Statistiques avancées — Régression
Régression en grande dimension

Geneviève Robin

Régression en grande dimension

Objectif du cours

- On reprend le modèle de régression linéaire multiple, mais...
- Dans le contexte de la grande dimension que l'on va expliciter
- Cela correspond au contexte d'une partie des données modernes
- "Big data" dans un certain sens

Rappel de régression linéaire

- Échantillon $(X_i, Y_i)_{1\leq i\leq n}$ avec $X_i \in \mathbb{R}^p$ un vecteur de covariables (prédicteurs) et $Y_i \in \mathbb{R}$ la réponse.
- Modèle linéaire avec bruit Gaussien additif :

  $$Y_i = X_i\beta + \varepsilon_i,$$

  où $\beta \in \mathbb{R}^p$ est le vecteur de coefficients de régression inconnu et $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.
- Dans les cours précédents on a vu l'estimateur du maximum de vraisemblance/des moindres carrés.

Estimateur des moindres carrés classiques — Rappel

- MLE/OLS: $\hat{\beta} \in \mathrm{argmin}_{\beta \in \mathbb{R}^p} \|Y - X\beta\|^2$, $Y \in \mathbb{R}^n$ vecteur de réponse, $X \in \mathbb{R}^{n \times p}$ matrice de design
- Rappel: **Si** $X^\top X$ **est inversible** alors $\hat{\beta}$ admet la forme close

$$\hat{\beta} = (X^\top X)^{-1} X^\top Y.$$

- $X^\top X$ inversible $\Leftrightarrow$ $X^\top X$ de rang plein (de rang p).
- **Si** $p > n$ **cette hypothèse ne peut être vérifiée.**

Qu'est-ce que la grande dimension?

- En sciences des données, on parle de "grande dimension" lorsque $p >> n$.
- Dans ce cas, l'estimateur des moindres carrés est mal spécifié : n équations à p inconnues, $p >> n$.
- De plus, l'erreur d'estimation/de prédiction peut devenir <u>grande</u> (voir TD de cet après-midi).
- Pour y remédier, une solution classique est de recourir à la <u>régularisation/pénalisation</u>.

Pénalisations

Pénalisation

En définissant

$$\hat{w}, \hat{b} \in \underset{w \in \mathbb{R}^d, b \in \mathbb{R}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle + b)$$

on définit en général un mauvais classifieur, notamment dans les cas où il y a beaucoup de features.

On considère plutôt

$$\hat{w}, \hat{b} \in \underset{w \in \mathbb{R}^d, b \in \mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle + b) + \frac{1}{C} \operatorname{pen}(w) \right\}$$

où

- pen est un terme de **penalisation**, ça permettra à w ne pas pas être trop "complexe"
- $C > 0$ est un paramètre qui contrôle la force de la pénalisation (appelé paramètre de **tuning** ou de **smoothing**)

Pénalisation ridge

La pénalisation **ridge** est définie par

$$\text{pen}(w) = \frac{1}{2}\|w\|_2^2 = \frac{1}{2}\sum_{j=1}^{d} w_j^2$$

Elle pénalise la taille de w.

- C'est simple
- Elle permet de "régler" les problèmes de corrélation entre variables
- Elle aide par ailleurs les algorithmes d'optimisation (problème plus simple)

Interprétation géométrique

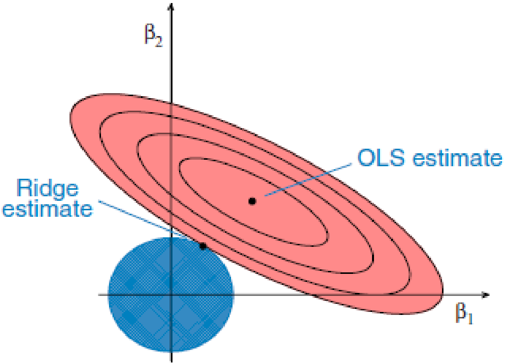


Figure 1: from https://online.stat.psu.edu/stat508/

Sparsité

On remarque que, si $\hat{w}_j = 0$, alors la feature j n'a pas d'impact sur la prédiction

$$\hat{y} = \text{sign}(\langle x, \hat{w} \rangle + \hat{b})$$

Si on a beaucoup de features (si d est grand), on aimerait obtenir un $\hat{w}$ qui contient beaucoup de **zeros**.

On obtientra alors un modèle plus simple avec une dimension "réduite" et donc plus facilement interprétable

Comment faire ?

Pénalisation par la norme $\| \cdot \|_0$

On aimerait définir

$$\hat{w}, \hat{b} \in \underset{w \in \mathbb{R}^d, b \in \mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle + b) + \frac{1}{C} \|w\|_0 \right\},$$

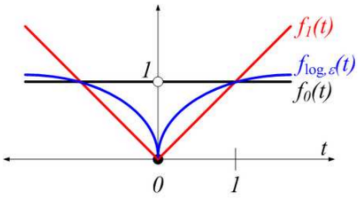
où

$$\|w\|_0 = \#\{j \in \{1, \ldots, d\} : w_j \neq 0\}.$$

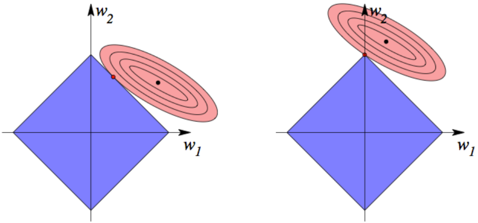
Mais, pour résoudre le problème de minimisation qui n'est pas convexe, il faudrait explorer tous les supports possibles de w : c'est trop long (NP-hard)

Pénalisation par la norme $\|\cdot\|_1$: le LASSO

Une solution est donc de trouver un "proxy" convexe de la $\|\cdot\|_0$: la **norme** ℓ_1
$\|w\|_1 = \sum_{j=1}^{d} |w_j|$



Pourquoi cela induit-il de la sparsité ?



LASSO vs ridge : interprétation géométrique

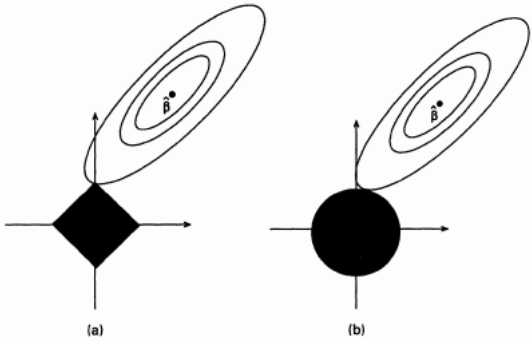


Fig. 2. Estimation picture for (a) the lasso and (b) ridge regression

Régression pénalisée

Considérons le problème de minimisation

$$\hat{w}, \hat{b} \in \underset{w \in \mathbb{R}^d, b \in \mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle + b) + \frac{1}{C} \operatorname{pen}(w) \right\},$$

Pour $\ell(y, y') = \frac{1}{2}(y - y')^2$ et $\operatorname{pen}(w) = \frac{1}{2}\|w\|_2^2$, c'est la **régression ridge**

Pour $\ell(y, y') = \frac{1}{2}(y - y')^2$ et $\operatorname{pen}(w) = \|w\|_1$, c'est le **Lasso** (Least absolute shrinkage and selection operator)

Pour $\ell(y, y') = \log(1 + e^{-yu'})$ et $\operatorname{pen}(w) = \|w\|_1$, c'est la **régression logistique pénalisée ℓ_1**

Il y a de nombreuses combinaisons possibles

Elastic-net

Les combinaisons

$$(\text{régression linéaire ou logistique}) + (\text{ridge or } \ell_1)$$

sont les plus utilisées.

Une autre pénalité très utilisée est

$$\mathrm{pen}(w) = \frac{1-\alpha}{2}\|w\|_2^2 + \alpha\|w\|_1$$

appelée **elastic-net**, elle bénéfie des avantages des pénalisations ridge et ℓ_1 ($\alpha \geq 0$ équilibre les deux)

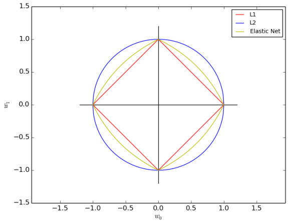


Figure 2: http://scikit-learn.sourceforge.net/

Descente de gradient proximale

Problème de minimisation

Nous avons vu des problèmes de minimisation de la forme

$$\operatorname*{argmin}_{w \in \mathbb{R}^d} f(w) + g(w)$$

où f est une fonction de goodness-of-fit

$$f(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle w, x_i \rangle)$$

où ℓ est une fonction de perte et

$$g(w) = \frac{1}{C} \operatorname{pen}(w)$$

où $\operatorname{pen}(\cdot)$ est une pénalisation, par exemple $\operatorname{pen}(w) = \frac{1}{2}\|w\|_2^2$ (ridge) et $\operatorname{pen}(w) = \|w\|_1$ (Lasso)

Remarque : on oublie dans cette partie le paramètre b.

Gradient et hessienne

On veut minimiser

$$F(w) = f(w) + g(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle) + \frac{1}{C} \operatorname{pen}(w)$$

Calculons le gradient et la hessienne de f

$$\nabla f(w) = \frac{1}{n} \sum_{i=1}^{n} \ell'(y_i, \langle x_i, w \rangle) x_i$$

$$\nabla^2 f(w) = \frac{1}{n} \sum_{i=1}^{n} \ell''(y_i, \langle x_i, w \rangle) x_i x_i^\top$$

avec

$$\ell'(y, y') = \frac{\partial \ell'(y, y')}{\partial y'} \quad \text{et} \quad \ell''(y, y') = \frac{\partial^2 \ell'(y, y')}{\partial y'^2}$$

Convexité et L-régularité

Remarquons que f est convexe si et seulement si

$$y' \mapsto \ell(y_i, y')$$

l'est pour tout $i = 1, \dots, n$.

Definition. On dit f est ***L*-régulière** si elle est continuement différentiable et si

$$\|\nabla f(w) - \nabla f(w')\|_2 \leq L\|w - w'\|_2 \quad \text{pour tout } w, w' \in \mathbb{R}^d$$

Si f est deux fois différentiable, c'est équivalent à supposer

$$\lambda_{\max}(\nabla^2 f(w)) \leq L \quad \text{for any } w \in \mathbb{R}^d$$

(la plus grande valeur propre de la hessienne de f est plus petite que L)

Cas particuliers : perte des moindres carrés

Pour la perte des moindres carrés (least-squares loss)

$$\nabla f(w) = \frac{1}{n}\sum_{i=1}^{n}(\langle x_i, w\rangle - y_i)x_i, \quad \nabla^2 f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i {x_i}^\top$$

donc

$$L = \frac{1}{n}\lambda_{\max}\left(\sum_{i=1}^{n} x_i {x_i}^\top\right)$$

Cas particuliers : perte logistique

Pour la perte logistique

$$\nabla f(w) = \frac{1}{n} \sum_{i=1}^{n} y_i(\sigma(y_i \langle x_i, w \rangle) - 1) x_i$$

et

$$\nabla^2 f(w) = \frac{1}{n} \sum_{i=1}^{n} \sigma(y_i \langle x_i, w \rangle)(1 - \sigma(y_i \langle x_i, w \rangle)) x_i {x_i}^\top$$

donc

$$L = \frac{1}{4n} \lambda_{\max} \Big(\sum_{i=1}^{n} x_i {x_i}^\top \Big)$$

Lemme de descente

Lemme de descente

Si f est L-régulière, alors

$$f(w) \leq f(w') + \langle \nabla f(w'), w - w' \rangle + \frac{L}{2} \|w - w'\|_2^2$$

pour tout $w, w' \in \mathbb{R}^d$

Preuve dans le cours d'optimisation

On a donc, autour du point w^k à l'itération k

$$f(w) \leq f(w^k) + \langle \nabla f(w^k), w - w^k \rangle + \frac{L}{2} \|w - w^k\|_2^2$$

pour tout $w \in \mathbb{R}^d$

Descente de gradient proximale

En considérant le problème de départ, on a donc à l'itération k

$$f(w) + g(w) \leq f(w^k) + \langle \nabla f(w^k), w - w^k \rangle + \frac{L}{2}\|w - w^k\|_2^2 + g(w)$$

et

$$\begin{aligned}
&\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \left\{ f(w^k) + \langle \nabla f(w^k), w - w^k \rangle + \frac{L}{2}\|w - w^k\|_2^2 + g(w) \right\} \\
&\quad = \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \left\{ \frac{L}{2} \left\| w - \left(w^k - \frac{1}{L}\nabla f(w^k) \right) \right\|_2^2 + g(w) \right\} \\
&\quad = \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| w - \left(w^k - \frac{1}{L}\nabla f(w^k) \right) \right\|_2^2 + \frac{1}{L} g(w) \right\} \\
&\quad = \; ????
\end{aligned}$$

Opérateur proximal

Pour tout $g : \mathbb{R}^d \to \mathbb{R}$ convexe (pas forcément différentiable), et tout $w \in \mathbb{R}^d$, on définit

$$\text{prox}_g(w) = \underset{w' \in \mathbb{R}^d}{\text{argmin}} \left\{ \frac{1}{2} \|w - w'\|_2^2 + g(w') \right\}$$

Prox de la pénalité ridge

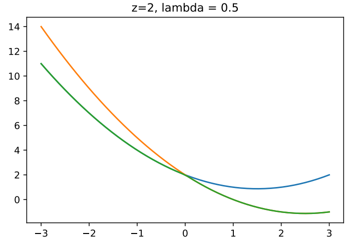
Calculer l'opérateur proximal de la pénalité ridge.

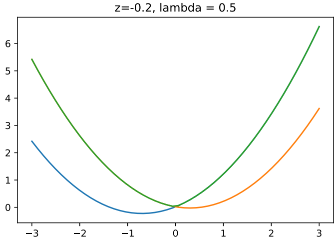
Calcul direct du prox du LASSO (1)

Considérons le problème de minimisation

$$\min_{z' \in \mathbb{R}} \frac{1}{2}(z' - z)^2 + \lambda |z'|$$

pour $\lambda > 0$ et $z \in \mathbb{R}$.





Calcul direct du prox du LASSO (2)

- La dérivée sur $\mathbb{R}++^{\star}$: $z' - z + \lambda$, en $0+$ $d_+ = -z + \lambda$
- La dérivée en $\mathbb{R}_+^{\star}$: $z' - z - \lambda$, en $0-$ $d_- = -z - \lambda$

Soit z_* la solution, elle vérifie

- $z_* = 0$ ssi $d_+ \geq 0$ et $d_- \leq 0$, soit $|z| \leq \lambda$
- $z_* \geq 0$ ssi $d_+ \leq 0$, soit $z \geq \lambda$ et $z_* = z - \lambda$
- $z_* \leq 0$ ssi $d_- \geq 0$, soit $z \leq -\lambda$ et $z_* = z + \lambda$

donc

$$z_* = \mathrm{sign}(z)(|z| - \lambda)_+$$

On l'appelle **l'opérateur de seuillage doux** (soft-thresholding operator).

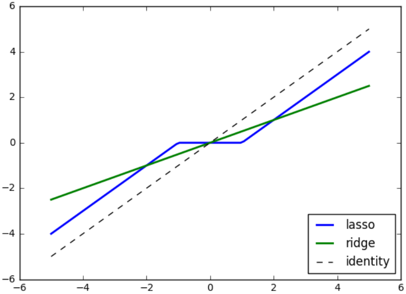
Calcul direct du prox du LASSO (3)

$$\underset{z' \in \mathbb{R}}{\operatorname{argmin}} \frac{1}{2}(z' - z)^2 + \frac{1}{C}|z'| = \operatorname{sign}(z)\left(|z| - \frac{1}{C}\right)_+$$

donc

$$\underset{w' \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{2}\|w' - w\|_2^2 + \frac{1}{C}\|w'\|_1 = \operatorname{sign}(w) \odot \left(|w| - \frac{1}{C}\right)_+ .$$

Exemple avec $C = 1$



Descente de gradient proximale (PGD)

- **Input**: initialisation w^0, constante de Lipschitz $L > 0$ pour ∇f,
- pour $k = 1, 2, \ldots$ jusqu'à *convergence* faire

$$w^k \leftarrow \operatorname{prox}_{g/L}\left(w^{k-1} - \frac{1}{L}\nabla f(w^{k-1})\right)$$

- **Renvoyer** w^k

Pour le Lasso

$$\hat{w} \in \operatorname*{argmin}_{w \in \mathbb{R}^d} \left\{ \frac{1}{2n}\|y - Xw\|_2^2 + \lambda\|w\|_1 \right\},$$

l'itération est donnée par

$$w^k \leftarrow S_{\lambda/L}\Big(w^{k-1} - \frac{1}{Ln}X^\top(Xw^{k-1} - y)\Big),$$

où S_λ est l'opérateur de seuillage doux.

Exercices

Avec l'intercept b

Récrire l'algorithme de descente de gradient proximale quand ℓ dépend à la fois de w et de b, c'est-à-dire pour le problème de minimisation

$$\hat{w}, \hat{b} \in \underset{w \in \mathbb{R}^d, b \in \mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{i=1}^n \ell(y_i, \langle x_i, w \rangle + b) + \frac{1}{C} \operatorname{pen}(w) \right\}.$$

Elastic-net

Récrire l'algorithme de descente de gradient proximale pour la pénalité elastic-net

$$\operatorname{pen}(w) = \frac{1-\alpha}{2} \|w\|_2^2 + \alpha \|w\|_1$$

Point d'étape

On sait calculer

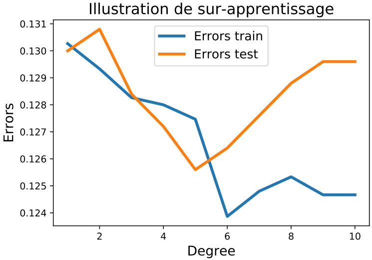
$$\hat{w}, \hat{b} \in \underset{w \in \mathbb{R}^d, b \in \mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle + b) + \frac{1}{C} \operatorname{pen}(w) \right\}.$$

dans les cas du LASSO $\operatorname{pen}(w) = \|w\|_1$ et de la ridge $\operatorname{pen}(w) = \frac{1}{2}\|w\|_2^2$ pour une valeur de C ou de $\lambda = 1/C$. Il reste donc à choisir $C > 0$ ou $\lambda > 0$.

Cross-validation

Sur-apprentissage / sur-ajustement / over-fitting

Sur le jeu de données d'exemple `linear` j'ai ajouté des features en prenant des polynômes des features initiales.



But de l'apprentissage statistique

- Le but de l'apprentissage supervisé (dans le cas de la classification) est en fait de trouver le classifieur qui minimise l'erreur de généralisation

$$c^*_{\text{generalisation}} \in \operatorname*{argmin}_{c} \mathbb{E}\Big(\ell(Y_+, c(X_+))\Big)$$

ou, dans les cas étudiés dans ce chapitre:

$$w^*_{\text{generalisation}} \in \operatorname*{argmin}_{w \in \mathbb{R}^d} \mathbb{E}(\ell(Y_+, \langle X_+, w \rangle))$$

- Pourtant nous définissions

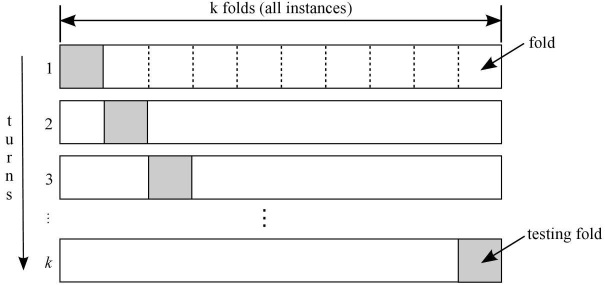
$$\hat{w}, \hat{b} \in \operatorname*{argmin}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle + b) + \frac{1}{C} \operatorname{pen}(w) \right\}.$$

- On doit donc trouver une valeur de C qui rend petite l'erreur de généralisation.
- On va utiliser la cross-validation en montrant comment elle permet d'estimer l'erreur de généralisation.

Take home message il n'y a pas de machine learning sans cross-validation !

Cross-validation V-Fold

- On prend $V = 5$ ou $V = 10$. On choisit une partition aléatoire $I_1, \dots, I_V$ de $\{1, \dots, n\}$, où $|I_v| \approx \frac{n}{V}$ pour tout $v = 1, \dots, V$



On choisit une grille

$$\mathcal{C} = \{C_1, \dots C_K\}$$

de valeurs possibles pour C. Pour tout $v = 1, \dots, V$

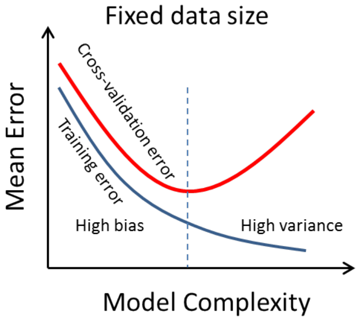
- Posons $I_{v,\texttt{train}} = \cup_{v' \neq v} I_{v'}$ et $I_{v,\texttt{test}} = I_v$
- Pour tout $C \in \mathcal{C}$, on cherche

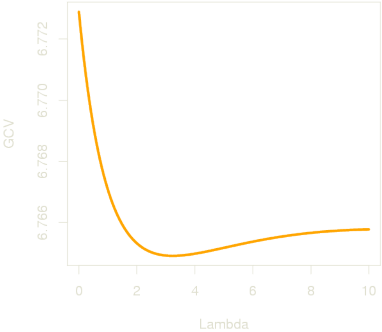
$$\hat{w}_{v,C} \in \operatorname*{argmin}_{w} \left\{ \frac{1}{|I_{v,\texttt{train}}|} \sum_{i \in I_{v,\texttt{train}}} \ell(y_i, \langle x_i, w \rangle) + \frac{1}{C} \operatorname{pen}(w) \right\}$$

On pose

$$\hat{C} \in \operatorname*{argmin}_{C \in \mathcal{C}} \sum_{v=1}^{V} \sum_{i \in I_{v,\texttt{test}}} \ell(y_i, \langle x_i, \hat{w}_{v,C} \rangle)$$

Remarque on peut utiliser d'autres pertes ou métriques pour choisir $\hat{C}$





- Erreur visible/erreur empirique/training error:

$$C \mapsto \sum_{v=1}^{V} \sum_{i \in I_{v,\mathtt{train}}} \ell(y_i, \langle x_i, \hat{w}_{v,C} \rangle)$$

- Erreur de test/de validation/de cross-validation/testing error

$$C \mapsto \sum_{v=1}^{V} \sum_{i \in I_{v,\mathtt{test}}} \ell(y_i, \langle x_i, \hat{w}_{v,C} \rangle)$$

Métriques en classification

Métriques standard classification (1)

- Precision, Recall, F-Score, AUC

Pour chaque individu i nous avons

- son vrai label y_i
- son label prédit $\hat{y}_i$

On peut construire **la matrice de confusion**

		Predicted Class	
		Yes	No
Actual Class	Yes	TP	FN
	No	FP	TN

with

$$
\begin{aligned}
TP &= \sum_{i=1}^{n} \mathbb{1}_{y_i=1,\hat{y}_i=1} \\
TN &= \sum_{i=1}^{n} \mathbb{1}_{y_i=-1,\hat{y}_i=-1} \\
FN &= \sum_{i=1}^{n} \mathbb{1}_{y_i=1,\hat{y}_i=-1} \\
FP &= \sum_{i=1}^{n} \mathbb{1}_{y_i=-1,\hat{y}_i=1}
\end{aligned}
$$

avec yes $= 1$ et no $= -1$

Métriques standard classification (2)

$$\text{Precision} = \frac{\text{TP}}{\#(\text{predicted P})} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\#(\text{real P})} = \frac{\text{TP}}{\text{TP} + \text{FN}} =$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{F-Score} = 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Un peu de vocabulaire

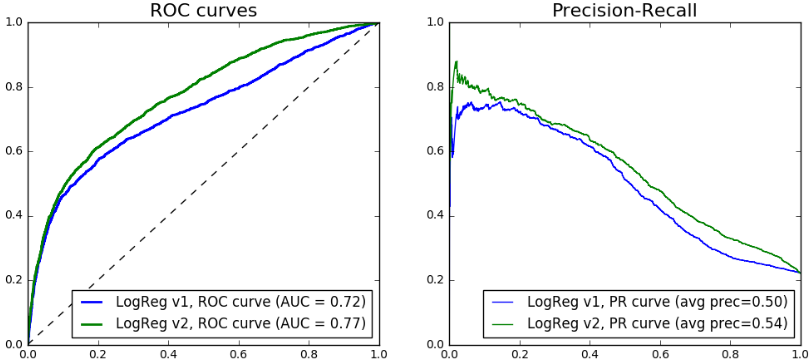
- Recall $=$ Sensitivity
- False-Discovery Proportion FDP $= 1 -$ Precision

Courbe ROC

- On part des probalitités estimées $\hat{\pi}_1(x_i) = \hat{\mathbb{P}}(Y = 1|X = x_i)$
- Chaque point A_t de la courbe a pour coordonnées $(\text{FPP}_s, \text{Recall}_s)$, où FPP_s et Recall_s sont les FPP et le recall de la matrice de confusion obtenue avec la règle de classification

  $$\hat{Y}_i = \begin{cases} 1 & \text{si } \hat{\pi}_1(x_i) \geq s \\ -1 & \text{sinon} \end{cases}$$

  pour un seuil s variant dans $[0, 1]$
- l'AUC est alors l'aire sous la courbe ROC.


ROC curves
LogReg v1, ROC curve (AUC = 0.72)
LogReg v2, ROC curve (AUC = 0.77)
0.0
0.2
0.4
0.6
0.8
1.0
Precision-Recall
LogReg v1, PR curve (avg prec=0.50)
LogReg v2, PR curve (avg prec=0.54)