

Cours Machine Learning

Chapitre 01

Fondamentaux Mathématiques

Objectifs d'apprentissage :

- Maîtriser les concepts d'algèbre linéaire essentiels (vecteurs, matrices, décompositions)
- Comprendre les probabilités et statistiques nécessaires au ML
- Appréhender le calcul différentiel et les techniques d'optimisation
- Savoir appliquer ces outils mathématiques aux algorithmes de ML

Prérequis : Mathématiques niveau Licence (algèbre, analyse)

Durée estimée : 5-7 heures

Notebooks : 01_demo_*.ipynb

Table des matières

| | | |
|-----------|--|-----------|
| I | Algèbre Linéaire | 2 |
| 1 | Vecteurs et Espaces Vectoriels | 2 |
| 1.1 | Définitions Fondamentales | 2 |
| 1.2 | Opérations sur les Vecteurs | 2 |
| 1.3 | Produit Scalaire (Dot Product) | 3 |
| 1.4 | Norme Vectorielle | 3 |
| 1.5 | Distance entre Vecteurs | 4 |
| 2 | Matrices | 4 |
| 2.1 | Définitions et Notation | 4 |
| 2.2 | Opérations Matricielles | 5 |
| 2.3 | Multiplication Matricielle | 5 |
| 2.4 | Matrices Spéciales | 5 |
| 2.5 | Déterminant | 6 |
| 2.6 | Inverse de Matrice | 6 |
| 2.7 | Trace | 7 |
| 2.8 | Rang | 7 |
| 3 | Systèmes Linéaires et Résolution | 7 |
| 3.1 | Formulation | 7 |
| 3.2 | Solutions | 7 |
| 3.3 | Solution des Moindres Carrés | 8 |
| 4 | Valeurs et Vecteurs Propres | 8 |
| 4.1 | Définitions | 8 |
| 4.2 | Calcul des Valeurs Propres | 8 |
| 4.3 | Diagonalisation | 9 |
| 4.4 | Matrices Symétriques Réelles | 9 |
| 5 | Décompositions Matricielles | 10 |
| 5.1 | Décomposition en Valeurs Singulières (SVD) | 10 |
| 5.2 | Applications de la SVD | 10 |
| 5.3 | Autres Décompositions | 11 |
| II | Probabilités et Statistiques | 11 |
| 6 | Probabilités Fondamentales | 11 |
| 6.1 | Concepts de Base | 11 |
| 6.2 | Indépendance | 11 |
| 6.3 | Théorème de Bayes | 12 |
| 7 | Variables Aléatoires | 12 |
| 7.1 | Variable Aléatoire Discrète | 12 |

| | | |
|------------|---|-----------|
| 7.2 | Variable Aléatoire Continue | 13 |
| 7.3 | Espérance et Variance | 13 |
| 7.4 | Loi Normale (Gaussienne) | 14 |
| 7.5 | Loi Normale Multivariée | 14 |
| 8 | Statistiques Descriptives | 15 |
| 8.1 | Mesures de Tendance Centrale | 15 |
| 8.2 | Mesures de Dispersion | 15 |
| 8.3 | Covariance et Corrélacion | 15 |
| III | Calcul Différentiel et Optimisation | 16 |
| 9 | Dérivées | 16 |
| 9.1 | Dérivée d'une Fonction Scalaire | 16 |
| 9.2 | Gradient | 17 |
| 9.3 | Matrice Hessienne | 17 |
| 9.4 | Jacobienne | 17 |
| 10 | Optimisation | 18 |
| 10.1 | Conditions d'Optimalité | 18 |
| 10.2 | Convexité | 18 |
| 10.3 | Descente de Gradient | 19 |
| 10.4 | Méthode de Newton | 19 |
| 11 | Résumé du Chapitre | 20 |
| 11.1 | Points Clés | 20 |
| 11.2 | Formules Essentielles | 20 |
| 12 | Exercices | 21 |
| 12.1 | Algèbre Linéaire | 21 |
| 12.2 | Probabilités et Statistiques | 21 |
| 12.3 | Calcul Différentiel et Optimisation | 21 |
| 13 | Pour Aller Plus Loin | 22 |
| 13.1 | Lectures Recommandées | 22 |
| 13.2 | Outils Pratiques | 22 |
| 13.3 | Prochaines Étapes | 22 |

Première partie

Algèbre Linéaire

1 Vecteurs et Espaces Vectoriels

1.1 Définitions Fondamentales

Définition : Vecteur

Un vecteur $\mathbf{v} \in \mathbb{R}^n$ est un tuple ordonné de n nombres réels :

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} \quad (1)$$

On le représente comme un vecteur colonne par défaut. Un vecteur ligne s'écrit \mathbf{v}^T .

Interprétations :

- **Géométrique** : Point ou flèche dans l'espace \mathbb{R}^n
- **ML** : Une instance de données (sample), un vecteur de features
- **Algébrique** : Élément d'un espace vectoriel

Exemple : Vecteur de features

Un appartement décrit par 3 features :

$$\mathbf{x} = \begin{pmatrix} 75 \\ 3 \\ 2010 \end{pmatrix} \quad \begin{array}{l} \text{(surface en m}^2\text{)} \\ \text{(nombre de pièces)} \\ \text{(année de construction)} \end{array} \quad (2)$$

1.2 Opérations sur les Vecteurs

Addition vectorielle :

$$\mathbf{u} + \mathbf{v} = \begin{pmatrix} u_1 + v_1 \\ u_2 + v_2 \\ \vdots \\ u_n + v_n \end{pmatrix} \quad (3)$$

Multiplication par un scalaire :

$$\alpha \mathbf{v} = \begin{pmatrix} \alpha v_1 \\ \alpha v_2 \\ \vdots \\ \alpha v_n \end{pmatrix} \quad (4)$$

1.3 Produit Scalaire (Dot Product)

Définition : Produit scalaire

Le produit scalaire de deux vecteurs $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ est :

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}^T \mathbf{v} = \sum_{i=1}^n u_i v_i = u_1 v_1 + u_2 v_2 + \dots + u_n v_n \quad (5)$$

Propriétés :

- Commutativité : $\mathbf{u} \cdot \mathbf{v} = \mathbf{v} \cdot \mathbf{u}$
- Linéarité : $(\alpha \mathbf{u} + \beta \mathbf{w}) \cdot \mathbf{v} = \alpha(\mathbf{u} \cdot \mathbf{v}) + \beta(\mathbf{w} \cdot \mathbf{v})$
- $\mathbf{v} \cdot \mathbf{v} \geq 0$, égalité ssi $\mathbf{v} = \mathbf{0}$

Interprétation géométrique :

$$\mathbf{u} \cdot \mathbf{v} = \|\mathbf{u}\| \|\mathbf{v}\| \cos(\theta) \quad (6)$$

où θ est l'angle entre \mathbf{u} et \mathbf{v} .

Astuce

En Machine Learning :

- Le produit scalaire mesure la **similarité** entre vecteurs
- Deux vecteurs orthogonaux ($\mathbf{u} \cdot \mathbf{v} = 0$) sont **non corrélés**
- Utilisé partout : régression linéaire ($\mathbf{w}^T \mathbf{x}$), réseaux de neurones, kernels...

1.4 Norme Vectorielle

Définition : Norme L^p

La norme L^p d'un vecteur $\mathbf{v} \in \mathbb{R}^n$ est :

$$\|\mathbf{v}\|_p = \left(\sum_{i=1}^n |v_i|^p \right)^{1/p} \quad (7)$$

Normes principales :

- Norme L^1 (Manhattan) :

$$\|\mathbf{v}\|_1 = \sum_{i=1}^n |v_i| \quad (8)$$

Utilisée pour la régularisation Lasso, robuste aux outliers.

- Norme L^2 (Euclidienne) :

$$\|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^n v_i^2} = \sqrt{\mathbf{v}^T \mathbf{v}} \quad (9)$$

La plus courante en ML. Utilisée pour mesurer des distances.

- Norme L^∞ (Maximum) :

$$\|\mathbf{v}\|_\infty = \max_i |v_i| \quad (10)$$

Exemple : Calcul de normes

Pour $\mathbf{v} = \begin{pmatrix} 3 \\ -4 \end{pmatrix}$:

$$\|\mathbf{v}\|_1 = |3| + |-4| = 7 \quad (11)$$

$$\|\mathbf{v}\|_2 = \sqrt{3^2 + (-4)^2} = \sqrt{25} = 5 \quad (12)$$

$$\|\mathbf{v}\|_\infty = \max(|3|, |-4|) = 4 \quad (13)$$

1.5 Distance entre Vecteurs**Définition : Distance euclidienne**

La distance entre deux points $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ est :

$$d(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2 = \sqrt{\sum_{i=1}^n (u_i - v_i)^2} \quad (14)$$

Applications en ML :

- K-Nearest Neighbors (KNN)
- Clustering (K-Means)
- Embeddings (mesurer similarité sémantique)

2 Matrices**2.1 Définitions et Notation****Définition : Matrice**

Une matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$ est un tableau rectangulaire de m lignes et n colonnes :

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \quad (15)$$

Notation :

- a_{ij} : élément ligne i , colonne j
- $\mathbf{A}_{i:}$: i -ème ligne (vecteur ligne)
- $\mathbf{A}_{:j}$: j -ème colonne (vecteur colonne)

Interprétation ML :

- **Dataset** : $\mathbf{X} \in \mathbb{R}^{n \times d}$ où n = nombre d'instances, d = nombre de features
- Chaque ligne $\mathbf{X}_{i:} = \mathbf{x}_i^T$ est une instance
- Chaque colonne $\mathbf{X}_{:j}$ est une feature

2.2 Opérations Matricielles

Addition : $(\mathbf{A} + \mathbf{B})_{ij} = a_{ij} + b_{ij}$ (même dimension)

Multiplication par scalaire : $(\alpha \mathbf{A})_{ij} = \alpha a_{ij}$

Transposée :

$$(\mathbf{A}^T)_{ij} = a_{ji} \quad (16)$$

Propriétés de la transposée :

- $(\mathbf{A}^T)^T = \mathbf{A}$
- $(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$
- $(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T$

2.3 Multiplication Matricielle

Définition : Produit matriciel

Si $\mathbf{A} \in \mathbb{R}^{m \times n}$ et $\mathbf{B} \in \mathbb{R}^{n \times p}$, alors $\mathbf{C} = \mathbf{AB} \in \mathbb{R}^{m \times p}$ avec :

$$c_{ij} = \sum_{k=1}^n a_{ik} b_{kj} = \mathbf{A}_{i:} \cdot \mathbf{B}_{:,j} \quad (17)$$

Attention

Attention :

- La multiplication matricielle n'est **pas commutative** : $\mathbf{AB} \neq \mathbf{BA}$ en général
- Les dimensions doivent être compatibles : $(m \times n) \times (n \times p) \rightarrow (m \times p)$

Complexité : Multiplication naïve : $O(mnp)$ opérations

Exemple : Prédiction en régression linéaire

Pour n instances avec d features, la prédiction vectorielle s'écrit :

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w} + b\mathbf{1} \quad (18)$$

où $\mathbf{X} \in \mathbb{R}^{n \times d}$, $\mathbf{w} \in \mathbb{R}^d$, $\hat{\mathbf{y}} \in \mathbb{R}^n$. Une seule opération matricielle remplace n produits scalaires !

2.4 Matrices Spéciales

Matrice identité :

$$\mathbf{I}_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}, \quad \mathbf{AI} = \mathbf{IA} = \mathbf{A} \quad (19)$$

Matrice diagonale :

$$\mathbf{D} = \begin{pmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_n \end{pmatrix} \quad (20)$$

Matrice symétrique : $\mathbf{A} = \mathbf{A}^T$ (ex : matrices de covariance)

Matrice orthogonale : $\mathbf{Q}^T \mathbf{Q} = \mathbf{Q} \mathbf{Q}^T = \mathbf{I}$

2.5 Déterminant

Définition : Déterminant

Pour une matrice carrée $\mathbf{A} \in \mathbb{R}^{n \times n}$, le déterminant $\det(\mathbf{A})$ mesure le volume orienté de la transformation linéaire associée.

Cas 2×2 :

$$\det \begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc \quad (21)$$

Propriétés :

- $\det(\mathbf{AB}) = \det(\mathbf{A}) \det(\mathbf{B})$
- $\det(\mathbf{A}^T) = \det(\mathbf{A})$
- $\det(\mathbf{A}^{-1}) = 1/\det(\mathbf{A})$
- Si $\det(\mathbf{A}) = 0$, alors \mathbf{A} n'est pas inversible (singulière)

2.6 Inverse de Matrice

Définition : Matrice inverse

Pour une matrice carrée $\mathbf{A} \in \mathbb{R}^{n \times n}$, l'inverse \mathbf{A}^{-1} (si elle existe) vérifie :

$$\mathbf{A} \mathbf{A}^{-1} = \mathbf{A}^{-1} \mathbf{A} = \mathbf{I} \quad (22)$$

Conditions d'existence :

- \mathbf{A} doit être carrée
- $\det(\mathbf{A}) \neq 0$ (matrice non-singulière)

Cas 2×2 :

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \quad (23)$$

Attention

En pratique, on **n'inverse jamais explicitement** une matrice ! On résout plutôt le système linéaire $\mathbf{Ax} = \mathbf{b}$ via des méthodes numériques (décomposition LU, Cholesky, etc.).

2.7 Trace

Définition : Trace

La trace d'une matrice carrée $\mathbf{A} \in \mathbb{R}^{n \times n}$ est la somme de ses éléments diagonaux :

$$\text{tr}(\mathbf{A}) = \sum_{i=1}^n a_{ii} \quad (24)$$

Propriétés :

- $\text{tr}(\mathbf{A} + \mathbf{B}) = \text{tr}(\mathbf{A}) + \text{tr}(\mathbf{B})$
- $\text{tr}(\mathbf{AB}) = \text{tr}(\mathbf{BA})$ (propriété cyclique)
- $\text{tr}(\mathbf{A}) = \sum_i \lambda_i$ où λ_i sont les valeurs propres

2.8 Rang

Définition : Rang

Le rang d'une matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$ est le nombre maximal de colonnes (ou lignes) linéairement indépendantes.

Propriétés :

- $\text{rank}(\mathbf{A}) \leq \min(m, n)$
- $\text{rank}(\mathbf{A}) = \text{rank}(\mathbf{A}^T)$
- Si $\text{rank}(\mathbf{A}) = n$ (rang plein en colonnes), alors $\mathbf{A}^T \mathbf{A}$ est inversible

En ML :

- Rang faible \Rightarrow redondance dans les features
- PCA, SVD exploitent les structures de rang faible

3 Systèmes Linéaires et Résolution

3.1 Formulation

Un système linéaire de m équations à n inconnues s'écrit :

$$\mathbf{Ax} = \mathbf{b} \quad (25)$$

où $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$.

Cas en ML :

- **Régression linéaire** : Trouver \mathbf{w} tel que $\mathbf{Xw} \approx \mathbf{y}$
- **Optimisation** : Résoudre $\nabla f(\mathbf{x}) = \mathbf{0}$ (gradient nul)

3.2 Solutions

Cas carré ($m = n$) :

- Si $\det(\mathbf{A}) \neq 0$: solution unique $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$
- Si $\det(\mathbf{A}) = 0$: aucune solution ou infinité de solutions

Cas rectangulaire ($m \neq n$) :

- $m > n$ (surdéterminé) : en général pas de solution exacte \Rightarrow solution des moindres carrés
- $m < n$ (sous-déterminé) : infinité de solutions \Rightarrow choisir celle de norme minimale

3.3 Solution des Moindres Carrés

Théorème : Solution des moindres carrés

Pour le système surdéterminé $\mathbf{Ax} \approx \mathbf{b}$, la solution qui minimise $\|\mathbf{Ax} - \mathbf{b}\|_2^2$ est :

$$\mathbf{x}^* = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \quad (26)$$

sous réserve que $\mathbf{A}^T \mathbf{A}$ soit inversible.

Application directe : Régression linéaire ! Si $\mathbf{X} \in \mathbb{R}^{n \times d}$ et $\mathbf{y} \in \mathbb{R}^n$, alors :

$$\mathbf{w}^* = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (27)$$

Astuce

La matrice $\mathbf{A}^\dagger = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T$ s'appelle la **pseudo-inverse de Moore-Penrose**. En Python : `numpy.linalg.pinv(A)`.

4 Valeurs et Vecteurs Propres

4.1 Définitions

Définition : Valeur propre et vecteur propre

Pour une matrice carrée $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\lambda \in \mathbb{C}$ est une **valeur propre** et $\mathbf{v} \neq \mathbf{0}$ est un **vecteur propre** associé si :

$$\mathbf{Av} = \lambda \mathbf{v} \quad (28)$$

Interprétation : La transformation \mathbf{A} étire le vecteur \mathbf{v} par un facteur λ , sans changer sa direction.

4.2 Calcul des Valeurs Propres

Les valeurs propres sont les racines du **polynôme caractéristique** :

$$\det(\mathbf{A} - \lambda \mathbf{I}) = 0 \quad (29)$$

Exemple : Matrice 2×2

Pour $\mathbf{A} = \begin{pmatrix} 4 & 2 \\ 1 & 3 \end{pmatrix}$:

$$\det(\mathbf{A} - \lambda \mathbf{I}) = \det \begin{pmatrix} 4 - \lambda & 2 \\ 1 & 3 - \lambda \end{pmatrix} \quad (30)$$

$$= (4 - \lambda)(3 - \lambda) - 2 \times 1 \quad (31)$$

$$= \lambda^2 - 7\lambda + 10 = 0 \quad (32)$$

Solutions : $\lambda_1 = 5, \lambda_2 = 2$.

4.3 Diagonalisation**Théorème : Diagonalisation**

Si $\mathbf{A} \in \mathbb{R}^{n \times n}$ possède n vecteurs propres linéairement indépendants $\mathbf{v}_1, \dots, \mathbf{v}_n$ avec valeurs propres $\lambda_1, \dots, \lambda_n$, alors :

$$\mathbf{A} = \mathbf{P} \mathbf{D} \mathbf{P}^{-1} \quad (33)$$

où $\mathbf{P} = [\mathbf{v}_1 \ \mathbf{v}_2 \ \dots \ \mathbf{v}_n]$ et $\mathbf{D} = \text{diag}(\lambda_1, \dots, \lambda_n)$.

Avantages de la diagonalisation :

- Calcul de puissances : $\mathbf{A}^k = \mathbf{P} \mathbf{D}^k \mathbf{P}^{-1}$
- Exponentielle : $e^{\mathbf{A}} = \mathbf{P} e^{\mathbf{D}} \mathbf{P}^{-1}$
- Comprendre la dynamique des systèmes linéaires

4.4 Matrices Symétriques Réelles**Théorème : Théorème spectral**

Toute matrice symétrique réelle $\mathbf{A} \in \mathbb{R}^{n \times n}$ est diagonalisable avec :

- Toutes les valeurs propres sont **réelles**
- Les vecteurs propres peuvent être choisis **orthonormaux**
- Décomposition : $\mathbf{A} = \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T$ où \mathbf{Q} est orthogonale

Applications en ML :

- PCA (Principal Component Analysis)
- Matrices de covariance
- Kernel PCA

5 Décompositions Matricielles

5.1 Décomposition en Valeurs Singulières (SVD)

Théorème : SVD (Singular Value Decomposition)

Toute matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$ peut être décomposée en :

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad (34)$$

où :

- $\mathbf{U} \in \mathbb{R}^{m \times m}$: matrice orthogonale (vecteurs singuliers gauches)
- $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$: matrice diagonale avec valeurs singulières $\sigma_1 \geq \sigma_2 \geq \dots \geq 0$
- $\mathbf{V} \in \mathbb{R}^{n \times n}$: matrice orthogonale (vecteurs singuliers droits)

Propriétés :

- Les valeurs singulières sont les racines carrées des valeurs propres de $\mathbf{A}^T \mathbf{A}$
- $\text{rank}(\mathbf{A})$ = nombre de valeurs singulières non nulles
- Fonctionne pour **toute** matrice (pas besoin qu'elle soit carrée ou symétrique)

Astuce

SVD en Python :

```
1 import numpy as np
2 U, Sigma, VT = np.linalg.svd(A, full_matrices=False)
3 # Reconstruction: A_approx = U @ np.diag(Sigma) @ VT
```

5.2 Applications de la SVD

1. Approximation de rang faible :

On peut approximer \mathbf{A} en gardant seulement les k plus grandes valeurs singulières :

$$\mathbf{A}_k = \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{v}_i^T \quad (35)$$

C'est l'approximation de rang k optimale au sens de la norme de Frobenius.

2. PCA (Principal Component Analysis) :

La PCA est équivalente à une SVD des données centrées :

- Centrer : $\mathbf{X}_c = \mathbf{X} - \bar{\mathbf{x}}$
- SVD : $\mathbf{X}_c = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$
- Composantes principales = colonnes de \mathbf{V}

3. Compression d'images :

Une image est une matrice. La SVD permet de compresser en ne gardant que les k premières valeurs singulières.

4. Recommandation (Matrix Factorization) :

Netflix Prize : décomposer la matrice utilisateurs \times films en produit de matrices de rang faible.

5.3 Autres Décompositions

Décomposition LU : $\mathbf{A} = \mathbf{LU}$ (Lower-Upper), pour résoudre $\mathbf{Ax} = \mathbf{b}$ efficacement.

Décomposition de Cholesky : Si \mathbf{A} est symétrique définie positive, $\mathbf{A} = \mathbf{LL}^T$.

Décomposition QR : $\mathbf{A} = \mathbf{QR}$ où \mathbf{Q} est orthogonale et \mathbf{R} est triangulaire supérieure.

Deuxième partie

Probabilités et Statistiques

6 Probabilités Fondamentales

6.1 Concepts de Base

Définition : Probabilité

Une probabilité P est une mesure sur un espace d'événements Ω telle que :

- $0 \leq P(A) \leq 1$ pour tout événement A
- $P(\Omega) = 1$
- Si A_1, A_2, \dots sont disjoints, $P(\cup_i A_i) = \sum_i P(A_i)$

Règles de base :

$$P(A \cup B) = P(A) + P(B) - P(A \cap B) \quad (36)$$

$$P(A^c) = 1 - P(A) \quad (37)$$

$$P(A | B) = \frac{P(A \cap B)}{P(B)} \quad (\text{probabilité conditionnelle}) \quad (38)$$

6.2 Indépendance

Définition : Indépendance

Deux événements A et B sont indépendants si :

$$P(A \cap B) = P(A) \cdot P(B) \quad (39)$$

Équivalent à : $P(A | B) = P(A)$.

6.3 Théorème de Bayes

Théorème : Théorème de Bayes

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)} \quad (40)$$

Terminologie :

- $P(A)$: probabilité a priori
- $P(A | B)$: probabilité a posteriori
- $P(B | A)$: vraisemblance (likelihood)
- $P(B)$: évidence

Forme étendue :

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{\sum_i P(B | A_i) \cdot P(A_i)} \quad (41)$$

Astuce

Applications en ML :

- Naive Bayes Classifier
- Inférence bayésienne
- Filtres bayésiens (spam, Kalman)

7 Variables Aléatoires

7.1 Variable Aléatoire Discrète

Définition : Variable aléatoire discrète

Une variable aléatoire X prend des valeurs dans un ensemble discret (fini ou dénombrable). Sa distribution est caractérisée par la **fonction de masse** :

$$p_X(x) = P(X = x) \quad (42)$$

avec $\sum_x p_X(x) = 1$.

Distributions classiques :

- **Bernoulli** : $X \in \{0, 1\}$, $P(X = 1) = p$

$$p_X(x) = p^x(1 - p)^{1-x} \quad (43)$$

- **Binomiale** : $X \sim \text{Bin}(n, p)$ (nombre de succès en n essais)

$$p_X(k) = \binom{n}{k} p^k (1 - p)^{n-k} \quad (44)$$

- **Poisson** : $X \sim \text{Poisson}(\lambda)$ (événements rares)

$$p_X(k) = \frac{\lambda^k e^{-\lambda}}{k!} \quad (45)$$

7.2 Variable Aléatoire Continue

Définition : Variable aléatoire continue

Une variable continue X est caractérisée par une **fonction de densité** $f_X(x)$ telle que :

$$P(a \leq X \leq b) = \int_a^b f_X(x) dx \quad (46)$$

avec $\int_{-\infty}^{\infty} f_X(x) dx = 1$.

Distributions classiques :

— **Uniforme** : $X \sim \text{Uniform}(a, b)$

$$f_X(x) = \begin{cases} \frac{1}{b-a} & \text{si } a \leq x \leq b \\ 0 & \text{sinon} \end{cases} \quad (47)$$

— **Normale (Gaussienne)** : $X \sim \mathcal{N}(\mu, \sigma^2)$

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad (48)$$

— **Exponentielle** : $X \sim \text{Exp}(\lambda)$ (temps entre événements)

$$f_X(x) = \lambda e^{-\lambda x} \quad (x \geq 0) \quad (49)$$

7.3 Espérance et Variance

Définition : Espérance

L'espérance (moyenne) d'une variable aléatoire X est :

$$\mathbb{E}[X] = \begin{cases} \sum_x x \cdot p_X(x) & (\text{discret}) \\ \int_{-\infty}^{\infty} x \cdot f_X(x) dx & (\text{continu}) \end{cases} \quad (50)$$

Propriétés :

- Linéarité : $\mathbb{E}[aX + bY] = a\mathbb{E}[X] + b\mathbb{E}[Y]$
- $\mathbb{E}[X + c] = \mathbb{E}[X] + c$

Définition : Variance

La variance mesure la dispersion autour de la moyenne :

$$\text{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 \quad (51)$$

L'écart-type est $\sigma_X = \sqrt{\text{Var}(X)}$.

Propriétés :

- $\text{Var}(aX + b) = a^2 \text{Var}(X)$

— Si X, Y indépendants : $\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y)$

Exemples :

| Loi | Espérance | Variance |
|----------------------------|-----------------|----------------------|
| Bernoulli(p) | p | $p(1 - p)$ |
| Binomiale(n, p) | np | $np(1 - p)$ |
| Poisson(λ) | λ | λ |
| Uniforme(a, b) | $\frac{a+b}{2}$ | $\frac{(b-a)^2}{12}$ |
| Normale(μ, σ^2) | μ | σ^2 |

7.4 Loi Normale (Gaussienne)

La loi normale est **fondamentale** en ML et statistiques.

Théorème : Théorème Central Limite (TCL)

Soit X_1, \dots, X_n des variables i.i.d. d'espérance μ et variance σ^2 . Alors :

$$\frac{\bar{X}_n - \mu}{\sigma/\sqrt{n}} \xrightarrow{n \rightarrow \infty} \mathcal{N}(0, 1) \quad (52)$$

où $\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i$.

Conséquence : Beaucoup de phénomènes naturels suivent approximativement une loi normale grâce au TCL.

Propriétés de la loi normale :

- Somme de gaussiennes = gaussienne
- 68% des valeurs dans $[\mu - \sigma, \mu + \sigma]$
- 95% dans $[\mu - 2\sigma, \mu + 2\sigma]$
- 99.7% dans $[\mu - 3\sigma, \mu + 3\sigma]$

7.5 Loi Normale Multivariée

Définition : Loi normale multivariée

Un vecteur aléatoire $\mathbf{X} \in \mathbb{R}^d$ suit une loi normale multivariée $\mathcal{N}(\mu, \Sigma)$ si sa densité est :

$$f_{\mathbf{X}}(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp \left(-\frac{1}{2} (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu) \right) \quad (53)$$

où $\mu \in \mathbb{R}^d$ est le vecteur moyenne et $\Sigma \in \mathbb{R}^{d \times d}$ est la matrice de covariance.

Utilisations en ML :

- Gaussian Mixture Models (GMM)
- Analyse discriminante linéaire (LDA)
- Processus gaussiens

8 Statistiques Descriptives

8.1 Mesures de Tendance Centrale

Moyenne empirique :

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i \quad (54)$$

Médiane : Valeur centrale quand les données sont triées. Robuste aux outliers.

Mode : Valeur la plus fréquente.

8.2 Mesures de Dispersion

Variance empirique :

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2 \quad (55)$$

Écart-type : $s = \sqrt{s^2}$

Intervalle interquartile (IQR) : $IQR = Q_3 - Q_1$ (robuste)

8.3 Covariance et Corrélation

Définition : Covariance

La covariance entre deux variables X et Y mesure leur variation conjointe :

$$\text{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y] \quad (56)$$

Version empirique :

$$\text{cov}(x, y) = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}) \quad (57)$$

Définition : Coefficient de corrélation de Pearson

$$\rho_{XY} = \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y} \in [-1, 1] \quad (58)$$

Interprétation :

- $\rho = 1$: corrélation linéaire positive parfaite
- $\rho = -1$: corrélation linéaire négative parfaite
- $\rho = 0$: pas de corrélation linéaire

Matrice de covariance : Pour un vecteur aléatoire $\mathbf{X} = (X_1, \dots, X_d)^T$:

$$\Sigma = \mathbb{E}[(\mathbf{X} - \mathbb{E}[\mathbf{X}])(\mathbf{X} - \mathbb{E}[\mathbf{X}])^T] \quad (59)$$

avec $\Sigma_{ij} = \text{Cov}(X_i, X_j)$.

Astuce

En Python (NumPy/Pandas) :

```

1 # Matrice de covariance
2 cov_matrix = np.cov(X, rowvar=False) # colonnes = variables
3
4 # Matrice de corrélation
5 corr_matrix = np.corrcoef(X, rowvar=False)
6 # ou avec pandas:
7 corr_matrix = df.corr()

```

Troisième partie

Calcul Différentiel et Optimisation

9 Dérivées

9.1 Dérivée d'une Fonction Scalaire

Définition : Dérivée

La dérivée de $f : \mathbb{R} \rightarrow \mathbb{R}$ en x est :

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h} \quad (60)$$

Interprétation : Pente de la tangente à la courbe en x .

Règles de dérivation :

$$(cf)' = cf' \quad (61)$$

$$(f+g)' = f' + g' \quad (62)$$

$$(fg)' = f'g + fg' \quad (\text{règle du produit}) \quad (63)$$

$$\left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2} \quad (\text{règle du quotient}) \quad (64)$$

$$(f \circ g)' = (f' \circ g) \cdot g' \quad (\text{règle de la chaîne}) \quad (65)$$

Dérivées usuelles :

| $f(x)$ | $f'(x)$ |
|-----------|------------|
| x^n | nx^{n-1} |
| e^x | e^x |
| $\ln(x)$ | $1/x$ |
| $\sin(x)$ | $\cos(x)$ |
| $\cos(x)$ | $-\sin(x)$ |

9.2 Gradient

Définition : Gradient

Pour une fonction $f : \mathbb{R}^n \rightarrow \mathbb{R}$, le gradient est le vecteur des dérivées partielles :

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} \quad (66)$$

Interprétation géométrique :

- Le gradient pointe dans la direction de **plus forte croissance**
- Sa norme $\|\nabla f\|$ mesure le taux de croissance
- $-\nabla f$ pointe vers la plus forte décroissance

Exemple : Gradient de fonctions courantes

$$f(\mathbf{x}) = \mathbf{a}^T \mathbf{x} = \sum_i a_i x_i \quad \Rightarrow \quad \nabla f = \mathbf{a} \quad (67)$$

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{A} \mathbf{x} \quad \Rightarrow \quad \nabla f = (\mathbf{A} + \mathbf{A}^T) \mathbf{x} \quad (68)$$

$$f(\mathbf{x}) = \|\mathbf{x}\|_2^2 = \mathbf{x}^T \mathbf{x} \quad \Rightarrow \quad \nabla f = 2\mathbf{x} \quad (69)$$

9.3 Matrice Hessienne

Définition : Hessienne

Pour $f : \mathbb{R}^n \rightarrow \mathbb{R}$ deux fois différentiable, la matrice hessienne est la matrice des dérivées secondes :

$$\mathbf{H}_f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \quad (70)$$

Propriétés :

- Si f est C^2 , alors \mathbf{H} est symétrique (théorème de Schwarz)
- La hessienne mesure la **courbure** de f
- Utilisée dans les méthodes d'optimisation du second ordre (Newton)

9.4 Jacobienne

Définition : Jacobienne

Pour une fonction vectorielle $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, la jacobienne est la matrice des dérivées

partielles :

$$\mathbf{J}_f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \dots & \frac{\partial f_m}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{m \times n} \quad (71)$$

Application en Deep Learning : La backpropagation utilise la règle de la chaîne avec des jacobiniennes pour calculer les gradients.

10 Optimisation

10.1 Conditions d'Optimalité

Théorème : Condition nécessaire du premier ordre

Si \mathbf{x}^* est un minimum local de $f : \mathbb{R}^n \rightarrow \mathbb{R}$ différentiable, alors :

$$\nabla f(\mathbf{x}^*) = \mathbf{0} \quad (72)$$

Théorème : Condition suffisante du second ordre

Si $\nabla f(\mathbf{x}^*) = \mathbf{0}$ et la hessienne $\mathbf{H}_f(\mathbf{x}^*)$ est **définie positive** (toutes les valeurs propres > 0), alors \mathbf{x}^* est un minimum local strict.

Cas de la hessienne :

- Définie positive \Rightarrow minimum local
- Définie négative \Rightarrow maximum local
- Indéfinie \Rightarrow point-selle

10.2 Convexité

Définition : Fonction convexe

Une fonction $f : \mathbb{R}^n \rightarrow \mathbb{R}$ est convexe si pour tous \mathbf{x}, \mathbf{y} et $\lambda \in [0, 1]$:

$$f(\lambda \mathbf{x} + (1 - \lambda) \mathbf{y}) \leq \lambda f(\mathbf{x}) + (1 - \lambda) f(\mathbf{y}) \quad (73)$$

Critère différentiel :

- Si $\mathbf{H}_f(\mathbf{x})$ est semi-définie positive partout, alors f est convexe
- Si $\mathbf{H}_f(\mathbf{x})$ est définie positive partout, alors f est strictement convexe

Propriété fondamentale : Pour une fonction convexe, **tout minimum local est global.**

Exemple : Fonctions convexes courantes

- Fonctions linéaires/affines : $f(\mathbf{x}) = \mathbf{a}^T \mathbf{x} + b$
- Norme L^2 au carré : $f(\mathbf{x}) = \|\mathbf{x}\|_2^2$
- Exponentielle : $f(x) = e^x$
- Logarithme négatif : $f(x) = -\ln(x)$ pour $x > 0$

10.3 Descente de Gradient

Algorithm 1 Descente de Gradient

Require : Fonction f , point initial \mathbf{x}_0 , learning rate $\alpha > 0$, tolérance ϵ

Ensure : Minimum approximatif \mathbf{x}^*

```

1 :  $k \leftarrow 0$ 
2 : repeat
3 :   Calculer le gradient :  $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$ 
4 :   Mettre à jour :  $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \mathbf{g}_k$ 
5 :    $k \leftarrow k + 1$ 
6 : until  $\|\mathbf{g}_k\| < \epsilon$ 
7 : return  $\mathbf{x}_k$ 
  
```

Intuition : On se déplace itérativement dans la direction opposée au gradient (plus forte descente).

Paramètres :

- **Learning rate** α : Trop grand \Rightarrow divergence ; trop petit \Rightarrow convergence lente
- **Critère d'arrêt** : $\|\nabla f\| < \epsilon$ ou nombre max d'itérations

Astuce

Variantes modernes (Deep Learning) :

- SGD (Stochastic Gradient Descent)
- Momentum
- Adam, RMSprop, AdaGrad

Voir Chapitre 06 pour les détails.

10.4 Méthode de Newton

L'idée est d'utiliser l'information du second ordre (hessienne) :

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{H}_f(\mathbf{x}_k)^{-1} \nabla f(\mathbf{x}_k) \quad (74)$$

Avantages :

- Convergence quadratique (très rapide près du minimum)
- Pas besoin de tuner le learning rate

Inconvénients :

- Coût : calcul et inversion de la hessienne ($O(n^3)$)
- Ne fonctionne que si \mathbf{H} est définie positive
- Rarement utilisé en Deep Learning (trop coûteux)

Compromis : Méthodes quasi-Newton (L-BFGS) qui approximent la hessienne.

11 Résumé du Chapitre

11.1 Points Clés

Algèbre Linéaire :

- Vecteurs et matrices sont omniprésents en ML (données, poids, transformations)
- Produit scalaire \Rightarrow similarité ; norme \Rightarrow distance
- SVD = décomposition universelle (PCA, compression, recommandation)
- Systèmes linéaires \Rightarrow régression linéaire (moindres carrés)

Probabilités et Statistiques :

- Loi normale = fondamentale (TCL, modèles génératifs)
- Théorème de Bayes = base de l'inférence bayésienne
- Covariance/corrélation \Rightarrow dépendance entre variables
- Espérance, variance = résumant une distribution

Calcul Différentiel et Optimisation :

- Gradient = direction de plus forte croissance
- Descente de gradient = algorithme d'optimisation de base en ML
- Convexité \Rightarrow minimum local = global
- Hessienne = courbure (méthodes du second ordre)

11.2 Formules Essentielles

Formules à retenir

Algèbre linéaire :

$$\text{Produit scalaire : } \mathbf{u} \cdot \mathbf{v} = \sum_i u_i v_i \quad (75)$$

$$\text{Norme } L^2 : \|\mathbf{v}\|_2 = \sqrt{\sum_i v_i^2} \quad (76)$$

$$\text{Moindres carrés : } \mathbf{w}^* = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (77)$$

$$\text{SVD : } \mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T \quad (78)$$

Probabilités :

$$\text{Bayes : } P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (79)$$

$$\text{Loi normale : } \mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma^2)} \quad (80)$$

$$\text{Variance : } \text{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 \quad (81)$$

Optimisation :

$$\text{Gradient : } \nabla f = \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)^T \quad (82)$$

$$\text{Descente de gradient : } \mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \nabla f(\mathbf{x}_k) \quad (83)$$

12 Exercices

12.1 Algèbre Linéaire

1. Calculer le produit scalaire de $\mathbf{u} = (1, 2, 3)$ et $\mathbf{v} = (4, -1, 2)$. Les vecteurs sont-ils orthogonaux ?
2. Pour la matrice $\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$, calculer :
 - Les valeurs propres et vecteurs propres
 - La décomposition spectrale
3. Appliquer la SVD à la matrice $\mathbf{A} = \begin{pmatrix} 3 & 1 \\ 1 & 3 \\ 1 & 1 \end{pmatrix}$ (en Python). Reconstruire \mathbf{A} à partir de la SVD.
4. Résoudre le système linéaire au sens des moindres carrés :

$$\begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \approx \begin{pmatrix} 2 \\ 3 \\ 5 \end{pmatrix} \quad (84)$$

12.2 Probabilités et Statistiques

1. Un test médical détecte une maladie avec 99% de précision (vrai positif). La prévalence de la maladie est 0.1%. Si le test est positif, quelle est la probabilité d'être réellement malade ? (Bayes)
2. Pour $X \sim \mathcal{N}(10, 4)$, calculer $P(8 \leq X \leq 12)$.
3. Générer 1000 échantillons d'une loi normale $\mathcal{N}(5, 2)$ en Python. Calculer la moyenne et variance empiriques. Tracer l'histogramme.
4. Calculer la matrice de covariance pour les données :

$$\mathbf{X} = \begin{pmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 5 \end{pmatrix} \quad (85)$$

Interpréter la corrélation entre les deux variables.

12.3 Calcul Différentiel et Optimisation

1. Calculer le gradient de $f(\mathbf{x}) = x_1^2 + 2x_1x_2 + 3x_2^2$.
2. Montrer que $f(\mathbf{x}) = \|\mathbf{Ax} - \mathbf{b}\|_2^2$ est convexe. Calculer son gradient.
3. Implémenter la descente de gradient pour minimiser $f(x) = (x - 3)^2 + 5$ en partant de $x_0 = 0$. Tester différents learning rates.
4. Minimiser $f(x_1, x_2) = x_1^2 + 4x_2^2$ par descente de gradient. Visualiser la trajectoire.

Solutions détaillées dans 01_exercices.ipynb (solutions intégrées dans le notebook)

13 Pour Aller Plus Loin

13.1 Lectures Recommandées

Livres :

- *Linear Algebra and Its Applications* (4e éd., 2006) - Gilbert Strang
- *Probability and Statistics for Engineers* (9e éd., 2016) - Montgomery & Runger
- *Convex Optimization* (2004) - Boyd & Vandenberghe
- *Mathematics for Machine Learning* (2020) - Deisenroth, Faisal, Ong (gratuit en ligne)

Ressources en ligne :

- MIT OCW : Linear Algebra (18.06) - Gilbert Strang
- 3Blue1Brown : Essence of Linear Algebra (YouTube)
- Khan Academy : Probabilités et statistiques

13.2 Outils Pratiques

NumPy (calcul numérique) :

```
1 import numpy as np
2
3 # Algèbre linéaire
4 A = np.array([[1, 2], [3, 4]])
5 eigenvalues, eigenvectors = np.linalg.eig(A)
6 U, Sigma, VT = np.linalg.svd(A)
7
8 # Statistiques
9 mean = np.mean(data)
10 std = np.std(data)
11 cov_matrix = np.cov(X, rowvar=False)
```

SciPy (fonctions avancées) :

```
1 from scipy import stats, optimize
2
3 # Distributions
4 rv = stats.norm(loc=0, scale=1) # N(0,1)
5 pdf_values = rv.pdf(x)
6
7 # Optimisation
8 result = optimize.minimize(f, x0, method='BFGS')
```

13.3 Prochaines Étapes

Chapitre suivant : **Chapitre 02 - Métriques d'Évaluation**

Ces fondamentaux seront utilisés tout au long du cours :

- Régression linéaire (Ch. 03) : moindres carrés, gradient
- PCA (Ch. 05) : SVD, valeurs propres
- Réseaux de neurones (Ch. 06) : backpropagation, descente de gradient
- Probabilités : modèles génératifs, bayésiens

Références

1. Strang, G. (2006). *Linear Algebra and Its Applications* (4e éd.). Cengage Learning.
2. Deisenroth, M. P., Faisal, A. A., & Ong, C. S. (2020). *Mathematics for Machine Learning*. Cambridge University Press.
3. Boyd, S., & Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.
4. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
5. Murphy, K. P. (2022). *Probabilistic Machine Learning : An Introduction*. MIT Press.
6. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.