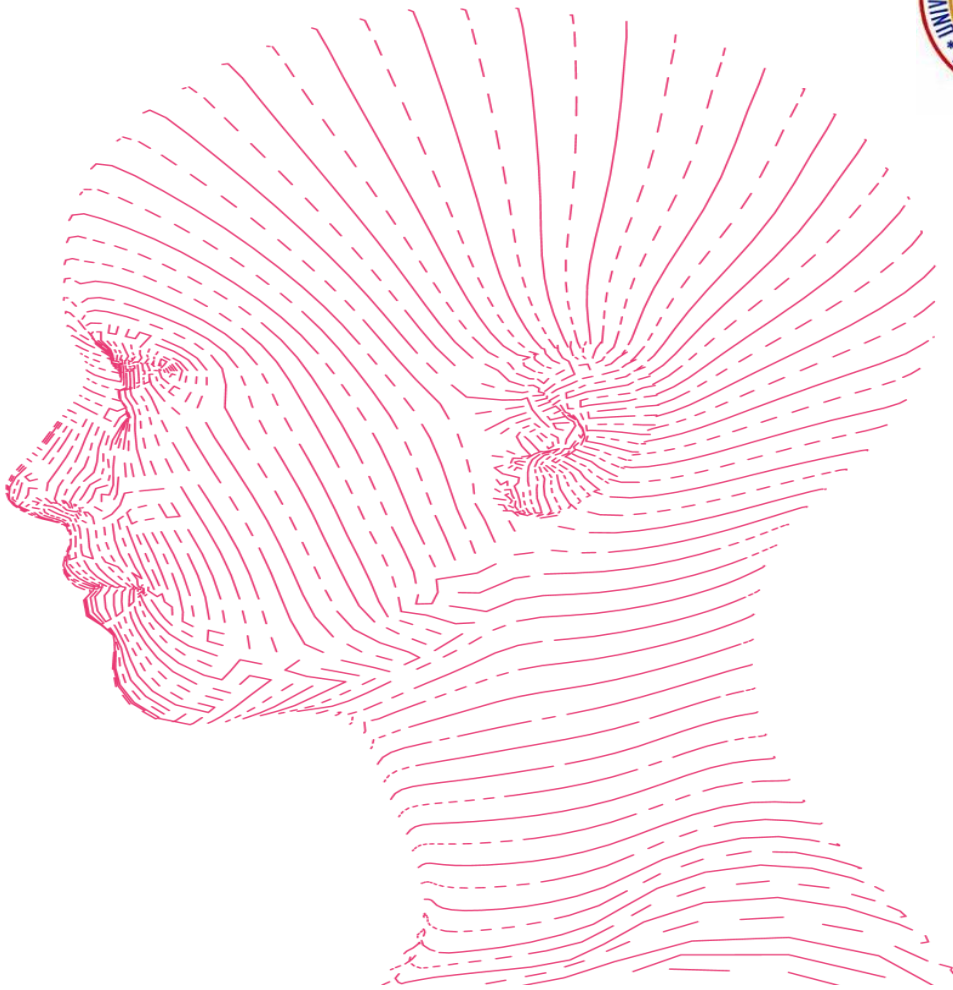
Machine Learning



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

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

Heure : 13H00 – 17H00

PLAN DU COURS

CHAPITRE 2 Concepts de base sur le Machine Learning

2.1. Terminologies d'apprentissage automatique

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

2.2 Types de données

– Catégorielles, Numériques, Textuelles, Images

– Training set (Donneés d'entrainement), Validation set (Donneés de validation), Test set (Donneés de test)

2.3 Introduction aux algorithms

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

2.4 Example Pratique de la Régression linéaire simple

CHAPITRE 2 Concepts clés sur l'apprentissage automatique [Machine Learning]

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 (Donneés d'entrainement), Validation set (Donneés de validation), Test set (Donneés de test)

2.3 Introduction aux algorithms

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

2.4 Example Pratique de la Régression linéaire simple

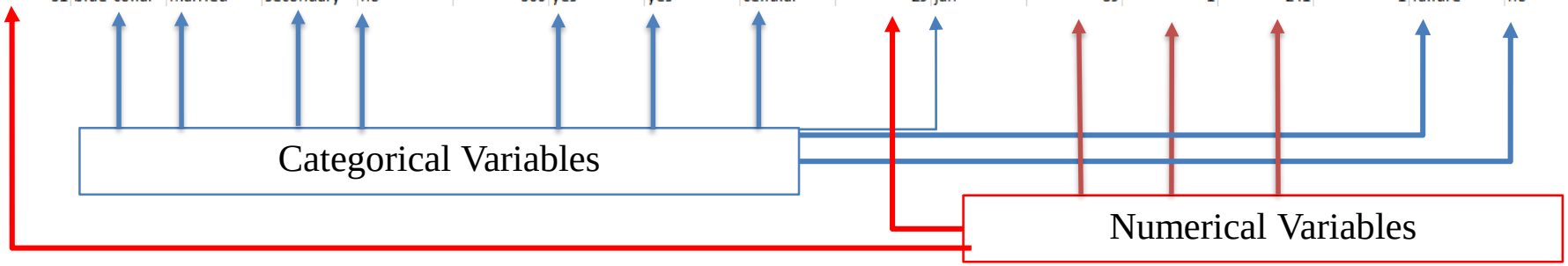
CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

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. Example 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 un algorithme de ML.

✓ **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.

✓ **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.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.1. Terminologies d'apprentissage automatique

✓ **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 qui mappera 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).

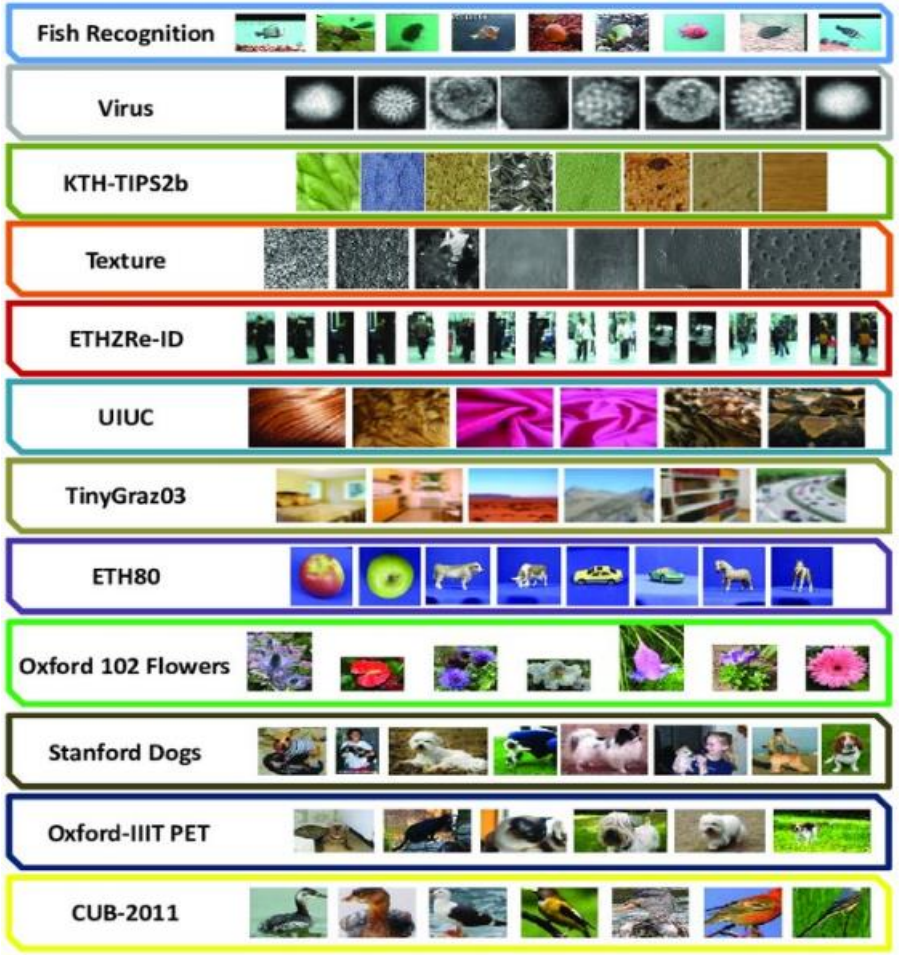
	id	bin_0	bin_1	bin_2	bin_3	bin_4	nom_0	nom_1	nom_2	nom_3	...	nom_8	nom_9	ord_0	ord_1	ord_2	ord_3	ord_4	ord_5	day	month
0	300000	0	0	1	T	Y	Blue	Triangle	Axolotl	Finland	...	9d117320c	3c49b42b8	2	Novice	Warm	j	P	be	5	11
1	300001	0	0	0	T	N	Red	Square	Lion	Canada	...	46ae3059c	285771075	1	Master	Lava Hot	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

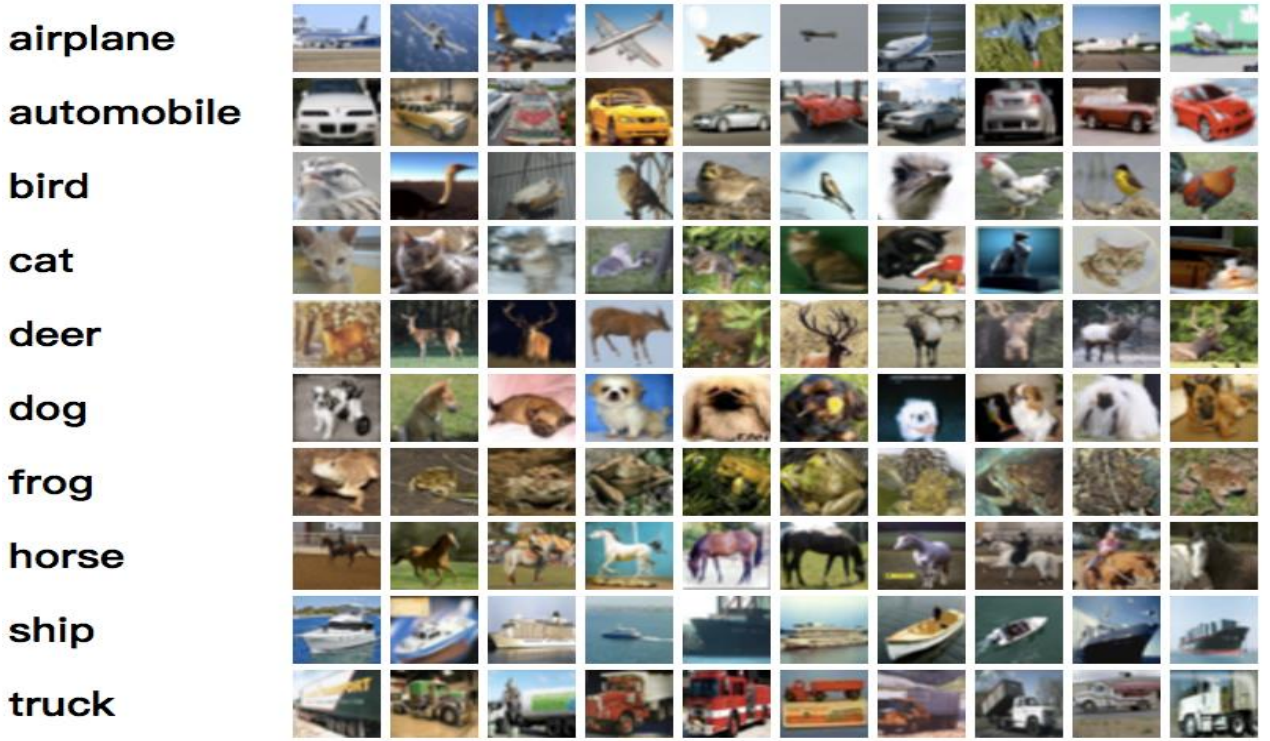
Variables numériques et catégorielles

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).





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.

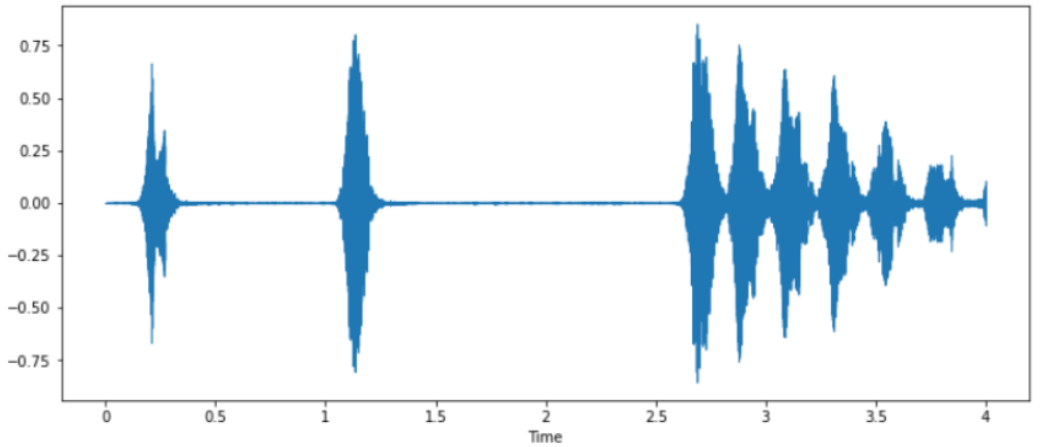
CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

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

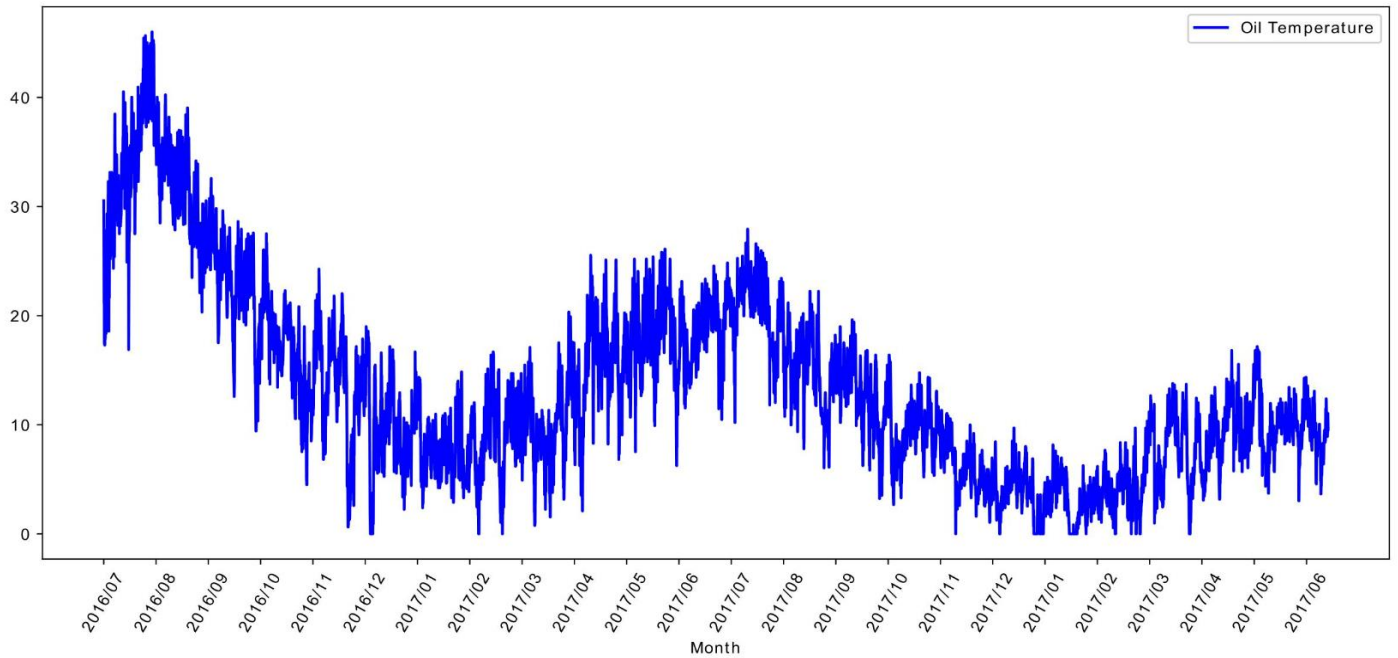
L'ensemble de données Urban Sound 8k. L'ensemble de données contient 8732 fichiers sonores de 10 classes différentes et est répertorié ci-dessous : 1. Air Conditioner, 2. Car Horn, 3. Children Playing, 4. Dog Bark, 5. Drilling Machine, 6. Engine Idling, 7. Gun Shot, 8. Jackhammer, 9.Siren, 10. Street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

- 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.

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 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 algorithms

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 algorithms

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 algorithms

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 algorithms

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 algorithms

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 algorithms

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 séquentielle ment, 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 entrainés, 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 algorithms

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 par renforcement	Q-Learning, DQN, Policy Gradients
Apprentissage d'ensemble	Bagging, AdaBoost, XGBoost, Stacking

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

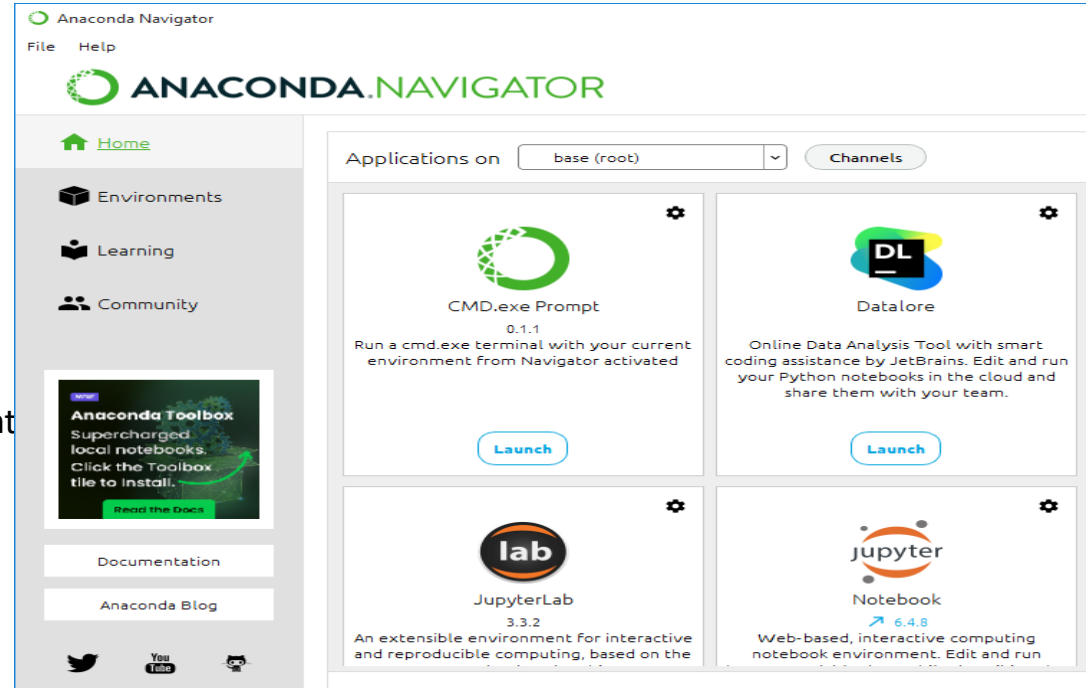
2.4 Python pour l'IA : Python Fundamentals

❑ Installer Anaconda et exécuter Jupyter Notebook

❖ Créer un environnement Anaconda à partir de Navigator

❑ Un environnement conda est un répertoire qui contient une collection spécifique de packages conda que vous avez installés.
Par exemple, vous pouvez disposer d'un environnement avec NumPy 1.7 et ses dépendances et d'un autre environnement avec NumPy 1.6 pour les tests existants.

❑ L'outil que nous utiliserons principalement est Jupyter Notebook
❑ (déjà fourni avec l'installation d'Anaconda).
❑ Il s'agit d'un bloc-notes dans lequel vous pouvez saisir et exécuter votre code ainsi qu'ajouter du texte et des notes.



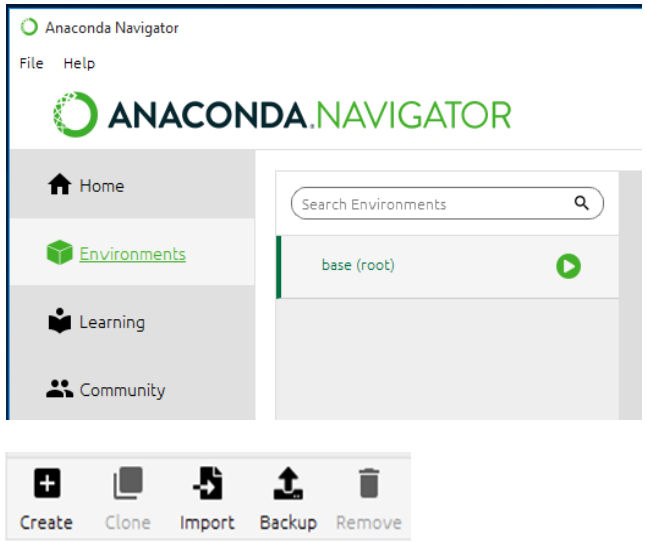
➢ **Ouvrir le navigateur Anaconda**
✓ Ouvrez Anaconda Navigator à partir du démarrage de Windows ou en le recherchant
✓ Anaconda Navigator est une application d'interface utilisateur dans laquelle vous pouvez contrôler les packages et les environnements Anaconda.

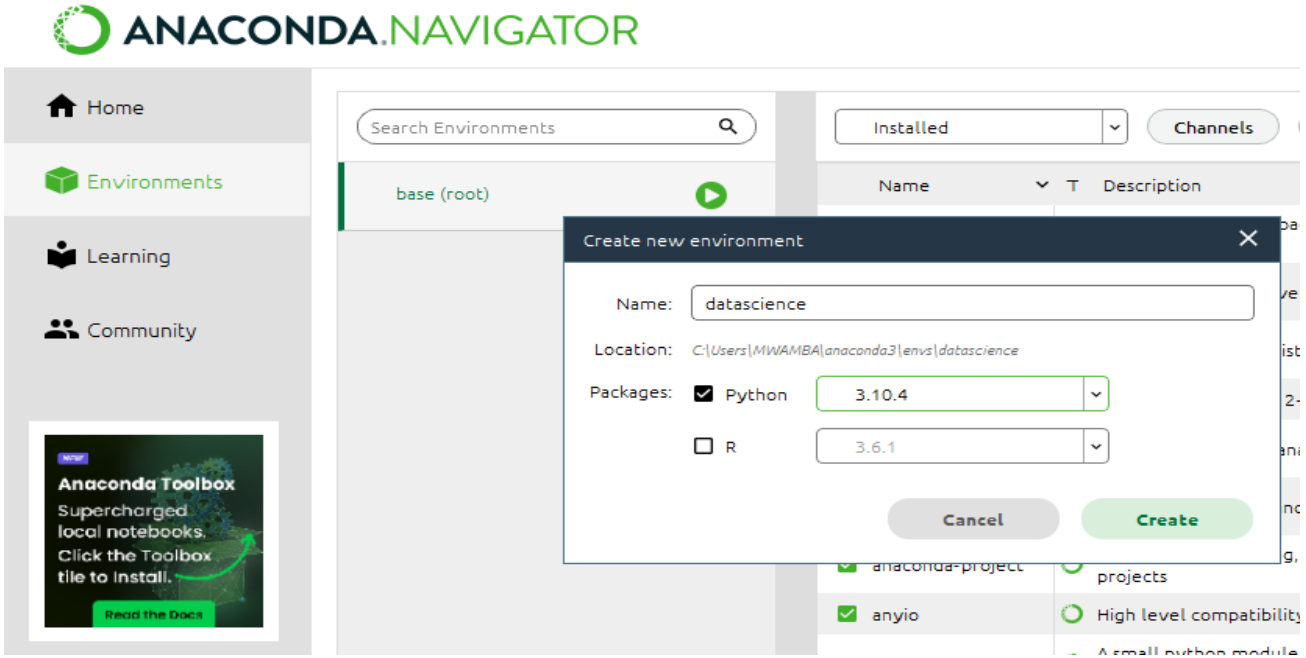
CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Installer Anaconda et exécuter Jupyter Notebook

- Ceci est facultatif mais recommandé pour créer un environnement avant de continuer.
- Cela donne une séparation complète des différentes installations de packages pour les différents projets sur lesquels vous aller travailler.
- Si vous disposez déjà d'un environnement, vous pouvez également l'utiliser sinon sélectionnez **+ Créer** l'icône en bas de l'écran pour créer un environnement Anaconda (Nom + version de python).





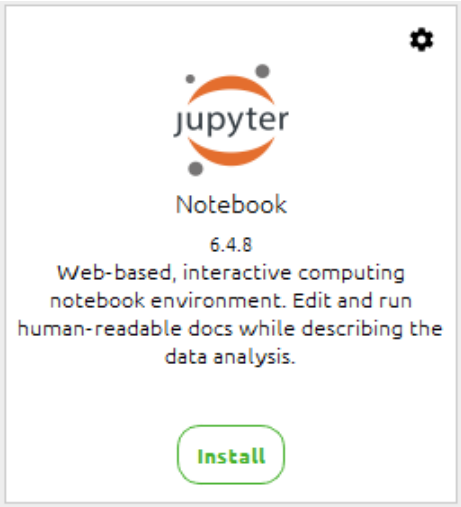
CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

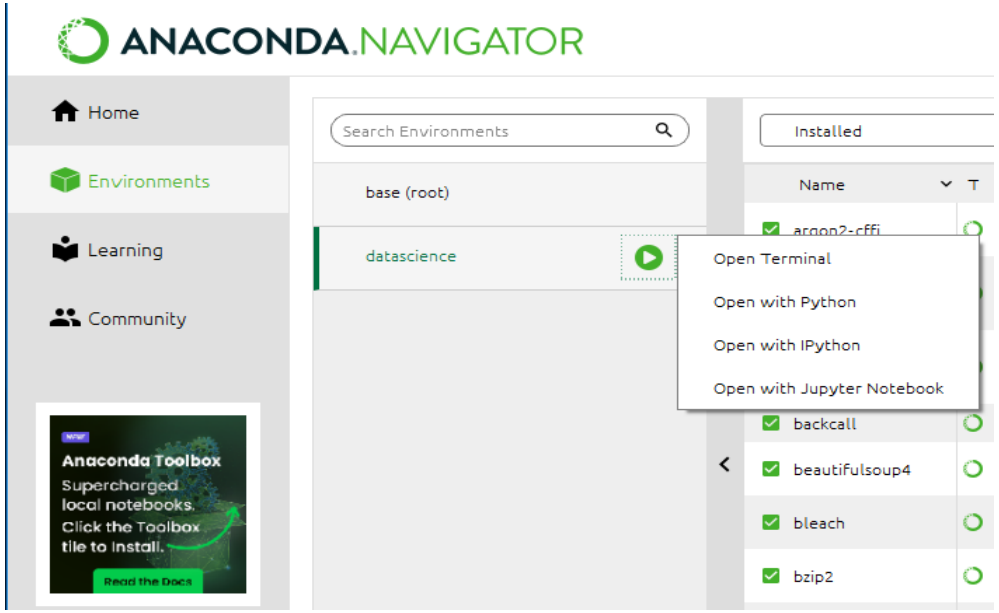
2.4 Python pour l'IA : Python Fundamentals

❑ Installer Anaconda et exécuter Jupyter Notebook

Installer et exécuter Jupyter Notebook

- Une fois que vous avez créé l'environnement Anaconda, revenez à la page d'accueil d'Anaconda Navigator et installez Jupyter Notebook à partir d'une application sur le panneau de droite. L'installation de Jupyter dans votre environnement prendra quelques secondes.





CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

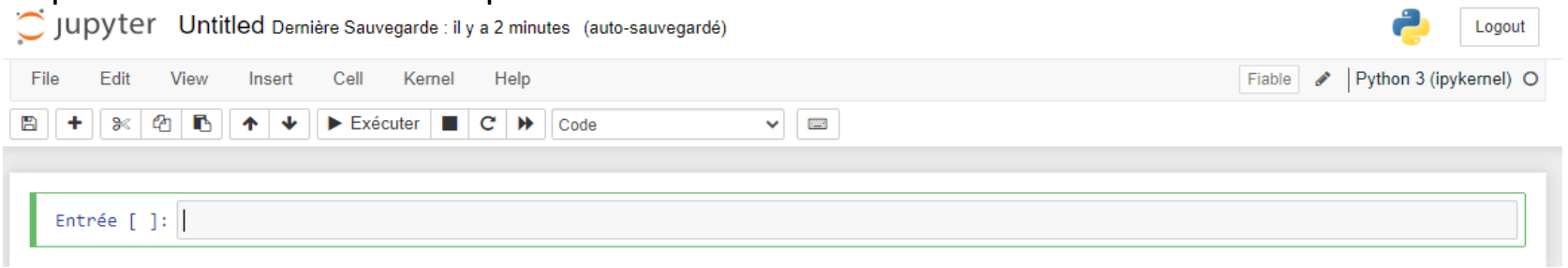
❑ Installer Anaconda et exécuter Jupyter Notebook

Installer et exécuter Jupyter Notebook

➢ Cela ouvre Jupyter Notebook dans le navigateur par défaut.

Sélectionnez maintenant Nouveau -> Python 3.

➢ Sur Jupyter, chaque cellule est une instruction, vous pouvez donc exécuter chaque cellule indépendamment lorsqu'il n'y a aucune dépendance avec les cellules précédentes.



- Ceci termine l'installation d'Anaconda et l'exécution de Jupyter Notebook.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❖ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❖ Bibliothèques Python pour l'IA

- ✓ NumPy : Manipulation de tableaux et calculs mathématiques.
- ✓ Pandas : Gestion et manipulation des datasets.
- ✓ Matplotlib & Seaborn : Visualisation des données.
- ✓ Scikit-learn : Algorithmes d'apprentissage automatique.
- ✓ Keras, TensorFlow et PyTorch : Outils pour la création et l'entraînement de réseaux de neurones

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

- Lab

1.Créer un dossier portant comme nom « Votre nom complet » a la racine C

2. Créer un environement conda avec Anaconda prompt : conda create –n bac4_SI_prenom_env python=3.10

3. Activer votre environment : conda activate bac4_SI_prenom_env

4. Verifier la liste des environement : conda env list

5. Verifier la liste des libraries déjà installed : pip list

6. Installler les libraries suivants :

pip install numpy
pip install pandas
pip install matplotlib
pip install seaborn
pip install scikit-learn -U

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❖ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❖ Bibliothèques Python pour l'IA

❖ NumPy (Numerical Python)

Pour installer: pip install numpy

https://pypi.org/project/numpy/

- ➢ **NumPy est un package fondamental pour le calcul numérique en Python.**
- ✓ Il prend en charge les tableaux multidimensionnels, les fonctions mathématiques, la génération de nombres aléatoires, l'algèbre linéaire, etc.
- ✓ Les tableaux NumPy peuvent être unidimensionnels (1D), bidimensionnels (2D) ou multidimensionnels (2D ou 3D ou 4D).

❑ Tableaux unidimensionnels (1D) :

- Les tableaux unidimensionnels sont similaires aux listes Python.
- Ils sont créés à l'aide de la fonction **numpy.array()** en passant une liste ou un tuple Python.

```
<class 'numpy.ndarray'>
[1 2 3 4]
```

[133]:
```
# Tableaux unidimensionnels (1D)

# Importer la library
import numpy as np

# Créez un tableau NumPy 1D à partir d'une liste Python
tableau_1D = np.array([1, 2, 3, 4])
print(type(tableau_1D))
print(tableau_1D)
```

```
<class 'numpy.ndarray'>
[1 2 3 4]
```

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❖ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❖ Bibliothèques Python pour l'IA

❖ NumPy (Numerical Python)

Pour installer: pip install numpy

https://pypi.org/project/numpy/

➢ NumPy est un package fondamental pour le calcul numérique en Python.

❑ Tableaux bidimensionnels (2D) :

- Les tableaux à deux dimensions sont comme des matrices ou des tableaux avec des lignes et des colonnes.
- Ils sont créés à l'aide de listes Python imbriquées ou en remodelant un tableau 1D.

```
<class 'numpy.ndarray'>
[[ 10  20  30  40  50]
 [ 60  70  80  90 100]]
(2, 5)
```

[136]:
```
# Tableaux bidimensionnels (2D)

# Importer la library
import numpy as np

# Créez un tableau NumPy 2D à partir d'une liste Python imbriquée
tableau_2D = np.array([[10, 20, 30, 40, 50],[60, 70, 80, 90, 100]])
print(type(tableau_2D))
print(tableau_2D)
print(tableau_2D.shape)
```

```
<class 'numpy.ndarray'>
[[ 10  20  30  40  50]
 [ 60  70  80  90 100]]
(2, 5)
```

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❖ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❖ Bibliothèques Python pour l'IA

❖ NumPy (Numerical Python)

Pour installer: pip install numpy

https://pypi.org/project/numpy/

➢ NumPy est un package fondamental pour le calcul numérique en Python.

❑ Tableaux multidimensionnels (3D) :

- Les tableaux multidimensionnels peuvent avoir plus de deux dimensions.

```
<class 'numpy.ndarray'>
[[[10 20 30]
  [40 50 60]
  [70 80 90]]]
(1, 3, 3)
```

- Ils sont créés à l'aide de listes Python imbriquées.

```
[137]: # Tableaux multidimensionnels (3D)

# Importer la library
import numpy as np

# Créez un tableau NumPy 3D à partir d'une liste Python imbriquée
tableau_3D = np.array([[
    [10, 20, 30],
    [40, 50, 60],
    [70, 80, 90]
]])
print(type(tableau_3D))
print(tableau_3D)
print(tableau_3D.shape)
```

```
<class 'numpy.ndarray'>
[[[10 20 30]
  [40 50 60]
  [70 80 90]]]
(1, 3, 3)
```

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ NumPy (Numerical Python)

➢ Opérations sur les tableaux Numpy

Soient 2 matrices : a et b

[150]:
```
import numpy as np

# Definir 2 matrices
a = np.array([10, 20, 30, 40, 50])

b = np.array([60, 70, 80, 90, 100])
```

[151]:
```
# 1. Addition
tableau_Add = a + b
print(tableau_Add)
```
```
[ 70  90 110 130 150]
```

[152]:
```
# 2. Soustraction
tableau_Soustrac = a - b
print(tableau_Soustrac)
```
```
[-50 -50 -50 -50 -50]
```

[153]:
```
# 3. Multiplication
tableau_Mul = a * b
print(tableau_Mul)
```
```
[ 600 1400 2400 3600 5000]
```

[154]:
```
# 4. Division
tableau_Div= b / a
print(tableau_Div)
```
```
[6.         3.5        2.66666667 2.25       2.        ]
```

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ Pandas

Pour installer: pip install pandas

https://pypi.org/project/pandas/

- ➢ **Pandas est une bibliothèque puissante pour la manipulation et l'analyse de données**.
- ✓ Il propose des structures de données telles que DataFrame et Series, qui permettent une manipulation et une manipulation faciles des données structurées.
- ✓ Pandas fournit des fonctionnalités pour le nettoyage, le remodelage [reshaping], la fusion [merging], le regroupement et l'agrégation des données.

Vous trouverez ci-dessous un exemple simple d'utilisation de pandas pour travailler avec des données tabulaires:

```
[179]: import pandas as pd
       # Créer un dictionnaire de données
       my_data = {
           "Nom":['KABAMBA', 'KASONGO', 'KAZADI', 'ILUNGA', 'KABULO'],
           "Age" : [24, 27, 28, 30, 32],
           "Ville":['Kinshasa', 'Lubumbashi', 'Goma', 'Kisangani', 'Kolwezi'],
           "Profession":['Medecin', 'Informaticien','Avocat','Professeur','Ingenieur']
       }
```

```
[180]: my_data

[180]: {'Nom': ['KABAMBA', 'KASONGO', 'KAZADI', 'ILUNGA', 'KABULO'],
        'Age': [24, 27, 28, 30, 32],
        'Ville': ['Kinshasa', 'Lubumbashi', 'Goma', 'Kisangani', 'Kolwezi'],
        'Profession': ['Medecin',
         'Informaticien',
         'Avocat',
         'Professeur',
         'Ingenieur']}
```

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ Pandas

Pour installer: pip install pandas

https://pypi.org/project/pandas/

➢ Créer un DataFrame à partir du dictionnaire

```
import pandas as pd
# Créer un dictionnaire de données
my_data = {
    "Nom":['KABAMBA', 'KASONGO', 'KAZADI', 'ILUNGA', 'KABULO'],
    "Age" : [24, 27, 28, 30, 32],
    "Ville":['Kinshasa', 'Lubumbashi', 'Goma', 'Kisangani', 'Kolwezi'],
    "Profession":['Medecin', 'Informaticien','Avocat','Professeur','Ingenieur']
}
```

```
# Créer un DataFrame à partir du dictionnaire
dataframe = pd.DataFrame(my_data)

# Afficher le dataframe
dataframe
```

```
[181]: # Créer un DataFrame à partir du dictionnaire
dataframe = pd.DataFrame(my_data)

# Afficher le dataframe
dataframe
```

[181]:

	Nom	Age	Ville	Profession
0	KABAMBA	24	Kinshasa	Medecin
1	KASONGO	27	Lubumbashi	Informaticien
2	KAZADI	28	Goma	Avocat
3	ILUNGA	30	Kisangani	Professeur
4	KABULO	32	Kolwezi	Ingenieur

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ Pandas

Pour installer: pip install pandas

https://pypi.org/project/pandas/

➢ Opérations de base avec DataFrame

▪ Accéder aux Colonnes

[182]:
```
# Opérations de base avec DataFrame
# Accéder aux colonnes

dataframe['Nom']
```

[182]:
```
0    KABAMBA
1    KASONGO
2     KAZADI
3     ILUNGA
4     KABULO
Name: Nom, dtype: object
```

[183]:
```
print(dataframe[['Nom','Age']])
```

```
      Nom  Age
0  KABAMBA   24
1  KASONGO   27
2   KAZADI   28
3   ILUNGA   30
4   KABULO   32
```

Accéder aux lignes

[184]:
```
# Accéder aux lignes
dataframe.iloc[2]
```

[184]:
```
Nom           KAZADI
Age               28
Ville           Goma
Profession     Avocat
Name: 2, dtype: object
```

[185]:
```
# Découper les lignes
dataframe.iloc[2:]
```

[185]:

	Nom	Age	Ville	Profession
2	KAZADI	28	Goma	Avocat
3	ILUNGA	30	Kisangani	Professeur
4	KABULO	32	Kolwezi	Ingenieur

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ Pandas

Pour installer: pip install pandas

➢ Opérations de base avec DataFrame

- **Filtrage des données en fonction de la condition**

```
[186]: # Filtrage des données en fonction de la condition
[187]: dataframe[dataframe['Age'] > 29]
```

[187]:

	Nom	Age	Ville	Profession
3	ILUNGA	30	Kisangani	Professeur
4	KABULO	32	Kolwezi	Ingenieur

- **Ajout d'une nouvelle colonne**

```
[189]: dataframe['Genre'] = ['M','F','M','F','M']
[190]: dataframe
```

[190]:

	Nom	Age	Ville	Profession	Genre
0	KABAMBA	24	Kinshasa	Medecin	M
1	KASONGO	27	Lubumbashi	Informaticien	F
2	KAZADI	28	Goma	Avocat	M
3	ILUNGA	30	Kisangani	Professeur	F
4	KABULO	32	Kolwezi	Ingenieur	M

- **Supprimer une colonne**

```
[222]: # Supprimer une colonne
dataframe.drop(columns=['Ville'], inplace=True)
[223]: dataframe
```

[223]:

	Nom	Age	Profession	Genre
0	KABAMBA	24	Medecin	M
1	KASONGO	27	Informaticien	F
2	KAZADI	28	Avocat	M
3	ILUNGA	30	Professeur	F
4	KABULO	32	Ingenieur	M

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ Matplotlib

Pour installer: pip install matplotlib

https://pypi.org/project/matplotlib/

- ➢ Matplotlib est une bibliothèque de traçage largement utilisée en Python pour créer des visualisations statiques, interactives et de bonne qualité.
- ✓ Il fournit une interface de type MATLAB et prend en charge divers types de tracés, notamment les tracés linéaires, les nuages de points, les tracés à barres, les histogrammes.

Example : **numpy.linspace()** est une fonction de la bibliothèque **NumPy** qui génère un tableau de nombres régulièrement espacés sur un intervalle spécifié.

- Paramètres:

- ✓ **start :** La valeur de départ de la séquence.
- ✓ **stop :** la valeur de fin de la séquence (incluse par défaut, sauf si le point final est défini sur False).
- ✓ **num :** Le nombre d'échantillons à générer (la valeur par défaut est 50).

```
import numpy as np

# Générez un tableau de 20 nombres régulièrement espacés entre 0 et 1
tableau = np.linspace(0, 1, num=20)
print(type(tableau))
print(tableau)
```

```
<class 'numpy.ndarray'>
[0.         0.05263158 0.10526316 0.15789474 0.21052632 0.26315789
 0.31578947 0.36842105 0.42105263 0.47368421 0.52631579 0.57894737
 0.63157895 0.68421053 0.73684211 0.78947368 0.84210526 0.89473684
 0.94736842 1.        ]
```

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ **Matplotlib**

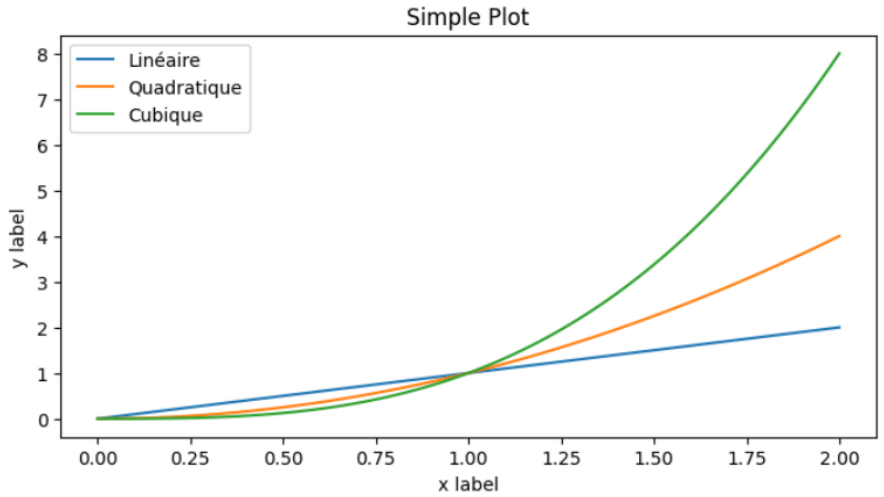
Pour installer: pip install matplotlib

https://pypi.org/project/matplotlib/

➢ Example : Nous allons tracer trois lignes représentant les fonctions linéaires, quadratiques et cubiques de x, ajouter des étiquettes aux axes, définir un titre pour le tracé et ajouter une légende pour distinguer les lignes.

```
[368]:
1  import matplotlib.pyplot as plt
2
3  x = np.linspace(0, 2, 100)
4
5  # Notez que même dans le style OO, nous utilisons `.pyplot.figure` pour créer la figure.
6  fig, ax = plt.subplots(figsize=(8, 4)) # largeur, hauteur
7  ax.plot(x, x, label='Linéaire') # Tracez des données sur les axes.
8  ax.plot(x, x**2, label='Quadratique')
9  ax.plot(x, x**3, label='Cubique')
10 ax.set_xlabel('x label') # Ajoute une étiquette x aux axes.
11 ax.set_ylabel('y label') # Ajoute une étiquette y aux axes.
12 ax.set_title('Simple Plot')
13 ax.legend()
```

```
[368]: <matplotlib.legend.Legend at 0x1f705478640>
```



CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Python pour l'IA : Python Fundamentals

❑ Quelques bases de python

Les bases du langage Python, les structures de données (listes, tuples, dictionnaires), les fonctions, les boucles et les conditions [Prérequis].

❑ Bibliothèques Python pour l'IA

❖ Seaborn

Pour installer: pip install matplotlib

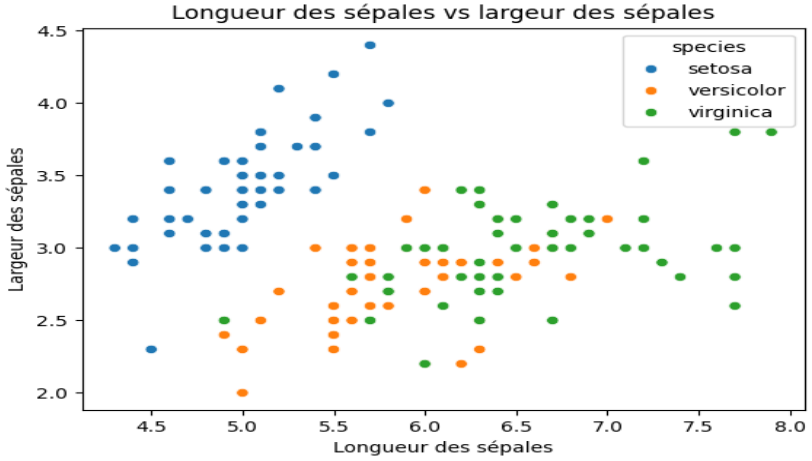
https://pypi.org/project/seaborn/

- ➢ Seaborn est une bibliothèque de visualisation de données statistiques basée sur Matplotlib.
- ✓ Il fournit une interface de haut niveau pour créer des graphiques statistiques attrayants et informatifs.
- ✓ Seaborn simplifie la création de visualisations complexes telles que des tracés catégoriels, des tracés de distribution et des tracés de régression.

[376]:

```
import seaborn as sns
import matplotlib.pyplot as plt
# Charger un exemple d'ensemble de données depuis Seaborn
iris = sns.load_dataset('iris')
# Créez un nuage de points en utilisant Seaborn
sns.scatterplot(x='sepal_length', y='sepal_width', data=iris, hue='species')
# Ajouter des étiquettes et un titre
plt.xlabel('Longueur des sépales')
plt.ylabel('Largeur des sépales')
plt.title('Longueur des sépales vs largeur des sépales')
# Affichage
plt.show()

```



TD Numero 2

1. C’est quoi l’Intelligence Artificielle

2. Expliquez ces commandes :

✓ conda create –n bac4SI python=3.10

✓ conda activate bac4SI

✓ pip list

✓ 3. A quoi sert

✓ import numpy as np

✓ Numpy, Pandas, matplotlib, searborn,

4. Numpy.linspace(0, 2, 10)

5. iloc et drop

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Example 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).

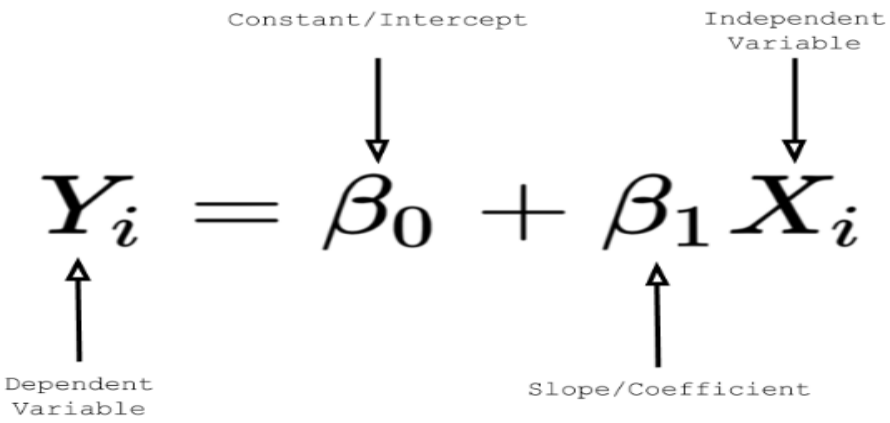
Les équations mathématique qui décrivent une régression linéaire simple et régression linéaire multiple sont présentée dans les équation suivante:

Simple Linear Regression

$$y = b_0 + b_1 * x_1$$

Multiple Linear Regression

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

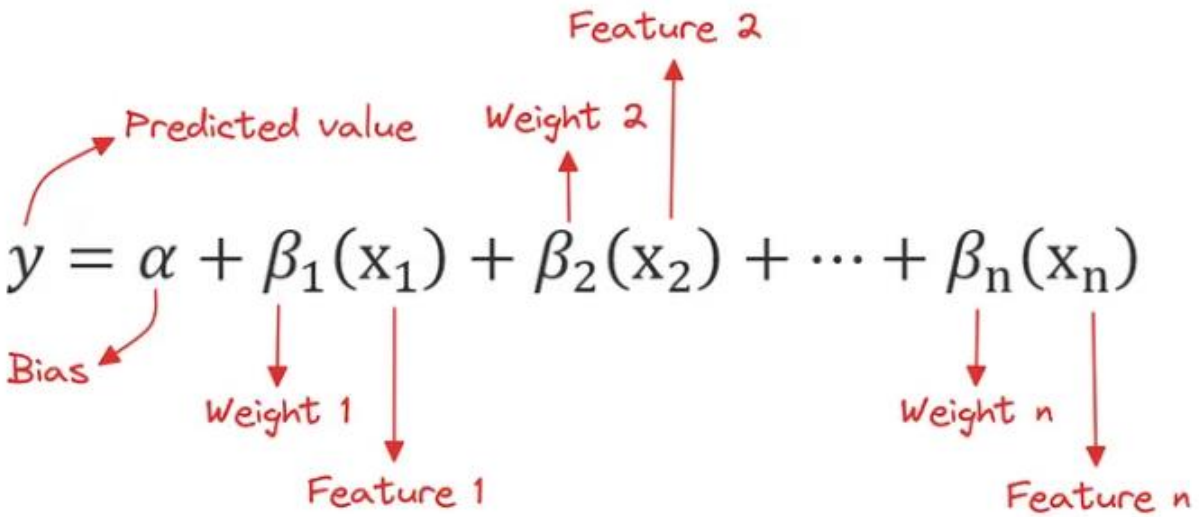


CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Example 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:



Dans cette équation :

y est la valeur prédite ou variable dépendante
x est les caractéristiques ou variable indépendante
α est le biais :
C'est une valeur ajoutée à la combinaison linéaire des entrées pondérées pour ajuster l'activation du neurone.
β est le poids de chaque caractéristique

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

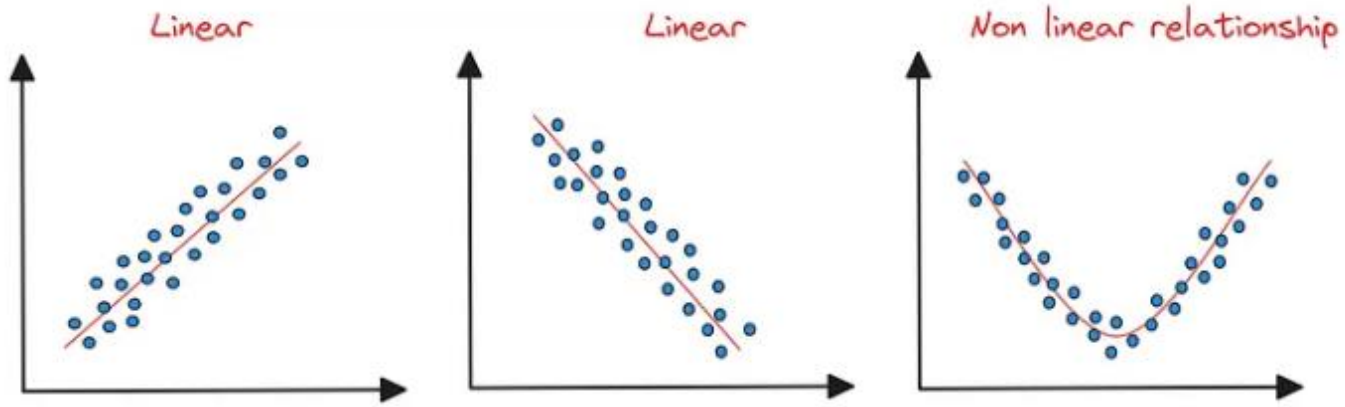
2.4 Example Pratique de la Prédiction

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 Example 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 Example 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 Example 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 Example 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 da ns chaque colonne de notre DataFrame.
- **dataset.isnull()** : Cette méthode renvoie un DataFrame de la même forme qu'une dataset, mais des valeur s 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 Example 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]:
```
X = dataset.iloc[ : , :1].values
y = dataset.iloc[ : , 1].values
```

[4]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,  test_size=0.2, random_state=0)
```

- **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 Example 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

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.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Example 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.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Example 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

- **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.

Cours-ML-Dahouda M., Ph.D.

CHAPITRE 2 CONCEPTS CLES SUR MACHINE LEARNING

2.4 Example 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 entrainé.

- **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 Example 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

```
plt.scatter(X_train, y_train, color='red') # Traiing Data
plt.plot(X_train, simple_linear_regression.predict(X_train), color='blue')
plt.title('Score Vs Hours (Trainig set)')
plt.xlabel("Score")
plt.ylabel('Hours')
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

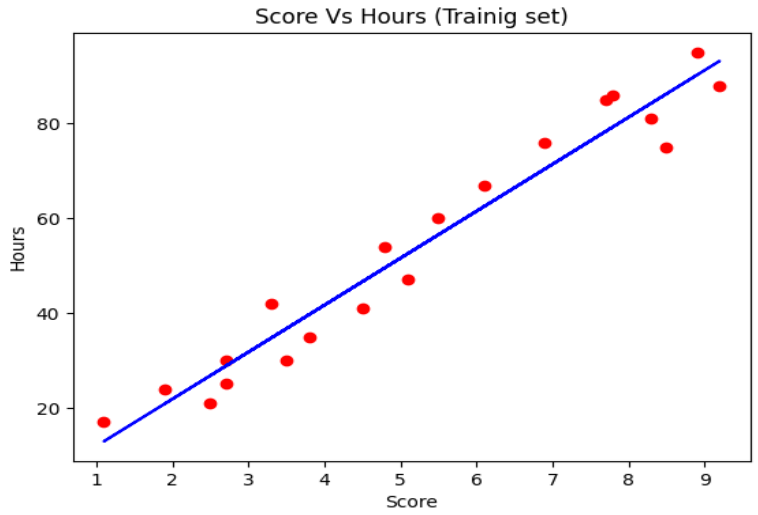
2.4 Example Pratique de la Prédiction

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') # Traiing Data
2 plt.plot(X_test, simple_linear_regression.predict(X_test), color='blue')
```

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