



Expert

# Le Machine Learning avec Python

## De la théorie à la pratique

Préface de **Patrick Albert**

Cofondateur d'ILOG et du HUB France IA

En téléchargement

 code source

 + QUIZ

Version en ligne  
**OFFERTE !**  
pendant 1 an

**Madjid KHICHANE**  
PhD en Intelligence Artificielle



Les éléments à télécharger sont disponibles à l'adresse suivante :  
**<http://www.editions-eni.fr>**  
Saisissez la référence de l'ouvrage **EIMLPYT** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

**Préface**

**Introduction générale**

**Avant-propos**

- 1. Pourquoi ce livre ? . . . . . 19
- 2. À qui s'adresse ce livre ? . . . . . 21
- 3. Comment est organisé ce livre ? . . . . . 23
- 4. Comment lire ce livre ? . . . . . 24
- 5. Quels sont les prérequis pour la lecture de ce livre ? . . . . . 25
- 6. Qui est l'auteur ? . . . . . 25
- 7. Remerciements . . . . . 27

**Partie 1 : La Data Science - Concepts généraux**

**Chapitre 1-1**

**La Data Science**

- 1. Objectif du chapitre . . . . . 29
- 2. L'objectif recherché en Machine Learning. . . . . 30
- 3. Une expérimentation Machine Learning . . . . . 34
  - 3.1 Types de données . . . . . 42
  - 3.2 Préparation des données. . . . . 44
- 4. Cycle de vie d'un projet Data Science . . . . . 47
- 5. Les algorithmes du Machine Learning. . . . . 50
- 6. Le problème de surapprentissage . . . . . 52

# 2 — Le Machine Learning avec Python

De la théorie à la pratique

7. Les paramètres et les hyperparamètres . . . . .	53
8. Validation croisée . . . . .	55
9. Données d'entraînement, de validation et de test . . . . .	59
10. Métriques de performance . . . . .	60
10.1 Métriques pour les problèmes de régression . . . . .	62
10.2 Métriques pour la classification. . . . .	65
10.2.1 Matrice de confusion binaire . . . . .	65
10.2.2 Matrice de confusion générale . . . . .	67
10.2.3 Exemple de matrice de confusion . . . . .	68
10.2.4 La courbe ROC . . . . .	70
10.3 Métriques pour le clustering . . . . .	71
11. Conclusion . . . . .	71

## Partie 2 : Outils techniques de la Data Science - Python, NumPy, Pandas et Jupyter

### Chapitre 2-2

#### Le langage Python

1. Objectif du chapitre . . . . .	73
2. Python en deux mots . . . . .	74
3. Installer l'interpréteur Python . . . . .	74
4. Les bases de la programmation Python. . . . .	77
4.1 Hello world avec Python . . . . .	77
4.1.1 La fonction print. . . . .	77
4.1.2 La fonction input . . . . .	81
4.2 Les structures de données . . . . .	81
4.2.1 Les variables numériques . . . . .	82
4.2.2 Les chaînes de caractères . . . . .	87
4.2.3 Le type booléen. . . . .	89
4.2.4 Les listes . . . . .	95
4.2.5 Les tuples. . . . .	98

4.2.6	Les dictionnaires . . . . .	99
4.2.7	Les ensembles . . . . .	101
4.2.8	Liste vs tuple vs dictionnaire vs ensemble . . . . .	105
4.3	Structurer un code Python. . . . .	106
4.3.1	L'indentation et les blocs de code . . . . .	106
4.3.2	Écrire une instruction sur plusieurs lignes . . . . .	107
4.3.3	Écrire plusieurs instructions sur une ligne . . . . .	109
4.3.4	Les commentaires en Python. . . . .	109
4.4	Les instructions conditionnelles . . . . .	109
4.4.1	Les conditions avec la structure if . . . . .	110
4.4.2	Les conditions avec la structure if-else . . . . .	111
4.4.3	Les conditions avec la structure if-elif-else . . . . .	112
4.5	Les boucles. . . . .	114
4.5.1	La boucle for . . . . .	114
4.5.2	La boucle for et la fonction zip . . . . .	119
4.5.3	La boucle while . . . . .	125
4.5.4	Contrôler les boucles avec break . . . . .	126
4.5.5	Contrôler les boucles avec continue . . . . .	127
4.6	Les fonctions. . . . .	128
4.6.1	Définir et utiliser une fonction sans paramètre . . . . .	129
4.6.2	Les fonctions avec paramètres . . . . .	131
4.6.3	Les valeurs par défaut des paramètres. . . . .	133
4.6.4	Renvoi de résultats . . . . .	136
4.6.5	La portée des variables . . . . .	137
4.6.6	Passage d'arguments à une fonction . . . . .	140
4.6.7	Les fonctions récursives . . . . .	143
4.7	Les listes en compréhension. . . . .	148
4.7.1	Les origines des listes en compréhension . . . . .	148
4.7.2	Construire une liste avec les listes en compréhension . .	149
4.7.3	Application de fonction avec une liste en compréhension. . . . .	150
4.7.4	Utiliser if-else avec les listes en compréhension . . . . .	151
4.7.5	Filtrer avec les listes en compréhension . . . . .	152

# 4 — Le Machine Learning avec Python

De la théorie à la pratique

4.8	Les expressions régulières . . . . .	152
4.8.1	Regex sans caractères spéciaux . . . . .	155
4.8.2	Regex avec caractères spéciaux . . . . .	157
4.8.3	Regex avec les multiplicateurs . . . . .	159
4.8.4	Regex avec un nombre d'occurrences limité . . . . .	162
4.8.5	Regex avec groupage des résultats . . . . .	163
4.8.6	Taille des motifs . . . . .	164
4.8.7	Aller plus loin avec les expressions régulières . . . . .	169
4.9	Gestion des exceptions . . . . .	170
4.9.1	La levée des exceptions . . . . .	170
4.9.2	Utiliser le bloc try-except . . . . .	172
4.9.3	Gérer plusieurs exceptions . . . . .	173
4.9.4	Utiliser la clause finally . . . . .	175
4.9.5	Utiliser la structure try-except-finally-else . . . . .	176
4.9.6	Lever une exception avec raise . . . . .	178
5.	Conclusion . . . . .	180

## Chapitre 2-2

### La bibliothèque NumPy

1.	Objectif du chapitre . . . . .	181
2.	NumPy en deux mots . . . . .	182
3.	Les tableaux NumPy . . . . .	182
3.1	Création de tableaux NumPy . . . . .	182
3.1.1	Créer un tableau à une dimension . . . . .	183
3.1.2	Créer un tableau à plusieurs dimensions . . . . .	183
3.2	Les dimensions d'un tableau NumPy . . . . .	185
3.3	Le type et la taille d'un tableau NumPy . . . . .	187
3.4	Fonction d'initialisation de tableaux NumPy . . . . .	189
4.	Accéder aux données d'un tableau NumPy . . . . .	191
4.1	Accès aux données d'un tableau à une dimension . . . . .	191
4.2	Accès aux données d'un tableau à deux dimensions . . . . .	193

- 4.3 Accès aux données d'un tableau à trois dimensions. . . . . 195
- 5. Modifier les données d'un tableau NumPy . . . . . 196
- 6. Copier un tableau NumPy dans un autre tableau NumPy . . . . . 197
- 7. Algèbre linéaire avec NumPy . . . . . 199
  - 7.1 Opérations mathématiques de base avec NumPy . . . . . 199
  - 7.2 Opérations sur les matrices avec NumPy . . . . . 201
- 8. Réorganiser des tableaux NumPy . . . . . 202
  - 8.1 Restructurer un tableau NumPy . . . . . 202
  - 8.2 Superposer des tableaux NumPy . . . . . 204
- 9. Statistiques descriptives avec NumPy. . . . . 205
- 10. Lire des données NumPy à partir d'un fichier. . . . . 207
- 11. Les masques booléens avec NumPy. . . . . 208
  - 11.1 Créer et utiliser un masque booléen . . . . . 208
  - 11.2 Un masque avec plusieurs conditions . . . . . 210
  - 11.3 Les fonctions numpy.any et numpy.all . . . . . 211
- 12. Tableaux NumPy versus listes Python . . . . . 213
  - 12.1 Comparaison des tailles en mémoire. . . . . 214
  - 12.2 Comparaison des temps de calcul . . . . . 215
    - 12.2.1 Temps de calcul sur une liste Python . . . . . 216
    - 12.2.2 Temps de calcul sur un tableau NumPy . . . . . 217
- 13. Conclusion . . . . . 218

**Chapitre 2-3**  
**La bibliothèque Pandas**

- 1. Objectif du chapitre . . . . . 219
- 2. C'est quoi, Pandas ? . . . . . 220
- 3. Installation de Pandas . . . . . 221

# 6 — Le Machine Learning avec Python

De la théorie à la pratique

4. DataFrame Pandas . . . . .	222
4.1 Création d'un DataFrame à partir d'un dictionnaire . . . . .	223
4.2 Création d'un DataFrame à partir d'un tableau NumPy . . . . .	225
4.3 Chargement des données à partir de fichiers . . . . .	226
4.3.1 Lecture des données d'un fichier CSV . . . . .	227
4.3.2 Lecture d'un fichier texte . . . . .	228
5. Accès aux données d'un DataFrame . . . . .	229
5.1 Lire les lignes d'un DataFrame . . . . .	230
5.1.1 Lire une ligne d'un DataFrame . . . . .	230
5.1.2 Lire plusieurs lignes d'un DataFrame . . . . .	230
5.1.3 Parcourir les lignes d'un DataFrame . . . . .	231
5.1.4 Filtrer les lignes avec une condition . . . . .	232
5.1.5 Filtrer les lignes avec plusieurs conditions . . . . .	232
5.1.6 Filtrage avec des critères textuels . . . . .	233
5.1.7 Réinitialiser les index . . . . .	234
5.1.8 Filtrer avec les valeurs uniques . . . . .	236
5.1.9 Filtrer avec une expression régulière . . . . .	236
5.2 Accéder aux variables d'un DataFrame . . . . .	237
5.2.1 Liste des variables d'un DataFrame . . . . .	237
5.2.2 Accès aux valeurs d'une colonne . . . . .	238
5.2.3 Accès à plusieurs colonnes . . . . .	238
5.3 Lire une cellule spécifique avec les index . . . . .	239
6. Modifier un DataFrame . . . . .	239
6.1 Modifier les valeurs dans un DataFrame . . . . .	239
6.2 Modifier la structure d'un DataFrame . . . . .	240
6.2.1 Ajouter une variable à un DataFrame . . . . .	240
6.2.2 Réordonner les variables d'un DataFrame . . . . .	243
6.2.3 Supprimer une variable d'un DataFrame . . . . .	244
6.2.4 Utiliser la méthode melt pour diminuer le nombre de variables . . . . .	245
6.3 Appliquer une fonction sur une variable avec la méthode apply . . . . .	247
6.4 Modification avec conditions . . . . .	249

6.5	Ajouter des lignes dans un DataFrame . . . . .	250
7.	Tri sur les données d'un DataFrame . . . . .	251
7.1	Tri avec un seul critère . . . . .	251
7.2	Tri avec plusieurs critères. . . . .	253
8.	Sauvegarder les données d'un DataFrame. . . . .	254
9.	Faire des statistiques sur un DataFrame . . . . .	255
9.1	Faire un résumé direct . . . . .	255
9.2	Faire un résumé par agrégation . . . . .	256
9.3	Agrégation avec plusieurs paramètres. . . . .	258
10.	Lecture des fichiers de grande taille. . . . .	259
11.	Conclusion . . . . .	261

## Chapitre 2-4

### Travailler avec Jupyter

1.	Objectif du chapitre . . . . .	263
2.	Installation de l'environnement Anaconda et Jupyter. . . . .	264
3.	Travailler avec Jupyter . . . . .	270
3.1	Les documents dans Jupyter . . . . .	272
3.1.1	Créer un dossier . . . . .	273
3.1.2	Renommer un dossier. . . . .	274
3.1.3	Déplacer un dossier. . . . .	275
3.1.4	Charger des documents . . . . .	276
3.1.5	Supprimer des éléments. . . . .	277
3.1.6	Navigation dans l'arborescence des dossiers . . . . .	278
3.1.7	Créer un notebook . . . . .	279
3.2	Utiliser un notebook Jupyter. . . . .	280
3.2.1	Renommer un notebook . . . . .	280
3.2.2	Les cellules Jupyter . . . . .	281
3.2.3	Les fonctionnalités d'un notebook . . . . .	285

# 8 — Le Machine Learning avec Python

De la théorie à la pratique

3.3	Utiliser les widgets Jupyter	291
3.3.1	Le widget FloatSlider	291
3.3.2	Associer une fonction à un slider	292
3.3.3	Le widget interact	294
3.3.4	Le widget Image	296
3.3.5	Le widget DatePicker	296
4.	Conclusion	297

## Partie 3 : Les statistiques

### Chapitre 3-1 Statistiques

1.	Objectif du chapitre	299
2.	Les statistiques descriptives	300
2.1	Paramètres de position	300
2.1.1	La moyenne	300
2.1.2	Le mode	301
2.1.3	La médiane	301
2.1.4	Les quartiles	304
2.2	Paramètres de dispersion	304
2.2.1	La variance	305
2.2.2	Calcul de la variance avec la formule de Koenig	305
2.2.3	L'écart-type	305
2.2.4	L'écart interquartile	306
3.	Les lois de probabilité	306
4.	La loi normale	308
5.	L'échantillonnage	312
5.1	Principe de l'échantillonnage	312
5.2	Résultats sur la distribution des moyennes	313
5.3	Résultats sur la distribution des proportions	322
5.4	Théorème central limite	326

- 6. Les statistiques inférentielles . . . . . 327
  - 6.1 Estimation ponctuelle . . . . . 328
  - 6.2 Estimation de la moyenne par intervalle de confiance. . . . . 332
  - 6.3 Estimation d'une proportion par intervalle de confiance. . . . . 336
  - 6.4 Test d'hypothèse. . . . . 340
    - 6.4.1 Tests paramétriques . . . . . 341
    - 6.4.2 Tests non paramétriques . . . . . 341
    - 6.4.3 Construire un test d'hypothèse . . . . . 342
  - 6.5 Types de tests d'hypothèse . . . . . 345
    - 6.5.1 Test de conformité . . . . . 345
    - 6.5.2 Test d'adéquation . . . . . 346
    - 6.5.3 Tests d'homogénéité. . . . . 347
    - 6.5.4 Test d'indépendance de variables . . . . . 348
  - 6.6 Exemple numérique de test de conformité d'une moyenne. . . 349
  - 6.7 Le paradoxe de Simpson. . . . . 352
- 7. Conclusion . . . . . 354

**Partie 4 : Les grands algorithmes du Machine Learning**

**Chapitre 4-1**

**La régression linéaire et polynomiale**

- 1. Objectif du chapitre . . . . . 355
- 2. La régression linéaire simple. . . . . 356
  - 2.1 La régression linéaire simple de point de vue géométrique . . . 357
  - 2.2 La régression linéaire simple de point de vue analytique . . . . 358
    - 2.2.1 La méthode des moindres carrés . . . . . 358
    - 2.2.2 Quelques considérations statistiques sur les données . . 360
- 3. La régression linéaire multiple . . . . . 361
  - 3.1 La méthode des moindres carrés pour la régression multiple . 362
  - 3.2 La méthode de la descente de gradient . . . . . 363

# 10 — Le Machine Learning avec Python

De la théorie à la pratique

3.3	Exemple de régression linéaire multiple . . . . .	364
3.3.1	Définition du jeu de données utilisées . . . . .	364
3.3.2	Régression linéaire multiple avec Scikit-learn . . . . .	365
3.3.3	Importer les modules Scikit-learn . . . . .	366
3.3.4	Lecture des données dans un DataFrame . . . . .	367
3.3.5	Normalisation des données . . . . .	368
3.3.6	Construction d'un modèle linéaire . . . . .	371
3.3.7	Évaluation d'un modèle linéaire. . . . .	373
3.3.8	Évaluer le futur comportement d'un modèle . . . . .	377
3.3.9	Cross-validation avec KFold. . . . .	380
4.	La régression polynomiale . . . . .	388
4.1	Exemple de régression polynomiale. . . . .	389
4.1.1	Construction d'un modèle polynomial. . . . .	389
4.1.2	Le coefficient de détermination $R^2$ . . . . .	396
4.1.3	$R^2$ et les valeurs extrêmes . . . . .	398
4.1.4	Modèle polynomial et surapprentissage . . . . .	398
5.	Aller plus loin avec les modèles de régression. . . . .	404
5.1	La régularisation Lasso . . . . .	404
5.2	La régularisation Ridge. . . . .	405
6.	Conclusion . . . . .	405

## Chapitre 4-2

### La régression logistique

1.	Objectif du chapitre . . . . .	407
2.	La régression logistique . . . . .	408
3.	Prédire les survivants du Titanic . . . . .	412
3.1	Définition du jeu de données Titanic . . . . .	412
3.2	Réalisation du modèle de régression logistique . . . . .	413
3.2.1	Chargement des modules Scikit-learn. . . . .	413
3.2.2	Lecture des données . . . . .	414
3.2.3	Traitement des valeurs manquantes. . . . .	415

- 3.2.4 Transformation de variables ..... 416
- 3.2.5 Sélection des variables ..... 418
- 3.2.6 Traitement des variables catégorielles..... 420
- 3.2.7 Entraînement du modèle logistique ..... 421
- 3.2.8 Le seuil de décision ..... 422
- 4. L'algorithme One-vs-All ..... 426
- 5. Conclusion ..... 426

**Chapitre 4-3**  
**Arbres de décision et Random Forest**

- 1. Objectif du chapitre ..... 427
  - 1.1 Construction d'un arbre de décision ..... 428
  - 1.2 Prédire la classe d'appartenance avec un arbre de décision ... 431
  - 1.3 Considérations théoriques sur les arbres de décision ..... 432
    - 1.3.1 Choix de la variable de segmentation ..... 433
    - 1.3.2 Profondeur d'un arbre de décision ..... 434
- 2. Problème de surapprentissage avec un arbre de décision ..... 439
- 3. Random Forest ..... 439
- 4. Exemple de Random Forest avec Scikit-learn ..... 440
- 5. Conclusion ..... 446

**Chapitre 4-4**  
**L'algorithme k-means**

- 1. Objectif du chapitre ..... 447
- 2. k-means du point de vue géométrique ..... 448
- 3. k-means du point de vue algorithmique ..... 455
- 4. Application de k-means avec Scikit-learn ..... 457
- 5. L'algorithme k-means et les valeurs extrêmes..... 464

# 12 — Le Machine Learning avec Python

De la théorie à la pratique

6. Choisir le $k$ de $k$ -means . . . . .	470
6.1 Déterminer $k$ avec la méthode Elbow . . . . .	473
6.2 Déterminer $k$ avec le coefficient de silhouette . . . . .	479
7. Les limites de $k$ -means . . . . .	485
8. Avantages et inconvénients de l'algorithme $k$ -means . . . . .	490
9. Quelques versions de l'algorithme $k$ -means . . . . .	491
10. Conclusion . . . . .	491

## Chapitre 4-5

### Support Vector Machine

1. Objectif du chapitre . . . . .	493
2. Le SVM du point de vue géométrique. . . . .	494
3. Le SVM du point de vue analytique . . . . .	498
4. Données non linéairement séparables. . . . .	503
4.1 Le Kernel Trick . . . . .	505
4.2 La condition de Mercer. . . . .	507
4.3 Exemple de fonction noyau . . . . .	507
5. Détecter les fraudes de cartes de crédit . . . . .	508
5.1 Les données des transactions de cartes de crédit . . . . .	508
5.2 Application de l'algorithme SVM pour la détection des transactions bancaires frauduleuses . . . . .	509
5.2.1 Application de l'algorithme SVM sur les données creditcard.csv . . . . .	509
5.2.2 Application du SVM sur un sous-ensemble de creditcard.csv . . . . .	516
5.2.3 Application du SVM sur des données normalisées. . . . .	518
5.3 Les paramètres de l'algorithme SVM. . . . .	523
5.3.1 Le paramètre Kernel pour la variation de la fonction noyau. . . . .	524
5.3.2 Le paramètre $C$ . . . . .	525
5.3.3 Le paramètre Gamma. . . . .	529

- 5.3.4 Le paramètre C versus le paramètre Gamma . . . . . 531
- 5.3.5 Tuning des hyperparamètres d'un SVM  
avec GridSearchCV . . . . . 531
- 6. Conclusion . . . . . 535

**Chapitre 4-6**  
**Analyse en composantes principales**

- 1. Objectif du chapitre . . . . . 537
- 2. Pourquoi l'ACP ? . . . . . 538
- 3. L'ACP du point de vue géométrique . . . . . 540
- 4. L'ACP du point de vue analytique . . . . . 542
- 5. Indicateurs de la qualité de la représentation des données . . . . . 545
  - 5.1 Indicateurs liés aux individus . . . . . 545
    - 5.1.1 Score des individus . . . . . 546
    - 5.1.2 Qualité de la représentation des individus . . . . . 546
    - 5.1.3 Contribution des individus . . . . . 546
  - 5.2 Indicateurs liés aux variables . . . . . 547
    - 5.2.1 Le cercle des corrélations . . . . . 547
    - 5.2.2 Qualité de la représentation des variables . . . . . 548
    - 5.2.3 Contribution des variables . . . . . 549
- 6. Exemple d'ACP avec Python . . . . . 549
  - 6.1 Déterminer le nombre de facteurs pertinents . . . . . 554
  - 6.2 Interprétation des résultats sur les individus . . . . . 559
    - 6.2.1 Représentation des individus . . . . . 560
    - 6.2.2 Calcul de la qualité de la représentation des individus . 564
    - 6.2.3 Calcul de la contribution des individus . . . . . 565
  - 6.3 Interprétation des résultats sur les variables . . . . . 566
    - 6.3.1 Tracer un cercle des corrélations . . . . . 566
    - 6.3.2 Calcul de la qualité de la représentation des variables . 569
    - 6.3.3 Calcul des contributions des variables . . . . . 570
  - 6.4 Représentation de nouveaux individus . . . . . 571

# 14 — Le Machine Learning avec Python

De la théorie à la pratique

7. Conclusion .....	573
---------------------	-----

## Chapitre 4-7

### Les réseaux de neurones

1. Objectif du chapitre .....	575
2. Modélisation d'un neurone artificiel .....	576
2.1 Le neurone biologique .....	577
2.2 Le neurone artificiel .....	578
3. Architecture d'un réseau de neurones .....	580
4. L'algorithme de rétropropagation .....	583
5. Exemple d'un réseau de neurones avec Scikit-learn .....	594
6. Conclusion .....	601

## Partie 5 : Le Deep Learning et le traitement automatique du langage

### Chapitre 5-1

#### Le Deep Learning avec TensorFlow

1. Objectif du chapitre .....	603
2. Le Deep Learning : notions générales .....	604
2.1 Réseau de neurones avec plusieurs couches d'entrée .....	607
2.2 Réseau de neurones avec plusieurs couches de sortie .....	608
2.3 Réseau de neurones avec des branchements conditionnels .....	609
2.4 Réseau de neurones avec de la récurrence RNN .....	610
2.5 Réseau de neurones avec couches de convolution CNN .....	611
2.6 Éviter le surapprentissage avec les couches Dropout .....	613
2.7 Le Transfer Learning .....	615

- 3. Introduction à TensorFlow ..... 619
  - 3.1 Installer TensorFlow ..... 619
    - 3.1.1 Créer un environnement virtuel ..... 620
    - 3.1.2 Installer des bibliothèques dans un environnement virtuel avec Anaconda ..... 624
    - 3.1.3 Installer la bibliothèque TensorFlow ..... 626
    - 3.1.4 Tester TensorFlow ..... 627
  - 3.2 Opérations élémentaires avec les tensors ..... 628
    - 3.2.1 Travailler avec les tensors ..... 629
    - 3.2.2 Les tensors variables ..... 633
    - 3.2.3 Initialiser les tensors ..... 633
    - 3.2.4 Opérations algébriques avec les tensors ..... 634
- 4. Les réseaux de neurones avec Sequential API ..... 635
  - 4.1 Charger les données ..... 636
  - 4.2 Définir un MLP avec Sequential API ..... 641
  - 4.3 Accéder aux éléments d'un réseau de neurones ..... 643
  - 4.4 Initialisation des poids et des biais d'un réseau de neurones . . 645
  - 4.5 Compiler un réseau de neurones ..... 647
  - 4.6 Entraîner un réseau de neurones ..... 648
  - 4.7 Analyser les résultats de l'entraînement d'un réseau de neurones ..... 650
  - 4.8 Évaluer un réseau de neurones ..... 652
  - 4.9 Prédire avec un réseau de neurones pour la classification .... 652
- 5. Utiliser Functional API ..... 654
  - 5.1 Un modèle Functional API avec plusieurs couches d'entrée . . 655
  - 5.2 Un modèle Functional API avec plusieurs couches de sortie . . 658
- 6. Opérations avancées sur les réseaux de neurones ..... 661
  - 6.1 Monitorer un réseau de neurones ..... 661
    - 6.1.1 Contrôler les critères d'arrêt avec les callbacks ..... 661
    - 6.1.2 TensorBoard ..... 664
    - 6.1.3 Sauvegarder un réseau de neurones ..... 668
    - 6.1.4 Charger et utiliser un réseau de neurones ..... 668
  - 6.2 Réseaux de neurones de convolution ..... 670

# 16 — Le Machine Learning avec Python

De la théorie à la pratique

6.3	Réutiliser un réseau de neurones . . . . .	674
6.4	Le Transfer Learning . . . . .	676
6.4.1	Chargement des données locales . . . . .	678
6.4.2	Chargement du modèle VGG16 . . . . .	679
6.4.3	Extraction des features . . . . .	679
6.4.4	Étendre un modèle . . . . .	680
6.4.5	Chargement des données de test pour le Transfer Learning . . . . .	681
7.	Aller plus loin avec le Deep Learning et TensorFlow . . . . .	684
8.	Conclusion . . . . .	685

## Chapitre 5-2

### Le traitement automatique du langage

1.	Objectif du chapitre . . . . .	687
2.	NLP : concepts généraux . . . . .	688
2.1	Le nettoyage des données textuelles . . . . .	690
2.1.1	Suppression des stopwords . . . . .	690
2.1.2	Appliquer le Stemming sur un texte . . . . .	692
2.1.3	Appliquer la Lemmatization sur un texte . . . . .	692
2.1.4	Stemming versus Lemmatization . . . . .	692
2.2	Vectorisation des données textuelles . . . . .	693
2.2.1	La vectorisation par comptage d'occurrences des mots . . . . .	694
2.2.2	La vectorisation avec TF-IDF . . . . .	696
2.2.3	La vectorisation avec N-Gram . . . . .	698
2.2.4	Feature Engineering sur des documents . . . . .	699
3.	Exemple complet pour la détection des spams . . . . .	700
3.1	Installation de la NLTK . . . . .	701
3.2	Modèle de détection de spams . . . . .	702
4.	Conclusion . . . . .	709

**Annexe**

**La programmation orientée objet avec Python**

- 1. Programmation orientée objet avec Python . . . . . 711
  - 1.1 Pourquoi la programmation orientée objet? . . . . . 711
  - 1.2 Classes et objets . . . . . 713
    - 1.2.1 Définir une classe . . . . . 713
    - 1.2.2 La fonction `__init__` . . . . . 714
    - 1.2.3 Instanciation d'un objet . . . . . 716
    - 1.2.4 Les attributs d'un objet . . . . . 717
    - 1.2.5 Les méthodes d'objet . . . . . 719
    - 1.2.6 Les attributs de classe . . . . . 721
    - 1.2.7 Les méthodes de classe . . . . . 722
    - 1.2.8 Les méthodes statiques . . . . . 724
    - 1.2.9 Sécuriser les attributs . . . . . 725
  - 1.3 L'héritage . . . . . 729
    - 1.3.1 L'héritage simple . . . . . 729
    - 1.3.2 L'héritage multiple . . . . . 733
  - 1.4 Les classes abstraites . . . . . 735
  - 1.5 Les interfaces . . . . . 737
  - 1.6 Les méthodes spéciales . . . . . 741
    - 1.6.1 Afficher un objet avec la fonction `print()` . . . . . 741
    - 1.6.2 Personnaliser les accès aux attributs d'une classe . . . . . 744
    - 1.6.3 Vérifier la validité d'un attribut . . . . . 745
    - 1.6.4 Comparer deux objets . . . . . 747
    - 1.6.5 Rendre les objets callable . . . . . 748
- 2. Les modules . . . . . 749
  - 2.1 Importer des modules . . . . . 750
  - 2.2 Le module principal . . . . . 753
- 3. Pour aller plus loin avec Python . . . . . 756

## Chapitre 4-4

# L'algorithme k-means

### 1. Objectif du chapitre

Les chapitres précédents ont abordé des exemples de deux types d'algorithmes de Machine Learning : les algorithmes de régression et de classification. Ce chapitre porte sur l'algorithme k-means, appelé l'algorithme des k-moyennes en français, qui est un algorithme simple à comprendre et qui fait partie des algorithmes de clustering les plus connus et les plus utilisés.

L'algorithme k-means a été introduit par J. McQueen en 1967. C'est un algorithme non supervisé qui permet de répartir un ensemble de  $n$  observations en  $k$  clusters. L'objectif après l'application de l'algorithme k-means sur un jeu de données est que chaque cluster contienne des observations homogènes et que deux observations de deux clusters différents soient hétérogènes.

Les domaines d'application de l'algorithme k-means sont nombreux. Par exemple, il est très utilisé pour la segmentation des clients à des fins de marketing, ou encore pour l'isolation des motifs dans les images, car justement les images présentent souvent des régions homogènes, notamment en matière d'intensité lumineuse.

De manière générale, le succès de l'algorithme k-means et de ses versions réside dans sa simplicité et sa capacité à traiter des données de grande taille.

À la fin de ce chapitre, le lecteur aura abordé :

Le fonctionnement de k-means via des illustrations.

- Les étapes principales de l'algorithme k-means classique.
- L'application de l'algorithme k-means avec Scikit-learn.
- L'impact des valeurs extrêmes sur les performances de l'algorithme k-means.
- La recherche de la valeur optimale du paramètre  $K$  de l'algorithme k-means.
- Les avantages et les inconvénients ainsi que les variantes de l'algorithme k-means.

## 2. k-means du point de vue géométrique

Comme précisé au début de ce chapitre, l'algorithme k-means est très intuitif et simple à comprendre. Avant d'entrer dans les détails, il faut noter que k-means, comme tous les algorithmes de clustering, ne nécessite pas l'étiquetage des données, car c'est une procédure non supervisée.

De façon informelle, étant donné  $n$  observations à répartir sur  $k$  clusters, k-means choisit initialement, de manière aléatoire,  $k$  observations parmi les  $n$  observations, comme étant les centres des  $k$  clusters recherchés. Chacune des  $n$  observations sera associée au cluster dont le centre est le plus proche parmi les  $k$  centres choisis initialement. Une fois que toutes les observations sont associées à leurs clusters respectifs, le centre de chaque cluster est recalculé en fonction des observations qu'il contient. Puis, de nouveau, chacune des observations est associée au cluster dont le centre est le plus proche de cette observation par rapport à tous les centres des autres clusters. Ces opérations de recalcul des centres des clusters puis d'association des observations aux clusters les plus proches sont répétées jusqu'à ce qu'un critère d'arrêt soit atteint.

L'algorithme k-means utilise une fonction pour calculer les distances entre les observations et les centres des clusters. Ce calcul des distances peut être basé sur la distance euclidienne, la distance de Manhattan ou toute autre fonction permettant de mesurer la dissimilarité entre les observations.

Pour mieux comprendre cet algorithme de clustering, cette section déroule l'algorithme k-means sur un exemple simple. Soit six observations a, b, c, d, e et f à répartir sur deux clusters  $C_1$  et  $C_2$  ; supposons que la distance utilisée est la distance euclidienne classique. Ces six observations sont définies dans un espace à deux dimensions et leurs coordonnées sont indiquées dans le tableau suivant :

Axes	a	b	c	d	e	f
x	2	4	2	4	10	10
y	4	4	2	2	2	4

Figure 10-1 : un simple jeu de données avec leurs coordonnées en deux dimensions

#### ■ Remarque

Pour rappel, la distance euclidienne entre deux observations  $A=(x_A, y_A)$  et  $B=(x_B, y_B)$  est calculée grâce à la formule :

$$\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}$$

Avant de faire un traitement quelconque sur les données, il est toujours intéressant de les visualiser lorsque c'est possible. Dans cet exemple, les données sont définies dans un espace à deux dimensions, donc elles peuvent être facilement visualisées sur deux axes comme dans la figure 10-2 ci-dessous.

#### ■ Remarque

Même lorsque les données sont définies dans un espace à grande dimension, supérieur à deux ou à trois dimensions, il existe des méthodes qui permettent de les visualiser en deux ou trois dimensions, avec une perte d'informations qu'on espère minimale. Ces méthodes sont appelées les méthodes de réduction de domaines. Le chapitre Analyse en composantes principales présente l'analyse du même nom, qui est l'une des méthodes de réduction de domaines les plus connues et qui permet d'avoir une vue en deux dimensions des données définies initialement dans un espace à grande dimension.

# 450 — Le Machine Learning avec Python

De la théorie à la pratique

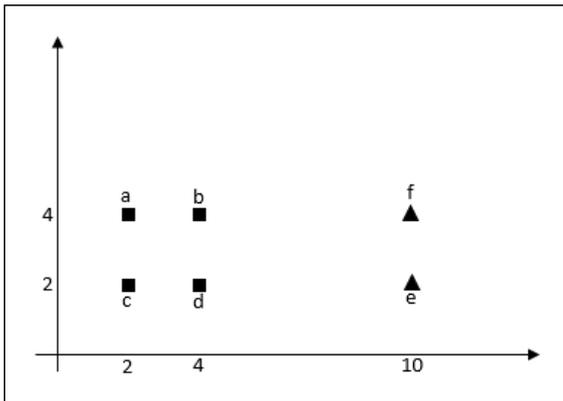


Figure 10-2 : représentation graphique en deux dimensions des données

Ce graphique montre clairement que les observations a, b, c et d, représentées par des carrés, sont très proches entre elles au sens de la distance euclidienne, par rapport aux deux observations e et f. Également, les deux dernières observations, représentées par des triangles, sont très proches entre elles.

Pour cet exemple, en se basant donc sur la distance euclidienne, un algorithme de clustering efficace proposerait certainement de répartir ces six observations dans les deux clusters  $C_1$  et  $C_2$  comme dans la figure 10-3 ci-dessous.

En effet, avec la distance euclidienne, cette répartition est optimale. La section suivante définit de façon plus formelle la notion de solution optimale pour un algorithme k-means et pour un nombre de clusters fixe.

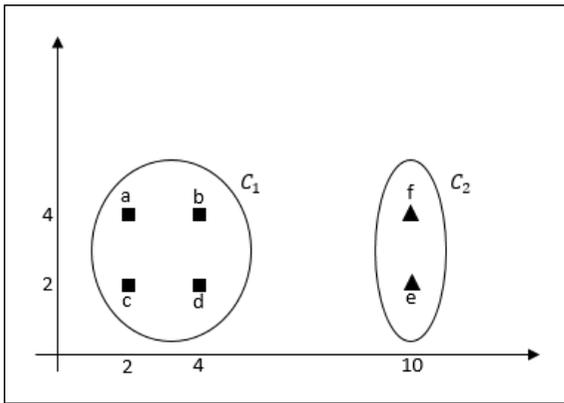


Figure 10-3 : répartition des six points dans les deux classes  $C_1$  et  $C_2$

En suivant les étapes classiques de l'algorithme k-means, le résultat optimal de la figure 10-3 peut être obtenu comme suit :

1. L'algorithme k-means commence initialement par sélectionner de façon aléatoire deux observations parmi les six observations disponibles. Les deux observations ainsi sélectionnées vont être considérées comme les centres des deux clusters recherchés  $C_1$  et  $C_2$ . Ici, nous supposons que l'algorithme k-means recherche un nombre de clusters qui est égal à 2.

Dans cet exemple, supposons que les deux observations a et d sont sélectionnées aléatoirement. Ces deux observations vont être considérées comme les centres respectifs des clusters  $C_1$  et  $C_2$ . L'algorithme k-means calcule les distances entre chacune des six observations avec les centres a et d. Les résultats sont reportés dans le tableau ci-dessous :

Les centres des clusters	a	b	c	d	e	f
Centre de $C_1$ =a=(2,4)	0	2	2	2.8284	8.2462	8
Centre de $C_2$ =d=(4,2)	2.8284	2	2	0	6	6.3245

Figure 10-4 : distances entre les observations et les centres de  $C_1$  et  $C_2$

2. Une fois que k-means dispose de toutes les distances entre toutes les observations et les deux centres a et d, il procède à l'association entre les observations et les clusters. Par exemple, l'observation e va être associée au cluster  $C_2$ , puisqu'elle est plus proche du centre de  $C_2$  que du centre de  $C_1$ . À la suite de cette étape, les deux clusters  $C_1$  et  $C_2$  vont être constitués comme suit  $C_1 = \{a, b, c\}$  et  $C_2 = \{d, e, f\}$

Lorsqu'une observation est à la même distance des clusters  $C_1$  et  $C_2$ , alors elle est affectée à l'un de ces deux clusters de manière aléatoire. Dans notre exemple nous avons affecté de manière arbitraire les deux observations b et c au cluster  $C_1$ .

La figure suivante présente graphiquement les deux clusters  $C_1$  et  $C_2$  obtenus à la suite de cette étape :

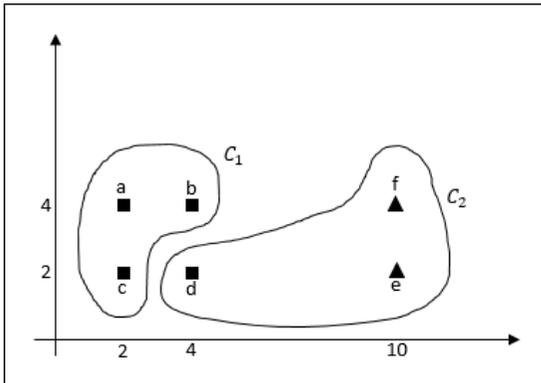


Figure 10-5 : affectation des observations aux clusters  $C_1$  et  $C_2$  après la première itération

3. Lors de la première étape, l'algorithme k-means a choisi de façon aléatoire a et d en tant que centres des deux clusters recherchés. À l'étape 2, les deux clusters  $C_1$  et  $C_2$  ont été concrètement construits.

À cette étape, l'algorithme k-means calcule le centre du cluster  $C_1$  en utilisant les observations a, b et c et calcule le centre du cluster  $C_2$  en utilisant les observations d, e et f.

Ainsi : le centre de  $C_1 = \left(\frac{1}{3}(2 + 2 + 4), \frac{1}{3}(4 + 4 + 2)\right) = (2.66, 3.33)$

le centre de  $C_2 = \left(\frac{1}{3}(4 + 10 + 10), \frac{1}{3}(2 + 2 + 4)\right) = (8, 2.66)$

Les centres des deux clusters  $C_1$  et  $C_2$  sont représentés avec le symbole étoile dans la figure 10-6 ci-dessous :

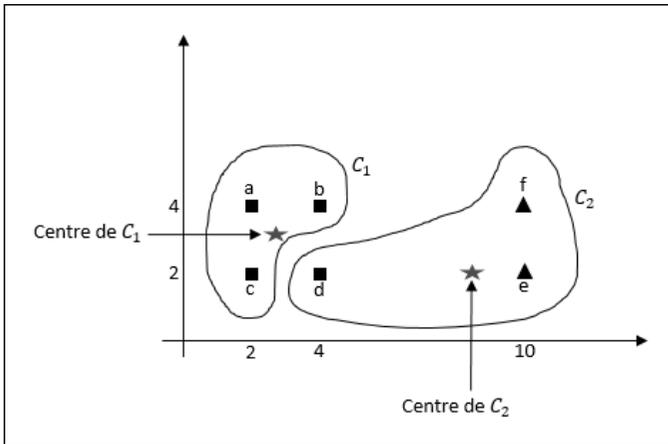


Figure 10-6 : les centres des clusters  $C_1$  et  $C_2$

- À cette étape, les six observations sont réparties de nouveau sur les deux clusters  $C_1$  et  $C_2$  en se basant sur les nouveaux centres calculés à l'étape précédente. Les distances entre les observations et les centres de  $C_1$  et  $C_2$  sont reportées dans le tableau suivant :

Les centres des clusters	a	b	c	d	e	f
Centre de $C_1 = (2.66, 3.33)$	0.9404	1.4981	1.4847	1.8879	7.4595	7.3505
Centre de $C_2 = (8, 2.66)$	6.1478	4.2184	6.0361	4.0540	2.1060	2.4074

Figure 10-7 : distances entre les observations et les centres de  $C_1$  et  $C_2$