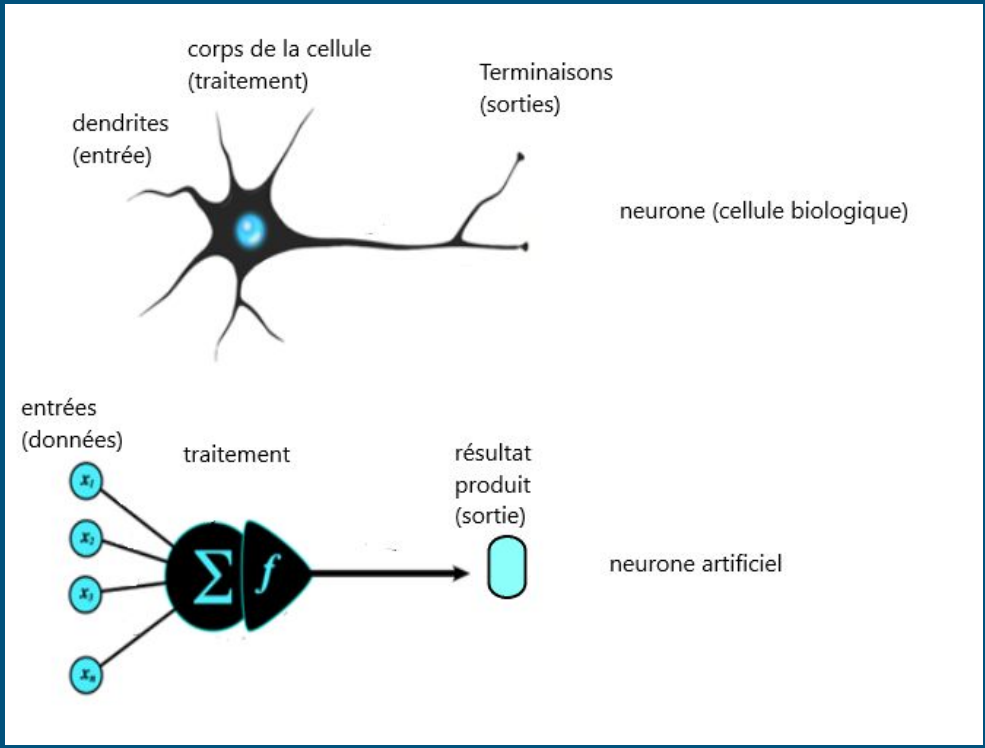
Chapitre I

Réseaux de neurones & Deep Learning

Réseaux de neurones & Deep Learning

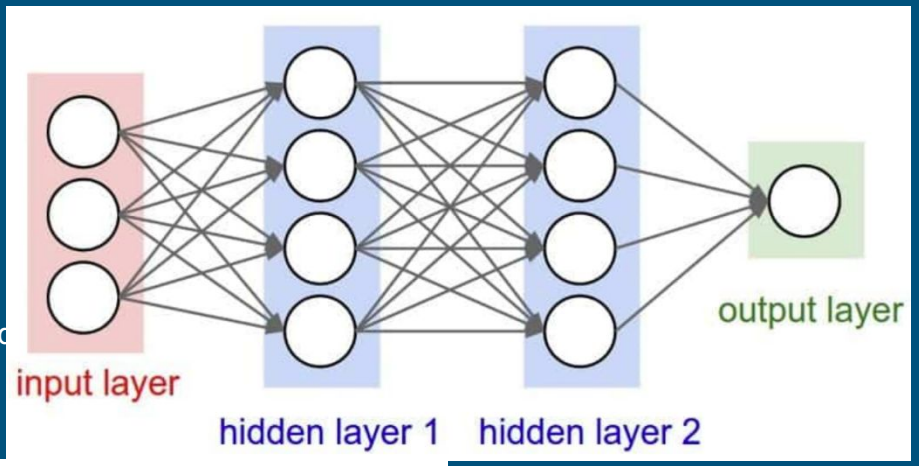
RNA ou ANN (Artificial Neural Network)

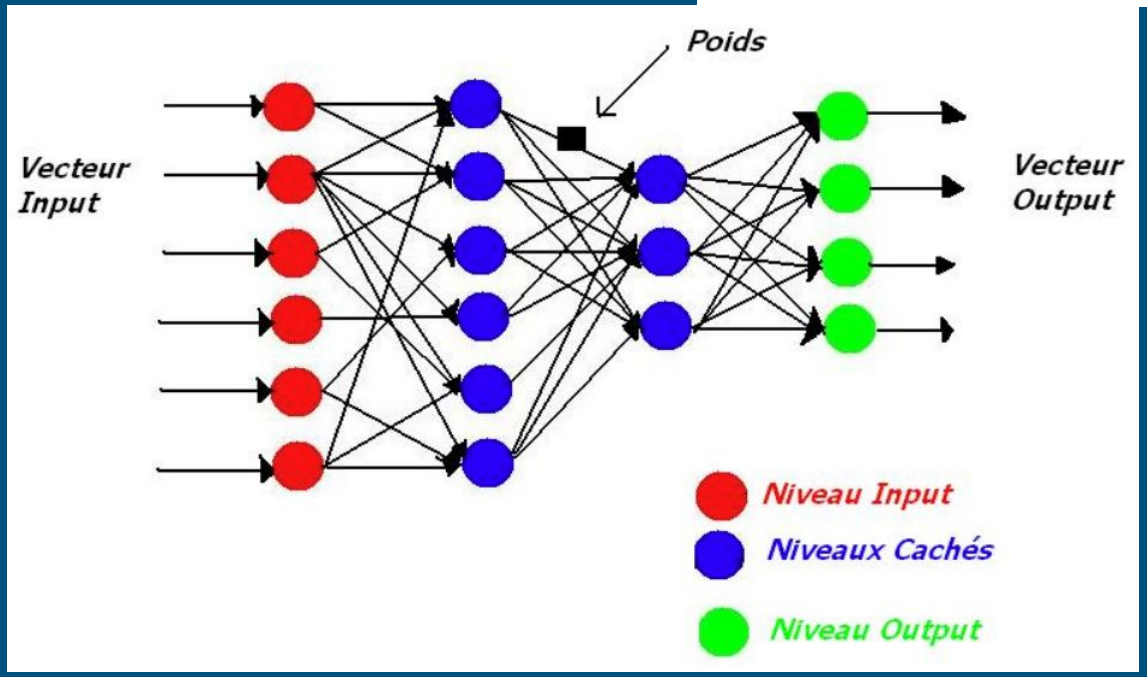
- un modèle de calcul
- conception schématiquement inspirée du fonctionnement des neurones biologiques.
- Il permet de réaliser des traitements d'apprentissage automatique.
- Grande vitesse de traitement



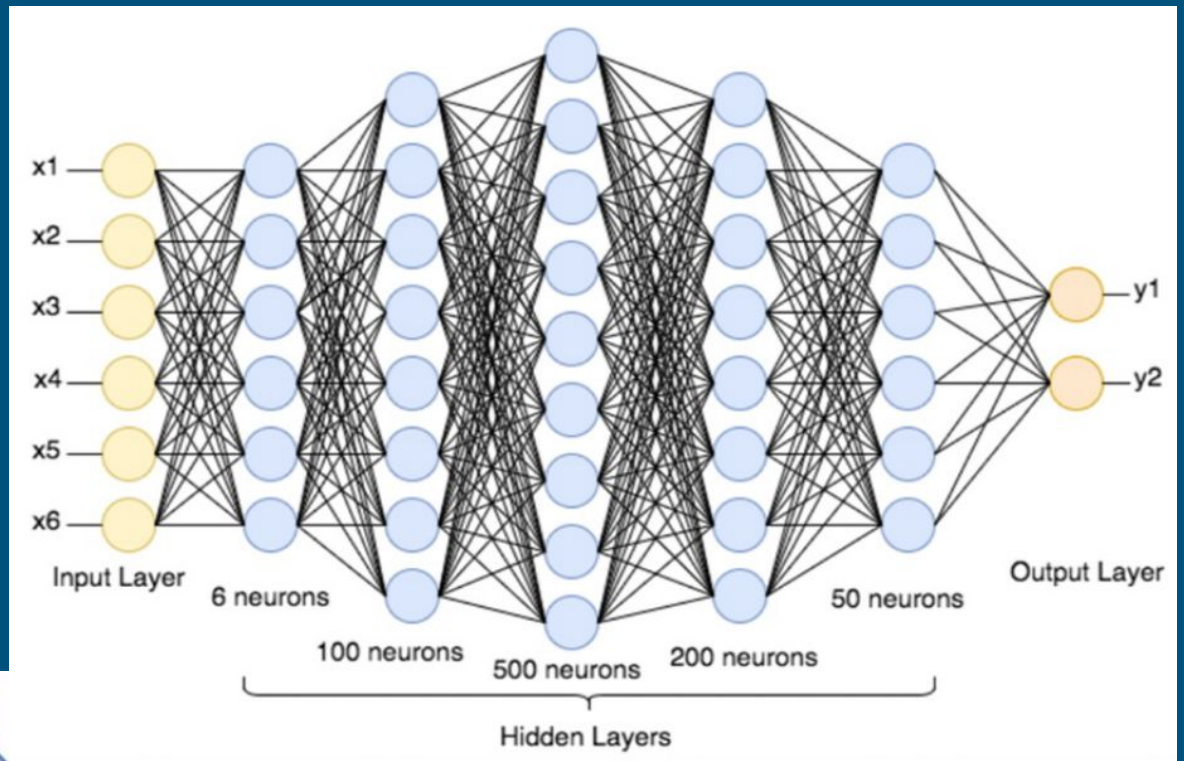
Architecture des RNA

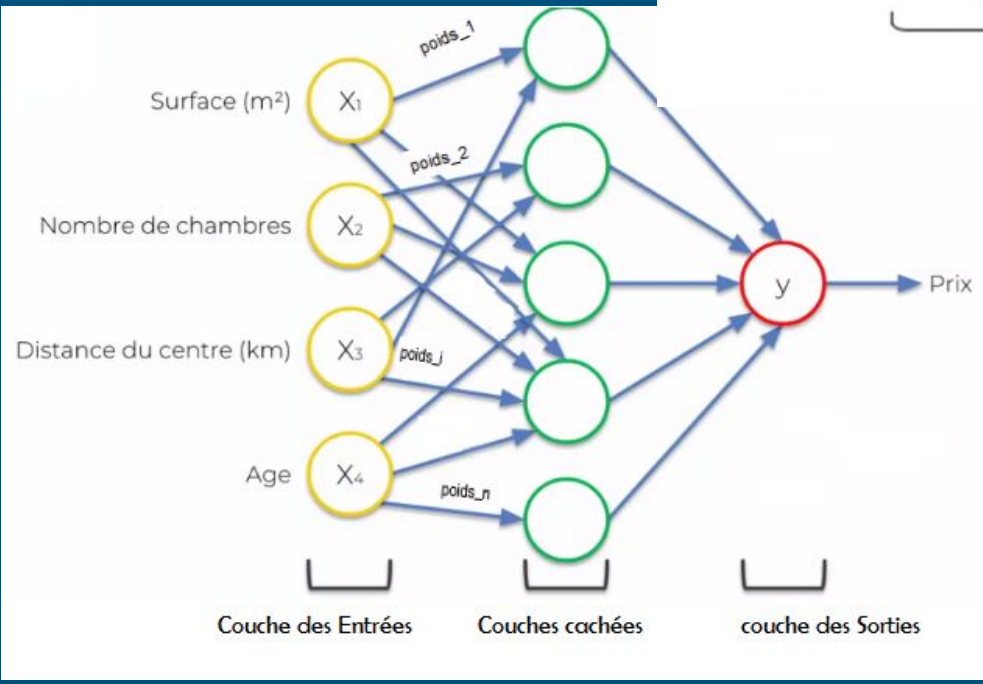
- Des blocs opérant en parallèle et organisés en couches
- Couche d'entrée : valeurs des attributs
- Chaque entrée à une couche est pondérée par un poids
- Couche sortie : fournit, grâce à une fonction particulière, le résultat du modèle
- 1 ou plusieurs couches intermédiaires.
- La couche intermédiaire reçoit en entrée les données en sortie de la couche précédente, qui ont subi des transformations par l'application d fonctions mathématiques appelées fonctions d'activation.
- La taille de la couche sortie dépend du résultat objectif

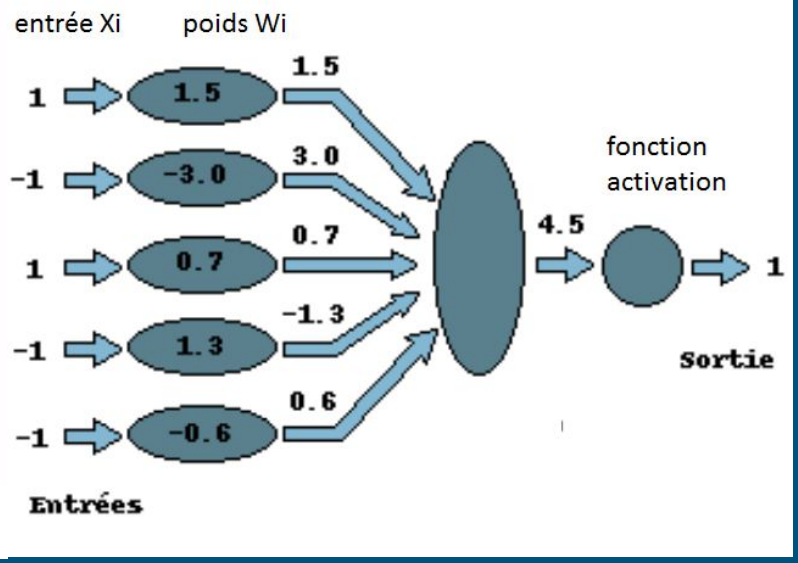




Exemples de RNA

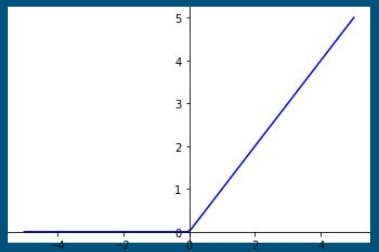




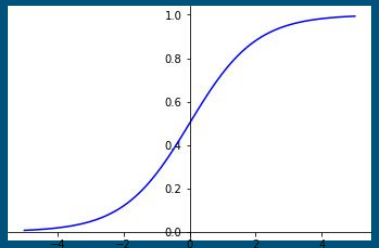


Exemples de Fonctions d'activation

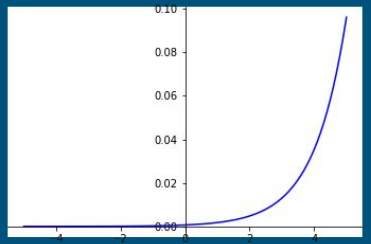
Rôle	Fonction	Calcul	valeur
Inter-couche	ReLU	F(x)= max(x, 0)	0 ou x
Classification Binaire Regression logistique	Sigmoid		[0, 1]
Multi-classes	Softmax		
Classification binaire à valeurs négatives	Tanh		[-1, 1]



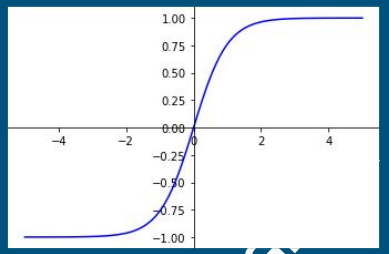
Fonction ReLU



Fonction Sigmoïde



Fonction Softmax

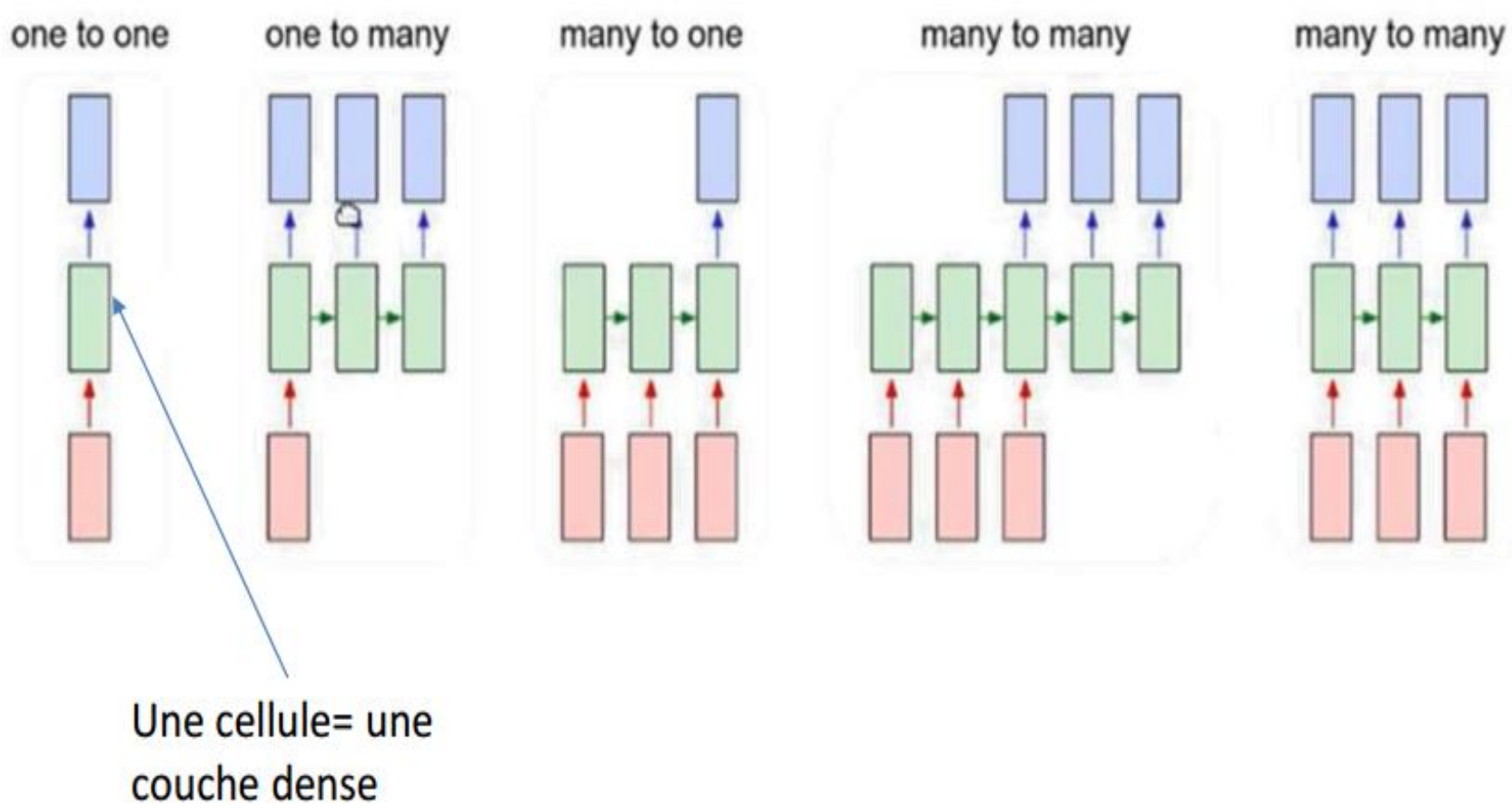


Fonction Tanh

Principe de L'apprentissage dans un RNA

- Propagation avant (feedforward) Pas de cycle dans le RNA
- A chaque itération une observation différente est fournit au RNA
- Initialisation aléatoire des poids
- La modification des poids se fait grâce à une règle d'apprentissage.
- La règle d'apprentissage tend à minimiser la fonction Coût (minimum, gradient, dérivée …) par des itérations successives(epochs)
- Consiste à ajuster progressivement les poids synaptiques wij jusqu'à ce que les réponses soient conformes à ce que l'on attend sur la base d'apprentissage.

Réseaux Récurrents
RNN (feedback network)



Lexique Apprentissage

(Hyperparamètres / paramètres)

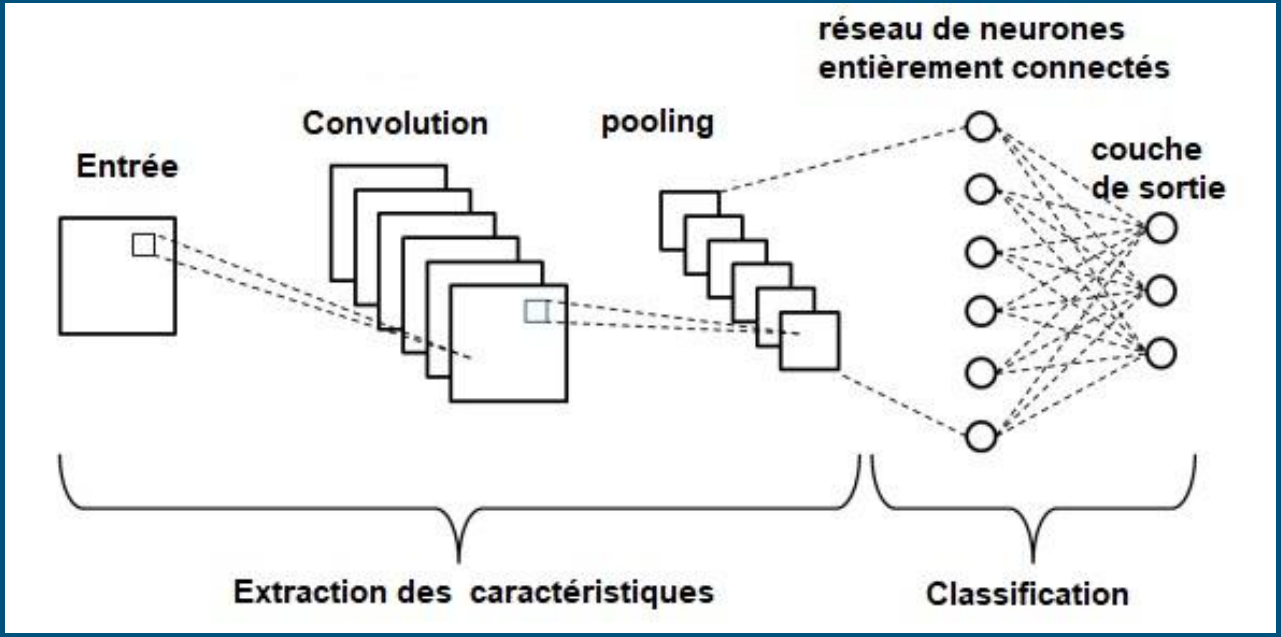
Terme	Signification	Exemples
batch	Nombre d'observations soumis en 1 fois	Généralement 32
epoch	Une itération complète tel que toutes les entrées ont été introduites	Au min 20
optimizer	Algorithme de calcul pour l'ajustement des poids pendant l'apprentissage	SGD, momentum, adagrad, RMSprop, Adam
learning-rate	Facteur multiplicatif aide à la convergence	0,001 ,,,
Loss function	Fonction globale du modèle	Categorical_crossentropy, MSE (mean square error)

Deep Learning
et ses applications

Domaine	Exemple
Vision par ordinateur	détection d'objets, la segmentation d'images, classification, génération d'images surveillance vidéo, diagnostic médical, robotique
Traitement langage naturel	traduction automatique, reconnaissance vocale, génération de texte, traitement du langage naturel (analyse de sentiments, chatbots, résumés automatiques, etc.
Jeux	StarCraft, AlphaStar, jeu d'échecs
Finance et éconimie	prévision des marchés financiers, analyse de données financières, détection de fraude, évaluation des risques,
Robotique	contrôle des robots, manipulation d'objets, navigation autonome, etc.

CNN : Convolutional Neural Network

- Très performant pour la classification d'images
- Comporte 2 parties :
  - Des couches qui permettent d'extraire les caractéristiques des images (couches de convolution et de pooling)
  - Des couches de classification (fully connected neurones)



Implémentation d’un CNN

Construisons un modèle CNN avec Tensorflow

Objectif

Créer un modèle CNN pour classifier des images.

Dataset utilisée

MNSIT Fashion de keras contient plus de 70000 images en niveau de gris

Librairies

numpy , tensorflow , matplotlib, tensorflow.keras, pandas, sklearn, seaborn

Description des données

Dimension : 28x28,

Niveau de Gris

10 étiquettes :

0 – T-shirt/haut

1 – Pantalon

2 – Pullover

3 – Robe

4 – Manteau

5 – Sandale

6 – Chemise

Etapes de Conception d’un modèle de classification CNN

1- Import des librairies

2- Import / ouverture dataset

3- préparation des données (normalisation, redimensionnement,…)

4- Diviser les données en 2 ensembles : Train et Test

5- Création des couches du modèle

6- Choix des paramètres, hyperparamètres et compilation du modèle

7- phase d’apprentissage

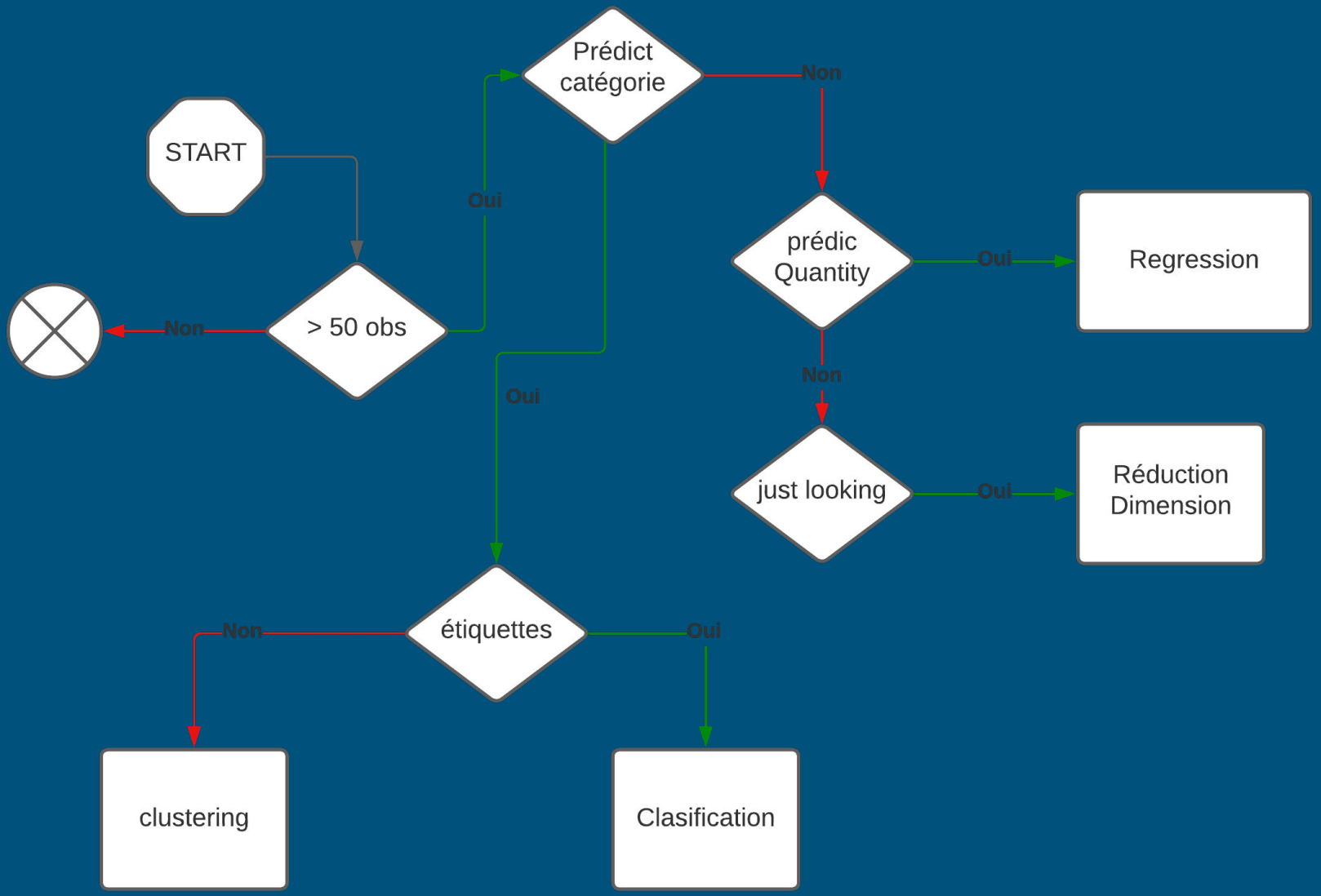
8- Evaluation des performances du modèle

9- Appliquer le modèle sur un jeu de données

explication instruction python	
fonction sur les tableaux	import numpy as np
import du framework tensorflow	import tensorflow as tf
afficher graphiques, courbes,…	import matplotlib.pyplot as plt
import des couches et différents éléments d'un modèle	from tensorflow.keras.layers import Dense, Conv2D, Input, Flatten, Dropout, MaxPooling2D
import composante model	from tensorflow.keras.models import Model
opérations sur les dataset…	import pandas as pd
éléments évaluation modèle	from sklearn.metrics import classification_report,confusion_matrix
arrêt de l'app si convergence	from tensorflow.keras.callbacks import EarlyStopping
Affichage graphique élaborés	import seaborn as sns
import du dataset mnist fashion	dataset_fashion_mnsit = tf.keras.datasets.fashion_mnist
séparation jeu train/test	(X_train, y_train), (X_test, y_test)=dataset_fashion_mnsit.load_data()
compter le nombre d'étiquettes	pd.DataFrame(y_train)[0].value_counts()
normalisation des tensor [0,1]	X_train = X_train / 255
	X_test = X_test / 255
afficher dimension tensor	print(f"Données entrainement: {X_train.shape}, Test: {X_test.shape}")
afficher la première image	plt.imshow(X_train[0])
et son étiquette	y_train[0]
ajouter 1 dimension pour couleur	X_train = X_train.reshape(60000, 28, 28, 1)
	X_test = X_test.reshape(10000, 28, 28, 1)
"""création du modèle"""	mon_cnn = tf.keras.Sequential()

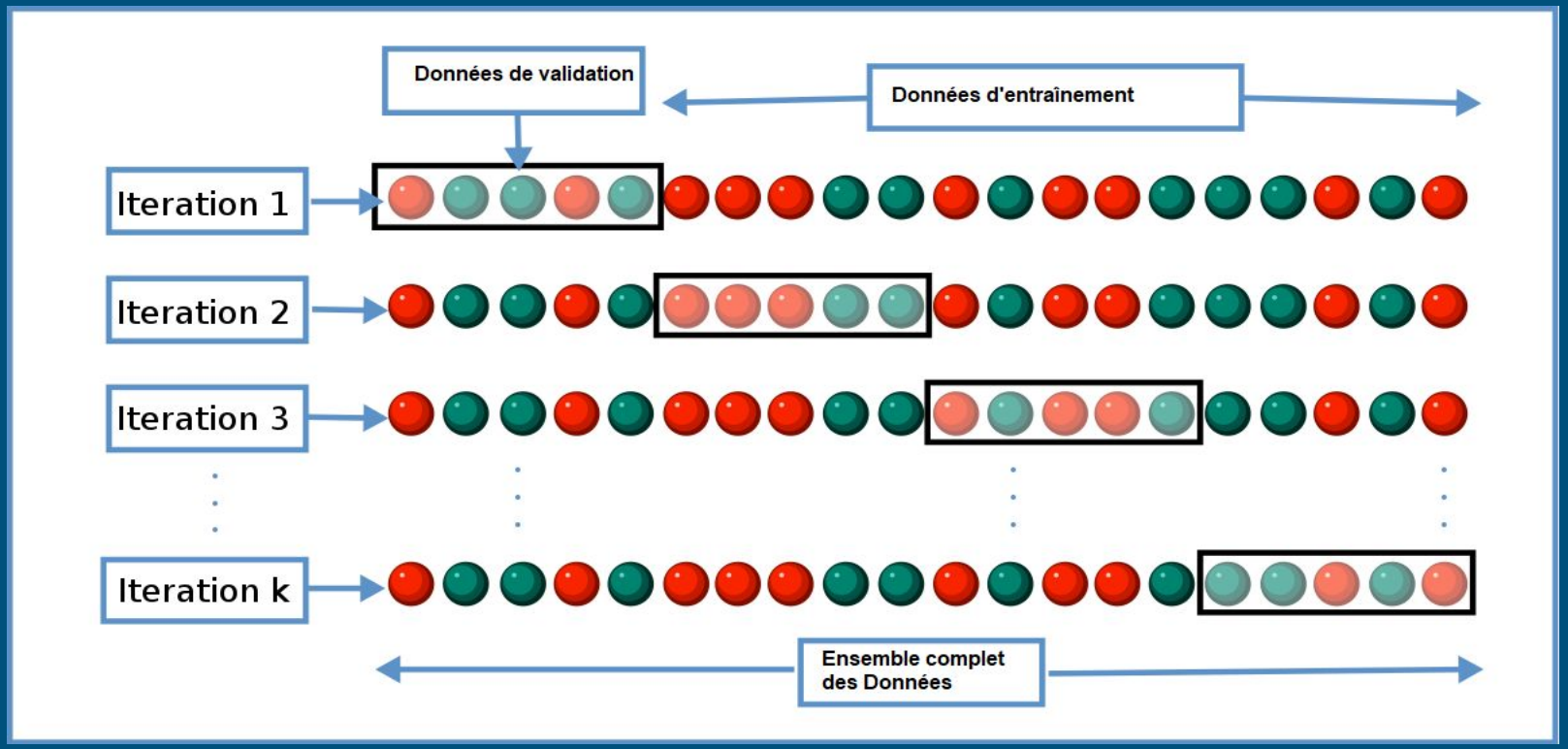
explication	instruction python
3 couches de convolution, avec Nb filtres progressif 32, 64 puis 128	mon_cnn.add(Conv2D(filters=32, kernel_size=(3,3), input_shape=(28, 28, 1), activation='relu')) mon_cnn.add(MaxPooling2D(pool_size=(2, 2))) mon_cnn.add(Conv2D(filters=64, kernel_size=(3,3), input_shape=(28, 28, 1), activation='relu')) mon_cnn.add(MaxPooling2D(pool_size=(2, 2))) mon_cnn.add(Conv2D(filters=64, kernel_size=(3,3), input_shape=(28, 28, 1), activation='relu'))
remise à plat Couche dense classique ANN Couche de sortie (classes de 0 à 9) meilleur nombres de epochs compiler le model	mon_cnn.add(MaxPooling2D(pool_size=(2, 2))) mon_cnn.add(Flatten()) mon_cnn.add(Dense(512, activation='relu')) mon_cnn.add(Dense(10, activation='softmax')) early_stop = EarlyStopping(monitor='val_loss',patience=2) mon_cnn.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
afficher résumé model """train model""«	mon_cnn.summary()
Evaluation du modèle Affichage courbe accuracy et Loss	mon_cnn.fit(x=X_train, y=y_train, validation_data=(X_test, y_test), epochs=25,callbacks=[early_stop]) losses = pd.DataFrame(mon_cnn.history.history) losses[['accuracy', 'val_accuracy']].plot() losses[['loss', 'val_loss']].plot()
Prédiction^pour X_test """Matrice de Confusion graphique""«	pred=np.argmax(mon_cnn.predict(X_test) , axis=-1)
Matrice confusion textuelle	plt.figure(figsize=(6,4)) sns.heatmap(confusion_matrix(y_test, pred),annot=True)
Classification Report	cm=confusion_matrix(y_test, pred) print(cm)

Feuille de route ML



Techniques d'amélioration des performances du modèle

- Réglage des hyperparamètres du modèle (gridsearch)
- Validation croisée pour l'évaluation du modèle (K-fold)
- Techniques de régularisation pour éviter overfitting (earlystop)

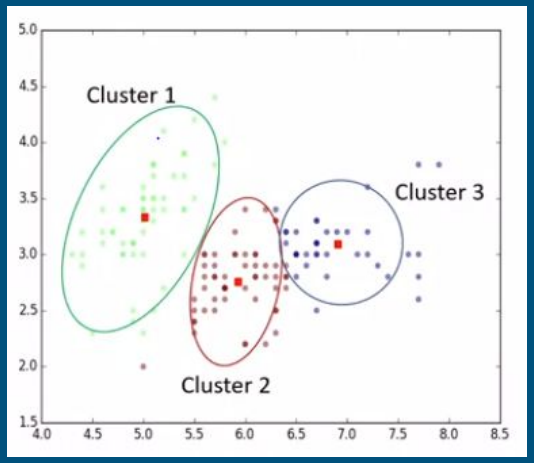


Apprentissage non supervisé

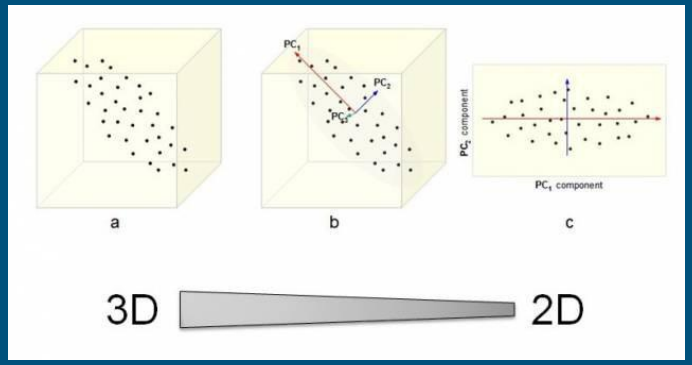
Pas d'étiquettes pour les observations apprentissage non supervisé

Rôle du modèle : analyser et comprendre la structure des données

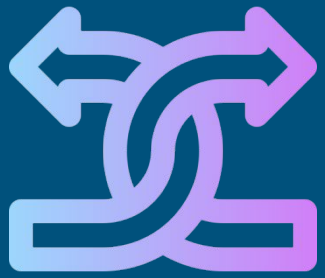
Clustering : former des clases et regrouper les observations



Réduction de dimensionnalité : réduire nombre d'attributs



Règles d'association : trouver des liaisons entre les observations et/ou les attributs



Clustering

Basée sur 2 notions :

- Calcul de distances
- Regroupement selon un critère de similarité ou désimilarité

Type regroupement	Algorithme de clustering
Catégorisation par partitionnement	• K-means et ses variantes • Partitionnement Hiérarchique Descendant • Partitionnement spectral
Catégorisation par agglomération	Regroupement Hiérarchique Ascendant
Catégorisation par modélisation	• Mélange de gaussiennes (GMM) • Cartes de Kohonen (Self-Organizing Maps, SOM)
Catégorisation basé sur la densité	DBSCAN

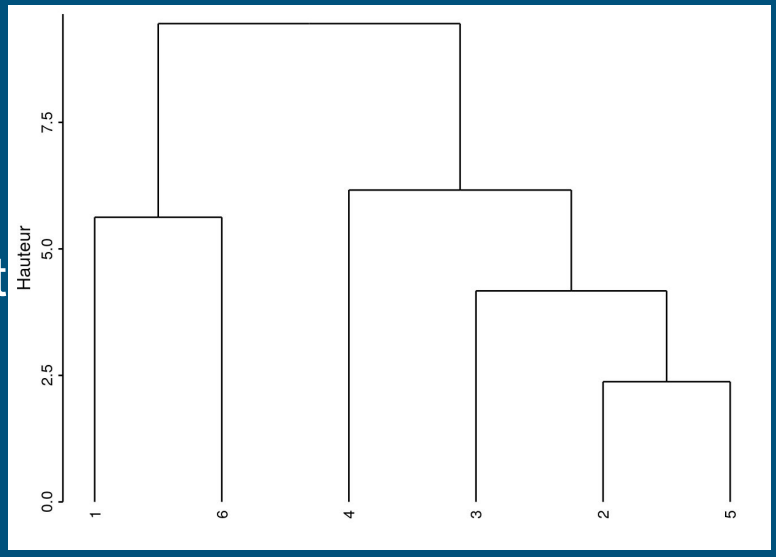
K-means, K-means++, K-médoides, Fuzzy C-means

Répartir chacun des N individus dans une certaine classe i en minimisant la variance intra-classe données par une fonction objectif

- Somme distances entre centre d'une classe et chaque membre

Hiérarchique ascendante CAH

On démarre avec N classes
Regrouper au fur et à mesure les classes 2 par 2 selon un indice d'agrégation Jusqu'à aboutir à une seule classe
Un diagramme (dendogramme) permet de déterminer le nombre de classes optimal



DBSCAN

DBSCAN : Density-Based Spatial Clustering of Applications with Noise.
– Catégorisation basée sur la densité
– Rechercher les zones se caractérisant par une grande
densité de données.
– Separation des classes de haute densité de ceux ayant une faible densité.
– Se base sur deux paramètres "eps" et "MinPts"

Activité 5 : clustering K-means

On désire réaliser une catégorisation des individus de la base "pima-indians-diabetes.csv" par l'algorithme k-means.

1- Préparation des données

Pour pouvoir afficher les individus dans un plan (2D), avec les classes d'appartenance on fera les restrictions suivantes :

Pour la description des individus, on se restreindra à deux attributs.

Pour le nombre des individus, on se restreindra à 100 individus.

a- Construire le nouvel ensemble de données (100 individus et 2 attributs).

b- Effectuer la normalisation des donnés afin que valeurs des deux attributs soient dans un intervalle [0-1].

c- Afficher dans un graphe les individus tel que le premier attribut représente les abscisses et le deuxième attribut représente les ordonnés.

2- Catégorisation : Effectuer la catégorisation par la méthode des k-moyennes puis afficher le résultat de la catégorisation pour des valeurs de k =2, 3 et 4.

3- Conclure

Activité 6 : clustering CAH

Effectuer la catégorisation automatique d'un ensemble de fromages (CAH).

1- Importer les données à partir du fichier "fromage.txt".

2- Afficher la dimension de l'ensemble de données et en déduire le nombre des variables utilisées.

3- Afficher les noms de ces variables.

4- Déterminer l'intervalle de variation de chacune des variables.

5- Changer l'échelle des données pour les ramener dans intervalle [0-1].

6- Afficher le dendrogramme correspond

a- Utiliser la distance euclidienne et tester plusieurs critères d'agrégation : "indice de Ward", "le lien minimum", "le lien maximum" et "le lien moyen".

b- Analyser les dendrogrammes obtenus en identifiant, pour chacun des critères, le meilleur choix du nombre de classes k.

7- Effectuer la catégorisation avec CAH et visualiser les résultats obtenus et ce en utilisant : Variables : 1er et 2ème attributs, Distance : euclidienne,

agrégation : "l'indice de Ward", "le lien maximum" et "le lien moyen".

Réduction de dimensionnalité

C'est Quoi?

• Réduire le nombre de variables en conservant au mieux leur structuration initiale et leur organisation globale.

• Permet d'améliorer la lisibilité des données.

Comment ?

• Choisir un sous ensemble de variables parmi les variables de départ

• Créer des nouvelles variables combinaisons linéaires des variables de départ

Méthodes

- L'analyse en composantes principales (**ACP**)
  - Adaptée à des observations décrites par des variables quantitatives.
  - Trois variantes : ACP générale, ACP centrée, ACP normée
- L'analyse des correspondances binaires (**AFCB**) ou multiples (**ACM**)
  - Adaptée à des observations décrites par des variables nominales (à modalités).
- L'analyse factorielle discriminante (**AFD**),
  - Adaptée à des observations décrites par des variables quantitatives et appartenant à plusieurs classes.

ACP : Analyse en Composantes Principales

C’est Quoi ?

L’ACP permet de réduire les dimensions d’une donnée multivariée à deux ou trois composantes principales, qui peuvent être visualisées graphiquement (2D ou 3D), en perdant le moins possible d’information.

Comment ?

1. Normalisation des variables
    - Centrer et réduire les données.
2. Trouver les nouvelles variables
    - Calculer les composantes principales
3. Décrire les données d’origine en utilisant les nouvelles variables

ACP avec

```
from sklearn import decomposition

pca = decompo sition.PCA()

pca.fit(X_CR)  #déterminer les nouveaux axes,

#   X_CR are inputs after being centered and scaled

#afficher les inerties expliquées pour les différents nouveaux axes
print(pca.explained_variance_ratio_)
```

- Un vecteur qui contient les inerties

Activité 7 ACP

Soit la base "IRIS" du package sklearn.datasets. La base contient des fleurs de type Iris appartenant à trois classes différentes (Setosa, Versicolour et Virginica).

1- Importer les données à partir de la base "IRIS".

2- Afficher la dimension de l'ensemble de données et en déduire le nombre des variables utilisées.

3- Afficher les noms de ces variables.

4- Centrer et réduire les données : normaliser les valeurs des variables pour qu'elles suivent une loi normale de moyenne 0 et de variance 1.

5- Effectuer l'analyse en composantes principales (ACP).

6- Projeter les données sur les nouveaux axes.

7- Afficher les inerties expliquées. En déduire le nombre approprié de composantes principales qu'on doit considérer.

8- Afficher les données en considérant 3 composantes principales, puis 2 composantes principales. Conclure.

Règles d’association

– Utiliser dans le cadre de la fouille de données (data Mining)
– Permet de dégager les implications conditionnelles entre un ensemble de variables appelées items
– Permet de découvrir des relations, corrélations, ou structures causales entre deux ou plusieurs variables à partir d’un entrepôt de données (BD, fichier, …)

Exemples d’application

– L’analyse des données de ventes dans un supermarché
• Une règle découverte pourrait indiquer qu'un client achetant des oignons et des pommes de terre simultanément, serait susceptible d'acheter un hamburger.
• Une telle information peut être utilisée pour prendre des décisions marketing
 des promotions ou des emplacements bien choisis pour les produits associés

– L’analyse des logs web sur un serveur web
• Permet de découvrir de comportements utilisateur (web usage mining) dans le but d’adapter ou de personnaliser le site ou de découvrir des comportements types sur certains sites (E-commerce par exemple)

Processus de découverte des règles d'associations

2 étapes :

1. L'extraction des ensembles fréquents d'items (Itemsets fréquents)
   = Ceux qui ont un support minimum (supérieur à un seuil)
2. La génération des règles d'associations à partir de ces ensembles
   Sous la forme : Antécédent Conséquent

Définitions

– Items : Un domaine d'application donné doit être décrit par une liste limitée d'atomes (items).

Par exemple, pour l'application du panier de ménagère la liste des items correspond à l'ensemble d'articles disponibles dans le supermarché [pain; fromage; chocolat; …].

– Itemset : Un ensemble d'items : c'est une succession d'items.

Règles d'association

1. Extraction des Itemsets fréquents

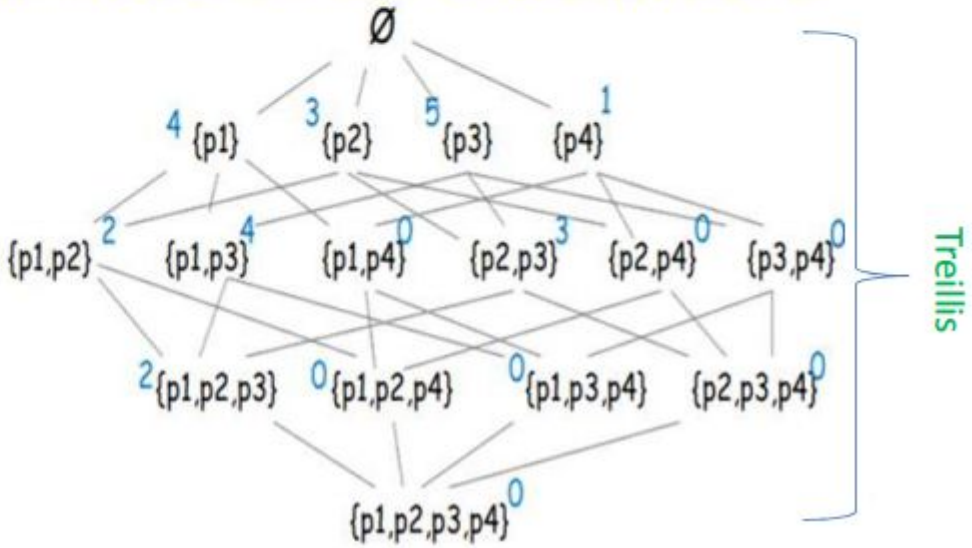
– 1ère méthode : lister toutes les configurations de Itemsets possibles

- Très couteux en complexité: Il s'agit de construire un treillis

➔ Essayer de minimiser le nombre de configurations.

Données

Caddie	p1	p2	p3	p4
1	1	1	1	0
2	1	0	1	0
3	1	1	1	0
4	1	0	1	0
5	0	1	1	0
6	0	0	0	1



Règles d'association

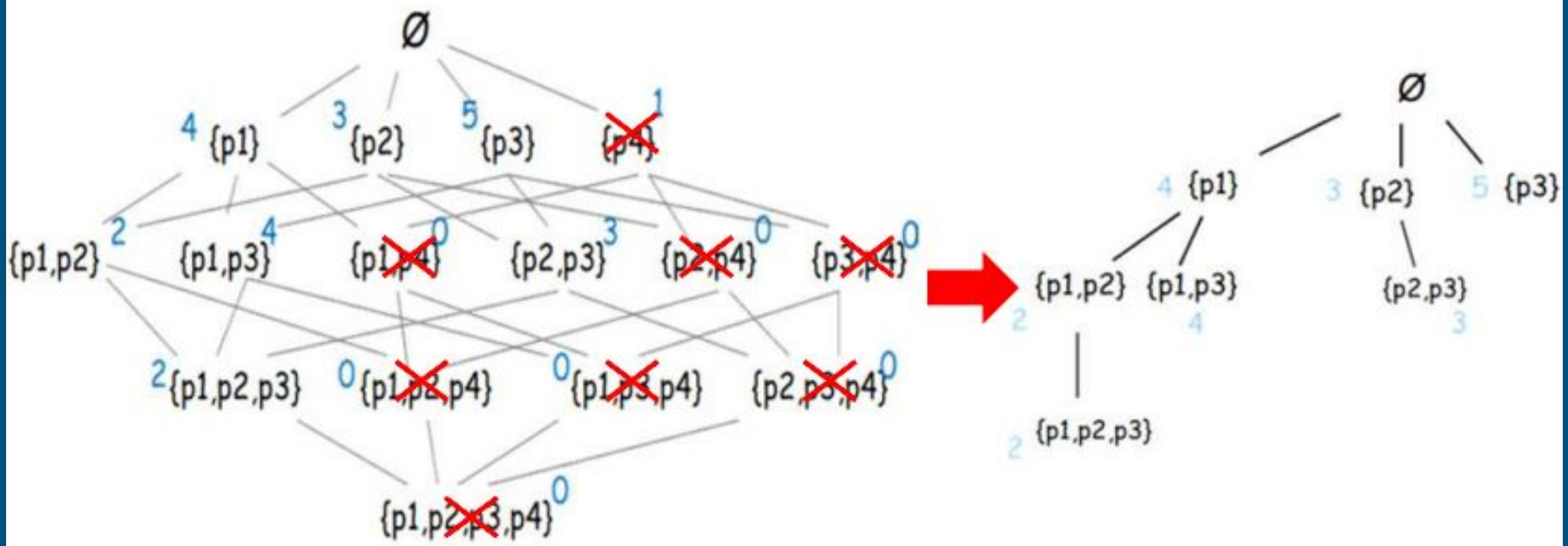
1. Extraction des Itemsets fréquents
   - Trouver seulement les Itemsets fréquents
     - Itemset fréquent : itemset dont le support est supérieur à support minimum (définit par l'utilisateur selon le contexte)
     - Utiliser la règle suivante : Un sous-ensemble d'un ensemble fréquent est fréquent
       - Exemple
         - » si $\{P_1,P_2\}$ est fréquent alors $\{P_1\}$ et $\{P_2\}$ le sont aussi

     ➔Si un itemset n'a pas un support minimum (n'est pas fréquent), alors aucun itemset l'incluant n'est fréquent.
       - Exemple
         - » Si $\{P_1\}$ n'est pas fréquent alors $\{P_1P_2\}$ ne peut pas l'être

Règles d'association

1. Extraction des Itemsets fréquents

- Exemple : Support minimum=2



Règles d'association

2. Génération des règles d'association

- Extraire toutes les règles de grande confiance à partir des ensembles d'items fréquents trouvés dans l'étape précédente.
- Ne choisir que les règles ayant une confiance supérieur à une confiance minimale
- Il faut tester toutes les combinaisons possibles

Fin

Merci Pour Votre Attention