

APPRENTISSAGE SUPERVISÉ : THÉORIE ET ALGORITHMES — COURS 2

CES DATA SCIENTIST, TÉLÉCOM PARISTECH

Aurélien Bellet

Inria Lille

March 30, 2016

1. Rappels sur l'apprentissage supervisé

2. Arbres de décision et de régression

Une structure efficace : les arbres

Séparateurs élémentaires

Algorithme CART

3. Sélection et évaluation de modèles

Sélection de modèles

Évaluation de modèles

RAPPELS SUR L'APPRENTISSAGE SUPERVISÉ

- X : variable explicative, vecteur aléatoire dans $\mathcal{X} = \mathbb{R}^p$
- Y : variable à prédire, aléatoire dans $\mathcal{Y} = \{1, \dots, C\}$ (classification) ou $\mathcal{Y} = \mathbb{R}$ (régression)
- P : loi de probabilité jointe de (X, Y) , fixée mais **inconnue**
- $\mathcal{S} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^n$: échantillon i.i.d. tiré selon la loi P
- \mathcal{H} : collection de classifieurs / modèles, $h \in \mathcal{H}$
- L : perte mesurant les erreurs d'un classifieur / modèle
 - Exemple (classification) : $L(y, h(x)) = \begin{cases} 1 & \text{si } h(x) \neq y \\ 0 & \text{sinon} \end{cases}$
 - Exemple (régression) : $L(y, h(x)) = (y - h(x))^2$
- **Objectif** : déterminer à partir de \mathcal{S} la fonction $h \in \mathcal{H}$ qui minimise $R(h) = \mathbb{E}_{(X,Y) \sim P}[L(Y, h(X))]$

Définir :

- l'espace de représentation des données

Définir :

- l'espace de représentation des données
- la classe des classifieurs considérés

Définir :

- l'espace de représentation des données
- la classe des classifieurs considérés
- la **fonction de perte** à minimiser pour obtenir le meilleur classifieur dans cette classe

Définir :

- l'espace de représentation des données
- la classe des classifieurs considérés
- la **fonction de perte** à minimiser pour obtenir le meilleur classifieur dans cette classe
- l'**algorithme de minimisation** de cette fonction de perte

Définir :

- l'espace de représentation des données
- la classe des classifieurs considérés
- la **fonction de perte** à minimiser pour obtenir le meilleur classifieur dans cette classe
- l'**algorithme de minimisation** de cette fonction de perte
- une **méthode de sélection de modèle** pour choisir les hyperparamètres

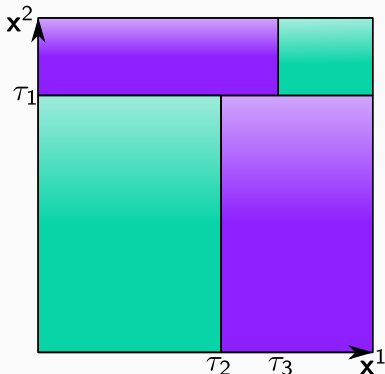
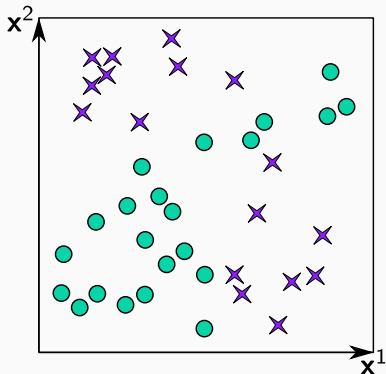
Définir :

- l'espace de représentation des données
- la classe des classifieurs considérés
- la **fonction de perte** à minimiser pour obtenir le meilleur classifieur dans cette classe
- l'**algorithme de minimisation** de cette fonction de perte
- une **méthode de sélection de modèle** pour choisir les hyperparamètres
- une **méthode d'évaluation** des performances

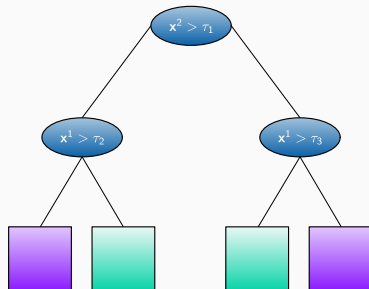
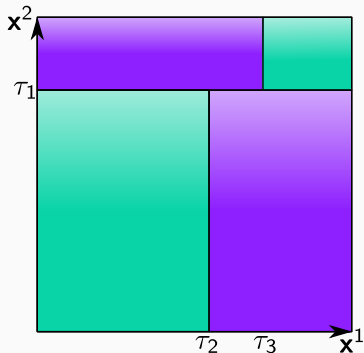
ARBRES DE DÉCISION ET DE RÉGRESSION

ARBRES DE DÉCISION

- Invention quasi simultanée entre 1979 et 1983 par L. Breiman et al. (CART, Berkeley, USA) et R. Quinlan (ID3, Sydney, Australie)
- Dans 2 communautés différentes: en statistique (CART), et dans une discipline nouvelle, le *machine learning* (ID3)



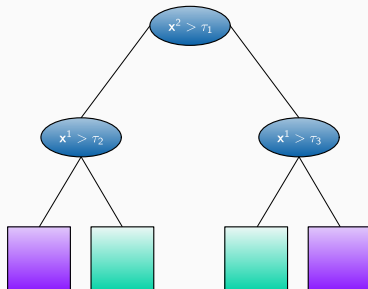
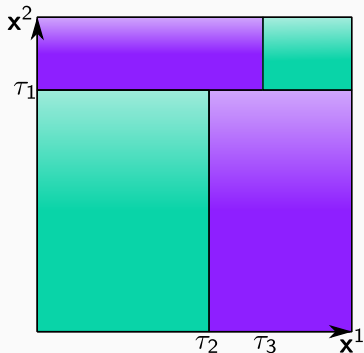
ARBRES DE DÉCISION



Première idée:

Utiliser non pas un, mais plusieurs séparateurs linéaires pour construire des frontières de décision non linéaires

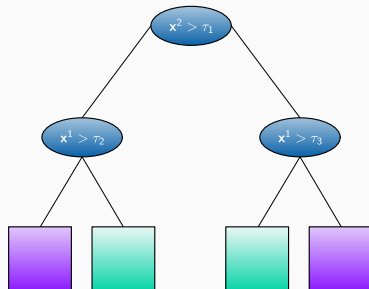
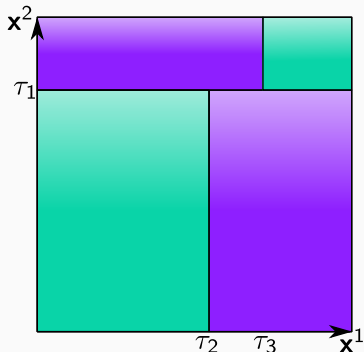
ARBRES DE DÉCISION



Deuxième idée:

Utiliser des séparateurs linéaires orthogonaux aux axes, i.e. des hyperplans $\{x \in \mathcal{X} : x^j = \tau\}$ pour l'interprétabilité / efficacité

ARBRES DE DÉCISION



Troisième idée:

Utiliser un prédicteur représenté par un arbre: chaque noeud est associé à un hyperplan séparateur $\{x \in \mathcal{X} : x^j = \tau\}$; chaque feuille est associée à une fonction constante, i.e. une classe

SÉPARATEUR LINÉAIRE ORTHOGONAL AUX AXES

- Rappel: $x = (x^1, \dots, x^p)$, p variables
- Pour une variable continue : j -ème variable x^j , seuil τ :

$$t_{j,\tau}(x) = \text{sign}(x^j - \tau)$$

- Pour une variable catégorielle à M modalités $\{v_1^j, \dots, v_M^j\}$:

$$t_{j,v,m}(x) = \mathbb{1}(x^j = v_m^j)$$

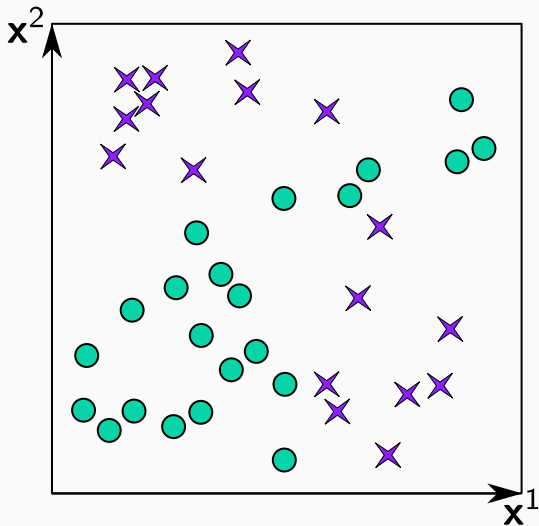
- L'arbre final encode un ensemble de règles logiques de type:
"si $(x^{j_1} > \tau_1)$ et $(x^{j_2} \leq \tau_2)$ et ... alors x est de la classe k "

ALGORITHME RÉCURSIF DE CONSTRUCTION (CART)

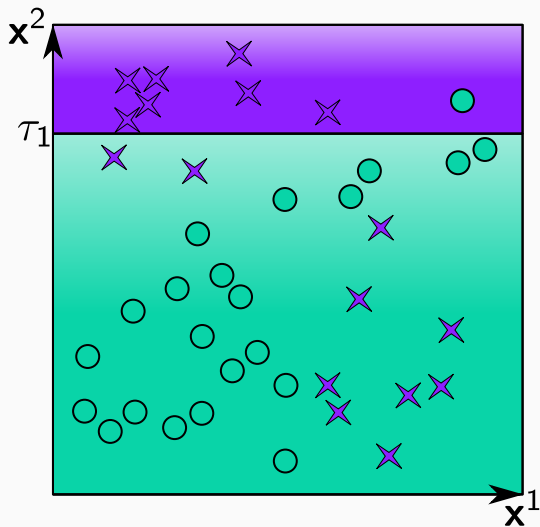
Cas d'un arbre binaire, avec \mathcal{S} ensemble d'apprentissage

1. Construire un noeud racine
2. Chercher la meilleure séparation $t : \mathcal{X} \rightarrow \{0, 1\}$ à appliquer sur \mathcal{S} telle que le coût local $L(t, \mathcal{S})$ soit minimal
3. Associer le séparateur choisi au noeud courant et séparer l'ensemble d'apprentissage courant \mathcal{S} en \mathcal{S}^d et \mathcal{S}^g à l'aide de ce séparateur
4. Construire un noeud à droite et un noeud à gauche
5. Mesurer le critère d'arrêt à droite, s'il est vérifié, le noeud droit devient une feuille sinon aller à l'étape 2 avec \mathcal{S}^d comme ensemble courant
6. Mesurer le critère d'arrêt à gauche, s'il est vérifié, le noeud gauche devient une feuille sinon aller à l'étape 2 avec \mathcal{S}^g comme ensemble courant

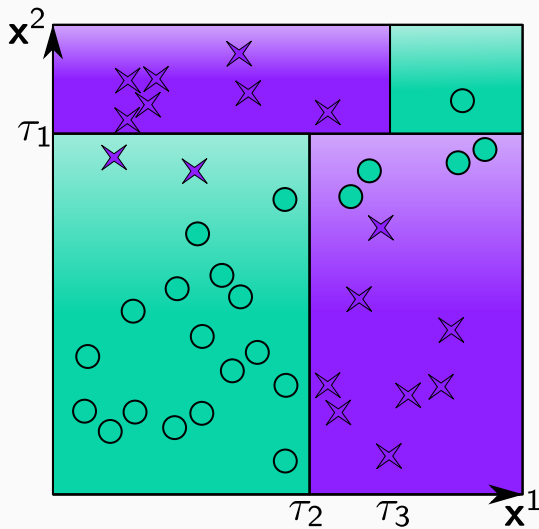
EXEMPLE VISUEL



EXEMPLE VISUEL



EXEMPLE VISUEL



Pour un ensemble d'exemples d'apprentissage \mathcal{S} et une fonction de séparation binaire $t_{j,\tau}$ avec $t_{j,\tau}(x) = \text{sign}(x^j - \tau)$, notons

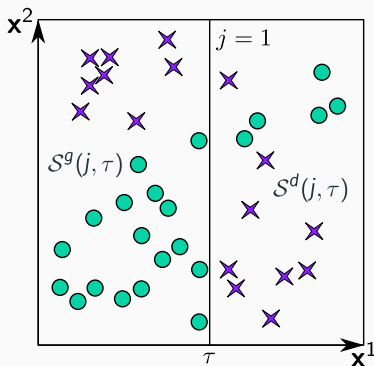
$$\mathcal{S}^d(j, \tau) = \{(x, y) \in \mathcal{S}, t_{j,\tau}(x) > 0\}$$

$$\mathcal{S}^g(j, \tau) = \{(x, y) \in \mathcal{S}, t_{j,\tau}(x) \leq 0\}$$

Pour un ensemble d'exemples d'apprentissage \mathcal{S} et une fonction de séparation binaire $t_{j,\tau}$ avec $t_{j,\tau}(x) = \text{sign}(x^j - \tau)$, notons

$$\mathcal{S}^d(j, \tau) = \{(x, y) \in \mathcal{S}, t_{j,\tau}(x) > 0\}$$

$$\mathcal{S}^g(j, \tau) = \{(x, y) \in \mathcal{S}, t_{j,\tau}(x) \leq 0\}$$

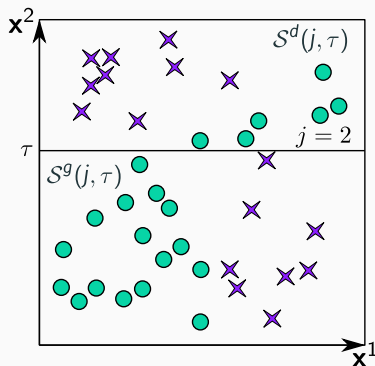
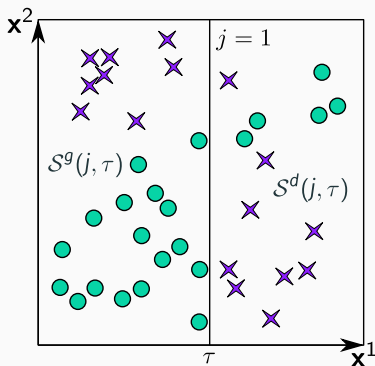


COUPURE

Pour un ensemble d'exemples d'apprentissage \mathcal{S} et une fonction de séparation binaire $t_{j,\tau}$ avec $t_{j,\tau}(x) = \text{sign}(x^j - \tau)$, notons

$$\mathcal{S}^d(j, \tau) = \{(x, y) \in \mathcal{S}, t_{j,\tau}(x) > 0\}$$

$$\mathcal{S}^g(j, \tau) = \{(x, y) \in \mathcal{S}, t_{j,\tau}(x) \leq 0\}$$



- Parmi tous les paramètres $(j, \tau) \in \{1, \dots, p\} \times \{\tau_1, \dots, \tau_r\}$, on cherche \hat{j} et $\hat{\tau}$ qui minimisent une fonction de coût :

$$L(t_{j,\tau}, \mathcal{S}) = \frac{n_g}{n} H(\mathcal{S}^g(j, \tau)) + \frac{n_d}{n} H(\mathcal{S}^d(j, \tau))$$
$$n_g = |\mathcal{S}^g(j, \tau)| \quad \text{et} \quad n_d = |\mathcal{S}^d(j, \tau)|$$

- H est une fonction “d’impureté”
- Le coût total est la somme de l’impureté de chaque sous partie, pondérée par le nombre d’échantillons

CRITÈRES DE COÛT POUR LA CLASSIFICATION

- Pour un ensemble \mathcal{S} de n exemples étiquetés, on définit

$$p_c(\mathcal{S}) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}(y_i = c)$$

- Entropie

$$H(\mathcal{S}) = - \sum_{c=1}^C p_c(\mathcal{S}) \log p_c(\mathcal{S})$$

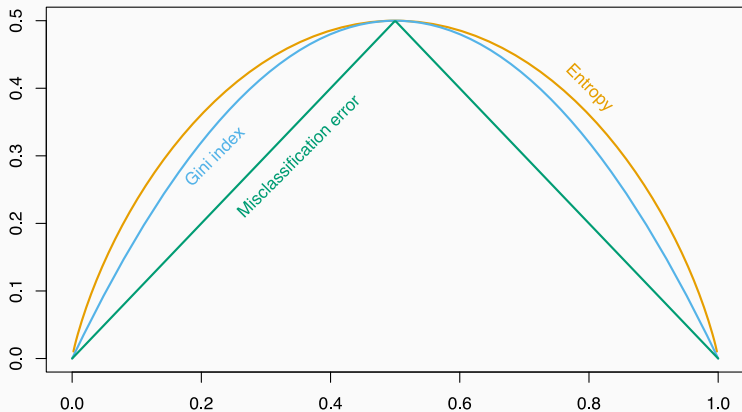
- Index de Gini

$$H(\mathcal{S}) = \sum_{c=1}^C p_c(\mathcal{S})(1 - p_c(\mathcal{S}))$$

- Erreur de classification

$$H(\mathcal{S}) = 1 - p_{c(\mathcal{S})}(\mathcal{S}), \quad c(\mathcal{S}) : \text{classe majoritaire dans } \mathcal{S}$$

CRITÈRES DE COÛT POUR LA CLASSIFICATION



(Hastie, Tibshirani & Friedman, 2009)

- On peut s'arrêter localement (dans une branche), dès qu'on atteint :
 - la profondeur maximale
 - le nombre maximale de feuilles
 - le nombre minimal d'exemples dans un noeud
- Remarque : si le nombre minimal d'exemples est 1, l'ensemble d'apprentissage est appris jusqu'au bout (dans les limites computationnelles et de mémoire) → **risque de sur-apprentissage !**
- Ces hyperparamètres sont à déterminer par **validation croisée** (voir plus loin)

Le fonctionnement pour la régression est pratiquement identique, pour construire l'arbre, seul le critère de coût change: on minimise

$$L(t_{j,\tau}, \mathcal{S}) = \frac{n_g}{n} H(\mathcal{S}^g(j, \tau)) + \frac{n_d}{n} H(\mathcal{S}^d(j, \tau))$$

avec la variance comme mesure d'impureté

$$H(\mathcal{S}) = \text{Var}(\mathcal{S}) := \frac{1}{|\mathcal{S}|} \sum_{(x_i, y_i) \in \mathcal{S}} (y_i - \bar{y}_n)^2$$

où

$$\bar{y}_n := \frac{1}{|\mathcal{S}|} \sum_{(x_i, y_i) \in \mathcal{S}} y_i$$

Avantages

- Construit une fonction de décision non linéaire, interprétable
- Fonctionne pour le multi-classe
- Prise de décision efficace: $O(\log F)$ (F : nombre de feuilles)
- Fonctionne pour des variables continues et catégorielles

Inconvénients

- Pas d'optimisation globale
- Estimateur à large variance, instabilité : une petite variation dans l'ensemble d'apprentissage engendre un arbre complètement différent

RÉDUIRE LA VARIANCE : LE BAGGING

- Il s'agit d'une **méthode ensembliste** : on combine les prédictions de plusieurs classifieurs
- Ici on va combiner les prédictions de B arbres
 - cas de la classification : vote majoritaire
 - cas de la régression : moyenne des prédictions
- Pour améliorer les performances, il faut générer de la **diversité**
- Première idée : le **bagging**. Pour $b \in \{1, \dots, B\}$:
 - On construit \mathcal{S}_b de même taille que \mathcal{S} par tirage **avec** remise dans \mathcal{S} (on dit que \mathcal{S}_b est un échantillon *bootstrap*)
 - Pour obtenir l'arbre b , on applique l'algorithme CART sur \mathcal{S}_b

RÉDUIRE LA VARIANCE : LES FORÊTS ALÉATOIRES

- Une limitation possible du bagging appliqué aux arbres est que l'on risque de souvent sélectionner les mêmes variables dans les différents arbres, malgré les variations d'échantillon
- Comment introduire davantage de diversité ?
- Deuxième idée : les **forêts aléatoires**. Pour $b \in \{1, \dots, B\}$:
 - On construit \mathcal{S}_b de même taille que \mathcal{S} par tirage **avec** remise dans \mathcal{S} (on dit que \mathcal{S}_b est un échantillon *bootstrap*)
 - Pour obtenir l'arbre b , on applique une version modifiée de CART sur \mathcal{S}_b : à chaque noeud, on choisit la coupure parmi $k < p$ variables tirées aléatoirement
- Un ordre de grandeur raisonnable est de prendre $k = O(\sqrt{p})$
- Très efficace et très utilisé en pratique

SÉLECTION ET ÉVALUATION DE MODÈLES

- Construire une fonction (classifieur, modèle) \hat{h} de \mathbb{R}^p vers $\{-1, 1\}$ (ou $\{1 \dots, C\}$) telle que

$$\hat{h} = \arg \min_{h \in \mathcal{H}} \sum_{i=1}^n L(y_i, h(x_i)) + \lambda \Omega(h)$$

- L est une **fonction de perte** : mesure à quel point $h(x_i)$ est proche de y_i (la valeur de sortie désirée)
 - $\sum_{i=1}^n L(y_i, h(x_i))$: terme d'attache aux données
 - $\Omega(h)$: pénalité sur la complexité du modèle h , par exemple la norme au carré du vecteur de paramètres de h
- Note : l'approche régularisée n'est pas la seule approche possible, mais c'est la plus courante !

SÉLECTION OU ÉVALUATION DE MODÈLE ?

- **Sélection de modèle** : estimer les performances de différents modèles afin de choisir le meilleur hyperparamètre
- **Évaluation de modèle** : ayant appris un modèle (pour un certain choix d'hyperparamètres), estimer ses performances
- Note : on choisit le même critère de performance (ex : erreur de prédiction) pour les deux étapes
- Dans la suite, on va se concentrer sur ces deux questions

SÉLECTION DE MODÈLES : PREMIER EXEMPLE

- On souhaite apprendre un classifieur linéaire dans le plan

$$h_{\beta}(x) = \text{sign}(\beta_1 x_1 + \beta_2 x_2 + \beta_0)$$

- Formulation du problème d'apprentissage

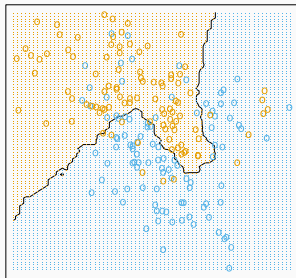
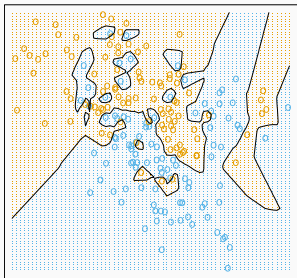
$$\arg \min_{\beta_0, \beta_1, \beta_2 \in \mathbb{R}} \sum_{i=1}^n L(y_i, h_{\beta}(x_i)) + \lambda \|\beta\|$$

avec $\|\beta\| = \|\beta\|_2^2$ ou $\|\beta\| = \|\beta\|_1$

- Quelle valeur de λ choisir ?

SÉLECTION DE MODÈLES : DEUXIÈME EXEMPLE

- On souhaite utiliser un classifieur des K -plus-proches-voisins
- **Comment choisir K ?**
 - K trop petit $\rightarrow h$ trop sensible aux données (variance importante)
 - K trop grand : $\rightarrow h$ peu sensible aux données (biais important)



(Hastie, Tibshirani & Friedman, 2009)

DÉCOMPOSITION BIAIS - VARIANCE

- On suppose $Y = f(X) + \epsilon$ avec ϵ centré et de variance σ_ϵ^2
- Soit \hat{f} la fonction apprise à partir de \mathcal{S} , l'échantillon d'apprentissage
- Erreur quadratique (régression) espérée après apprentissage:

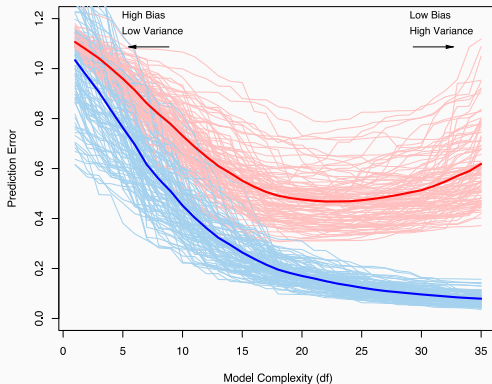
$$\begin{aligned}\mathbb{E}_{X,Y,\mathcal{S}}[(Y - \hat{f}(X))^2] &= \mathbb{E}_X \left[\mathbb{E}_{Y|X,\mathcal{S}}[(Y - \hat{f}(X))^2] \right] \\ &= \mathbb{E}_X \left[\mathbb{E}_{Y|X,\mathcal{S}}(Y - f(X))^2 + \mathbb{E}_{Y|X,\mathcal{S}}[(\bar{f}(X) - \hat{f}(X))^2] + \mathbb{E}_{Y|X}[(f(X) - \bar{f})^2] \right] \\ &= \sigma_\epsilon^2 + \mathbb{E}_X [\text{Var}_{\mathcal{S}}[\hat{f}(X)] + \text{Biais}_{\mathcal{S}}^2[\hat{f}(X)]]\end{aligned}$$

avec $\bar{f}(X) = \mathbb{E}_{\mathcal{S}} \hat{f}(X)$

- **Bruit des données** : terme incompressible
- **Biais au carré** : mesure à quel point \hat{f} est loin de la cible
- **Variance de $\hat{f}(X)$** : mesure à quel point \hat{f} est sensible aux données d'apprentissage

DÉCOMPOSITION BIAIS - VARIANCE

- Faisons varier la complexité du modèle



(Hastie, Tibshirani & Friedman, 2009)

- Dépend aussi de la taille n de l'échantillon d'apprentissage \mathcal{S}

- Posons $X = x_0$ pour que l'aléa ne vienne pas de x_0 . On a

$$\mathbb{E}[(Y - \hat{f}(x_0))^2] = \text{Var}[Y] + \text{Biais}^2[f(x_0)] + \text{Var}(\hat{f}(x_0))$$

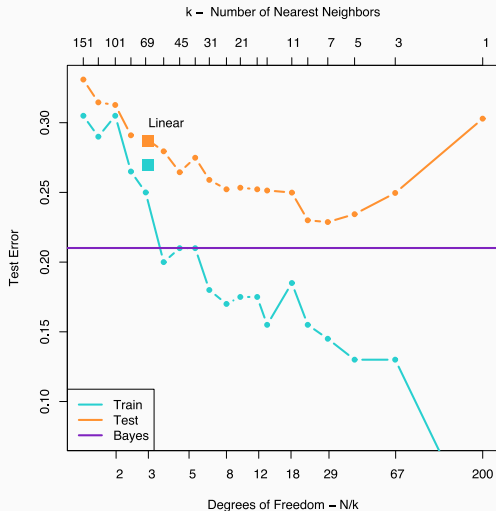
- On fixe \mathcal{S} , donc $\hat{f}(x_0)$ est déterministe

$$\mathbb{E}[(Y - \hat{f}(x_0))^2] = \sigma_\epsilon^2 + \left(f(x_0) - \frac{1}{K} \sum_{\ell=1}^K f(x_{(\ell)}) \right)^2 + \frac{\sigma_\epsilon^2}{K}$$

- **Dilemme biais-variance:**
 - K contrôle le terme de variance : plus K est grand, plus elle décroît
 - K contrôle aussi le biais : plus K est petit, plus celui-ci est faible
- Le choix de K est donc primordial

DÉCOMPOSITION BIAIS - VARIANCE : K-PPV

Erreur de test en fonction de n/K



(Hastie, Tibshirani & Friedman, 2009)

- Le classifieur K -PPV est “paresseux” : pas besoin d’algorithme d’apprentissage !
- On a seulement besoin de \mathcal{S} , d’une distance et d’une valeur de K
- **Questions**
 - Comment choisir K ?
 - Ayant choisi K , comment estimer l’erreur en généralisation de ce classifieur ?

- Stratégie classique : partager les données disponibles en 3 sous-échantillons
 - **Apprentissage** : données pour entraîner les modèles
 - **Validation** : données pour sélectionner les hyperparamètres
 - **Test** : données pour estimer l'erreur en généralisation du modèle
- Les hyperparamètres sont pris dans une grille finie
- Exemple pour les K -PPV : on sélectionne la valeur de K qui donne l'erreur la plus faible sur l'ensemble de validation
- Comment faire pour “gaspiller” moins de données pour la validation ?

SÉLECTION DE MODÈLE : VALIDATION CROISÉE

1. Créer seulement 2 sous-échantillons : **apprentissage** et **test**
2. Diviser les données \mathcal{S}_{app} en B parties de même taille (approximativement) et disjointes $\mathcal{S}_{app}^1, \dots, \mathcal{S}_{app}^B$
3. Pour $b \in \{1, \dots, B\}$
 - Entraîner sur toutes les données de \mathcal{S}_{app} **sauf** \mathcal{S}_{app}^b pour obtenir un modèle \hat{h}_λ^b
 - Calculer le risque empirique sur les données restantes \mathcal{S}_{app}^b

$$R^b(\lambda) = \frac{1}{|\mathcal{S}_{app}^b|} \sum_{i \in \mathcal{S}_{app}^b} L(x_i, y_i, \hat{h}_\lambda^b)$$

4. Risque estimé par validation croisée

$$R_{CV}^B(\lambda) = \frac{1}{B} \sum_{b=1}^B R^b(\lambda)$$

- Répéter la procédure pour tous les λ dans la grille Λ et choisir

$$\hat{\lambda}_{CV}^B = \arg \min_{\lambda \in \Lambda} R_{CV}^B(\lambda)$$

- R_{app} nous dit à quel point le classifieur a bien réussi à approcher les données d'apprentissage
- R_{CV}^B nous dit à quelle erreur en généralisation nous attendre en apprenant sur un ensemble de taille $|\mathcal{S}_{app}| - |\mathcal{S}_{app}|/B$
- R_{test} nous dit à quel point le classifieur réussit à approcher des données nouvelles

- On choisit en général comme critère d'évaluation
 - l'erreur quadratique pour la régression
 - l'erreur de prédiction (0-1) pour la classification
- Cependant pour la classification binaire, l'erreur 0-1 est parfois insuffisante
 - classes très déséquilibrées
 - besoin de détails sur la nature des erreurs (faux positifs, faux négatifs, etc)
- On se tourne alors vers la **courbe ROC**, ainsi que son résumé : l'aire sous la courbe ROC

	Prédit OUI	Prédit NON
POS	Vrais positifs	Faux négatifs
NEG	Faux positifs	Vrais négatifs

- TPR : taux de vrais positifs

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

- FPR : taux de faux positifs

$$FPR = \frac{FP}{FP + TN}$$

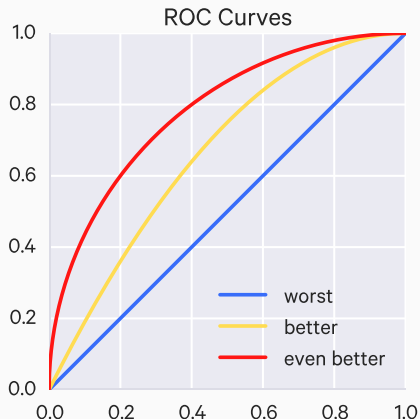
- Soit h un classifieur défini par

$$h(x) = \text{sign}(f(x) - s)$$

- Habituellement, en classification binaire, $s = 0.5$ si les sorties sont dans $\{0, 1\}$, ou $s = 0$ si dans $\{-1, 1\}$
- Le choix du seuil s fait varier TPR et FPR
- Tracer la courbe ROC consiste à faire varier s et à reporter le point $(FPR(s), TPR(s))$ sur un graphique 2D
- On associe ainsi plusieurs points à une même fonction f issue d'un algorithme d'apprentissage

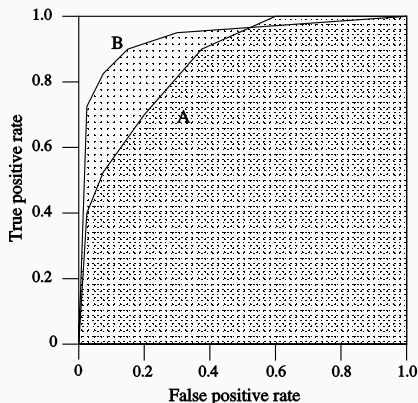
COMPARER DES CLASSIFIEURS AVEC UNE COURBE ROC

- Sur un ensemble test, pour une fonction f donnée, on mesure $(FPR(s), TPR(s))$ en faisant varier s



AIRE SOUS LA COURBE ROC

- L'aire sous la courbe ROC (AUC) est un résumé de la courbe ROC
 - Compris entre 0 (pire) and 1 (meilleur)
 - Indicateur très utilisé de la qualité d'un classifieur



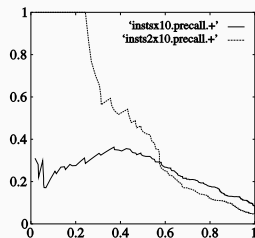
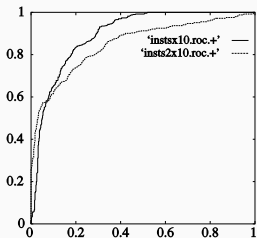
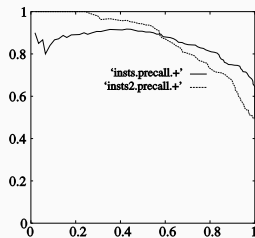
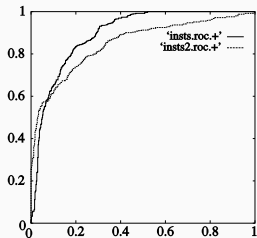
D'AUTRES MESURES POUR ALLER PLUS LOIN

		<u>True class</u>			
		p	n		
<u>Hypothesized class</u>	Y	True Positives	False Positives	$fp\ rate = \frac{FP}{N}$	$tp\ rate = \frac{TP}{P}$
	N	False Negatives	True Negatives	$precision = \frac{TP}{TP+FP}$	$recall = \frac{TP}{P}$
Column totals:		P	N	$accuracy = \frac{TP+TN}{P+N}$	
				$F\text{-measure} = \frac{2}{1/precision + 1/recall}$	

- En recherche d'information, on cherche à bien prédire mais aussi à ne pas manquer d'information pertinente
- Exemple : un moteur de recherche veut retourner une liste de documents pertinents la plus exhaustive possible
- On s'intéresse alors aux courbes Précision-Rappel

EXEMPLES DE COURBES ROC ET PR

- Les courbes ROC sont insensibles au déséquilibre de classes, alors que les courbes PR le sont



- Arbres de décision et de régression
 - Chapitre 9 de **Elements of Statistical Learning** (Hastie, Tibshirani & Friedman, 2009)
 - Article : **Classification and regression trees** (L. Breiman et al., Wadsworth Statistics/Probability Series 1984)
 - Article : **Induction of decision trees** (J. R. Quinlan, Machine Learning 1986)
 - Article : **Bagging predictors** (L. Breiman, Machine Learning 1996)
 - Article : **Random Forests** (L. Breiman, Machine Learning 2001)
- Sélection et évaluation de modèles
 - Chapitres 4 et 7 de **Elements of Statistical Learning** (Hastie, Tibshirani & Friedman, 2009)
 - Article : **Introduction to ROC analysis** (T. Fawcett, Pattern Recognition Letters 2006)