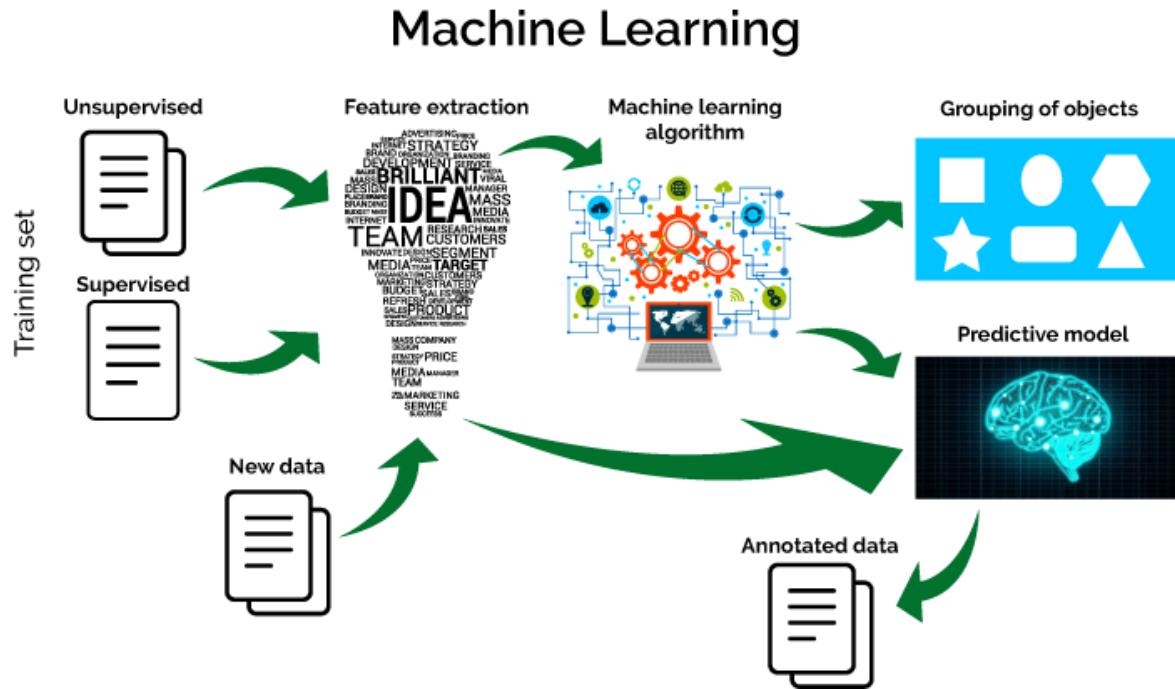


Auteur: CAMARA Laby Damaro

Email: [Idamaro98@gmail.com](mailto:Idamaro98@gmail.com)

Github: <https://github.com/camara94>



## Comment Choisir le Modèle Approprié à Un Problème de AI

Le Machine Learning, aussi appelé apprentissage automatique en français, est une forme d'intelligence artificielle permettant aux ordinateurs d'apprendre sans avoir été programmés explicitement à cet effet.

En Machine Learning, il existe une tonne de modèles: La régression logistique, les arbres de décision, les support vector machine, les k-nerest neighbors, les random forests, les réseaux de neurones, ....

Bref il est difficile faire son choix parmi toutes ces possibilités, dans ce document nous allons explorer quelques possibilité de critère de choix de modèle pour selectionner le bon modèle pour son projet en machine learning.

### Conseil #1

Ne travailler qu'avec les modèles que l'on comprend vraiment. Car nous permet d'optimiser et régler ses hyper paramètres et ensuite cela nous permet également d'éviter d'utiliser un modèle dans un contexte qu'il n'est pas adapté pour ne pas avoir des comportements anormaux sans qu'on ne les comprennent.

Il vaut mieux du bon travail sur un modèle basique que du mauvais travail sur des modèles sofistiqués.

## Conseil #2

Par les modèles que vous comprenez, commencez toujours par implementer le plus simple possible. Car la plupart des modèles sofistiqués consomment beaucoup de ressources et prennent beacoups de temps pour exécuter et complexes à regler.

### Exemple

Par exemple si vous travailliez sur un modèle de régression commencez toujour par:

- LinearRegression
- Lasso
- Ridge

Si vous travailliez sur problème de classification commencez par:

- LogisticRegression
- LinearSVC
- KNeighborsClassifier

Car ce sont des modèles simples, facile à entrainer et facile regler également.

## Critère de choix du modèle d'un Data Scientiste

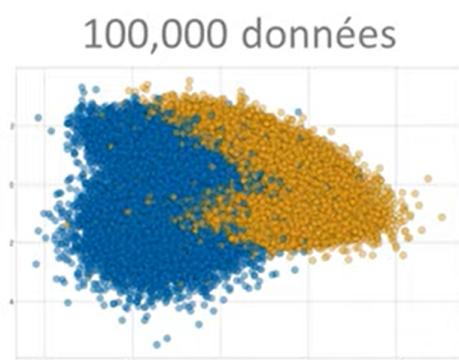
Quelles sont les critères qui permettent à un Data Scientiste de choisir un modèle plutôt qu'un autre.

Par exemple choisir un arbre decision plutôt qu'une régression Logistique.

### Critère #1: Quantité de données

Le premier critère de choix d'un modèle est la quantité de donée avec laquelle nous travaillons.

#### Modèle adaptés aux gros Datasets

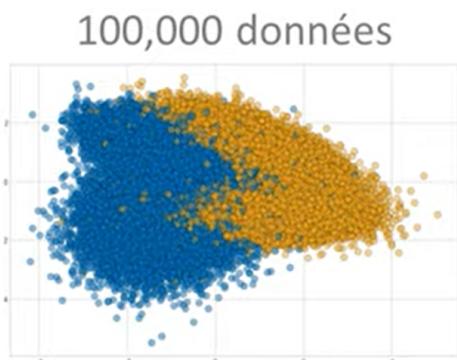


Certains modèles peuvent travailler avec de **gros datasets** (+100,000 données)

- Régression Logisitique
- Réseaux de Neurones

- Régression Logisitique
- Réseaux de Neurones

## Modèle adaptés aux petits Datasets



D'autres modèles **sont très lents** pour manipuler toutes ces données...

- Support Vector Machines
- K-Nearest Neighbours



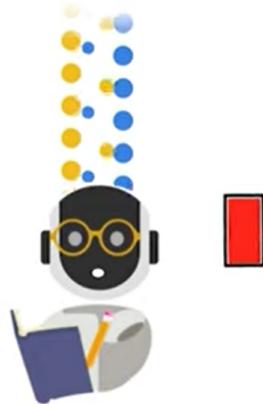
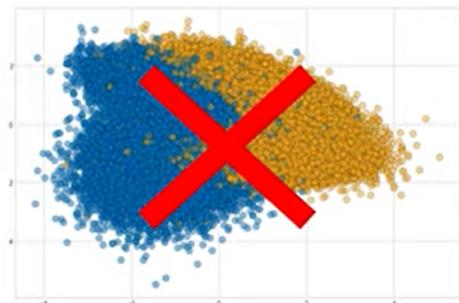
Qu'est ce qui explique cela, le fonctionnement des algorithmes (d'où le premier conseil #1: ne travaillez qu'avec des algorithme qu'on comprend).

### Par exemple KNeighborsClassifier

Lorsqu'on comprend le fonctionnement d'un **KNeighborsClassifier** alors on sait pertinemment qu'on ne pas lui fournir un million de données car il stocke toutes les données en mémoire pour ensuite calculer la distance entre chaque point de notre dataset.

Donc si on a un million de points dans notre dataset ce modèle va peser très sur mémoire amis aussi il va être très lent car il va calculer la distance entre chaque de notre dataset.

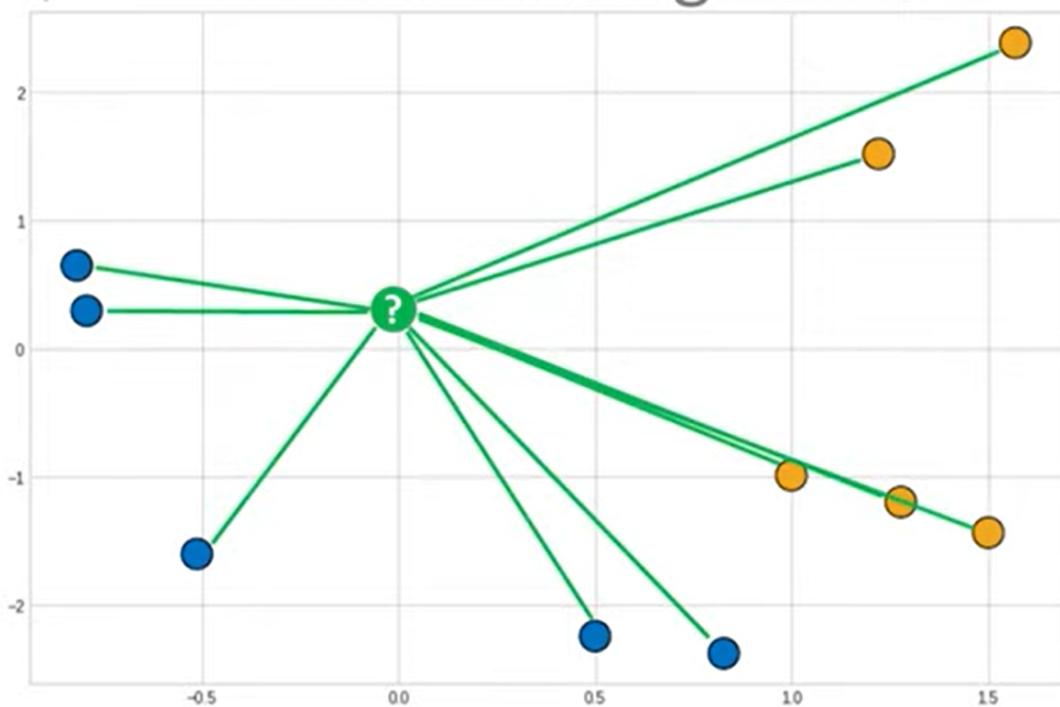
100,000 données



D'autres modèles **sont très lents** pour manipuler toutes ces données...

- Support Vector Machines
- K-Nearest Neighbours

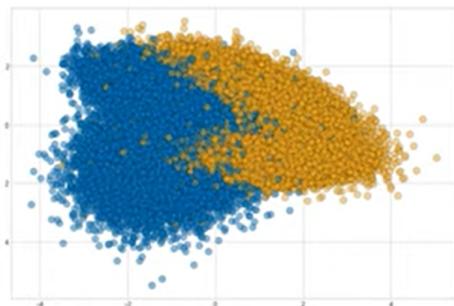
Trop de données !



## Régression Logistique

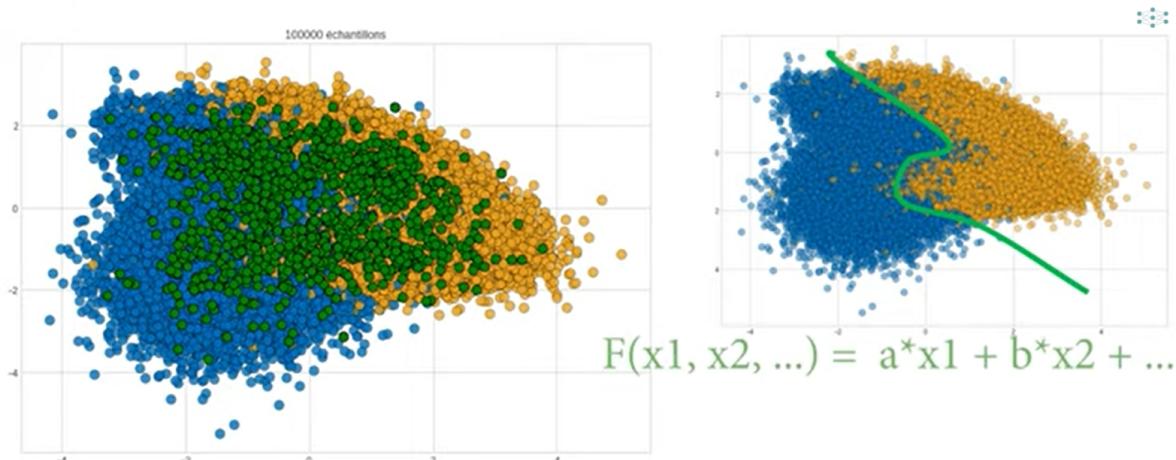
A l'inversion si on utilise une régression logistique qui repose sur un algorithme de descente de gradient, nous pouvons progressivement fournir nos données à la descente de gradient par petit batch ce qui permettra d'entraîner notre machine beaucoup plus rapidement pour ensuite obtenir un modèle de régression de régression logistique qui est une simple fonction linéaire donc c'est quelque chose de beaucoup plus léger que de stocker un million de points dans la mémoire d'un **KNeighborsClassifier**.

100,000 données



Certains modèles peuvent travailler avec de **gross datasets** (+100,000 données)

- Régression Logistique
- Réseaux de Neurones → **Descente de Gradient**



En analysant les données par **petits Batchs**, la **descente de gradient** est capable d'apprendre à partir de grands datasets.

## NB

Critère 1 : Quantité de données

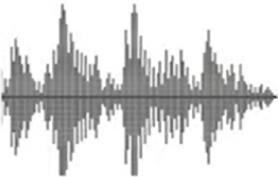
- 100,000 → N'importe quel modèle de ML
- + 100,000 → Descente de Gradient
  - SGDRegressor
  - SGDClassifier
  - Neural Nets - Tensorflow, Torch

## Critère #2: Données Structurées / Non Structurées

Le deuxième critère de sélection de modèle repose sur la structure des données.

### Données Non-Structurées

En effet en machine learning, on considère que les images, les textes, les sons, ... sont des données non-structurées.



Lorsqu'on a à faire avec les données non-structurées il est préférable d'utiliser les algorithmes de **Deep Learning**

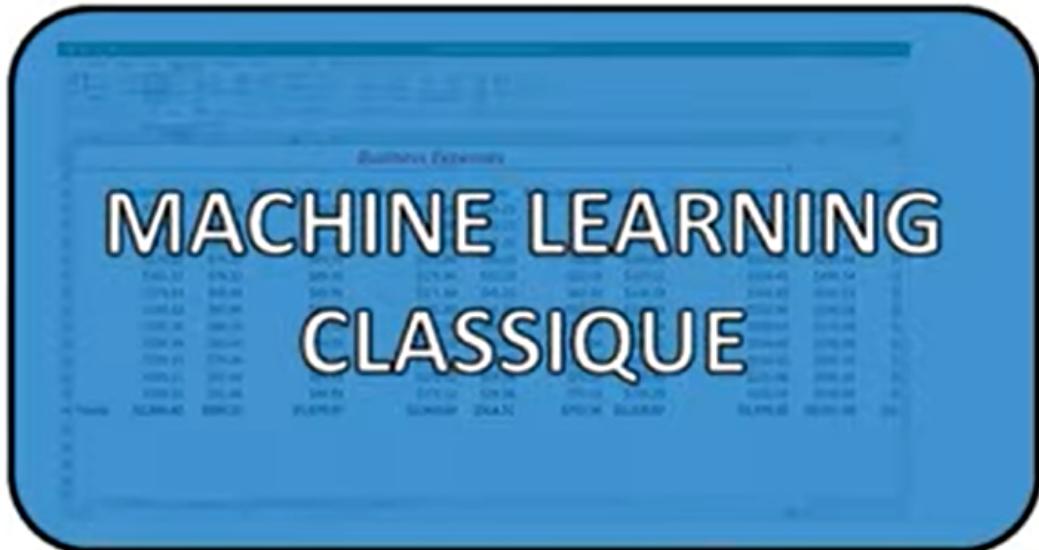
DEEP LEARNING – RÉSEAUX DE NEURONES

# Données Structurées

Alors que les données tabulaires

Business Expenses										
Expense Type	Supplier	Amount	Category	Quantity	Subtotal	Unit Price	Comments	Responsible Person	Location	Notes
Equipment	Supplier A	\$100.00	Category 1	1	\$100.00	\$100.00		Person X	Office	Item 1
Supplies	Supplier B	\$200.00	Category 2	1	\$200.00	\$200.00		Person Y	Office	Item 2
Business Services	Supplier C	\$300.00	Category 3	1	\$300.00	\$300.00		Person Z	Office	Item 3
Cell Phone Service	Supplier D	\$400.00	Category 4	1	\$400.00	\$400.00		Person A	Office	Item 4
Utilities	Supplier E	\$500.00	Category 5	1	\$500.00	\$500.00		Person B	Office	Item 5
Subscription	Supplier F	\$600.00	Category 6	1	\$600.00	\$600.00		Person C	Office	Item 6
Software	Supplier G	\$700.00	Category 7	1	\$700.00	\$700.00		Person D	Office	Item 7
Computer Hardware	Supplier H	\$800.00	Category 8	1	\$800.00	\$800.00		Person E	Office	Item 8
Education	Supplier I	\$900.00	Category 9	1	\$900.00	\$900.00		Person F	Office	Item 9
Meals	Supplier J	\$1000.00	Category 10	1	\$1000.00	\$1000.00		Person G	Office	Item 10
Totals		\$5,000.00			\$5,000.00					

Lorsqu'on a à faire avec les données structurées il est préférable d'utiliser les algorithmes de **Machine Learning**



## Critère #3: Normalité des données

Le troisième critère peut être axé sur la normalité des données pourquoi par ce qu'en **Machine Learning**, il y a deux types de modèles: (Paramétrique et Non-Paramétrique)

### Modèles Paramétriques

Exemples :

- Régression linéaire:  $f(x) = \textcolor{teal}{a}x + \textcolor{blue}{b}$
- Régression logistique
- Naive Bayes
- Réseau de Neurones

**Condition importante !** Les données doivent suivre des **lois de probabilités**

Loi normale :  $X \sim N(\mu, \sigma^2)$

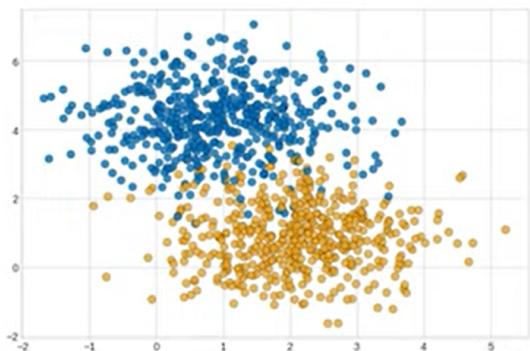
### Modèles Non-Paramétriques

Exemples :

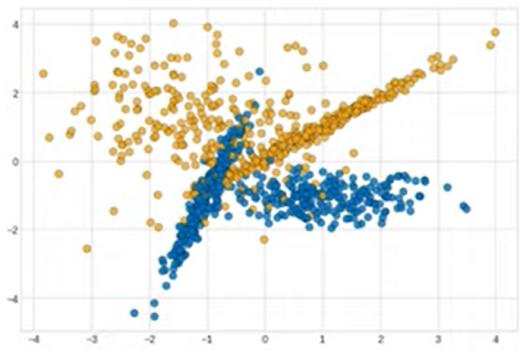
- Support Vector Machines
- Arbres de décision
- Random Forest
- K-Nearest Neighbours

Pourquoi est-ce qu'il est important parce que les modèles paramétriques fonctionnent très bien avec les données qui suivent la loi de la distribution normale pensez à utiliser les modèles paramétriques sinon dans tous les autres cas utilisez les modèles non-paramétriques ou il faut faire du preprocessing pour obtenir les données normées.

## Modèles Paramétriques



## Modèles Non-Paramétriques



## Critère #4: Variables Quantitatives / Qualitatives

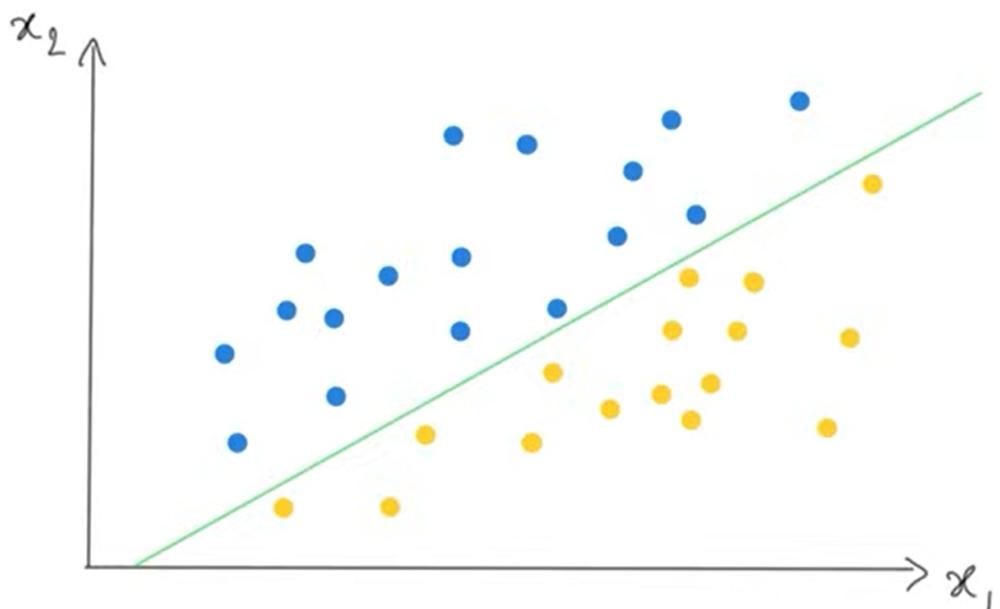
Un quatrième critère à prendre en compte est la quantité des variables quantitatives et qualitatives dont vous disposez.

### Variables Quantitatives

Sachez qu'il existe certains types de modèles tel que: les arbres de décision et tout ce qui en découle qui ne sont pas très efficace avec les données qui contiennent beaucoup de variables quantitatives et surtout lorsque vous observez des relations linéaires entre ces variables quantitatives.

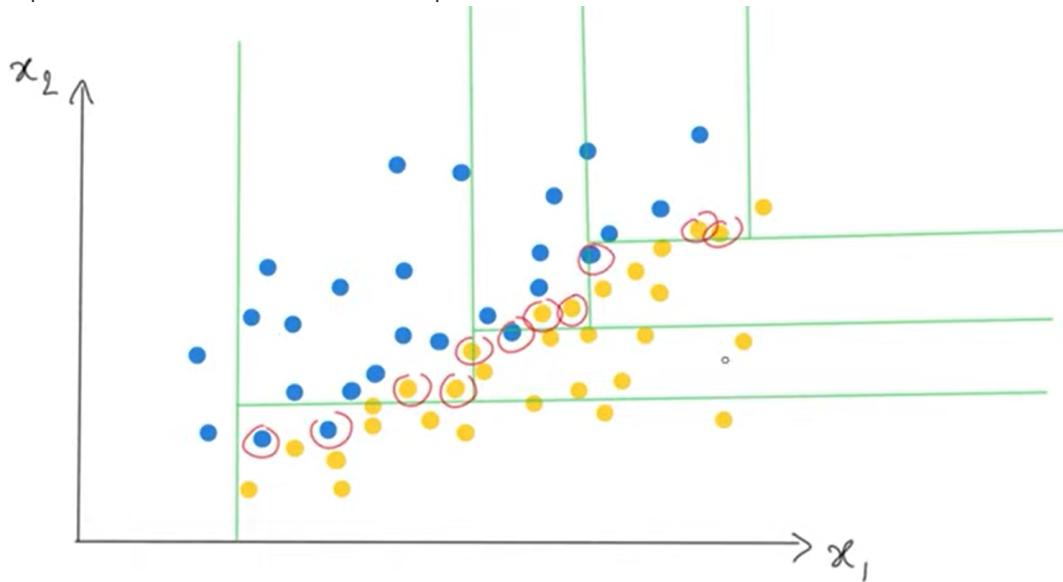
Pour illustrer cela:

- **Exemple 1:**  
on a un dataset qui contient deux variables quantitatives qui ont une relation parfaitement linéaire.



Seulement quand on utilise les arbres de décisions ils ne peuvent pas tracer ce genre de droite.

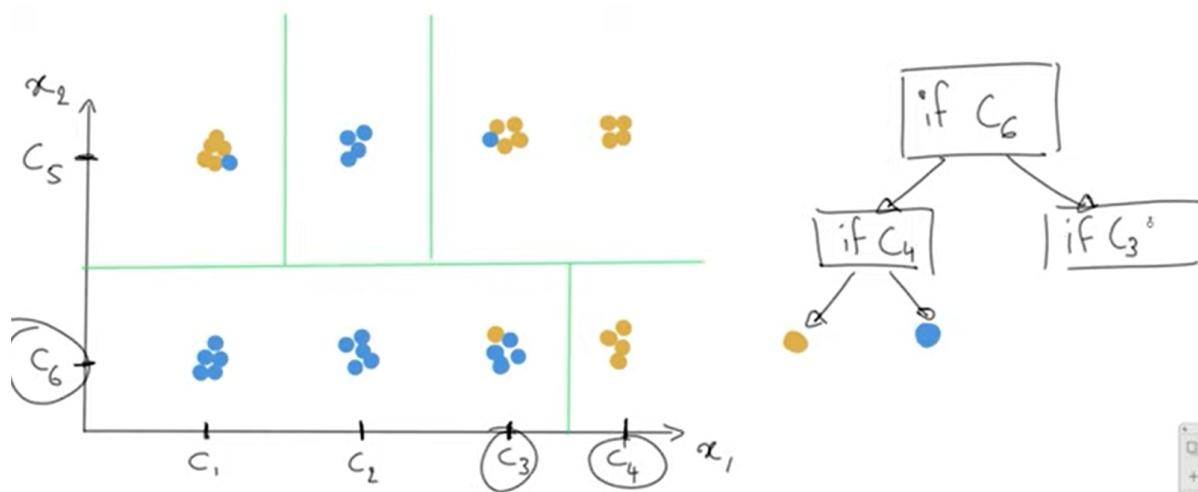
- A la place ils les decoupent les axes de notre plan de manière orthogonale de façon à former des escaliers et c'est justement ça le problème. Ces escaliers ont un fort risque d'avoir de l'**Over-fitting** pourquoi parce que si on ajoute plus de points dans nos données c'est à dire d'autre points venant du testset beaucoup seront mal classés à cause de la forme d'escalier.



voilà pour quand on a beaucoup de variable quantitative avec des relation lineaires on risque fort d'avoir de l'**Over-fitting**

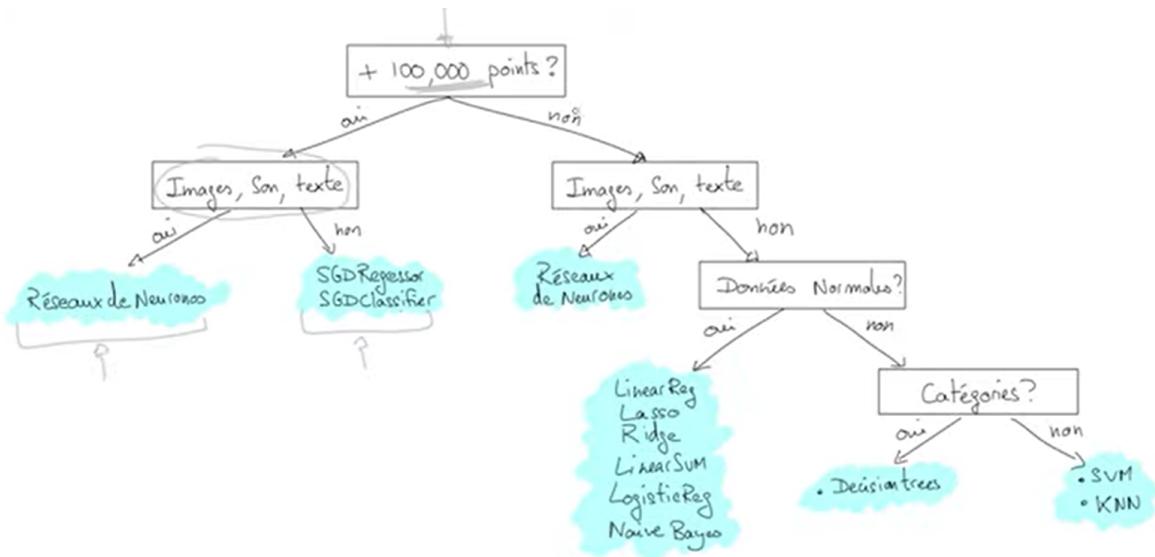
## Variables Qualitative

A l'inverse lorsqu'on a beaucoup de variables qualitative avec beaucoup de classe, il ne faut pas hésiter à utiliser les arbres de décisions dans ce cas. C'est pour ça qu'ils ont été conçus et c'est pour qu'ils sont performants.



## Résumé: MIND MAP

---



## En Pratique

Dans la vraie vie, les données partent dans tous les sens. Parfois on observe les tendances linéaires, non-linéaires, ... .

Alors il faut tester tous les modèles possibles et on choisit celui qui aura la meilleure performance.

```
[19] preprocessor = make_pipeline(PolynomialFeatures(2, include_bias=False), SelectKBest(f_classif, k=10))
```

```
RandomForest = make_pipeline(preprocessor, RandomForestClassifier(random_state=0))
AdaBoost = make_pipeline(preprocessor, AdaBoostClassifier(random_state=0))
SVM = make_pipeline(preprocessor, StandardScaler(), SVC(random_state=0))
KNN = make_pipeline(preprocessor, StandardScaler(), KNeighborsClassifier())
```

```
[21] dict_of_models = {'RandomForest': RandomForest,
                      'AdaBoost' : AdaBoost,
                      'SVM': SVM,
                      'KNN': KNN
                     }
```

```
[24] for name, model in dict_of_models.items():

    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(name, ': f1 score', f1_score(y_test, y_pred) )
```

```
RandomForest : f1 score 0.39999999999999997
AdaBoost : f1 score 0.5185185185185185
SVM : f1 score 0.48000000000000001
KNN : f1 score 0.5161290322580646
```

## Conclusion

Voilà donc quatre critères de sélection du modèle lorsqu'on a des projets de **Machine Learning** et du **Deep Learning** à réaliser.

Bien qu'il en existe d'autres. Alors n'hésitez pas à aller plus loin et chercher à vous armez pour construire un background solide et incontestable.

