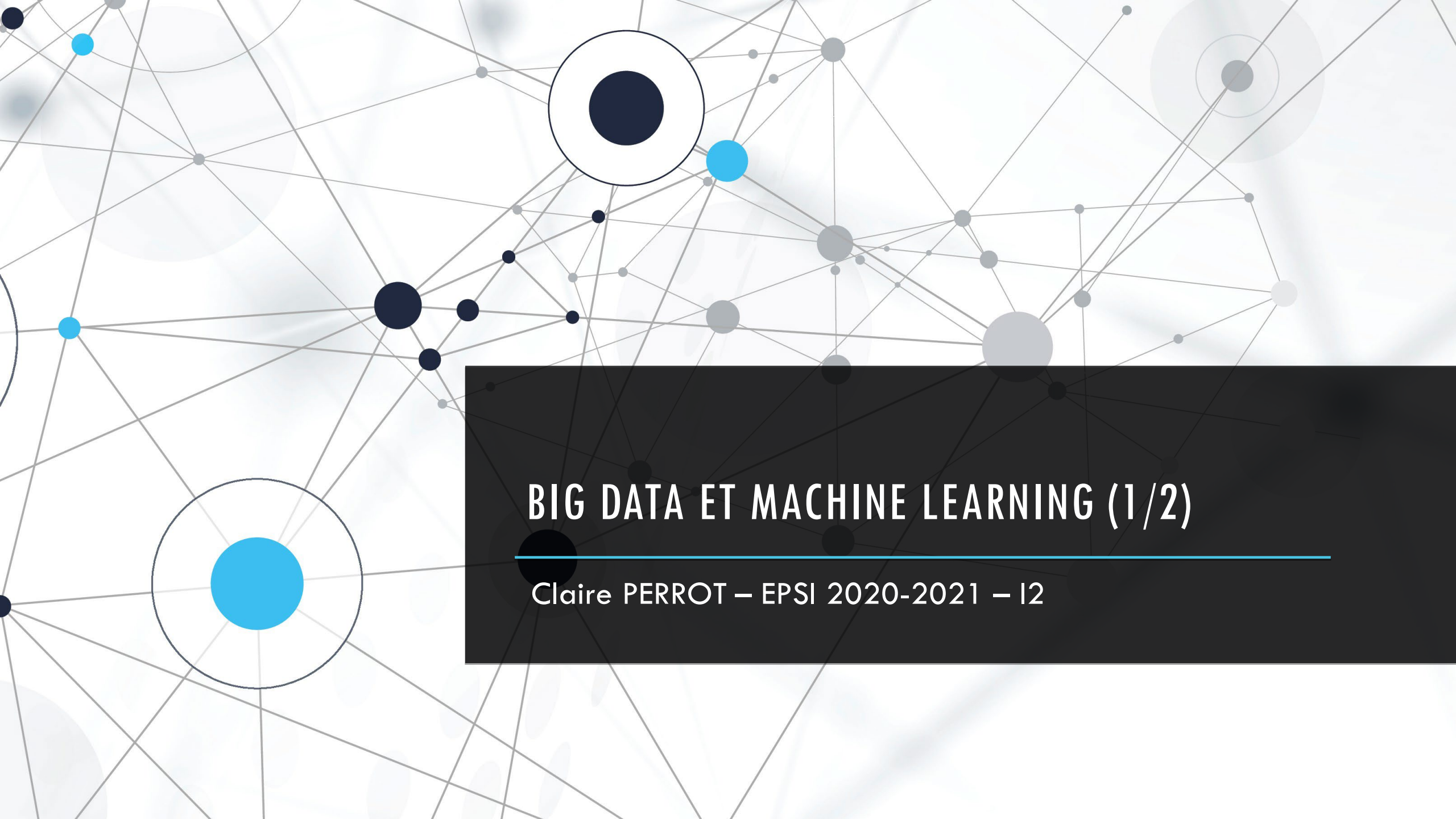


BIG DATA ET MACHINE LEARNING (1/2)

Claire PERROT – EPSI 2020-2021 – I2

OBJECTIFS DU MODULE

- Découvrir les différents types d'apprentissages : supervisé / non-supervisé / par renforcement…

- Connaître des critères d'évaluation d'un modèle d'apprentissage pour des problèmes de classification et de régression, et leurs limites

- Connaître et implémenter différents algorithmes d'apprentissage supervisé

- Implémenter l'architecture Map/Reduce sur des exemples simples

- Découvrir l'apprentissage par renforcement à travers un exemple long

ORGANISATION DU MODULE

20h de module (6h en autonomie, 14h avec l'intervenante)
Prérequis : langage Python (bases), notions de statistiques, apprentissage non-supervisé

Notes de cours / slides / fichiers / rendus : sur MyLearningBox
https://mylearningbox.reseau-cd.fr/course/view.php?id=24699

Evaluation :
- Exercices guidés à rendre (fin de séance 4)
- QCM sur MyLearningBox (à faire avant le 1er mars)
- Projet à rendre (14 mars)

Contact : claire.perrot@campus-cd.com

C'EST QUOI LE MACHINE LEARNING ?

Machine learning = Apprentissage automatique

/!\ Machine learning > Deep learning (apprentissage profond)

Apprentissage automatique = transformer des données en nombres et trouver des motifs dans ces nombres

Développement du machine learning grâce au Big Data (grandes bases de données, machines plus rapides…)

Qualité du machine learning = dépend principalement de la qualité et du nombre de données fournies à l'algorithme

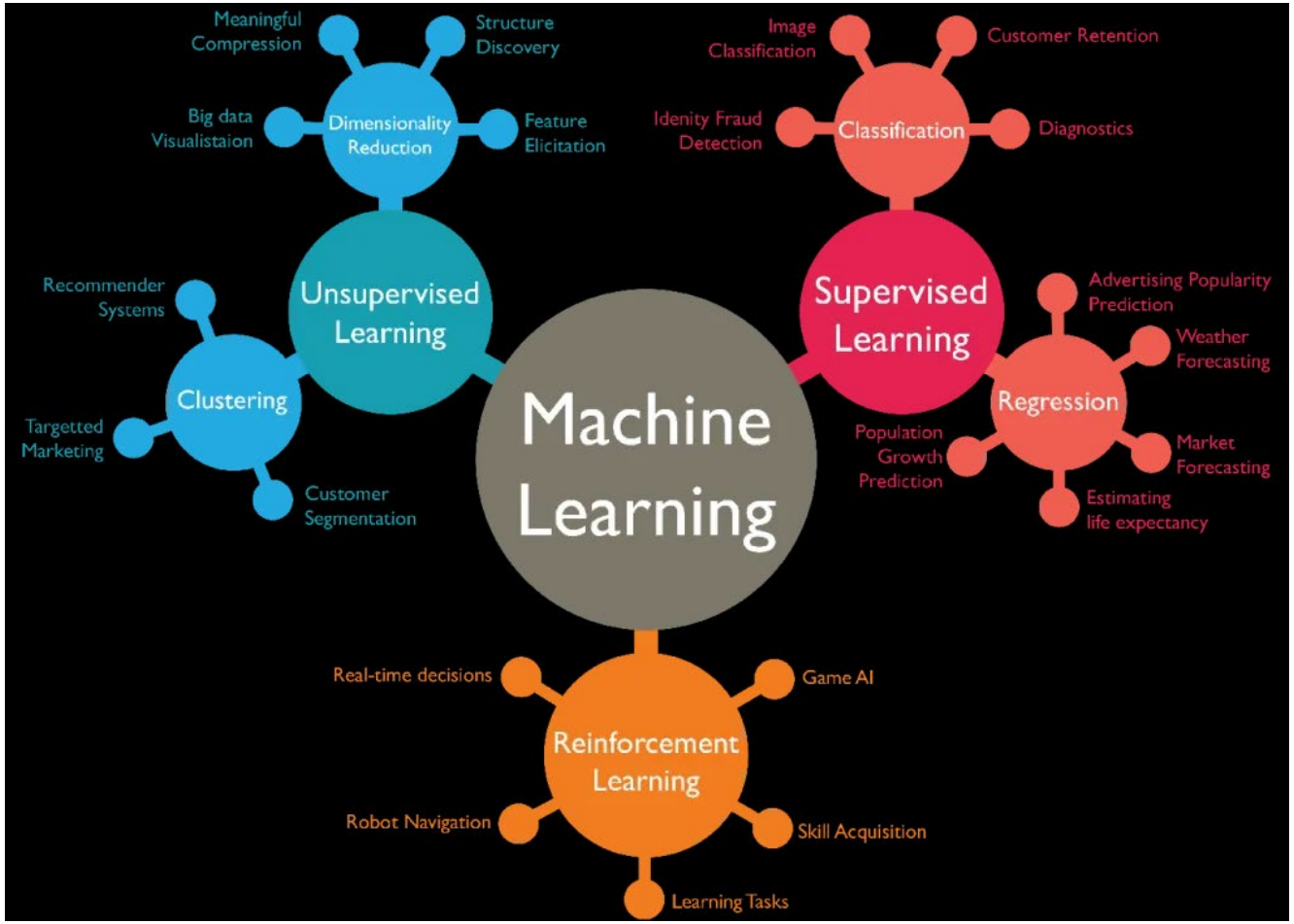
LES PROBLÈMES D'APPRENTISSAGE

Différents types de problèmes d'apprentissage :

- **Non-supervisé** : on doit découvrir la structure des données (vu en module « Datamining » en I1)

- **Supervisé** : on connaît la structure des données

- **Par renforcement** = on cherche à maximiser une valeur et l'algo « retient » les essais précédents pour s'améliorer (utilisation d'un agent dans un environnement prédéfini)



APPRENTISSAGE SUPERVISÉ

1. On fixe le type de problème étudié (classification, régression)
2. On établit un corpus d'apprentissage (données de départ + résultats attendus = données *étiquettées*)
3. On sépare les données en deux groupes (entrainement + test)
4. On choisit un algorithme conforme au type de problème (*)
5. On lance l'algorithme sur les données d'entrainement
   -> Entrainement d'un modèle (optimisation d'une fonction d'erreur)
6. On évalue le modèle sur les données de test (inconnues du modèle et dont on connait le résultat attendu)
7. (Repeat)

(*) Attention, certains algorithmes peuvent être utilisés pour plusieurs types de problèmes différents !

PROBLÈME DE CLASSIFICATION

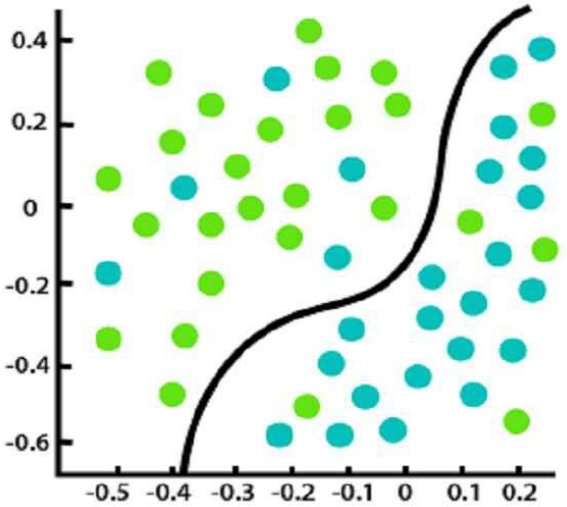
Problème : identifier la classe (= catégorie) d'une donnée

Connu : un ensemble de données ayant une ou plusieurs classes (ensemble d'entrainement)

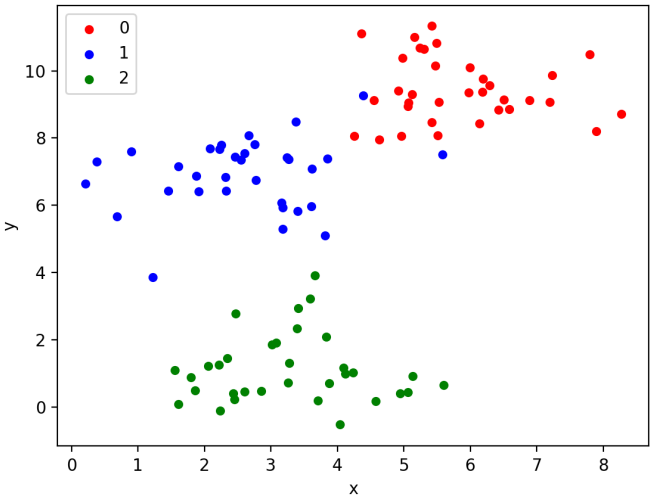
/!\ Classification != Clustering <- création de classes à partir des données

Classification binaire / multiclasses / un-contre-reste …

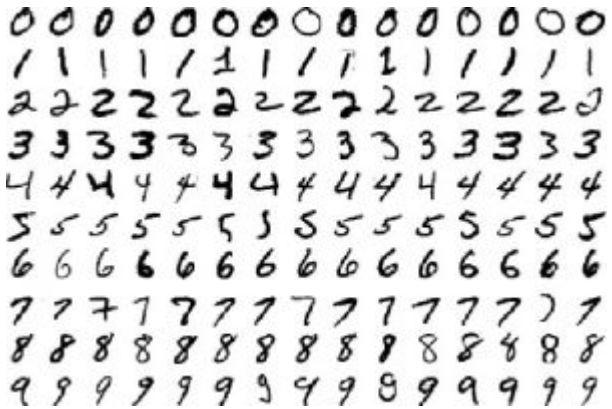
Exemples :
- dire si un patient a la maladie X ou Y en fonction des symptômes
- reconnaître des chiffres manuscrits
- prédire la mention d'un étudiant au bac…

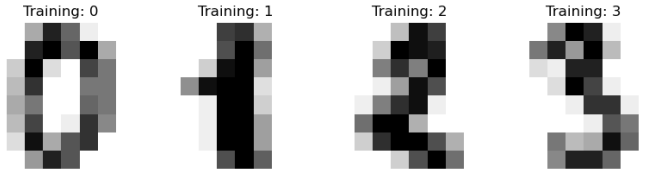


Classification



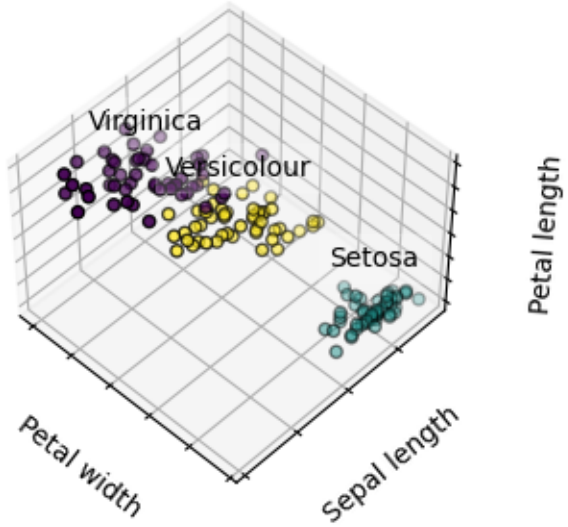
CLASSIFICATION – EXEMPLES





Données étiquettées : 1797 images 8x8 de chiffres manuscrits, chaque pixel ayant une valeur entre 0 et 16.

Données étiquettées : iris de 3 espèces différentes avec 4 propriétés (longueur et largeur des sépales et des pétales)



PROBLÈME DE RÉGRESSION

Problème : Déterminer la valeur d'une variable y à partir de N variables qui sont (à priori) corrélées

Connu : un ensemble de données pour lesquelles on connait la valeur de y

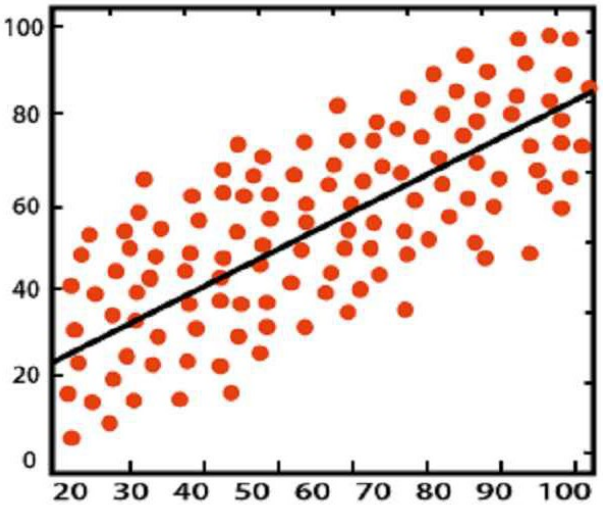
-> Prédire une valeur quantitative

Plusieurs types de « courbes » : régression linéaire (droite), régression logistique (fonction logistique)…

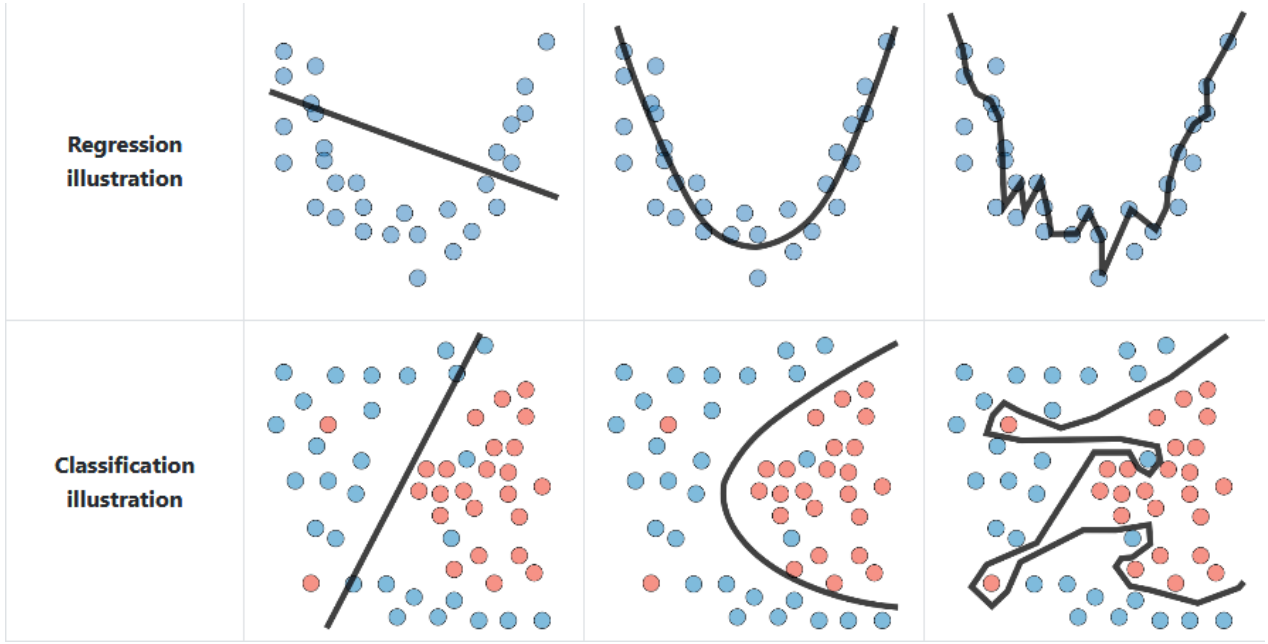
Régression linéaire simple si N=1 (x ne dépend que d'une seule variable, y = ax+b)

Exemples :

- déterminer le prix d'une maison à partir de plusieurs critères
- déterminer l'âge d'un client à partir de son historique de visite



OVERFITTING ET UNDERFITTING



Underfitting Overfitting

Underfitting = sous-apprentissage

Le modèle ne peut pas être déduit des données (mauvais choix d'algorithme ou de paramètres), on ne peut pas généraliser

Overfitting = surapprentissage

Le modèle « colle » trop aux données d'entraînement et ne généralise pas
-> bon score sur l'entrainement et mauvais sur le test

Le modèle apprend des exemples par cœur au lieu de déduire des motifs génériques

Les données d'entraînement doivent être assez diverses !

BIAIS ET VARIANCE

Biais (*Bias*) : le biais d'un algorithme est faible s'il fait peu d'hypothèses sur la forme du modèle final (exemple : la régression linéaire a un biais élevé car on suppose que la fonction est une droite)

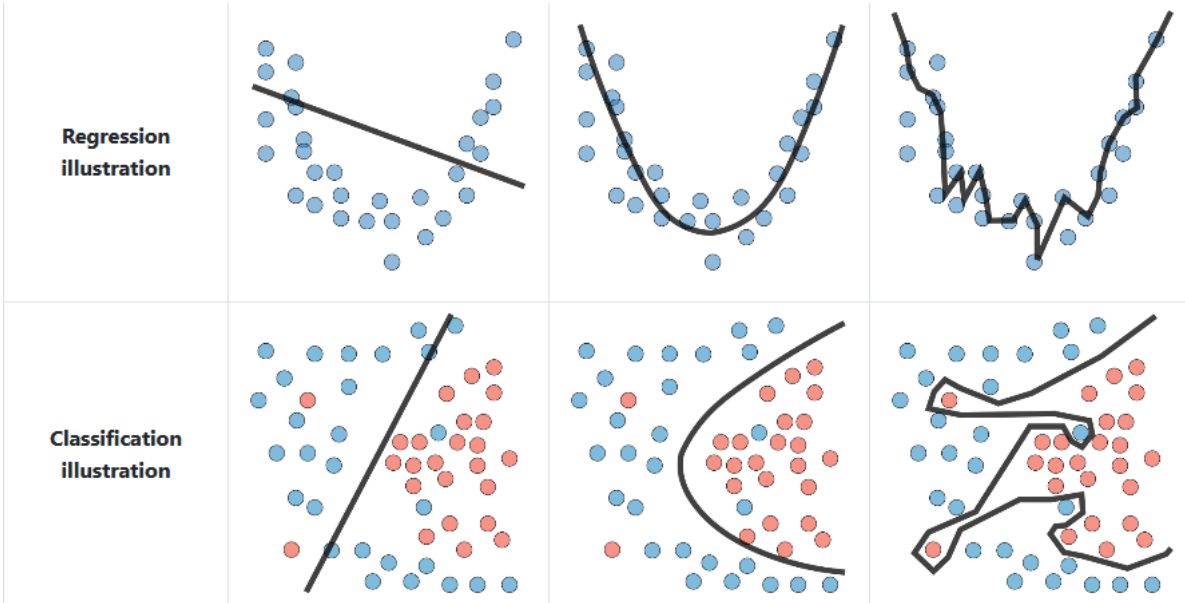
Variance : la variance d'un algorithme est la sensibilité aux changements dans les données d'entraînement.

Erreur finale d'un modèle = Biais + Variance + ε

-> On veut un faible biais et une faible variance

/!\ C'est impossible (*« there is no free lunch »*)

Il faut trouver un **compromis**



Biais +
Variance -

Biais –
Variance +

ÉVALUER UN MODÈLE DE CLASSIFICATION BINAIRE

Problème de classification binaire : les données font partie d'une classe parmi deux

Matrice de confusion :

		Classe réelle	
		-	+
Classe prédite	-	True Negatives *(vrais négatifs)*	False Negatives *(faux négatifs)*
	+	False Positives *(faux positifs)*	True Positives *(vrais positifs)*

Mesures intéressantes :
Recall (rappel) = **sensibilité** = TP / (TP + FN)
« est-ce que les vrais positifs sont détectés »

Précision = TP / (TP + FP)
« est-ce que les positifs prédits sont vrais »

Spécificité = TN / (FP + TN)
« est-ce que les vrais négatifs sont détectés »

Score global : F1-score (ou mieux, MCC – Matthews correlation coefficient !)

ÉVALUER UN MODÈLE DE CLASSIFICATION BINAIRE

Modèle 1

		Actual class	
		Cat	Dog
Predicted class	Cat	5	2
	Dog	3	3

Calculer la sensibilité, la précision, la spécificité et le score F du modèle 1. Qu'en déduisez-vous ?

Modèle 2

		Actual class	
		Dog	Cat
Predicted class	Dog	3	3
	Cat	2	5

Calculer la sensibilité, la précision, la spécificité et le score F du modèle 2. Qu'en déduisez-vous ?

Recall (rappel) = **sensibilité** = TP / (TP + FN)
« est-ce que les vrais positifs sont détectés »

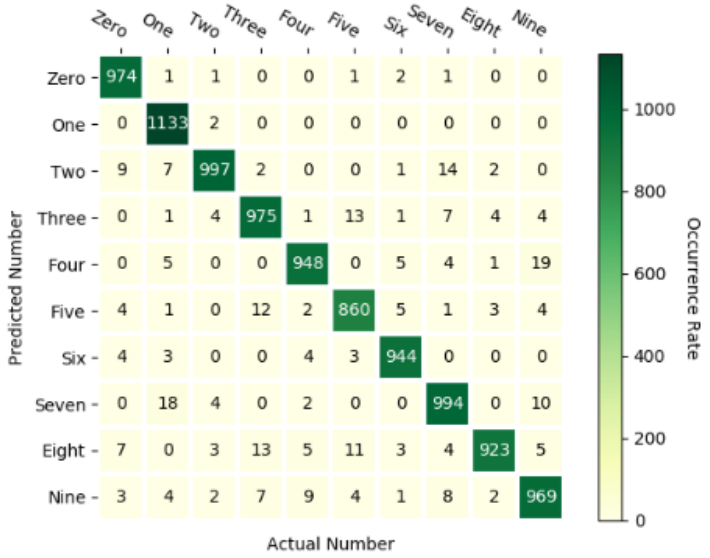
Précision = TP / (TP + FP)
« est-ce que les positifs prédits sont vrais »

Spécificité = TN / (FP + TN)
« est-ce que les vrais négatifs sont détectés »

Score F = 2*(precision*sensibilité) / (precision + sensibilité)

ÉVALUER UN MODÈLE DE CLASSIFICATION GÉNÉRALISÉE

Matrice de confusion : utile pour avoir une vision globale du comportement du modèle



Utilisation des notions de précision / recall / score F1 (par classe) ->

On peut toujours utiliser le MCC

	precision	recall	f1-score	support
0	1.00	1.00	1.00	43
1	0.95	1.00	0.97	37
2	1.00	1.00	1.00	38
3	0.98	0.98	0.98	46
4	0.98	0.98	0.98	55
5	0.98	1.00	0.99	59
6	1.00	1.00	1.00	45
7	1.00	0.98	0.99	41
8	0.97	0.95	0.96	38
9	0.96	0.94	0.95	48
avg / total	0.98	0.98	0.98	450

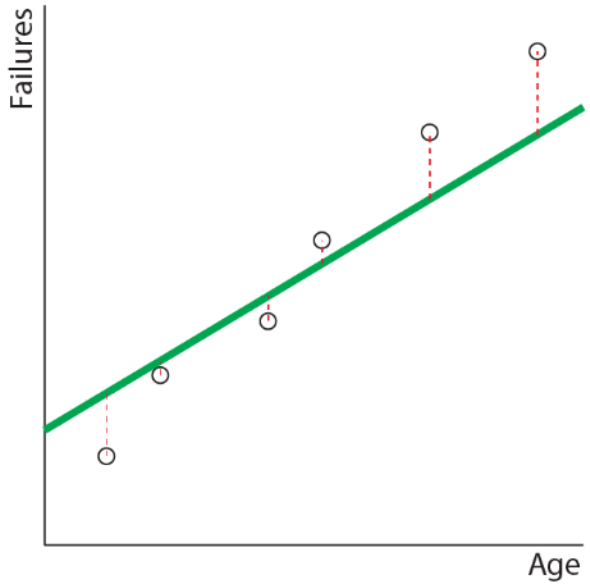
Pour aller plus loin

Propagation des étiquettes (label propagation) et évaluation
[démo (scikit-learn)]

Validation croisée
https://en.wikipedia.org/wiki/Cross-validation_(statistics)
https://machinelearningmastery.com/k-fold-cross-validation/

ÉVALUER UN MODÈLE DE RÉGRESSION

Comment quantifier la performance du modèle ?



Age	Failures	Prediction	Error
10	15	26	11
20	30	32	2
40	40	44	4
50	55	50	-5
70	75	62	-13
90	90	74	-16

MAE (Mean Absolute Error) moyenne des valeurs absolues des erreurs

$$MAE = \frac{\Sigma|y - \hat{y}|}{N}$$

RMSE (Root mean square error)

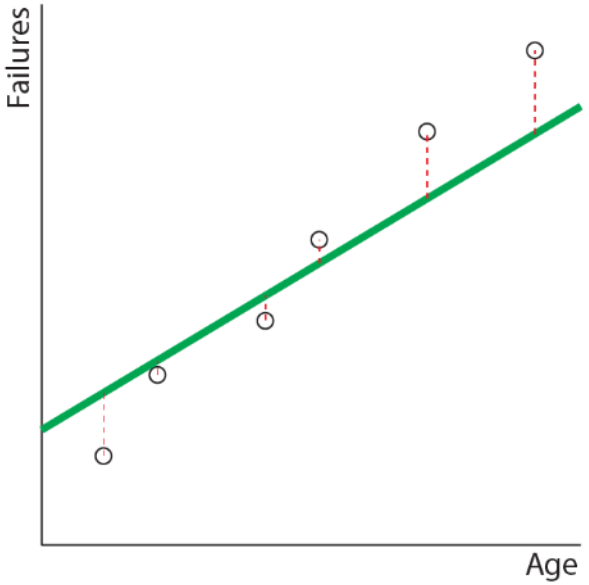
$$RMSE = \sqrt{\frac{\Sigma(y - \hat{y})^2}{N}}$$

Donne plus de poids aux grandes erreurs + est dans l'unité de la mesure

R^2 = variance par rapport à la droite moyenne

$$R^2 = \frac{\Sigma(y - \bar{y})^2 - \Sigma(y - \hat{y})^2}{\Sigma(y - \bar{y})^2}$$

ÉVALUER UN MODÈLE DE RÉGRESSION



Age	Failures	Prediction	Error
10	15	26	11
20	30	32	2
40	40	44	4
50	55	50	-5
70	75	62	-13
90	90	74	-16

Calculer le MAE, RMSE, et R^2 pour ce modèle (la droite moyenne est à y = 50.8).

Limitations du calcul de R^2 :
Si on fait des régressions avec de plus en plus de variables, R^2 ne fera qu'augmenter.

Pour limiter cela, on peut calculer « adjusted R^2 », qui prend en compte le nombre de variables dans son calcul (moins notre modèle utilise de variables, et plus cette valeur sera bonne).

K PLUS PROCHES VOISINS (= K NEAREST NEIGHBORS)

Algo utilisé à la fois en classification et en régression

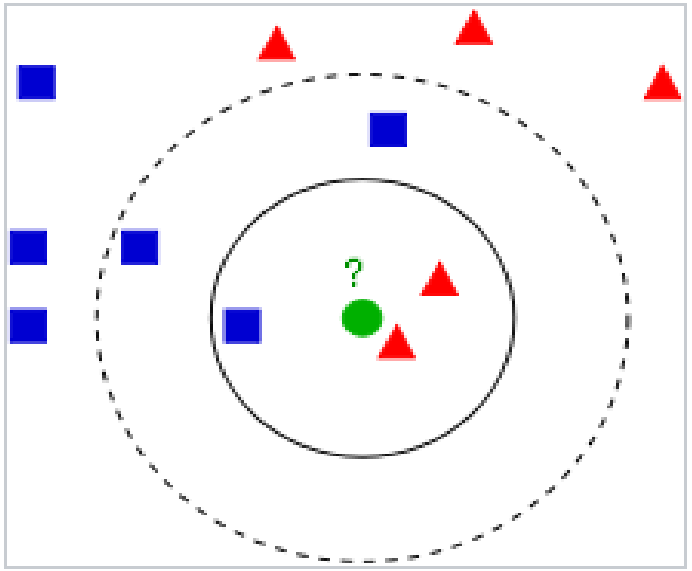
Pas d'entraînement à proprement parler

Données déjà étiquetées (valeurs ou classes) et placées dans un espace

Nouvelle donnée : on regarde les k plus proches voisins de ce point

Classification : on prend la classe majoritaire parmi ces k voisins

Régression : la valeur du point est la moyenne de celles des k voisins



K PLUS PROCHES VOISINS (= K NEAREST NEIGHBORS)

Difficulté : comment fixer k ? Si k est trop grand, on augmente le biais et on diminue la variance (*hyperparameter optimization*)

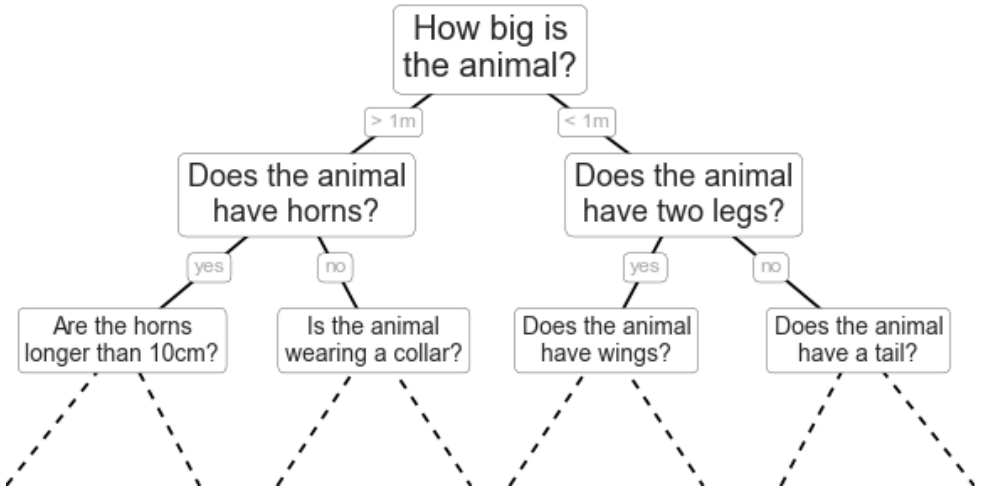
Possibilité de pondérer la contribution des voisins pour favoriser les voisins proches (*weighted k-NN*)

Normalisation des données recommandée

Si besoin (+ de 10 dimensions) : souvent une PCA avant cet algo (diminuer le # de dimensions)

Si énormément de dimensions (flux vidéo, séries temporelles, analyse d'ADN…) : approximation de k-NN au lieu de l'algo classique

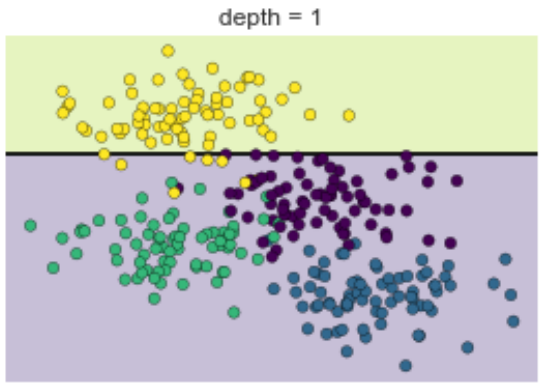
ARBRES DE DÉCISION

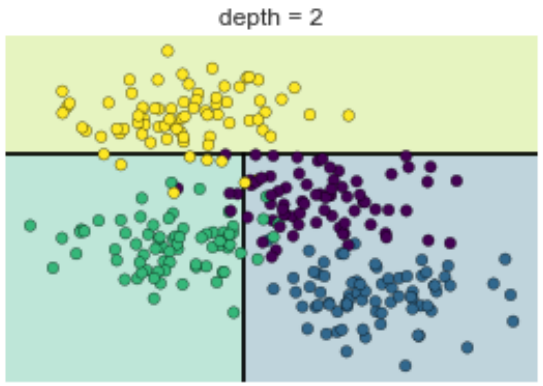


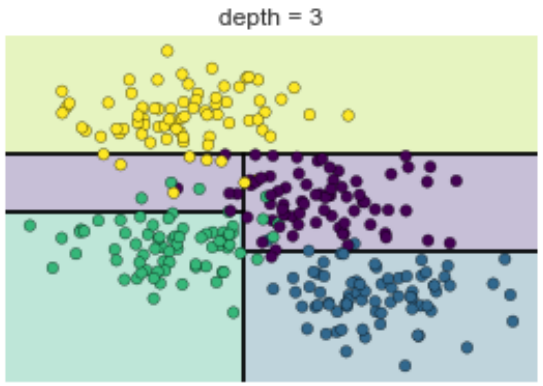
-> Principalement algorithme pour de la classification

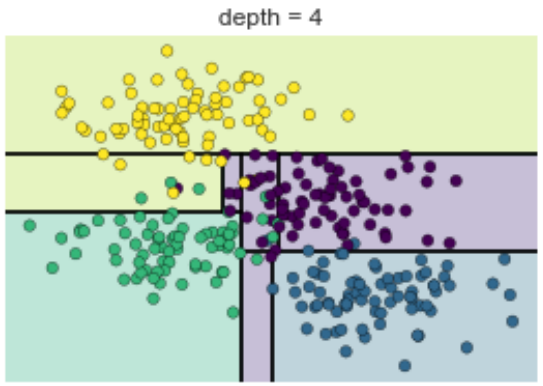
L'algo construit un arbre binaire qui retrouve au mieux les classes de données

Profondeur de l'arbre paramétrable
/!\ il faut limiter la profondeur !









ARBRES DE DÉCISION

Algo qui gère aussi des propriétés non numériques et des problèmes à multi-sortie (multi-output problems)

Utilisable aussi pour faire de la régression

Attention à l'overfitting !

Le modèle est meilleur si les données sont équilibrées dans les différentes classes

Non-robuste (petit changement dans les données → grand changement dans l'arbre)
-> Mise en place d'aléatoire : forêts aléatoires (*random forests*)

POUR ALLER PLUS LOIN…

Autres algorithmes :

- Naive Bayes
- Autres algorithmes de régression
- Support vector machines

Problèmes multi-sorties (ex : compléter un visage)
Problèmes « sequence-to-sequence » (ex : traitement automatique des langues / *natural language processing – NLP*)

Des chaînes Youtube intéressantes : StatQuest

MACHINE LEARNING EN PYTHON

Python : langage très utilisé dans le machine learning (aussi : R, Tanagra…)

Ensemble de modules **scikit-learn** : https://scikit-learn.org/
basé sur les modules numpy, scipy, matplotlib …

Grande documentation et nombreux exemples, données …

Dernière version en date : 0.24.0 (décembre 2020)

Attention : requiert Python 3.6 ou plus !

EXERCICES GUIDÉS (A RENDRE !)

Utilisation des notebooks Python avec Google Colab : https://colab.research.google.com

Fichier **k-NN_decision_trees.ipynb**

A compléter et rendre avant le 14 février à 23h59 (sur MyLearningBox exclusivement)

Par groupes de 2 ou 3 apprenants possible

MERCI DE RENOMMER LE FICHIER AVEC VOS NOMS

SÉANCE EN AUTONOMIE – 1ER MARS

- TP « Big data » (disponible sur MyLearningBox) à faire sur Google Colab
- Rappels de Python : cheatsheet sur MLB
- Pour aller plus loin : une vidéo sur les modèles SVM (Support Vector Machines)
- une vidéo générale sur le machine learning (très complète, attention 2h37 !)