

Intelligence-Artificielle



- Dispensé par **MWAMBA KASONGO Dahouda**
- Docteur en génie logiciel et systèmes d'information
- Machine and Deep Learning Engineer

- Assisté par Ass. **Jason MUSA**

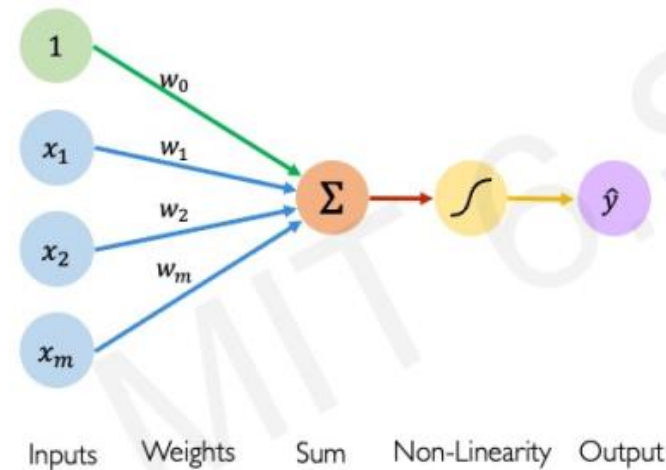
Mardi : 8H00 – 12H00

Mercredi : 13H00 – 17H00

Vendredi : 13H00 – 13H00

- E-mail : dahouda37@gmail.com
- Tel.: +243 99 66 55 265

The Perceptron: Forward Propagation

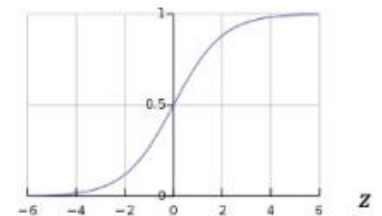


Activation Functions

$$\hat{y} = g(w_0 + \mathbf{X}^T \mathbf{W})$$

- Example: sigmoid function

$$g(z) = \sigma(z) = \frac{1}{1 + e^{-z}}$$



PLAN DU COURS

CHAPITRE 2 Concepts clés sur l'apprentissage automatique

2.1. Terminologies d'apprentissage automatique

- **Features (Caractéristiques), Label (étiquettes) et Dataset (ensembles de données)**

2.2 Types de données

- **Catégorielles, Numériques, Textuelles, Images, audio**
- Training set (Données d'entraînement), Validation set (Données de validation), Test set (Données de test)

2.3 Introduction aux algorithmes (Type de Machine Learning)

- **Qu'est-ce qu'un algorithme ?**
- **Les algorithmes de Machine Learning**

2.4 Exemple Pratique de la Régression linéaire simple

2.5 Exemple Pratique de la Régression linéaire multiple

2.6. Introduction à l'apprentissage automatique sur AWS



CHAPITRE 2 MACHINE LEARNING



2.1. Terminologies d'apprentissage automatique

✓ **Ensemble de données [Dataset]** : est un ensemble de données organisées de manière structurée ou non structurée.

Les ensembles de données peuvent se présenter sous différents formats et peuvent contenir différents types de données, tels que du texte, des nombres, des images, des vidéos ou de l'audio. Exemple d'une Dataset: <https://archive.ics.uci.edu/ml/datasets/Bank+Marketing>

age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
30	unemployed	married	primary	no	1787	no	no	cellular	19	oct	79	1	-1	0	unknown	no
33	services	married	secondary	no	4789	yes	yes	cellular	11	may	220	1	339	4	failure	no
35	managemen	single	tertiary	no	1350	yes	no	cellular	16	apr	185	1	330	1	failure	no
30	managemen	married	tertiary	no	1476	yes	yes	unknown	3	jun	199	4	-1	0	unknown	no
59	blue-collar	married	secondary	no	0	yes	no	unknown	5	may	226	1	-1	0	unknown	no
35	managemen	single	tertiary	no	747	no	no	cellular	23	feb	141	2	176	3	failure	no
36	self-employ	married	tertiary	no	307	yes	no	cellular	14	may	341	1	330	2	other	no
39	technician	married	secondary	no	147	yes	no	cellular	6	may	151	2	-1	0	unknown	no
41	entrepreneu	married	tertiary	no	221	yes	no	unknown	14	may	57	2	-1	0	unknown	no
43	services	married	primary	no	-88	yes	yes	cellular	17	apr	313	1	147	2	failure	no
39	services	married	secondary	no	9374	yes	no	unknown	20	may	273	1	-1	0	unknown	no
43	admin.	married	secondary	no	264	yes	no	cellular	17	apr	113	2	-1	0	unknown	no
36	technician	married	tertiary	no	1109	no	no	cellular	13	aug	328	2	-1	0	unknown	no
20	student	single	secondary	no	502	no	no	cellular	30	apr	261	1	-1	0	unknown	yes
31	blue-collar	married	secondary	no	360	yes	yes	cellular	29	jan	89	1	241	1	failure	no

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.1. Terminologies d'apprentissage automatique

- ✓ **Modèle [Model]** : un modèle est une représentation spécifique apprise à partir de données en appliquant un algorithme de ML.
Un modèle est également appelé hypothèse.
- ✓ **Fonctionnalité [Feature]** : une caractéristique est une propriété individuelle mesurable de nos données.
Un ensemble de caractéristiques numériques peut être décrit de manière pratique par un vecteur de caractéristiques.

- ❖ Par exemple, pour prédire un fruit, il peut y avoir des caractéristiques comme la couleur, l'odeur, le goût, etc.
Remarque : le choix de caractéristiques informatives, discriminantes et indépendantes est une étape cruciale pour des algorithmes efficaces.

- ✓ **Cible ou étiquette [Target ou Label]** : une variable cible ou une étiquette est la valeur à prédire par notre modèle.
Pour l'exemple de fruit abordé dans la section sur les fonctionnalités, l'étiquette de chaque ensemble d'entrées serait le nom du fruit comme la pomme, l'orange, la banane, etc.
- ✓ **Entraînement [Training]** : l'idée est de fournir un ensemble d'entrées (caractéristiques) et ses sorties attendues (étiquettes), de sorte qu'après l'entraînement, nous aurons un modèle (hypothèse) qui mapperait ensuite les nouvelles données à l'une des catégories sur lesquelles l'entraînement a été effectué.
- ✓ **Prédiction** : une fois que notre modèle est prêt, il peut être alimenté par un ensemble d'entrées auxquelles il fournira une sortie prédite (étiquette). Mais assurez-vous que si la machine fonctionne bien sur des données invisibles (Test data), alors seulement nous pouvons dire que le modèle fonctionne bien.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.2. Types de données

1. **Ensemble de données tabulaires** : organisé en lignes et en colonnes.

Couramment utilisé dans les affaires, les soins de santé et la finance (par exemple, les feuilles de calcul).

Out[9]:

	price	resid_area	air_qual	room_num	age	dist1	dist2	dist3	dist4	teachers	poor_prop	n_hos_beds	n_hot_rooms	rainfall	parks	Sold
0	24.0	32.31	0.538	6.575	65.2	4.35	3.81	4.18	4.01	24.7	4.98	5.480	11.1920	23	0.049347	0
1	21.6	37.07	0.469	6.421	78.9	4.99	4.70	5.12	5.06	22.2	9.14	7.332	12.1728	42	0.046146	1
2	34.7	37.07	0.469	7.185	61.1	5.03	4.86	5.01	4.97	22.2	4.03	7.394	101.1200	38	0.045764	0
3	33.4	32.18	0.458	6.998	45.8	6.21	5.93	6.16	5.96	21.3	2.94	9.268	11.2672	45	0.047151	0
4	36.2	32.18	0.458	7.147	54.2	6.16	5.86	6.37	5.86	21.3	5.33	8.824	11.2896	55	0.039474	0

In [10]: `df.shape`

Out[10]: (506, 16)

Variables numériques

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.2. Types de données

1. **Ensemble de données tabulaires** : organisé en lignes et en colonnes.

Couramment utilisé dans les affaires, les soins de santé et la finance (par exemple, les feuilles de calcul).

	id	bin_0	bin_1	bin_2	bin_3	bin_4	nom_0	nom_1	nom_2	nom_3	...	nom_8	nom_9	ord_0	ord_1	ord_2	ord_3	ord_4	ord_5	day	month
0	300000	0	0	1	T	Y	Blue	Triangle	Axolotl	Finland	...	9d117320c	3c49b42b8	2	Novice	Warm	j	P	be	5	11
1	300001	0	0	0	T	N	Red	Square	Lion	Canada	...	46ae3059c	285771075	1	Master	Lava Hot	l	A	RP	7	5
2	300002	1	0	1	F	Y	Blue	Square	Dog	China	...	b759e21f0	6f323c53f	2	Expert	Freezing	a	G	tP	1	12
3	300003	0	0	1	T	Y	Red	Star	Cat	China	...	0b6ec68ff	b5de3dcc4	1	Contributor	Lava Hot	b	Q	ke	2	3
4	300004	0	1	1	F	N	Red	Trapezoid	Dog	China	...	f91f3b1ee	967cfa9c9	3	Grandmaster	Lava Hot	l	W	qK	4	11

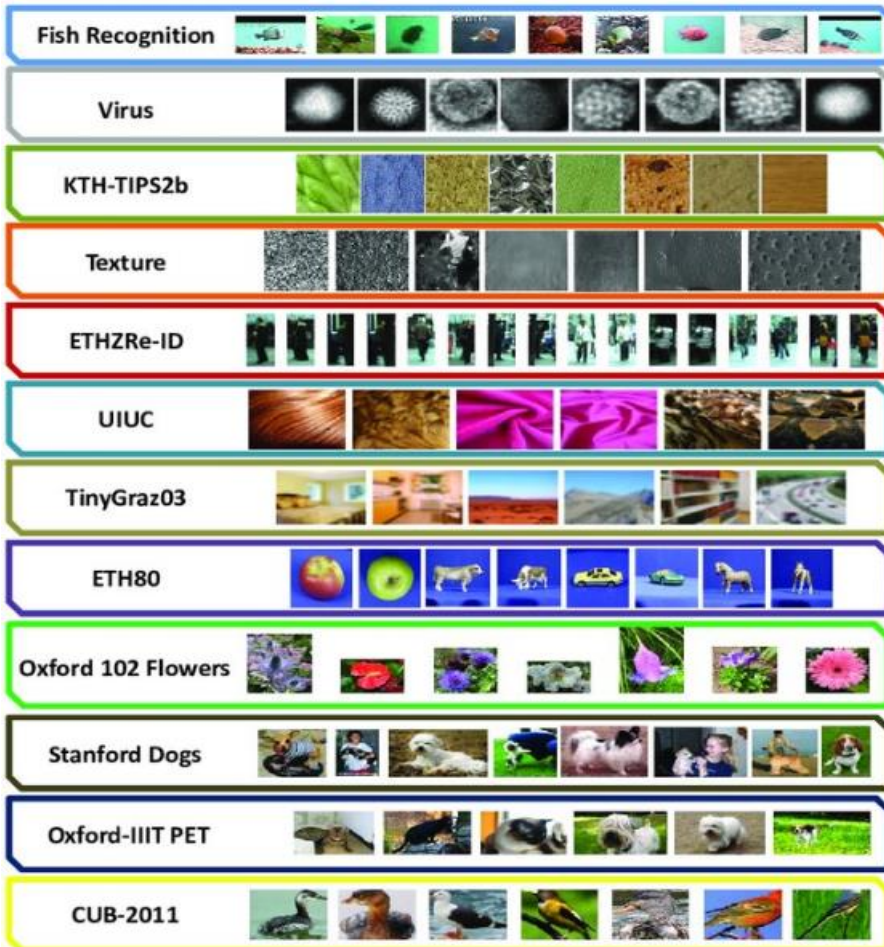
Variables numériques et catégorielles

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.2. Types de données

2. Ensemble de données d'images : Contient des images, souvent utilisées dans les tâches de vision par ordinateur (MNIST, CIFAR-10).



airplane

automobile

bird

cat

deer

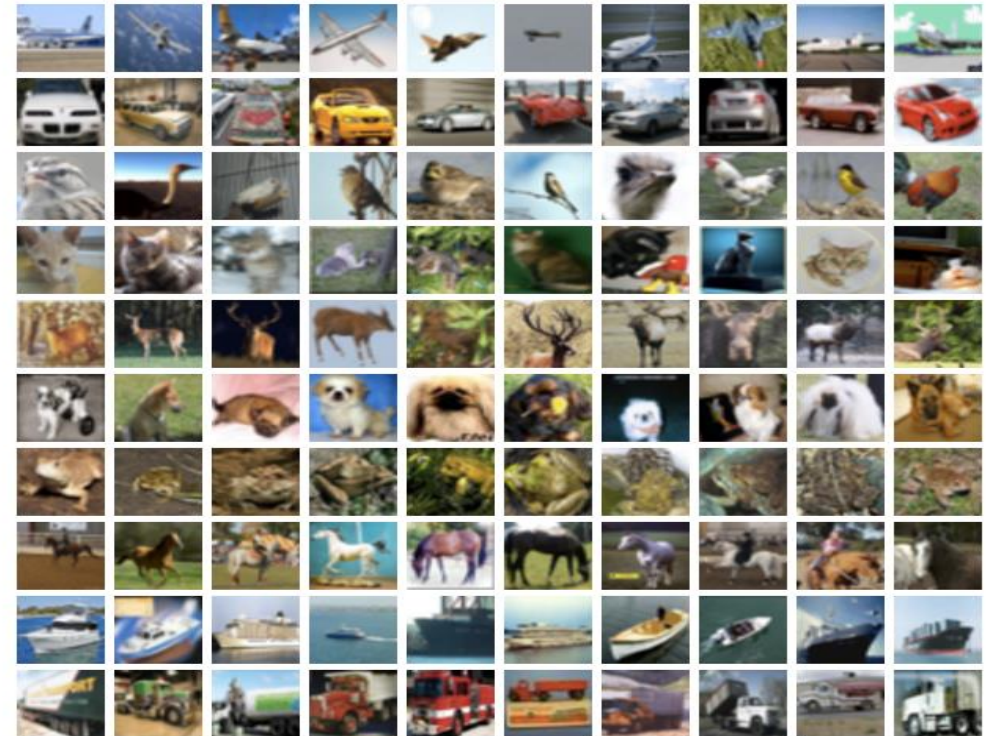
dog

frog

horse

ship

truck



CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.2. Types de données

3. Ensemble de données textuelles : contient des données textuelles pour le traitement du langage naturel (par exemple, l'analyse des sentiments, la traduction linguistique).

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive
5	Probably my all-time favorite movie, a story o...	positive
6	I sure would like to see a resurrection of a u...	positive
7	This show was an amazing, fresh & innovative i...	negative
8	Encouraged by the positive comments about this...	negative
9	If you like original gut wrenching laughter yo...	positive

L'ensemble de données IMDB contient 50 000 critiques de films étiquetées comme des sentiments « positifs » ou « négatifs ».

L'analyse des sentiments est une tâche cruciale de traitement du langage naturel (NLP) qui consiste à déterminer le sentiment ou l'émotion exprimé dans un texte.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

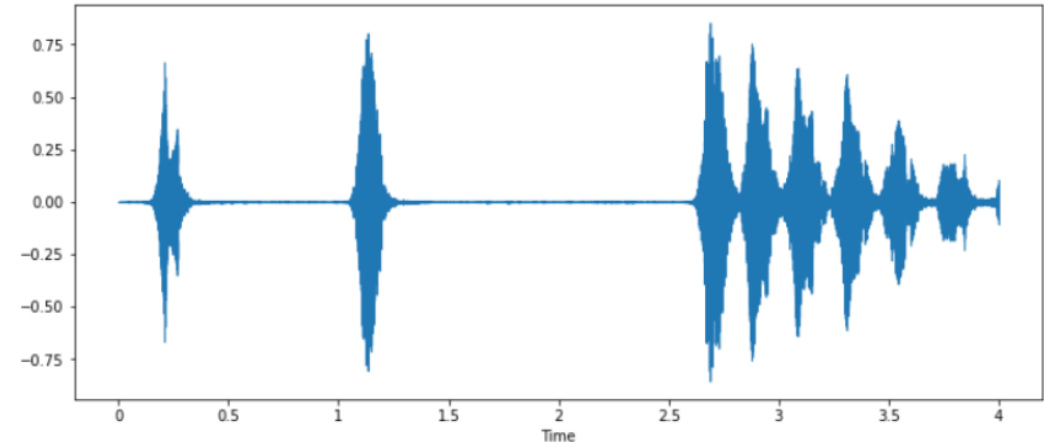
2.2. Types de données

4. **Ensemble de données audio** : contient des données sonores ou vocales (par exemple, utilisées pour des tâches de reconnaissance vocale).



Military Audio Dataset

Un ensemble de données audio militaires pour la connaissance de la situation et la surveillance



Urban Sound 8k Dataset

L'ensemble de données Urban Sound 8k. L'ensemble de données contient 8732 fichiers sonores de 10 classes différentes et est répertorié ci-dessous : 1. Air Conditioner, 2. Car Horn, 3. Children Playing, 4. Dog Bark, 5. Drilling Machine, 6. Engine Idling, 7. Gun Shot, 8. Jackhammer, 9. Siren, 10. Street Music

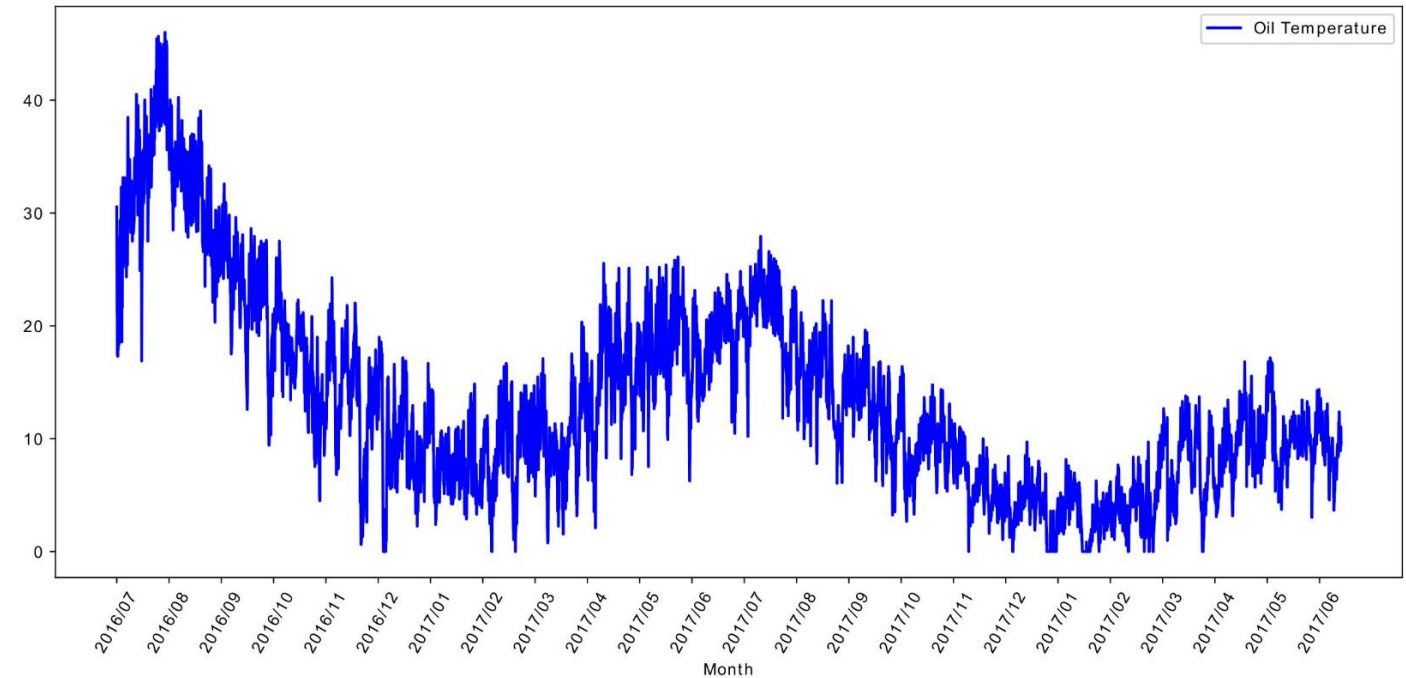
CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.2. Types de données

5. Ensemble de données de séries chronologiques : Points de données indexés dans le temps, utilisés dans les prévisions ou les analyses de tendances (par exemple, cours des actions, données météorologiques).

A	B	C	D	E	F
date	product_type	location	discount	weather_temp	sales
2011-01-01	A	X	0.2	25	50000
2011-01-01	A	Y	0.15	27	6000
2011-01-01	A	Z	0.1	26	70000
2011-01-01	B	X	0.3	25	60000
2011-01-01	B	Y	0.25	27	8000
2011-01-01	B	Z	0.2	26	9000
2011-01-01	C	X	0.13	25	10000
2011-01-01	C	Y	0.14	27	65000
2011-01-01	C	Z	0.16	26	30000
2011-01-02	A	X	0.2	25	50000
2011-01-02	A	Y	0.15	27	6000
2011-01-02	A	Z	0.1	26	70000
2011-01-02	B	X	0.3	25	60000
2011-01-02	B	Y	0.25	27	8000
2011-01-02	B	Z	0.2	26	9000
2011-01-02	C	X	0.13	25	10000
2011-01-02	C	Y	0.14	27	65000
2011-01-02	C	Z	0.16	26	30000



ETT (Electricity Transformer Temperature) Dataset

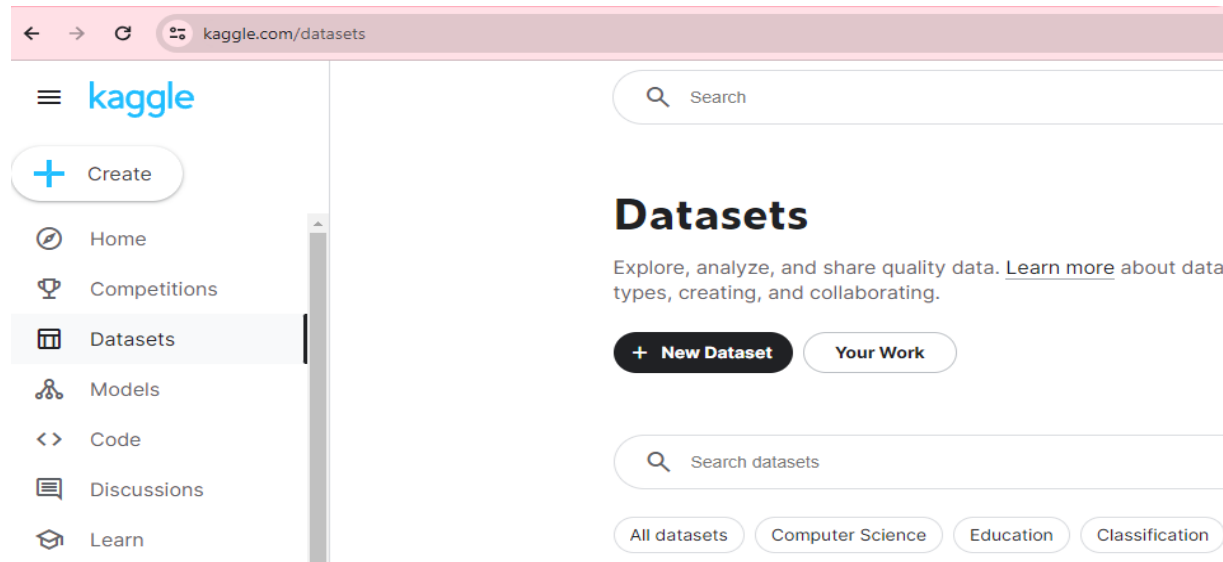
CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3. Source des données Publiques

Il existe de nombreuses sources où vous pouvez obtenir des données pour l'analyse de la science des données, en fonction de vos intérêts spécifiques et du type d'analyse que vous souhaitez effectuer. Voici quelques options populaires :

1. Kaggle : Kaggle est une plate-forme de concours de science des données, mais elle héberge également un grand nombre d'ensembles de données disponibles gratuitement pour l'exploration et l'analyse. Lien du Kaggle : <https://www.kaggle.com/datasets>



CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3. Source des données Publiques

2. Référentiel UCI Machine Learning : le référentiel UCI Machine Learning est un ensemble de bases de données, de théories de domaine et de générateurs de données largement utilisés par la communauté de Machine Learning. Lien de UCI : <https://archive.ics.uci.edu/>


[Datasets](#) [Contribute Dataset](#) [About Us](#)

Welcome to the UC Irvine Machine Learning Repository

We currently maintain 664 datasets as a service to the machine learning community. Here, you can donate and find datasets used by millions of people all around the world!


[VIEW DATASETS](#) [CONTRIBUTE A DATASET](#)

Popular Datasets



Iris
A small classic dataset from Fisher, 1936. One of the earliest known datasets used for ev...


🔍 Classification 📊 150 Instances 📋 4 Features



Dry Bean
Images of 13,611 grains of 7 different registered dry beans were taken with a high-resol...


🔍 Classification 📊 13.61K Instances 📋 16 Features

New Datasets



PhiUSIIL Phishing URL (Website)
PhiUSIIL Phishing URL Dataset is a substantial dataset comprising 134,850 legitimate an...

🔍 Classification 📊 235.8K Instances 📋 54 Features



RT-IoT2022
The RT-IoT2022, a proprietary dataset derived from a real-time IoT infrastructure, is intro...

🔍 Classification, Regressi... 📊 123.12K Instances 📋 84 Features

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3. Source des données Publiques

3. Recherche d'ensembles de données Google : Google Dataset Search vous aide à trouver des ensembles de données stockés sur le Web.

C'est un outil utile pour découvrir des ensembles de données provenant de diverses sources : <https://datasetsearch.research.google.com/>

Dataset Search

The screenshot shows the Google Dataset Search interface. At the top, there is a search bar with the text 'covid' entered. Below the search bar, a list of search results is displayed. The first result is 'coronavirus covid-19'. The second result, 'covid', is highlighted. Other results include 'covid 19', 'covid-19', 'Vaccines.gov: COVID-19 vaccinating provider locations', 'COVID-19 Hospital Data Coverage Report', 'COVID-19-Open-Research-Dataset-Challenge--CORD-19-', 'Assessment of Effectiveness of COVID 19 Pandemic Scheduling Triage in an Academic Dermatology Clinic', and 'Number of COVID-19 people killed by age'.

Search Results
coronavirus covid-19
covid
covid 19
covid-19
Vaccines.gov: COVID-19 vaccinating provider locations
COVID-19 Hospital Data Coverage Report
COVID-19-Open-Research-Dataset-Challenge--CORD-19-
Assessment of Effectiveness of COVID 19 Pandemic Scheduling Triage in an Academic Dermatology Clinic
Number of COVID-19 people killed by age

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3. Source des données Publiques

4. **Portails de données ouvertes gouvernementaux** : de nombreux gouvernements proposent des portails de données ouvertes où vous pouvez trouver des ensembles de données liés à la démographie, à l'économie, à la santé, etc. Lien : <https://data.gov/>

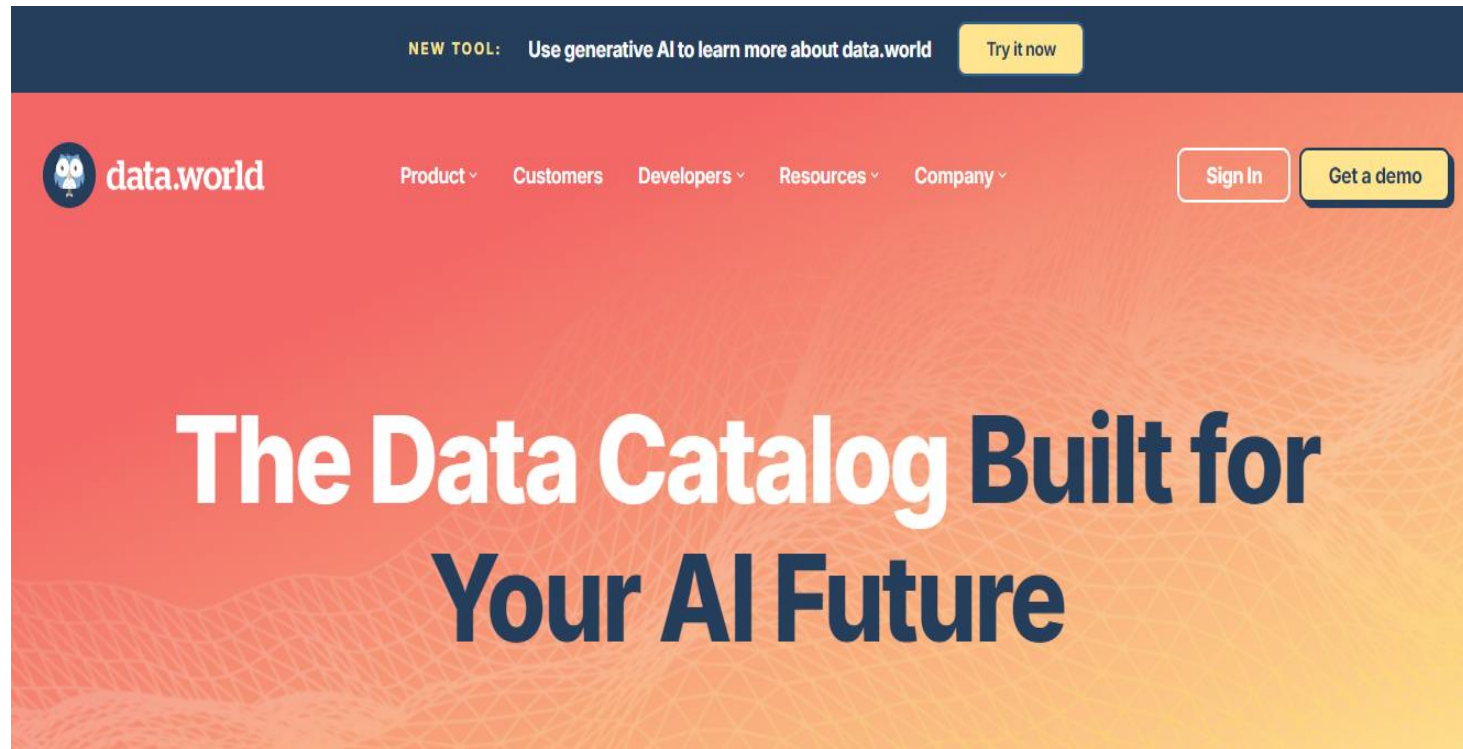


CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.3. Source des données Publiques

5. **Data.world** : Data.world est une plateforme où vous pouvez trouver et partager des ensembles de données.

Il héberge une gamme diversifiée d'ensembles de données fournis par la communauté. Lien : <https://data.world/>



CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3. Source des données Publiques

6. Ensembles de données Reddit : le sous-reddit r/datasets est une communauté où les gens partagent et demandent des ensembles de données. Vous pourriez trouver des ensembles de données intéressants ici <https://www.reddit.com/r/datasets/>

The screenshot displays the Reddit interface for the r/datasets subreddit. On the left, the navigation menu includes 'Home', 'Popular', 'RECENT', and 'TOPICS' (Gaming, Sports, Business, Crypto, Television). The main content area shows the subreddit header with the name 'r/datasets' and a search bar. Below the header, a post by user 'u/Swat_Sam2' is visible, titled 'Help with data analysis project (mysql online server help)'. The post content describes a problem with a MySQL server and asks for help. The right sidebar shows community statistics: 190K members, 35 online, and a top 1% rank by size. There are also links for 'Create a post', 'Join', and 'Community Bookmarks'.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

- ❑ Les algorithmes d'apprentissage automatique sont des techniques utilisés pour créer des systèmes capables d'apprendre à partir de données et de faire des prédictions ou de prendre des décisions sans être explicitement programmés.
- ❑ Ces algorithmes se répartissent en différentes catégories en fonction du type d'apprentissage qu'ils prennent en charge : apprentissage supervisé, non supervisé, semi-supervisé ou par renforcement.

2.3.1. Algorithmes d'apprentissage supervisé

Dans l'apprentissage supervisé, l'algorithme est formé sur des données étiquetées (où l'entrée et la sortie correspondante sont connues). L'objectif est d'apprendre une correspondance entre les entrées et les sorties.

1. Régression linéaire [Linear Regression]

- **Objectif** : Prédire des valeurs continues (par exemple, les prix des maisons).
- **Description** : Modélise la relation entre les caractéristiques d'entrée (variables indépendantes) et une sortie continue (variable dépendante) en ajustant une ligne droite (ou hyperplan) aux données.

2. Régression logistique [Logistic Regression]

- **Objectif** : Problèmes de classification binaire (par exemple, détection de spam)
- **Description** : Similaire à la régression linéaire, mais utilisée pour prédire les résultats catégoriels.
Elle génère des probabilités à l'aide d'une fonction logistique.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

2.3.1. Algorithmes d'apprentissage supervisé

3. Arbres de décision [Decision Trees]

- **Objectif** : Classification et régression.
- **Description** : Modèle de décisions de type arborescence, où les nœuds internes représentent des tests sur des caractéristiques, les branches représentent les résultats de ces tests et les nœuds feuilles représentent le résultat prédit.

4. Random Forest

- **Objectif** : Classification et régression.
- **Description** : Ensemble d'arbres de décision dans lesquels plusieurs arbres sont construits sur des sous-ensembles aléatoires de données et de caractéristiques. La prédiction finale est basée sur le vote majoritaire (classification) ou la moyenne (régression) de tous les arbres.

5. Support Vector Machines (SVM)

- **Objectif** : Classification et régression.
- **Description** : Recherche l'hyperplan optimal qui sépare au maximum les données en différentes classes.
Il fonctionne bien pour les ensembles de données de grande dimension.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

2.3.1. Algorithmes d'apprentissage supervisé

6. K-Nearest Neighbors (k-NN)

- **Objectif** : Classification et régression.
- **Description** : Algorithme non paramétrique dans lequel la prédiction est faite sur la base de la classe majoritaire ou de la moyenne des « k » points les plus proches dans l'espace des caractéristiques.

7. Naive Bayes

- **Objectif** : Classification (par exemple, classification de texte, filtrage du spam)
- **Description** : Un classificateur probabiliste basé sur le théorème de Bayes, supposant que les caractéristiques sont indépendantes les unes des autres (ce qui est une hypothèse « naïve », d'où son nom).

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

2.3.2. Algorithmes d'apprentissage non supervisé

Dans l'apprentissage non supervisé, l'algorithme travaille avec des données non étiquetées et essaie de trouver des modèles ou des structures cachés en leur sein.

1. Clustering k-Means [K-Means Clustering]

- **Objectif** : Clustering (regroupement de points de données similaires).
- **Description** : Partitionne les données en « k » clusters où chaque point de données appartient au cluster avec le centroïde le plus proche (moyenne).

2. Clustering hiérarchique [Hierarchical Clustering]

- **Objectif** : Clustering.
- **Description** : Construit une hiérarchie de clusters en fusionnant ou en divisant de manière répétée des clusters.
Le résultat est un arbre de clusters.

3. Analyse en composantes principales [Principal Component Analysis (PCA)]

- **Objectif** : Réduction de la dimensionnalité.
- **Description** : Transforme les données de grande dimension en un espace de dimension inférieure tout en conservant autant de variance que possible. Souvent utilisé pour la visualisation des données ou la réduction du bruit.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

2.3.3. Algorithmes d'apprentissage semi-supervisé

Dans l'apprentissage semi-supervisé, l'algorithme utilise une petite quantité de données étiquetées et une grande quantité de données non étiquetées.

1. Algorithmes d'auto-apprentissage [Self-training Algorithms]

Dans ces méthodes, le modèle est d'abord formé sur les données étiquetées, puis étiquette de manière itérative les données non étiquetées pour améliorer les performances du modèle.

2.3.4. Algorithmes d'apprentissage par renforcement

L'apprentissage par renforcement consiste à apprendre en interagissant avec un environnement, où l'agent apprend à prendre des mesures pour maximiser les récompenses cumulatives.

1. Q-Learning

- **Objectif** : Apprendre des politiques optimales pour les problèmes de prise de décision séquentielle.
- **Description** : Un algorithme basé sur la valeur où l'agent apprend la valeur de chaque action dans un état donné en maximisant les récompenses futures.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

2.3.4. Algorithmes d'apprentissage par renforcement

2. Deep Q-Networks (DQN)

- **Objectif** : Version avancée de Q-Learning qui fonctionne bien pour les environnements complexes.
- **Description** : Combine le Q-learning avec l'apprentissage profond[Deep Learning] en utilisant des réseaux neuronaux pour estimer les valeurs Q, souvent utilisées dans les jeux vidéo ou la robotique.

3. Méthodes de gradient de politique [Policy Gradient Methods]

- **Objectif** : Apprendre directement la politique au lieu de la fonction de valeur.
- **Description** : L'agent apprend la distribution de probabilité des actions plutôt que d'estimer la valeur des actions. Couramment utilisé dans les espaces d'action continue.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

2.3.5. Algorithmes d'apprentissage d'ensemble [Ensemble Learning Algorithms]

L'apprentissage d'ensemble consiste à combiner plusieurs modèles d'apprentissage automatique pour améliorer les performances globales.

1. Bagging

- **Objectif** : Réduire la variance et éviter le surajustement (Overfitting).
- **Description** : Plusieurs modèles (par exemple, des arbres de décision) sont formés sur différents sous-ensembles de données, et leurs prédictions sont combinées (par exemple, par vote majoritaire ou par moyenne) pour améliorer les performances.

2. Boosting

- **Objectif** : Réduire les biais et améliorer la précision.
- **Description** : Les apprenants faibles (modèles qui fonctionnent légèrement mieux que les devinettes aléatoires) sont formés séquentiellement, chaque nouveau modèle se concentrant sur la correction des erreurs des modèles précédents.
- **Exemples** : **AdaBoost**, **Gradient Boosting Machines** (GBM), **XGBoost**.

3. Stacking

- **Objectif** : Combiner les points forts de différents modèles.
- **Description** : Plusieurs modèles sont entraînés, et leurs prédictions sont utilisées comme entrées dans un méta-modèle de niveau supérieur, qui apprend à combiner les prédictions.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.3 Introduction aux algorithmes

2.3.6. Résumé des algorithmes par catégorie

Catégorie	Algorithmes
Apprentissage supervisé	Linear Regression, Logistic Regression, SVM, k-NN, Random Forest, Decision Trees, Naïve Bayes
Apprentissage non supervisé	k-Means, Hierarchical Clustering, PCA
Apprentissage semi-supervisé	Q-Learning, DQN, Policy Gradients
Apprentissage d'ensemble	Bagging, AdaBoost, XGBoost, Stacking

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

- ❑ La prédiction d'une réponse à l'aide d'une ou de plusieurs caractéristiques est une méthode de prédiction de la variable dépendante (Y) en fonction des valeurs des variables indépendantes (X).
- ❑ On suppose que les deux variables sont linéairement liées. Par conséquent, nous essayons de trouver une fonction linéaire qui prédit la réponse en fonction de la caractéristique ou de la variable indépendante (x).

Les équations mathématiques qui décrivent une régression linéaire simple et régression linéaire multiple sont présentées dans les équations suivantes:

Simple
Linear
Regression

$$y = b_0 + b_1 * x_1$$

Multiple
Linear
Regression

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + \dots + b_n * x_n$$

$$\begin{array}{ccccccc} & & \text{Constant/Intercept} & & \text{Independent} & & \\ & & \downarrow & & \text{Variable} & & \\ & & \beta_0 & + & \beta_1 & X_i & \\ \uparrow & & & & \uparrow & & \\ \text{Dependent} & & & & \text{Slope/Coefficient} & & \\ \text{Variable} & & & & & & \end{array}$$

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire multiple [Multiple Linear Regression]

□ L'équation mathématique qui décrit une régression linéaire multiple est présentée dans l'équation suivante:

$$y = \alpha + \beta_1(x_1) + \beta_2(x_2) + \dots + \beta_n(x_n)$$

Diagram illustrating the components of the Multiple Linear Regression equation:

- y : Predicted value
- α : Bias
- β_1 : Weight 1
- x_1 : Feature 1
- β_2 : Weight 2
- x_2 : Feature 2
- β_n : Weight n
- x_n : Feature n

Dans cette équation :

y est la valeur prédite ou variable dépendante

x est les caractéristiques ou variable indépendante

α est le biais

β est le poids de chaque caractéristique

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

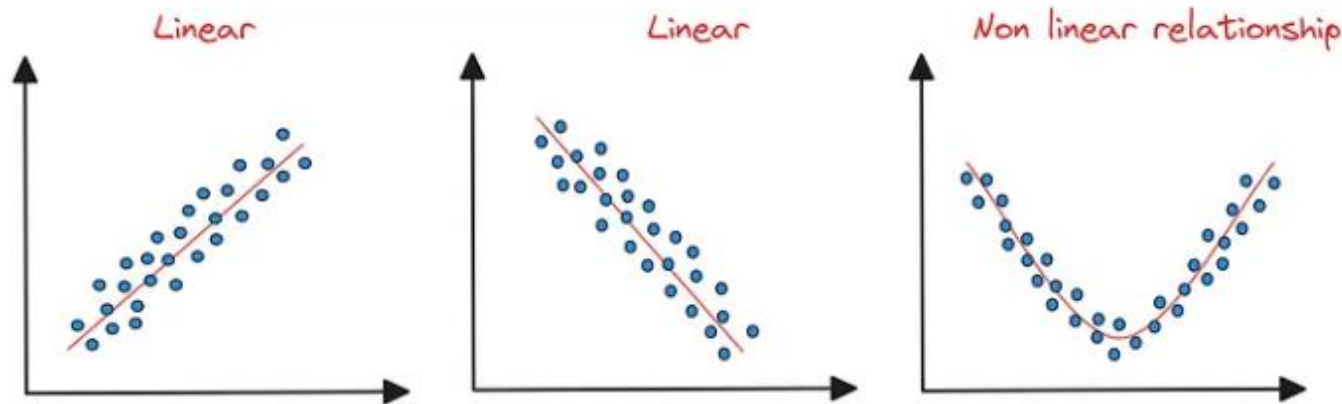


2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire multiple [Multiple Linear Regression]

❑ Remarque importante !

- ❑ Il existe plusieurs hypothèses importantes pour effectuer une analyse de régression.
- ❑ Certaines des hypothèses que vous devez confirmer sont les suivantes :
 - ✓ **Linéarité** : la relation entre la variable dépendante et la variable indépendante doit être linéaire. Autrement dit, chaque changement d'unité dans la valeur de la variable indépendante entraîne le même changement dans la variable dépendante.



Relation entre les variables : linéaire (corrélation positive), linéaire (corrélation négative), relation non linéaire

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

❑ L'équation mathématique qui décrit une régression linéaire simple est présentée dans l'équation (1).

Variable Dépendante

$$y = b_0 + b_1 x_1 \quad (1)$$

variables indépendantes

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées

Formulation du probleme :

Bias

Feature [variables indépendante]

$$Score = b_0 + b_1 hours$$

Valeur predate [variables dépendante]

Weight [Coefficient]

Dataset

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées

Nous allons suivre les étapes de prétraitement des données, puis construire le modèle de régression linéaire simple.
Les étapes sont les suivantes :

1. Importer les bibliothèques

```
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
```

- ❖ Nous importons trois bibliothèques essentielles en Python qui sont couramment utilisées pour l'analyse, la manipulation et la visualisation des données.
- ❖ Nous connaissons déjà numpy et pandas. À la ligne 3, nous importons la bibliothèque matplotlib, qui est une bibliothèque de traçage pour le langage de programmation Python.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées

```
1 dataset = pd.read_csv("../Data/studentscores.csv")
2 dataset.head(5)
```

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30

- **pd.read_csv** : cette fonction lit un fichier CSV (comma-separated values) dans un DataFrame.
- **"../Data/studentscores.csv"** : le chemin d'accès au fichier CSV. Le chemin relatif ../Data/ signifie que le fichier se trouve dans le répertoire Data, un niveau au-dessus du répertoire de travail actuel.
- **dataset.head(5)** : cette méthode renvoie les cinq premières lignes du DataFrame. Si vous omettez l'argument, les cinq premières lignes sont affichées par défaut.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées

3. Vérifier les données manquantes

```
[10]: 1 dataset.isnull().sum()
```

```
[10]: Hours      0  
      Scores    0  
      dtype: int64
```

- La méthode `dataset.isnull().sum()` permet d'identifier et de compter le nombre de valeurs manquantes dans chaque colonne de notre DataFrame.
- `dataset.isnull()` : Cette méthode renvoie un DataFrame de la même forme qu'une dataset, mais des valeurs booléennes : True lorsque les éléments du DataFrame d'origine sont NaN (manquants) et False dans le cas contraire.
- `sum()` : Lorsqu'elle est appliquée au DataFrame renvoyé par `isnull()`, cette méthode compte le nombre de valeurs True dans chaque colonne. Essentiellement, elle additionne le nombre de valeurs manquantes pour chaque colonne du DataFrame.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées

4. Diviser l'ensemble de données

❑ Avant de diviser l'ensemble de données, nous devons séparer les variable indépendantes (Feature: X) et la variable dépendante (Target: y).

```
[3]: 1 X = dataset.iloc[ : , :1].values  
     2 y = dataset.iloc[ : , 1].values
```

```
[4]: 1 from sklearn.model_selection import train_test_split  
     2  
     3 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

X : l'ensemble de fonctionnalités (données d'entrée).

y : les étiquettes cibles (données de sortie).

- **train_test_split** : une fonction qui divise les tableaux ou les matrices en sous-ensembles aléatoires d'entraînement et de test.
- **test_size=0.2** : cela signifie que 20 % des données seront utilisées pour les tests et 80 % pour l'entraînement.
- **random_state=0** : garantit que la division est reproductible. La même valeur `random_state` produira toujours la même division.
- **X_train, X_test** : il s'agit des sous-ensembles de l'ensemble de fonctionnalités pour l'entraînement et les tests, respectivement.
- **y_train, y_test** : il s'agit des sous-ensembles correspondants des étiquettes cibles pour l'entraînement et les tests, respectivement.

- **dataset.iloc[:, :-1]**: Ceci sélectionne toutes les colonnes de la dataset sauf la dernière (c'est-à-dire toutes les colonnes de Features : X).
- **.values**: Convertit les colonnes sélectionnées (généralement au format DataFrame) en un tableau NumPy.
- **dataset.iloc[:, -1]**: Ceci sélectionne la dernière colonne de la dataset (qui est souvent la colonne cible ou d'étiquette : y).

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées

5. Ajustement du modèle de régression linéaire simple aux données d'entraînement

```
1 from sklearn.linear_model import LinearRegression
2
3 simple_linear_regression = LinearRegression()
4 simple_linear_regression = simple_linear_regression.fit(X_train, y_train)
```

- **from sklearn.linear_model import LinearRegression** : Ceci importe la classe **LinearRegression** du module **sklearn.linear_model**. Cette classe est utilisée pour effectuer une régression linéaire, qui est une méthode pour modéliser la relation entre une variable dépendante et une ou plusieurs variables indépendantes.
- **LinearRegression()** : Cela crée une instance de la classe **LinearRegression**, initialisant un nouveau modèle de régression linéaire.
- **simple_linear_regression** : Il s'agit de la variable qui stocke l'instance du modèle de régression linéaire.
- **simple_linear_regression.fit(X_train, y_train)** : Cette méthode entraîne le modèle de régression linéaire à l'aide des données d'entraînement. Elle trouve la ligne la mieux ajustée qui minimise l'erreur quadratique moyenne entre les valeurs prédites et les valeurs réelles dans les données d'entraînement.
- **X_train** : Les données d'entraînement pour les caractéristiques (variables indépendantes).
- **y_train** : Les données d'entraînement pour la variable cible (variable dépendante).

La méthode d'ajustement ajuste les paramètres du modèle (coefficients) en fonction des données d'entraînement.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées.

5. Prédire les résultats

```
1 y_pred = simple_linear_regression.predict(X_test)
```

Le code `y_pred = simple_linear_regression.predict(X_test)` est utilisé pour faire des prédictions sur les données de test en utilisant le modèle de régression linéaire entraîné.

- `simple_linear_regression.predict(X_test)` : Cette méthode utilise le modèle de régression linéaire formé (stocké dans `simple_linear_regression`) pour prédire les valeurs cibles des données de test (`X_test`).
- `y_pred` : Les valeurs prédites sont stockées dans la variable `y_pred`.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées.

5. Visualisation des résultats d'entraînement

```
1 plt.scatter(X_train, y_train, color='red') # Training Data
2 plt.plot(X_train, simple_linear_regression.predict(X_train), color='blue')
3 plt.title('Score Vs Hours (Trainig set)')
4 plt.xlabel("Score")
5 plt.ylabel('Hours')
6 plt.show()
```

Nous avons créé un scatter des points de données d'entraînement ainsi que la ligne de régression prédite par le modèle de régression linéaire entraîné. Expliquons maintenant chaque ligne de codes :

- **plt.scatter(X_train, y_train, color='red')** : cette ligne crée un nuage de points des données d'entraînement.
- **X_train** : les valeurs caractéristiques (par exemple, les heures d'étude).
- **y_train** : les valeurs cibles correspondantes (par exemple, les scores).
- **color='red'** : définit la couleur des points de données en rouge.
- **plt.plot(X_train, simple_linear_regression.predict(X_train), color='blue')** : cette ligne trace la ligne de régression prédite par le modèle de régression linéaire entraîné.
- **simple_linear_regression.predict(X_train)** : valeurs cibles prédites à l'aide du modèle de régression linéaire.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



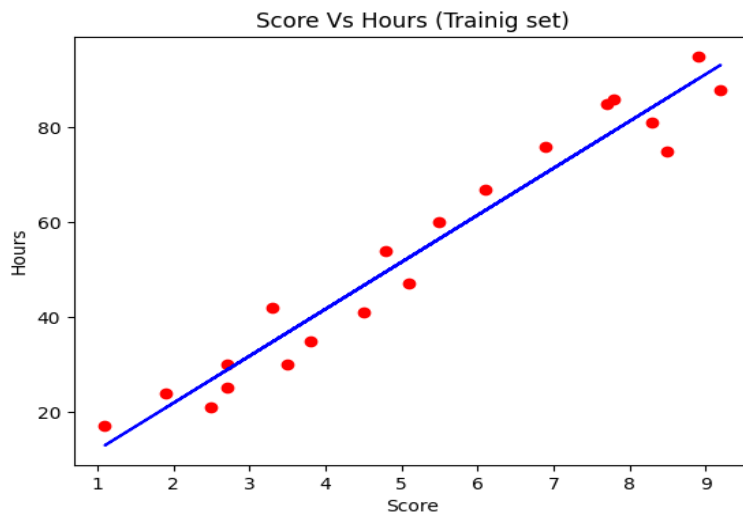
2.4 Exemple Pratique de la Prédiction

2.4.1. Régression linéaire simple [Simple Linear Regression]

Exemple 1 : Prédiction du pourcentage de notes qu'un étudiant devrait obtenir en fonction du nombre d'heures qu'il a étudiées.

5. Visualisation des résultats d'entraînement

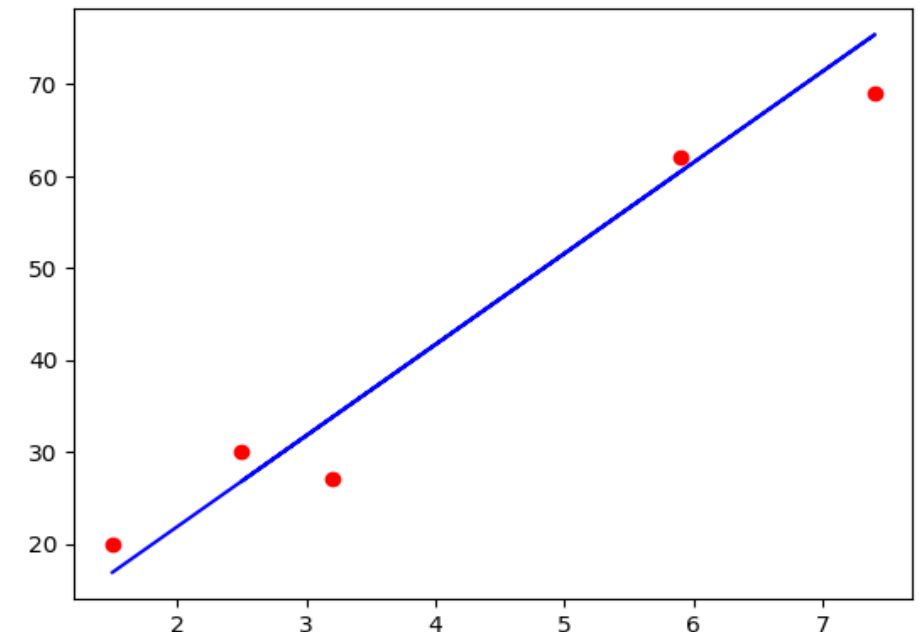
- `color='blue'` : Définit la couleur de la ligne de régression en bleu.
- `plt.title('Score Vs Hours (Training set)')` : Définit le titre du graphique.
- `plt.xlabel('Score')` : Définit l'étiquette de l'axe des x.
- `plt.ylabel('Hours')` : Définit l'étiquette de l'axe des y.
- `plt.show()` : Affiche le tracé avec les fonctionnalités spécifiées, la ligne de régression, le titre et les étiquettes.



6. Visualisation des résultats de test

```
1 plt.scatter(X_test, y_test, color='red') # Training Data
2 plt.plot(X_test, simple_linear_regression.predict(X_test), color='blue')
```

[<matplotlib.lines.Line2D at 0x2530e3d36d0>]



Regression Linear simple [Lab]

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING



2.4 Exemple Pratique de la Prédiction

2.4.2 Régression linéaire Multiple [Multiple Linear Regression]

- ❑ Dans ce chapitre, nous allons découvrir la régression linéaire multiple à l'aide de scikit-learn dans le langage de programmation Python.
- ❑ La régression linéaire multiple, souvent appelée régression multiple, est une méthode statistique qui prédit le résultat d'une variable de réponse en combinant de nombreuses variables explicatives.
- ❑ La régression multiple est une variante de la régression linéaire dans laquelle une seule variable explicative est utilisée.

Formulation Mathématique:

$$y = \alpha + \beta_1(x_1) + \beta_2(x_2) + \dots + \beta_n(x_n)$$

Diagram illustrating the components of the Multiple Linear Regression equation:

- y : Predicted value
- α : Bias
- β_1 : Weight 1
- x_1 : Feature 1
- β_2 : Weight 2
- x_2 : Feature 2
- β_n : Weight n
- x_n : Feature n

$$y = \beta_0 + \beta_1x_1 + \beta_2x_2 + \beta_3x_3 \dots$$

ici, y est la variable dépendante.

x_1, x_2, x_3, \dots sont des variables indépendantes.

b_0 = interception de la droite.

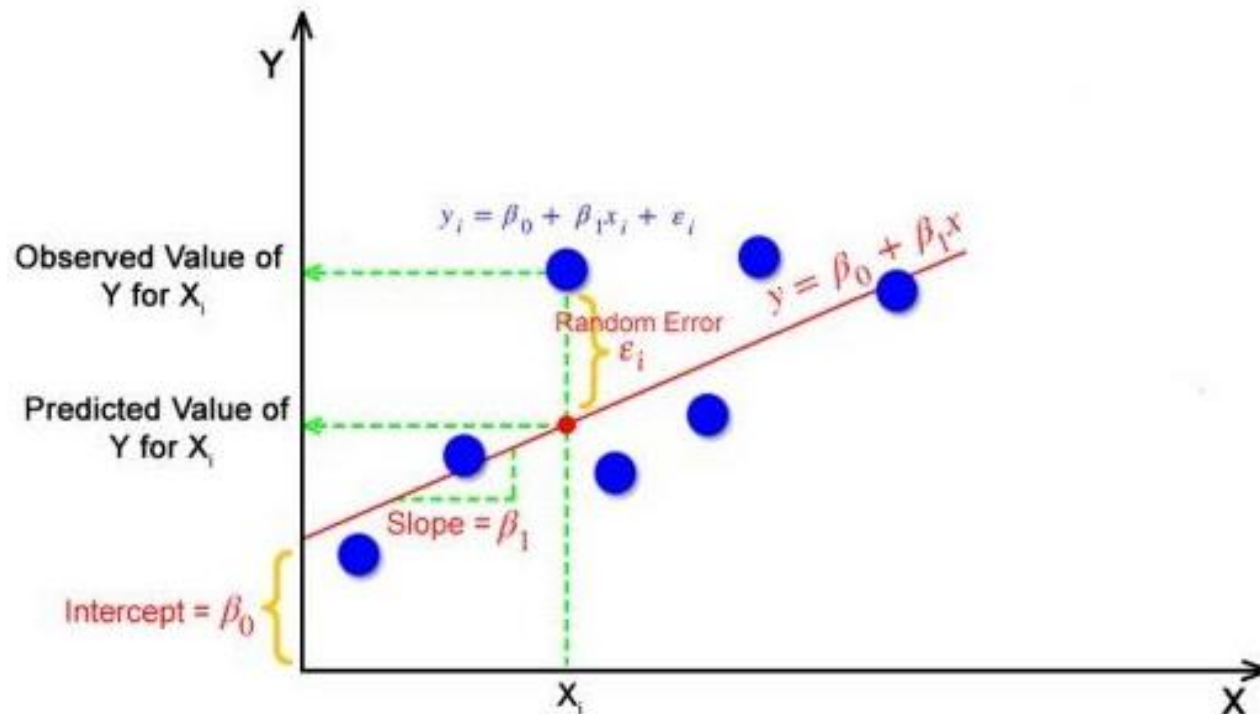
b_1, b_2, \dots sont des coefficients.

CHAPITRE 2 MACHINE LEARNING



2.4.2 Régression linéaire Multiple:

- ❑ La régression linéaire multiple, souvent appelée régression multiple, est une méthode statistique qui prédit le résultat d'une variable de réponse en combinant de nombreuses variables explicatives.



Formulation Mathématique:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \dots$$

ici, **y** est la variable dépendante.

x₁, **x**₂, **x**₃,... sont des variables indépendantes.

b₀ = interception de la droite.

b₁, **b**₂, ... sont des coefficients.

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Labo: Étapes de mise en œuvre de la régression linéaire multiple:

Étape 1 : Importer les packages nécessaires

```
1 # Importation de modules et de packages
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6 from sklearn.model_selection import train_test_split
7 from sklearn.linear_model import LinearRegression
8 from sklearn.metrics import mean_squared_error, mean_absolute_error
9 from sklearn import preprocessing
10
```


CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 2 : Importez le fichier CSV

```
1 # Importation d'un ensemble de données
2 df = pd.read_csv("../Data/Real-estate.csv")
3 df.head()
```

[93]:

	No	X1 date de transaction	X2 age de la maison	X3 distance jusqu'a la gare la plus proche	X4 nombre de supermarket de proximite	X5 latitude	X6 longitude	Y prix de la maison par unite de surface
0	1	2012.917	32.0	84.87882	10	24.98298	121.54024	37.9
1	2	2012.917	19.5	306.59470	9	24.98034	121.53951	42.2
2	3	2013.583	13.3	561.98450	5	24.98746	121.54391	47.3
3	4	2013.500	13.3	561.98450	5	24.98746	121.54391	54.8
4	5	2012.833	5.0	390.56840	5	24.97937	121.54245	43.1

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 2 : Importez le fichier CSV

```
1 df.drop('No', inplace = True, axis=1)
2 df.head()
```

`df.drop('No', inplace=True, axis=1)` :

- **'No'** : nom de la colonne à supprimer (généralement une colonne d'index ou un identifiant inutile).
- **inplace=True** : modifie le DataFrame d'origine df en place, ce qui signifie qu'aucun nouveau DataFrame n'est renvoyé ; celui existant est mis à jour.
- **axis=1** : spécifie que l'opération doit supprimer une colonne. (axis=0 supprimerait des lignes.)

	X1 date de transaction	X2 age de la maison	X3 distance jusqu'a la gare la plus proche	X4 nombre de supermarket de proximite	X5 latitude	X6 longitude	Y prix de la maison par unite de surface
0	2012.917	32.0	84.87882	10	24.98298	121.54024	37.9
1	2012.917	19.5	306.59470	9	24.98034	121.53951	42.2
2	2013.583	13.3	561.98450	5	24.98746	121.54391	47.3
3	2013.500	13.3	561.98450	5	24.98746	121.54391	54.8
4	2012.833	5.0	390.56840	5	24.97937	121.54245	43.1

```
: 1 df.shape
```

```
: (414, 7)
```

```
1 df.columns
```

```
Index(['X1 date de transaction', 'X2 age de la maison',  
      'X3 distance jusqu'a la gare la plus proche',  
      'X4 nombre de supermarket de proximite', 'X5 latitude', 'X6 longitude',  
      'Y prix de la maison par unite de surface'],  
      dtype='object')
```

CHAPITRE 2 MACHINE LEARNING



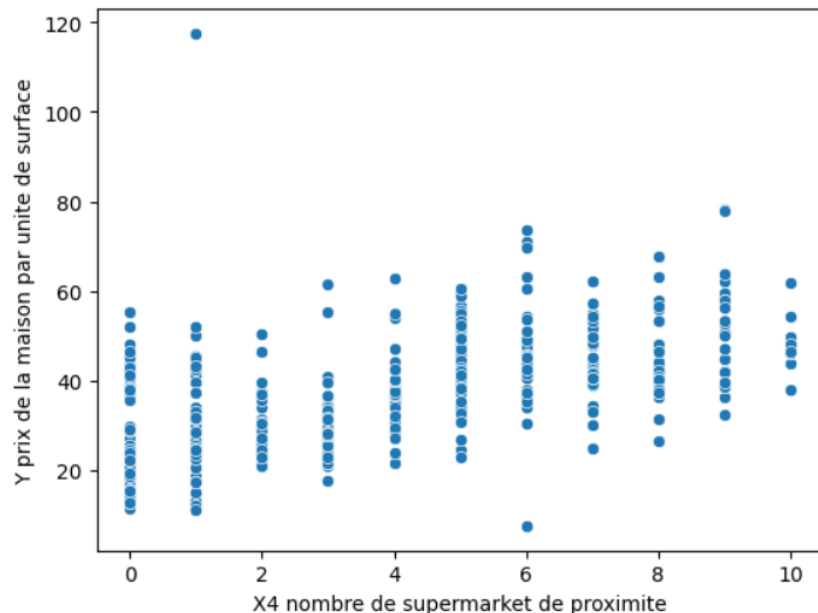
2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 3 : Créez un nuage de points pour visualiser les données

```
1 # tracer scatterplot
2 sns.scatterplot(x='X4 nombre de supermarket de proximite', y='Y prix de la maison par unite de surface', data=df)
```

[115]: <Axes: xlabel='X4 nombre de supermarket de proximite', ylabel='Y prix de la maison par unite de surface'>



- ✓ **sns.scatterplot** : il s'agit d'une fonction de Seaborn utilisée pour générer des nuages de points.
- ✓ **x='X4 nombre de supermarchés de proximité'** : l'axe des x représente le nombre de supermarchés à proximité.
- ✓ **y='Y prix de la maison par unité de surface'** : l'axe des y représente le prix de la maison par unité de surface.
- ✓ **data=df** : spécifie le DataFrame à partir duquel les données sont extraites

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 4 : Créer des variables de fonctionnalité

- ☐ Pour modéliser les données, nous devons créer des variables de caractéristiques.
- ☐ La variable X contient des variables indépendantes et la variable y contient une variable dépendante.

```
|: 1 # Création de variables de fonctionnalités
2 X = df.drop('Y prix de la maison par unite de surface',axis= 1)
3 y = df['Y prix de la maison par unite de surface']
4 (X)
```

☐ Les variables de caractéristiques X

	X1 date de transaction	X2 age de la maison	X3 distance jusqu'a la gare la plus proche	X4 nombre de supermarket de proximite	X5 latitude	X6 longitude
0	2012.917	32.0	84.87882	10	24.98298	121.54024
1	2012.917	19.5	306.59470	9	24.98034	121.53951
2	2013.583	13.3	561.98450	5	24.98746	121.54391
3	2013.500	13.3	561.98450	5	24.98746	121.54391
4	2012.833	5.0	390.56840	5	24.97937	121.54245

☐ La variables cible Y

1	y
0	37.9
1	42.2
2	47.3
3	54.8
4	43.1

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 5 : Diviser les données en données d'entraînement et de test

```
1 # Création de données d'entraînement et de données de test
2 X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.3, random_state=101)
```

```
1 print("X_train:", X_train.shape)
2 print("X_test:", X_test.shape)
3 print("y_train:", y_train.shape)
4 print("y_test:", y_test.shape)
```

X_train: (289, 6)

X_test: (125, 6)

y_train: (289,)

y_test: (125,)

❑ Paramètres :

- ✓ **test_size=0.3** : cela signifie que 30 % des données seront allouées à l'ensemble de test (X_test, y_test) et 70 % à l'ensemble d'entraînement (X_train, y_train).
- ✓ **random_state=101** : garantit la reproductibilité de la division. Le même état aléatoire produira toujours la même division lorsque vous exécuterez à nouveau le code.

❑ Résultat :

- **X_train** : 70 % des caractéristiques, qui seront utilisées pour entraîner le modèle.
- **X_test** : 30 % des caractéristiques, qui seront utilisées pour évaluer les performances du modèle sur des données non vues.
- **y_train** : 70 % des étiquettes correspondantes, qui seront utilisées pour l'entraînement.
- **y_test** : 30 % des étiquettes correspondantes, qui seront utilisées pour les tests.

X : l'ensemble de caractéristiques (données d'entrée).

y : l'ensemble cible (données de sortie).

train_test_split() : une fonction qui divise les données en sous-ensembles aléatoires d'entraînement et de test.

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 6 : Créer un modèle de régression linéaire

- ❑ La classe `LinearRegression()` est utilisée pour créer un modèle de régression multiple, la classe est importée du package `sklearn.linear_model`.

```
7 from sklearn.linear_model import LinearRegression
8 from sklearn.metrics import mean_squared_error, mean_absolute_error
```

```
1 # Création d'un modèle de régression
2 multi_reg_linear = LinearRegression()
```

Étape 7 : ajuster le modèle avec les données d'entraînement

```
: 1 # Adapter le modèle aux données d'entraînement
   2 multi_reg_linear.fit(X_train,y_train)
```

```
:  LinearRegression ⓘ ⓘ
   LinearRegression()
```

✓ **multi_reg_linear** : il s'agit de votre objet de modèle de régression linéaire multiple, qui est vraisemblablement une instance de `LinearRegression()` de scikit-learn.

✓ **.fit(X_train, y_train)** : cette méthode est utilisée pour entraîner le modèle à l'aide des données d'entraînement.

➤ Le modèle apprendra la relation entre les caractéristiques d'entrée (`X_train`) et les valeurs cibles (`y_train`).

CHAPITRE 2 MACHINE LEARNING



24.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 8 : Faire des prédictions sur l'ensemble de données de test

- ❑ Dans ce modèle, la méthode `predict()` est utilisée pour faire des prédictions sur les données `X_test`, car les données de test sont des données invisibles et le modèle n'a aucune connaissance des statistiques de l'ensemble de test.

```
1 # Faire la prédiction
2 y_pred = multi_reg_linear.predict(X_test)
```

```
1 y_pred
```

```
array([12.63830383, 10.0304461, 22.98807375, 48.50264837, 32.67140451,
       37.82572669, 36.09178068, 41.05953639, 47.84830793, 40.4574746,
       45.0361603, 32.86533457, 40.48623576, 36.48827849, 44.30595729,
       46.59668235, 38.42798244, 44.26307337, 48.81959723, 45.50409246,
       42.23260833, 54.6526397, 48.07373298, 37.48194231, 33.57091525,
       48.26293154, 40.23479801, 50.42675437, 47.22333423, 38.99458517,
       48.11033139, 40.47035604, 45.61060308, 43.98441528, 46.54336092,
       8.18725886, 38.08375879, 39.82608171, 8.5339677, 55.72740213,
       32.17950939, 49.72698264, 24.85604948, 47.64473233, 41.23026871,
       51.17703175, 42.04716292, 37.32689765, 44.24427856, 36.27028988,
       47.55408451, 34.74054504, 43.53329366, 15.95586215, 38.29001222,
       48.93912385, 44.69790471, 44.91934627, 45.33972278, 41.33035787,
       34.39899173, 44.02303164, 41.99940522, 43.9553153, 53.61420366,
       44.24994361, 24.68926603, 47.06140631, 31.22031534, 40.4930635,
       43.12220556, 48.76235412, 15.60855454, 35.80561434, 12.76370021,
```

- ✓ **multi_reg_linear.predict(X_test)** : cela appelle la méthode **.predict()** du modèle de régression linéaire entraîné (**multi_reg_linear**) et génère des valeurs cibles prédites (**y_pred**) en fonction des caractéristiques de test (**X_test**).
- ✓ **X_test** : l'ensemble de tests des caractéristiques d'entrée qui ont été retenues. Le modèle utilise ces caractéristiques pour prédire les valeurs cibles correspondantes.
- ✓ **y_pred** : les valeurs prédites générées par le modèle, qui peuvent être comparées aux valeurs cibles réelles (**y_test**) pour évaluer les performances du modèle.

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 9 : Évaluer le modèle à l'aide de mesures

- ☐ Le modèle de régression linéaire est évalué avec les mesures **mean_squared_error**, **mean_absolute_error** et **Score R²** (coefficient de détermination).
 - ☐ En comparant avec la moyenne de la variable cible, nous comprendrons dans quelle mesure notre modèle est prédictif.
- **mean_squared_error** (erreur quadratique moyenne) est la moyenne de la somme des carrés des résidus.

$$\frac{1}{n} \sum_{i=0}^n (y - \bar{y})^2$$

- **mean_absolute_error** (erreur absolue moyen) est la moyenne de la somme des valeurs absolues des résidus.

$$\frac{1}{n} \sum_{i=0}^n |y - \bar{y}|$$

- ☐ Moins l'erreur est importante, meilleures sont les performances du modèle.

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 9 : Évaluer le modèle à l'aide de mesures

- Le score R^2 (également appelé coefficient de détermination) est une mesure utilisée pour évaluer la performance d'un modèle de régression.

$$R^2 = 1 - \frac{\sum (y_{\text{true}} - y_{\text{pred}})^2}{\sum (y_{\text{true}} - \bar{y})^2}$$

$R^2 = 1$: Ajustement parfait. Le modèle prédit exactement les points de données.

$R^2 = 0$: le modèle n'explique aucune variabilité de la variable cible ; les prédictions ne sont pas meilleures que la moyenne de la cible.

$R^2 < 0$: le modèle est moins performant que la simple prédiction de la moyenne des valeurs cibles.

Cela signifie que les prédictions du modèle sont très médiocres.

Vous pouvez calculer le score R^2 à l'aide de la fonction `r2_score` de **scikit-learn** après avoir généré vos prédictions.

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 9 : Évaluer le modèle à l'aide de mesures

```
8 from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
```

```
1 # évaluation du modèle
2 mse = mean_squared_error(y_test, y_pred)
3 mae = mean_absolute_error(y_test, y_pred)
4 r2 = r2_score(y_test, y_pred)
5
6 print(f'Mean Squared Error: {mse}')
7 print(f'Mean Absolute Error: {mae}')
8 print(f'R2 score: {r2}')
```

Mean Squared Error: 46.21179783493614

Mean Absolute Error: 5.392293684756542

R² score: 0.6509058479986625

Si vous obtenez un score R² de 0,65, cela signifie que 65 % de la variabilité de la variable cible peut être expliquée par le modèle en fonction des caractéristiques d'entrée.

À l'inverse, 35 % de la variabilité est due à d'autres facteurs non pris en compte par le modèle.

Point clé à retenir :

Un R² plus élevé signifie de meilleures performances du modèle.

Un R² proche de 1 est idéal, tandis qu'une valeur négative ou 0 indique de mauvaises performances.

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 10 : Faire une simple Prediction

```
1 print(multi_reg_linear.predict([[2012.917, 32.0, 84.87882, 10, 24.98298, 121.54024]]))  
[48.05954278]
```

❑ **Remarque importante** : Notez que les valeurs des caractéristiques ont toutes été saisies entre crochets doubles. C'est parce que la méthode « **predict** » attend toujours un tableau 2D comme format de ses entrées. Et mettre nos valeurs entre crochets doubles fait de l'entrée exactement un tableau 2D.

En termes simples :

2012.917, 32.0, 84.87882, 10, 24.98298, 121.54024 → Scalaires

[2012.917, 32.0, 84.87882, 10, 24.98298, 121.54024] → Tableau 1D

[[2012.917, 32.0, 84.87882, 10, 24.98298, 121.54024]] → Tableau 2D

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 10 : Faire une simple Prediction

```
1 print(multi_reg_linear.predict([[2012.917, 32.0, 84.87882, 10, 24.98298, 121.54024]]))  
[48.05954278]
```

- ✓ **multi_reg_linear.predict()** : cette méthode permet de réaliser des prédictions avec votre modèle de régression linéaire multiple entraîné.
- ✓ **[[2012.917, 32.0, 84.87882, 10, 24.98298, 121.54024]]** : il s'agit des nouvelles données d'entrée (ensemble de caractéristiques) pour lesquelles vous souhaitez réaliser une prédiction.
- ☐ Ces valeurs correspondent probablement aux caractéristiques suivantes :
 - **Date de transaction** : 2012.917 (date de vente de la propriété).
 - **Âge de la maison** : 32,0 ans.
 - **Distance jusqu'à la station** : 84,87882 mètres.
 - **Nombre de magasins à proximité** : 10 magasins.
 - **Latitude** : 24,98298.
 - **Longitude** : 121,54024
- ✓ **.predict()** : cette méthode génère la valeur cible prédite pour les caractéristiques d'entrée que vous fournissez.
- ✓ Dans ce cas, il s'agit du prix prévu de la maison par unité de surface.

CHAPITRE 2 MACHINE LEARNING



2.4.1 Régression linéaire Multiple:

Prédire des valeurs continues (par exemple, prédire les prix des maisons)

Étape 11 : Obtenir l'équation de régression linéaire finale avec les valeurs des coefficients

- ❑ **Remarque importante** : pour obtenir ces coefficients, nous avons appelé les attributs « `coef_` » et « `intercept_` » de notre objet régresseur

```
1 print("Les coefficient de l'equation sont:",multi_reg_linear.coef_)
2 print("L'intercep est:", multi_reg_linear.intercept_)
```

```
Les coefficient de l'equation sont: [ 4.83926101e+00 -2.74749120e-01 -4.18860818e-03  1.18123112e+00
 2.42384317e+02  2.33991349e+01]
L'intercep est: -18595.055034519715
```

Par conséquent, l'équation de notre modèle de régression linéaire multiple est :

$$Y = 4,839 \times \text{Date de transaction} - 0,275 \times \text{Âge de la maison} - 0,0042 \times \text{Distance} + 1,181 \times \text{Nombre de magasins} + 242,384 \times \text{Latitude} + 23,399 \times \text{Longitude}$$

Les attributs en Python sont différents des méthodes et renvoient généralement une valeur simple ou un tableau de valeurs.

Regression Linear Multiple [Lab]