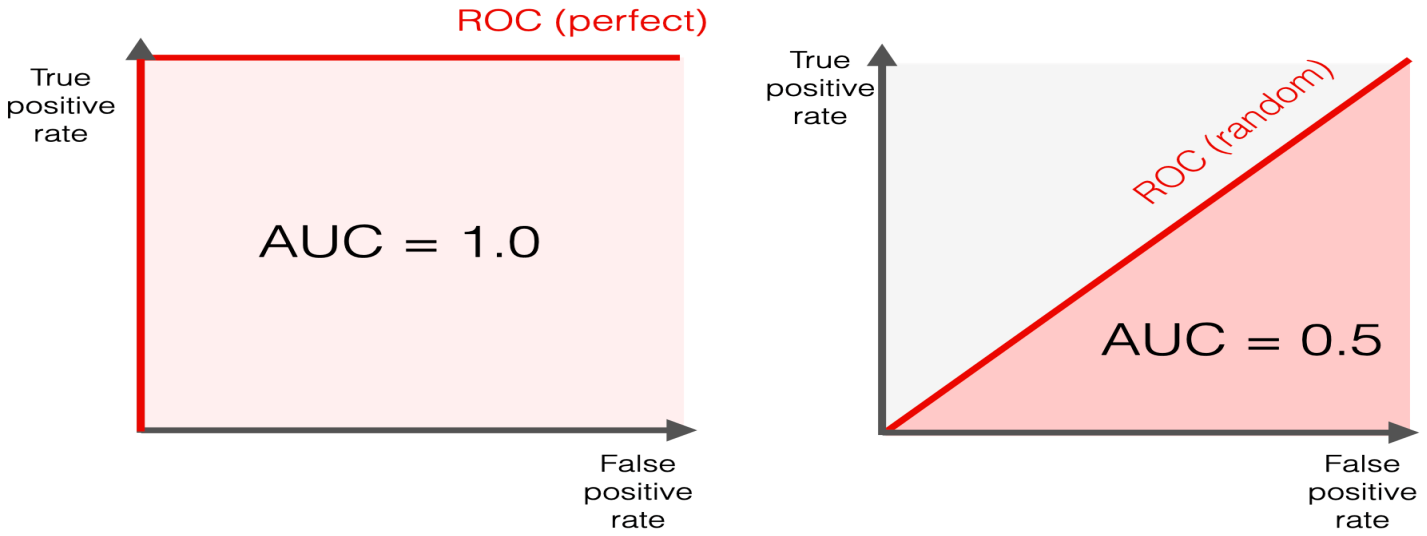
$$\frac{70}{70+10} = \frac{70}{80} = 0.875 \Rightarrow 87.5\%$$

Recall :

$$\frac{70}{70+30} = \frac{70}{100} = 0.7 \Rightarrow 70\%$$

AUC-ROC

- AUC (Area Under the Curve) est la surface sous la courbe ROC (Receiver Operating Characteristic).
- La courbe ROC trace le taux de vrais positifs (TPR) en fonction du taux de faux positifs (FPR) pour différents seuils de classification.
- Une AUC proche de 1 indique un bon modèle, tandis qu'une AUC proche de 0.5 indique un modèle aléatoire.



Comment évaluer la performance dans un problème de ***classification multi-classe*** *?*

Évaluation des performances
- *classification multi-classe-*

- Les calculs de précision, rappel et accuracy suivent des principes similaires, mais adaptés à plusieurs classes.

En fonction de l'agrégation des résultats, deux approches possibles:

- Micro-average

Calculer les mesures globales en considérant toutes les classes ensemble, comme s'il s'agissait d'un problème de classification binaire.

- Macro-average

Calculer les mesures pour chaque classe, puis prend la moyenne de ces mesures.

Exemple:

	Prédit : Classe 0	Prédit : Classe 1	Prédit : Classe 2
Classe 0	50	10	5
Classe 1	5	40	10
Classe 2	0	5	60

$$\text{Micro-Precision} = \frac{TP_{\text{total}}}{TP_{\text{total}} + FP_{\text{total}}} = \frac{150}{150 + 35} = \frac{150}{185} \approx 0.8108$$

Précision pour Classe 0 :

$$\text{Precision}_0 = \frac{TP_0}{TP_0 + FP_0} = \frac{50}{50 + 5 + 0} = \frac{50}{55} \approx 0.909$$

Précision pour Classe 1 :

$$\text{Precision}_1 = \frac{TP_1}{TP_1 + FP_1} = \frac{40}{40 + 10 + 5} = \frac{40}{55} \approx 0.727$$

Précision pour Classe 2 :

$$\text{Precision}_2 = \frac{TP_2}{TP_2 + FP_2} = \frac{60}{60 + 5 + 10} = \frac{60}{75} \approx 0.800$$

$$\text{Macro-Precision} = \frac{\text{Precision}_0 + \text{Precision}_1 + \text{Precision}_2}{3} = \frac{0.909 + 0.727 + 0.800}{3} = \frac{2.436}{3} \approx 0.812$$

Travaux pratiques

TP : Classification des tumeurs mammaires avec le dataset Breast Cancer

- **Objectif**
- L'objectif de ce TP est de vous familiariser avec la classification binaire en utilisant le **jeu de données de cancer du sein (Breast Cancer dataset)** disponible dans **scikit-learn**.
- Entraînez un modèle de classification pour prédire si une tumeur mammaire est **bénigne** ou **maligne** en fonction de plusieurs (30) caractéristiques mesurées.

```python
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
import pandas as pd

# Charger le jeu de données de cancer du sein
data = load_breast_cancer()

# Convertir en DataFrame pour une meilleure lisibilité
df = pd.DataFrame(data=data.data, columns=data.feature_names)
df['target'] = data.target

# Afficher les premières lignes du DataFrame
print("Premier aperçu des données :")
print(df.head())
```

Exploration des données

```python
# Séparer les données en caractéristiques (X) et labels (y)
X = data.data  # Caractéristiques
y = data.target  # Labels (0: bénin, 1: malin)

# Séparation en ensemble d'entraînement et ensemble de test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialisation du modèle (Random Forest dans cet exemple)
model = RandomForestClassifier(n_estimators=100, random_state=42)

# Entraînement du modèle
model.fit(X_train, y_train)

# Prédictions sur l'ensemble de test
y_pred = model.predict(X_test)

# Calcul de la précision et rapport de classification
accuracy = accuracy_score(y_test, y_pred)
print(f"Précision du modèle : {accuracy:.2f}")
print("\nRapport de classification :\n", classification_report(y_test, y_pred))
```

Préparation des données

Entraînement du modèle

Évaluation du modèle