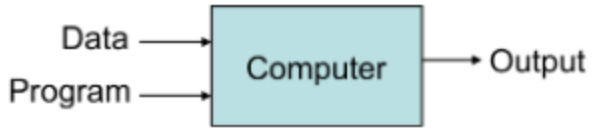
Introduction au Machine Learning

Aurélien Marie, Data Scientist
aurelien.marie@bnpparibas-pf.com
Université de Bordeaux
MASTER IREF 2024-2025

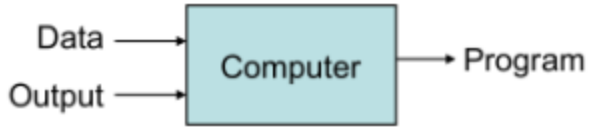
DÉFINITION DU MACHINE LEARNING

"The goal of machine learning is to build computer systems that can adapt and learn from their experience." - Tom Dietterich

- Traditional Programming



- Machine Learning

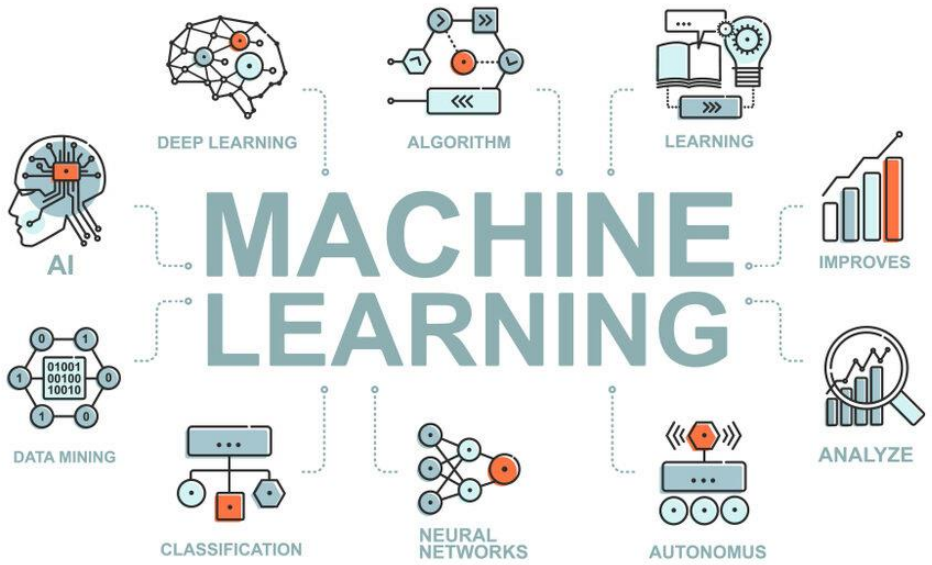


EXEMPLE D'APPLICATIONS

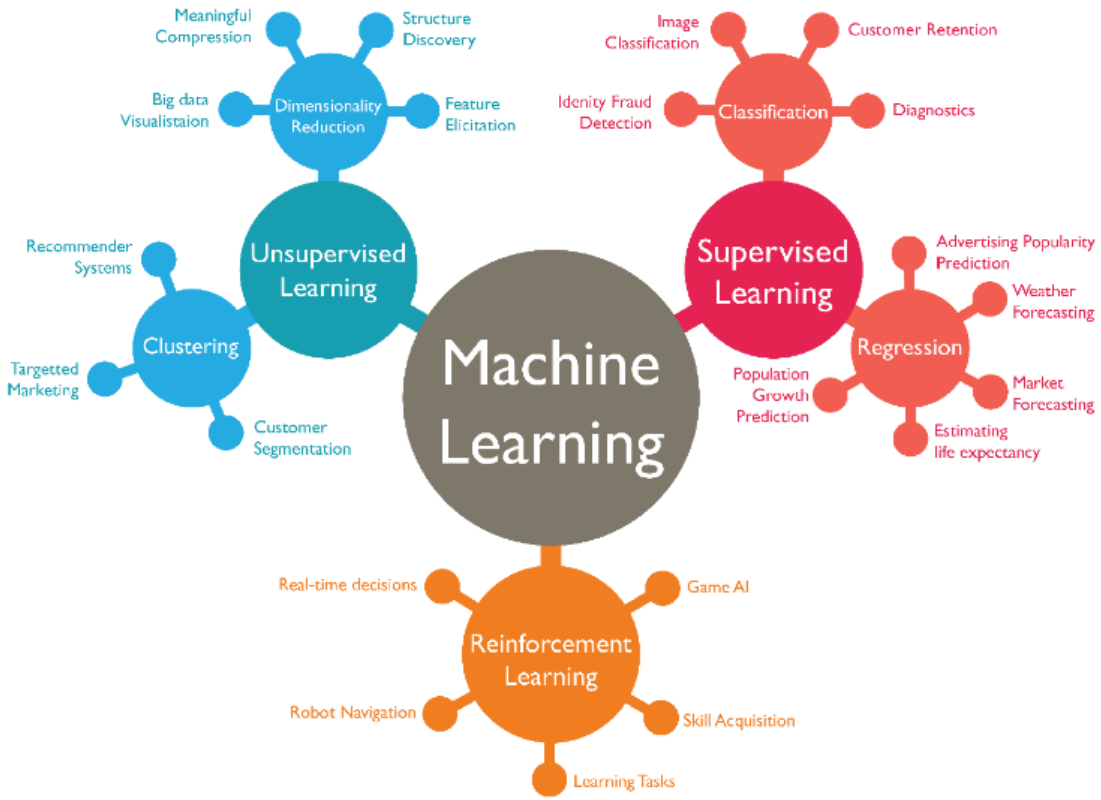
- On dispose de données bancaires. Un client risque-t-il de faire défaut pendant la durée de son emprunt ?
- Peut-on prédire le provisionnement de nos crédits ?
- Peut-on détecter de la fraude au sein de nos portefeuilles ?
- Proposer des outils de détection d'attaque type Spam
- Segmenter nos clients pour recommander d'autres produits ? (assurance, livret …)
- Améliorer nos sites internet via de l'A/B Testing

CONCEPTION DES ALGORITHMES DE MACHINE LEARNING

1. Déterminer la tâche d'apprentissage
2. Collecter les données
3. Extraire les caractéristiques (features)
4. Choisir la classe de modèles adaptée
5. Entraîner le modèle
6. Evaluer le modèle

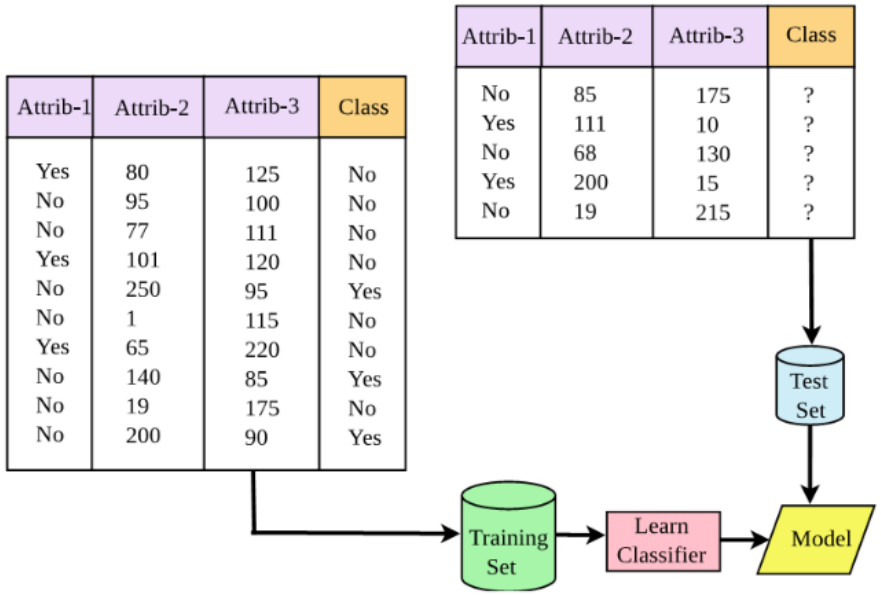


TYPE DE MODÈLE DE MACHINE LEARNING



APPRENTISSAGE SUPERVISÉ

• Les données d'apprentissage (observations) sont accompagnées par les labels indiquant leurs classes

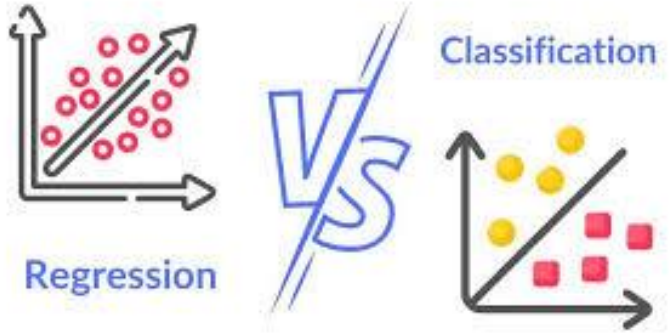


APPRENTISSAGE SUPERVISÉ

Classification

Binaire : affecter chaque observation à une des deux classes
Exemple : Octroi de crédit (oui ou non)

Multiclasse : affecter chaque observation à un classe N (N > 2)
Exemple : Carte Bancaire de nos clients : Silver, Gold, Platinum



Régression

Affecter une valeur réelle à chaque observation

Exemple : Estimer la perte estimée d'un client faisant défaut (va-t-il rembourser en partie ? la totalité ?) → LGD (Loss Given Default)

APPRENTISSAGE NON SUPERVISÉ

- Le label de classe n’est pas connu
- L’objectif de segmenter des classes ou groupe dans les données

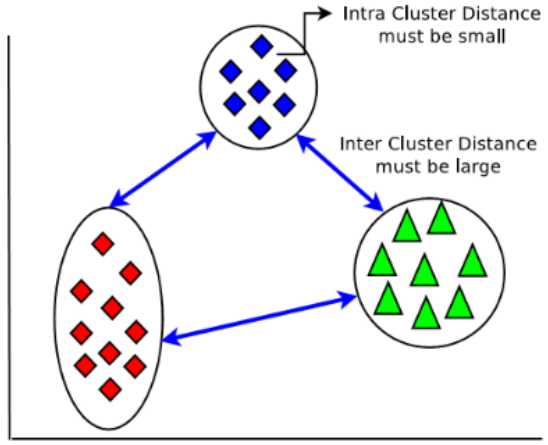


Figure: Unsupervised Learning

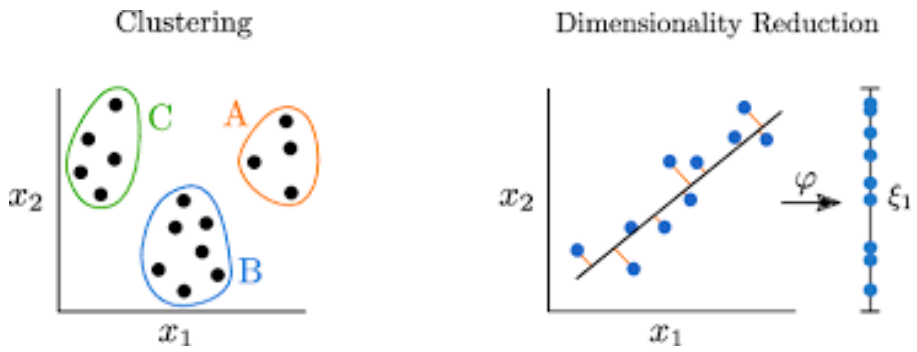
APPRENTISSAGE NON SUPERVISÉ

Réduction de dimension

- Réduire la dimension des données en minimisant la perte d'information
- Analyse en composante principale (ACP), T-SNE
- Exemple : Modèle de comportement de crédit (plus de 1000 variables

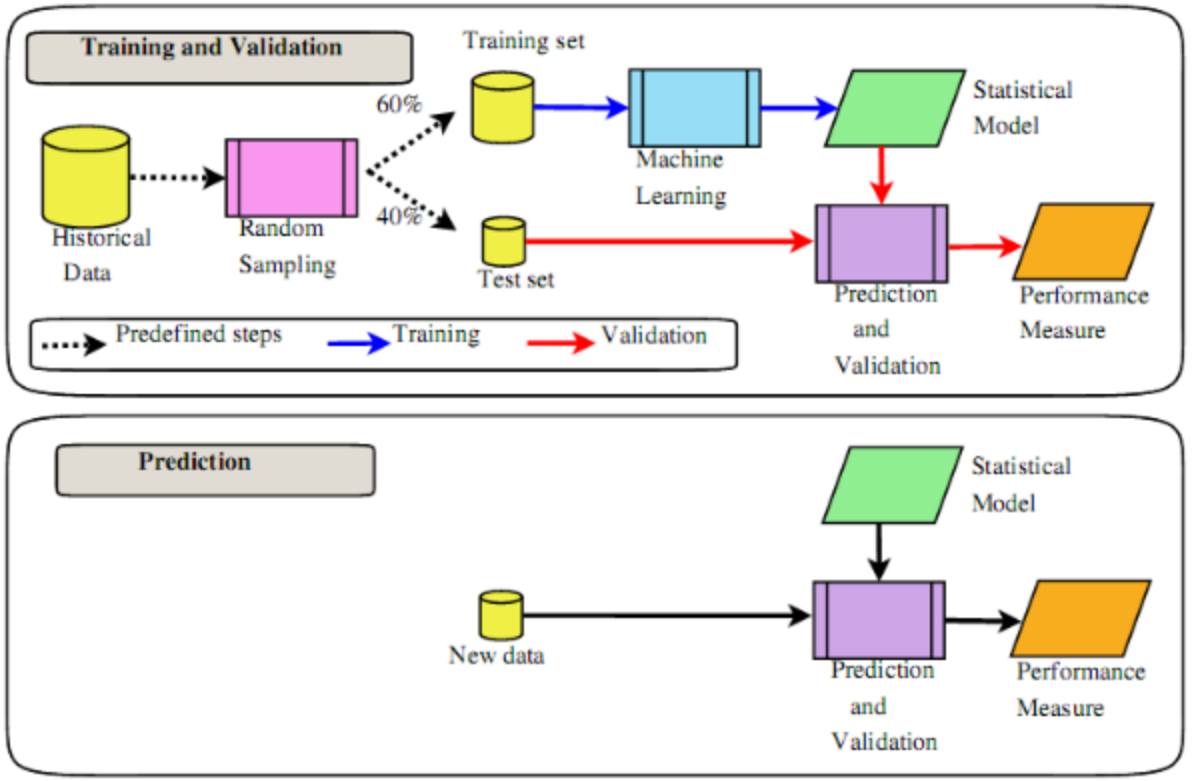
Clustering

- Regrouper les données en groupes homogènes par rapport à une mesure de similarité
- K-means, Hierarchical clustering, Dbscan
- Exemple : suggérer des typologies de clients par cluster



PROCESSUS D'ÉVALUATION D'UN MODÈLE

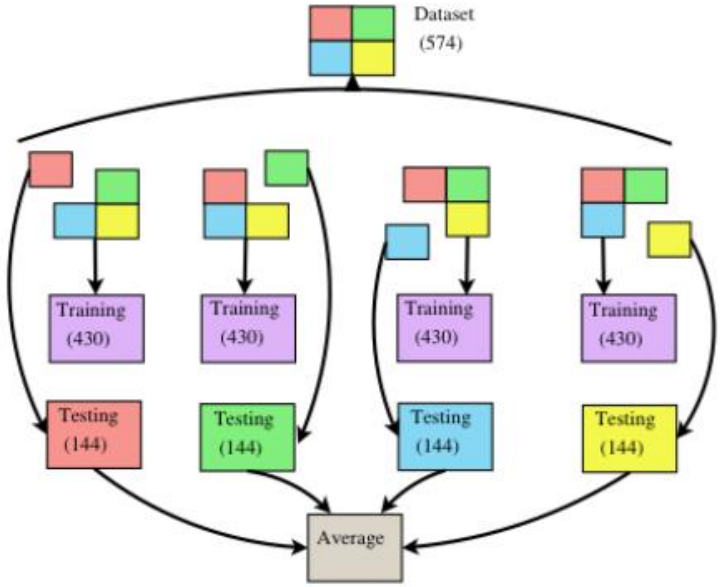
Si la volumétrie est suffisante : Train (60-80%) et Test (20-40%)



PROCESSUS D'ÉVALUATION D'UN MODÈLE

Cross-Validation :

- Lorsque la volumétrie n'est pas suffisante
- Utile pour optimiser les paramètres de son modèle



MÉTRIQUES D'ÉVALUATION

Matrice de confusion

	Actual class (Observation)	
Predicted class (Expectation)	TP	FP
	FN	TN

- TP : True Positif
- FP : False Positif
- FN : False Negative
- TN : True Negative

MÉTRIQUES D'ÉVALUATION

Accuracy

Il s'agit de la fraction des exemples bien classés par rapport à toutes les prédictions

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

Problème : s'il existe un déséquilibre entre les deux classes

Exemple : si 1% des clients sont des mauvais payeurs, vous pouvez obtenir 99% d'accuracy en créant un modèle ne prédisant que 1 (bon payeur)

Solution : Calcul d'un ratio pour rééquilibrer les coûts N / P avec N = TN+FP et P=TP+FN

MÉTRIQUES D'ÉVALUATION

- Précision

C'est la fraction des vrais positifs parmi les exemples prédits comme positifs

$$Pre = \frac{TP}{TP + FP}$$

- Rappel

Il s'agit du rapport entre le nombre d'exemples de la classe c correctement prédits et le nombres d'exemples appartenant à la classe c

$$Rec = \frac{TP}{TP + FN}$$

MÉTRIQUES D'ÉVALUATION

F-measure

$$F_\beta = \frac{(1+\beta^2)(Pre * Rec)}{\beta^2 Pre + Rec}$$

- La précision et le rappel sont complémentaires, augmenter la précision diminue le rappel et inversement.
- F-measure combine les deux mesures en les équilibrant
- β est un paramètre du niveau de compromis, exemple β=1

$$F_\beta = \frac{2(Pre * Rec)}{Pre + Rec}$$

MÉTRIQUES D'ÉVALUATION

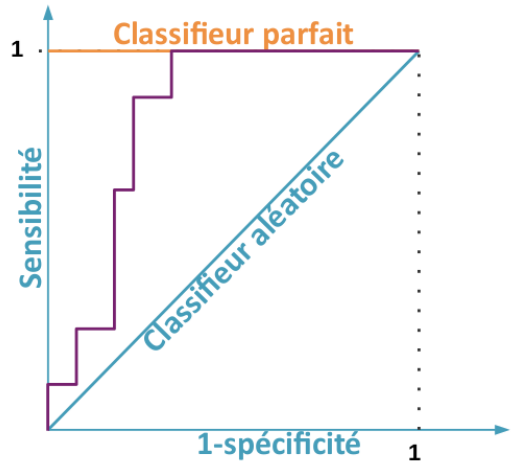
Courbe ROC (receiver operating characteristic)

Graphique représentant les performances d'un modèle de classification pour tous les seuils de classification

Note : Gini = 2 * AUC -1

Seuil : un modèle de classification établi des probabilités pour l'ensemble des observations, exemple : le client a 0,67% de probabilité de faire défaut. Il faut donc déterminer le seuil pour lequel le client sera classé 1 (seuil = 0,5 par défaut)

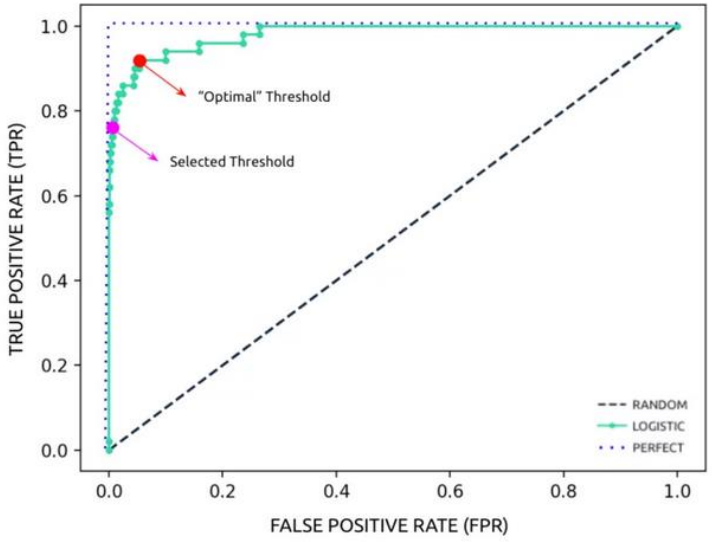
Sensibilité = TP / TP + FN
Spécificité = TN / TN + FP



Courbe ROC

MÉTRIQUES D'ÉVALUATION

Courbe ROC

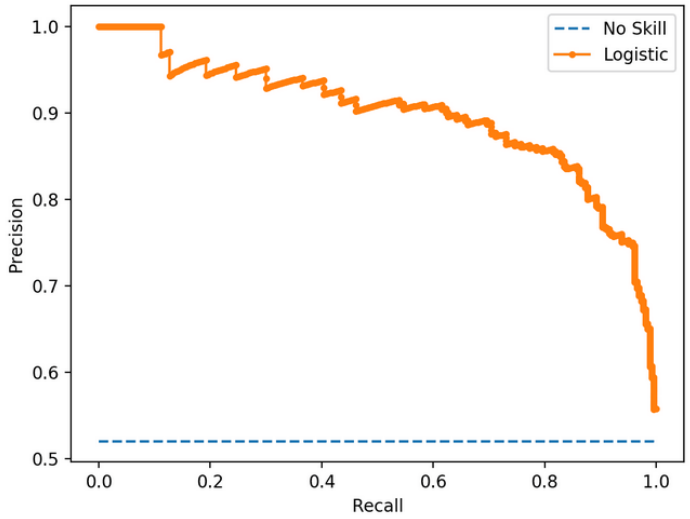


- Invariant au choix du seuil : mesure la qualité de prédiction du modèle
- Invariant à l'échelle des prédictions
- Equilibré : AUC fourni une métrique capturant le trade-off entre le TPR et le FPR sur l'ensemble des seuils
- Moins informatif en cas de déséquilibre des données (sur-optimiste dans la performance)

MÉTRIQUES D'ÉVALUATION

Précision-Recall Curve

Graphique représentant la précision et le rappel pour l'ensemble des seuils



- Informatif pour des jeux de données déséquilibrés (FPR)
- Concentre sur la classe positive (utile quand les classe positive est plus importante)
- Dépendance forte au seuil
- Généralement moins comprise : AUC est majoritairement utilisée

TECHNIQUE D'ÉCHANTILLONNAGE POUR LE DÉSÉQUILIBRE DES CLASSES

Oversampling : dupliquer la catégorie minoritaire

Undersampling : supprimer aléatoirement des observations de la classe majoritaire

SMOTE (Synthetic Minority Oversampling Technique)

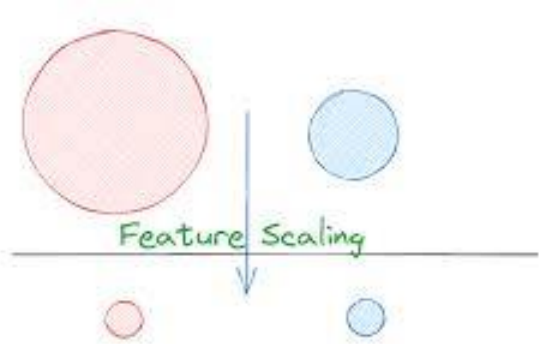
1. Sélectionner aléatoirement une observation minoritaire "initiale".
2. Identifier ses k plus proches voisins parmi les observations minoritaires (où k est un paramètre défini par l'utilisateur).
3. Choisir aléatoirement l'un des k plus proches voisins.
4. Générer aléatoirement un coefficient $0 < \alpha < 1$.
5. Créer un nouvel individu entre l'observation initiale et le plus proche voisin choisi, selon la valeur du coefficient . Par exemple, si $\alpha = 0.5$, le nouvel individu sera positionné à mi-chemin entre l'observation initiale et le plus proche voisin choisi.

- Normaliser les variables numériques avant d'appliquer SMOTE
- Optimiser les paramètres vie la cross-validation
- Retraiter les variables discrètes (création de valeurs inexistantes)
- Ne pas encoder les catégorielles, utiliser **SMOTE-Nominal Continuous**
- Ne pas utiliser SMOTE sur les données de validation et de test

FEATURE SCALING

Certains algorithmes sont sensibles aux échelles des données :

- Régression logistique et linéaire
- KNN : distance euclidienne
- Réduction de dimension : ACP, T-SNE / Clustering : K-means etc …
- Méthode CS : mise en classe pour éviter ces problématiques



FEATURE SCALING

Standard Scaler :

$$z = \frac{x - \mu}{\sigma}$$

- Facile à implémenter
- Applique une régularisation
- Assume une distribution normale des données
- Problématique de mise à l'échelle avec des larges volumes de données
- Sensible aux valeurs extrêmes / outliers

FEATURE SCALING

MinMax Scaler :

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- Facile à implémenter
- Alternative à la standardisation si on doit éviter une moyenne nulle et une variance égale à 1
- Sensible aux valeurs extrêmes / outliers
- Compression entre [0, 1] peut signifier une perte significative d'information

FEATURE SCALING

Robust Scaler :

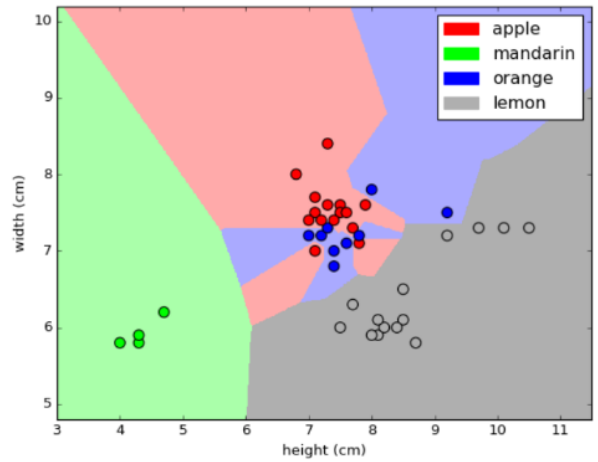
$$X_{\text{scale}} = \frac{x_i - x_{\text{med}}}{x_{75} - x_{25}}$$

- Gère les valeurs extrêmes et la « skewness » (asymétrie) des données
- Ne prend pas en compte la moyenne

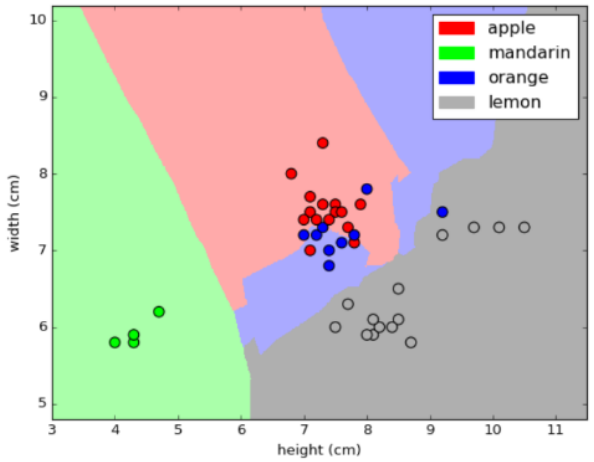
K-NN NEAREST-NEIGHBOR CLASSIFIER

Exemple :

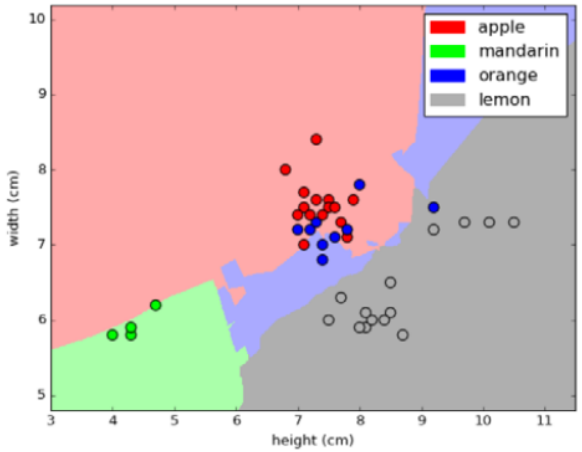
- Distance euclidienne
- K = 1, 5, 10 plus proches voisins
- Vote majoritaire



K=1



K=5



K=10

K-NN NEAREST-NEIGHBOR CLASSIFIER

Algorithme :

Soit k le nombre de voisins les plus proches et D l'ensemble d'entraînement.

1. pour chaque exemple $z = (x', ?)$ de l'ensemble de test :
   - 1.1 Calculer $d(x, x')$, la distance de z et chaque exemple (x, y) de D;
   - 1.2 Choisir $D_z \subset D$, l'ensemble des k exemples les plus proches de z;
   - 1.3 $y' = \arg\max_v \sum_{(x_i, y_i) \in D_z} I(v = y_i)$
2. Fin pour

Choix de la valeur k :

- Si k est trop petit, la classification sera trop sensible au bruit
- Si k est trop grand, le voisinage peut contenir des éléments d'autres classes

Normalisation impérative pour éviter les problèmes d'échelles entre les variables

Exemple :

1. Age : [18, 80]
2. Salaire : [0, 10000]
3. Montant du crédit : [1000, 10000]

En grande dimension, la distance euclidienne des observations sont proches, il peut être intéressant d'utiliser une technique de réduction de dimension avant le traitement (ACP)

NAIVE BAYES CLASSIFIER

• **Naive Bayes** est un algorithme se basant sur le théorème de Bayes, il est fondé sur les probabilités conditionnelles.

• **Probabilités conditionnelles** : Quelle est la probabilité qu'un événement se produise sachant qu'un autre événement s'est déjà produit**.**

• Supposons qu'on ait un ensemble de client . Soit A et B deux événements :

A : le client est marié
B : le client est locataire

P(marié et locataire) = P(marié) * P(locataire | marié)

	Marié	Non Marié	Total
Locataire	10	7	17
Non locataire	4	9	13
Total	14	16	30

NAIVE BAYES CLASSIFIER

- P(locataire | marié) = P(marié| locataire)* P(locataire) / P(marié)
- P(marié) = 14 / 30 = 0,46
- P(locataire et marié) = 10 / 30 = 0,33
- P(locataire | marié) = 0,33 / 0,46 = 0,71

	Marié	Non Marié	Total
Locataire	10	7	17
Non locataire	4	9	13
Total	14	16	30

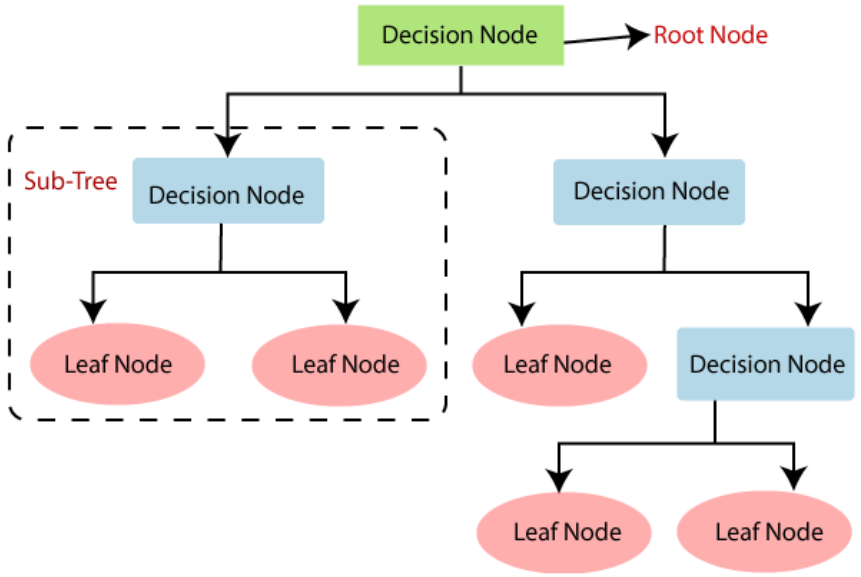
Dans le cas > 2 variables, on émet l'hypothèse d'indépendance entre les variables (d'où le termes naïf)

$$P(BP \mid a,b,c) = P(BP \mid a) * P(BP \mid b) * P(BP \mid c) \ / P(a) * P(b) * P(c)$$

- Avantages : Calcul très rapide pour la classification (probabilités) et faisable sur des petits jeux de données
- Inconvénients : Indépendance des variables (hypothèses violées dans la majorité des cas)

NAIVE BAYES CLASSIFIER

- Un arbre décision est une représentation graphique de l'ensemble des solutions d'un problème décisionnel basé sur des conditions
- On appelle cela arbre de décision car c'est similaire à des arbres, en commençant par un nœud racine qui s'étend vers d'autres branches et se construit comme la structure d'un arbre.
- **CART algorithm** : Classification and Regression Tree Algorithm



Les arbres de décisions reproduisent la capacité humaine à prendre des décisions et ils sont faciles à comprendre

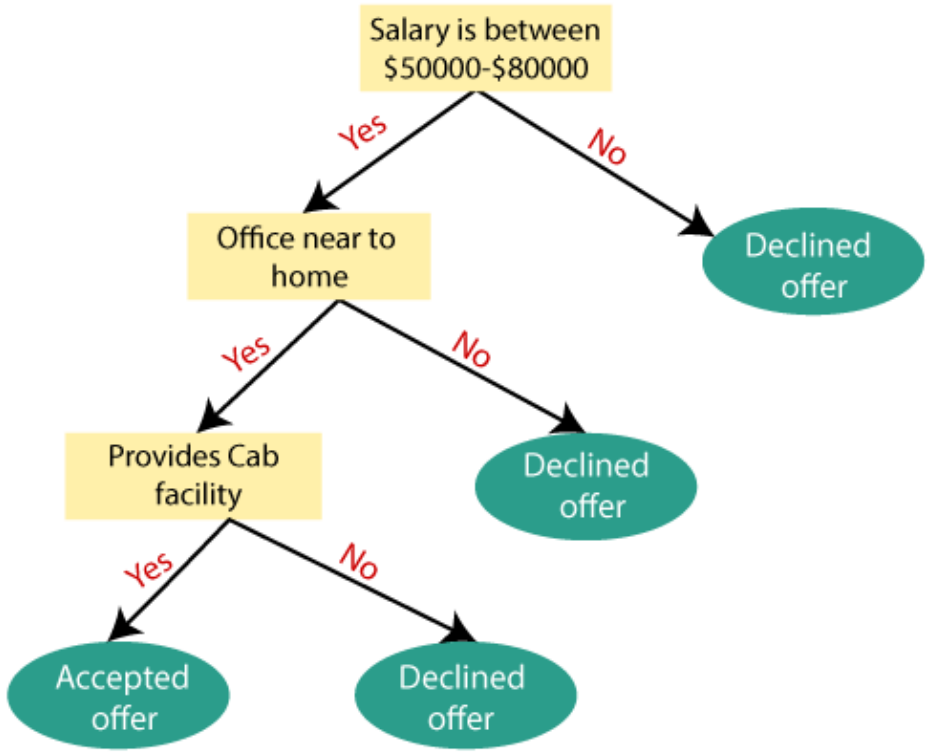
DECISION TREE CLASSIFIER

Terminologie :

- **Root node :** point de départ des arbres de décisions. Ils représentent le jeu de données complets, qui sera divisés en deux ou plus sous-ensembles de données.
- **Leaf node :** nœuds terminaux de l'arbre, ils ne peuvent plus être séparés.
- **Splitting :** processus de division des nœuds en sous-nœuds selon certaines conditions
- **Branch/Sub Tree :** un arbre formé par une règle de splitting
- **Pruning :** processus de suppression de branche dans l'arbre de décision

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

DECISION TREE CLASSIFIER



DECISION TREE CLASSIFIER

Information Gain : Comment quantifier la qualité d'un split ?

- **Information entropy :** estimer la variance du jeu de données
- C le nombre de classes, pi la probabilité de tirer aléatoirement un élément de la classe i

$$E = -\sum_{i}^{C} p_i \log_2 p_i$$

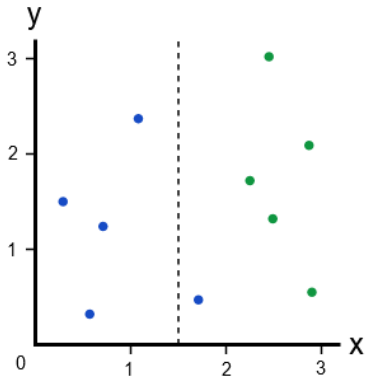
Exemple :

- A dataset of only blues ●●●● would have very **low** (in fact, zero) entropy.
- A dataset of mixed blues, greens, and reds ●●●●●● would have relatively **high** entropy.

DECISION TREE CLASSIFIER

Information Gain : Comment quantifier la qualité d'un split ?

$$E_{before} = -(0.5 \log_2 0.5 + 0.5 \log_2 0.5)$$
$$= \boxed{1}$$



An Imperfect Split

$$E_{left} = \boxed{0}$$

$$E_{right} = -(\frac{1}{6} \log_2(\frac{1}{6}) + \frac{5}{6} \log_2(\frac{5}{6}))$$
$$= \boxed{0.65}$$

$$E_{split} = 0.4 * 0 + 0.6 * 0.65$$
$$= \boxed{0.39}$$

$$\text{Gain} = 1 - 0.39 = \boxed{0.61}$$

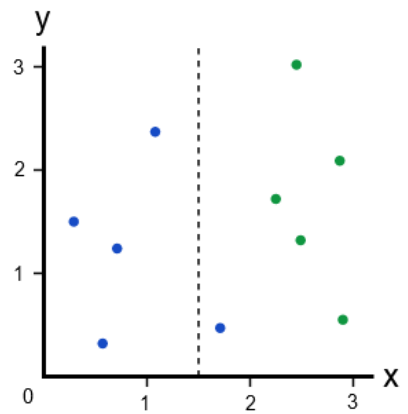
DECISION TREE CLASSIFIER

Gini Impurity: Comment quantifier la qualité d'un split ?

- Sélectionner aléatoirement une observation
- Classer aléatoirement cette observation
- Quel est la probabilité que la classification est incorrecte ?

$$G = \sum_{i=1}^{C} p(i) * (1 - p(i))$$

- Exemple :



An Imperfect Split

$$G_{left} = \boxed{0}.$$

$$\begin{aligned} G_{right} &= \frac{1}{6} * (1 - \frac{1}{6}) + \frac{5}{6} * (1 - \frac{5}{6}) \\ &= \frac{5}{18} \\ &= \boxed{0.278} \end{aligned}$$

$$(0.4 * 0) + (0.6 * 0.278) = 0.167$$

$$0.5 - 0.167 = \boxed{0.333}$$

DECISION TREE CLASSIFIER

Arbre de décision : décision non linéaire

Avantage :

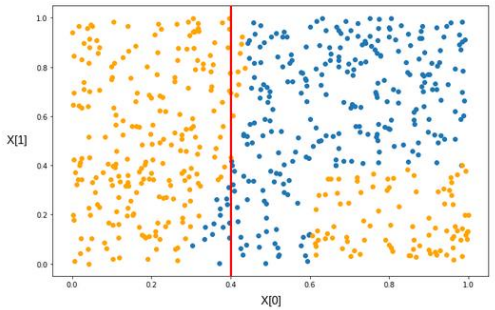
- Facile à interpréter
- Gère des données numériques et catégorielles
- Fonctionne sur de larges volumes de données
- Peu sensible aux outliers
- Non paramétriques

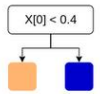
Inconvénients :

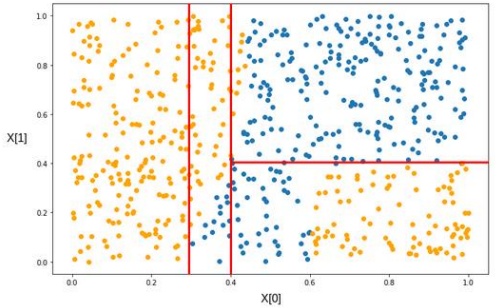
- Attention au sur-apprentissage : risque élevé pour les modèles non paramétriques
- Variance élevé / biais faible : les résultats peuvent varier fortement d'un échantillon à un autre
- Ne garantit pas d'être optimal

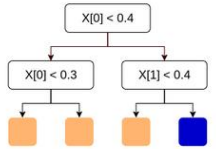
Choix des hyperparamètres :

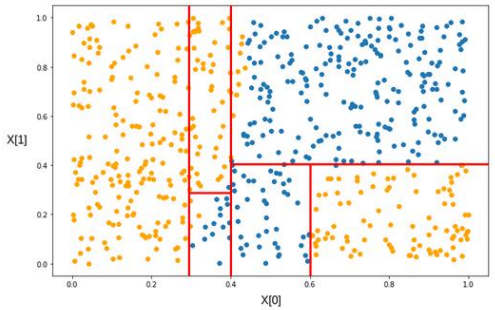
- Profondeur maximal (max_depth)
- Gain minimum de performances (min_impurity_decrease)
- Nombre d'observation minimale dans les nœuds (min_samples_split ou min_samples_leaf)

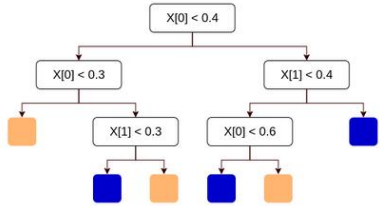












RANDOM FOREST CLASSIFIER

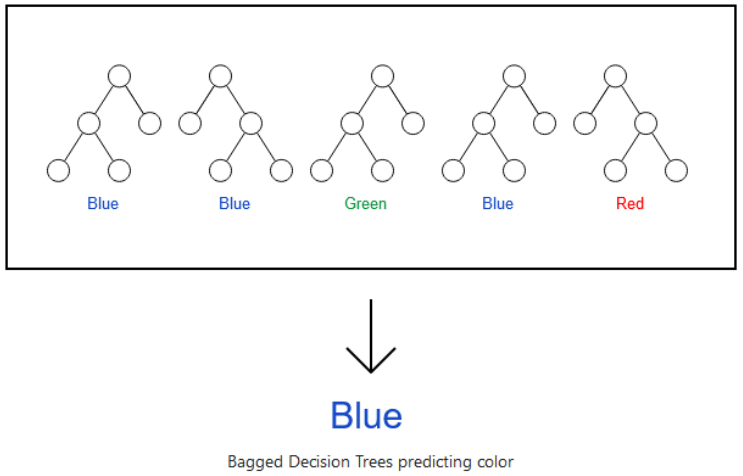
Le Random Forest (forêt aléatoire) est un ensemble d'arbre décision

Bagging (bootstrap bagging):

1. Sélectionner un échantillon du jeu de données de taille n avec remise
2. Entrainer l'arbre de décision avec ces n observations
3. Répéter t fois le processus

Prédiction :

- Classification : vote majoritaire
- Régression : moyenne



RANDOM FOREST CLASSIFIER

Le Bagging Tree possède seulement un unique paramètre : le nombre d'arbre

Le Random Forest inclus un second paramètre : le nombre de variables à considérer pour déterminer le meilleur split

Cet ajout de tirage aléatoire des variables à sélectionner permet de :

- Créer des arbres uniques
- Réduire les corrélations entre les arbres

Le Random Forest est finalement constitué d'un ensemble de faible estimateur (les arbres de décisions), cependant la somme de leurs contributions fait de cet algorithme un candidat performant et très utilisé dans la pratique.

Avantages :

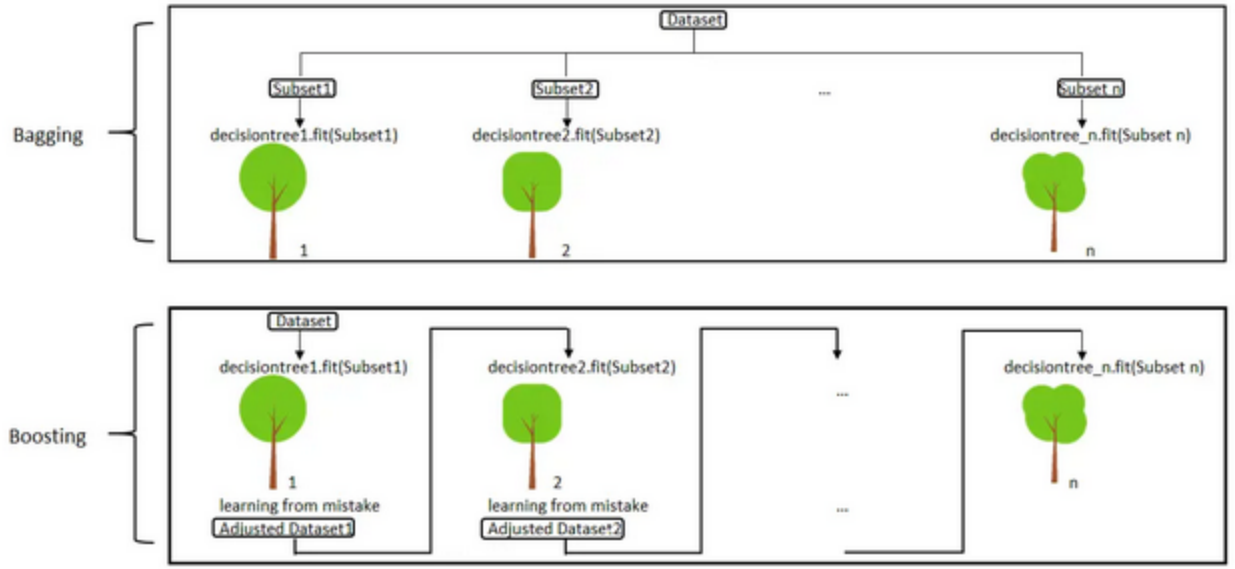
- Robuste aux outliers
- Fonctionne bien avec des données non linéaires
- Peu de risque du sur-apprentissage
- Temps de calcul efficace grâce à l'indépendance des arbres (calcul parallèle, n_jobs)

Inconvénients :

- Biais dans le traitement des variable catégorielle (favorise les variables numériques et à haute cardinalité)
- Peu efficace lorsque les données sont dispersées (sparse features, matrices creuses)

GRADIENT BOOSTING CLASSIFIER

La différence entre le **boosting** et le **bagging** est que l'ensemble des faibles estimateurs d'un modèle sont entraînés de manière séquentielle. Chacun des nouveaux estimateurs apprennent des erreurs de leurs prédécesseurs.



GRADIENT BOOSTING CLASSIFIER

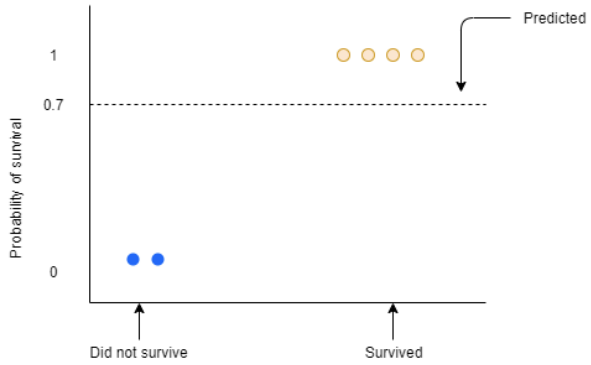
Exemple de classification sur les survivants du Titanic

Variables :

- Catégorie/classe du passager
- Age du passager
- Prix du billet
- Genre du passager
- Target : 1 si le passager a survécu

On calcul les log(odds) du dataset : log(survived/not survived)

Log(4/2) = 0,7

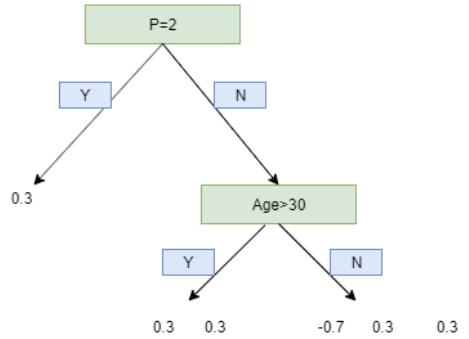


GRADIENT BOOSTING CLASSIFIER

Dataset :

$$Residual = Observed - Predicted$$

Pclass	Age	Fare	Sex	Survived	Residual
3	22	7.25	male	0	-0.7
1	38	71.2833	female	1	0.3
2	26	7.925	female	1	0.3
1	35	53.1001	female	1	0.3
3	8	21.07	male	0	-0.7
3	27	11.133	female	1	0.3



Branching out data points using the residual values

GRADIENT BOOSTING CLASSIFIER

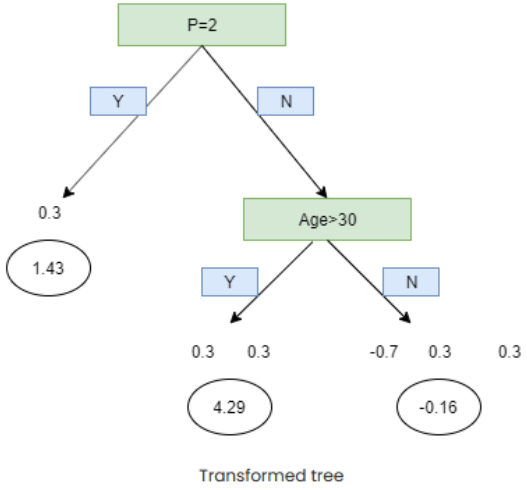
Il existe plusieurs résidus par nœud, il est usuel d'utilise la formule suivante pour calculer les résidus :

$$\frac{\sum Residual}{\sum[PreviousProb * (1 - PreviousProb)]}$$

- 1er nœud : $\frac{0.3}{[0.7 * (1 - 0.7)]} = 1.43$
- 2ème nœud : $\frac{0.3 + 0.3}{[0.7 * (1 - 0.7)] + [0.7 * (1 - 0.7)]} = 4.29$
- 3ème nœud : $\frac{-0.7 + 0.3 + 0.3}{[0.7 * (1 - 0.7)] + [0.7 * (1 - 0.7)] + [0.7 * (1 - 0.7)]} = -0.16$

GRADIENT BOOSTING CLASSIFIER

Le nouvel arbre calculé :



Une fois calculé, on applique un taux d'apprentissage pour contrôlé la contribution des nouveaux arbres (learning_rate, valeur par défaut = 0,1).

$$OldTree + LearningRate * NewTree$$

Le logodds P=2 et Age<30 pour le passager 1 : $0.7 + (0.1 * (-0.16)) = 0.684$

GRADIENT BOOSTING CLASSIFIER

Hyperparamètres :

- N_estimators : nombre d'étape de boosting
- Learning_rate: module la contribution de chacun des arbres
- Max_features: nombre de features max dans les splits
- Max_depth : profondeur maximale des arbres

Avantages :

- Performance élevée
- Flexibilité : peut-être optimisé avec différentes fonctions de pertes (loss functions) et possèdent un nombre important d'hyperparamètre à optimiser
- Fonctionne généralement bien sans avoir à effectuer de nombreux pré-traitement

Inconvénients :

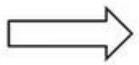
- Coûteux en temps de calcul et en mémoire
- Gradient boosting continuera à minimiser ses erreurs, cela peut engendrer du sur-apprentissage et accorder trop d'importance aux outliers
- Pour trouver les meilleurs hyperparamètres, cela demande de tester de nombreuse combinaison
- Naturellement moins facile à interpréter (outils d'interprétabilité nécessaire)

ENCODAGE DES VARIABLES CATÉGORIELLES

LabelEncoding : on assigne une unique valeur à chacune des catégories

Label Encoding

Places
New York
Boston
Chicago
California
New Jersey



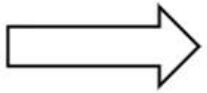
Places	Map
New York	1
Boston	2
Chicago	3
California	4
New Jersey	5

Places	Encoded
New York	1
Boston	2
New York	1
California	4
Boston	2

OrdinalEncoding : catégorie possédant un ordre naturel

Ordinal Encoding

Grades
A
B
C
D
Fail

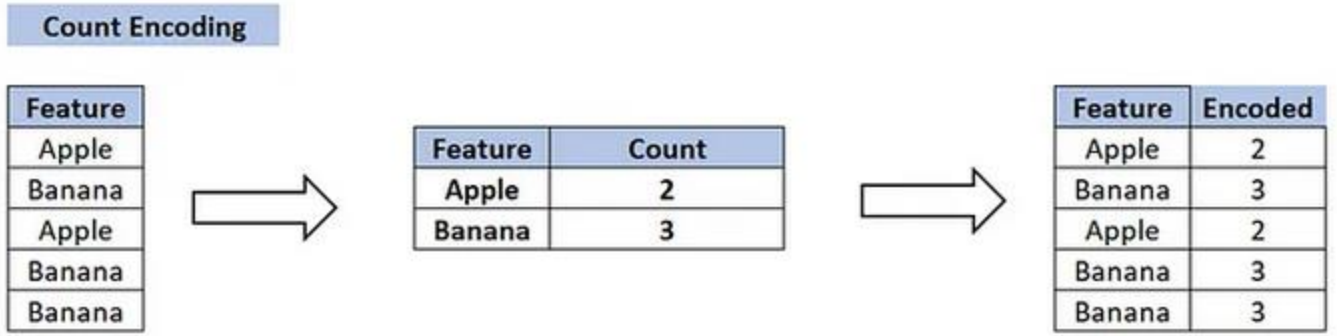


Grades	Encoded
A	4
B	3
C	2
D	1
Fail	0

ENCODAGE DES VARIABLES CATÉGORIELLES

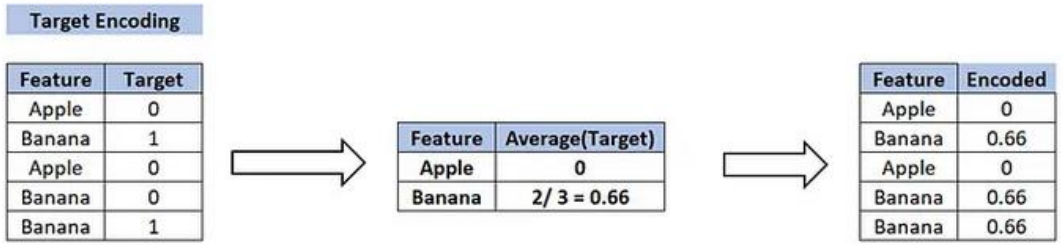
BinaryEncoding : on transforme les variables en digits

CountEncoding : comptage par nombre d'apparition d'une modalité



ENCODAGE DES VARIABLES CATÉGORIELLES

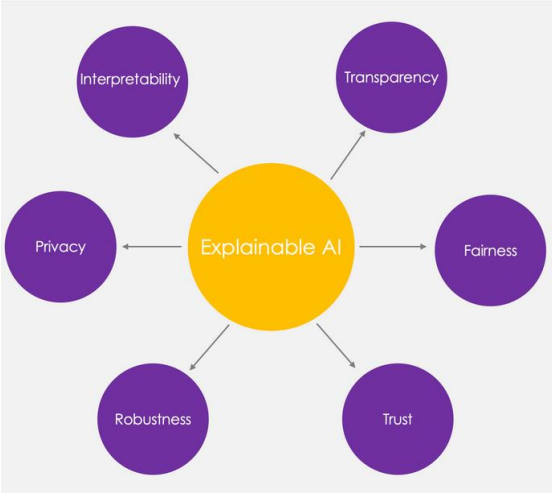
- **TargetEncoding** : on transforme les variables en digits



INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING

La complexité des modèles tend à améliorer la performance des prédictions par rapport à des modèles plus simples (régression, bayes …), cependant leurs complexités augmentent également. Ces modèles dit « black box » ont besoin de respecter certaines règles, tels que :

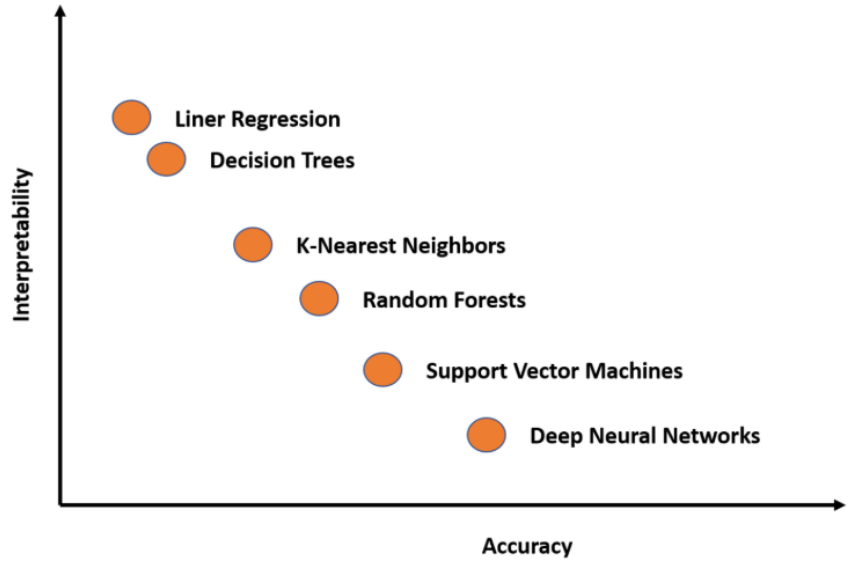
- **Transparence** : assurer à l'entité la compréhension de processus de décision
- **Fairness** : assurer que le modèle prend des décisions non discriminatoire (genre, religion, handicap)
- **Confiance**: avoir un niveau de confiance élevé dans son utilisation au quotidien
- **Robustesse**: le modèle doit être résiliant au changement des données ou des paramètres du modèle
- **Privé**: garantir la protection des informations sensible utilisé
- **Interprétabilité**: fournir aux entités un moyen de compréhension des prédictions et des résultats



INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING

Méthodes d'interprétabilité :

- Méthodes **agnostique** ou **spécifique** : les méthodes agnostiques s'utilisent pour n'importe quel type de modèles.
- Méthodes **intrinsèques** ou **post-hoc** : les méthodes intrinsèques, l'interprétabilité est directement liée à la simplicité du modèle.
- Méthodes **locales** ou **globales**: les méthodes locales donnent une interprétation pour un seul ou petit nombre d'observation. Les méthodes globales permettent d'expliquer toutes les observations.
- Méthodes **a priori** ou **a posteriori**: les approches a priori sont utilisés sans hypothèse sur les données et avant la création du modèle.



INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING

Régression Logistique : changement d'une unité change le ratio des odds d'un facteur de

$$\frac{P(y=1)}{1-P(y=1)} = odds = exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p\right)$$

$$\frac{odds_{x_j+1}}{oddsx_j} = \frac{exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_j(x_j+1) + \ldots + \beta_p x_p\right)}{exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_j x_j + \ldots + \beta_p x_p\right)}$$

$$\frac{odds_{x_j+1}}{oddsx_j} = exp\left(\beta_j(x_j+1) - \beta_j x_j\right) = exp\left(\beta_j\right)$$

Decision Tree :

Feature Importance : chaque split pour lequel une variable est utilisée, une mesure de réduction de l'impureté du Gini ou de l'entropy a été calculée. La somme des importances pour chacune des variables est calculée pour établir un classement des variables les plus discrimantes de l'arbre de décision.

Tree Decomposition : les prédictions individuelles d'un arbre de décision peuvent être expliquées en décomposant leurs « decision path ». On peut expliquer une prédiction par les contributions ajoutées à chacun des nœuds de décisions

$$\hat{f}(x) = \bar{y} + \sum_{d=1}^{D} \text{split.contrib(d,x)} = \bar{y} + \sum_{j=1}^{p} \text{feat.contrib(j,x)}$$

Les arbres de décisions ne captent pas les effets linéaires entre les variables et la variable cible

INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING

Naives Bayes: probabilité conditionnelle sous hypothèse d'indépendance

$$P(C_k|x) = \frac{1}{Z} P(C_k) \prod_{i=1}^{n} P(x_i|C_k)$$

K-Nearest Neighbors:

- Absence d'interprétation globale
- Interprétation locale en regardant le k-voisinage d'une observation

Random Forest & Gradient Boosting :

- Moyenne des contributions individuelles des variables dans l'ensemble des arbres de décisions
- Calcul de l'importance des variables très rapide
- Préférence pour des variables numériques ou variable catégorielle avec une cardinalité élevée
- Possibilité de négliger une variable importante corrélée avec une autre variable

INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING

Permutation Based Feature Importance

1. Entraîner un modèle de ML
2. Sélectionner une métrique d'évaluation de performance
3. Evaluer la performance sur le jeu de validation
4. Bruiter/Mélanger une feature du modèle
5. Mesurer la performance du modèle avec la feature modifiée
6. Mesurer la différence de performance

Attention : technique très sensible aux variables très corrélées

Out[2]:

Weight	Feature
0.1750 ± 0.0848	Goal Scored
0.0500 ± 0.0637	Distance Covered (Kms)
0.0437 ± 0.0637	Yellow Card
0.0187 ± 0.0500	Off-Target
0.0187 ± 0.0637	Free Kicks
0.0187 ± 0.0637	Fouls Committed
0.0125 ± 0.0637	Pass Accuracy %
0.0125 ± 0.0306	Blocked
0.0063 ± 0.0612	Saves
0.0063 ± 0.0250	Ball Possession %
0 ± 0.0000	Red
0 ± 0.0000	Yellow & Red
0.0000 ± 0.0559	On-Target
-0.0063 ± 0.0729	Offsides
-0.0063 ± 0.0919	Corners
-0.0063 ± 0.0250	Goals in PSO
-0.0187 ± 0.0306	Attempts
-0.0500 ± 0.0637	Passes

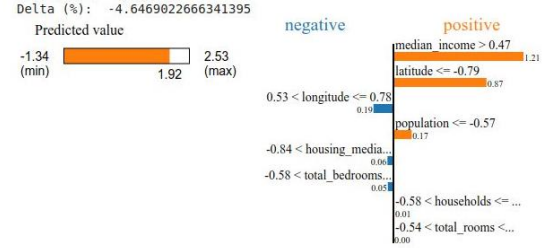
LIME (Local Interpretable Model-agnostic Explanation)

LIME est un modèle local qui cherche à expliquer la prédiction d'un individu par analyse de son voisinage

1. Génération de nouvelles données dans un voisinage proche de l'individu à expliquer
2. Entrainement d'un modèle sur les prédictions du modèle « black box » (ex: régression linéaire)
3. Expliquer les prédictions à partir du modèle local

LIME est généralement instable en termes d'interprétabilité

```
Index:  1
Scaled Prediction:  0    1.923631
Name: 1, dtype: float64
Scaled Actual:      median_house_value    1.758632
Name: 7766, dtype: float64
Predicted value:  429366.2399999999
Real value:  410300.0
Delta (%):  -4.6469022666341395
```



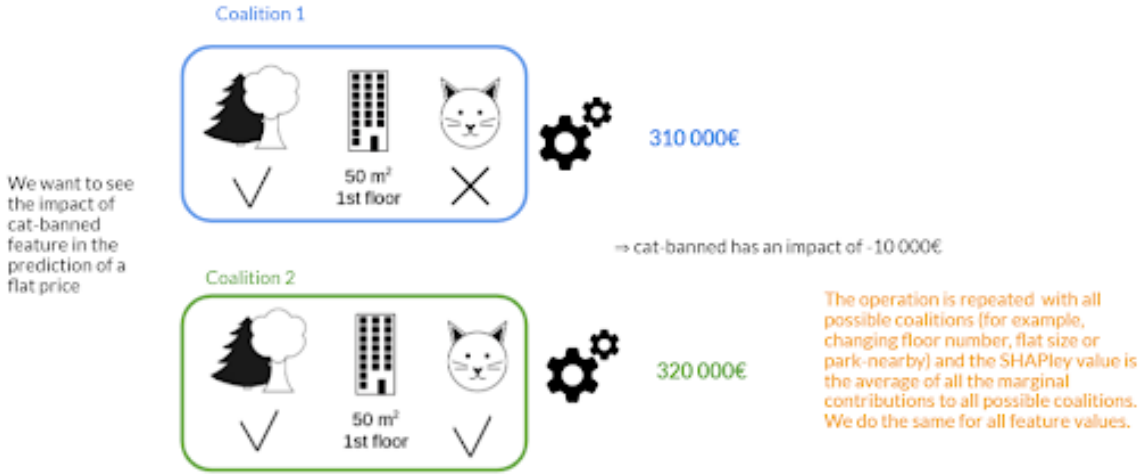
INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING

SHAP Values (Shapley Additive Explanations)

Basé sur la théorie des jeux en économie

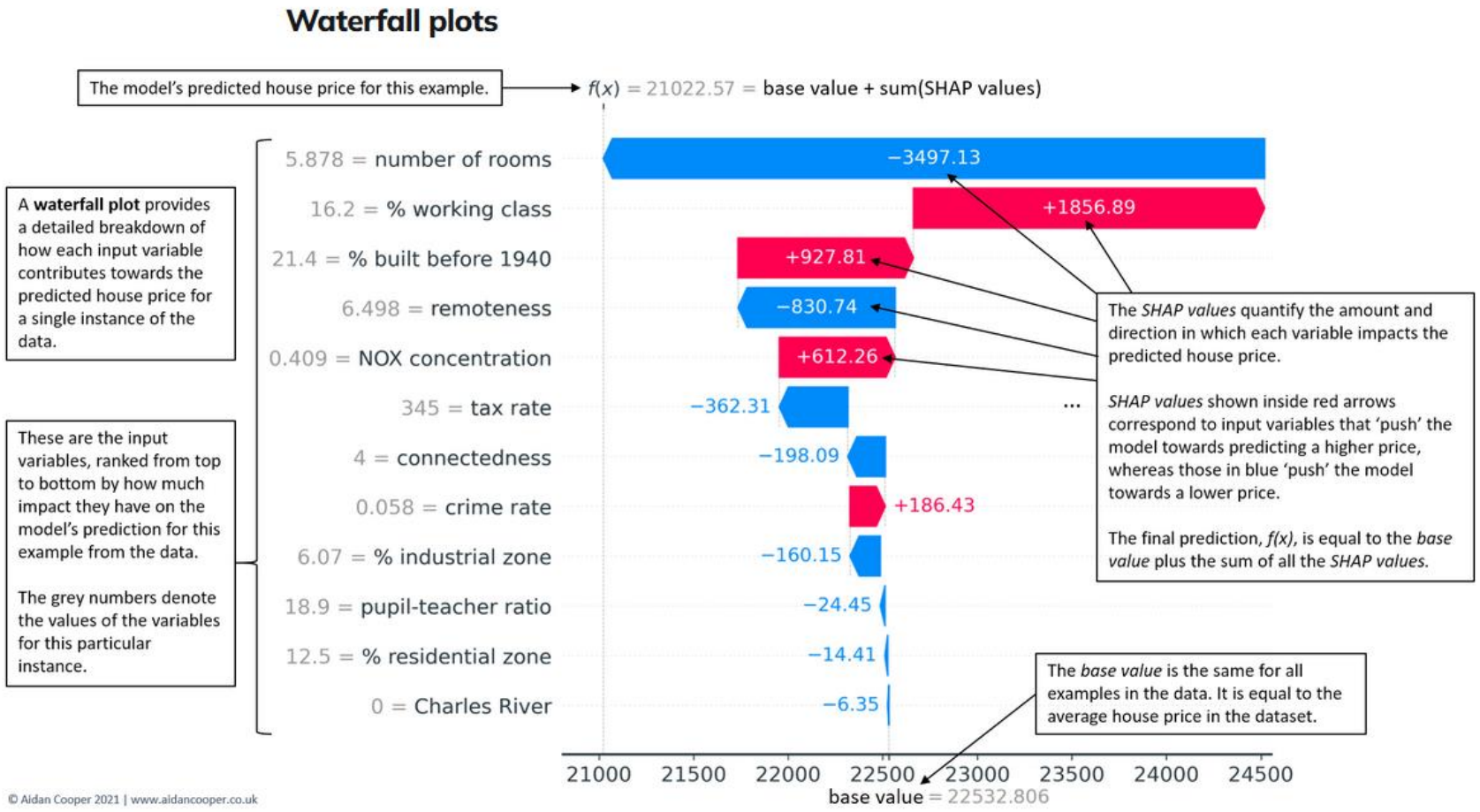
Définition : pour une observation, la valeur de shapley d'une variable est sa contribution à la différence entre la valeur prédite par le modèle et la moyenne des prédictions de tous les individus

1. Calcul des valeurs de Shapley pour une observation : simuler différentes combinaisons de valeurs pour les variables d'entrées
2. Pour chaque combinaison, calculer la différence entre la valeur prédite et la moyenne des prédictions.



INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING

SHAP Values, Exemple :



INTERPRÉTABILITÉ DES MODÈLES DE MACHINE LEARNING