



ANALYSE DE DONNÉES

Nassim Laga
nassimlaga@gmail.com
Janvier 2019

Plan du cours

- Introduction au data mining (1h)
- **Apprentissage supervisé et non supervisé**
- **Focus sur l'apprentissage supervisé + mesures de performance et de validation (2h)**
- Focus sur l'apprentissage non supervisé + mesures de performance et de validation (2h)
- Focus sur l'analyse de texte (2h)



APPRENTISSAGE SUPERVISÉ

Nassim Laga
nassimlaga@gmail.com
Janvier 2019

Plan du jour

- Apprentissage supervisé et non supervisé

 - Motivation & Use cases

 - Formulation mathématique

 - Apprentissage supervisé

 - Apprentissage non supervisé

- Apprentissage supervisé

 - Les différentes catégories des algorithmes

 - La régression linéaire

 - La régression logistique

 - Arbres de décision

 - Random Forest

 - Les mesures de performance



APPRENTISSAGE SUPERVISÉ ET NON SUPERVISÉ

Apprentissage supervisé et non supervisé - Motivation et use cases

- Marketing
 - Classification client pour
 - adapter les offres
 - ciblage publicitaire
 - anticiper le churn
- Santé
 - Prédiction de la propagation d'épidémies
 - Classification d'images médicales
 - détection de tumeur dans un scanner
 - Prédiction du traitement approprié selon les symptômes
- Assurance
 - prédire le risque d'un accident/catastrophe en fonction de ...
- Sécurité
 - détection d'attaque, de fraude, etc.
- ...etc.

Apprentissage supervisé et non supervisé – Formulation mathématique

- Apprentissage supervisé – modélisation mathématique
 - Soit $K(X_1, \dots, X_n, Y)$ les données d'apprentissage. Il s'agit d'un ensemble **d'observations** des variables **explicatives** X_i (\mathcal{X}) et de la variable **à expliquer** Y (\mathcal{Y}).
 - L'objectif d'un apprentissage supervisé est de définir une **bonne** fonction $f: \mathcal{X} \rightarrow \mathcal{Y}$ qui permet de trouver la valeur probable de Y en fonction des X_i .

Apprentissage supervisé et non supervisé – Formulation mathématique

- Apprentissage supervisé – modélisation mathématique
 - Que veut dire une **bonne** fonction f
 - Celles qui minimise l'erreur
 - On définit alors la fonction de perte, ou Cost function, ou loss function en anglais
 - $l(Y, f(X_i))$ pour mesurer les erreurs de prédiction de la fonction f
 - il peut avoir plusieurs formules à la fonction de coût
 - $l(Y, f(X)) = 1_{Y \neq f(X)}$
 - $l(Y, f(X)) = |Y - f(X)|^2$

Apprentissage supervisé et non supervisé – Formulation mathématique

- Apprentissage supervisé – modélisation mathématique
 - Une **bonne** fonction f est celle qui minimise la moyenne de l'erreur, à savoir la moyenne de $l(Y, f(X))$ pour tous les couples de valeurs X et Y .
 - $E[l(Y, f(X))]$

Trouver la fonction f qui minimise l'erreur en moyenne **à partir des données d'apprentissage**.

En pratique, il faudra également que la fonction f minimise l'erreur en moyenne **sur les nouvelles données**.

Apprentissage supervisé et non supervisé – Formulation mathématique

- Apprentissage non supervisé – modélisation mathématique
 - Soit $K(X_1, \dots, X_n)$ les données d'apprentissage. Il s'agit d'un ensemble **d'observations** des variables **explicatives** $X_i (X)$
 - But :
 - Créer des regroupements (clusters) **cohérents/homogène**
 - Que veut dire alors **cohérent et homogène** ?

Apprentissage supervisé et non supervisé – Formulation mathématique

- Apprentissage non supervisé – modélisation mathématique
 - Il s'agit de définir une fonction $f: X \rightarrow D: \{1, \dots, k\}$
 - k étant le nombre de groupes qu'on souhaite créer à partir des observations X
 - Contrairement à un problème de classification
 - la cible n'est pas connue, pas de base de vérité
 - l'analyse se fait uniquement sur les données d'apprentissage

Apprentissage supervisé et non supervisé – Formulation mathématique


- Apprentissage non supervisé – modélisation mathématique
 - L'absence de base de vérité → nécessite de définir des mesures de qualité, qui permettraient de valider ou pas le résultat
 - Un **bon** regroupement (clustering) est souvent défini
 - Homogénéité intra-cluster
 - Distance (inhomogénéité) inter-clusters

Apprentissage supervisé et non supervisé – Formulation mathématique

- Apprentissage non supervisé – modélisation mathématique
 - Ces métriques sont souvent basées sur la notion de distance entre les individus ou encore entre les clusters
 - Exemple de distance
 - variance intra-cluster
 - variance inter-cluster



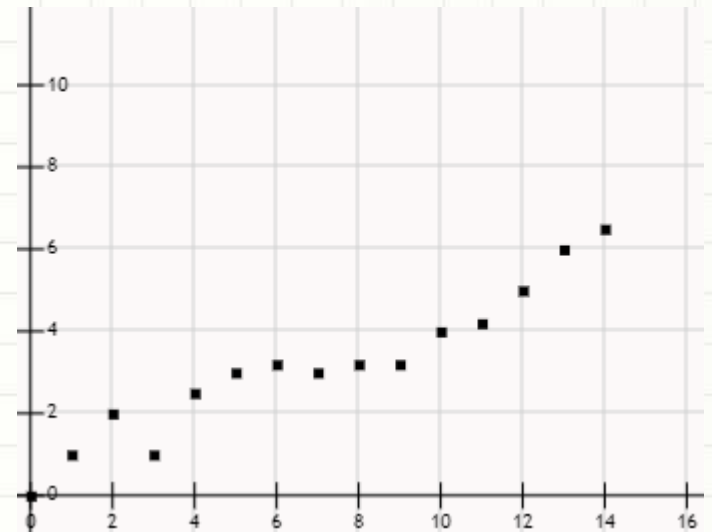
APPRENTISSAGE SUPERVISÉ



Apprentissage supervisé – Les différentes catégories des algorithmes

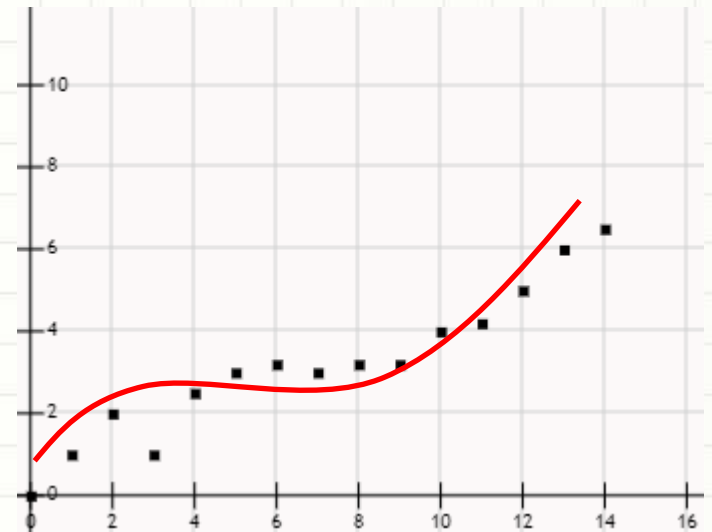
Apprentissage supervisé – Régression linéaire

- Problème de régression
 - Soit $K(X_1, \dots, X_n, Y)$ les données d'apprentissage
 - Y est une valeur **continue**, ou valeur **quantitative**
 - il faut définir la fonction f qui minimise l'erreur



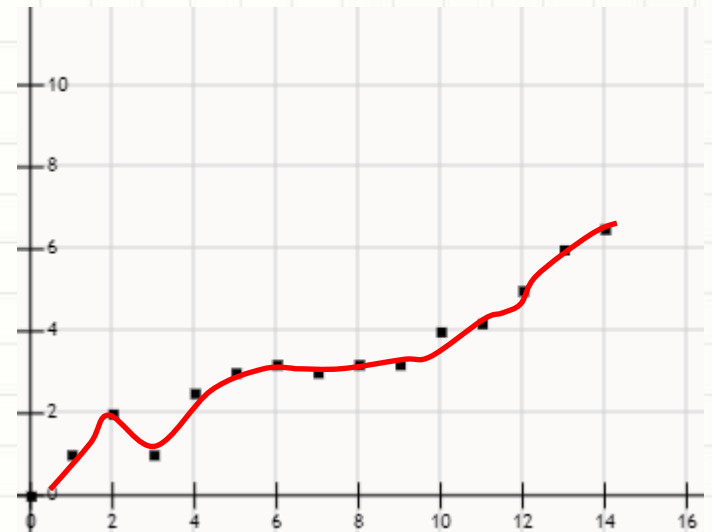
Apprentissage supervisé – Régression linéaire

- Problème de régression
 - Soit $K(X_1, \dots, X_n, Y)$ les données d'apprentissage
 - Y est une valeur **continue**, ou valeur **quantitative**
 - il faut définir la fonction f qui minimise l'erreur



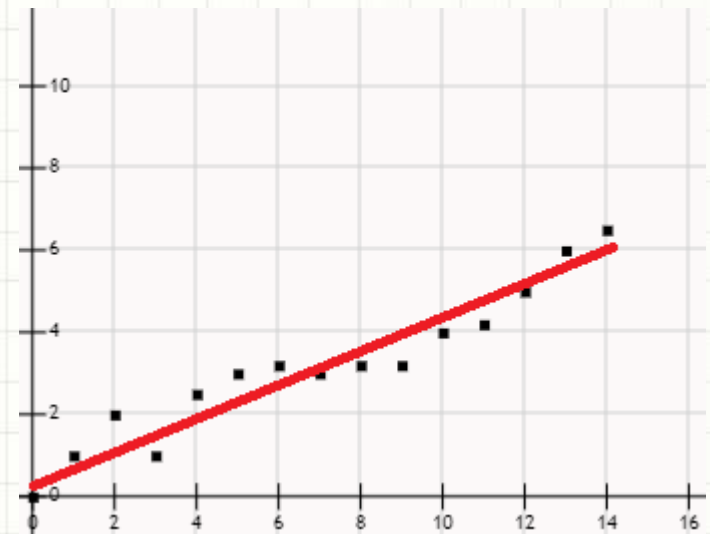
Apprentissage supervisé – Régression linéaire

- Problème de régression
 - Soit $K(X_1, \dots, X_n, Y)$ les données d'apprentissage
 - Y est une valeur **continue**, ou valeur **quantitative**
 - il faut définir la fonction f qui minimise l'erreur



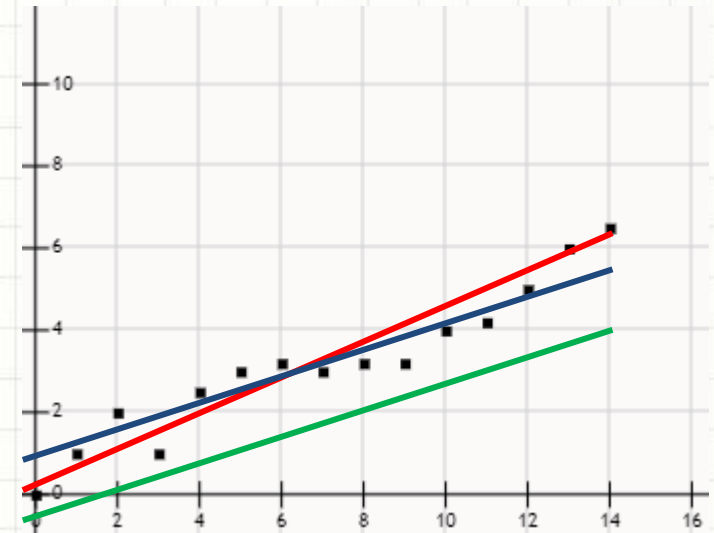
Apprentissage supervisé – Régression linéaire

- Problème de régression
 - Soit $K(X_1, \dots, X_n, Y)$ les données d'apprentissage
 - Y est une valeur **continue**, ou valeur **quantitative**
 - il faut définir la fonction f qui minimise l'erreur
- Une régression est dite linéaire lorsque f est une fonction d'une droite.
 - $y = f(X) = a_0 + a_1X$



Apprentissage supervisé – Régression linéaire

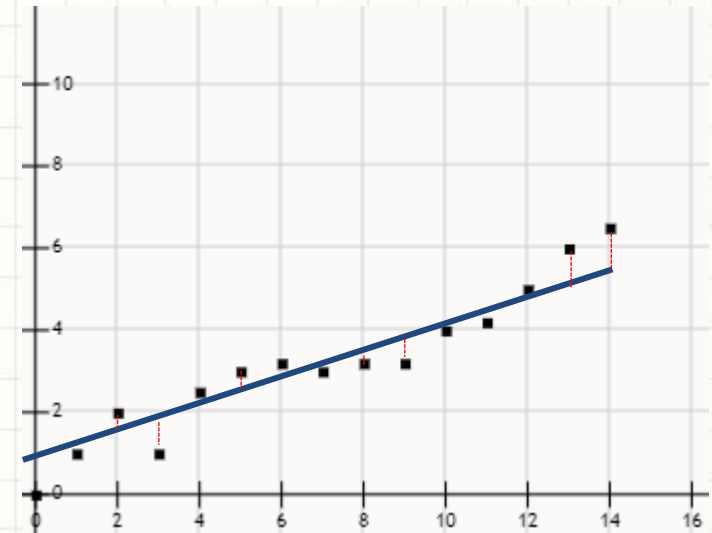
- $y = f(X) = a_0 + a_1X$
- Comment trouver la **meilleur** f linéaire
- Comment définir mathématiquement la notion de **meilleure** dans notre cas ?



Apprentissage supervisé – Régression linéaire

- $y = f(X) = a_0 + a_1X$
- Comment définir mathématiquement la notion de **meilleure** dans notre cas ?
 - Cela correspond à la fonction f qui minimise la somme des distances entre la prédiction et la valeur réelle dans les données d'apprentissage
 - Mathématiquement, cela revient à minimiser

$$\sum_{k=0}^n (f(x_k) - y_k)^2 = \sum_{k=0}^n (a_0 + a_1x_k - y_k)^2$$



Apprentissage supervisé – Régression linéaire

- L'algorithme de descente de gradient (Gradient Descent)
 - Un algorithme d'optimisation (pour trouver le min ou le max d'une fonction)
 - Fonction à optimiser
 - $J(a_0, a_1) = \sum_{k=0}^n (a_0 + a_1 x_k - y_k)^2$
 - Résumé de l'algorithme
 - Définir un point aléatoire (a_0, a_1)
 - Déplacer itérativement le point de sorte à réduire la fonction J jusqu'à ce qu'on arrive au minimum, ou si le nombre maximum d'itération atteint

Apprentissage supervisé – Régression linéaire

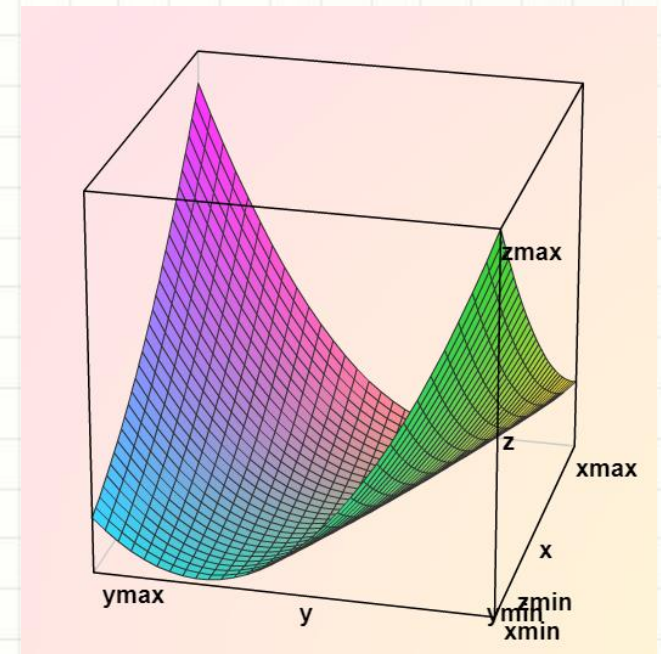
- L'algorithme de descente de gradient (Gradient Descent)
 - Soit le jeu de données suivant
- Tracez ces points sur un plan
- Ecrire la fonction J
- En supposant que $a_0=0$, réécrire la fonction, que déduisez vous ?

x	y
1	1
2	2,1
3	2,9

Apprentissage supervisé – Régression linéaire

- L'algorithme de descente de gradient (Gradient Descent)
 - Soit le jeu de données suivant
- Tracez ces points sur un plan
- Ecrire la fonction J
- En supposant que $a_0=0$, réécrire la fonction, que déduisez vous ?

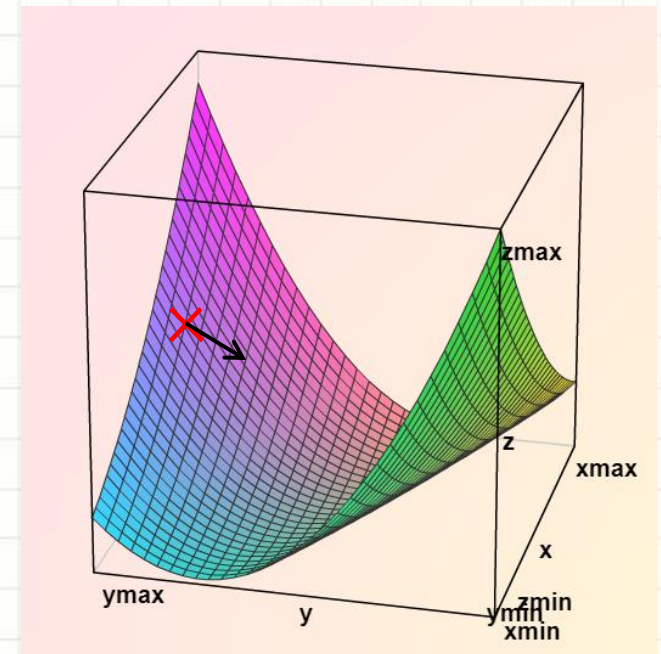
x	y
1	1
2	2,1
3	2,9



Apprentissage supervisé – Régression linéaire

- L'algorithme de descente de gradient (Gradient Descent)
 - Résumé de l'algorithme (illustration sur le graphe)
 - Définir un point aléatoire (a_0, a_1)
 - Déplacer itérativement le point de sorte à réduire la fonction J jusqu'à ce qu'on arrive au minimum, ou nombre maximum d'itération atteint

x	y
1	1
2	2,1
3	2,9



Apprentissage supervisé – Régression linéaire

- L'algorithme de descente de gradient (Gradient Descent)
 - L'algorithme du point vue mathématique
 - Répéter jusqu'à convergence
 - $a_i = a_i - \alpha \frac{\partial}{\partial a_i} J(a_0, a_1) \quad (i = 0, i = 1)$
 - *La dérivé permet de définir le sens du déplacement*
 - *Attention les a_i doivent être modifiés simultanément*
 - *Explication sur le tableau à travers un exemple sur 2D*

La régression linéaire en python

- **numpy.corrcoef** permet d'avoir une estimation de la corrélation que peut exister entre deux variables. <https://docs.scipy.org/doc/numpy-1.15.0/reference/generated/numpy.corrcoef.html>

```
In [5]: from numpy import corrcoef
        corrcoef(dataset['CholRate'].values.tolist(), dataset['Age'].values.tolist())

Out[5]: array([[ 1.          ,  0.97824162],
               [ 0.97824162,  1.          ]])
```

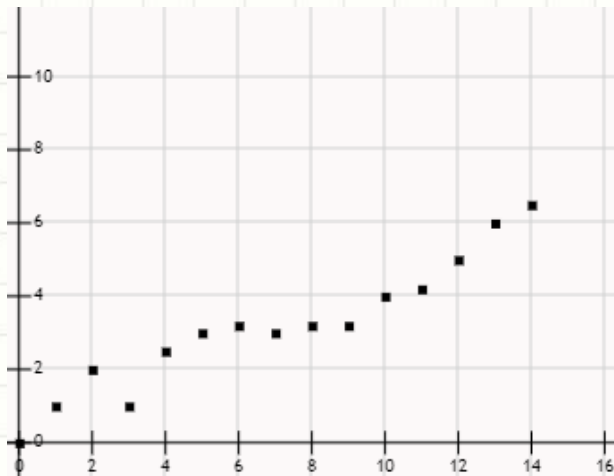
- **sklearn.linear_model.LinearRegression** implémente l'algorithme de régression linéaire. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

```
In [9]: # Fitting simple linear regression to training set
        from sklearn.linear_model import LinearRegression
        regressor = LinearRegression()
        regressor.fit(X_train, y_train)

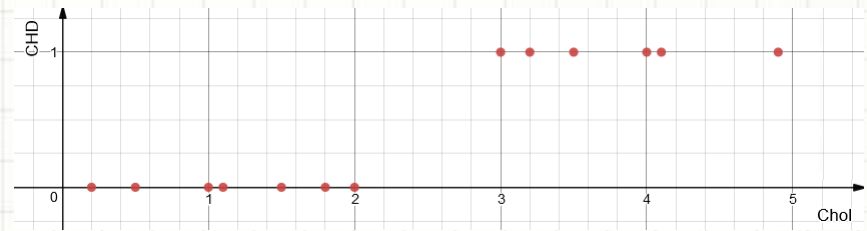
Out[9]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

Apprentissage supervisé – La régression logistique

- La régression logistique est similaire à la régression linéaire sauf que



régression linéaire

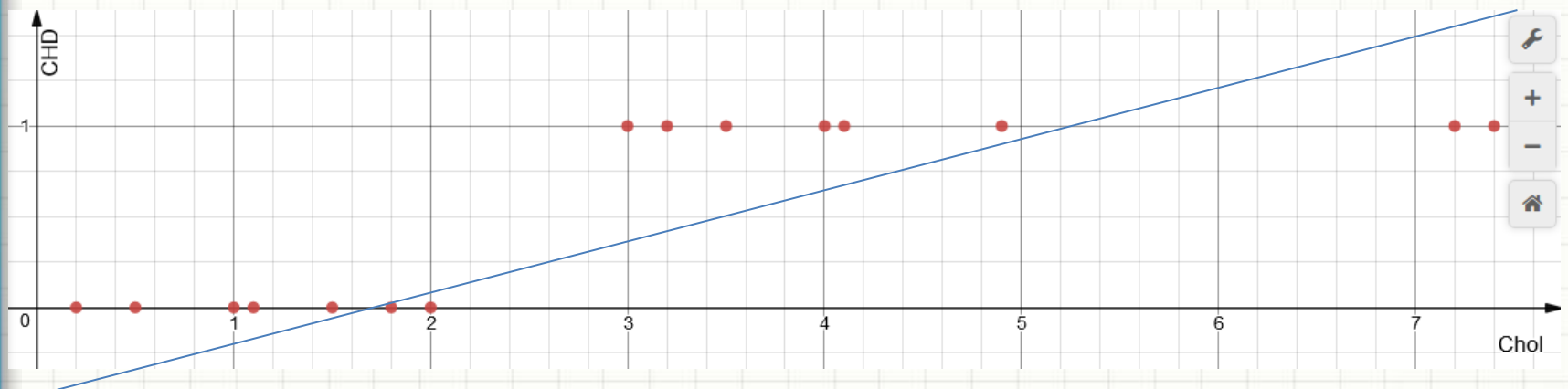
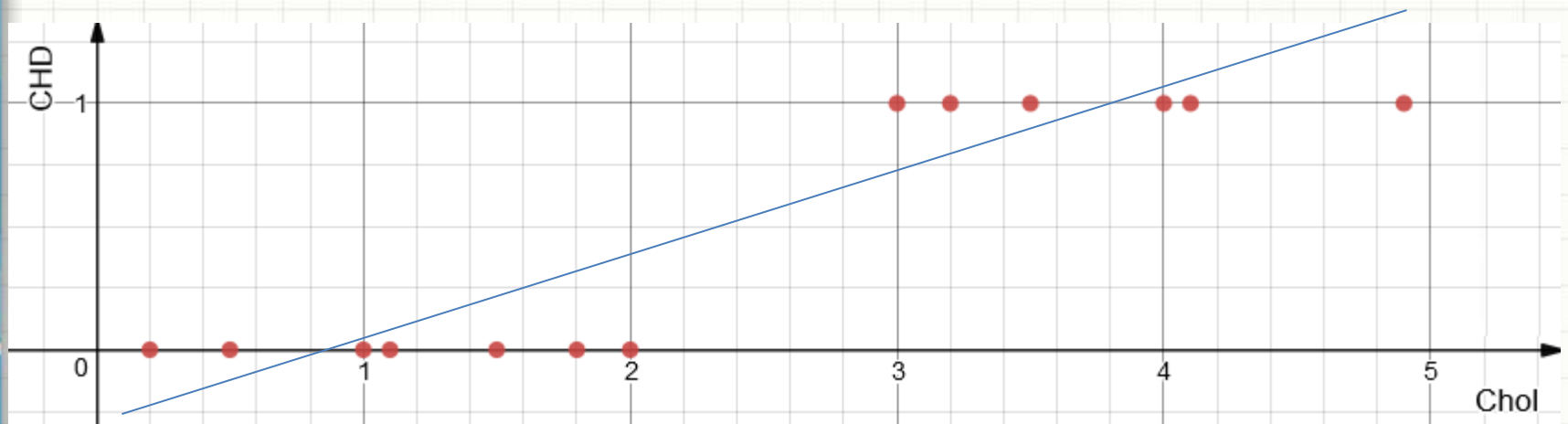


régression logistique

- Qlqs exemples
 - Prédiction d'un prix optimal d'un produit
 - Prédiction d'un prix de maison selon ses caractéristiques
 - Prédiction d'un poids en fonction des habitudes alimentaires
 - ...etc.
- Qlqs exemples
 - détection de spams
 - classification de mails (important, non important)
 - tumeur ou pas en fonction de la taille
 - client vip ou pas
 - ...etc.

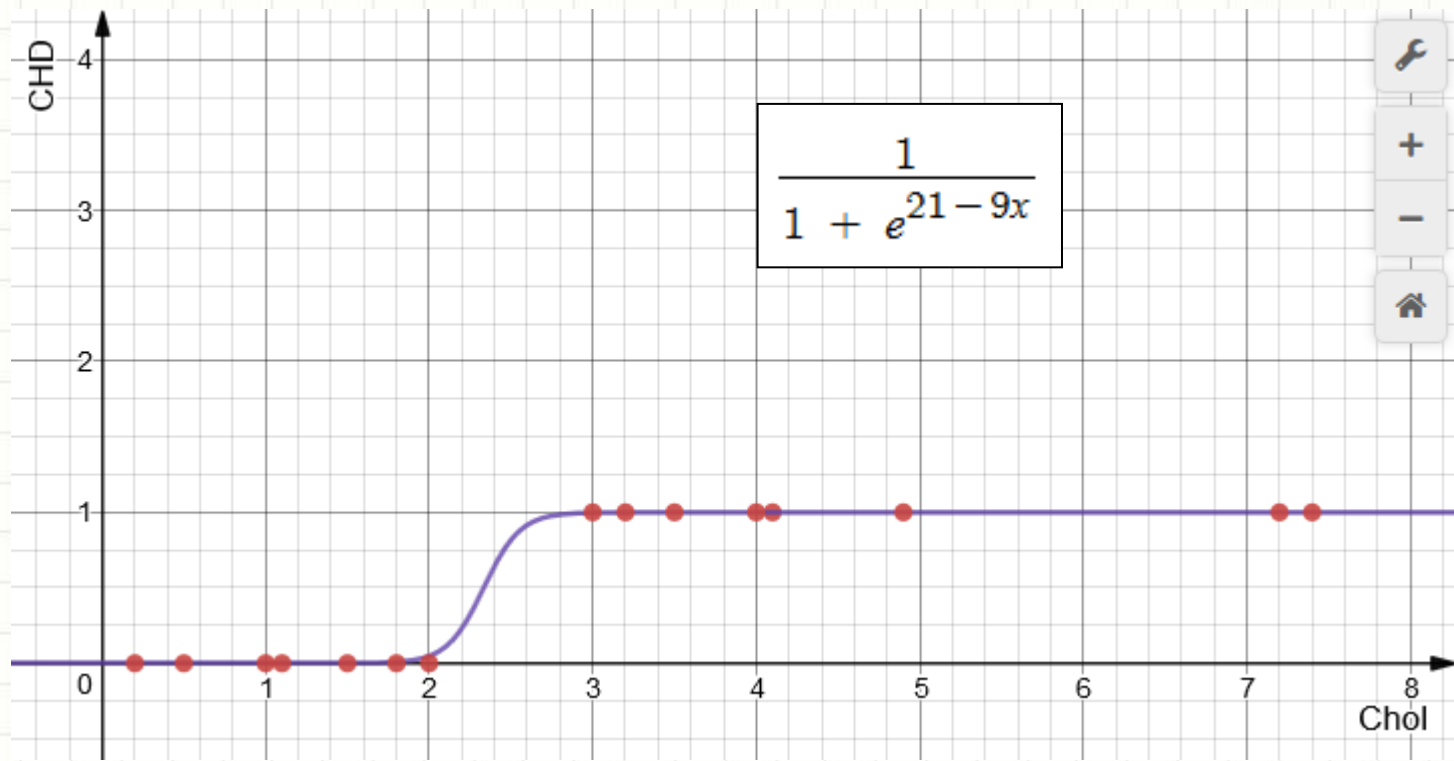
Apprentissage supervisé – La régression logistique

- Pourquoi on ne peut pas appliquer une régression linéaire ?



Apprentissage supervisé – La régression logistique

- Pourquoi on ne peut pas appliquer une régression linéaire ?



Apprentissage supervisé – La régression logistique

- La régression logistique est donc basée sur la fonction **sigmoïde**, appelée aussi, courbe en **S**, ou alors **logistique**

$$- f(x) = \frac{1}{1 + e^{-\sum_0^m a_i x_i}}$$

- la fonction f renvoie une probabilité que y soit égale 1 sachant x

$$- \underline{f(x) = P(y = 1/x)}$$

- Comment définir f ? cela revient à identifier les paramètres a_i qui minimiseraient l'erreur.

Apprentissage supervisé – La régression logistique

- Minimiser l'erreur (cost function)

- L'erreur dans la régression linéaire

$$\text{cost}(f(x), y) = J(A) = \sum_{k=0}^n (f(x_k) - y_k)^2$$

- Peut on appliquer la même fonction dans notre cas ?

- L'erreur dans la régression logistique

$$\text{cost}(f(x_k), y) = \begin{cases} -\log(f(x_k)) & \text{si } y = 1 \\ -\log(1 - f(x_k)) & \text{si } y = 0 \end{cases}$$

Apprentissage supervisé – La régression logistique

- Minimiser l'erreur (cost function)
- $cost(f(x_k), y_k) = -y_k(\log(f(x_k))) - (1 - y_k)\log(1 - f(x_k))$
- $J(A) = \frac{1}{n} \sum_{k=0}^n cost(f(x_k), y_k)$
 - $f(x_k) = \frac{1}{1 + e^{-\sum_{i=0}^m a_i * x_{k,i}}}$
- La fonction J présente l'avantage d'être convexe, et donc on peut appliquer l'algorithme de descente de gradient.

La régression logistique en python

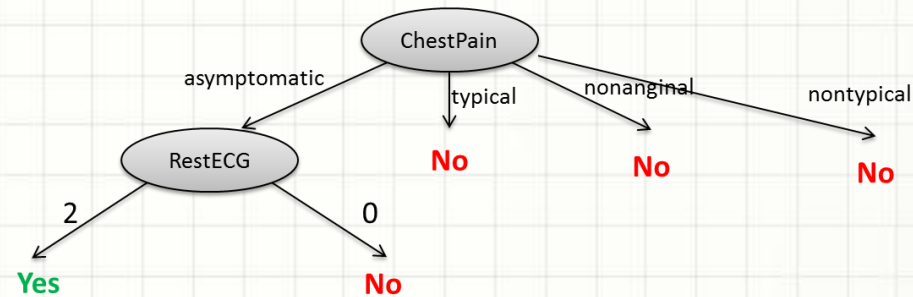
- **sklearn.linear_model.LogisticRegression**. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

```
In [ ]: from sklearn.datasets import load_iris
        from sklearn.linear_model import LogisticRegression
        X, y = load_iris(return_X_y=True)
        clf = LogisticRegression(random_state=0, solver='lbfgs',
                                multi_class='multinomial').fit(X, y)

        clf.predict(X[:2, :])
        clf.predict_proba(X[:2, :])
        clf.score(X, y)
```

Apprentissage supervisé – Les arbres de décision

- Un arbre de décision, de manière très basique, est
 - un outil d'aide à la décision
 - un arbre où les nœuds représentent des **conditions**, et les feuilles, des **décisions**.



- Comment construire un arbre de décision à partir des données d'apprentissage ?
 - Algorithmes ID3 (1986), C4.5 (1993)

Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD	
0	1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
1	2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
2	3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
3	4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
4	5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
5	6	56	1	nontypical	120	236	0	0	178	0	0.8	1	0.0	normal	No
6	7	62	0	asymptomatic	140	268	0	2	160	0	3.6	3	2.0	normal	Yes
7	8	57	0	asymptomatic	120	354	0	0	163	1	0.6	1	0.0	normal	No
8	9	63	1	asymptomatic	130	254	0	2	147	0	1.4	2	1.0	reversable	Yes
9	10	53	1	asymptomatic	140	203	1	2	155	1	3.1	3	0.0	reversable	Yes

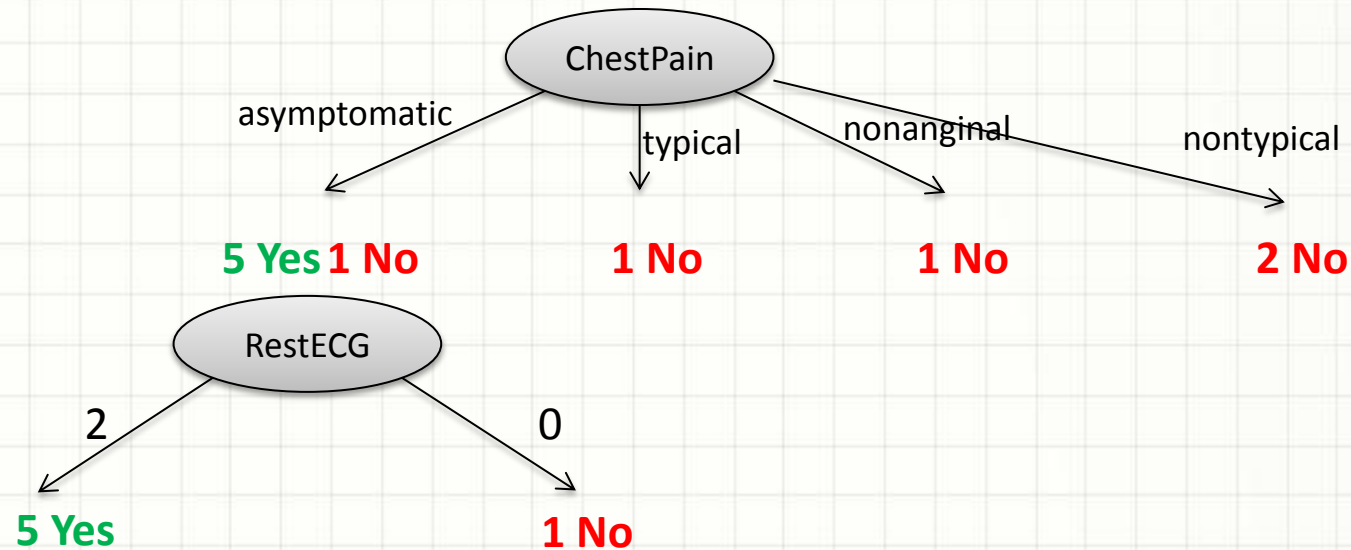
[1] J.R. Quinlan (1986) : "Induction of Decision Trees," **Machine Learning**, Vol. 1, pp.81-106.

[2] S.L. SALZBERG (1994) : "C4. 5: Programs for machine learning," by j. ross quinlan. morgan kaufmann publishers, inc., 1993." **Machine Learning**, 1994, vol. 16, no 3, p. 235-240.

Apprentissage supervisé – Les arbres de décision

- Algorithme ID3
 - Intuition

Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD	
0	1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
1	2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
2	3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversible	Yes
3	4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
4	5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
5	6	56	1	nontypical	120	236	0	0	178	0	0.8	1	0.0	normal	No
6	7	62	0	asymptomatic	140	268	0	2	160	0	3.6	3	2.0	normal	Yes
7	8	57	0	asymptomatic	120	354	0	0	163	1	0.6	1	0.0	normal	No
8	9	63	1	asymptomatic	130	254	0	2	147	0	1.4	2	1.0	reversible	Yes
9	10	53	1	asymptomatic	140	203	1	2	155	1	3.1	3	0.0	reversible	Yes



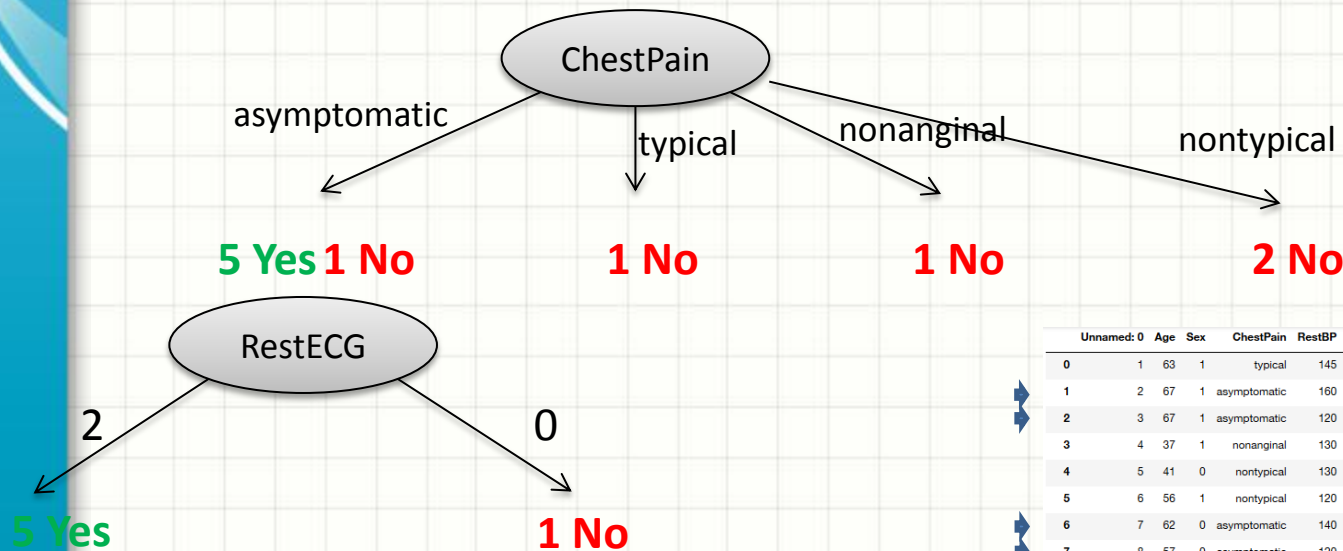
Apprentissage supervisé – Les arbres de décision

- Algorithme ID3
 - Pseudo code (détails)
 - ID3 (examples)
 - $A \leftarrow$ le meilleur attribut (variable) à tester et partager le dataset d'apprentissage
 - Créer un nœud A (decision node)
 - Pour chaque valeur V distincte de A , créer un sous ensemble des exemples S , tel que S contient les exemples dont la valeur de A est égale à V
 - Pour chaque sous ensemble S , si S est « pure », s'arrêter, sinon, ré-exécuter ID3(S)

Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD	
0	1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
1	2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
2	3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
3	4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
4	5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
5	6	56	1	nontypical	120	236	0	0	178	0	0.8	1	0.0	normal	No
6	7	62	0	asymptomatic	140	268	0	2	160	0	3.6	3	2.0	normal	Yes
7	8	57	0	asymptomatic	120	354	0	0	163	1	0.6	1	0.0	normal	No
8	9	63	1	asymptomatic	130	254	0	2	147	0	1.4	2	1.0	reversable	Yes
9	10	53	1	asymptomatic	140	203	1	2	155	1	3.1	3	0.0	reversable	Yes

Apprentissage supervisé – Les arbres de décision

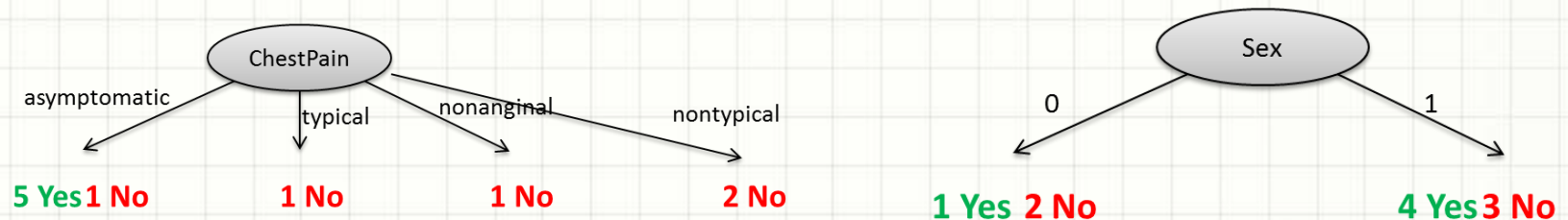
- Algorithme ID3
 - Pseudo code (détails)
 - ID3 (examples)
 - $A \leftarrow$ le meilleur attribut (variable) à tester et partager le dataset d'apprentissage
 - Créer un nœud A (decision node)
 - Pour chaque valeur V distincte de A , créer un sous ensemble des exemples S , tel que S contient les exemples dont la valeur de A est égale à V
 - Pour chaque sous ensemble S , si S est « pure », s'arrêter, sinon, ré-exécuter ID3(S)



Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD	
0	1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
1	2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
2	3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversible	Yes
3	4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
4	5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
5	6	56	1	nontypical	120	236	0	0	178	0	0.8	1	0.0	normal	No
6	7	62	0	asymptomatic	140	268	0	2	160	0	3.6	3	2.0	normal	Yes
7	8	57	0	asymptomatic	120	354	0	0	163	1	0.6	1	0.0	normal	No
8	9	63	1	asymptomatic	130	254	0	2	147	0	1.4	2	1.0	reversible	Yes
9	10	53	1	asymptomatic	140	203	1	2	155	1	3.1	3	0.0	reversible	Yes

Apprentissage supervisé – Les arbres de décision

- Algorithme ID3
 - Comment repérer le **meilleur** attribut sur lequel on crée le nœud de décision (decision node) ?



- Le but étant d'aboutir à des ensembles le plus « pure » possible
 - 5 yes/0 no, complètement pure
 - 3 yes/3 no, complètement impure
 - 0 yes/5 no, ?

Apprentissage supervisé – Les arbres de décision

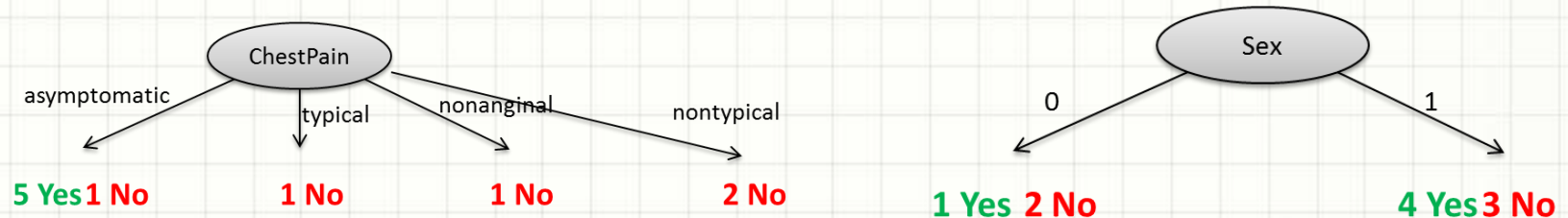
- Algorithme ID3
 - Il nous faut une métrique pour mesurer l'incertitude, ou la certitude
 - En d'autres termes, soit un ensemble S d'exemples positifs et de négatifs, vous tirez un exemple au hasard, à quel point vous êtes sûr d'avoir un positif ou négatif?
 - Probabilité ? $P(x=\text{yes})$
 - **Entropie de Shannon**

Apprentissage supervisé – Les arbres de décision

- Algorithme ID3
 - Entropie de Shannon
 - $H(S) = - \sum_{i=1}^n P_i * \log(P_i)$
 - $H(S) = -P_{yes} \log(P_{yes}) - P_{no} \log(P_{no})$
 - Interprétation : supposons qu'un item $X \in S$, combien d'essais vous avez besoin pour savoir si X est positif ou négatif ?
 - L'entropie peut être interprétée comme le nombre moyens de questions qu'il faut poser pour deviner une lettre tirée au hasard.

Apprentissage supervisé – Les arbres de décision

- Algorithme ID3
 - Comment repérer le **meilleur** attribut sur lequel on crée le nœud de décision (decision node) ?

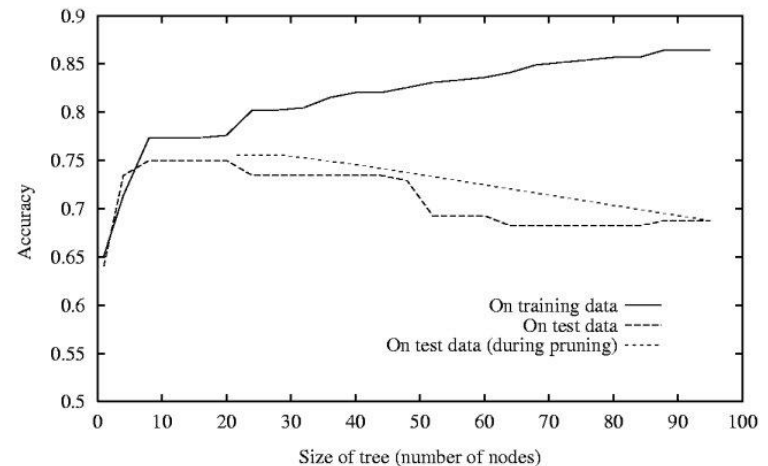


- Celui qui minimise l'entropie moyenne, pondérée

$$- Gain = \sum_{i=1}^n \frac{|S_i|}{|S|} H(S_i)$$

Apprentissage supervisé – Les arbres de décision

- Quelques remarques
 - Un algorithme récursif, qui divise le dataset jusqu'à avoir des sous-ensembles pures
 - Marcherai parfaitement sur les données d'entraînement, mais pas sur les nouvelles données (unseen data)
 - Solution :
 - faire un test de signification statistique à chaque division
 - Utiliser les données de validation
 - » supprimer et merger certains nœuds
 - » voir l'impact sur les données de validation



Apprentissage supervisé – Les arbres de décision

- Quelques remarques
 - Variables quantitative ?
 - utilisation de seuils

Apprentissage supervisé – Les arbres de décision

- Quelques remarques

- Avantages

- Interprétables
 - Ignore naturellement les attributs non importants; n'apportant d'information
 - Rapide à l'exécution $O(\text{profondeur})$

- Limitations

- Peut ne pas trouver le meilleur arbres (car optimisation locale, étape par étape)
 - division se font par un seul attribut

Apprentissage supervisé – Les forêts (Random Forest)

- Objectif:
 - Résoudre les problèmes des arbres de décision
- Comment ?
 - ...d'une manière assez étrange, mais qui marche !
 - Au lieu de se baser est construire un seul arbre, on construit plusieurs arbres (k arbres)
 - Prendre un sous ensemble S des données d'apprentissage
 - Sous ensemble au niveau **des lignes et au niveau des colonnes**
 - Pour chaque sous ensemble, exécuter l'algorithme ID3 pour générer un arbre de décision
 - Lors de la prédiction, exécuter les k arbres de décision. Le résultat est le résultat de la **majorité** (chaque arbre donne un résultat, on prend le résultat majoritaire)

Les arbres de décision en python

- **`sklearn.tree.DecisionTreeClassifier`** : <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

```
In [3]: #Apprentissage sur une colonne
        from sklearn import tree
        clf = tree.DecisionTreeClassifier(criterion='entropy', max_depth=3)

        clf.fit(fl[['PetalWidthCm']], fl.Species)
```

```
Out[3]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=3,
                               max_features=None, max_leaf_nodes=None,
                               min_impurity_decrease=0.0, min_impurity_split=None,
                               min_samples_leaf=1, min_samples_split=2,
                               min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                               splitter='best')
```

Les forêts en python...et autre

- **sklearn.ensemble.RandomForestClassifier**: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>
- **sklearn.naive_bayes.MultinomialNB**: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html#sklearn.naive_bayes.MultinomialNB
- **sklearn.svm.SVC**: <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>
- **sklearn.linear_model.SGDClassifier**: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

```
In [65]: from sklearn.ensemble import RandomForestClassifier
         clfRF300 = RandomForestClassifier(n_estimators=300, max_depth=None, min_samples_split=2, random_state=0).fit(X_train, y_train)

         from sklearn.naive_bayes import MultinomialNB
         clfNB = MultinomialNB()
         clfNB.fit(X_train, y_train)

         from sklearn import svm
         clfSVM1 = svm.SVC(kernel="linear", C=1, probability=True).fit(X_train, y_train)

         from sklearn import neighbors
         clfKNN10Distance = neighbors.KNeighborsClassifier(10, weights='distance').fit(X_train, y_train)

         from sklearn.linear_model import SGDClassifier
         clfSGD = SGDClassifier(loss='log', shuffle=True, learning_rate="optimal", max_iter=100, tol=None).fit(X_train, y_train)
```

Les mesures de performance

- Définitions
 - Faux positif : un individu faussement classifié positif
 - un mail classifié comme spam, mais qui ne l'est pas
 - un patient classifié comme gravement malade alors qu'il est sain
 - Faux négatif: un individu faussement classifié négatif
 - un mail classifié comme non spam alors qu'il est
 - un patient classifié comme sain, alors qu'il est gravement malade
- Selon le cas d'utilisation, on peut vouloir minimiser l'un ou l'autre
 - détection de spam
 - détection de patient sain

Les mesures de performance

- Il existe 3 mesures majeures, qui sont indicatives de la qualité d'un modèle de classification
 - la précision (precision)
 - le rappel (recall)
 - F1

- $$Precision = \frac{Vrai\ positif}{Vrai\ positif + Faux\ positif}$$

- $$Recall = \frac{Vrai\ positif}{Vrai\ positif + Faux\ negatif}$$

- $$F1 = 2 * \frac{Recall * Precision}{Recall + Precision}$$

Les mesures de performance

- Sur quelle données prendre ces mesures ?
Sur les données d'apprentissage ? sur les données de production ?
- Une des solutions est de diviser les données d'apprentissage en données d'apprentissage et données de test :
 - 4/5 Données sur lesquelles on entraine l'algorithme
 - 1/5 Données sur lesquelles on teste le modèle généré par l'algorithme

Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD	
0	1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
1	2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
2	3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
3	4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
4	5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
5	6	56	1	nontypical	120	236	0	0	178	0	0.8	1	0.0	normal	No
6	7	62	0	asymptomatic	140	268	0	2	160	0	3.6	3	2.0	normal	Yes
7	8	57	0	asymptomatic	120	354	0	0	163	1	0.6	1	0.0	normal	No
8	9	63	1	asymptomatic	130	254	0	2	147	0	1.4	2	1.0	reversable	Yes
9	10	53	1	asymptomatic	140	203	1	2	155	1	3.1	3	0.0	reversable	Yes

Les mesures de performance

- Inconvénient ?
- Solution : validation croisée

Les mesure de performance en python

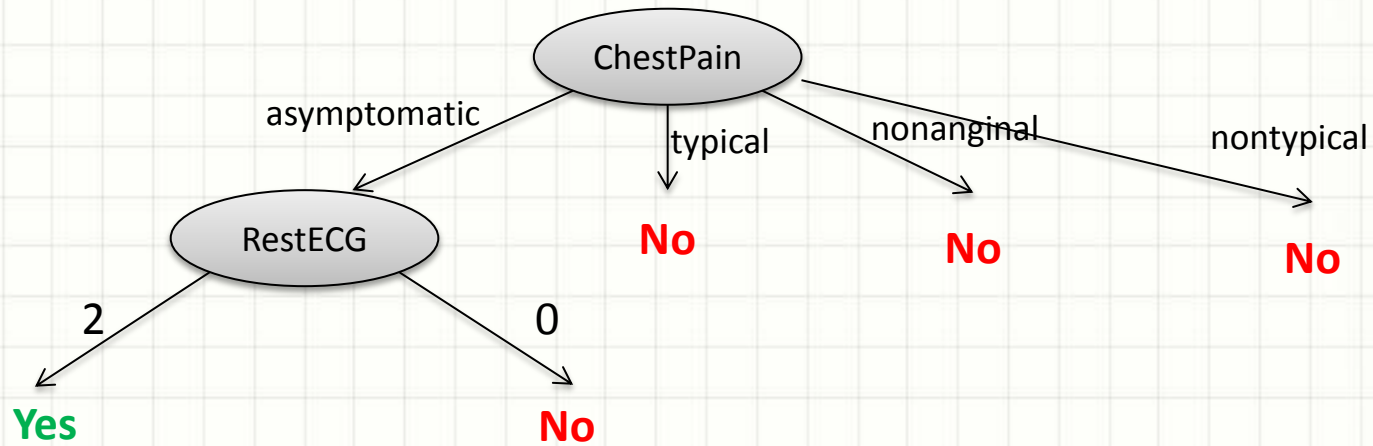
- **data splitting**

- **sklearn.model_selection.train_test_split** : https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

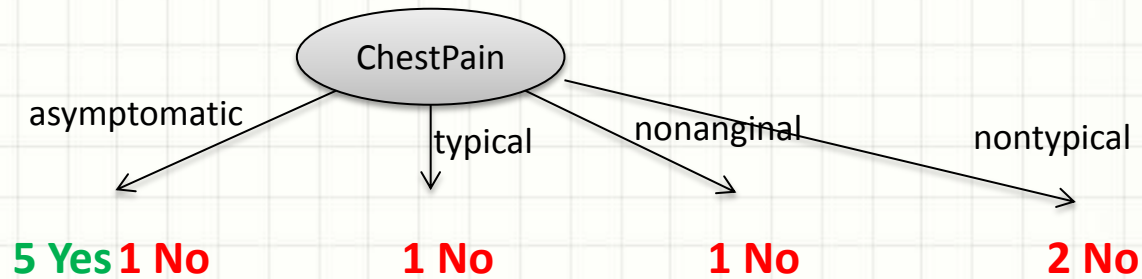
- **Cross-validation**

- **sklearn.model_selection.cross_val_score** : https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html#sklearn.model_selection.cross_val_score

Annexes

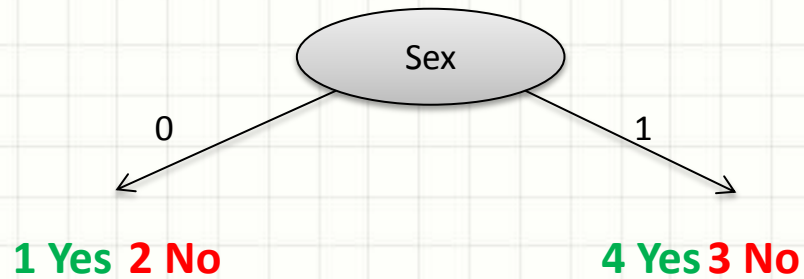


Annexes



	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD	
	0	1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
	1	2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
	2	3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
	3	4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
	4	5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
	5	6	56	1	nontypical	120	236	0	0	178	0	0.8	1	0.0	normal	No
	6	7	62	0	asymptomatic	140	268	0	2	160	0	3.6	3	2.0	normal	Yes
	7	8	57	0	asymptomatic	120	354	0	0	163	1	0.6	1	0.0	normal	No
	8	9	63	1	asymptomatic	130	254	0	2	147	0	1.4	2	1.0	reversable	Yes
	9	10	53	1	asymptomatic	140	203	1	2	155	1	3.1	3	0.0	reversable	Yes

Annexes



Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD	
0	1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
1	2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
2	3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversible	Yes
3	4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
4	5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
5	6	56	1	nontypical	120	236	0	0	178	0	0.8	1	0.0	normal	No
6	7	62	0	asymptomatic	140	268	0	2	160	0	3.6	3	2.0	normal	Yes
7	8	57	0	asymptomatic	120	354	0	0	163	1	0.6	1	0.0	normal	No
8	9	63	1	asymptomatic	130	254	0	2	147	0	1.4	2	1.0	reversible	Yes
9	10	53	1	asymptomatic	140	203	1	2	155	1	3.1	3	0.0	reversible	Yes