Formation Chef de Projet IA

Fondamentaux du Machine Learning

Chloé-Agathe Azencott

Center for Computational Biology (CBIO)
Mines ParisTech – Institut Curie – INSERM U900
PSL Research University & PR[AI]RIE, Paris, France

Janvier 2022

http://cazencott.info chloe-agathe.azencott@mines-paristech.fr @cazencott

Chloé-Agathe Azencott

Depuis 2013 **Chargée de recherche** puis **Maîtresse-assistante** au CBIO

2011–2013 **Chargée de recherche post-doctorante**
Machine Learning for Computational Biology, MPI Tübingen (Allemagne).

2005–2010 **Doctorante**
Institute for Genomics and Bioinformatics, University of California Irvine (USA).

2002–2005 **Double diplôme Ingénieur & Master**
IMT Atlantlique (à l'époque, ENST Bretagne
Informatique et Mathématiques.

Objectifs

- Présenter les concepts et outils fondamentaux pour développer et analyser des algorithmes d'apprentissage automatique :
  - Fondements théoriques
  - Panorama d'algorithmes
  - Choisir et évaluer un algorithme / un modèle d'apprentissage

Programme

- Introduction à l'apprentissage automatique
- **Cours 1 :** Minimisation du risque empirique
- **Cours 2 :** Régularisation et sélection de modèle
- **Cours 3 :** Arbres et méthodes ensemblistes
- **Cours 4 :** Méthodes à noyaux
- **Cours 5 :** Réduction de dimension
- **Cours 6 :** Clustering
- Travaux pratiques : mise en œuvre des notions et algorithmes avec scikit-learn (Python numérique).

Introduction à l'apprentissage

Qu’est-ce que l’intelligence artificielle ?

Qu'est-ce que l'intelligence artificielle ?

- Dartmouth Conference, 1956

  The science and engineering of making computers behave in ways that, until recently, we thought required human intelligence.

- Reproduire **artificiellement** les comportements du vivant **perçus comme intelligents**

Qu'est-ce que l'intelligence artificielle ?

- Dartmouth Conference, 1956

  The science and engineering of making computers behave in ways that, until recently, we thought required human intelligence.

- Reproduire **artificiellement** les comportements du vivant **perçus comme intelligents**

- **Motivations :**
  - **Comprendre** l'intelligence naturelle ;
  - Créer des **outils** qui nous aident.

Quelques sous-domaines de l'IA

- **Robotique**
  - mécanique, électronique, contrôle
- **Systèmes experts**
  - utilisent des règles
  - logique des propositions, logique des prédicats
- **Traitement du langage naturel**
- **Vision par ordinateur**
- **Informatique affective**
  - reconnaître, modéliser, exprimer les émotions humaines
  - sciences cognitives, psychologie
- **Apprentissage automatique**

Quelques sous-domaines de l'IA

- **Robotique**
  - mécanique, électronique, contrôle
- **Systèmes experts**
  - utilisent des règles
  - logique des propositions, logique des prédicats
- **Traitement du langage naturel**
- **Vision par ordinateur**
- **Informatique affective**
  - reconnaître, modéliser, exprimer les émotions humaines
  - sciences cognitives, psychologie
- **Apprentissage automatique**

Apprentissage automatique

- **Apprendre :**

Apprentissage automatique

- **Apprendre :** acquérir une **compétence** par l'**expérience**, la pratique.
- Pour une machine :
  - **compétence** = algorithme
  - **expérience,** pratique = exemples, données

Apprentissage automatique

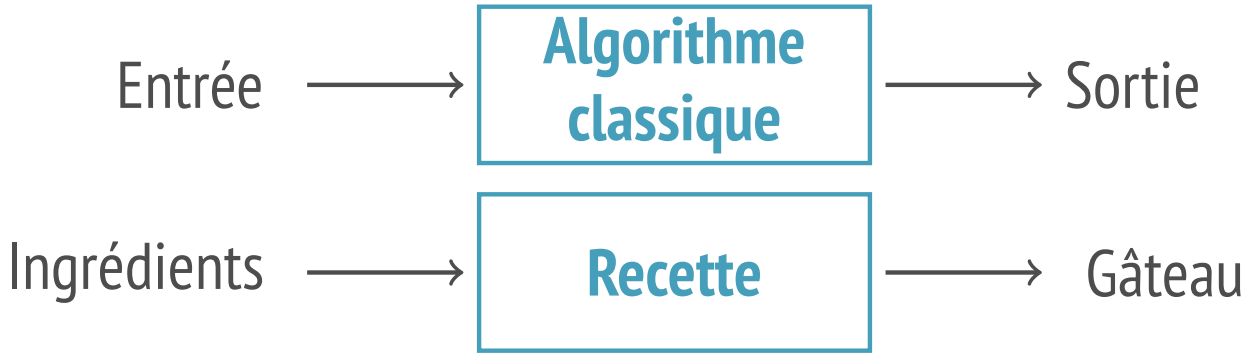
- **Apprendre :** acquérir une **compétence** par l'**expérience**, la pratique.
- Pour une machine :
  - **compétence** = algorithme
  - **expérience,** pratique = exemples, données
- Autres noms : apprentissage statistique ; machine learning.

Algorithme d'apprentissage

- **Algorithme classique :**

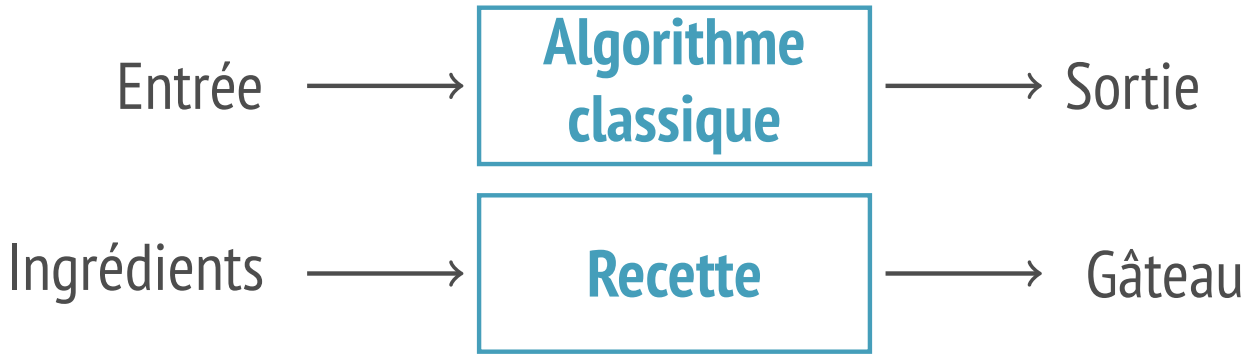
Algorithme d'apprentissage

- **Algorithme classique :**

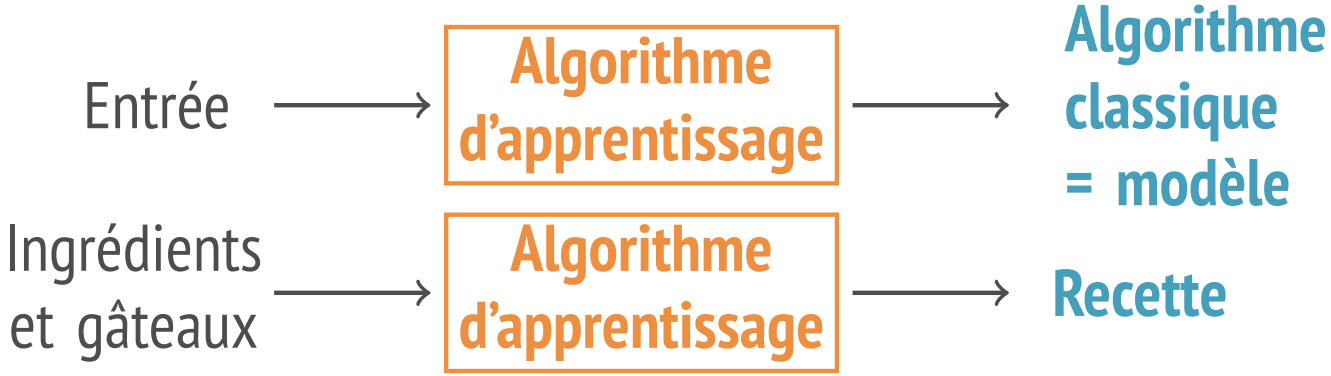


Algorithme d'apprentissage

- **Algorithme classique :**



- **Algorithme d'apprentissage :**



Deep learning, ML et IA

- La plupart des succès récents de l'« intelligence artificielle » sont en fait des progrès en **apprentissage automatique.**

  Il s'agit d'utiliser (beaucoup) d'exemples d'une tâche pour apprendre à faire cette tâche

- La plupart de ces progrès sont des progrès en **apprentissage profond** (deep learning) : un des domaines de l'apprentissage automatique qui utilise des **réseaux de neurones à plusieurs couches.**

Deep learning

- **Exemples**
  - **Traitement automatique d'images :** reconnaître si une photo contient un chat / est celle d'une personne spécifique / est un échantillon histopathologique d'une tumeur
  - **Traitement automatique du langage :** traduction automatique, chatbots
  - **Génération automatique** de texte, d'images, de son, de vidéos réalistes (en particulier deepfakes)
- **Ingrédients du succès :**
  - Énormes **quantités de données**
  - **Puissance de calcul**
  - Compréhension des données et problèmes
    $\Rightarrow$ modélisation (= **architecture** des réseaux de neurones) appropriée.

Autres exemples d'application du ML

- Détection de fraude
- Segmentation de marché
- Contrôle qualité
- Maintenance prédictive
- Systèmes de recommendation
- Trading algorithmique
- Assistance au diagnostic
- Médecine prédictive
- Et beaucoup, beaucoup d'autres.

Défis actuels du ML

- Apprendre à partir de **peu d'exemples**

  Un enfant ou un animal n'a pas besoins de milliers/millions d'exemples pour apprendre

- Résoudre des problèmes **que l'humain ne sait pas résoudre**

  Utilisation dans la recherche scientifique (biologie, santé, chimie, science des matériaux, astrophysique, etc.)

- **Confiance**

  validation formelle, explicabilité, robustesse aux attaques

- **Questions éthiques**

  biais algorithmiques ; coûts environementaux ; équité et moralité

1. Types de problèmes d'apprentissage automatique

1.1 Apprentissage supervisé

– **But : apprendre** un modèle **prédictif**

1.1 Apprentissage supervisé

- **But : apprendre** un modèle **prédictif**
- **Formalisation :**

  Données :
  - n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$
  - n étiquettes $\{y_1, y_2, \ldots, y_n\}$; $y_i \in \mathcal{Y}$

  Sortie :
  - **modèle** $f : \mathcal{X} \to \mathcal{Y}$ tel que $f(\vec{x}_i) \approx y_i$.

1.1 Apprentissage supervisé

- **But : apprendre** un modèle **prédictif**
- **Formalisation :**

  Données :
  - n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$
  - n étiquettes $\{y_1, y_2, \ldots, y_n\}$; $y_i \in \mathcal{Y}$
    - **Régression** : $\mathcal{Y} = \mathbb{R}$

  Sortie :
  - **modèle** $f : \mathcal{X} \to \mathcal{Y}$ tel que $f(\vec{x}_i) \approx y_i$.

1.1 Apprentissage supervisé

- **But : apprendre** un modèle **prédictif**
- **Formalisation :**

  Données :
  - n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$
  - n étiquettes $\{y_1, y_2, \ldots, y_n\}$; $y_i \in \mathcal{Y}$
    - **Régression** : $\mathcal{Y} = \mathbb{R}$
    - **Classification** : $\mathcal{Y} = \{0, 1, \ldots, C-1\}$

  Sortie :
  - **modèle** $f : \mathcal{X} \to \mathcal{Y}$ tel que $f(\vec{x}_i) \approx y_i$.

1.1 Apprentissage supervisé

- **But : apprendre** un modèle **prédictif**
- **Formalisation :**

  Données :
  - n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$
  - n étiquettes $\{y_1, y_2, \ldots, y_n\}$; $y_i \in \mathcal{Y}$
    - **Régression** : $\mathcal{Y} = \mathbb{R}$
    - **Classification** : $\mathcal{Y} = \{0, 1, \ldots, C-1\}$
    - **Classification binaire** : $\mathcal{Y} = \{0, 1\}$

  Sortie :
  - **modèle** $f : \mathcal{X} \to \mathcal{Y}$ tel que $f(\vec{x}_i) \approx y_i$.

1.1 Apprentissage supervisé

- **But : apprendre** un modèle **prédictif**
- **Formalisation :**

  Données :
  - n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$
  - n étiquettes $\{y_1, y_2, \ldots, y_n\}$; $y_i \in \mathcal{Y}$
    - **Régression** : $\mathcal{Y} = \mathbb{R}$
    - **Classification** : $\mathcal{Y} = \{0, 1, \ldots, C-1\}$
    - **Classification binaire** : $\mathcal{Y} = \{0, 1\}$

  Sortie :
  - **modèle** $f : \mathcal{X} \to \mathcal{Y}$ tel que $f(\vec{x}_i) \approx y_i$.
- **Inférence** : étant donné $\vec{x} \in \mathcal{X}$, prédire $\hat{y} = f(\vec{x})$.

Exemples

- Exemples de **problèmes de régression :** $\mathcal{Y} = \mathbb{R}$
  - **Prédiction de clics :** combien de personnes vont cliquer sur ce lien ?
  - Prédiction de la **solubilité d'une molécule** dans l'éthanol en mg/mL

Exemples

- Exemples de **problèmes de régression :** $\mathcal{Y} = \mathbb{R}$
  - **Prédiction de clics :** combien de personnes vont cliquer sur ce lien ?
  - Prédiction de la **solubilité d'une molécule** dans l'éthanol en mg/mL
- Exemples de **problèmes de classification binaire :** $\mathcal{Y} = \{0, 1\}$
  - **Filtrage de spam :** cet email est-il un spam ?
  - **Reconnaissance faciale :** est-ce JM Blanquer sur cette photo ?

Exemples

- Exemples de **problèmes de régression :** $\mathcal{Y} = \mathbb{R}$
  - **Prédiction de clics :** combien de personnes vont cliquer sur ce lien ?
  - Prédiction de la **solubilité d'une molécule** dans l'éthanol en mg/mL
- Exemples de **problèmes de classification binaire :** $\mathcal{Y} = \{0, 1\}$
  - **Filtrage de spam :** cet email est-il un spam ?
  - **Reconnaissance faciale :** est-ce JM Blanquer sur cette photo ?
- Exemples de **problèmes de classification multiclasse :**
  - **OCR**, reconnaissance de chiffres manuscrits : $\mathcal{Y} = \{0, 1, 2, \ldots, 9\}$
  - Reconnaissance de **plantes** ou de **chants d'oiseaux**

1.2 Apprentissage non supervisé

– **But : explorer** les données ; apprendre une nouvelle représentation

1.2 Apprentissage non supervisé

- **But : explorer** les données ; apprendre une nouvelle représentation
- **Formalisation :**

  Données :

  n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$

  Sortie :

1.2 Apprentissage non supervisé

- **But : explorer** les données ; apprendre une nouvelle représentation
- **Formalisation :**

  Données :

  n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$

  Sortie :

  - **Réduction de dimension :** $f : \mathbb{R}^p \to \mathbb{R}^d$ avec $d \ll p$ et de telle sorte à ce que $f(\vec{x})$ soit une **représentation informative** de $\vec{x}$

1.2 Apprentissage non supervisé

- **But : explorer** les données ; apprendre une nouvelle représentation
- **Formalisation :**

  Données :

  n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$

  Sortie :

  - **Réduction de dimension :** $f : \mathbb{R}^p \to \mathbb{R}^d$ avec $d \ll p$ et de telle sorte à ce que $f(\vec{x})$ soit une **représentation informative** de $\vec{x}$
  - **Clustering :** partition de $\{1, 2, \ldots, n\}$ en ensembles d'éléments semblables, appelés **clusters**

1.2 Apprentissage non supervisé

- **But : explorer** les données ; apprendre une nouvelle représentation
- **Formalisation :**

  Données :
  n observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$; $\vec{x}_i \in \mathcal{X} = \mathbb{R}^p$

  Sortie :
  - **Réduction de dimension :** $f : \mathbb{R}^p \to \mathbb{R}^d$ avec $d \ll p$ et de telle sorte à ce que $f(\vec{x})$ soit une **représentation informative** de $\vec{x}$
  - **Clustering :** partition de $\{1, 2, \ldots, n\}$ en ensembles d'éléments semblables, appelés **clusters**
  - **Estimation de densité :** la loi de probabilité de X dont $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$ est supposé être un échantillon.

Réduction de dimension

Apprendre $f : \mathbb{R}^p \rightarrow \mathbb{R}^d$ avec $d \ll p$ et de telle sorte à ce que $f(\vec{x})$ soit une **représentation informative** de $\vec{x}$

- **Utilisation :**
  - Visualiser les données (en particulier $d = 2$)
  - Réduire la taille des données en mémoire
  - Améliorer la performance d'un algorithme supervisé

Clustering

Apprendre une partition de $\{1, 2, \ldots, n\}$ en ensembles d'éléments semblables, appelés **clusters**

- **Exemples d'application :**
  - **Segmentation de marché :** regrouper des clients qui ont des profils similaires
  - **Topic modeling:** regrouper les documents d'un corpus par thème ; les thèmes émergent de l'analyse et ne sont pas fournis.

1.3 Autres problèmes d'apprentissage

- **Apprentissage semi-supervisé**

  Une partie seulement des données est étiquetée

- **Apprentissage par renforcement**

  Définir une **politique** permettant de maximiser sa **récompense**

- **Régression structurée**

  Apprentissage supervisé avec $\mathcal{X}$ un espace de vecteurs, de séquences, de texte, d'images, etc.

2. Premier exemple : les plus proches voisins

2.1 Algorithme du plus proche voisin

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes : $y_i \in \mathcal{Y}$

2.1 Algorithme du plus proche voisin

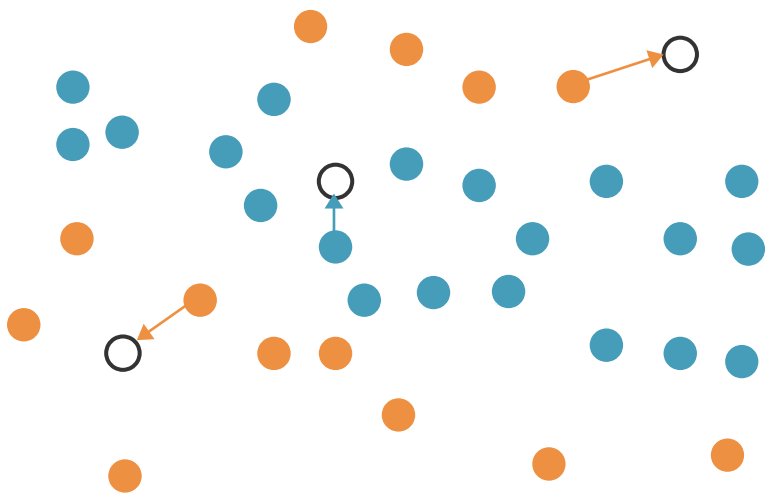
- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes : $y_i \in \mathcal{Y}$
- **Modèle :** Associer à $\vec{x} \in \mathbb{R}^p$ l'étiquette de l'exemple dans $\mathcal{D}$ dont $\vec{x}$ est le plus proche selon la distance euclidienne

$$f(\vec{x}) = y_{\arg\min_{i=1,\ldots,n} ||\vec{x} - \vec{x}_i||_2}$$

2.1 Algorithme du plus proche voisin

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes : $y_i \in \mathcal{Y}$
- **Modèle :** Associer à $\vec{x} \in \mathbb{R}^p$ l'étiquette de l'exemple dans $\mathcal{D}$ dont $\vec{x}$ est le plus proche selon la distance euclidienne

$$f(\vec{x}) = y_{\arg\min_{i=1,\ldots,n} ||\vec{x} - \vec{x}_i||_2}$$



2.1 Algorithme du plus proche voisin

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes : $y_i \in \mathcal{Y}$
- **Modèle :** Associer à $\vec{x} \in \mathbb{R}^p$ l'étiquette de l'exemple dans $\mathcal{D}$ dont $\vec{x}$ est le plus proche selon la distance euclidienne

$$f(\vec{x}) = y_{\arg\min_{i=1,\ldots,n} ||\vec{x} - \vec{x}_i||_2}$$

- Régression, classification binaire, classification multiclasse.

2.1 Algorithme du plus proche voisin

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes : $y_i \in \mathcal{Y}$
- **Modèle :** Associer à $\vec{x} \in \mathbb{R}^p$ l'étiquette de l'exemple dans $\mathcal{D}$ dont $\vec{x}$ est le plus proche selon la distance euclidienne

$$f(\vec{x}) = y_{\arg\min_{i=1,\ldots,n} ||\vec{x} - \vec{x}_i||_2}$$

- Régression, classification binaire, classification multiclasse.
- **Pas de phase d'apprentissage !**

2.2 kNN pour la classification

- L'algorithme du plus proche voisin est **peu robuste** au bruit
- Plutôt que d'utiliser **le** plus proche voisin : utiliser les k observations les plus proches
- **Vote de la majorité :** on donne à $\vec{x}$ **l'étiquette majoritaire** parmi ses k plus proches voisins

2.2 kNN pour la classification

- L'algorithme du plus proche voisin est **peu robuste** au bruit
- Plutôt que d'utiliser **le** plus proche voisin : utiliser les k observations les plus proches
- **Vote de la majorité :** on donne à $\vec{x}$ **l'étiquette majoritaire** parmi ses k plus proches voisins

$$f(\vec{x}) = \underset{c \in \{0,1,\ldots,C-1\}}{\arg\max} \sum_{i:\vec{x}_i \in \mathcal{N}_k(\vec{x})} \delta(y_i, c)$$

- $\mathcal{N}_k(\vec{x})$ est l'ensemble des k plus proches voisins de $\vec{x}$ dans $\mathcal{D}$
- $\delta(y_i, c) = 1$ si $y_i = c$ et 0 sinon; C est le nombre de classes.

2.2 kNN pour la classification

- L'algorithme du plus proche voisin est **peu robuste** au bruit
- Plutôt que d'utiliser **le** plus proche voisin : utiliser les k observations les plus proches
- **Vote de la majorité :** on donne à $\vec{x}$ **l'étiquette majoritaire** parmi ses k plus proches voisins

$$f(\vec{x}) = \underset{c \in \{0,1,\ldots,C-1\}}{\arg\max} \sum_{i:\vec{x}_i \in \mathcal{N}_k(\vec{x})} \delta(y_i, c)$$

- $\mathcal{N}_k(\vec{x})$ est l'ensemble des k plus proches voisins de $\vec{x}$ dans $\mathcal{D}$
- $\delta(y_i, c) = 1$ si $y_i = c$ et 0 sinon; C est le nombre de classes.

- Pour la classification binaire, on utilise un nombre **impair** de voisins pour ne pas avoir d'ex-aequo.

2.3 kNN pour la régression

– **Moyenne :** on donne à $\vec{x}$ **la moyenne des étiquettes** de ses k plus proches voisins

$$f(\vec{x}) = \frac{1}{k} \sum_{i:\vec{x}_i \in \mathcal{N}_k(\vec{x})} y_i$$

2.3 kNN pour la régression

- **Moyenne :** on donne à $\vec{x}$ **la moyenne des étiquettes** de ses k plus proches voisins

$$f(\vec{x}) = \frac{1}{k} \sum_{i:\vec{x}_i \in \mathcal{N}_k(\vec{x})} y_i$$

- **Remarque :** Les modèles donnés par des algorithmes de plus proches voisins sont des modèles **non-paramétriques :**
  - il ne s'agit pas d'apprendre les paramètres d'une expression explicite des variables $x_1, \ldots, x_p$
  - par opposition aux **modèles linéaires** et aux **réseaux de neurones**
  - k est un **hyperparamètre** : l'algorithme ne permet pas de l'apprendre.

2.4 Variantes

- **Epsilon-voisins** Au lieu de considérer k voisins les plus proches, on considère toutes les observations contenues dans une boule de rayon ϵ centrée sur $\vec{x}$.
- **Pondérations des voisins** La contribution de chaque voisin est pondérée par un coefficient inversement proportionnel à sa distance à $\vec{x}$.
- **Autres distances** L'algorithme s'applique aussi en considérant d'autres distances / notions de similarité :
  - Distance de Minkowski
  - Similarité cosinus
  - Distance de Hamming entre vecteurs binaires

Cours 1 – Minimisation du risque empirique

Minimisation du risque empirique

- Principe sur lequel de nombreux algorithmes **d'apprentissage supervisé** sont fondés.
- Ingrédients :
  - **Espace des hypothèses** (hypothesis space)
    Quels modèles peut-on apprendre ?
  - **Fonction de perte / coût / erreur** (loss / cost / error function)
    Comment quantifier l'erreur d'un modèle ?
  - Algorithme d'**optimisation**
    Comment trouver un modèle qui fasse peu d'erreurs ?

1. Minimisation du risque empirique

1.1 Espace des hypothèses

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes (labels) : $y_i \in \mathcal{Y}$

1.1 Espace des hypothèses

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes (labels) : $y_i \in \mathcal{Y}$
- **Espace des hypothèses :** l'ensemble $\mathcal{F}$ des **modèles** que l'algorithme d'apprentissage va considérer.

1.1 Espace des hypothèses

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations en p dimensions : $\vec{x}_i \in \mathbb{R}^p$
  - n étiquettes (labels) : $y_i \in \mathcal{Y}$
- **Espace des hypothèses :** l'ensemble $\mathcal{F}$ des **modèles** que l'algorithme d'apprentissage va considérer.
- **Modèles paramétriques :** on considère un ensemble paramétré de fonctions.

  Exemple : $\mathcal{F} = \{\vec{x} \mapsto \alpha_1 x_1^{\beta_1} + \alpha_2 x_2^{\beta_2} + \cdots + \alpha_p x_p^{\beta_p}\}$
  Le but de l'apprentissage est de déterminer $\alpha_1, \ldots, \alpha_p, \beta_1, \ldots, \beta_p$.

1.2 Fonction de perte

- **Quantifier** l'erreur d'un modèle pour une observation.
- **Fonction de perte :** une fonction $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$

  $L(y, f(\vec{x}))$ est d'autant plus grande que l'erreur consistant à prédire $f(\vec{x})$ au lieu de y est grande.

1.2 Fonction de perte

- **Quantifier** l'erreur d'un modèle pour une observation.
- **Fonction de perte :** une fonction $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$

  $L(y, f(\vec{x}))$ est d'autant plus grande que l'erreur consistant à prédire $f(\vec{x})$ au lieu de y est grande.
  - **Perte 0/1** (0/1 loss) pour la classification :

$$L : \quad \{0,1\} \times \{0,1\} \to \mathbb{R}$$

$$(y, f(\vec{x})) \mapsto \begin{cases} 0 & \text{si } f(\vec{x}) = y \\ 1 & \text{sinon.} \end{cases}$$

1.2 Fonction de perte

- **Quantifier** l'erreur d'un modèle pour une observation.
- **Fonction de perte :** une fonction $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$

  $L(y, f(\vec{x}))$ est d'autant plus grande que l'erreur consistant à prédire $f(\vec{x})$ au lieu de y est grande.
  - **Perte 0/1** (0/1 loss) pour la classification :

$$L : \quad \{0,1\} \times \{0,1\} \to \mathbb{R}$$
$$(y, f(\vec{x})) \mapsto \begin{cases} 0 & \text{si } f(\vec{x}) = y \\ 1 & \text{sinon.} \end{cases}$$

  - **Perte quadratique** (quadratic loss) pour la régression :

$$L : \quad \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$(y, f(\vec{x})) \mapsto (y - f(\vec{x}))^2$$

1.3 Risque empirique

- **Risque :**
  - Erreur attendue du modèle sur toutes données possibles

Formellement : l'espérance de la fonction de perte

$$\mathcal{R}(f) = \mathbb{E}[L(Y, f(X)]$$

- $(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)$ échantillon de (X, Y)
- X vecteur aléatoire réel de dimension p ; Y variable aléatoire dans $\mathcal{Y}$

- **Risque empirique :** moyenne de la fonction de perte sur les données

$$\mathcal{R}_{\text{emp}}(f) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\vec{x}_i))$$

1.4 Minimisation du risque empirique

- **Apprendre un modèle** = **trouver** un modèle **de l'espace des hypothèses** qui **minimise** le **risque empirique**

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\vec{x}_i))$$

- C'est donc un problème **d'optimisation**
- Selon l'espace des hypothèses, la fonction de perte, et la procédure d'optimisation choisie :
  - La solution peut être unique (mais pas nécessairement)
  - La solution peut être explicite (rarement)
  - La solution peut être atteinte avec la précision souhaitée (parfois)
  - La solution ne peut être déterminée que par des heuristiques

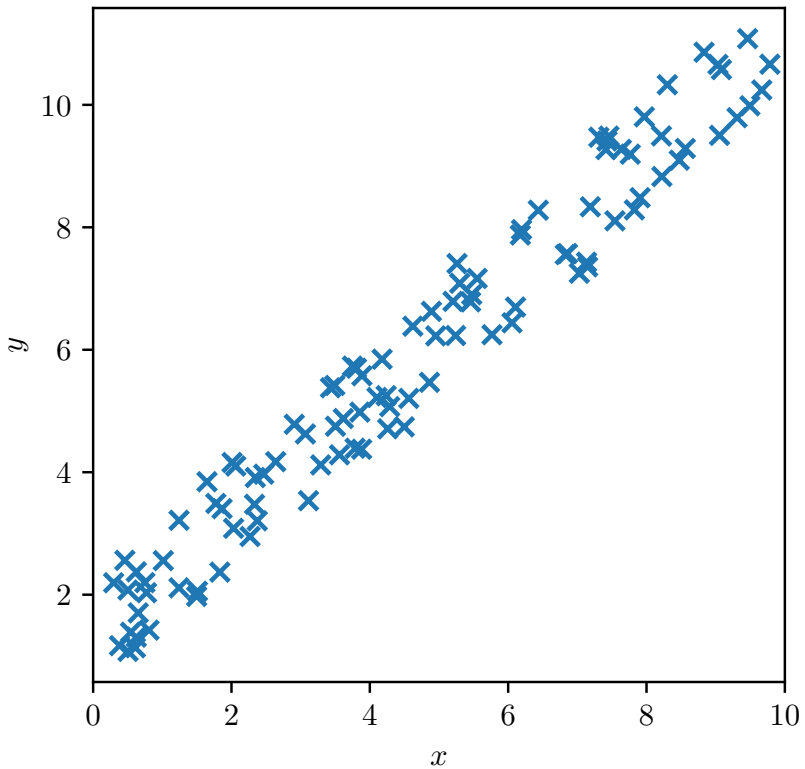
1.5 Maximisation de la vraisemblance

- Dans le cas d'un modèle de **régression paramétrique :**
  $\mathcal{F} = \{f_{\vec{\theta}} : \mathbb{R}^p \to \mathbb{R}; \vec{\theta} \in \mathbb{R}^d\}$
- En utilisant la **perte quadratique :** $L(y, f(\vec{x})) = (y - f(\vec{x}))^2$
- En supposant :
  - $(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)$ échantillon de (X, Y)
  - X vecteur aléatoire réel de dimension p ; Y variable aléatoire réelle
  - Il existe $f^* : \mathbb{R}^p \to \mathbb{R}$ telle que $Y = f^*(X) + \epsilon$
  - $\epsilon \sim \mathcal{N}(0, \sigma^2)$ (bruit gaussien)
- Alors : l'estimation par maximum de vraisemblance de $\vec{\theta}$ est **équivalente** à la minimisation du risque empirique

2. Modèles linéaires

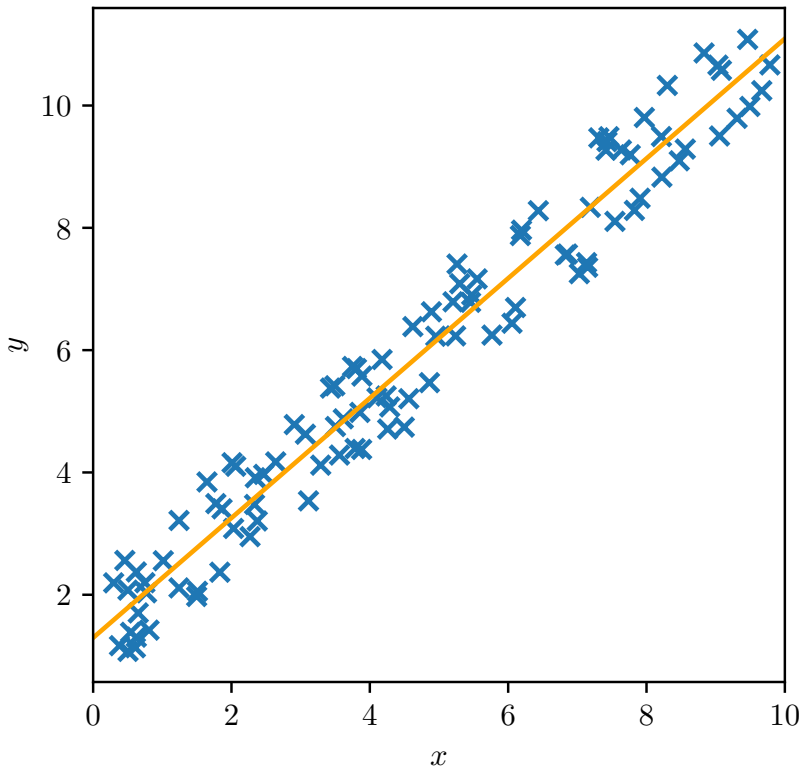
2.1 Régression linéaire univariée

- **Données :** $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
  - n observations dans $\mathbb{R}$: $x_i \in \mathbb{R}$ $(p = 1)$
  - n étiquettes : $y_i \in \mathbb{R}$



2.1 Régression linéaire univariée

- **Données :** $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
  - n observations dans $\mathbb{R}$: $x_i \in \mathbb{R}$ $(p = 1)$
  - n étiquettes : $y_i \in \mathbb{R}$



- **Espace des hypothèses :** $\mathcal{F} = \{x \mapsto ax + b; (a, b) \in \mathbb{R}^2\}$

2.1 Régression linéaire univariée

- **Espace des hypothèses :** $\mathcal{F} = \{x \mapsto ax + b; (a, b) \in \mathbb{R}^2\}$

2.1 Régression linéaire univariée

- **Espace des hypothèses :** $\mathcal{F} = \{x \mapsto ax + b; (a, b) \in \mathbb{R}^2\}$
- **Fonction de perte :** perte quadratique $L(y, f(x)) = (y - f(x))^2$

2.1 Régression linéaire univariée

- **Espace des hypothèses :** $\mathcal{F} = \{x \mapsto ax + b; (a, b) \in \mathbb{R}^2\}$
- **Fonction de perte :** perte quadratique $L(y, f(x)) = (y - f(x))^2$
- **Minimisation du risque empirique :**

$$(\hat{a}, \hat{b}) = \underset{(a,b)\in\mathbb{R}^2}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} (y_i - (ax_i + b))^2$$

2.1 Régression linéaire univariée

- **Espace des hypothèses :** $\mathcal{F} = \{x \mapsto ax + b; (a, b) \in \mathbb{R}^2\}$
- **Fonction de perte :** perte quadratique $L(y, f(x)) = (y - f(x))^2$
- **Minimisation du risque empirique :**

$$(\hat{a}, \hat{b}) = \underset{(a,b)\in\mathbb{R}^2}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} (y_i - (ax_i + b))^2$$

Méthode des moindres carrés (ordinary least squares) connue depuis Legendre et Gauss.

Solution

- **Par annulation des dérivées partielles :** système de deux équations à deux inconnues

  Solution explicite :

  - Si $\sum_{i=1}^{n} x_i^2 \neq \left(\sum_{i=1}^{n} x_i\right)^2$, une **unique solution**

$$a = \frac{\frac{1}{n}\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \frac{1}{n}\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \qquad b = \frac{1}{n}\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i$$

Solution

- **Par annulation des dérivées partielles :** système de deux équations à deux inconnues

  Solution explicite :

  - Si $\sum_{i=1}^{n} x_i^2 \neq \left(\sum_{i=1}^{n} x_i\right)^2$, une **unique solution**

$$a = \frac{\frac{1}{n}\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \frac{1}{n}\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \qquad b = \frac{1}{n}\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i$$

  - Sinon, le système est **indéterminé** et on a une infinité de solutions.

Solution

- **Par annulation des dérivées partielles :** système de deux équations à deux inconnues

  Solution explicite :

  - Si $\sum_{i=1}^{n} x_i^2 \neq \left(\sum_{i=1}^{n} x_i\right)^2$, une **unique solution**

$$a = \frac{\frac{1}{n}\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \frac{1}{n}\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \qquad b = \frac{1}{n}\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i$$

  - Sinon, le système est **indéterminé** et on a une infinité de solutions.

- **Par algorithme du gradient**

2.2 Régression linéaire multivariée

- Autre nom : **régression linéaire multiple**
- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations dans $\mathbb{R}^p$
  - n étiquettes : $y_i \in \mathbb{R}$
- **Espace des hypothèses :**

$$\mathcal{F} = \left\{ \vec{x} \mapsto \beta_0 + \sum_{j=1}^{p} \beta_j x_j \, ; \vec{\beta} \in \mathbb{R}^{p+1} \right\}$$

2.2 Régression linéaire multivariée

- Autre nom : **régression linéaire multiple**
- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations dans $\mathbb{R}^p$
  - n étiquettes : $y_i \in \mathbb{R}$
- **Espace des hypothèses :**

$$\mathcal{F} = \left\{ \vec{x} \mapsto \beta_0 + \sum_{j=1}^{p} \beta_j x_j \,;\, \vec{\beta} \in \mathbb{R}^{p+1} \right\}$$

- **Fonction de perte :** perte quadratique $L(y, f(\vec{x})) = (y - f(\vec{x}))^2$

2.2 Régression linéaire multivariée

- Autre nom : **régression linéaire multiple**
- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations dans $\mathbb{R}^p$
  - n étiquettes : $y_i \in \mathbb{R}$
- **Espace des hypothèses :**

$$\mathcal{F} = \left\{ \vec{x} \mapsto \beta_0 + \sum_{j=1}^{p} \beta_j x_j \,;\, \vec{\beta} \in \mathbb{R}^{p+1} \right\}$$

- **Fonction de perte :** perte quadratique $L(y, f(\vec{x})) = (y - f(\vec{x}))^2$
- **Minimisation du risque empirique :**

$$\hat{\vec{\beta}} = \underset{\vec{\beta} \in \mathbb{R}^{p+1}}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_j \right) \right)^2$$

Écriture matricielle

$$\hat{\vec{\beta}} = \underset{\vec{\beta}\in\mathbb{R}^{p+1}}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_j \right) \right)^2$$

- La transformation $\vec{x} \leftarrow (1, x_1, \ldots, x_p)$ permet d'écrire $f(\vec{x}) = \langle \vec{\beta}, \vec{x} \rangle$
- En notant $X \in \mathbb{R}^{n\times(p+1)}$ la matrice de données : $X_{i.} = \vec{x}_i$ et $\vec{y} \in \mathbb{R}^n$ le vecteur des étiquettes,

$$\sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_j \right) \right)^2 = \left(\vec{y} - X\vec{\beta} \right)^\top \left(\vec{y} - X\vec{\beta} \right)$$

- Le problème devient

$$\underset{\vec{\beta}\in\mathbb{R}^{p+1}}{\arg\min} \frac{1}{n} \left(\vec{y} - X\vec{\beta} \right)^\top \left(\vec{y} - X\vec{\beta} \right)$$

Solution

$$\text{Minimiser } J : \mathbb{R}^{p+1} \to \mathbb{R}$$

$$\vec{\beta} \mapsto \left(\vec{y} - X\vec{\beta}\right)^\top \left(\vec{y} - X\vec{\beta}\right)$$

– Problème **d'optimisation convexe**

Solution

$$\text{Minimiser } J : \mathbb{R}^{p+1} \to \mathbb{R}$$

$$\vec{\beta} \mapsto \left(\vec{y} - X\vec{\beta}\right)^\top \left(\vec{y} - X\vec{\beta}\right)$$

- Problème **d'optimisation convexe**
- **Gradient** de J : $\nabla J(\vec{\beta}) = -2X^\top(\vec{y} - X\vec{\beta})$
- En **annulant** ce gradient, on obtient

$$X^\top X\vec{\beta} = X^\top \vec{y}$$

Solution

$$\text{Minimiser } J : \mathbb{R}^{p+1} \to \mathbb{R}$$
$$\vec{\beta} \mapsto \left(\vec{y} - X\vec{\beta}\right)^\top \left(\vec{y} - X\vec{\beta}\right)$$

- Problème **d'optimisation convexe**
- **Gradient** de J : $\nabla J(\vec{\beta}) = -2X^\top(\vec{y} - X\vec{\beta})$
- En **annulant** ce gradient, on obtient

$$X^\top X\vec{\beta} = X^\top\vec{y}$$

- Solution :
  - Si $X^\top X$ **inversible**, alors $\vec{\beta}^* = \left(X^\top X\right)^{-1} X^\top\vec{y}$
  - Sinon, le système est indéterminé et on a une infinité de solutions.

Inversibilité de $X^\top X$

– $X^\top X$ est inversible ssi X est de rang colonne plein.

Inversibilité de $X^\top X$

- $X^\top X$ est inversible ssi X est de rang colonne plein.
  - Ce n'est pas le cas si $p + 1 > n$
    i.e. s'il y a **plus de variables que d'observations**
  - Ce n'est pas le cas si des colonnes sont colinéaires
    i.e. si des **variables sont fortement corrélées**

Inversibilité de $X^\top X$

- $X^\top X$ est inversible ssi X est de rang colonne plein.
  - Ce n'est pas le cas si $p + 1 > n$
    i.e. s'il y a **plus de variables que d'observations**
  - Ce n'est pas le cas si des colonnes sont colinéaires
    i.e. si des **variables sont fortement corrélées**
- L'inversion d'une matrice carrée de taille p est une opération en $\mathcal{O}(p^3)$

  On utilise généralement **l'algorithme du gradient** même si $X^\top X$ est inversible.

2.3 Régression logistique

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations dans $\mathbb{R}^p$
  - n étiquettes : $y_i \in \{0, 1\}$: **classification binaire**

2.3 Régression logistique

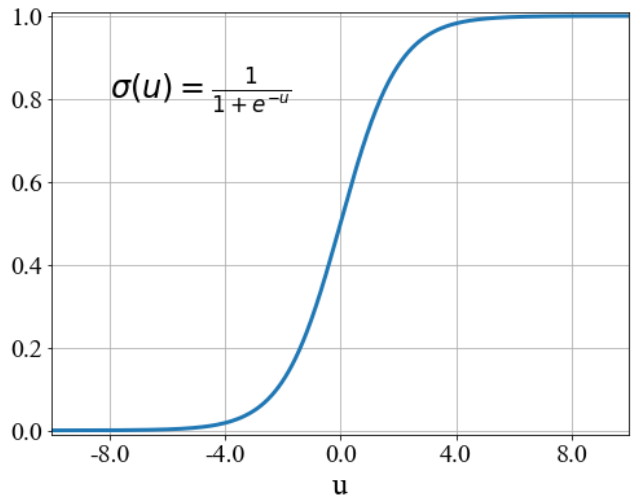
- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations dans $\mathbb{R}^p$
  - n étiquettes : $y_i \in \{0, 1\}$: **classification binaire**
- **Espace des hypothèses :** Considérer des fonctions linéaires n'a pas de sens pour prédire une valeur binaire.

2.3 Régression logistique

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations dans $\mathbb{R}^p$
  - n étiquettes : $y_i \in \{0, 1\}$: **classification binaire**
- **Espace des hypothèses :** Considérer des fonctions linéaires n'a pas de sens pour prédire une valeur binaire.
  - f va modéliser **la probabilité d'appartenir à la classe positive**

2.3 Régression logistique

- **Données :** $\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)\}$
  - n observations dans $\mathbb{R}^p$
  - n étiquettes : $y_i \in \{0, 1\}$: **classification binaire**
- **Espace des hypothèses :** Considérer des fonctions linéaires n'a pas de sens pour prédire une valeur binaire.
  - f va modéliser **la probabilité d'appartenir à la classe positive**
  - la **fonction logistique** σ transforme un réel en une "probabilité"

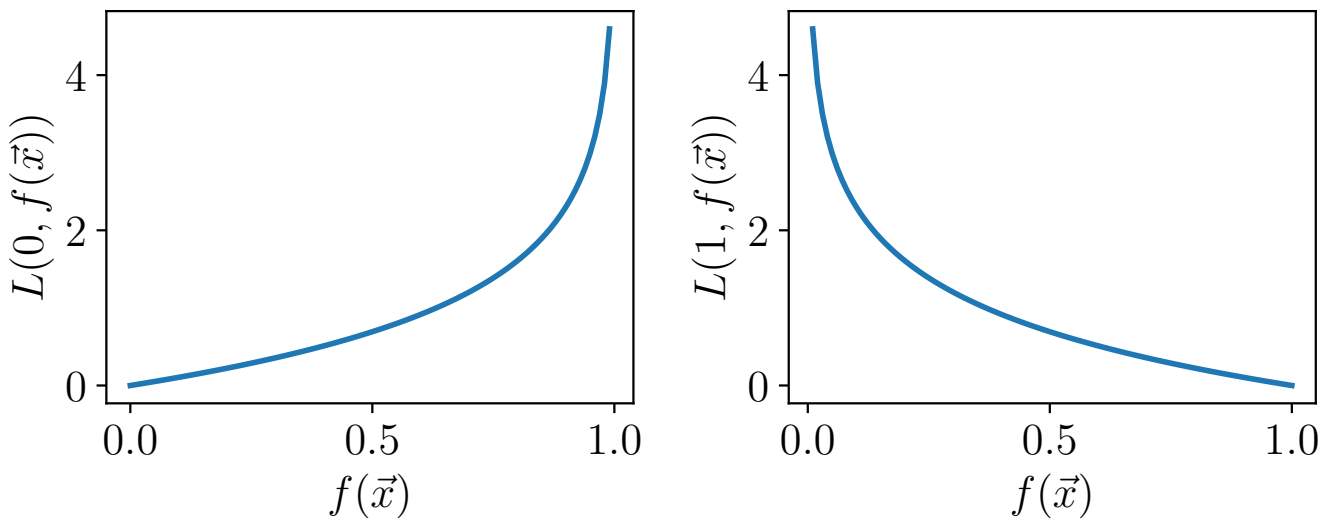


  - $\mathcal{F} = \left\{ \vec{x} \mapsto \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x} \rangle\right) ; \vec{\beta} \in \mathbb{R}^{p+1} \right\}$

2.3 Régression logistique

– Fonction de perte : **entropie croisée** (cross-entropy)

$$L : \{0,1\} \times]0,1[\to \mathbb{R}$$
$$(y, f(\vec{x})) \mapsto -y \log f(\vec{x}) - (1-y) \log(1 - f(\vec{x}))$$



2.3 Régression logistique

- **Espace des hypothèses :** $\mathcal{F} = \{\vec{x} \mapsto \sigma\left(\beta_0 + \langle\vec{\beta}, \vec{x}\rangle\right) ; \vec{\beta} \in \mathbb{R}^{p+1}\}$
- **Fonction de perte :**
  $L : (y, f(\vec{x})) \mapsto -y \log f(\vec{x}) - (1-y)\log(1 - f(\vec{x}))$
- **Minimisation du risque empirique :**

$$\arg\min_{\vec{\beta}\in\mathbb{R}^{p+1}} \frac{1}{n}\sum_{i=1}^{n} -y_i \log\sigma\left(\beta_0 + \langle\vec{\beta}, \vec{x}_i\rangle\right) - (1-y_i)\log\left(1 - \sigma\left(\beta_0 + \langle\vec{\beta}, \vec{x}_i\rangle\right)\right)$$

Solution

- **Minimiser**

$$J : \vec{\beta} \mapsto \sum_{i=1}^{n} -y_i \log \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right) - (1-y_i) \log\left(1 - \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right)\right)$$

Solution

- **Minimiser**

$$J : \vec{\beta} \mapsto \sum_{i=1}^{n} -y_i \log \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right) - (1-y_i) \log\left(1 - \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right)\right)$$

- J est une fonction **convexe** de $\vec{\beta}$

Solution

- **Minimiser**

$$J : \vec{\beta} \mapsto \sum_{i=1}^{n} -y_i \log \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right) - (1-y_i) \log\left(1 - \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right)\right)$$

- J est une fonction **convexe** de $\vec{\beta}$
- En utilisant $\sigma'(u) = \sigma(u)(1 - \sigma(u))$,

$$\nabla J(\vec{\beta}) = \sum_{i=1}^{n} \left(y_i - \frac{1}{1 + \exp - \left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right)} \right)$$

$\Rightarrow$ Pas de solution explicite.

Solution

- **Minimiser**

$$J : \vec{\beta} \mapsto \sum_{i=1}^{n} -y_i \log \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right) - (1-y_i) \log\left(1 - \sigma\left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right)\right)$$

- J est une fonction **convexe** de $\vec{\beta}$
- En utilisant $\sigma'(u) = \sigma(u)(1-\sigma(u))$,

$$\nabla J(\vec{\beta}) = \sum_{i=1}^{n} \left(y_i - \frac{1}{1 + \exp - \left(\beta_0 + \langle \vec{\beta}, \vec{x}_i \rangle\right)} \right)$$

$\Rightarrow$ Pas de solution explicite.

- On utilise donc systématiquement **l'algorithme du gradient.**

Interprétation

- Le **coefficient de régression** β_j peut être interprété comme la contribution de la variable correspondante (x_j) au modèle
- À condition que les variables soient **centrées-réduites** pour être à la même échelle :

$$x_j \leftarrow \frac{x_j - \overline{x_j}}{\sigma_j}$$

  - $\overline{x_j}$: moyenne de x_j sur les données
  - σ_j : écart-type de x_j sur les données

- Attention à calculer $\overline{x_j}$ et σ_j sur le jeu d'entraînement uniquement, pour ne pas toucher au jeu de test.

Régressions polynomiales