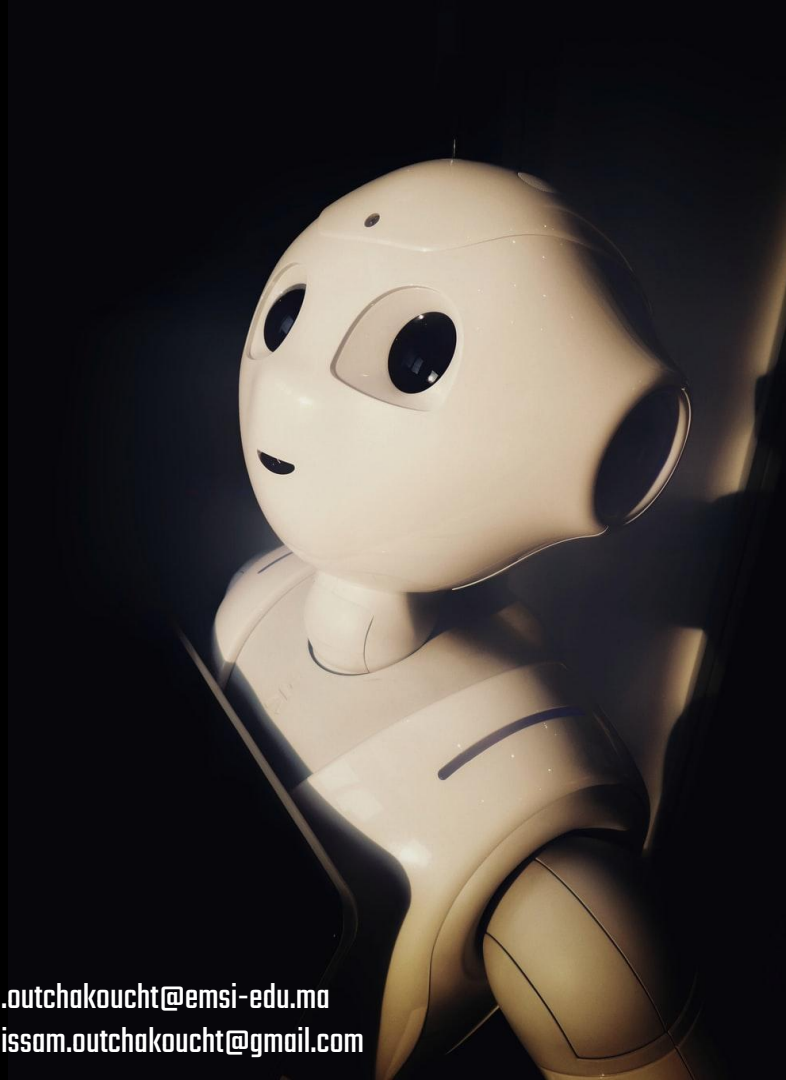
ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

ML / DL

MACHINE LEARNING: INTRODUCTION



Aissam Outchakoucht

a.outchakoucht@emsi-edu.ma
aissam.outchakoucht@gmail.com

TAKEAWAY N° 1 : L'IA d'aujourd'hui n'en est qu'à ses débuts.

TAKEAWAY N° 2 : L'IA n'a rien de magique, mais elle est extrêmement pratique

TAKEAWAY N° 3 : NumPy : Optimisé pour les calculs numériques.

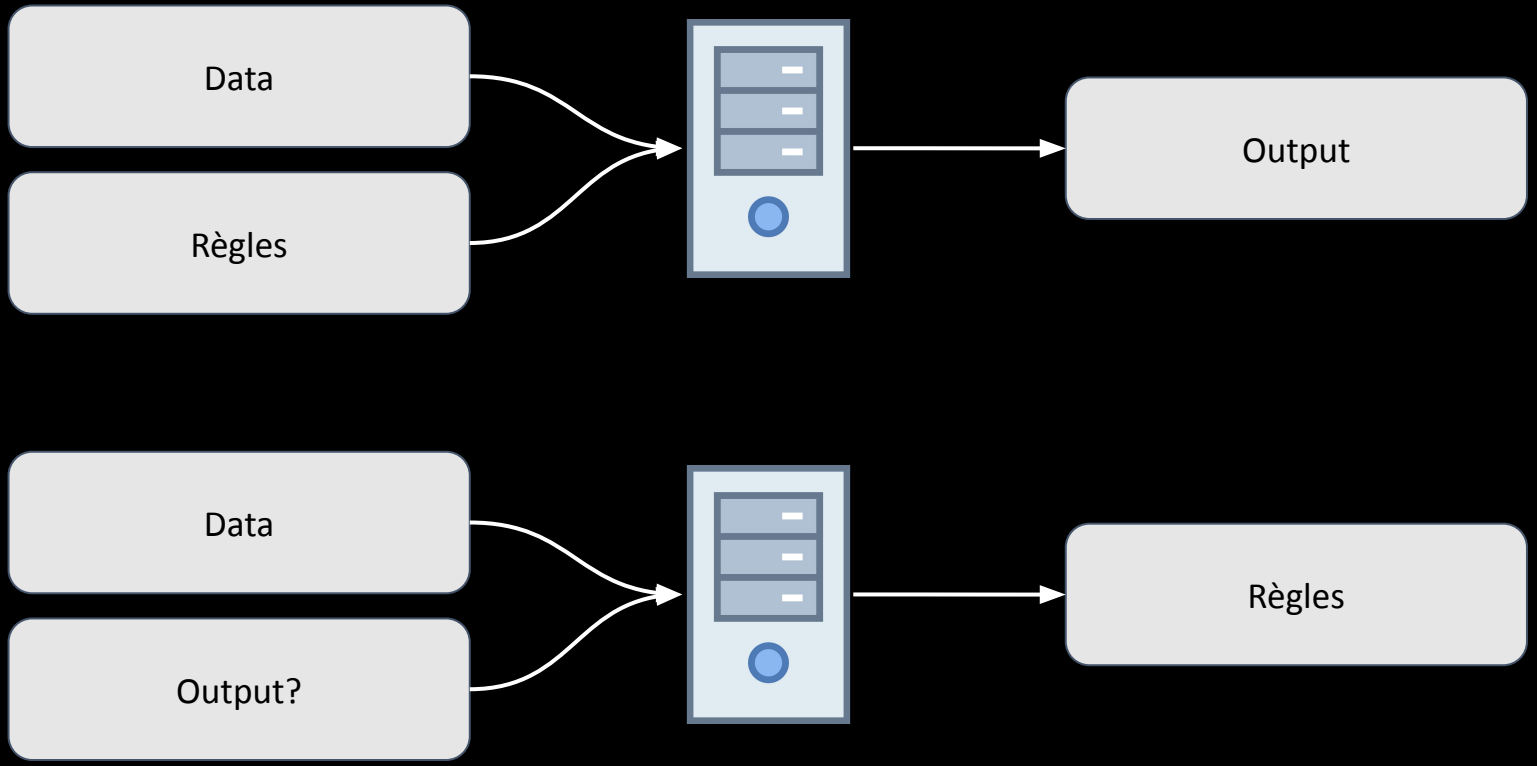
TAKEAWAY N° 4 : Pandas : Simplifie l'analyse et la manipulation des données.

ML ?

“L'apprentissage automatique est un domaine d'étude qui donne aux ordinateurs la capacité d'apprendre sans être explicitement programmés".

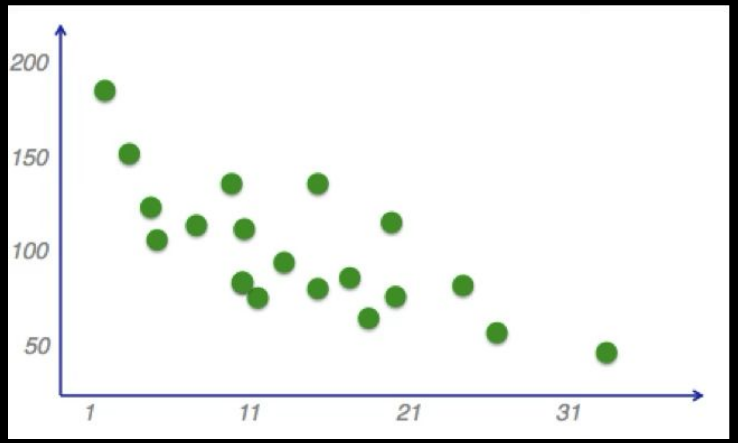
Arthur Samuel (1959)

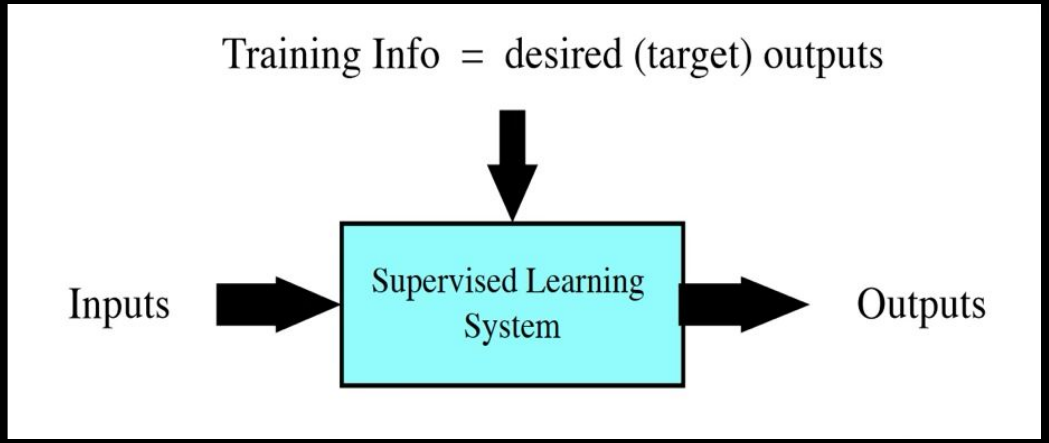
PROG. TRAD. VS ML



LES 3 CATÉGORIES DE ML (ET DL)

* L'apprentissage supervisé

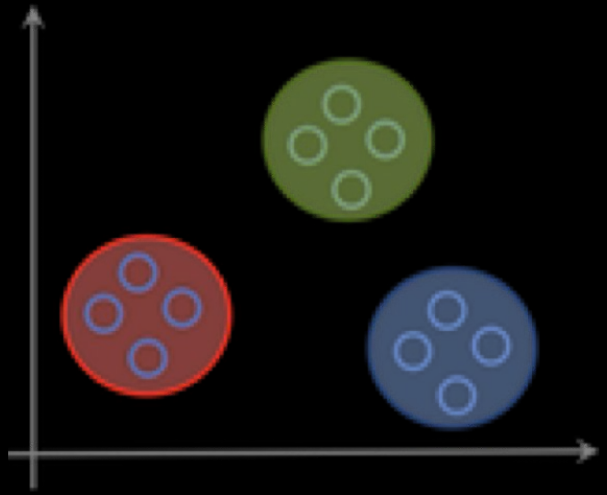


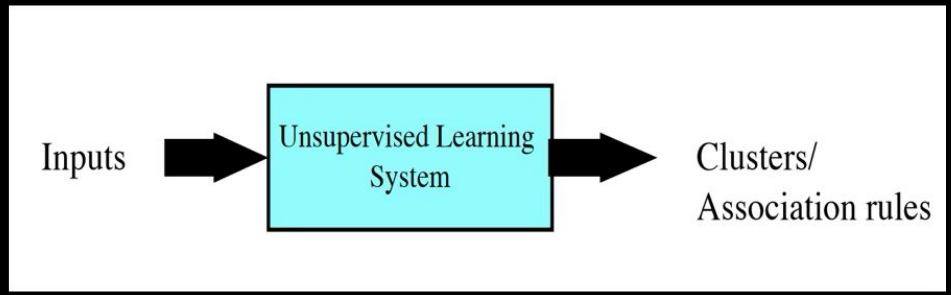



```
Input  : (x, f(x))
Output : f
```

LES 3 CATÉGORIES DE ML (ET DL)

* L'apprentissage non supervisé

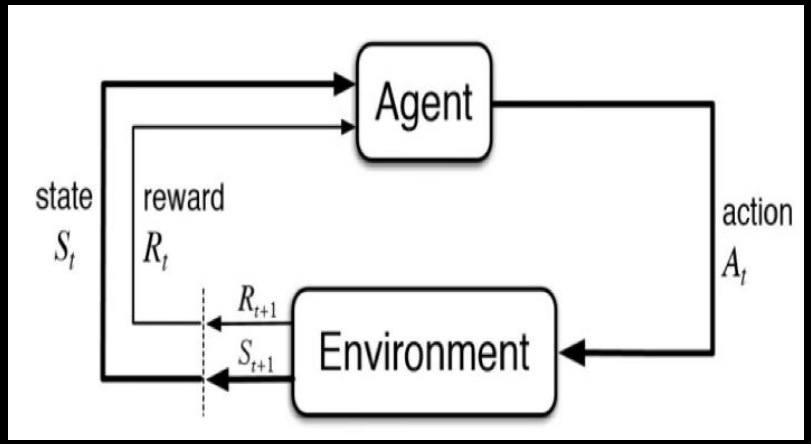


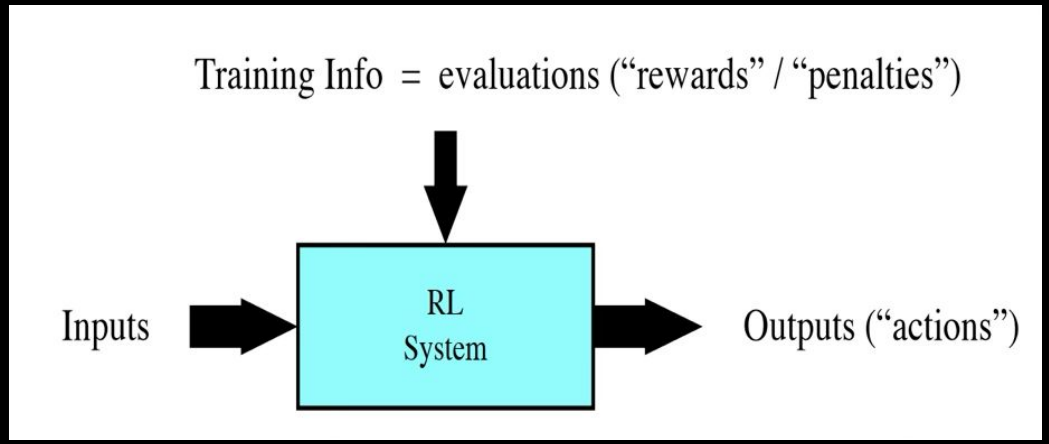


Input : X
Output : Structure de X

LES 3 CATÉGORIES DE ML (ET DL)

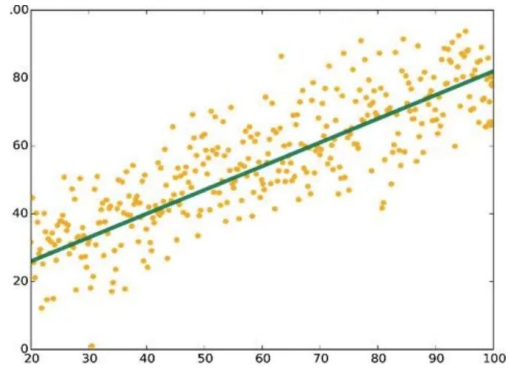
* L'apprentissage par renforcement

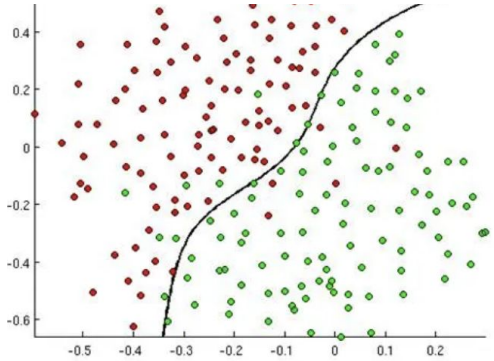




Input : X, récompenses différées
Output : Apprend à interagir de manière optimale dans un environnement en temps réel

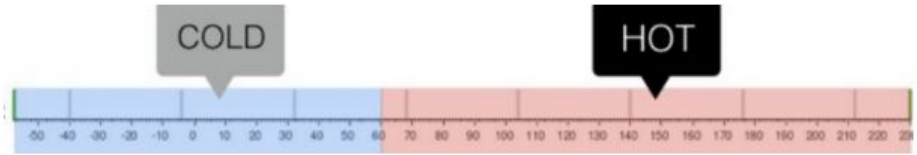
REGRESSION VS CLASSIFICATION



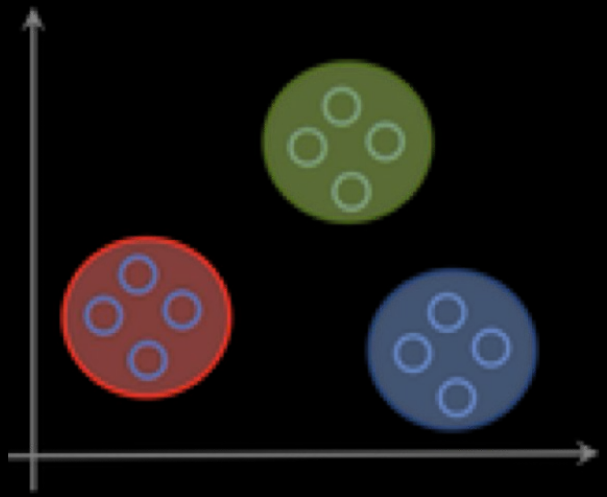


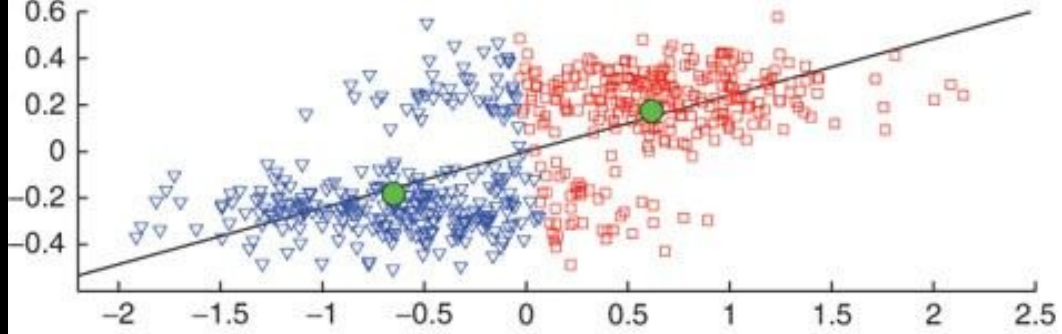
What will be the temperature tomorrow?

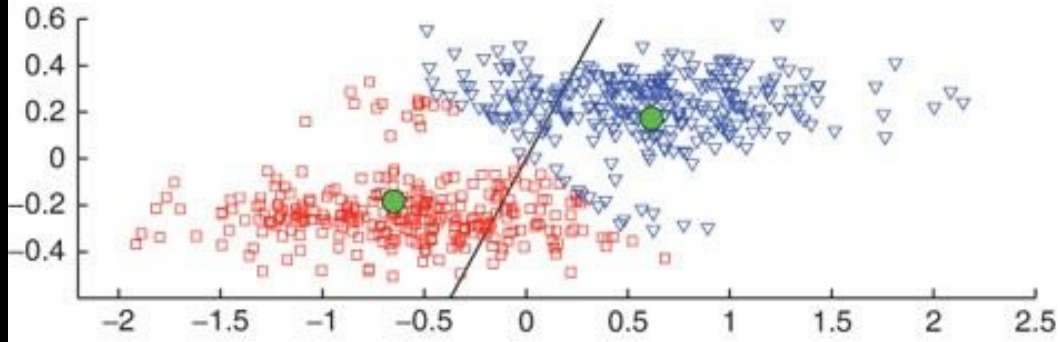
Will it be hot or cold tomorrow?

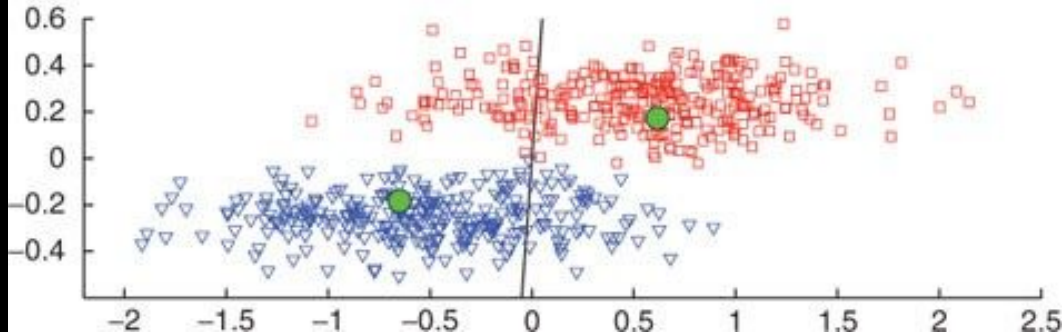


CLASSIFICATION VS RÉDUC. DE DIM.









Problème	Type d'apprentissage	Sous-type
Prédire le prix d'une maison en fonction de sa surface	SL	Régression
Grouper des clients selon leurs comportements d'achat		
Former un robot à marcher dans un environnement inconnu		
Reconnaître des visages sur des photos		
Réduire la dimensionnalité des données pour simplification		
Détecter des anomalies dans un réseau		
Chatgpt		

Problème	Type d'apprentissage	Sous-type
Prédire le prix d'une maison en fonction de sa surface	SL	Régression
Grouper des clients selon leurs comportements d'achat	UL	Clustering
Former un robot à marcher dans un environnement inconnu	RL	-
Reconnaître des visages sur des photos	SL	Classification
Réduire la dimensionnalité des données pour simplification	UL	Réduction de dimensions
Détecter des anomalies dans un réseau	UL	Détection d’anomalies
Chatgpt	SL/Self-SL/RL	-

ML WORKFLOW

1. **Collecte des données :** Identifier et obtenir les données nécessaires pour résoudre le problème.
2. **Préparation des données :** Nettoyer et organiser les données pour assurer leur qualité et leur pertinence.
3. **Division Train-Test-Validation :** Séparer les données pour entraîner, valider et tester le modèle.
4. **Entraînement du modèle :** Utiliser les données d'entraînement pour créer un modèle prédictif.
5. **Évaluation du modèle :** Vérifier les performances du modèle sur des données qu'il n'a jamais vues.
6. **Déploiement et surveillance :** Mettre le modèle en production et suivre ses performances pour maintenir sa fiabilité.

ML WORKFLOW (TRAIN-TEST-VALID.)

Train Set (Ensemble d'entraînement) :

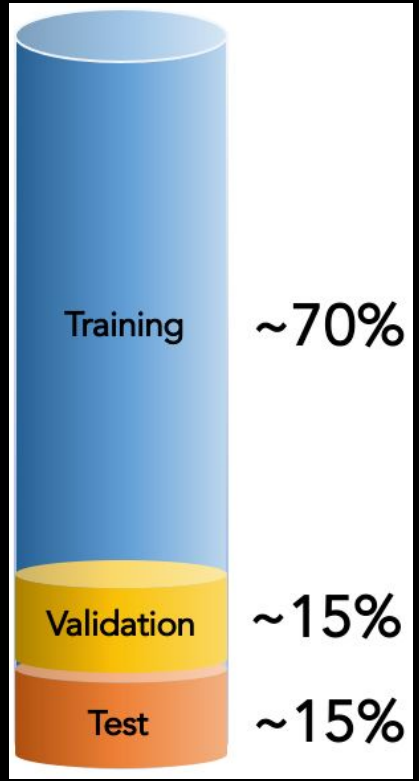
- Utilisé pour former le modèle en ajustant ses paramètres.

Validation Set (Ensemble de validation) :

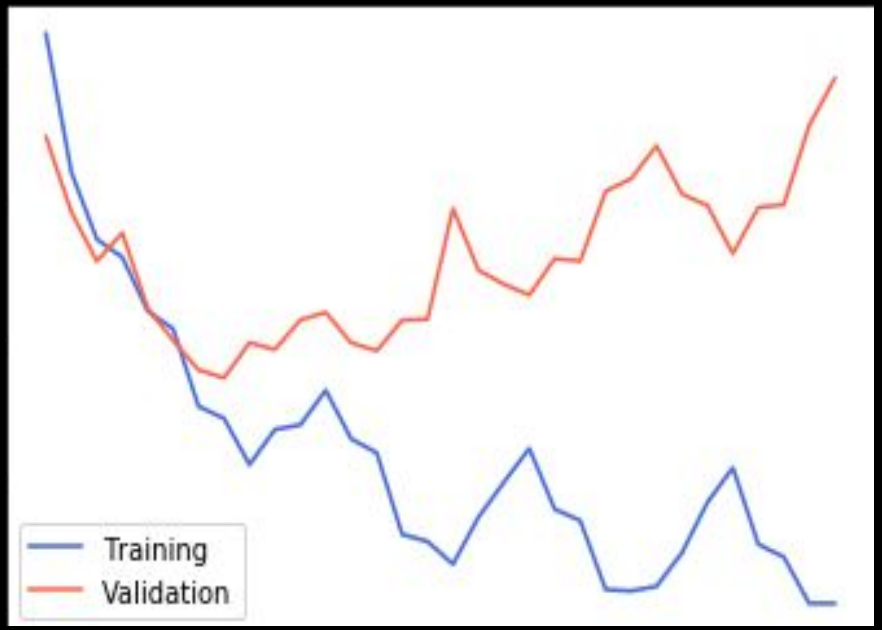
- Sert à ajuster les hyperparamètres (ex. : taux d’apprentissage, nombre de couches).
- Permet de prévenir le surapprentissage (overfitting).

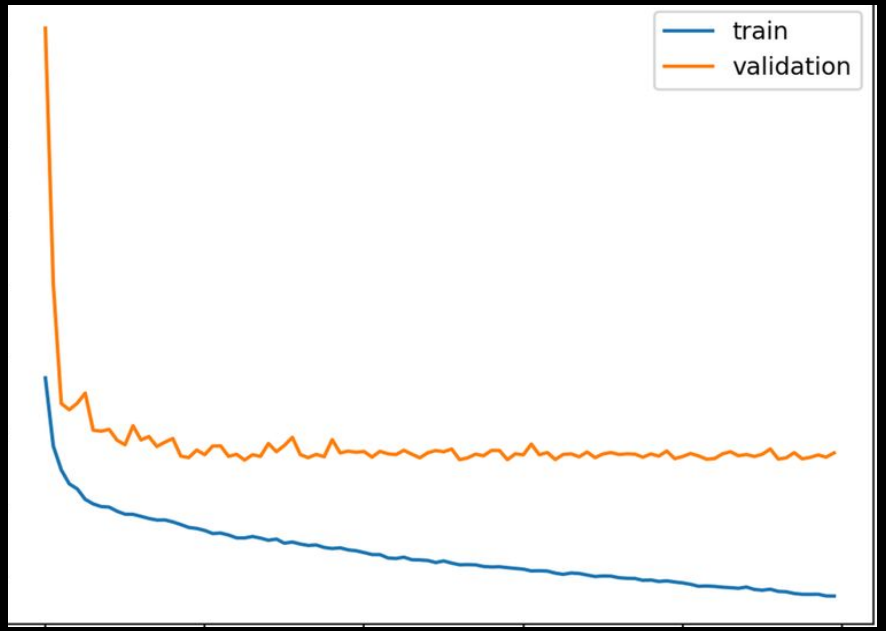
Test Set (Ensemble de test) :

- Évalue les performances finales du modèle sur des données totalement inédites.
- Fournit une mesure de généralisation.

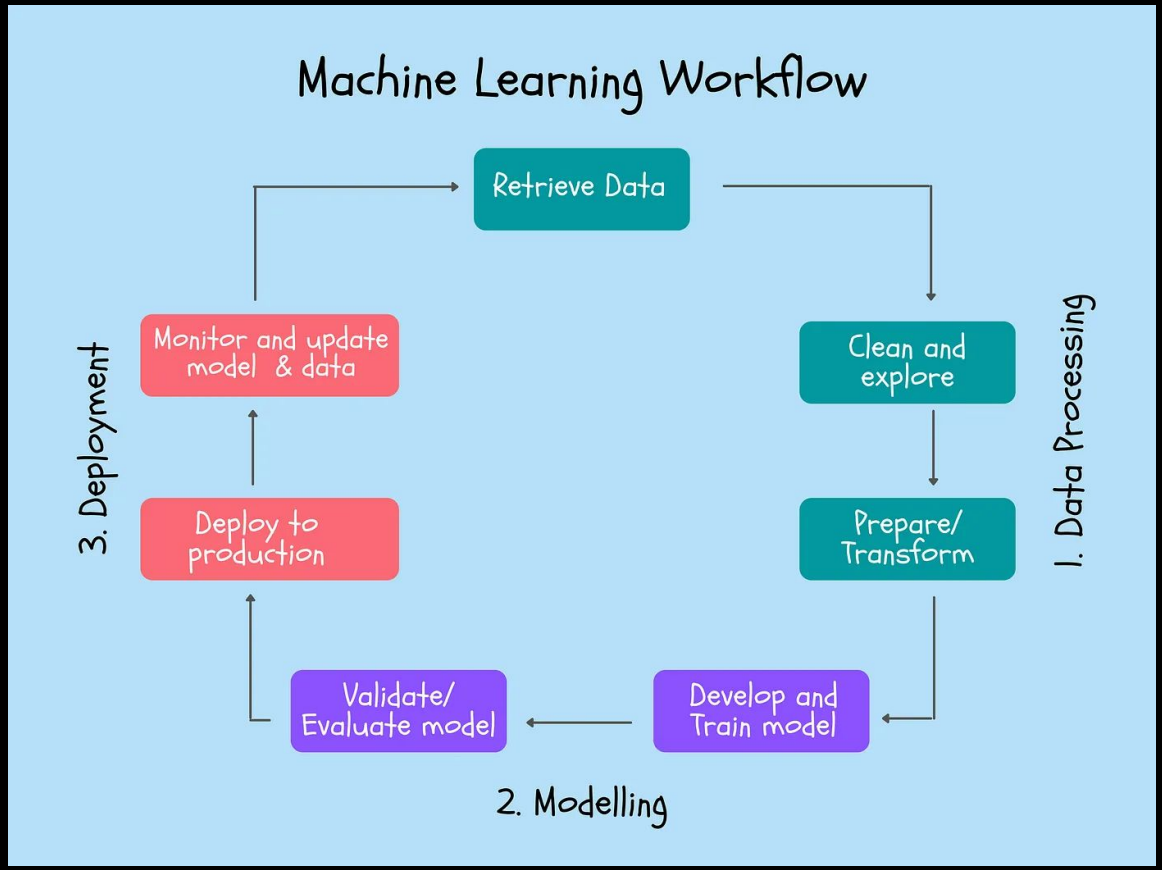


ML WORKFLOW (TRAIN-TEST-VALID.)

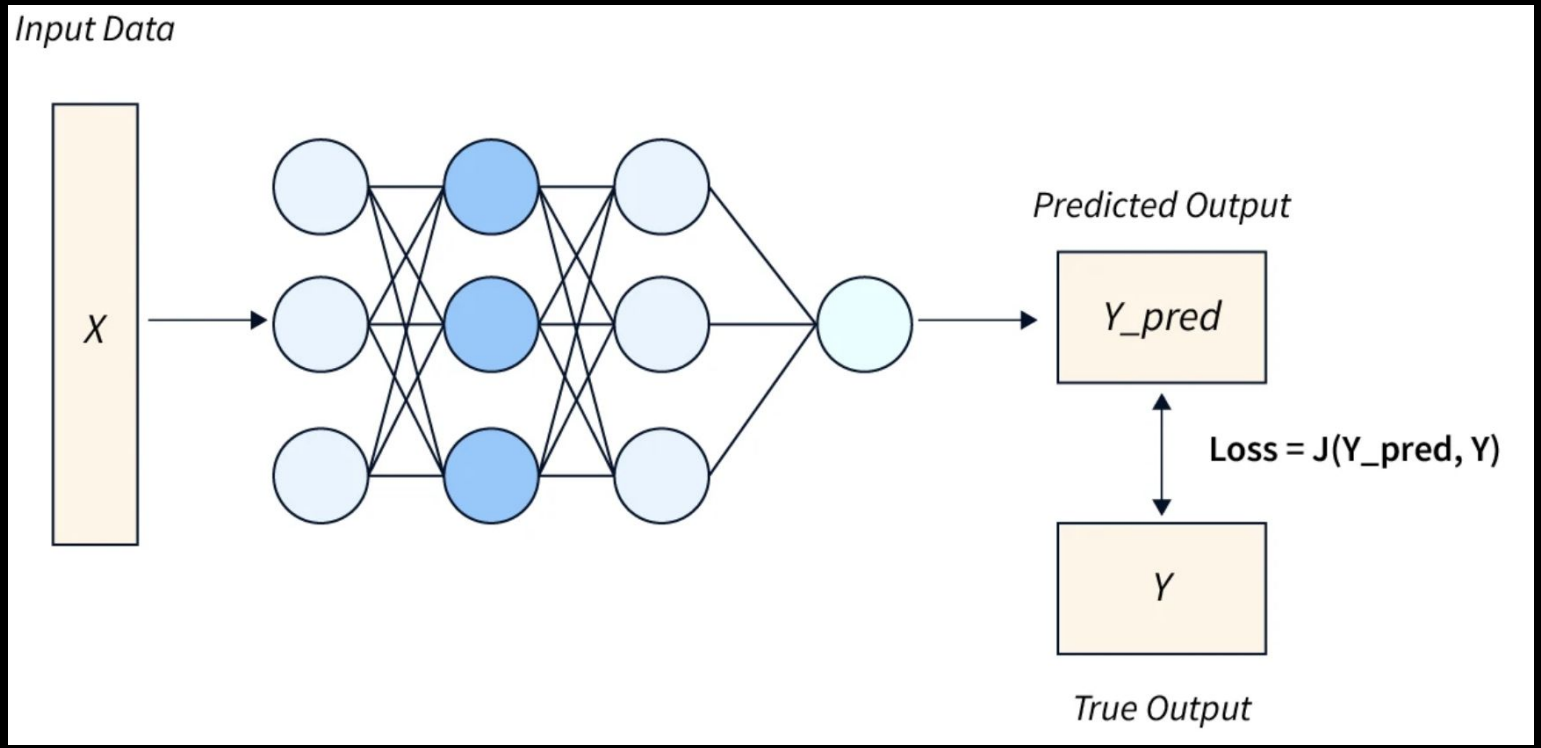




ML WORKFLOW - MLOPS



ML WORKFLOW - MODEL TRAINING



MÉTRIQUES POUR LA RÉGRESSION

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Métrique	Explication	Utilisation
Mean Squared Error (MSE)	Amplifie les grandes erreurs en les élevant au carré.	- Utile lorsque les grandes erreurs doivent être fortement pénalisées. - Couramment utilisé comme fonction de coût dans les algorithmes (ex. : régression linéaire).
Mean Absolute Error (MAE)	Traite toutes les erreurs de manière égale.	- Idéal lorsque toutes les erreurs doivent avoir une importance équivalente. - Utilisé dans des cas où les données contiennent des valeurs aberrantes non critiques.
Root Mean Squared Error (RMSE)	- Reste sensible aux grandes erreurs - Même unité que les données.	- Utilisé pour des comparaisons faciles et interprétables. - Populaire dans les domaines où les grandes erreurs ont un impact significatif (ex. : prévisions météo, finances).

MÉTRIQUES POUR LA RÉGRESSION

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **Valeurs réelles (y)** : [100, 200, 300]
- **Prédictions ($\hat{y}$)** : [110, 210, 500]
- Les erreurs absolues ($y - \hat{y}$) : [10, 10, 200]

Calcul du MAE :

$$MAE = \frac{1}{3}\left(|10| + |10| + |200|\right) = \frac{220}{3} \approx 73.33$$

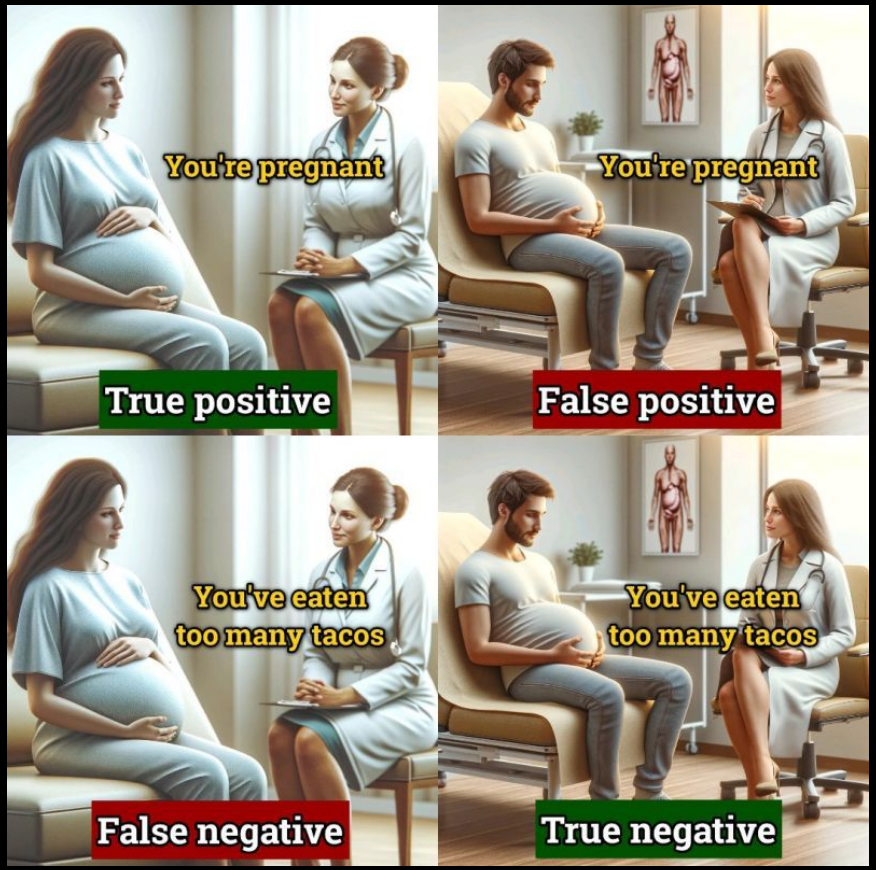
Calcul du MSE :

$$MSE = \frac{1}{3}\left(10^2 + 10^2 + 200^2\right) = \frac{1}{3}(100 + 100 + 40000) = 13400$$

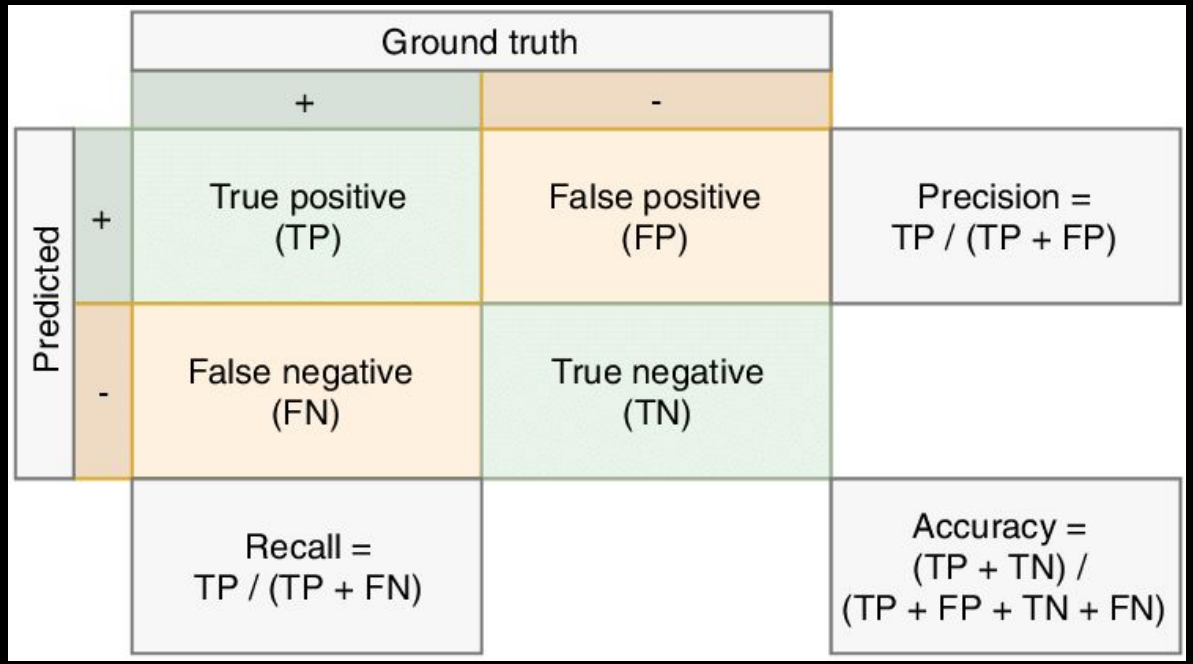
Calcul du RMSE :

$$RMSE = \sqrt{MSE} = \sqrt{13400} \approx 115.82$$

MÉTRIQUES POUR LA CLASSIFICATION



MÉTRIQUES POUR LA CLASSIFICATION



Accuracy (Précision globale)

- Proportion de prédictions correctes parmi toutes les observations.

Precision (Précision)

- Parmi les prédictions positives, combien sont réellement correctes.

Recall (Rappel ou Sensibilité)

- Parmi les cas réellement positifs, combien ont été correctement prédits

MÉTRIQUES POUR LA CLASSIFICATION

1. Accuracy (Précision globale)

Scénario : Prédiction de cancer (Classes : **Cancer** = Positif, **Pas de cancer** = Négatif).

- Données : 100 patients, dont 95 n'ont pas de cancer, et 5 ont un cancer.
- Modèle : Prédit toujours **Pas de cancer**.

Classe réelle	Prédiction du modèle	Correct ?
Cancer (Positif)	Pas de cancer (Négatif)	Faux
Pas de cancer	Pas de cancer	Vrai

Résultat :

- Accuracy = $\frac{\text{Prédictions correctes}}{\text{Total}} = \frac{95}{100} = 95\%$.
- **Limite :** L'accuracy est trompeuse, car le modèle ignore totalement les 5 cas de cancer (Faux négatifs).

MÉTRIQUES POUR LA CLASSIFICATION

2. Precision (Précision)

Scénario : Détection de spam dans des emails (Classes : **Spam** = Positif, **Non spam** = Négatif).

- Données : 100 emails, 20 sont des spams.
- Modèle : Prédit **30 spams**, dont 15 sont corrects.

Classe réelle	Prédictions positives	Correct ?
Spam (Positif)	15 (Vrai positif)	Oui
Non spam	15 (Faux positif)	Non

Résultat :

- Precision = $\frac{\text{Vrai positifs}}{\text{Total prédictions positives}} = \frac{15}{30} = 50\%$.
- **Importance :** Mesure la fiabilité des prédictions positives. Si vous recevez un email marqué "spam", la précision vous indique la probabilité qu'il soit réellement un spam.

MÉTRIQUES POUR LA CLASSIFICATION

3. Recall (Rappel ou Sensibilité)

Scénario : Détection de fraude bancaire (Classes : **Fraude** = Positif, **Non fraude** = Négatif).

- Données : 1,000 transactions, 10 sont frauduleuses.
- Modèle : Identifie 6 fraudes sur 10.

Classe réelle	Vrai positif (Fraude détectée)	Faux négatif (Fraude non détectée)
Fraude	6	4

Résultat :

- Recall = $\frac{\text{Vrai positifs}}{\text{Total positifs}} = \frac{6}{10} = 60\%$.
- **Importance :** Dans la détection de fraude, le rappel est crucial pour minimiser les **faux négatifs** (fraudes non détectées).

MÉTRIQUES POUR LA CLASSIFICATION

4. F1-Score

Scénario : Même situation que la détection de spam, avec :

- Precision = 50%, Recall = 75%.

Résultat :

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \times \frac{0.5 \times 0.75}{0.5 + 0.75} = 0.6\,(60\%)$$

- **Importance :** L'équilibre entre la précision (faux positifs) et le rappel (faux négatifs). Un bon F1-score est utile pour des jeux de données **déséquilibrés**.

ML, PRATIQUEMENT?

```
# Importation des données sous forme de DataFrame
data = pd.DataFrame(housing, columns=housing.feature_names)

# Mise à l'échelle des données
X_scaled = scaler.fit_transform(data[selected_features])
y = data['MedHouseVal']

# 2. Division des données
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42)

# 3. Création et entraînement du modèle
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

# 4. Prédiction sur les données de test
y_pred_dt = lr_model.predict(X_test)

# 5. Évaluation du modèle
rmse_linear = np.sqrt(mean_squared_error(y_test, y_pred_dt))
```