


APPRENTISSAGE SUPERVISÉ: MODÈLES ALGORITHMIQUES



I know kung fu

MODÈLES ALGORITHMIQUES

Les K plus proches voisins

LES K PLUS PROCHES VOISINS

(K-nearest neighbors)

La valeur d'un
(nouveau) point
dépend de la
valeur des K
points qui lui
ressemblent le
plus.

K = constante
choisie.

Exemple: K=1,
K=3, K=20, ...

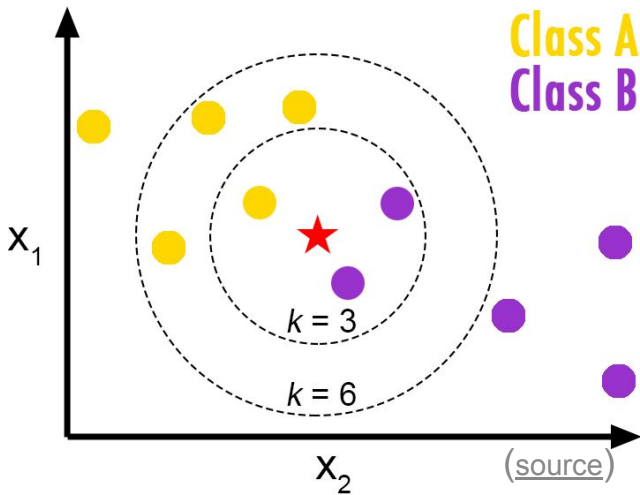
LES K PLUS PROCHES VOISINS

(K-nearest neighbors)

La valeur d'un
(nouveau) point
dépend de la
valeur des K
points qui lui
ressemblent le
plus.

K = constante
choisie.

Exemple: K=1,
K=3, K=20, ...



MODELISATION/APPRENTISSAGE

Presque rien à faire! Seulement choisir:

- un **nombre de voisins K**
- et une **distance**

CLASSIFICATION

Pour chaque élément test, on regarde la valeur des points voisins de l'ensemble d'apprentissage et on procède à un **vote majoritaire**.

INPUT : Ensemble d'apprentissage (X_{train} , Y_{train}), ensemble de test (X_{test} , Y_{test}), distance **D**, nombre de voisins K

for $x \in X_{\text{test}}$:

for $x' \in X_{\text{train}}$:

 Calculer distance **D**(x , x')

 Trouver K voisins x' les + proches au sens de **D**

Classe(x) := classe + fréquente des K voisins

REGRESSION

Pour chaque élément test, on regarde la valeur des points voisins de l'ensemble d'apprentissage et on procède à une **moyenne (*)**.

INPUT : Ensemble d'apprentissage (X_{train} , Y_{train}), ensemble de test (X_{test} , Y_{test}), distance **D**, nombre de voisins K

for $x \in X_{\text{test}}$:

for $x' \in X_{\text{train}}$:

 Calculer distance **D**(x , x')

 Trouver K voisins x' les + proches au sens de **D**

Valeur(x) := moyenne des K voisins

COMPLEXITE

for $\mathbf{x} \in X_{\text{test}}$ do

 for $\mathbf{x}' \in X_{\text{train}}$ do

 Calculer distance $D(\mathbf{x}, \mathbf{x}')$

→ $O(|X_{\text{test}}| \times |X_{\text{train}}| \times C_D)$

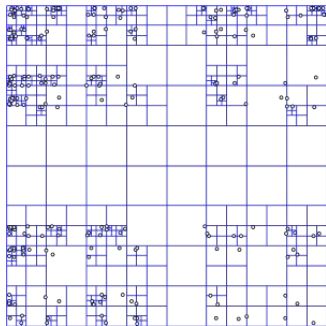
C_D = Complexité d'un calcul de distance

Imaginez un corpus de 1M images de chats/chien, et vous devez classifier 10K images. Calculer la distance entre 2 images est compliqué, disons que ça prend 1ms.
 $1M \times 10k \times 1ms \approx 116$ jours

ACCÉLÉRATION

Il existe plusieurs techniques, déjà.....
implémentées pour vous. Ça peut être très
compliqué: ne le réinventez pas!

- Sous-échantillonnage (on utilise un sous-ensemble X_{train} de plus petite taille, quitte à réduire la qualité)
- pré-Clustering de X_{train}
- K-D trees (**Approximate Nearest Neighbors**) →
- etc .. (wikipedia)



PROBLEMES POTENTIELS

- Disproportion des classes
- Mauvaise fonction de distance

BLIBS, BLOBS, BLABS ET BLUBS

Blib



Blob



Blab



Blub



BLIBS, BLOBS, BLABS ET BLUBS

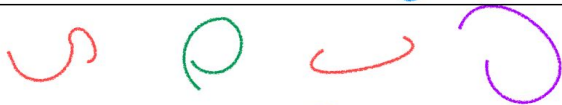
Blib



Blob



Blab



Blub



?



?



?



?

BLIBS, BLOBS, BLABS ET BLUBS

K=1

Blib



Blob



Blab



Blub



?



?



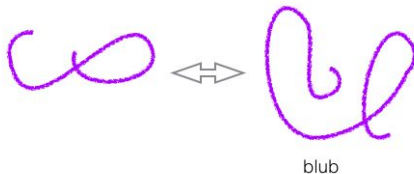
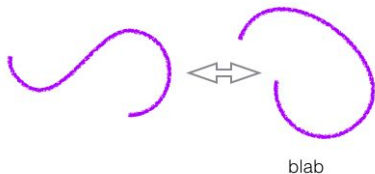
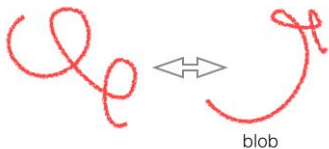
?



?

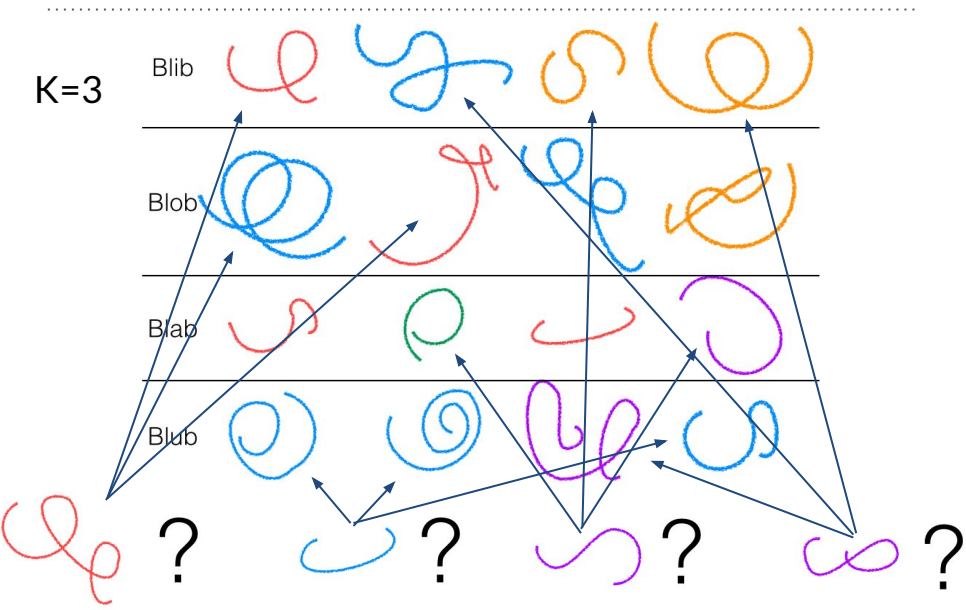
BLIBS, BLOBS, BLABS ET BLUBS

Algorithme du plus proche voisin (K=1):



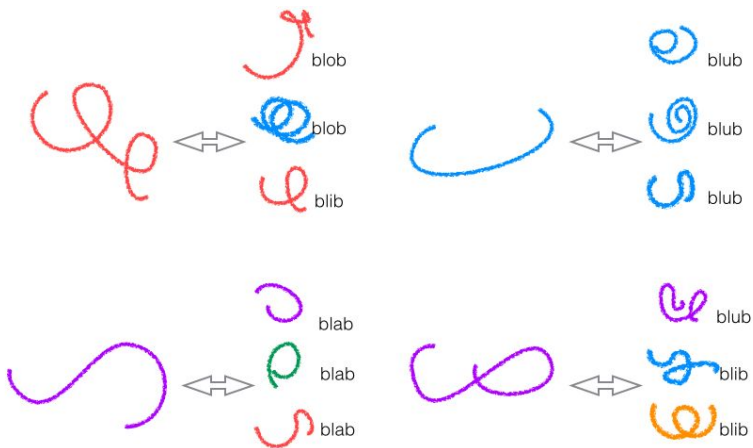
Résultat : 4/4 = 100%

BLIBS, BLOBS, BLABS ET BLUBS



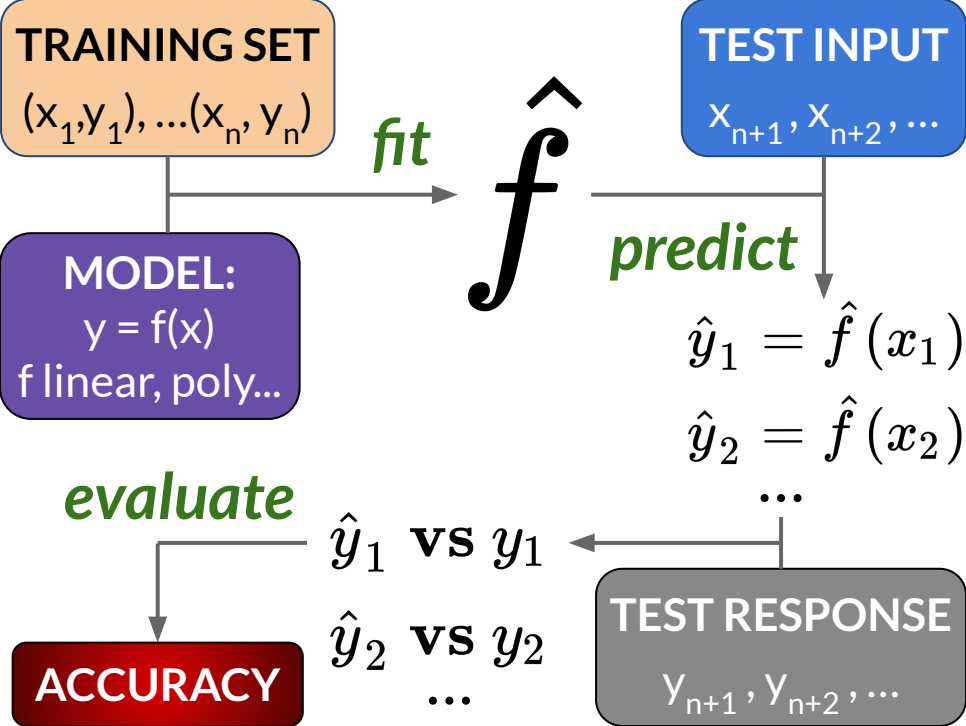
BLIBS, BLOBS, BLABS ET BLUBS

Algorithme des 3 plus proches voisins (K=3):.....



Résultat : $3/4 = 75\%$

COMMENT OPTIMISER UN PARAMÈTRE ?



OPTIMISER UN PARAMETRE

- J'ai **bien** fait attention de séparer mon dataset en ensembles de “**train**” + “**test**”
- L'ensemble de “**train**” est la donnée utilisée par mon algo pour s'**entraîner**
- L'ensemble de “**test**”, qui n'a **pas** été donné à l'algorithme pendant son entraînement, permet d'**évaluer** mon algorithme

Ce qu'il ne faut **PAS** faire

Comment déterminer quel est le meilleur K?

Je vais **évaluer** (donc avec l'ensemble de **test**) la performance de mon algo avec différents K, et garder le K qui donne le meilleur résultat.

Ce qu'il ne faut **PAS** faire

Comment déterminer quel est le meilleur K?

Je vais **évaluer** (donc avec l'ensemble de **test**) la performance de mon algo avec différents K, et garder le K qui donne le meilleur résultat.

NON!!!

Comment **évaluer** mon algo, une fois le K choisi?
Mon ensemble de test n'est plus "inconnu", l'algo (*) a été choisi en fonction de lui !

→ Ne **jamais** utiliser un ensemble de **test** à part dans l'évaluation finale, après que l'algorithme ait été complètement choisi, + paramètres, etc!

Ce qu'il ne faut **PAS** faire

Comment déterminer quel est le meilleur K?

On reprend. Je vais **évaluer** la performance de mon algo avec divers K en utilisant l'ensemble **train** pour l'évaluation (et donc le choix de K)

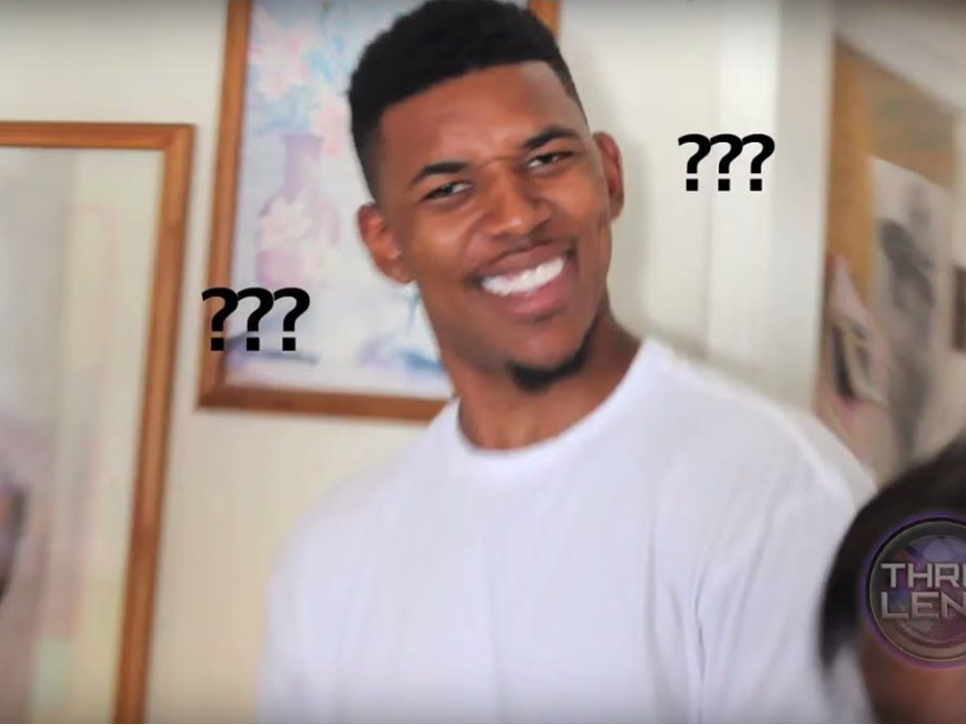
Ce qu'il ne faut **PAS** faire

Comment déterminer quel est le meilleur K?

On reprend. Je vais **évaluer** la performance de mon algo avec divers K en utilisant l'ensemble **train** pour l'évaluation (et donc le choix de K)

NON!!!

Si je fais comme ça je suis en plein dans l'**over-fitting** : j'essaie uniquement d'obtenir la meilleure performance sur mon ensemble d'entraînement. L'algo (ou ses paramètres) que je choisirai sera sûrement over-fitté, donc **mauvais** en pratique (par exemple sur mon test).



???

???



Ce qu'il faut faire

Comment déterminer quel est le meilleur K?

J'ai trouvé!

Je vais séparer mon dataset en 3 parties disjointes: (**train**, **validation**, **test**).

Pour chaque valeur possible de K, j'entraîne mon algo avec les data de **train**, et j'évalue sa performance avec les data de **validation**.

→ Je trouve ainsi le meilleur K.

Et j'ai bien préservé mon mon ensemble test, qui n'a pas joué de rôle dans le choix de l'algo.

Ce qu'il faut faire

Comment déterminer quel est le meilleur K?

J'ai trouvé!

Je vais séparer mon dataset en 3 parties disjointes: (**train**, **validation**, **test**).

Pour chaque valeur possible de K, j'entraîne mon algo avec les data de **train**, et j'évalue sa performance avec les data de **validation**.

→ Je trouve ainsi le meilleur K.

Et j'ai bien préservé mon mon ensemble test, qui n'a pas joué de rôle dans le choix de l'algo.

OUI! [[wikipedia](https://fr.wikipedia.org/wiki/Validation_crois%C3%A9e)]

Ce qu'il faut faire

$K = 1 \rightarrow$ taux de classification correct = 0.873

$K = 3 \rightarrow$ taux de classification correct = 0.862

$K = 5 \rightarrow$ taux de classification correct = 0.855

$K = 7 \rightarrow$ taux de classification correct = **0.884**

$K = 9 \rightarrow$ taux de classification correct = 0.861

\rightarrow je choisis **$K = 7$**

Pour accélérer:

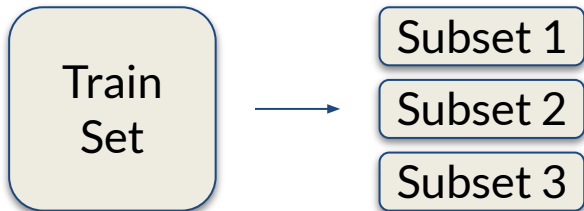
- Sous-échantillonnage (ne pas essayer tous les K)
 - Exponentiel: ($K=1, 2, 3, 5, 8, 13, 21, 34, \dots$)
- Compromis coût/qualité de l'évaluation

LA VALIDATION CROISEE

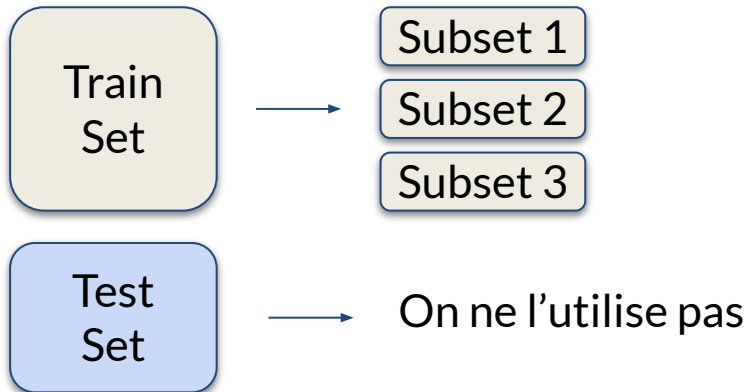
Une manière d'évaluer la performance de mon algorithme (aka “valider”) de manière **plus robuste**.

Important notamment quand on fait des **choix** en fonction de cette évaluation (e.g. choix du paramètre K , mais aussi de techniques ou procédés, e.g. “moyenne ou mediane ?”

LA VALIDATION CROISEE



LA VALIDATION CROISEE



LA VALIDATION CROISEE

Subset 1

Subset 2

Subset 3

LA VALIDATION CROISEE

Subset 1 = train } Qu'on fusionne en un seul
Subset 2 = train } ensemble d'entraînement
Subset 3 = validation

Accuracy
= 0.931

LA VALIDATION CROISEE

Subset 1 = train

Subset 2 = train

Subset 3 = validation

Accuracy
= 0.931

Subset 1 = train

Subset 2 = validation

Subset 3 = train

Accuracy
= 0.917

LA VALIDATION CROISEE

Subset 1 = train

Subset 2 = train

Subset 3 = validation

Accuracy
= 0.931

Subset 1 = validation

Subset 2 = train

Subset 3 = train

Accuracy
= 0.948

Subset 1 = train

Subset 2 = validation

Subset 3 = train

Accuracy
= 0.917

LA VALIDATION CROISEE

Subset 1 = train

Subset 2 = train

Subset 3 = validation

Accuracy
= 0.931

Subset 1 = validation

Subset 2 = train

Subset 3 = train

Accuracy
= 0.948

Subset 1 = train

Subset 2 = validation

Subset 3 = train

Accuracy
= 0.917

→ **Estimated Accuracy**
= avg(0.931, 0.917, 0.948)
= **0.932**

VALIDATION CROISEE, SUITE

Pourquoi utiliser la **validation** ?

Lors de l'apprentissage, on ne peut **pas** utiliser l'ensemble de **test**, jamais, ce serait "tricher".

Car si on utilise l'ensemble de test pour apprendre, le modèle sera forcément bon en test : **sur-apprentissage**, évaluation trompeuse (on ne connaît pas la vraie perf), et mauvais choix!

Il faut donc choisir les paramètres en se basant **uniquement** sur l'ensemble d'apprentissage.

La **validation** est un moyen d'**évaluer** durant l'**apprentissage**.

VALIDATION CROISEE, SUITE


Pourquoi utiliser la **validation croisée** ?

La **validation croisée** est un moyen robuste et efficace de valider. Elle permet notamment d'utiliser la totalité de l'ensemble X_{train} pour entraîner, mais aussi pour évaluer.

C'est incontournable! On l'utilisera pour **tout choix**: de paramètre, mais aussi pour choisir un algorithme, une technique, etc :

- Je choisis A ou B ?
- J'évalue A avec la validation croisée, puis B.
- Je choisis celui qui marche le mieux.

VALIDATION CROISEE, SUITE

 c'est tellement bien...
... qu'on oublie trop facilement de
tester le résultat final!

EN PYTHON

```
from sklearn.cluster import \
    KNeighborsClassification, \
    KNeighborsRegression

model = KNeighborsClassification(
    n_neighbors = 5)

model.fit(Xtrain, Ytrain)

predictions = model.predict(Xtest)
```


MODÈLES ALGORITHMIQUES

Arbres de décision

RAPPEL: VARIABLES DISCRETES

On dit qu'une variable est **discrète** lorsqu'elle prend un nombre dénombrable de valeurs.
(Dénombrable = fini, en pratique).

Exemples :

- nombre de pièces d'un appartement: 1,2,3,4...
- nom de l'auteur d'un livre: Hugo, Balzac, ...
- match à domicile (oui/non)
- 1-Hot encoding
- ...

RAPPEL: VARIABLES CONTINUES

On dit qu'une variable est **continue** lorsqu'elle prend un nombre non dénombrable de valeurs.

Exemples:

- prix d'un appartement: $[0, +\infty[$
- valeur TF/IDF
- etc

Remarque 1 : on peut **discrétiser** une variable continue. Exemple du prix d'appart :
classe 1 = $< 500\text{€}/\text{mois}$, classe 2 = $[500, 700[, \dots$

Remarque 2 : À l'inverse, si une var discrète a trop de valeurs, on peut la considérer continue.


INTUITION

Principe : réfléchir dimension par dimension, probablement le plus proche d'une décision humaine.

Les arbres s'utilisent dans des problèmes de régression et de classification.

Ils acceptent les données de types différents : qualitatives ou quantitatives, discrètes ou continues.

APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
F	F	F	V	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P
F	F	V	F	P
V	F	V	V	P

Colonnes

Dom:


Match à Domicile ?

Sam: Match a lieu le Samedi?

 : Est-ce qu'il pleut?

P.G.: Le match précédent a-t-il été gagné?

APPRENTISSAGE : EXEMPLE SIMPLE


Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
F	F	F	V	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P
F	F	V	F	P
V	F	V	V	P

Samedi ?

4V

4F


APPRENTISSAGE : EXEMPLE SIMPLE

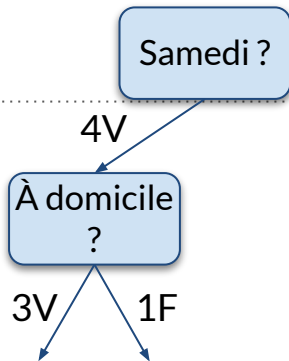
Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P

Samedi ?


4V

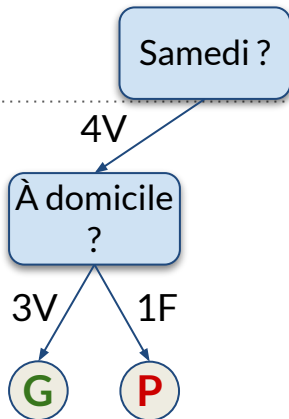
APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P




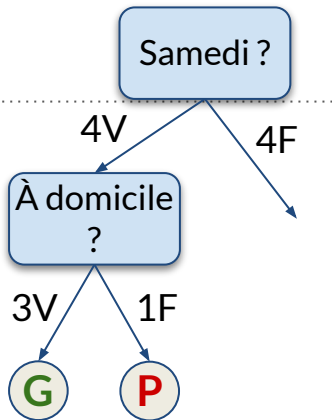
APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P




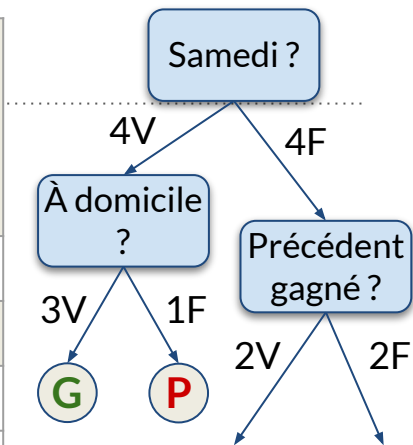
APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
F	F	F	V	G
F	F	V	F	P
V	F	V	V	P




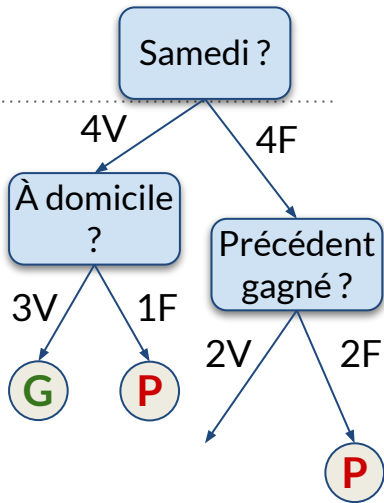
APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
F	F	F	V	G
F	F	V	F	P
V	F	V	V	P




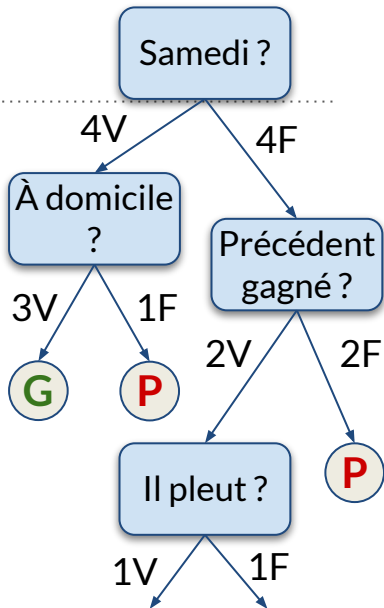
APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
F	F	V	F	P




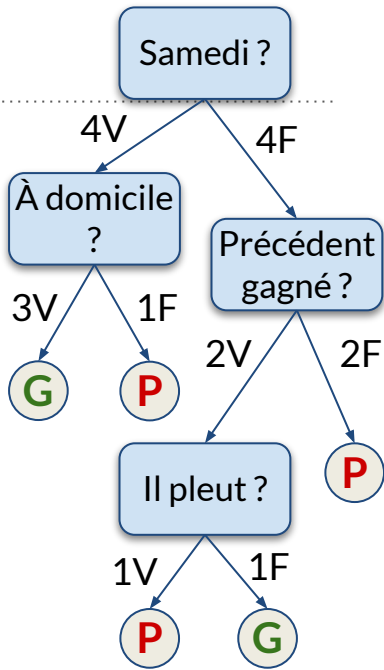
APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
F	F	F	V	G
V	F	V	V	P




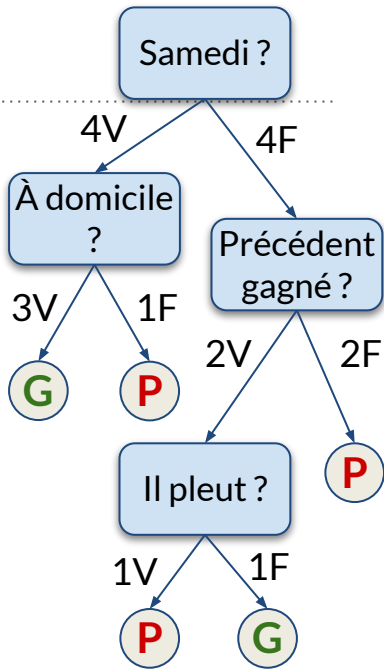
APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
F	F	F	V	G
V	F	V	V	P



APPRENTISSAGE : EXEMPLE SIMPLE

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
F	F	F	V	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P
F	F	V	F	P
V	F	V	V	P

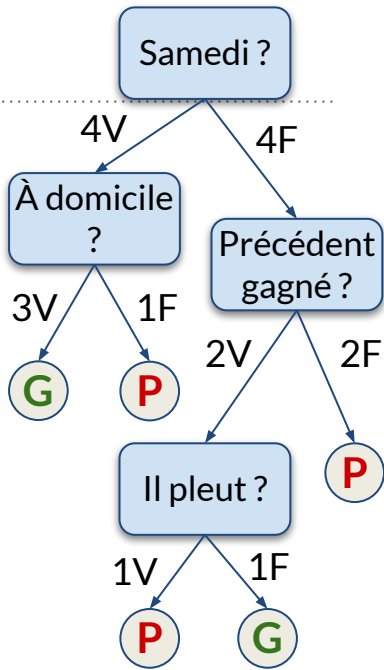


APPRENTISSAGE : EXEMPLE SIMPLE

Le prochain match a les caractéristiques suivantes:

- À domicile
- Dimanche
- Il ne pleut pas
- Match précédent perdu

Prédiction : le match sera ... ?



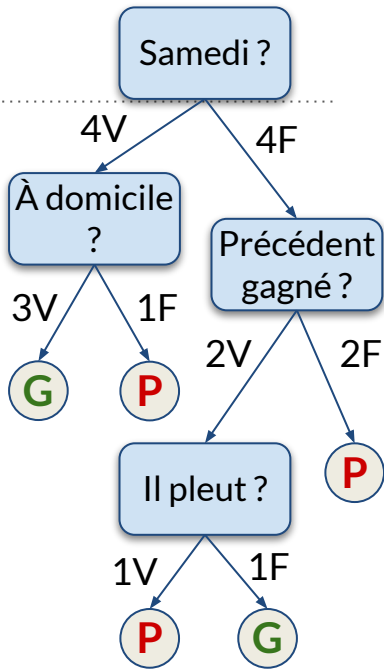
APPRENTISSAGE : EXEMPLE SIMPLE

Le prochain match a les caractéristiques suivantes:

- À domicile
- Dimanche
- Il ne pleut pas
- Match précédent perdu

Prédiction : le match sera

Perdu!

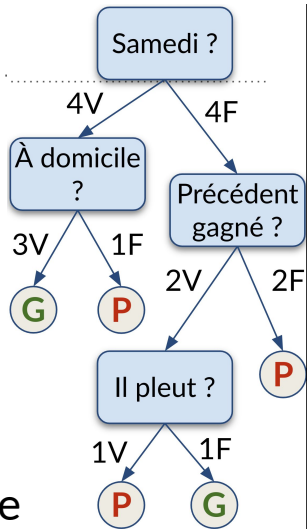


APPRENTISSAGE : EXTENSIONS

Cet exemple était très petit: avec juste 4 décisions on a à des “feuilles” parfaites.


En pratique on a des milliers de lignes, mais on ne veut **pas** des milliers de noeuds (pour éviter le sur-apprentissage)

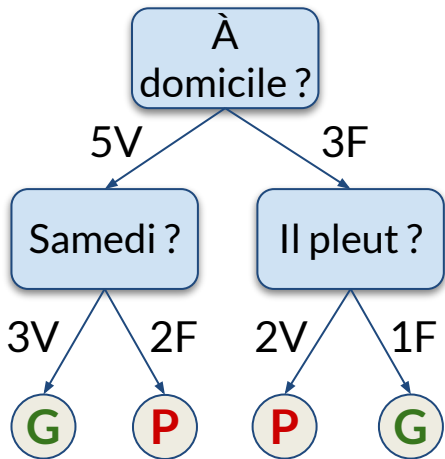
On s'arrête donc dès qu'une feuille est suffisamment *homogène* (ex: 153 “gagnés”, 8 “perdus”).



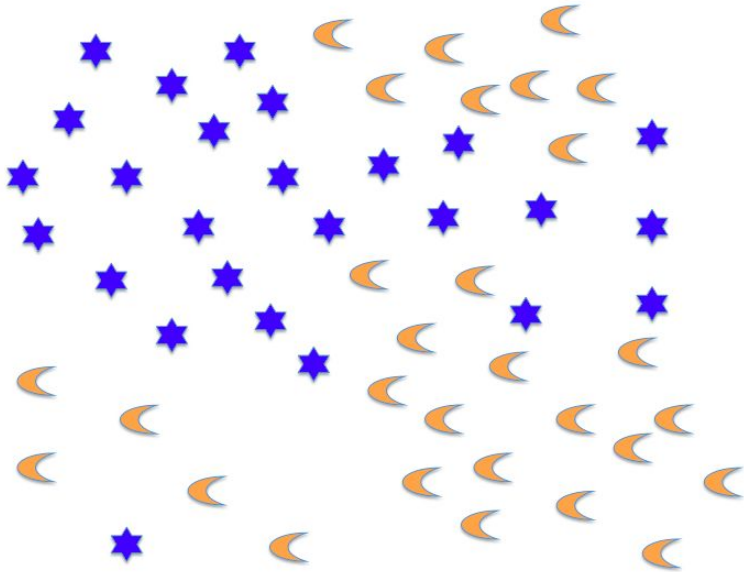
APPRENTISSAGE : EXTENSIONS

Au lieu de prendre des décisions **aléatoires** (e.g. colonne?); on cherche les décisions (colonnes) qui au final donnent l'arbre le plus **compact**.

Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
F	F	F	V	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P
F	F	V	F	P
V	F	V	V	P
V	F	V	F	P

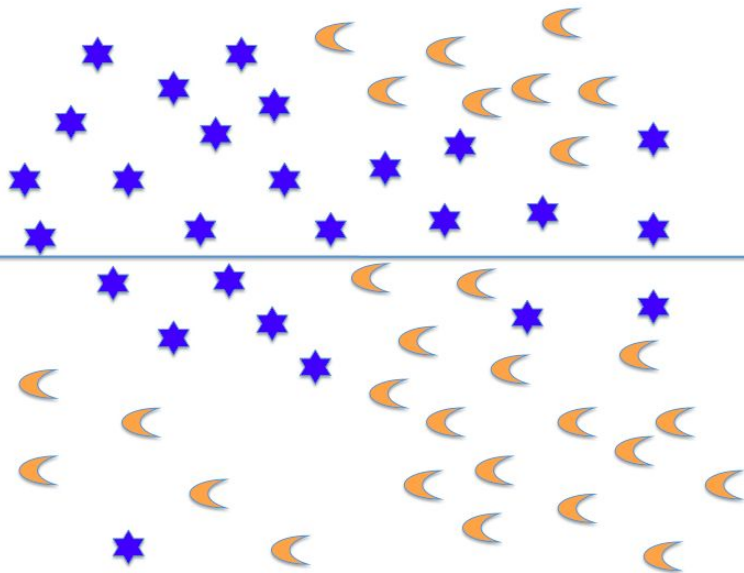


APPRENTISSAGE : VAR. CONTINUES

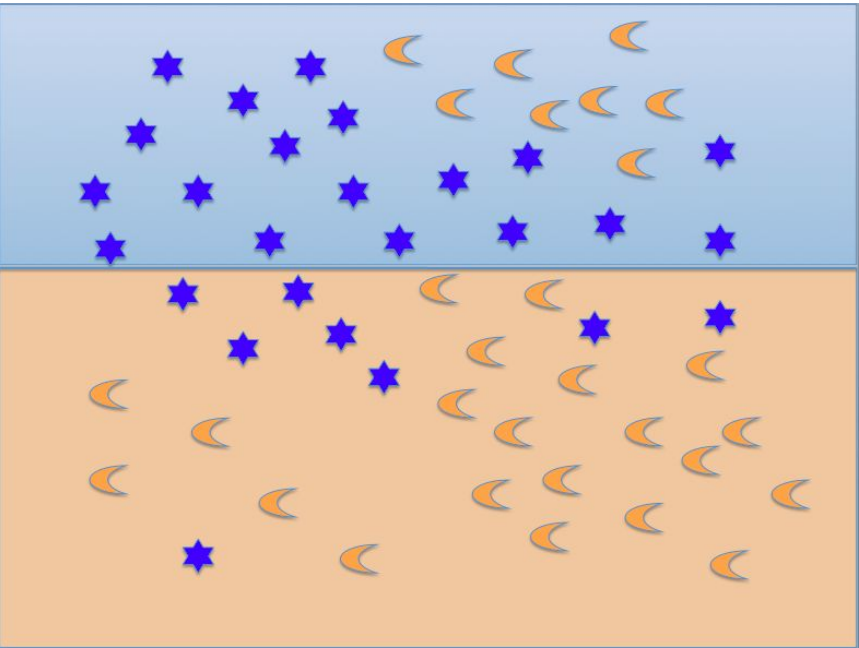


.....

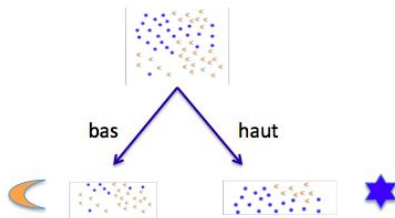
APPRENTISSAGE : VAR. CONTINUES



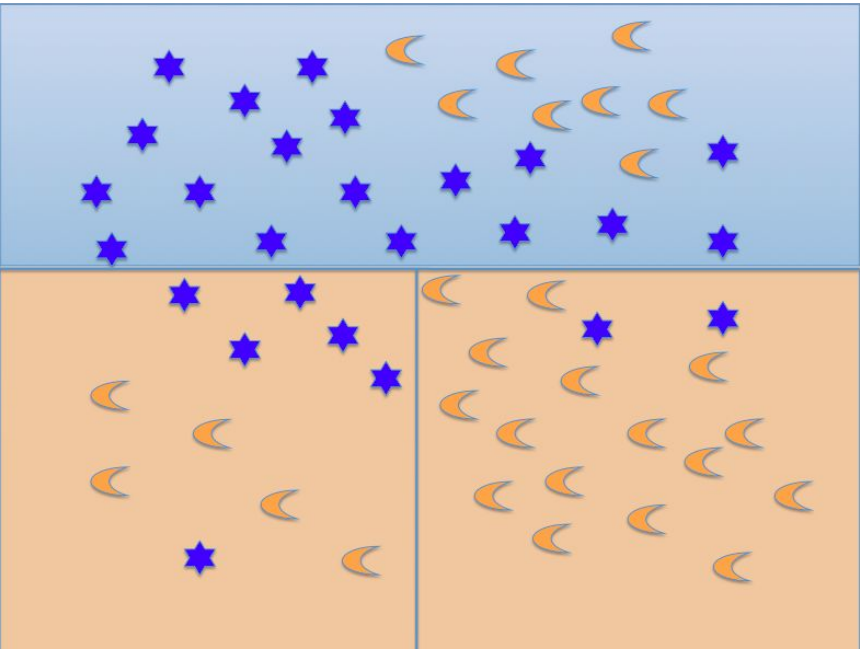
APPRENTISSAGE : VAR. CONTINUES



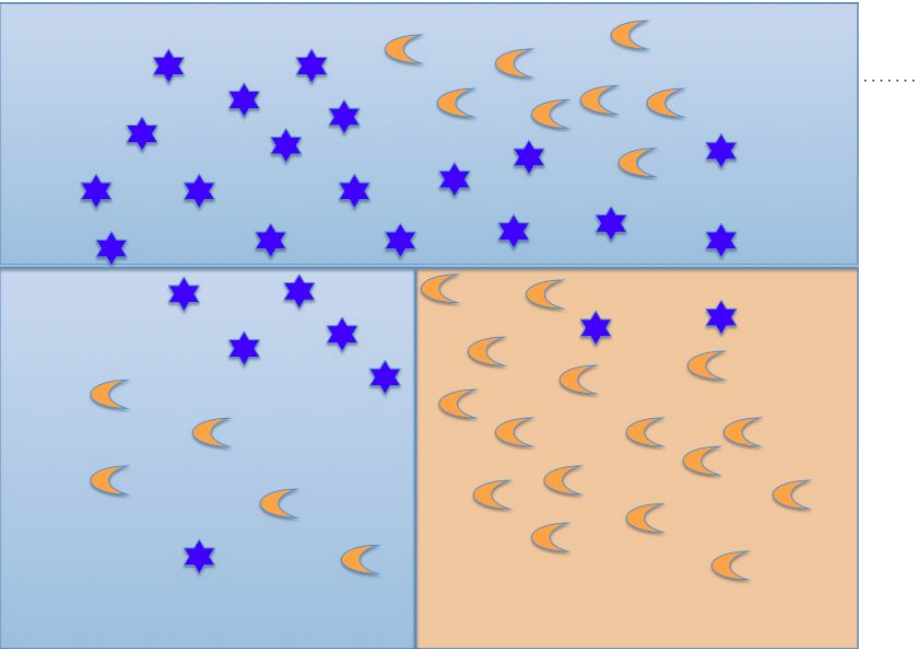
APPRENTISSAGE : VAR. CONTINUES



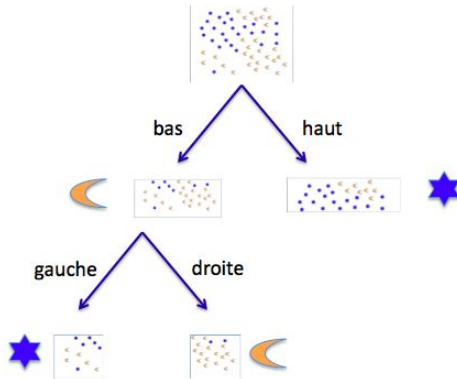
APPRENTISSAGE : VAR. CONTINUES



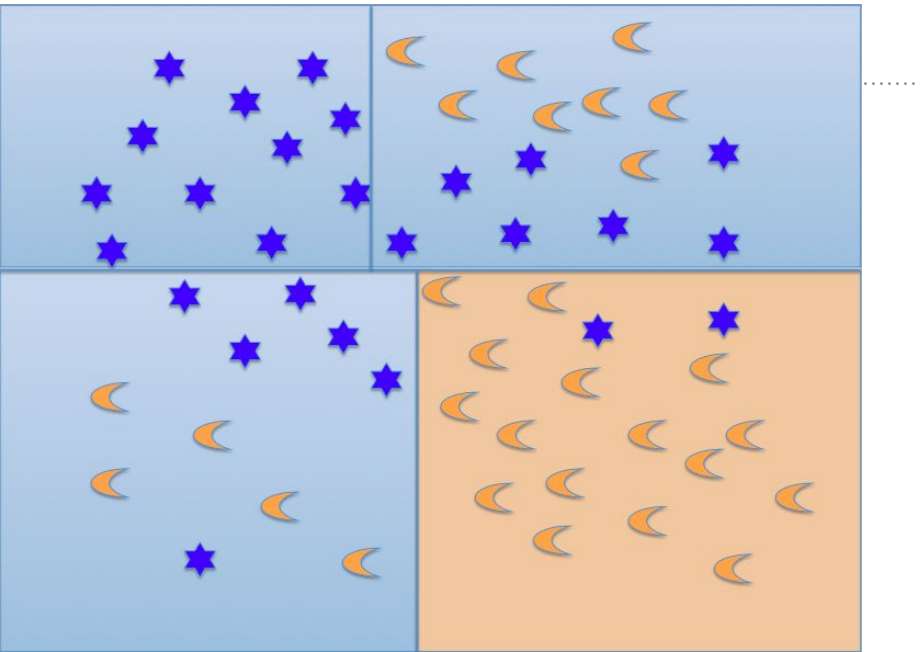
APPRENTISSAGE : VAR. CONTINUES



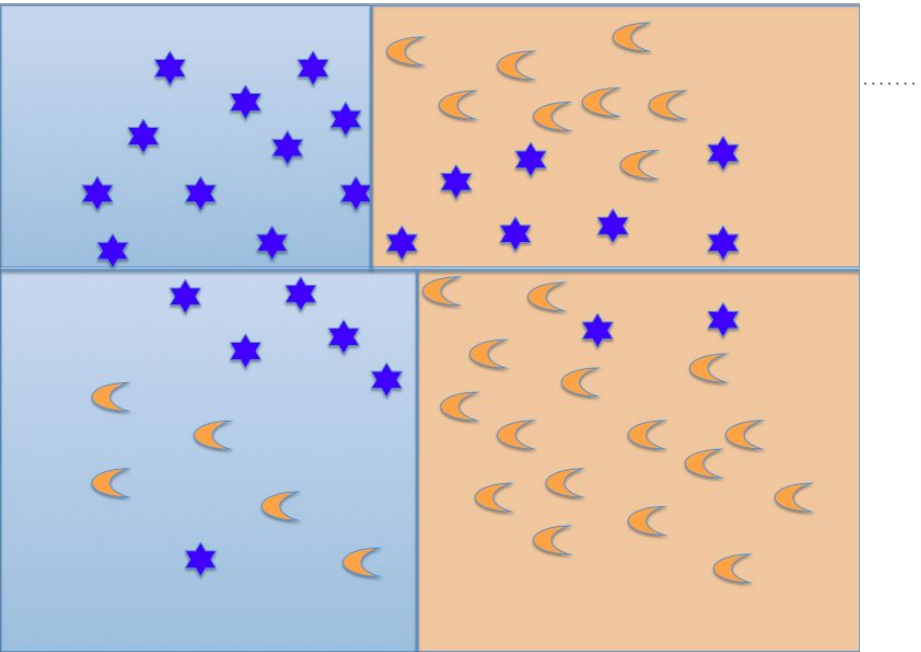
APPRENTISSAGE : VAR. CONTINUES



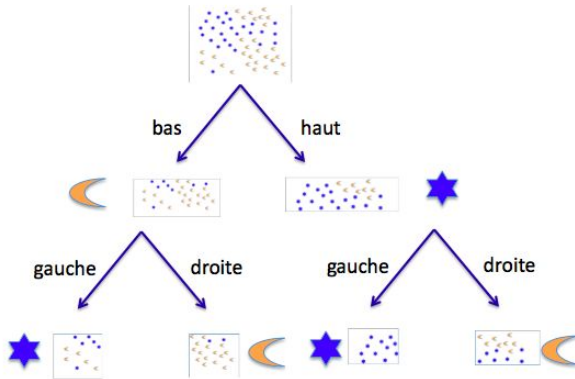
APPRENTISSAGE : VAR. CONTINUES



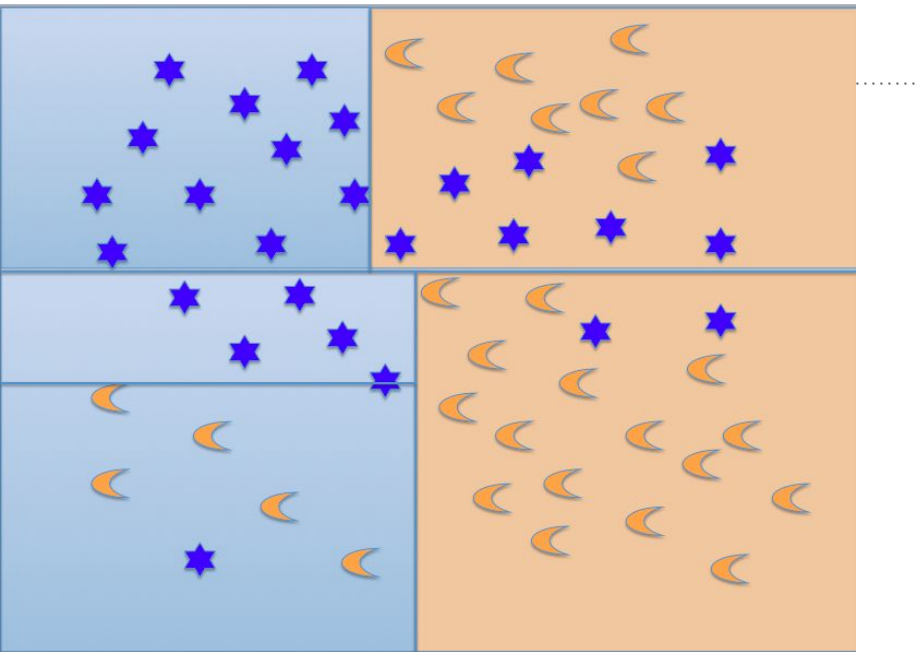
APPRENTISSAGE : VAR. CONTINUES



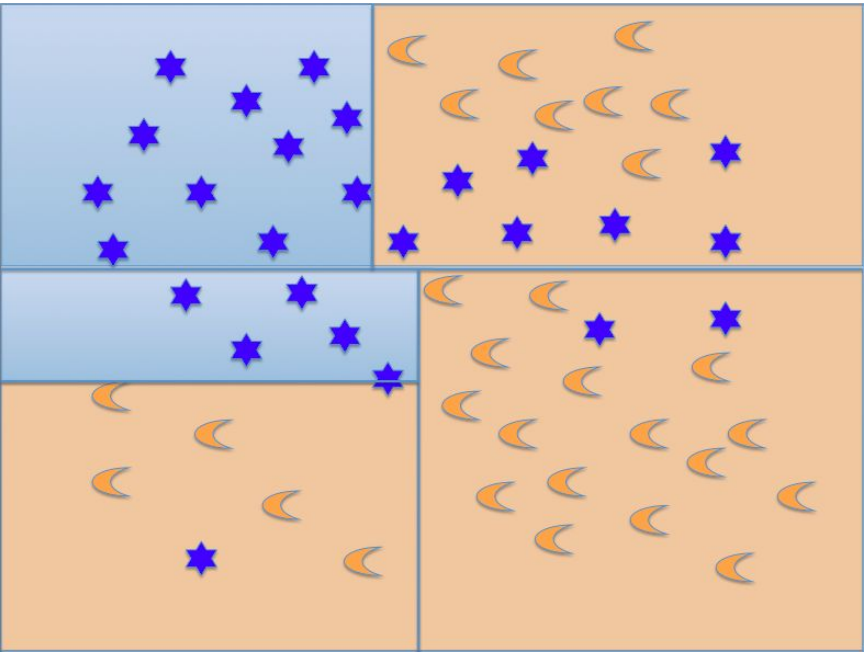
APPRENTISSAGE : VAR. CONTINUES



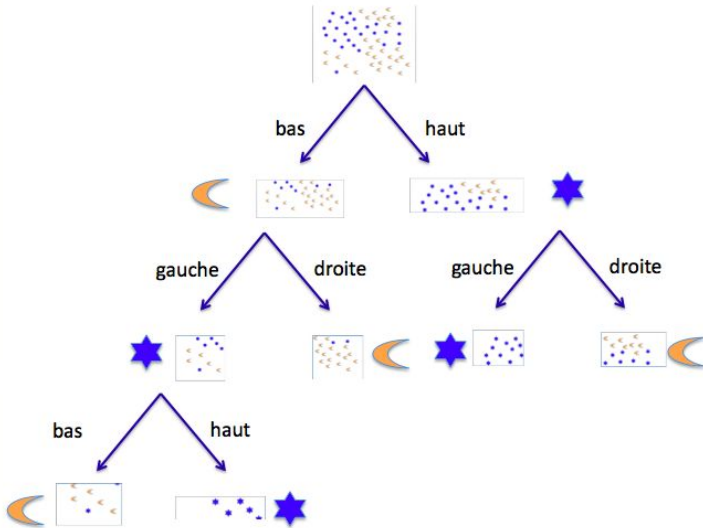
APPRENTISSAGE : VAR. CONTINUES



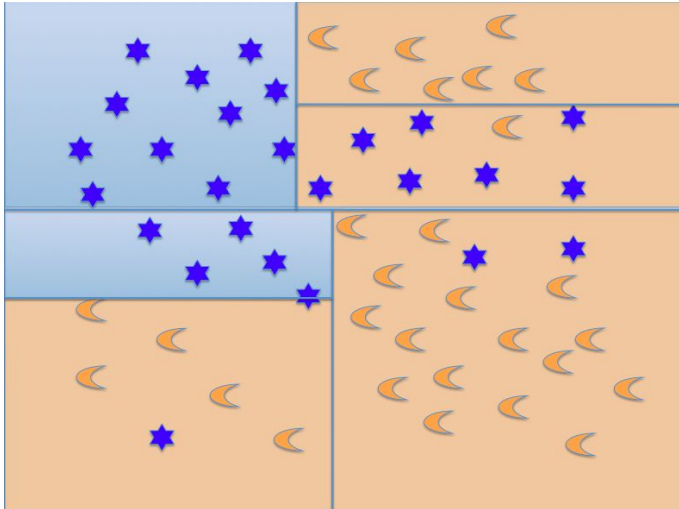
APPRENTISSAGE : VAR. CONTINUES



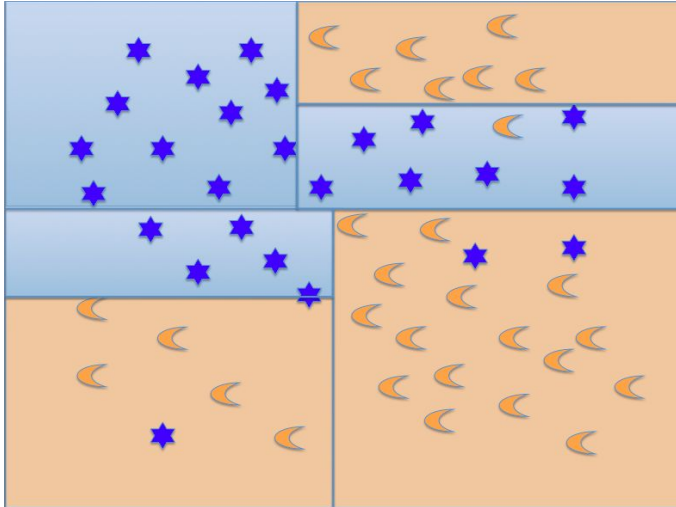
APPRENTISSAGE : VAR. CONTINUES



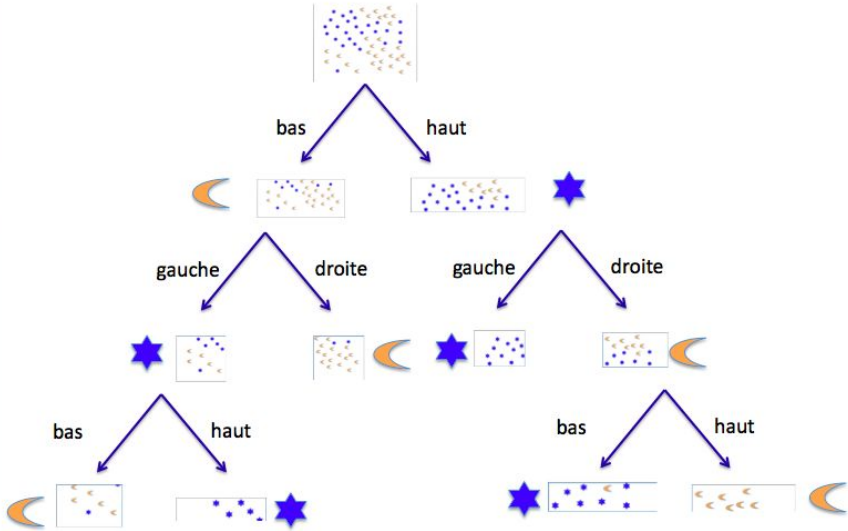
APPRENTISSAGE : VAR. CONTINUES



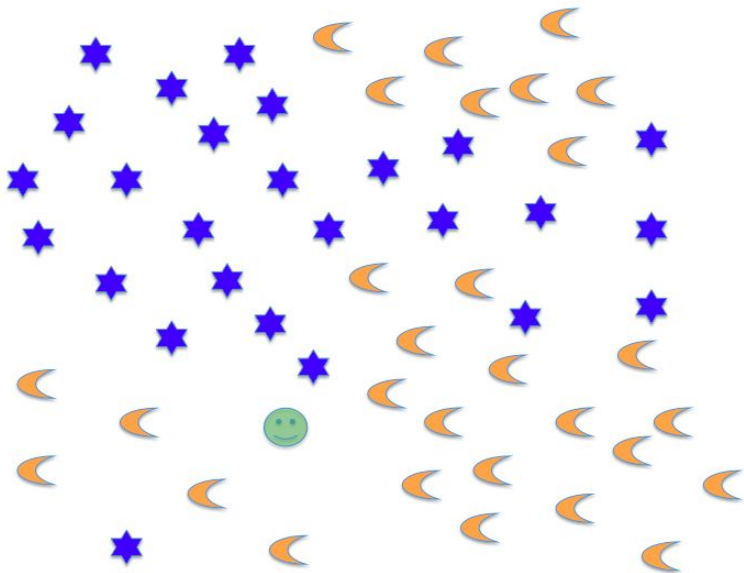
APPRENTISSAGE : VAR. CONTINUES



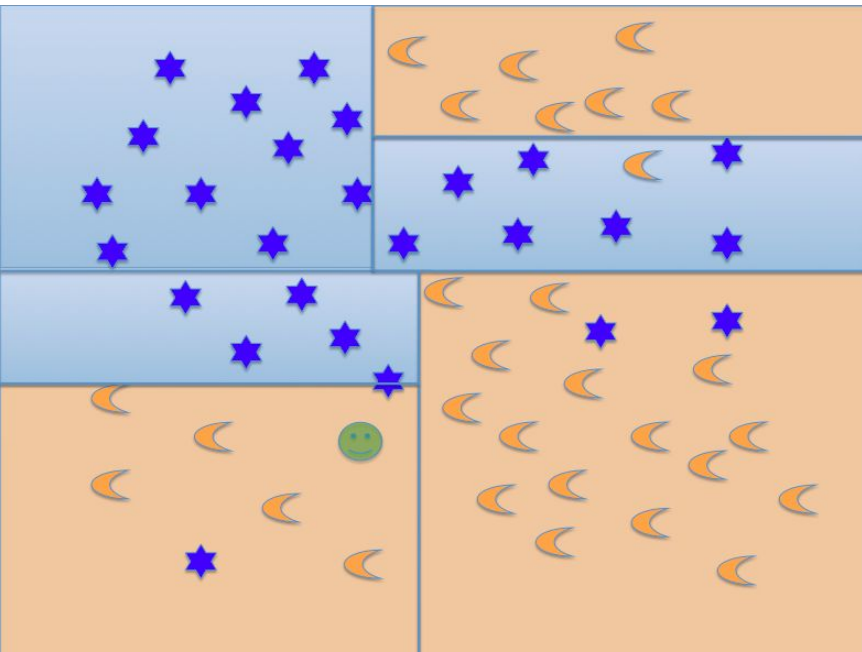
APPRENTISSAGE : VAR. CONTINUES



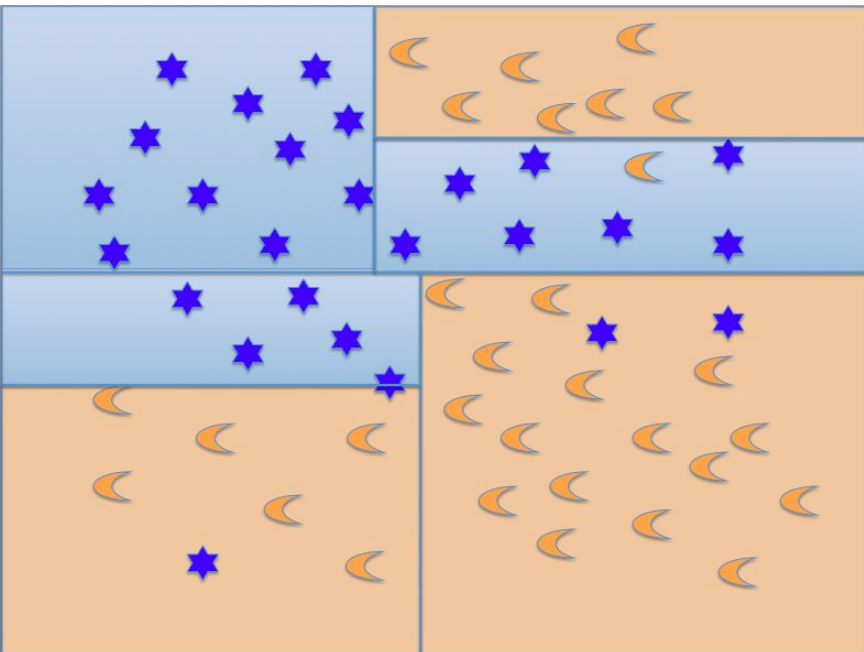
TEST/PREDICTION : VAR. CONTINUES



TEST/PREDICTION : VAR. CONTINUES



TEST/PREDICTION : VAR. CONTINUES



VOCABULAIRE

Noeud : endroit de coupure.....

Racine : noeud initial, aucune coupure faite.

Feuille : extrémité de l'arbre (noeud non divisé).

Profondeur : nombre de niveaux de l'arbre

Taille minimale des noeuds : nombre minimal de points de l'ensemble d'apprentissage toléré dans un noeud non-feuille.

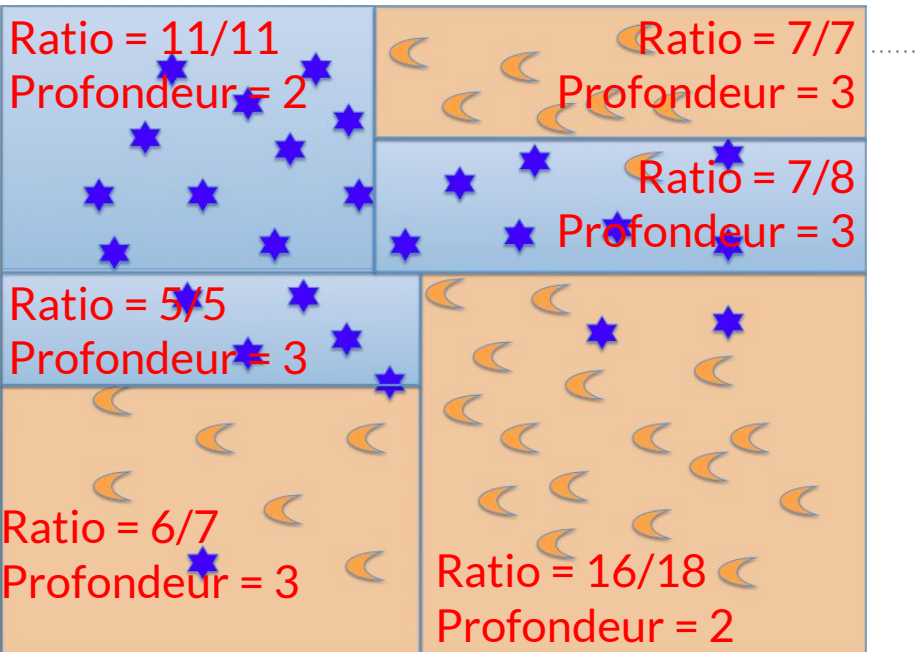
Critère de séparation : critère selon lequel on choisit la variable et/ou la coupure à faire au niveau d'un noeud.

TESTER SI UN NOEUD EST TERMINAL

Un noeud est dit “terminal” quand on arrête de le séparer. C’est donc une feuille. Plusieurs critères (c’est un **OU**) :

- **Profondeur** : le noeud a atteint la profondeur max.
- **Taille** : le noeud a moins que le nombre minimal d’éléments.
- **Ratio des classes dans le noeud** : le ratio classe majoritaire / total des éléments du noeud est supérieur à une valeur seuil
- Ou autres critères similaires du type “séparer le noeud ne sert visiblement à rien”.

TESTER SI UN NOEUD EST TERMINAL



CHOIX DE LA COUPURE

Objectif: Choisir la variable, **et** sa coupe, qui maximise le gain en **pureté** ou en information (on ne segmente que sur **1 variable à la fois**).

- **Continu** : la segmentation est typiquement de forme "variable > valeur" ou "variable \leq valeur"
- **Discret** : la segmentation se fait sur 1^(*) valeur (e.g.: "variable = valeur" ou "variable \neq valeur")

Approche gloutonne : on teste chaque variable et chaque coupure possible et on choisit la meilleure *à ce stade*.

→ Mais comment **évalue**-t-on?

INDICE DE GINI

.....
Définition : L'indice de Gini (ou coefficient d'**impureté** de Gini) mesure combien de fois un exemple choisi au hasard dans un ensemble serait mal classifié **si il était classifié selon la distribution dans l'ensemble**.

Traduction : Combien me coûterait-il de choisir cette coupure à tort?

INDICE DE GINI

.....
Définition : L'indice de Gini (ou coefficient d'**impureté** de Gini) mesure combien de fois un exemple choisi au hasard dans un ensemble serait mal classifié **si il était classifié selon la distribution dans l'ensemble**.

Traduction : Combien me coûterait-il de choisir cette coupure à tort?

Exemple : soit un ensemble avec 12 éléments de la classe 1 et 8 éléments de la classe 2. La distribution est donc $P_1 = 12/20 = 3/5$ et $P_2 = 8/20 = 2/5$. Si l'on prend un élément au hasard, on lui attribuerait $3/5$ du temps le label 1 et $2/5$ du temps le label 2. En moyenne, cet élément serait mal classé $\frac{3}{5} * \frac{2}{5} + \frac{2}{5} * \frac{3}{5} = 12/25$ du temps (calcul d'espérance).

INDICE DE GINI

Définition : Il mesure la probabilité de mal.....
classifier si on tire au hasard *selon la repartition*.

Exemple [au tableau] :

Avec 20 éléments: 12 de la classe 1, 8 de la
classe 2, on obtient l'indice de Gini:

$$3/5 * 2/5 + 2/5 * 3/5 = 12/25$$

On calcule donc l'indice de Gini pour *chaque*
coupure possible, on choisit la variable/coupure
pour laquelle cet indice est le plus petit.

TESTS DE COUPURE: CAS DISCRET

On calcule l'indice pour chaque variable.

Exemple : couper selon

"domicile": 5V, 3F

V \Rightarrow 3 gagné, 2 perdus


F \Rightarrow 1 gagné, 2 perdus

Moyenne **pondérée**

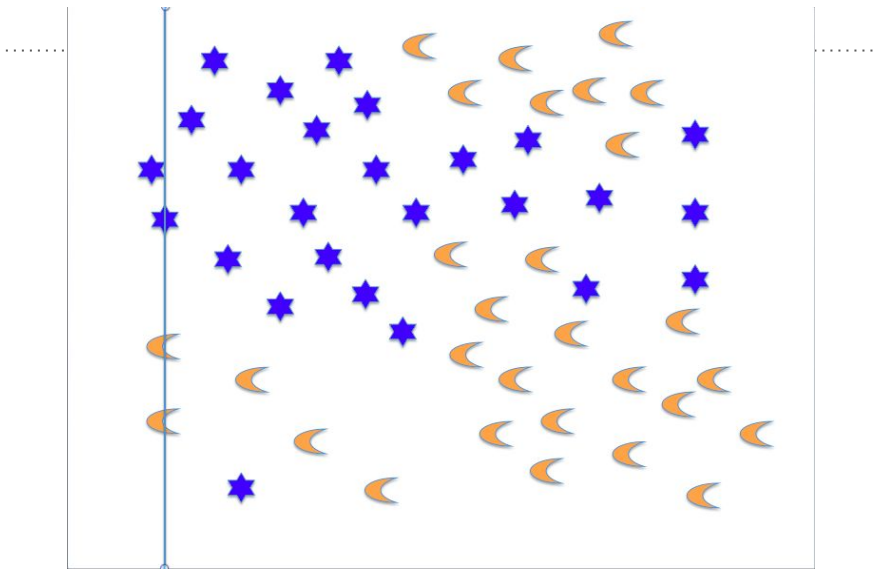
des Indices de Gini :

$$\frac{5}{8} \times \text{Gini}(\frac{3}{5}, \frac{2}{5}) + \frac{3}{8} \times \text{Gini}(\frac{1}{3}, \frac{2}{3}) \approx 0.4666$$

\rightarrow On fait de même pour **toutes** les variables, on garde celle qui **minimise** cette valeur.

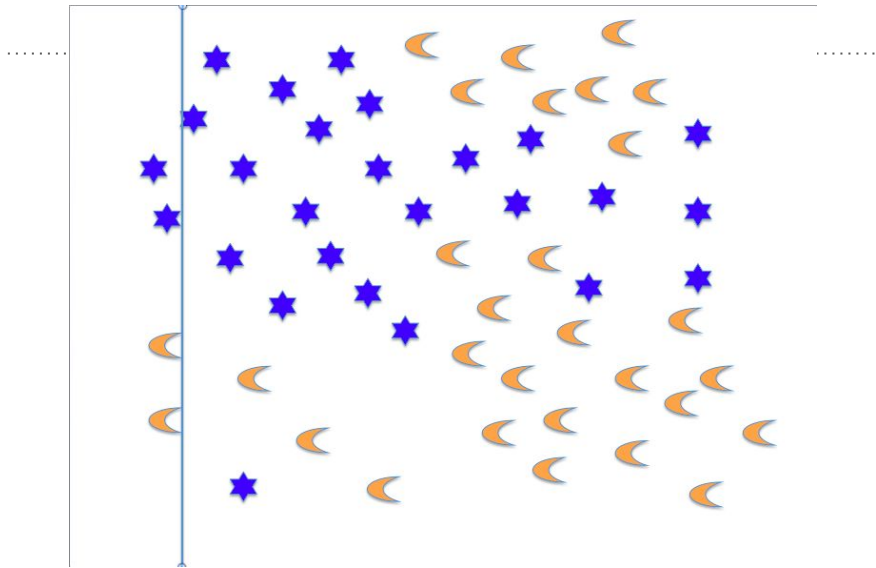
Dom. ?	Sam. ?	 ?	P.G.?	Match Gagné
V	V	F	F	G
F	F	F	V	G
V	V	V	F	G
V	V	F	V	G
F	V	V	V	P
F	F	V	F	P
V	F	V	V	P
V	F	V	F	P

TESTS DE COUPURE: CAS CONTINU



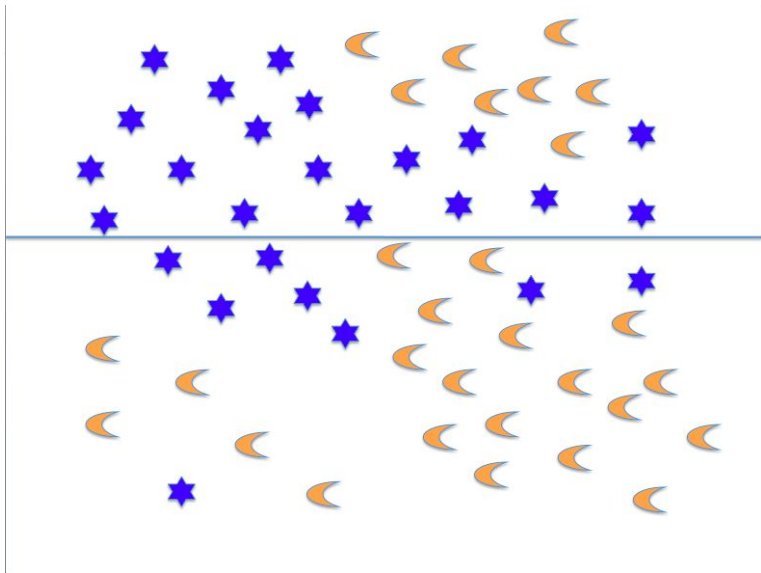
$$\frac{1}{55} \times \text{Gini}(1, 0) + \frac{54}{55} \times \text{Gini}\left(\frac{25}{54}, \frac{29}{54}\right) \approx 0.488$$

TESTS DE COUPURE: CAS CONTINU



$$4/55 \times \text{Gini}(2/4, 2/4) + 51/55 \times \text{Gini}(24/51, 27/51) \approx 0.498$$

TESTS DE COUPURE: CAS CONTINU



$$26/55 \times \text{Gini}(18/26, 8/26) + 29/55 \times \text{Gini}(8/29, 21/29) \approx 0.412$$

ENTROPIE DE SHANNON

L'**entropie** de Shannon est un autre critère selon lequel on peut choisir et couper la bonne variable.

$$\sum_i p_i \cdot \log_2(p_i)$$

Principe : plus les données sont dispersées, plus l'entropie est grande. On cherche à la **réduire**.

Minimisation des maths en cours : nous ne détaillerons pas cette notion. Cf [Wikipedia](#).

AUTRES

Variance reduction (cf [wikipedia](#))
etc...

Pour approfondir le sujet:
[en.wikipedia.org/wiki/Decision tree learning](https://en.wikipedia.org/wiki/Decision_tree_learning)
(lien en [français](#))

BLIBS, BLOBS, BLABS ET BLUBS

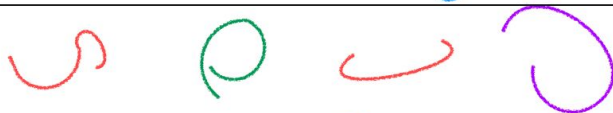
Blib



Blob



Blab



Blub



BLIBS, BLOBS, BLABS ET BLUBS



?



?

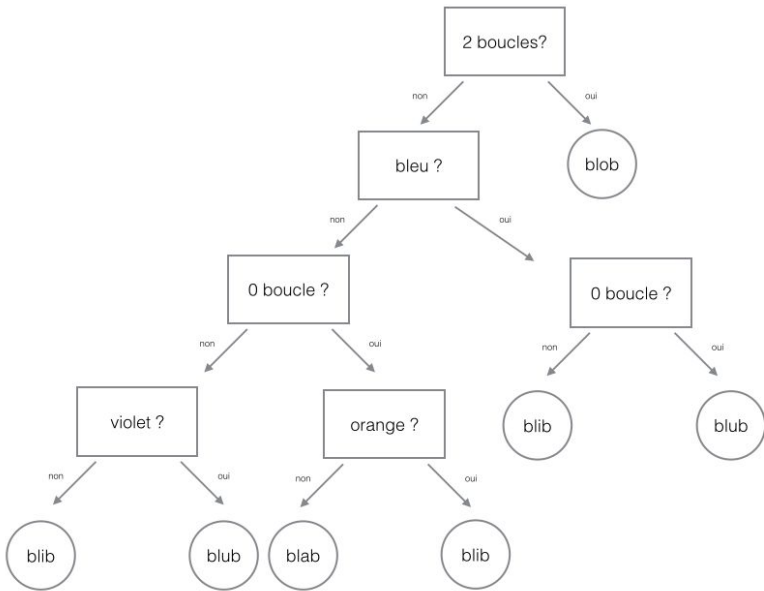


?



?

BLIBS, BLOBS, BLABS ET BLUBS



EN PYTHON

```
from sklearn.tree import \
    DecisionTreeClassifier

model = DecisionTreeClassifier(
    criterion = "gini",
    max_depth = 4,
    min_samples_split = 3)
model.fit(X,y)
predictions = model.predict(Xtest)
```

CONCLUSION

Avantages:

- Ultra adaptables à tout type de feature
- Très interprétables !
- Faciles à implémenter

Inconvénients:

- Forte élasticité aux exemples : si l'on change un exemple, l'arbre peut changer complètement.
- Risque de **sur-apprentissage**.
- Peuvent nécessiter beaucoup de calculs, à cause de l'approche gloutonne, si il y a énormément d'éléments (peut se résoudre...)

Forêts aléatoires

(RANDOM FORESTS)

INTRODUCTION

Algorithme de 1999 [Breiman, "Random Forests"]

Idée : au lieu de faire un arbre, on fait une forêt.

→ Algorithme très puissant, qui règle les problèmes liés à un arbre seul.

Un des algorithmes les plus efficaces sur de nombreux problèmes.

Basé sur la **randomisation**

LE BOOTSTRAP

Idee : on transforme l'ensemble d'apprentissage en B ensembles d'apprentissage.

- On part avec N exemples.
- B fois de suite, on tire au hasard **avec remise** N exemples parmi les N .

Exemple: $N = 10$. $B = 3$.

→ {3, 6, 7, 4, 1, 8, 3, 3, 2, 9}

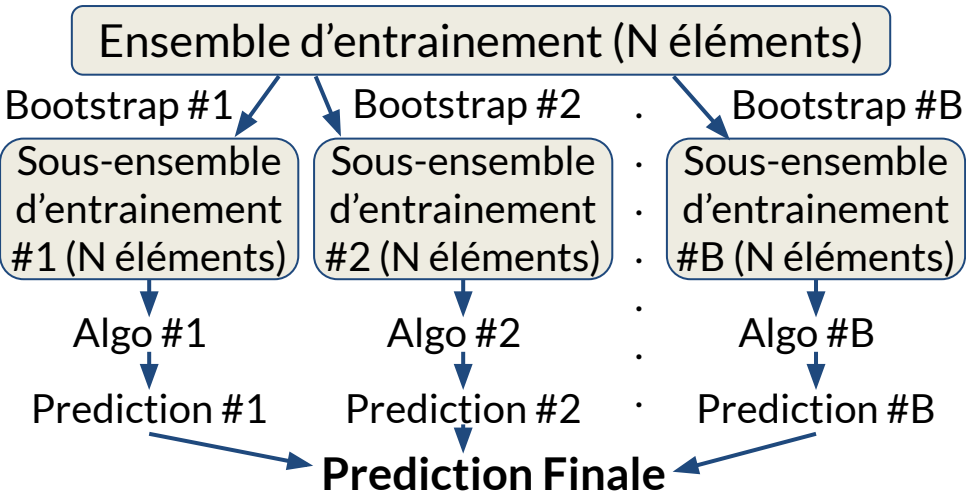
→ {2, 8, 5, 0, 0, 8, 6, 1, 0, 4}

→ {5, 9, 4, 9, 7, 8, 2, 9, 9, 1}

Ainsi on décuple la **variabilité** de l'ensemble d'apprentissage et on évite le **sur-apprentissage**

ALGORITHMES D'ENSEMBLE

Pour booster la puissance d'un algorithme, on peut le lancer sur B ensembles d'apprentissage bootstrappés et agréger les résultats :



ALGORITHME

Algorithm: Random Forests

INPUT:

- Ensemble d'apprentissage (X, Y)
- Nombre d'arbres B; Paramètres

For $i = 1 \dots B$:

 random.seed(i)

$(X_i, Y_i) = \text{Bootstrap}(X, Y, \text{random}())$

 Faire un arbre T_i à partir de X_i, Y_i . Le stocker.

PRECISIONS

Afin de limiter le temps de calcul, on introduit un paramètre supplémentaire : à chaque noeud, on choisit au hasard un nombre de variables (i.e. colonnes) à regarder.

Exemple : si on fait du texte, on a 10k valeurs TF-IDF. A chaque noeud, on choisit la coupe parmi un petit nombre (~ 100) pris au hasard.

Plus B est grand plus l'algorithme est efficace (et plus il est long).

Les autres paramètres sont les mêmes que pour les arbres seuls.

AGGREGATION

Dans le cas de la classification, on fera un **vote majoritaire** sur tous les arbres :

- Le premier arbre prédit "perdu"
 - Le second arbre prédit "perdu"
 - Le troisième arbre prédit "gagné"
- ⇒ Prédiction finale : **"perdu"**

En régression (*), on fera une **moyenne**:

- Le premier arbre prédit 1.1
 - Le second arbre prédit 2.0
 - Le troisième arbre prédit 1.9
- ⇒ Prédiction finale : **1.7**

EN PYTHON

```
from sklearn.ensemble import \
    RandomForestClassifier

model = RandomForestClassifier(
    n_estimators = 1000,
    criterion = "gini",
    max_features = "sqrt")

model.fit(X,y)

predictions = model.predict(Xtest)
print(model.feature_importances_)
```


FEATURE IMPORTANCES

Il est possible de donner un score aux variables du problème en calculant le pourcentage des fois où elles ont été choisies pour la coupure: plus une variable a été choisie souvent, plus elle est “importante”.

On appelle ce score le **feature importance**.

On utilise parfois les random forest uniquement pour produire ces scores! (e.g. réduction du nombre de colonnes)

CONCLUSION

Avantages:

- Très puissant car basé sur **la sagesse des foules** (B avis valent mieux qu'un)
- La vitesse de calcul peut être réglée en fonction des paramètres.
- **Parallélisable.**
- Empiriquement un des **meilleurs** algorithmes de prédiction.

Inconvénients:

- Difficile à interpréter.
- Peut être lent si B est grand.

CONCLUSION SUR LES METHODES ALGORITHMIQUES

- L'apprentissage par des modèles algorithmiques ne demande **pas d'hypothèse** sur la **forme** des données.
- Il faut choisir les paramètres **méthodiquement** : un changement peut modifier votre résultat.
- Comprendre **la logique de l'algorithme** permet de mieux l'utiliser et de mieux savoir quoi lui donner en entrée, et les choix à explorer.
- Mieux vaut un algo moins bon sur l'ensemble d'apprentissage **mais qui sait généraliser**.