

Machine Learning

Apprentissage supervisé

Stéphanie Bricq
stephanie.bricq@u-bourgogne.fr

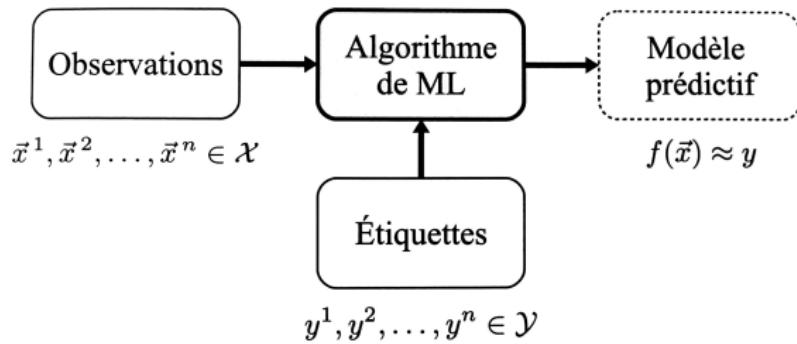
Université de Bourgogne

09/2022

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

Apprentissage supervisé



- But : apprendre à faire des prédictions à partir d'une liste d'exemples étiquetés

Classification binaire

- cas des étiquettes binaires \Rightarrow indiquent l'appartenance à une classe
 - \Rightarrow classification binaire
- Exemples
 - identifier si un email est un spam ou non
 - identifier si une image contient une voiture ou non
 - identifier si une transaction financière est frauduleuse

Classification multi-classe

- cas des étiquettes discrètes \Rightarrow correspondent à plusieurs classes
 \Rightarrow classification multi-classe
- Exemples
 - identifier à quelle espèce appartient une plante
 - identifier les objets présents sur une photographie
 - identifier en quelle langue un texte est écrit
 - identifier un chiffre manuscrit
 - identifier l'expression d'un visage parmi une liste de possibilités (colère, tristesse, joie,...)

Régression

- cas des étiquettes à valeurs réelles
- ⇒ régression
- Exemples
 - prédire le nombre de clics sur un lien
 - prédire le prix d'une action en bourse
 - prédire le rendement d'un plan de maïs

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

Régression

- spécifier la relation entre une variable numérique endogène (valeur à prédire)
- et une ou plusieurs variables numériques exogènes (variables explicatives)
- forme la + simple : relation linéaire

$$y = \alpha + \beta x$$

Régression

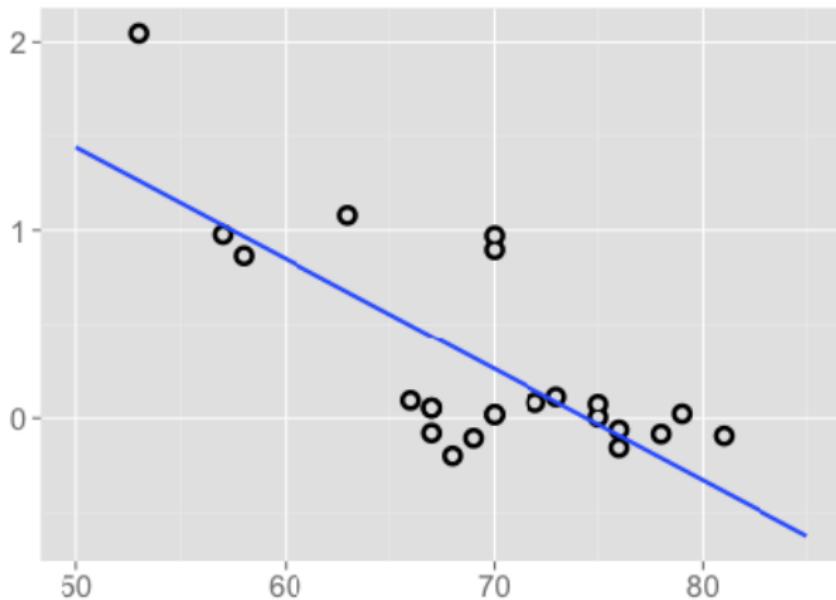
- utilisé pour de nombreuses tâches
 - pour modéliser des relations complexes parmi des données
 - pour estimer l'impact d'un traitement sur un résultat
 - exemples
 - examiner les variations au sein de populations à partir de caractéristiques mesurées pour différents champs de recherche (économie, sociologie, psychologie, physique, ...)
 - quantifier la relation causale entre un évènement et la réponse : essais cliniques, recherche marketing, ...
 - identifier les modèles pouvant être utilisés pour prédire un comportement en connaissant certains critères : dommages de catastrophes naturelles, résultats d'élections, ...

Régression

- Modèles basiques de régression : **régression linéaire**
 - si une seule variable explicative : régression linéaire simple
 - sinon : régression multiple
- régression peut également être utilisée pour des tâches de classification : régression logistique

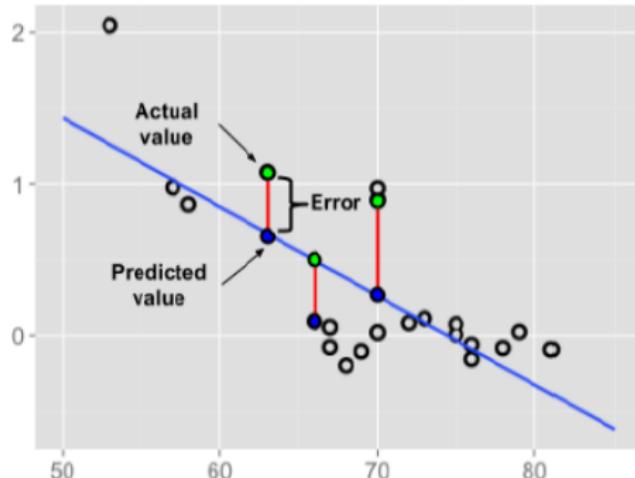
Régression linéaire simple

$$y = \alpha + \beta x$$



Méthode des moindres carrés ordinaires

- estimation de α et β
- méthode : Ordinary Least Squares estimation (OLS)



- régression : minimisation de l'équation suivante

$$\sum (y_i - \hat{y}_i)^2 = \sum e_i^2$$

Méthode des moindres carrés ordinaires

- solution pour α dépend de la valeur de β :

$$\alpha = \bar{y} - \beta \bar{x}$$

- valeur de β minimisant l'erreur quadratique :

$$\beta = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

$$\beta = \frac{Cov(x, y)}{Var(x)}$$

avec

- variance $Var(x) = \frac{\sum(x_i - \bar{x})^2}{n}$
- covariance $Cov(x) = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{n}$

Corrélation linéaire

- coefficient de corrélation de Pearson

$$\rho_{x,y} = \text{Corr}(x, y) = \frac{\text{Cov}(x, y)}{\sigma_x \sigma_y}$$

avec σ_x écart-type de x

- renseigne sur le degré de dépendance linéaire entre les deux variables
- compris entre -1 et 1
 - Plus la valeur absolue du coefficient est importante, plus la relation linéaire entre les variables est forte
 - signe du coefficient indique la direction de la relation.
 - Si les deux variables ont tendance à augmenter ou à diminuer ensemble, le coefficient est positif
 - Si une variable a tendance à augmenter lorsque l'autre diminue, le coefficient est négatif

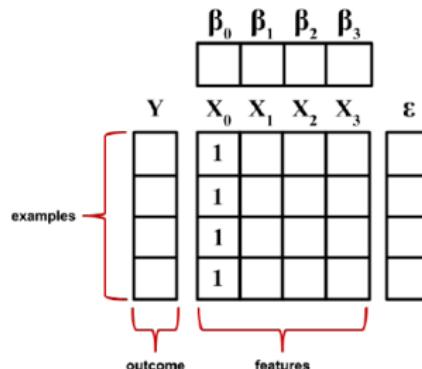
Régression linéaire multiple

Avantages	Inconvénients
Approche la plus commune pour modéliser des données numériques	Hypothèses fortes sur les données La forme du modèle doit être spécifiée à l'avance par l'utilisateur
Peut être adapté pour modéliser la plupart des tâches	Ne traite pas les données manquantes Fonctionne seulement sur des données numériques, les données de type catégorie nécessitent des préparations supplémentaires

Régression linéaire multiple

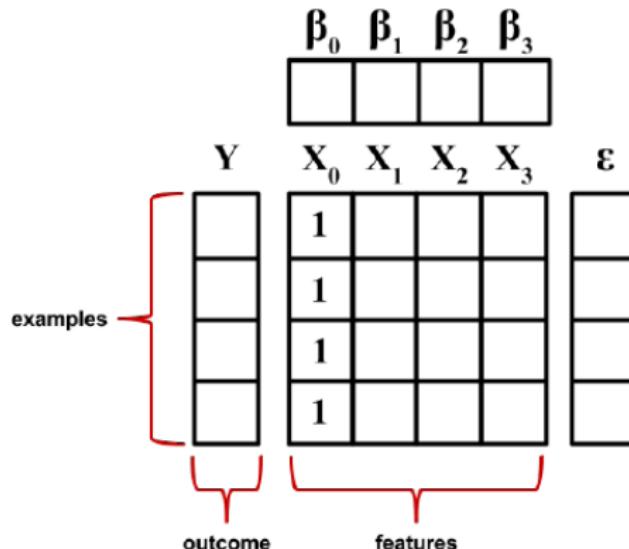
$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_i x_i + \epsilon$$

$$y = \beta_0 x_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_i x_i + \epsilon$$



$$Y = X\beta + \epsilon$$

Régression linéaire multiple : Moindres carrés



$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

Régression linéaire multiple : Descente de gradient

- méthode des moindres carrés :
 - contrainte de la réversibilité
 - peut être consommatrice en temps de calcul si le nb d'observations et le nb de variables sont très grands

Régression linéaire multiple : Descente de gradient

- méthode des moindres carrés :
 - contrainte de la réversibilité
 - peut être consommatrice en temps de calcul si le nb d'observations et le nb de variables sont très grands
- ⇒ méthode de la descente de gradient

- 1 Initialisation aléatoire du vecteur $\beta = (\beta_0, \beta_1, \dots, \beta_m)$
- 2 Pour chaque β_i , Faire

$$\beta_i = \beta_i + \alpha \frac{dE}{d\beta_i}$$

- 3 si pas de convergence, retour au niveau (2), sinon Fin
- E fonction d'erreur et α facteur d'apprentissage
 - + α est grand, + grand sera le pas du déplacement de l'algorithme, + rapide sera la convergence mais risque de rater l'optimum global

Exemple

- Considérons 5 observations en 2D, étiquetées par y . Quels sont les paramètres d'une régression linéaire sur ces données ?

x_1	0.12	0.14	0.31	0.37	0.49
x_2	29	33	17	21	12
y	21	24.3	12.7	15.6	9

Evaluation de la performance

- mesure de performance typique pour les problèmes de régression
 - RMSE (*Root Mean Square Error*)

$$RMSE(X, h) = \sqrt{\frac{1}{m} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2}$$

- MAE (*Mean Absolute Error*)

$$MAE(X, h) = \frac{1}{m} \sum_{i=1}^m |h(x^{(i)}) - y^{(i)}|$$

Régression polynomiale

- Généralisation de la régression linéaire multiple
- Modèle de régression polynomiale : polynôme à plusieurs variables
 - + permet de décrire des relations entre les variables qui ne sont pas linéaires
 - temps de calculs + importants
 - sujets au pb de surapprentissage

Régularisation

- Fonction d'erreur à minimiser

$$E = \sum_{i=1}^n [y_i - \hat{y}_i]^2$$

- problème de surapprentissage

Régularisation

- Fonction d'erreur à minimiser

$$E = \sum_{i=1}^n [y_i - \hat{y}_i]^2$$

- problème de surapprentissage
- ⇒ introduction du concept de régularisation
- Régularisation Lasso (ou L_1)

$$E_{Lasso} = \sum_{i=1}^n [y_i - \hat{y}_i]^2 + \lambda \sum_{i=0}^m |\beta_i|$$

- Régularisation Ridge (ou L_2)

$$E_{Ridge} = \sum_{i=1}^n [y_i - \hat{y}_i]^2 + \lambda \sum_{i=0}^m \beta_i^2$$

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

Régression logistique

- utilisée pour estimer la probabilité qu'une observation appartienne à une classe particulière
 - si probabilité estimée > 50%, alors le modèle prédit que l'observation appartient à cette classe
 - sinon il prédit qu'elle appartient à l'autre classe

Régression logistique

- utilisée pour estimer la probabilité qu'une observation appartienne à une classe particulière
 - si probabilité estimée > 50%, alors le modèle prédit que l'observation appartient à cette classe
 - sinon il prédit qu'elle appartient à l'autre classe

⇒ classifieur binaire

Régression logistique

- Probabilité estimée par le modèle de régression logistique

$$\hat{p} = h_{\Theta}(x) = \sigma(\Theta^T \cdot x)$$

Régression logistique

- Probabilité estimée par le modèle de régression logistique

$$\hat{p} = h_{\Theta}(x) = \sigma(\Theta^T \cdot x)$$

- Fonction logistique

$$\sigma(t) = \frac{1}{1 + \exp(-t)}$$

Régression logistique

- Probabilité estimée par le modèle de régression logistique

$$\hat{p} = h_{\Theta}(x) = \sigma(\Theta^T \cdot x)$$

- Fonction logistique

$$\sigma(t) = \frac{1}{1 + \exp(-t)}$$

- Prédiction du modèle de régression logistique

$$\hat{y} = \begin{cases} 0 & \text{si } \hat{p} < 0.5 \\ 1 & \text{si } \hat{p} \geq 0.5 \end{cases}$$

Régression logistique

- Fonction de coût pour une seule observation d'entraînement

$$c(\theta) \begin{cases} -\log(\hat{p}) & \text{si } y = 1 \\ -\log(1 - \hat{p}) & \text{si } y = 0 \end{cases}$$

- Fonction de coût de la régression logistique (perte logistique, *log loss*)

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log(\hat{p}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{p}^{(i)})]$$

- Pas de solution analytique connue pour calculer la valeur de θ qui minimise cette fonction de coût
- ⇒ algorithme de descente de gradient pour trouver le minimum global

Régression logistique

Avantages	Inconvénients
<ul style="list-style-type: none">- classification d'une nouvelle observation très rapide- algo simple, faible risque de sur-apprentissage	<ul style="list-style-type: none">- hypothèse de linéarité du score ⇒ empêche de tenir compte des interactions entre variables- phase d'apprentissage peut être longue

Régression softmax

- généralisation du modèle de régression logistique pour prendre en compte plusieurs classes directement \Rightarrow Régression softmax
- Principe :
 - observation x
 - calcul d'un score $s_k(x)$ pour chaque classe k

$$s_k(x) = (\theta^{(k)})^T \cdot x$$

- estimation de la probabilité de chaque classe en appliquant la fonction softmax

$$\hat{p}_k = \sigma(s(x))_k = \frac{\exp(s_k(x))}{\sum_{j=1}^K \exp(s_j(x))}$$

- prédiction du classifieur softmax

$$\hat{y} = \operatorname{argmax}_k \sigma(s(x))_k = \operatorname{argmax}_k s_k(x) = \operatorname{argmax}_k ((\theta^{(k)})^T \cdot x)$$

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

k Plus Proches Voisins

- k Nearest Neighbors (k-NN)
- basé sur le principe que les choses qui se ressemblent ont tendance à avoir des propriétés semblables
- classification des données en les plaçant dans des catégories similaires (plus proches voisins)

k-NN

- Principe : classer des données non étiquetées en lui assignant la classe de données étiquetées similaires
- Exemple d'applications
 - vision par ordinateur : reconnaissance de caractères, ou faciale
 - systèmes de recommandation
 - identification de motifs dans des données génétiques

k-NN

- Principe : classer des données non étiquetées en lui assignant la classe de données étiquetées similaires
- Exemple d'applications
 - vision par ordinateur : reconnaissance de caractères, ou faciale
 - systèmes de recommandation
 - identification de motifs dans des données génétiques
- bien adapté aux tâches de classification où les relations sont complexes mais où les items de classes similaires sont assez homogènes
- si les données sont bruitées ou s'il n'y a pas de distinction claire entre les groupes \Rightarrow difficile d'identifier les frontières des classes

k-NN : Principe

- utilise l'information des k plus proches voisins pour classifier des données non labellisées
- k : variable indiquant le nb de voisins à utiliser
- après choix de k , l'algorithme nécessite un jeu d'entraînement avec des exemples classés en plusieurs catégories

k-NN : Principe

- utilise l'information des k plus proches voisins pour classifier des données non labellisées
- k : variable indiquant le nb de voisins à utiliser
- après choix de k , l'algorithme nécessite un jeu d'entraînement avec des exemples classés en plusieurs catégories
- pour chaque donnée du jeu de test, l'algorithme identifie les k enregistrements du jeu d'entraînement qui sont les plus "proches" en terme de similarité
- on assigne à l'observation du jeu de test la classe représentant la majorité des k plus proches voisins

k-NN

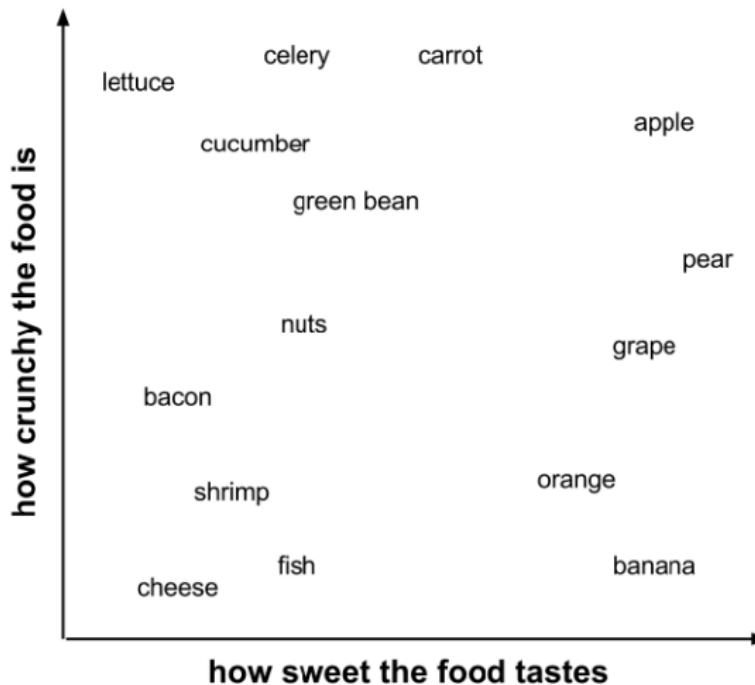
Avantages	Inconvénients
Simple et efficace Pas d'hypothèses sur la distribution des données Phase d'entraînement rapide	Ne produit pas un modèle Nécessite la sélection d'un k approprié Phase de classification lente Données manquantes requièrent des traitements supplémentaires

k-NN : Exemple

ingredient	sweetness	crunchiness	food type
apple	10	9	fruit
bacon	1	4	protein
banana	10	1	fruit
carrot	7	10	vegetable
celery	3	10	vegetable
cheese	1	1	protein

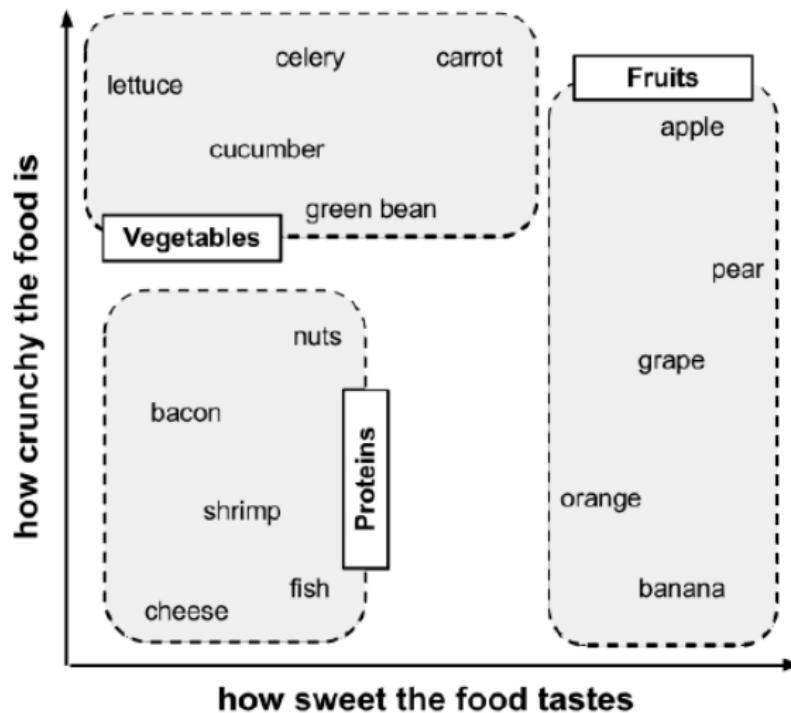
Source [Lantz19]

k-NN : Exemple



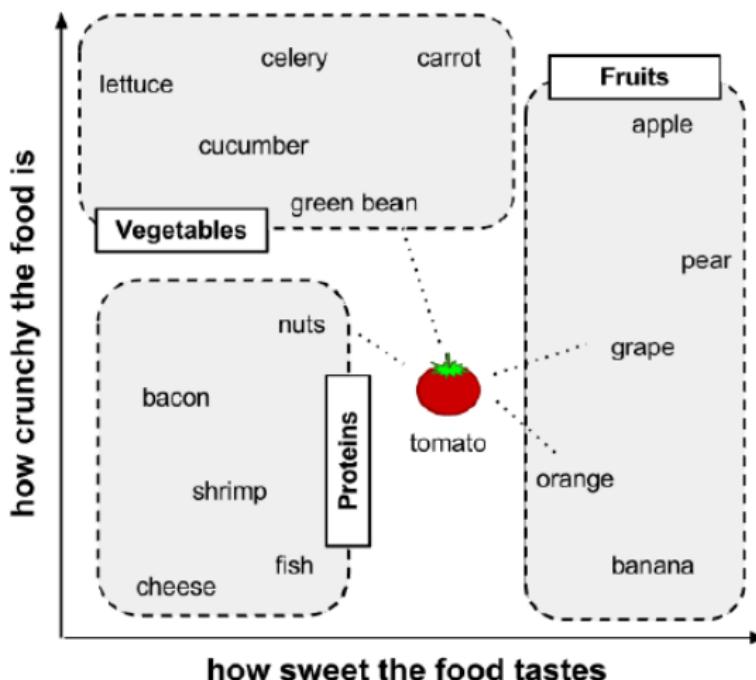
Source [Lantz19]

k-NN : Exemple



Source [Lantz19]

k-NN : Exemple



Source [Lantz19]

Mesure de la similarité

- distance pour mesurer la similarité entre 2 observations
- distance Euclidienne :

$$dist(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2}$$

Mesure de la similarité

- distance pour mesurer la similarité entre 2 observations
- distance Euclidienne :

$$dist(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2}$$

- Exemple : Tomate : sweetness =6 et crunchiness=4

ingredient	sweetness	crunchiness	food type	distance to the tomato
grape	8	5	fruit	$sqrt((6 - 8)^2 + (4 - 5)^2) = 2.2$
green bean	3	7	vegetable	$sqrt((6 - 3)^2 + (4 - 7)^2) = 4.2$
nuts	3	6	protein	$sqrt((6 - 3)^2 + (4 - 6)^2) = 3.6$
orange	7	3	fruit	$sqrt((6 - 7)^2 + (4 - 3)^2) = 1.4$

Source [Lantz19]

Mesure de la similarité

- distance pour mesurer la similarité entre 2 observations
- distance Euclidienne :

$$dist(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2}$$

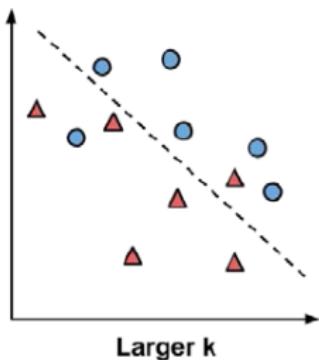
- Exemple : Tomate : sweetness =6 et crunchiness=4
 - classe du plus proche voisin qui est l'orange \Rightarrow fruit

ingredient	sweetness	crunchiness	food type	distance to the tomato
grape	8	5	fruit	$sqrt((6 - 8)^2 + (4 - 5)^2) = 2.2$
green bean	3	7	vegetable	$sqrt((6 - 3)^2 + (4 - 7)^2) = 4.2$
nuts	3	6	protein	$sqrt((6 - 3)^2 + (4 - 6)^2) = 3.6$
orange	7	3	fruit	$sqrt((6 - 7)^2 + (4 - 3)^2) = 1.4$

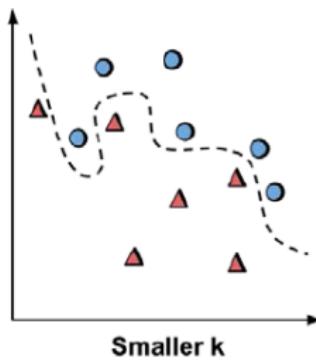
Source [Lantz19]

Choix de la valeur de k

- choix de k détermine comment le modèle va généraliser aux données futures
 - choix d'un grand k réduit l'impact de la variance causée par des données bruitées mais peut biaiser le classifieur (risque d'ignorer les petits motifs)
⇒ compromis biais-variance



Larger k



Smaller k

Source [Lantz19]

Choix de la valeur de k

En pratique

- souvent $k = \sqrt{N}$ avec N : nb de données d'entraînement
- alternative :
 - tester différentes valeurs de k sur différents jeux de tests
 - et choisir celle qui fournit la meilleure classification

Exemple

On souhaite prédire si une boisson est un thé ou un café. On a recueilli les données suivantes :

Volume (mL)	250	100	125	250
Caféine (g)	0.025	0.010	0.050	0.100
Boisson	Thé	Thé	Café	Café

- En utilisant l'algorithme du plus proche voisin avec une distance Euclidienne, quelle est l'étiquette prédite pour une boisson de 125 mL contenant 0.015 g de caféine ?

Exemple

On souhaite prédire si une boisson est un thé ou un café. On a recueilli les données suivantes :

Volume (mL)	250	100	125	250
Caféine (g)	0.025	0.010	0.050	0.100
Boisson	Thé	Thé	Café	Café

- En utilisant l'algorithme du plus proche voisin avec une distance Euclidienne, quelle est l'étiquette prédite pour une boisson de 125 mL contenant 0.015 g de caféine ?
- Cette classification semble-t-elle correcte ? Si non, que faire pour y remédier ?

Préparation des données

- Mise à l'échelle des différentes caractéristiques en rétrécissant ou en étendant leur intervalle pour que chacune contribue de manière équitable dans la formule de distance

Préparation des données

- Mise à l'échelle des différentes caractéristiques en rétrécissant ou en étendant leur intervalle pour que chacune contribue de manière équitable dans la formule de distance
- Méthodes
 - normalisation min-max

$$X_{new} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

Préparation des données

- Mise à l'échelle des différentes caractéristiques en rétrécissant ou en étendant leur intervalle pour que chacune contribue de manière équitable dans la formule de distance
- Méthodes
 - normalisation min-max

$$X_{new} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

- standardisation z-score

$$X_{new} = \frac{X - \mu}{\sigma}$$

Préparation des données

Cas des caractéristiques de type catégorique

- conversion dans un format numérique
- solution : valeur 1 pour une catégorie et 0 pour une autre

$$homme = \begin{cases} 1 & \text{si } x = \text{homme} \\ 0 & \text{sinon} \end{cases} \quad (1)$$

Préparation des données

Cas des caractéristiques de type catégorique

- variable n -catégories peut être codée par des variables binaires pour $n - 1$ niveaux
- ex : variable de température avec 3 catégories (chaud, tiède, froid) peut être remplacée par 2 caractéristiques

$$chaud = \begin{cases} 1 & \text{si } x = \text{chaud} \\ 0 & \text{sinon} \end{cases} \quad (1)$$

$$ti\acute{e}de = \begin{cases} 1 & \text{si } x = \text{ti\acute{e}de} \\ 0 & \text{sinon} \end{cases} \quad (2)$$

k-NN : en résumé

- entraînement paresseux (*lazy learning*)
- coût algorithmique d'une précision peut être élevé si la base d'entraînement est grande
- qualité des prédictions dépend du choix d'une bonne distance ou similarité

k-NN : en résumé

- entraînement paresseux (*lazy learning*)
- coût algorithmique d'une précision peut être élevé si la base d'entraînement est grande
- qualité des prédictions dépend du choix d'une bonne distance ou similarité
- fonctionne mieux en faible dimension
 - exécution plus rapide
 - moins susceptible d'être biaisé par des variables non pertinentes
 - moins susceptible de souffrir du fléau de la dimension (*curse of dimensionality*)

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

Méthodes Bayésiennes

- classifieurs basés sur les méthodes Bayésiennes utilisent les données d'entraînement pour calculer la probabilité de chaque résultat basée sur les informations fournies par les caractéristiques
- classifieur utilise ces probabilités pour prédire la classe la plus probable pour une nouvelle observation

Méthodes Bayésiennes

- classifieurs basés sur les méthodes Bayésiennes utilisent les données d'entraînement pour calculer la probabilité de chaque résultat basée sur les informations fournies par les caractéristiques
- classifieur utilise ces probabilités pour prédire la classe la plus probable pour une nouvelle observation
- Exemples d'utilisation :
 - classification de texte(filtrage de spam, ...)
 - détection d'anomalies ou d'intrusion dans des réseaux
 - diagnostic médical à partir d'un ensemble de symptômes

Rappel sur les probabilités

- probabilité d'un évènement A est estimée à partir des données observées

$$P(A) = \frac{\text{nb d'expériences où A apparaît}}{\text{nb total d'expériences}}$$

- ex

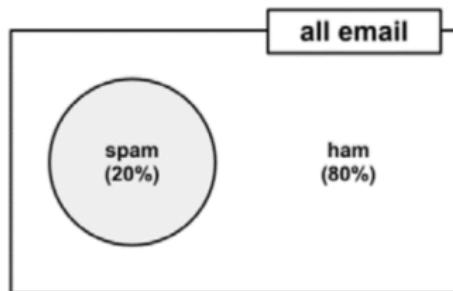
- $P(\text{spam})=0.2$
- $\Rightarrow P(\text{ham})=1-0.2=0.8$

Rappel sur les probabilités

- probabilité d'un évènement A est estimée à partir des données observées

$$P(A) = \frac{\text{nb d'expériences où A apparaît}}{\text{nb total d'expériences}}$$

- ex
 - $P(\text{spam})=0.2$
 - $\Rightarrow P(\text{ham})=1-0.2=0.8$
- A et B indépendants : $P(A \cap B) = P(A)P(B)$



Rappel sur les probabilités

$$P(A \setminus B) = \frac{P(A \cap B)}{P(B)}$$

théorème de Bayes :

$$P(A \setminus B) = \frac{P(B \setminus A)P(A)}{P(B)}$$

Rappel sur les probabilités

$$P(A \setminus B) = \frac{P(A \cap B)}{P(B)}$$

théorème de Bayes :

$$P(A \setminus B) = \frac{P(B \setminus A)P(A)}{P(B)}$$

$$P(\text{spam} \mid \text{Win}) = \frac{P(\text{Win} \mid \text{spam})P(\text{spam})}{P(\text{Win})}$$

Diagram illustrating the components of the Bayes' theorem formula:

- likelihood**: points to $P(\text{Win} \mid \text{spam})$
- prior probability**: points to $P(\text{spam})$
- marginal likelihood**: points to $P(\text{Win})$
- posterior probability**: points to $P(\text{spam} \mid \text{Win})$

Rappel sur les probabilités

- création d'une table de fréquence et d'une table de vraisemblance

Frequency	Win		Total
	Yes	No	
spam	4	16	20
ham	1	79	80
Total	5	95	100

Likelihood	Win		Total
	Yes	No	
spam	4 / 20	16 / 20	20
ham	1 / 80	79 / 80	80
Total	5 / 100	95 / 100	100

Algorithme Naïve Bayes

- définit une méthode simple pour appliquer le théorème de Bayes aux problèmes de classification
- Naïve Bayes : hypothèses "naïves" sur les données :
 - hypothèse : toutes les caractéristiques du jeu de données sont **d'importance égale et indépendantes**
 - rarement vrai dans la plupart des applications réelles

Algorithme Naïve Bayes

- définit une méthode simple pour appliquer le théorème de Bayes aux problèmes de classification
- Naïve Bayes : hypothèses "naïves" sur les données :
 - hypothèse : toutes les caractéristiques du jeu de données sont **d'importance égale et indépendantes**
 - rarement vrai dans la plupart des applications réelles
- fonctionne cependant bien dans la plupart des cas, même lorsque ces hypothèses ne sont pas respectées

Naïve Bayes

Avantages	Inconvénients
<ul style="list-style-type: none">- Simple, rapide et efficace- Fonctionne bien sur des données manquantes et bruitées- peu d'exemples d'apprentissage requis mais fonctionne également bien sur des grandes bases- Facile d'obtenir la probabilité estimée pour une prédiction	<ul style="list-style-type: none">- basé sur une hypothèse souvent fausse : caractéristiques indépendantes et d'importance égale- pas idéal pour des jeux de données avec bcp de caractéristiques numériques- probabilités estimées moins fiables que les classes prédites

Classification Bayésienne naïve

	Win (W_1)		Money (W_2)		Groceries (W_3)		Unsubscribe (W_4)		
Likelihood	Yes	No	Yes	No	Yes	No	Yes	No	Total
spam	4 / 20	16 / 20	10 / 20	10 / 20	0 / 20	20 / 20	12 / 20	8 / 20	20
ham	1 / 80	79 / 80	14 / 80	66 / 80	8 / 80	71 / 80	23 / 80	57 / 80	80
Total	5 / 100	95 / 100	24 / 100	76 / 100	8 / 100	91 / 100	35 / 100	65 / 100	100

Classification Bayésienne naïve

	Win (W_1)		Money (W_2)		Groceries (W_3)		Unsubscribe (W_4)		
Likelihood	Yes	No	Yes	No	Yes	No	Yes	No	Total
spam	4 / 20	16 / 20	10 / 20	10 / 20	0 / 20	20 / 20	12 / 20	8 / 20	20
ham	1 / 80	79 / 80	14 / 80	66 / 80	8 / 80	71 / 80	23 / 80	57 / 80	80
Total	5 / 100	95 / 100	24 / 100	76 / 100	8 / 100	91 / 100	35 / 100	65 / 100	100

$$P(C_L | F_1, \dots, F_n) = \frac{1}{Z} p(C_L) \prod_{i=1}^n p(F_i | C_L)$$

Estimateur de Laplace

- Problème : si un évènement ne se produit jamais pour un ou plusieurs niveaux de la classe, alors leur probabilité jointe sera nulle
⇒ probabilité a posteriori sera nulle

Estimateur de Laplace

- Problème : si un évènement ne se produit jamais pour un ou plusieurs niveaux de la classe, alors leur probabilité jointe sera nulle
 - ⇒ probabilité a posteriori sera nulle
 - ⇒ estimateur de Laplace : ajoute un petit nb à chacun des compteurs de la table des fréquences
 - ⇒ chaque caractéristique a une probabilité non nulle de se produire dans chaque classe.

$$P(w' | \text{positive}) = \frac{\text{number of reviews with } w' \text{ and } y = \text{positive} + \alpha}{N + \alpha * K}$$

Estimateur de Laplace

- Problème : si un évènement ne se produit jamais pour un ou plusieurs niveaux de la classe, alors leur probabilité jointe sera nulle
 - ⇒ probabilité a posteriori sera nulle
 - ⇒ estimateur de Laplace : ajoute un petit nb à chacun des compteurs de la table des fréquences
 - ⇒ chaque caractéristique a une probabilité non nulle de se produire dans chaque classe.

$$P(w' | \text{positive}) = \frac{\text{number of reviews with } w' \text{ and } y = \text{positive} + \alpha}{N + \alpha * K}$$

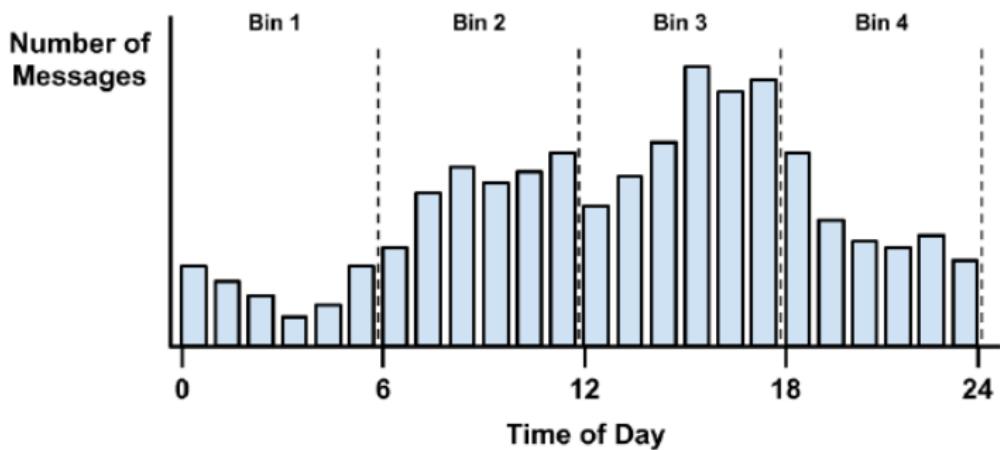
- en pratique on rajoute 1

Utilisation de caractéristiques numériques

- Naive Bayes utilise des tables de fréquence pour l'apprentissage des données,
- ⇒ chaque caractéristique doit être de type catégorie

Utilisation de caractéristiques numériques

- Naive Bayes utilise des tables de fréquence pour l'apprentissage des données,
⇒ chaque caractéristique doit être de type catégorie
- Pour des données numériques : discréétisation



Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

Arbres de décision

- modèles de Machine Learning supervisés et non paramétriques
- utilisables pour la classification ou pour la régression
- utilisent des méthodes purement algorithmiques

Principe

- classer une observation au moyen d'une succession de questions (critères de segmentation) concernant les valeurs des variables prédictives x_i de cette observation

Principe

- classer une observation au moyen d'une succession de questions (critères de segmentation) concernant les valeurs des variables prédictives x_i de cette observation
- 1 question représentée par 1 noeud d'un arbre de décision

Principe

- classer une observation au moyen d'une succession de questions (critères de segmentation) concernant les valeurs des variables prédictives x_i de cette observation
- 1 question représentée par 1 noeud d'un arbre de décision
- Chaque branche sortante du noeud correspond à 1 réponse possible à la question posée

Principe

- classer une observation au moyen d'une succession de questions (critères de segmentation) concernant les valeurs des variables prédictives x_i de cette observation
- 1 question représentée par 1 noeud d'un arbre de décision
- Chaque branche sortante du noeud correspond à 1 réponse possible à la question posée
- Feuille (ou noeud terminal) dans laquelle parvient l'observation à l'issue de la suite de questions \Rightarrow classe

Phase d'apprentissage

- trouver les bonnes questions à poser pour ranger correctement un ensemble $x^{(1)}, \dots, x^{(N)}$ de N observations dotées d'étiquettes $y^{(1)}, \dots, y^{(N)}$

Phase d'apprentissage

- trouver les bonnes questions à poser pour ranger correctement un ensemble $x^{(1)}, \dots, x^{(N)}$ de N observations dotées d'étiquettes $y^{(1)}, \dots, y^{(N)}$
- Trouver un arbre optimal : problème complexe

Arbres de décision

- \neq variantes : stratégies \neq pour associer des critères de segmentation aux noeuds

Arbres de décision

- \neq variantes : stratégies \neq pour associer des critères de segmentation aux noeuds
- objectif : feuilles homogènes

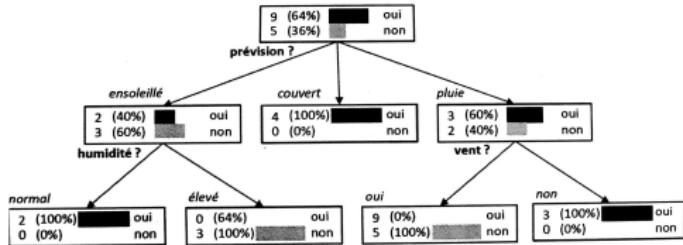
Arbres de décision

- \neq variantes : stratégies \neq pour associer des critères de segmentation aux noeuds
- objectif : feuilles homogènes
- stratégie : associer aux noeuds des critères de segmentation qui décision après décision accroiront progressivement cette homogénéité

Arbres de décision

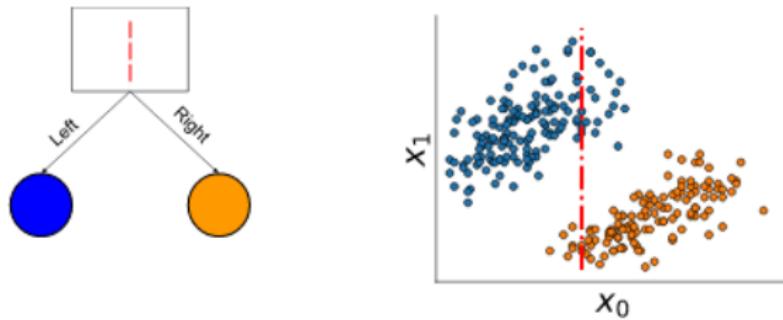
	prévision	température	humidité	vent	activité
1	couvert	froid	normal	oui	oui
2	couvert	chaud	élevé	non	oui
3	couvert	chaud	normal	non	oui
4	couvert	moyen	élevé	oui	oui
5	pluvieux	froid	normal	oui	non
6	pluvieux	moyen	élevé	oui	non
7	pluvieux	froid	normal	non	oui
8	pluvieux	moyen	élevé	non	oui
9	pluvieux	moyen	normal	non	oui
10	ensoleillé	chaud	élevé	non	non
11	ensoleillé	chaud	élevé	oui	non
12	ensoleillé	moyen	élevé	non	non
13	ensoleillé	froid	normal	non	oui
14	ensoleillé	moyen	normal	oui	oui

(a)

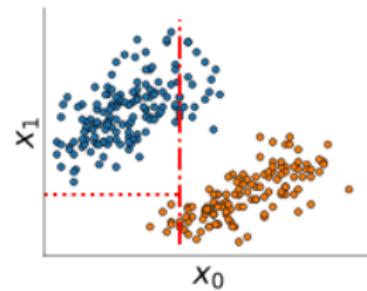
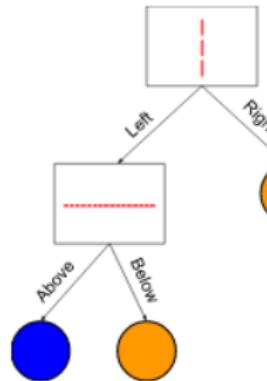


(b)

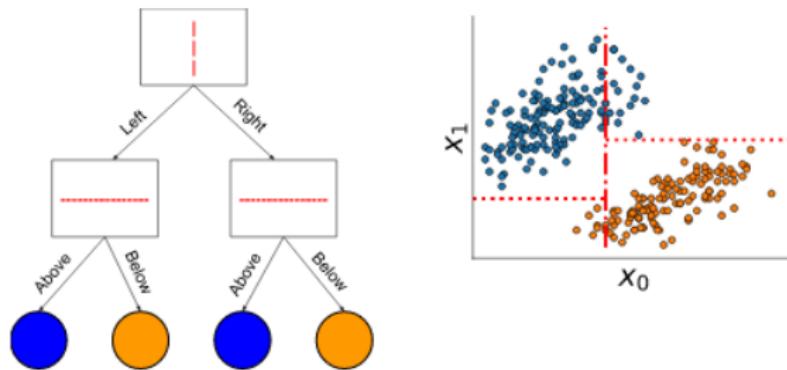
Classification avec un arbre de décision



Classification avec un arbre de décision



Classification avec un arbre de décision



Critère d'homogénéité/hétérogénéité

- Impureté de Gini

$$G_i = 1 - \sum_{k=1}^n p_{i,k}^2$$

- $p_{i,k}$: pourcentage d'observation de la classe k parmi toutes les observations d'entraînement dans le $i^{\text{ème}}$ nœud
- entropie

$$H_i = - \sum_{k=1}^n p_{i,k} \log_2(p_{i,k})$$

Algorithme CART

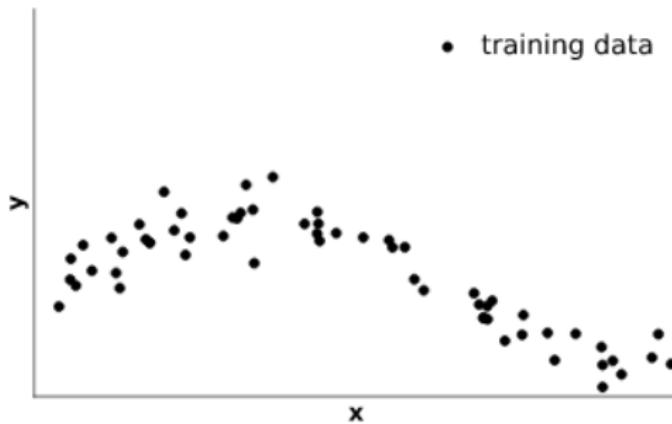
- *Classification And Regression Trees*

- séparation du jeu d'entraînement en 2 sous-ens en utilisant une seule caractéristique k et un seuil t_k
- k, t_k ?
- recherche la paire (k, t_k) qui produit les sous-ens les + "purs" (pondérés par leur taille)
- fonction de coût que l'algorithme cherche à minimiser

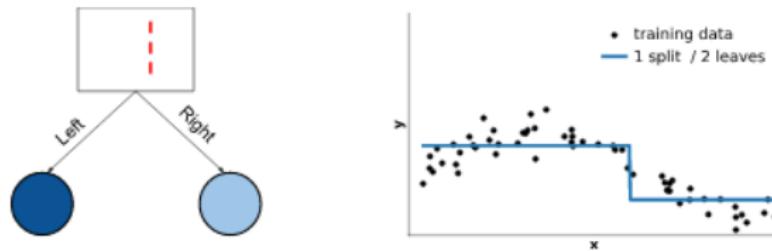
$$J(k, t_k) = \frac{m_{\text{gauche}}}{m} G_{\text{gauche}} + \frac{m_{\text{droite}}}{m} G_{\text{droite}}$$

- $G_{\text{gauche/droite}}$ mesure l'impureté du sous-ens de gauche/droite
- $m_{\text{gauche/droite}}$: nb d'observations du sous-ens de gauche/droite
- récursivité pour partager les sous-ens
- arrêt lorsque la profondeur max est atteinte ou qu'il n'existe plus de partage réduisant l'impureté

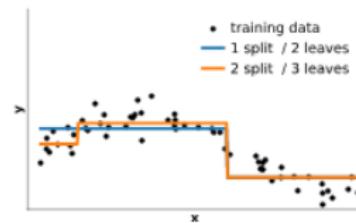
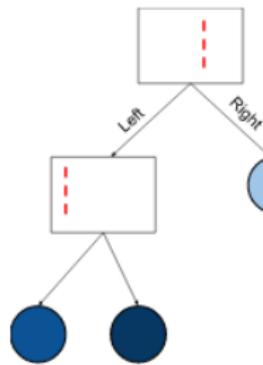
Régression avec un arbre de décision



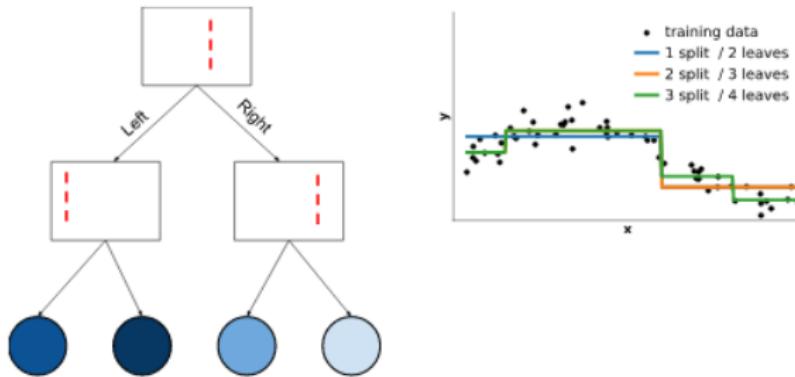
Régression avec un arbre de décision



Régression avec un arbre de décision



Régression avec un arbre de décision



Arbres de décision

- profondeur de l'arbre ?

Arbres de décision

- profondeur de l'arbre ?
 - risque de surapprentissage si on exige que toutes les observations soient parfaitement rangées

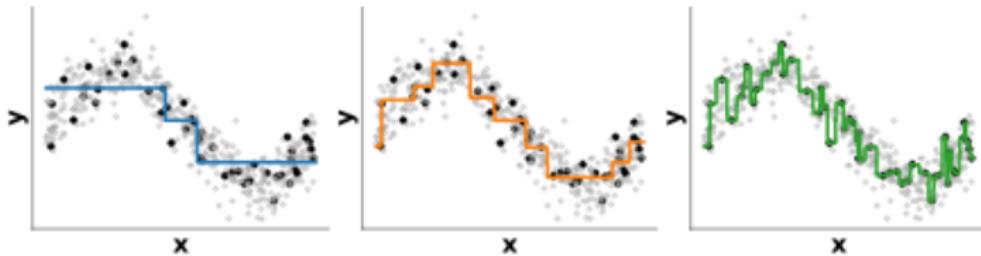
Arbres de décision

- profondeur de l'arbre ?
 - risque de surapprentissage si on exige que toutes les observations soient parfaitement rangées
 - ⇒ on décide de ne plus rajouter de noeuds
 - quand la profondeur de l'arbre > seuil
 - quand nb d'observations par feuille est trop faible pour être représentatif des \neq classes (pré-élagage)

Arbres de décision

- profondeur de l'arbre ?
 - risque de surapprentissage si on exige que toutes les observations soient parfaitement rangées
⇒ on décide de ne plus rajouter de noeuds
 - quand la profondeur de l'arbre > seuil
 - quand nb d'observations par feuille est trop faible pour être représentatif des \neq classes (pré-élagage)
- **prunning** : opérations d'élagage a posteriori sur des arbres dont les feuilles sont homogènes
 - utilisant un jeu de données distinct (*prunning set*) de celui qui a permis la construction de l'arbre original

Surajustement/Sous-ajustement



Underfitting

`max_depth` or
`max_leaf_nodes`
too small

Best trade-off

Overfitting

`max_depth` or
`max_leaf_nodes`
too large

Arbres de décision

Avantages	Inconvénients
<ul style="list-style-type: none">- variables d'entrée peuvent être qualitatives ou quantitatives- phase de préparation des données réduite- tient compte des interactions entre variables	<ul style="list-style-type: none">- risque de surapprentissage si l'arbre n'est pas correctement élagué- forte influence du critère de segmentation affecté au 1er noeud sur le modèle de prédiction

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

Forêts aléatoires

- Objectif : conserver la plupart des atouts des arbres de décision tout en éliminant leurs inconvénients
 - vulnérabilité au surapprentissage
 - complexité des opérations d'élargissement
- algorithme de classification ou de régression non paramétrique flexible et robuste

Forêts aléatoires

- ➊ échantillon initial de N obs, chacune décrite au moyen de p variables prédictives
 - B nouveaux échantillons de m taille N par tirage avec remise
 - ⇒ **bootstrap**
 - entraînement de B arbres de décisions \neq

Forêts aléatoires

- ➊ échantillon initial de N obs, chacune décrite au moyen de p variables prédictives
 - B nouveaux échantillons de m taille N par tirage avec remise
 - ⇒ **bootstrap**
 - entraînement de B arbres de décisions ≠
- ➋ utilisation de m variables ($m < p$) pour effectuer la segmentation associée au noeud d'un arbre

Forêts aléatoires

- ➊ échantillon initial de N obs, chacune décrite au moyen de p variables prédictives
 - B nouveaux échantillons de m taille N par tirage avec remise
⇒ **bootstrap**
 - entraînement de B arbres de décisions ≠
- ➋ utilisation de m variables ($m < p$) pour effectuer la segmentation associée au noeud d'un arbre
- ➌ classement d'une nouvelle observation :
 - on la fait passer par les B arbres
 - on sélectionne la classe majoritaire parmi les B prédictions

Forêts aléatoires

- estimation de l'erreur de prédiction
 - validation croisée incorporée à l'algorithme
 - $\simeq 1/3$ des observations réservé pour estimer l'erreur de prédiction

Forêts aléatoires

- estimation de l'erreur de prédiction
 - validation croisée incorporée à l'algorithme
 - $\simeq 1/3$ des observations réservé pour estimer l'erreur de prédiction
- estimation de l'importance des variables dans un cadre non linéaire
 - permutation des valeurs de la variable aléatoirement parmi le $1/3$ des observations réservées
 - évaluation de l'impact moyen de cette permutation sur les prédictions des B arbres
 - + l'impact est significatif, + la variable est importante

Forêts aléatoires

- estimation de l'erreur de prédiction
 - validation croisée incorporée à l'algorithme
 - $\approx 1/3$ des observations réservé pour estimer l'erreur de prédiction
- estimation de l'importance des variables dans un cadre non linéaire
 - permutation des valeurs de la variable aléatoirement parmi le $1/3$ des observations réservées
 - évaluation de l'impact moyen de cette permutation sur les prédictions des B arbres
 - + l'impact est significatif, + la variable est importante
- définition d'une notion de proximité entre observations
 - à chaque couple d'observations est associé un compteur
 - parcours des B arbres : quand 1 couple d'observations est rangé dans la même feuille \Rightarrow incrémentation du compteur associé
 - moyenne de chaque compteur sur les B arbres : mesure de proximité

Forêts aléatoires

Avantages	Inconvénients
<ul style="list-style-type: none">- un des meilleurs algos- ne souffre pas du surapprentissement- incorpore une étape de validation croisée- possibilité de traiter des données avec des milliers de var prédictives- bonne puissance prédictive même en cas de données manquantes	<ul style="list-style-type: none">- complexité importante-> implémentation délicate- ne conserve pas le caractère intelligible des arbres de décision

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

SVM

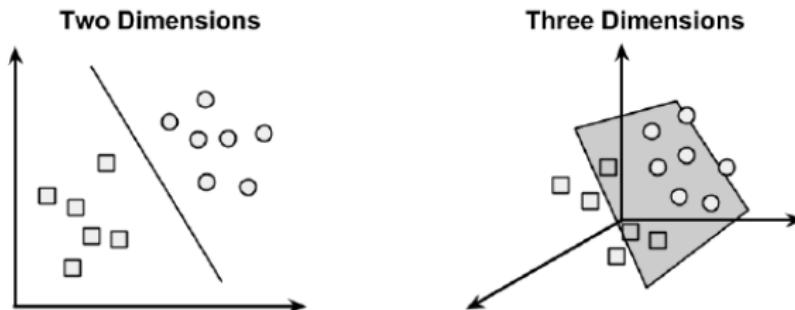
- machines à vecteurs de support (*Support Vector Machines*) ou séparateurs à vaste marge
- création d'hyperplans divisant l'espace pour créer des partitions homogènes
- permet de modéliser des relations très complexes

SVM

- machines à vecteurs de support (*Support Vector Machines*) ou séparateurs à vaste marge
- création d'hyperplans divisant l'espace pour créer des partitions homogènes
- permet de modéliser des relations très complexes
- Applications : classification et prédiction numérique
 - reconnaissance de formes (visages ...)
 - catégorisation de texte
 - détection d'événements rares
 - classification de données génétiques en bioinformatique

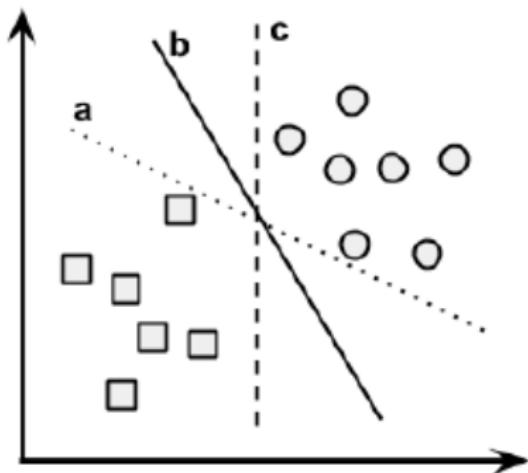
SVM

- hyperplan pour partitionner les données en groupes de valeurs similaires
- Exemple : données linéairement séparables



Source [Lantz2019]

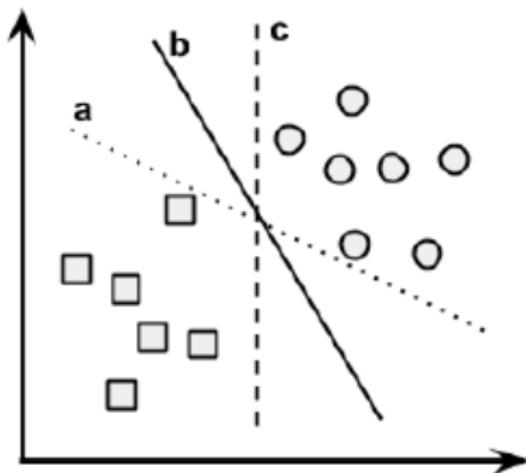
SVM



Source [Lantz2019]

- Quelle ligne choisir ?

SVM

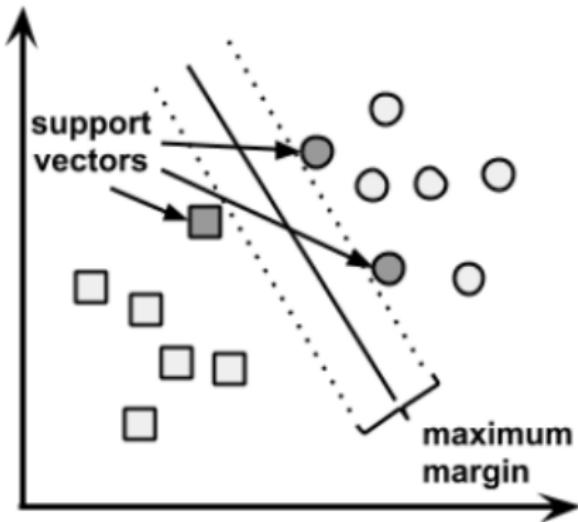


Source [Lantz2019]

- Quelle ligne choisir ?
- recherche de l'hyperplan à marge optimale/maximale (*maximum margin hyperplane*)

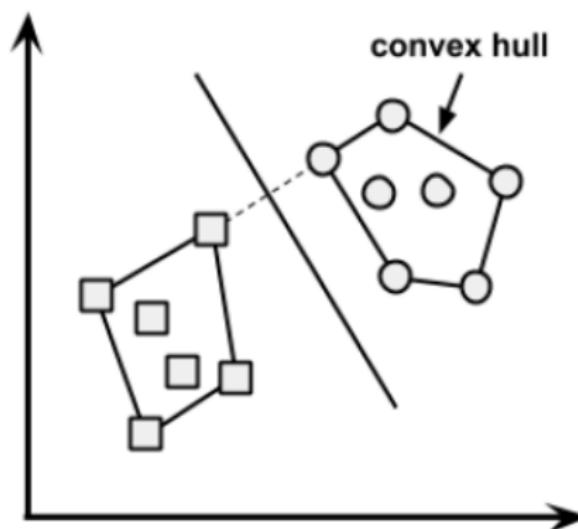
SVM

- vecteurs supports (*support vectors*) : points de chaque classe les plus proches de l'hyperplan à marge maximale : définissent l'hyperplan à marge maximale
- possibilité de stocker de façon compacte le modèle de classification



Cas de données linéairement séparables

- hyperplan à marge maximale : aussi loin que possible des frontières externes des 2 groupes de données
- frontières externes : **enveloppe convexe**(*convex hull*)



Source [Lantz2019]

Cas de données linéairement séparables

Equations

- hyperplan

$$f(x) = \langle w, x \rangle + b = 0$$

w vecteur orthogonal au plan, b biais

- un point est bien classé ssi $yf(x) > 0$
- on peut imposer $|yf(x)| \geq 1$
- distance d'un point à l'hyperplan est donnée par

$$d(x) = \frac{|\langle w, x \rangle + b|}{\|w\|} = \frac{|f(x)|}{\|w\|}$$

Cas de données linéairement séparables

Equations

- on cherche à maximiser la marge
- revient à chercher w qui minimise

$$\min_w \frac{1}{2} \|w\|^2$$

sous les contraintes $\forall i, \langle w, x_i \rangle + b \geq 1$

Cas de données linéairement séparables

Equations

- on cherche à maximiser la marge
- revient à chercher w qui minimise

$$\min_w \frac{1}{2} \|w\|^2$$

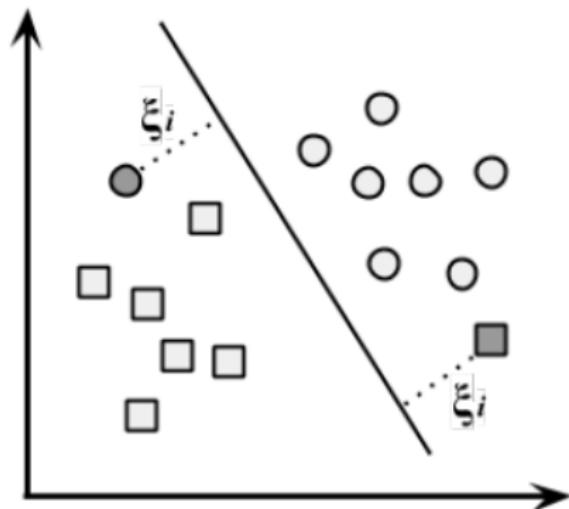
sous les contraintes $\forall i, \langle w, x_i \rangle + b \geq 1$

- problème d'optimisation est résolu par méthode «primal/dual» et la solution s'écrit sous la forme

$$f(x) = \sum_{i=1}^n \lambda_i^* y_i \langle x, x_i \rangle + b^*$$

- la plupart des λ_i^* sont nuls sauf pour certaines observations (vecteurs supports)

Cas non séparable



Source [Lantz2019]

Cas non séparable

- Lorsque les données ne sont pas séparables par un plan, on assouplit les contraintes en introduisant une variable d'ajustement (*slack variable*) par :

$$y_i f(x_i) \geq 1 - \xi_i$$

Cas non séparable

- Lorsque les données ne sont pas séparables par un plan, on assouplit les contraintes en introduisant une variable d'ajustement (*slack variable*) par :

$$y_i f(x_i) \geq 1 - \xi_i$$

- le problème d'optimisation devient

$$\min_w \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i$$

sous les contraintes $\forall i, \langle w, x_i \rangle + b \geq 1 - \xi_i$

Cas non séparable

- Lorsque les données ne sont pas séparables par un plan, on assouplit les contraintes en introduisant une variable d'ajustement (*slack variable*) par :

$$y_i f(x_i) \geq 1 - \xi_i$$

- le problème d'optimisation devient

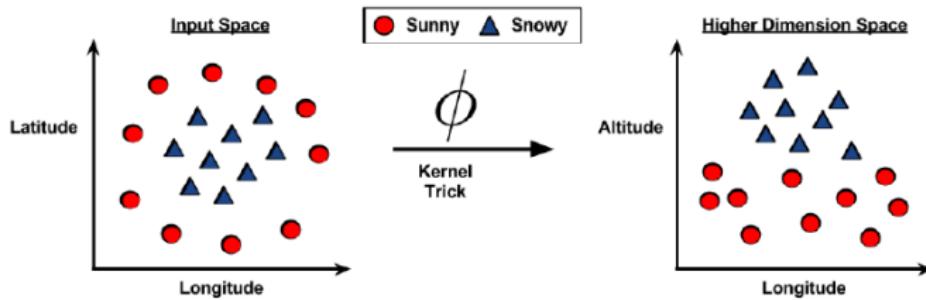
$$\min_w \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i$$

sous les contraintes $\forall i, \langle w, x_i \rangle + b \geq 1 - \xi_i$

- Le problème se résout sous la même forme que précédemment, les λ_i sont bornés par C

Espaces non-linéaires : utilisation de noyaux

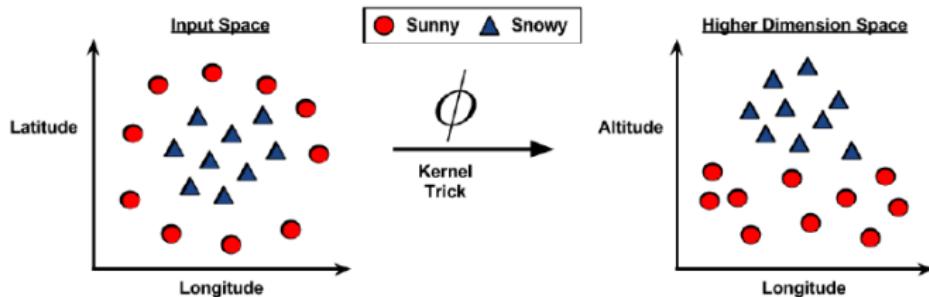
- relations entre variables souvent non-linéaires
- possibilité de transposer les données dans un espace de dimension supérieur : astuce du noyau (*kernel trick*) : espace de description \Rightarrow espace de redescription



Source [Lantz2019]

Espaces non-linéaires : utilisation de noyaux

- relations entre variables souvent non-linéaires
- possibilité de transposer les données dans un espace de dimension supérieur : astuce du noyau (*kernel trick*) : espace de description \Rightarrow espace de redescription



Source [Lantz2019]

- Fonction noyau

$$K(x, x') = \Phi(x) \cdot \Phi(x')$$

Exemple de noyaux

- linéaire

$$K(x, x') = x \cdot x'$$

- polynômial (pour d=2 : noyau quadratique)

$$K(x, x') = (c + x \cdot x')^d$$

- Gaussien (ou noyau RBF *radial basis function*)

$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

- Laplacien

$$K(x, x') = \exp\left(-\frac{\|x - x'\|}{\sigma}\right)$$

SVM

Avantages	Inconvénients
<ul style="list-style-type: none">- utilisé pour des problèmes de classification ou de prédiction numérique- pas trop influencé par des données bruitées et peu enclin au surajustement- populaires grâce à leur précision	<ul style="list-style-type: none">-pour trouver le bon modèle, nécessité de tester différentes combinaisons de noyaux et de paramètres du modèle- peut être lent à entraîner si le jeu de données d'entrée a bcp de caractéristiques ou entrées- modèle "black box" complexe difficile voire impossible à interpréter

Plan

- 1 Apprentissage supervisé
- 2 Régression
- 3 Régression logistique
- 4 k-NN
- 5 Classification Naïve Bayésienne
- 6 Arbres de décision
- 7 Forêts aléatoires
- 8 SVM (Support Vector Machines)
- 9 Réseaux de neurones

Réseaux de neurones

Artificial Neural Networks (ANN)

- modélise la relation entre un ensemble de signaux d'entrée et un signal de sortie
- analogie avec un cerveau
- ANN utilise un réseau de neurones artificiels ou noeuds (*nodes*) pour résoudre des pb d'apprentissage

Réseaux de neurones

Artificial Neural Networks (ANN)

- utilisé depuis une cinquantaine d'années pour simuler l'approche cérébrale pour la résolution de pb
 - au début : apprentissage de fonctions simples (ET / OU logiques)

Réseaux de neurones

Artificial Neural Networks (ANN)

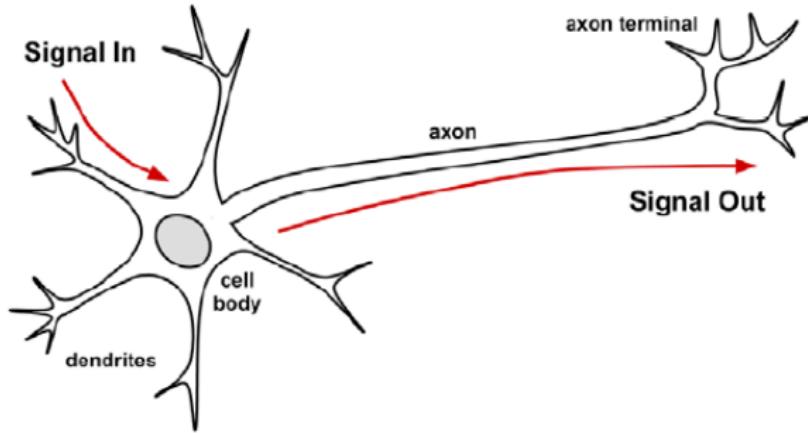
- utilisé depuis une cinquantaine d'années pour simuler l'approche cérébrale pour la résolution de pb
 - au début : apprentissage de fonctions simples (ET / OU logiques)
- Applications :
 - reconnaissance de paroles, écriture, images
 - modèles sophistiqués de temps ou climats ...

Réseaux de neurones

Artificial Neural Networks (ANN)

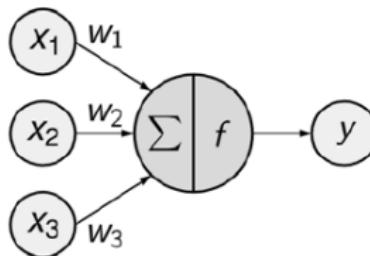
- utilisé depuis une cinquantaine d'années pour simuler l'approche cérébrale pour la résolution de pb
 - au début : apprentissage de fonctions simples (ET / OU logiques)
- Applications :
 - reconnaissance de paroles, écriture, images
 - modèles sophistiqués de temps ou climats ...
- souvent appliqués à des pb où les données d'entrée et de sortie sont bien définies mais où le processus qui relie l'entrée à la sortie est très complexe et difficile à définir

Neurones biologiques



Machine Learning With R - Brett Lantz

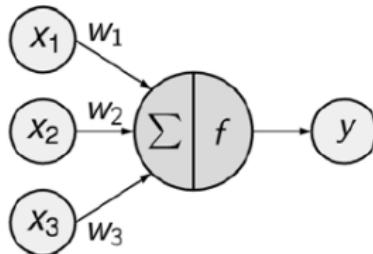
Neurones artificiels



Machine Learning With R - Brett Lantz

- variables x : signaux d'entrée, variable y : signal de sortie
- w : poids
- f : fonction d'activation

Neurones artificiels



Machine Learning With R - Brett Lantz

- variables x : signaux d'entrée, variable y : signal de sortie
- w : poids
- f : fonction d'activation

$$y(x) = f \left(\sum_{i=1}^n w_i x_i \right)$$

Réseaux neuronaux

- de nombreuses variantes, mais des caractéristiques communes

Caractéristiques

- fonction d'activation

Réseaux neuronaux

- de nombreuses variantes, mais des caractéristiques communes

Caractéristiques

- fonction d'activation
- topologie du réseau (ou architecture) : décrit le nombre de neurones du modèle, le nombre de couches et la manière dont ils sont connectés

Réseaux neuronaux

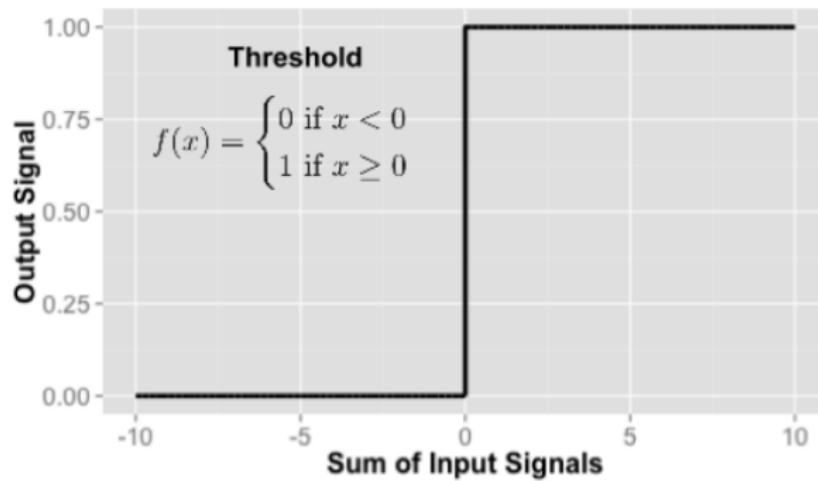
- de nombreuses variantes, mais des caractéristiques communes

Caractéristiques

- fonction d'activation
- topologie du réseau (ou architecture) : décrit le nombre de neurones du modèle, le nombre de couches et la manière dont ils sont connectés
- algorithme d'entraînement : spécifie comment les poids de connexion sont définis

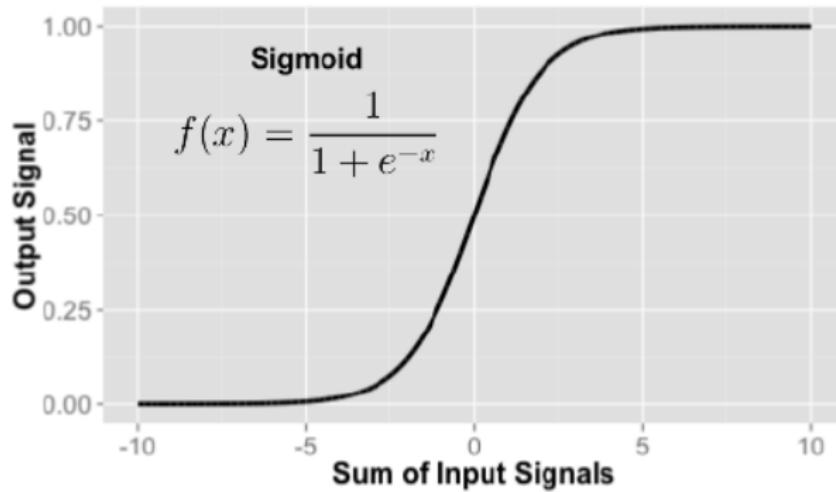
Fonctions d'activation

- *threshold activation function (unit step activation function) :* échelon unité (fonction de Heaviside)



Fonctions d'activation

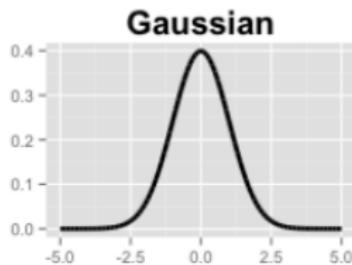
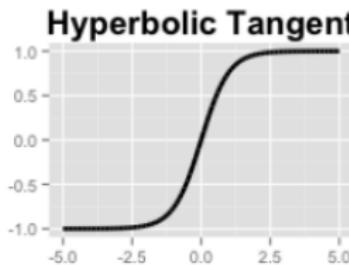
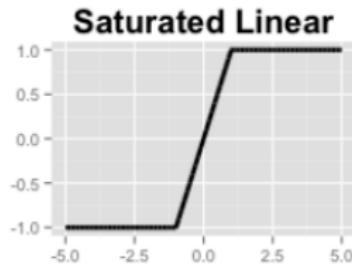
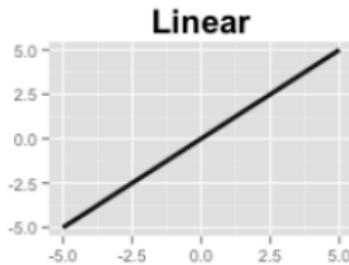
- Sigmoïde : la plus utilisée, souvent choisie par défaut



⇒ Fonction différentiable

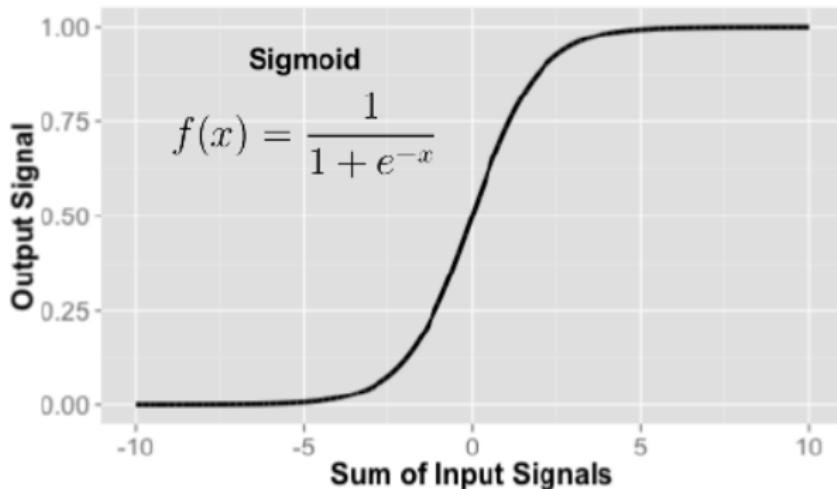
Fonctions d'activation

- Autres exemples de fonctions d'activation



Fonctions d'activation

- Pour certaines fonctions d'activation, l'intervalle des valeurs d'entrée qui affecte le signal de sortie est relativement étroit



- ⇒ *squashing function* (Fonction d'écrasement)
- ⇒ solution : normalisation/standardisation

Topologie du réseau

- la capacité d'un réseau de neurones à apprendre est lié à sa topologie
- 3 caractéristiques principales

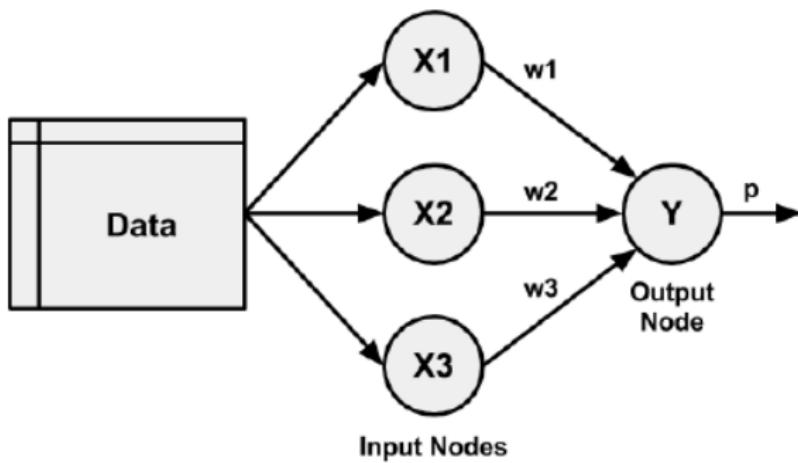
Topologie du réseau

- la capacité d'un réseau de neurones à apprendre est lié à sa topologie
- 3 caractéristiques principales
 - nb de couches
 - direction de l'information dans le réseau
 - nb de noeuds de chaque couche

Topologie du réseau

- la capacité d'un réseau de neurones à apprendre est lié à sa topologie
- 3 caractéristiques principales
 - nb de couches
 - direction de l'information dans le réseau
 - nb de noeuds de chaque couche
- la topologie détermine la complexité des tâches pouvant être apprises par le réseau

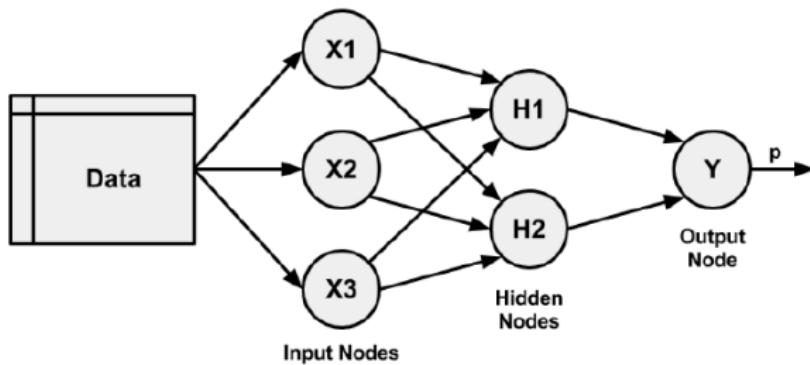
Nombre de couches



Single-layer network [Lantz2019]

- Single-layer network : pour des tâches de classification basiques

Nombre de couches

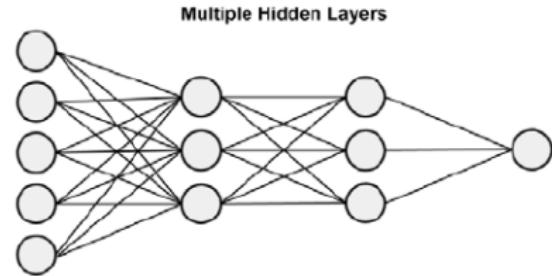
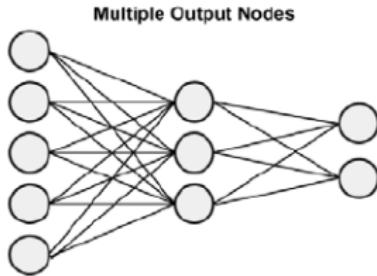


Multilayer network [Lantz2019]

- réseau multi-couche (*multi-layer network*) : ajout d'une ou plusieurs couches cachées
- souvent *fully connected*

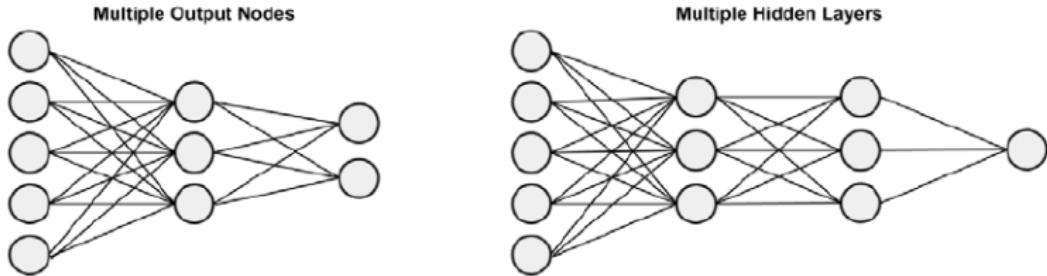
Direction du trajet de l'information

- Réseau de neurones à propagation avant (*feedforward network*) : l'information ne se déplace que dans une seule direction, vers l'avant



Direction du trajet de l'information

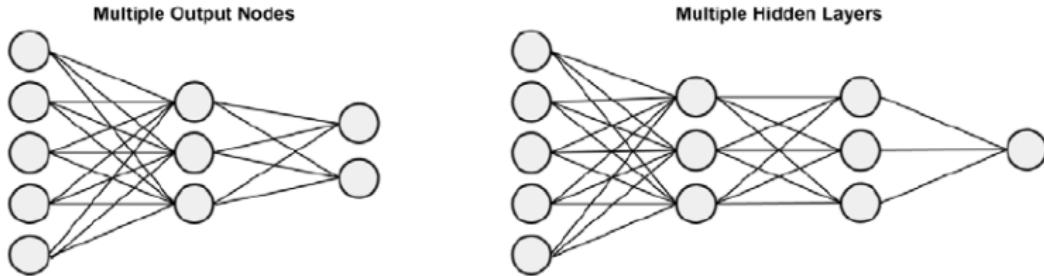
- Réseau de neurones à propagation avant (*feedforward network*) : l'information ne se déplace que dans une seule direction, vers l'avant



- réseau de neurones avec plusieurs couches cachées : réseau de neurones profond (*deep neural network*)

Direction du trajet de l'information

- Réseau de neurones à propagation avant (*feedforward network*) : l'information ne se déplace que dans une seule direction, vers l'avant



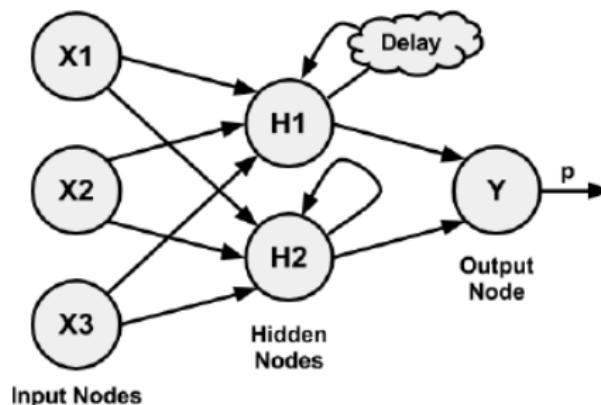
- réseau de neurones avec plusieurs couches cachées : réseau de neurones profond (*deep neural network*)
- **perceptron multicouche** (*multilayer perceptron*) : multilayer feedforward network : topologie standard des réseaux de neurones artificiels

Direction du trajet de l'information

- réseaux récurrents (*recurrent network, feedback network*)
 - ajout d'un retard (*delay*) \Rightarrow augmente la puissance du réseau

Direction du trajet de l'information

- réseaux récurrents (*recurrent network, feedback network*)
 - ajout d'un retard (*delay*) \Rightarrow augmente la puissance du réseau
 - ex : séquences d'évènements sur une période donnée
- \Rightarrow Applications : prédictions de marché boursier, compréhension de la parole, prévisions météorologiques



Ex de réseau récurrent [Lantz2019]

Nombre de noeuds dans chaque couche

- nb de noeuds d'entrée dépend du nb de caractéristiques des données d'entrée
- nb de noeuds de sortie dépend du nb de résultat à modéliser ou du nb de classes du résultat

Nombre de noeuds dans chaque couche

- nb de noeuds d'entrée dépend du nb de caractéristiques des données d'entrée
- nb de noeuds de sortie dépend du nb de résultat à modéliser ou du nb de classes du résultat
- nb de noeuds cachés : à l'utilisateur de décider avant l'entraînement du modèle
 - pas de règle fiable

Nombre de noeuds dans chaque couche

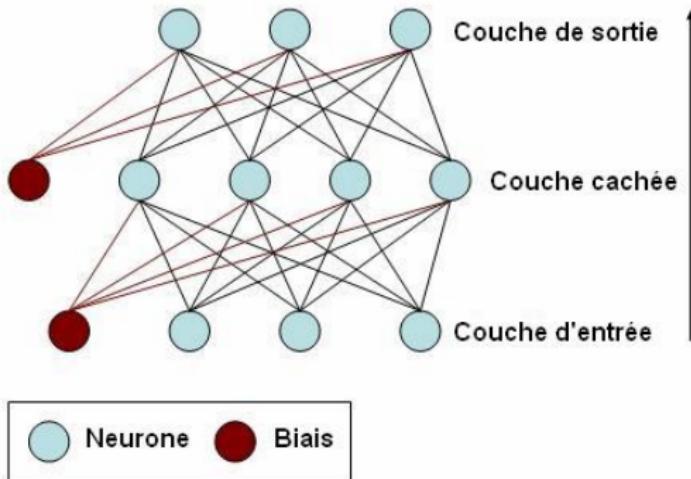
- nb de noeuds d'entrée dépend du nb de caractéristiques des données d'entrée
- nb de noeuds de sortie dépend du nb de résultat à modéliser ou du nb de classes du résultat
- nb de noeuds cachés : à l'utilisateur de décider avant l'entraînement du modèle
 - pas de règle fiable
 - topologies complexes avec un grand nb de connexions permettent l'apprentissage de pb plus complexes
 - nb élevé de neurones \Rightarrow modèle qui reflète très fortement les données d'entraînement \Rightarrow risque de sur-ajustement des données

Nombre de noeuds dans chaque couche

- nb de noeuds d'entrée dépend du nb de caractéristiques des données d'entrée
- nb de noeuds de sortie dépend du nb de résultat à modéliser ou du nb de classes du résultat
- nb de noeuds cachés : à l'utilisateur de décider avant l'entraînement du modèle
 - pas de règle fiable
 - topologies complexes avec un grand nb de connexions permettent l'apprentissage de pb plus complexes
 - nb élevé de neurones \Rightarrow modèle qui reflète très fortement les données d'entraînement \Rightarrow risque de sur-ajustement des données
 - réseaux complexes : coûteux en calcul, et phase d'apprentissage lente
 - en pratique : utiliser réseau avec le moins de noeuds possible qui obtient des performances adéquates sur le jeu de validation

Biais

- biais : neurone qui émet un signal d'intensité 1



Entraînement

- entraînement d'un réseau de neurones en ajustant les poids de connexion est très coûteux en calcul

Entraînement

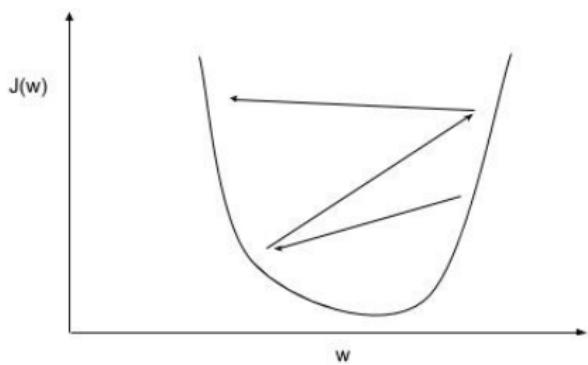
- entraînement d'un réseau de neurones en ajustant les poids de connexion est très coûteux en calcul
- algorithme de **rétropropagation** (*backpropagation*)
 - plusieurs cycles d'apprentissage (*epoch*)
 - au départ : poids définis de manière aléatoire
 - chaque cycle de l'algorithme contient 2 phases
 - phase forward : propagation de l'activité
 - phase backward : rétropropagation de l'erreur

Entraînement

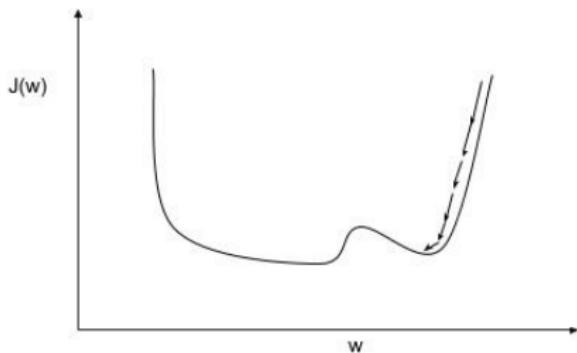
- entraînement d'un réseau de neurones en ajustant les poids de connexion est très coûteux en calcul
- algorithme de **rétropropagation** (*backpropagation*)
 - plusieurs cycles d'apprentissage (*epoch*)
 - au départ : poids définis de manière aléatoire
 - chaque cycle de l'algorithme contient 2 phases
 - phase forward : propagation de l'activité
 - phase backward : rétropropagation de l'erreur
 - descente de gradient
 - vitesse d'apprentissage (*learning rate*)

Rétropropagation

- vitesse d'apprentissage (*learning rate*)



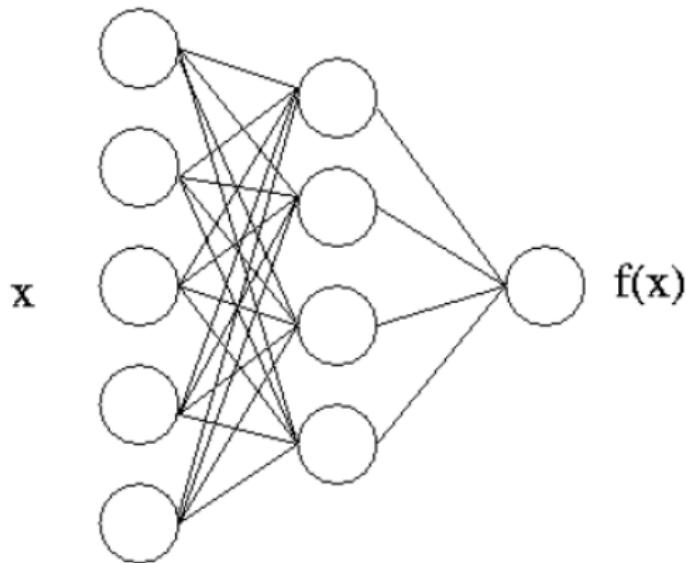
Vitesse d'apprentissage élevée :
dépassement du minimum,
pas de convergence



Vitesse d'apprentissage faible :
piège d'un minimum local
convergence lente

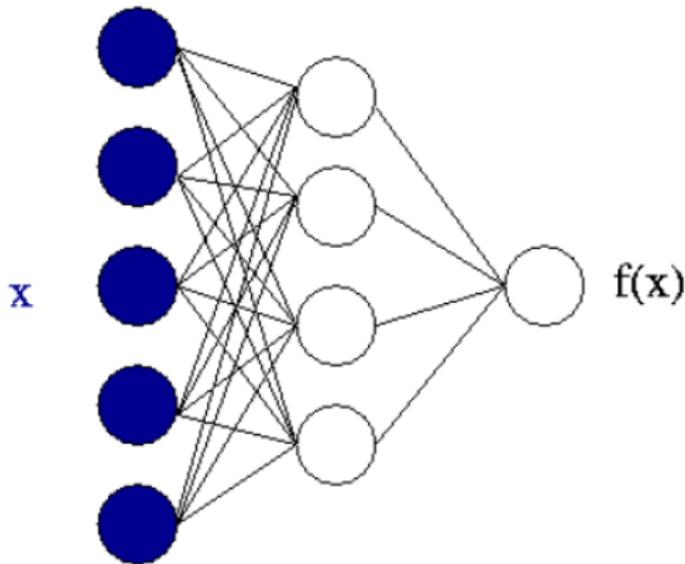
Rétropropagation

- Propagation de l'activité
- Initialiser les poids des liens entre les neurones



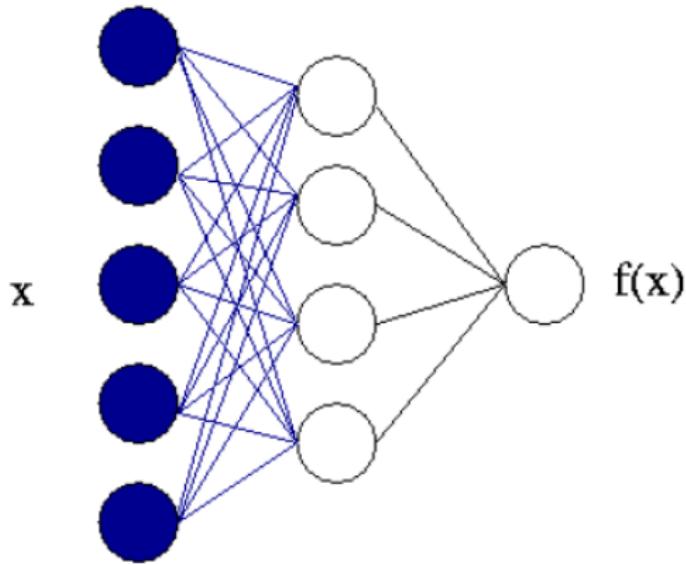
Rétropropagation

- Propagation de l'activité
 - Application d'un vecteur entrées-sorties à apprendre



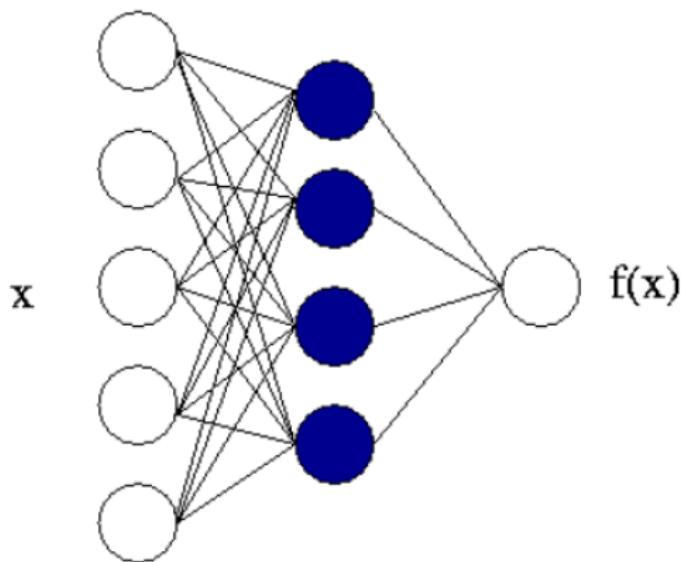
Rétropropagation

- Propagation de l'activité
 - Calcul des sorties du réseau à partir des entrées qui lui sont appliquées



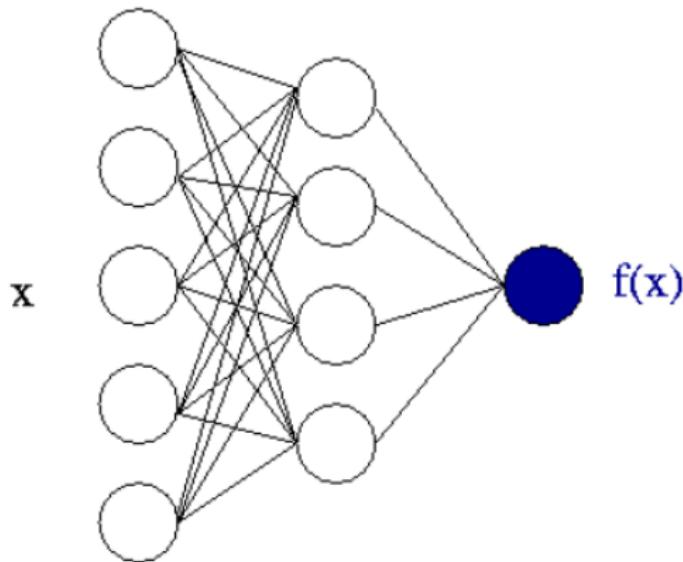
Rétropropagation

- Propagation de l'activité
 - Calcul des sorties du réseau à partir des entrées qui lui sont appliquées



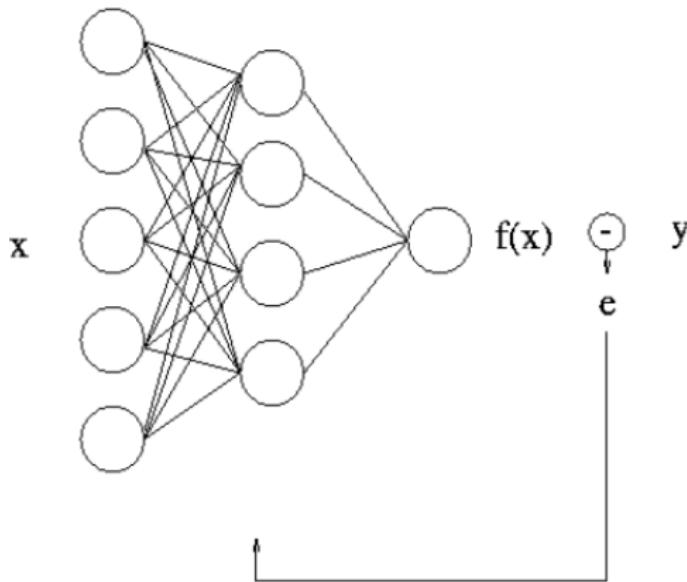
Rétropropagation

- Propagation de l'activité
 - Calcul des sorties du réseau à partir des entrées qui lui sont appliquées



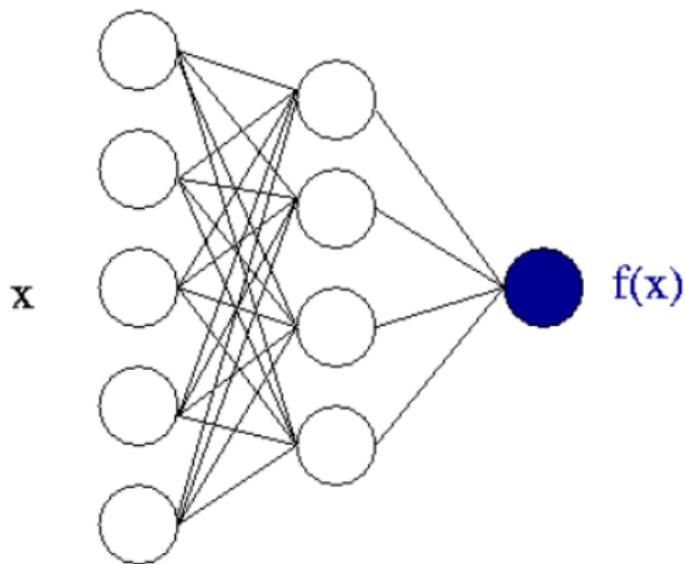
Rétropropagation

- Rétropropagation de l'erreur
 - Calcul de l'erreur entre les sorties calculées et les sorties idéales à apprendre



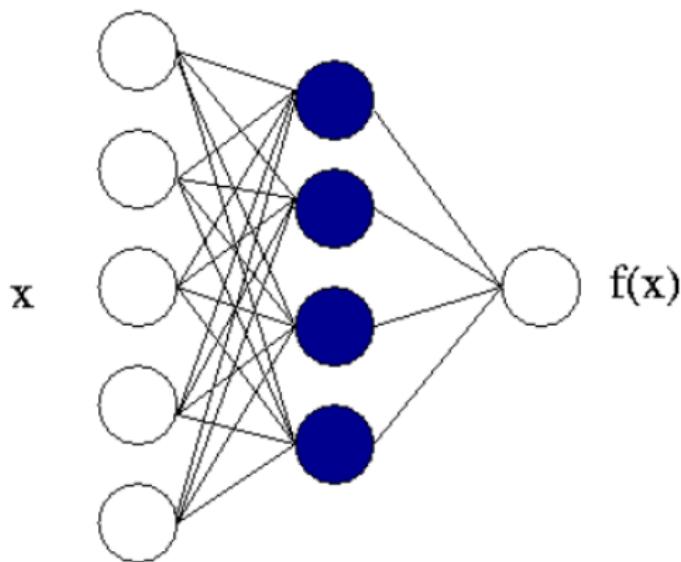
Rétropropagation

- Rétropropagation de l'erreur
 - Correction des poids des liens entre les neurones de la couche de sortie et de la couche cachée selon l'erreur présente en sortie



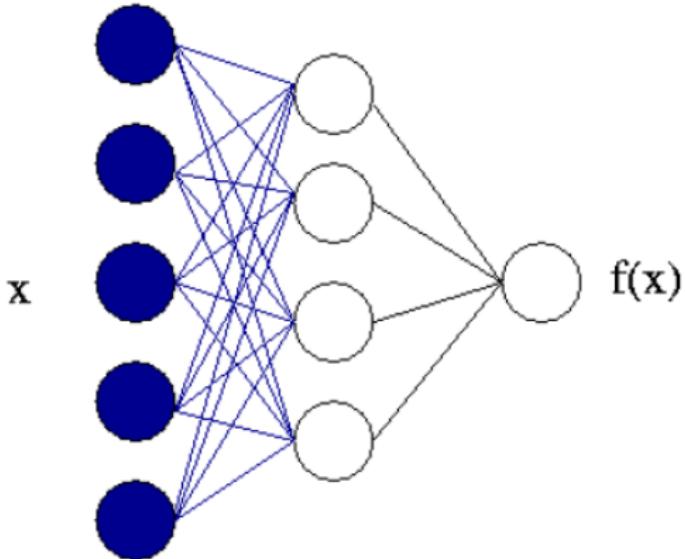
Rétropropagation

- Rétropropagation de l'erreur
 - Correction des poids des liens entre les neurones de la couche de sortie et de la couche cachée selon l'erreur présente en sortie

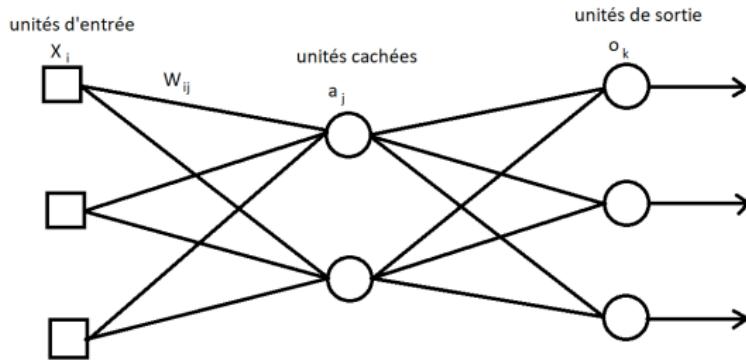


Rétropropagation

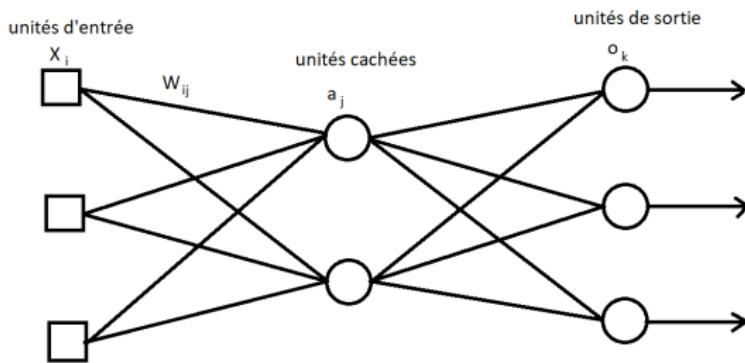
- Rétropropagation de l'erreur
 - Propagation de l'erreur sur la couche précédente et correction des poids des liens entre les neurones de la couche cachée et ceux en entrée



Rétropropagation



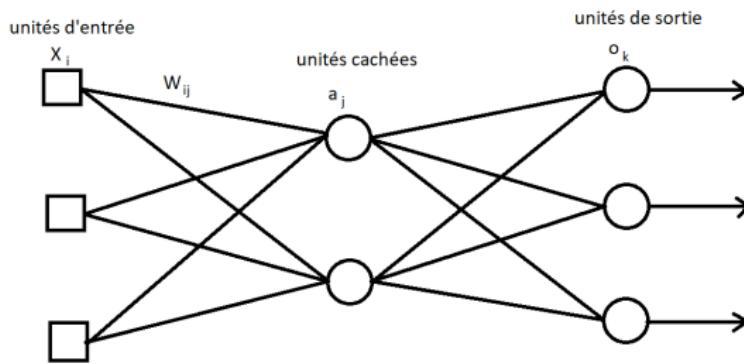
Rétropropagation



1) Calcul activations unités cachées

$$a_i = f(S_i)$$

Rétropropagation



1) Calcul activations unités cachées

$$S_j = \sum x_i w_{ij}$$

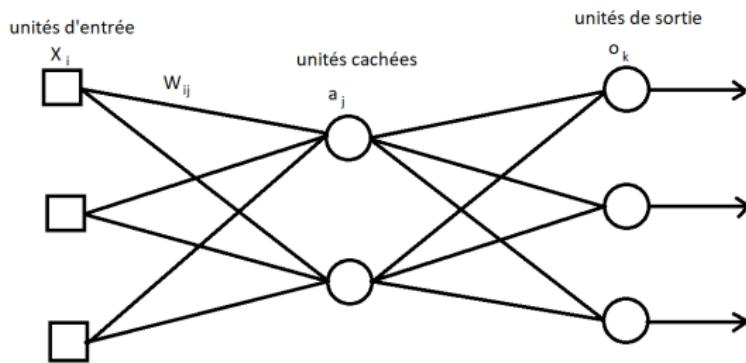
$$a_i = f(s_i)$$

2) Calcul activations unités de sortie

$$S_k = \sum_j a_j W_{jk}$$

$$o_k = f(S_k)$$

Rétropropagation



1) Calcul activations unités cachées

$$S_j = \sum_i x_i W_{ij}$$

$$a_j = f(S_j)$$

2) Calcul activations unités de sortie

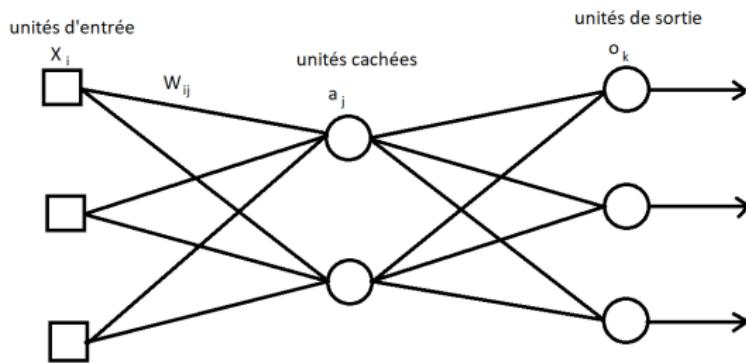
$$S_k = \sum_j a_j W_{jk}$$

$$o_k = f(S_k)$$

3) Calcul Erreur entre sorties désirées et sorties obtenues

$$e_k = d_k - o_k$$

Rétropropagation



1) Calcul activations unités cachées

$$S_j = \sum_i x_i W_{ij}$$

$$a_j = f(S_j)$$

2) Calcul activations unités de sortie

$$S_k = \sum_j a_j W_{jk}$$

$$o_k = f(S_k)$$

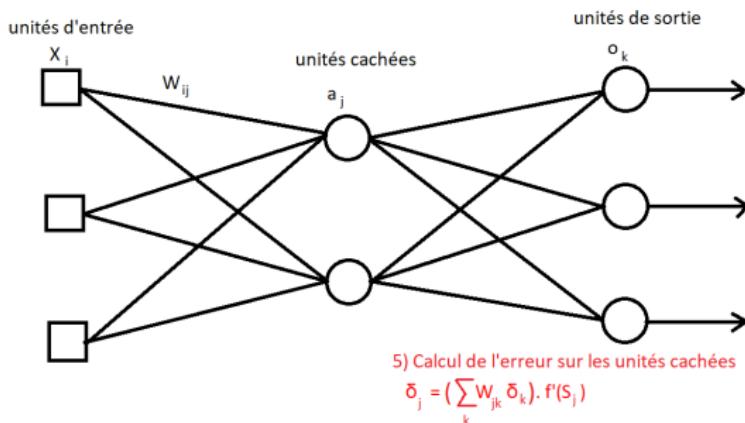
3) Calcul Erreur entre sorties désirées et sorties obtenues

$$e_k = d_k - o_k$$

4) Calcul de l'erreur sur les unités de sortie

$$\delta_k = e_k f'(S_k)$$

Rétropropagation



1) Calcul activations unités cachées

$$S_j = \sum x_i w_{ij}$$

$$a_1 = f(S)$$

2) Calcul activations unités de sortie

$$S_k = \sum_j a_j W_{jk}$$

$$o_k = f(S_k)$$

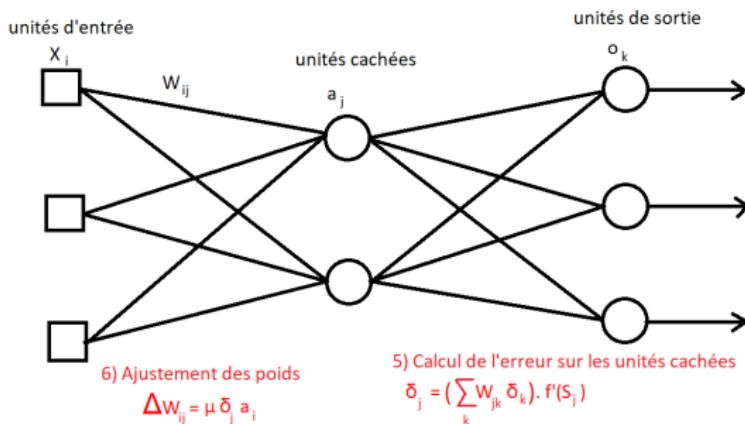
3) Calcul Erreur entre sorties désirées et sorties obtenues

$$e_k = d_k - o_k$$

4) Calcul de l'erreur sur les unités de sortie

$$\delta_k = e_k f'(S_k)$$

Rétropropagation



1) Calcul activations unités cachées

$$S_j = \sum x_i w_{ij}$$

$$a_1 = f(S)$$

2) Calcul activations unités de sortie

$$S_k = \sum_j a_j W_{jk}$$

$$o_k = f(S_k)$$

3) Calcul Erreur entre sorties désirées et sorties obtenues

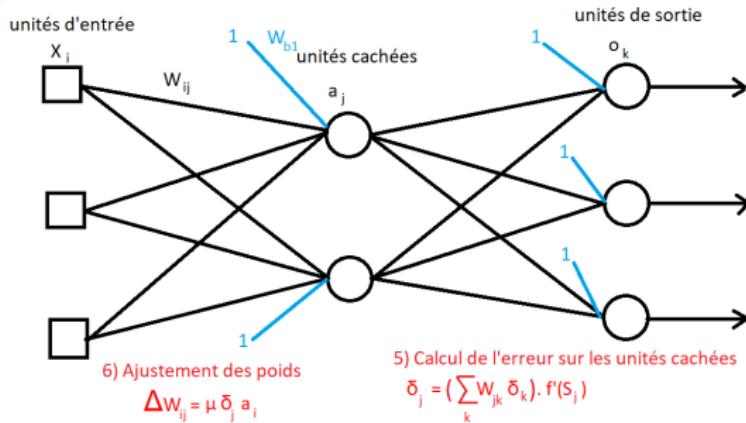
$$e_k = d_k - o_k$$

4) Calcul de l'erreur sur les unités de sortie

$$\delta_k = e_k f'(S_k)$$

Rétropropagation

Avec les biais



1) Calcul activations unités cachées

$$S_j = \sum x_i W_{ij} + W_{bj}$$

$$a_1 = f(S)$$

2) Calcul activations unités de sortie

$$S_k = \sum_i a_i W_{ik} + W_{bk}$$

$$o_k = f(S_k)$$

3) Calcul Erreur entre sorties désirées et sorties obtenues

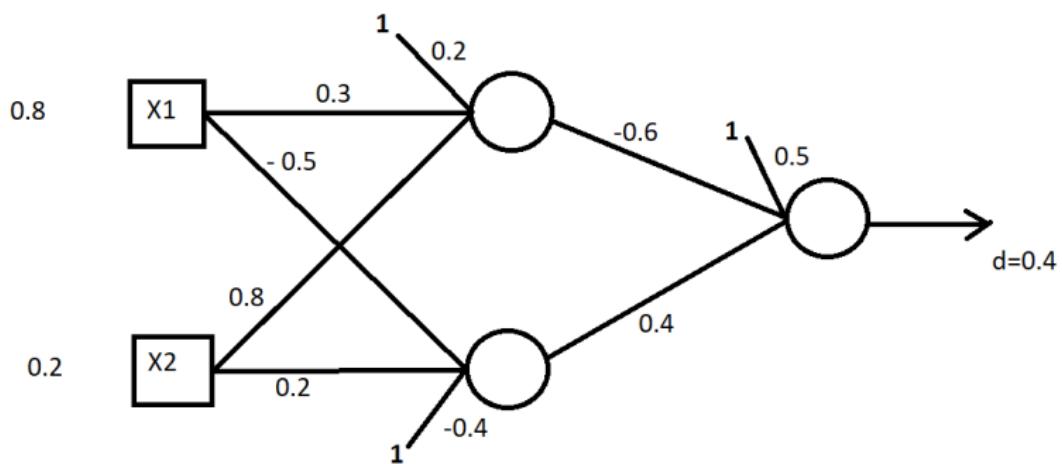
$$e_k = d_k - o_k$$

4) Calcul de l'erreur sur les unités de sortie

$$\delta_k = e_k f'(S_k)$$

Rétropropagation

- ex : $x_1 = 0.8, x_2 = 0.2$, sortie souhaitée $d = 0.4$



Réseaux de neurones à propagation avant multicouches

Avantages	Inconvénients
<ul style="list-style-type: none">- peut être adapté pour des pb de classification ou de prédiction numérique- capable de modéliser des patterns complexes- fait peu d'hypothèses sur les relations entre les données	<ul style="list-style-type: none">- coût en calcul, entraînement lent, en particulier si la topologie du réseau est complexe- enclin à sur-ajuster les données- modèle "black box", difficile voire impossible à interpréter