

Données déséquilibrées

L. Rouvière

laurent.rouviere@univ-rennes2.fr

Novembre 2020

- **Objectifs** : Adapter les algorithmes machine learning aux cas de données déséquilibrées.
- **Pré-requis** : Modélisation statistique, Régression logistique, Machine Learning. R, niveau avancé.
- **Enseignant** : Laurent Rouvière laurent.rouviere@univ-rennes2.fr
 - **Recherche** : statistique non paramétrique, apprentissage statistique
 - **Enseignements** : statistique et probabilités (Université, école d'ingénieur et de commerce, formation continue).
 - **Consulting** : energie, finance, marketing.

- 8h : 4h CM + 3 TP + 1h TD.
- Matériel : slides + Tutoriel R. Disponible à l'url :
<https://lrouviere.github.io/INP-HB/>
- 3 parties :
 1. Introduction : Données déséquilibrées et modèle logistique
 2. Stratégies générales
 3. Choisir un algorithme pour des données déséquilibrées

Introduction

- Dans de nombreux problèmes de **classification binaires** (détection de fraudes, maladies rares...), les deux classes ne sont **pas représentées de façon égale** dans le jeu de données.
- On parle de **données déséquilibrées** (unbalanced data).
- Les algorithmes classiques peuvent être **mis en défaut** dans ce contexte.
- Nécessité d'envisager **différentes stratégies** pour répondre à ce problème.

Un exemple

- On considère 3 jeux de données générées selon le même processus mais avec des proportions de 0 et de 1 différentes.

```
> summary(df1$Y)
##    0    1
## 559 441
> summary(df2$Y)
##    0    1
## 692 308
> summary(df3$Y)
##    0    1
## 842 158
```

Un exemple

- On considère 3 jeux de données générées selon le même processus mais avec des proportions de 0 et de 1 différentes.

```
> summary(df1$Y)
##    0    1
## 559 441
> summary(df2$Y)
##    0    1
## 692 308
> summary(df3$Y)
##    0    1
## 842 158
```

- On sépare les données en 2, on ajuste une forêt aléatoire sur l'apprentissage et on prédit sur le test.

- On obtient les tables de contingence suivantes :

```
> table(p1,Y=test1$Y)
##      Y
## p1    0    1
##    0 142  67
##    1  42  81
> table(p2,Y=test2$Y)
##      Y
## p2    0    1
##    0 196  51
##    1  37  48
> table(p3,Y=test3$Y)
##      Y
## p3    0    1
##    0 253  46
##    1  21  12
```

- En terme d'erreur de classement, les meilleurs résultats sont pour $df3$.
- Mais

- En terme d'erreur de classement, les meilleurs résultats sont pour **df3**.
- Mais on détecte moins bien les 1 avec **df3**.

- En terme d'erreur de classement, les meilleurs résultats sont pour **df3**.
- Mais on détecte moins bien les 1 avec **df3**.

Questions

1. Critères spécifiques aux données déséquilibrées ?

- En terme d'erreur de classement, les meilleurs résultats sont pour **df3**.
- Mais on détecte moins bien les 1 avec **df3**.

Questions

1. Critères spécifiques aux données déséquilibrées ?
2. Algorithmes standards efficaces avec des données déséquilibrées ?

- En terme d'erreur de classement, les meilleurs résultats sont pour **df3**.
- Mais on détecte moins bien les 1 avec **df3**.

Questions

1. Critères spécifiques aux données déséquilibrées ?
2. Algorithmes standards efficaces avec des données déséquilibrées ?
3. Comment les adapter ?

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

- Critères basés sur des règles

- Critères basés sur des scores

Ré-échantillonnage

- Oversampling

- Undersampling

- Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

- Critères basés sur des règles

- Critères basés sur des scores

Ré-échantillonnage

- Oversampling

- Undersampling

- Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

Un exemple

- On considère les modèles logistiques

$$\text{logit } p_{\beta}(x_i) = \beta_0 + \beta_1 x_i, \quad i = 1, \dots, 200$$

où les x_i sont uniformes sur $[0, 1]$, $\beta_1 = -6$ et

- $\beta_0 = 3$ pour le modèle 1
- $\beta_0 = -1$ pour le modèle 2.

Un exemple

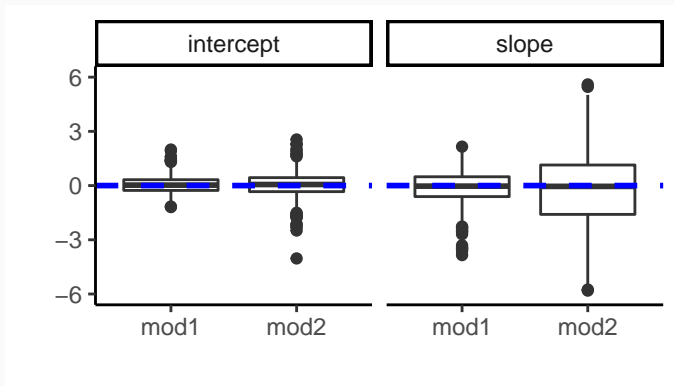
- On considère les modèles logistiques

$$\text{logit } p_{\beta}(x_i) = \beta_0 + \beta_1 x_i, \quad i = 1, \dots, 200$$

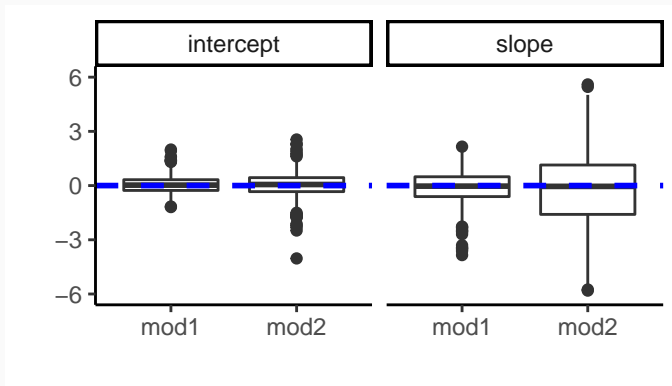
où les x_i sont uniformes sur $[0, 1]$, $\beta_1 = -6$ et

- $\beta_0 = 3$ pour le modèle 1
 - $\beta_0 = -1$ pour le modèle 2.
- On génère $B = 1000$ échantillons $(x_1, y_1), \dots, (x_n, y_n)$ et on s'intéresse à la distribution de l'emv $\hat{\beta}_1$ de β_1 pour les deux modèles.

- On visualise la **distribution de $\hat{\beta}_j - \beta_j$** dans les deux cas :



- On visualise la **distribution de $\hat{\beta}_j - \beta_j$** dans les deux cas :



Conclusion

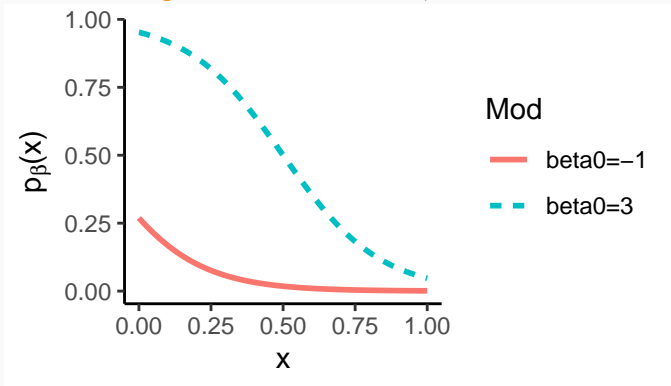
Les emv ont l'air d'être **sans biais**, mais la variance de $\hat{\beta}_1$ est **plus élevée** dans le cas $\beta_0 = -1$ que dans le cas $\beta_0 = 3$ (les écarts types estimés sont respectivement de 0.65 et 2.88).

Pourquoi ?

- Les données sont dans les deux cas générées selon un modèle logistique. Seul changement : la valeur de β_0 .

Pourquoi ?

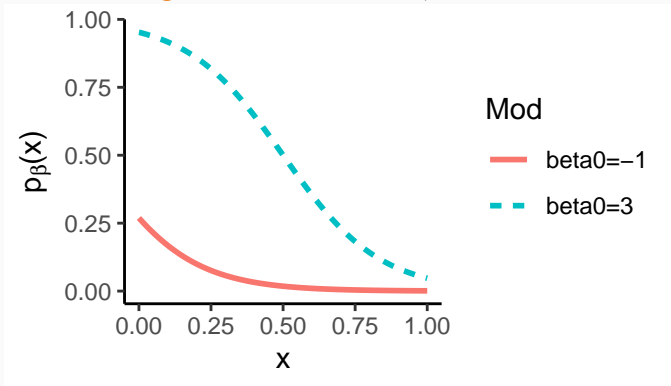
- Les données sont dans les deux cas générées selon un modèle logistique. **Seul changement** : la valeur de β_0 .



- On remarque que $p_\beta(x)$ prend de plus **fortes valeurs** lorsque $\beta = 5$

Pourquoi ?

- Les données sont dans les deux cas générées selon un modèle logistique. **Seul changement** : la valeur de β_0 .



- On remarque que $p_\beta(x)$ prend de plus **fortes valeurs** lorsque $\beta = 5$

Conclusion

La proportion de 1 sera **faible** lorsque $\beta_0 = -1$.

- En effet, sur tous les échantillons simulés, on a en moyenne
 - 50% de 1 lorsque $\beta_0 = 3$;
 - 5% de 1 lorsque $\beta_0 = -1$.

- En effet, sur tous les échantillons simulés, on a en moyenne
 - 50% de 1 lorsque $\beta_0 = 3$;
 - 5% de 1 lorsque $\beta_0 = -1$.
- Nous sommes clairement dans un cas de données déséquilibrées lorsque $\beta_0 = -1$.

- En effet, sur tous les échantillons simulés, on a en moyenne
 - 50% de 1 lorsque $\beta_0 = 3$;
 - 5% de 1 lorsque $\beta_0 = -1$.
- Nous sommes clairement dans un cas de données déséquilibrées lorsque $\beta_0 = -1$.
- Ce déséquilibre semble jouer un rôle sur la performance des estimateurs.

Justification théorique

- On rappelle que, pour n assez grand, la **matrice de variance covariance de l'emv** $\hat{\beta}$ du modèle logistique est

$$\mathcal{I}_n(\beta)^{-1} = (\mathbb{X}' W_{\beta} \mathbb{X})^{-1}$$

avec

$$W_{\beta} = \begin{pmatrix} p_{\beta}(x_1)(1 - p_{\beta}(x_1)) & 0 & \dots & 0 \\ 0 & \ddots & & 0 \\ \vdots & \dots & \ddots & \vdots \\ 0 & \dots & p_{\beta}(x_n)(1 - p_{\beta}(x_n)) \end{pmatrix}$$

Justification théorique

- On rappelle que, pour n assez grand, la **matrice de variance covariance de l'emv $\hat{\beta}$** du modèle logistique est

$$\mathcal{I}_n(\beta)^{-1} = (\mathbb{X}' W_{\beta} \mathbb{X})^{-1}$$

avec

$$W_{\beta} = \begin{pmatrix} p_{\beta}(x_1)(1 - p_{\beta}(x_1)) & 0 & \dots & 0 \\ 0 & \ddots & & 0 \\ \vdots & \dots & \ddots & \vdots \\ 0 & \dots & & p_{\beta}(x_n)(1 - p_{\beta}(x_n)) \end{pmatrix}$$

Conclusion

En présence de fort déséquilibre entre les proportions de 1 et de 0, les éléments de la diagonale de W_{β} vont se rapprocher de 0, ce qui conduit à une **augmentation de la variance des estimateurs**.

- Une solution classique consiste à essayer de "s'arranger" pour rééquilibrer les valeurs de Y dans l'échantillon.

- Une solution classique consiste à essayer de "s'arranger" pour rééquilibrer les valeurs de Y dans l'échantillon.
- On ne peut bien entendu pas faire n'importe comment...

- Une solution classique consiste à essayer de "s'arranger" pour rééquilibrer les valeurs de Y dans l'échantillon.
- On ne peut bien entendu pas faire n'importe comment...
- Cela va forcément affecter le schéma d'échantillonnage.

- Une solution classique consiste à essayer de "s'arranger" pour rééquilibrer les valeurs de Y dans l'échantillon.
- On ne peut bien entendu pas faire n'importe comment...
- Cela va forcément affecter le schéma d'échantillonnage.

Conclusion

Il faut le prendre en compte dans l'écriture du modèle.

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

- On cherche à expliquer une variable binaire Y par une variable X à l'aide d'un modèle logistique : pour $x_i \in \mathbb{R}$, la loi de Y_i est une Bernoulli de paramètre $p_\beta(x_i)$ tel que

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

- On cherche à expliquer une variable binaire Y par une variable X à l'aide d'un modèle logistique : pour $x_i \in \mathbb{R}$, la loi de Y_i est une Bernoulli de paramètre $p_\beta(x_i)$ tel que

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

- Problème : estimer $\beta = (\beta_0, \beta_1)$.

- On cherche à expliquer une variable binaire Y par une variable X à l'aide d'un modèle logistique : pour $x_i \in \mathbb{R}$, la loi de Y_i est une Bernoulli de paramètre $p_\beta(x_i)$ tel que

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

- Problème : estimer $\beta = (\beta_0, \beta_1)$.
- On se place dans le cas où $\pi_1 = \mathbf{P}(Y = 1)$ est petit devant $\pi_0 = \mathbf{P}(Y = 0)$.

- On cherche à expliquer une variable binaire Y par une variable X à l'aide d'un modèle logistique : pour $x_i \in \mathbb{R}$, la loi de Y_i est une Bernoulli de paramètre $p_\beta(x_i)$ tel que

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

- **Problème** : estimer $\beta = (\beta_0, \beta_1)$.
- On se place dans le cas où $\pi_1 = \mathbf{P}(Y = 1)$ est petit devant $\pi_0 = \mathbf{P}(Y = 0)$.
- On a vu que, dans ce cas, la proportion de 1 dans un échantillon $(x_1, y_1), \dots, (x_n, y_n)$ risque d'être faible devant celle de 0, ce qui risque de nous donner des emv avec une forte variance.

- On cherche à expliquer une variable binaire Y par une variable X à l'aide d'un modèle logistique : pour $x_i \in \mathbb{R}$, la loi de Y_i est une Bernoulli de paramètre $p_\beta(x_i)$ tel que

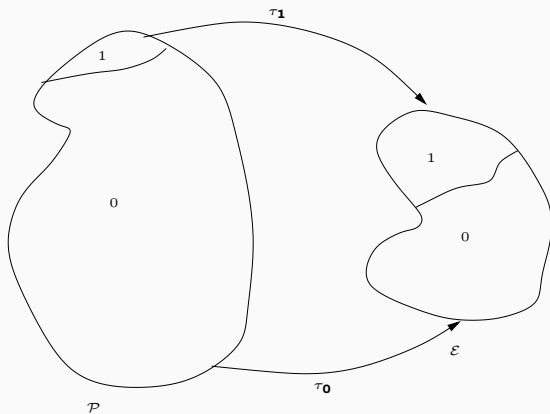
$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

- **Problème** : estimer $\beta = (\beta_0, \beta_1)$.
- On se place dans le cas où $\pi_1 = \mathbf{P}(Y = 1)$ est petit devant $\pi_0 = \mathbf{P}(Y = 0)$.
- On a vu que, dans ce cas, la proportion de 1 dans un échantillon $(x_1, y_1), \dots, (x_n, y_n)$ risque d'être faible devant celle de 0, ce qui risque de nous donner des emv avec une forte variance.

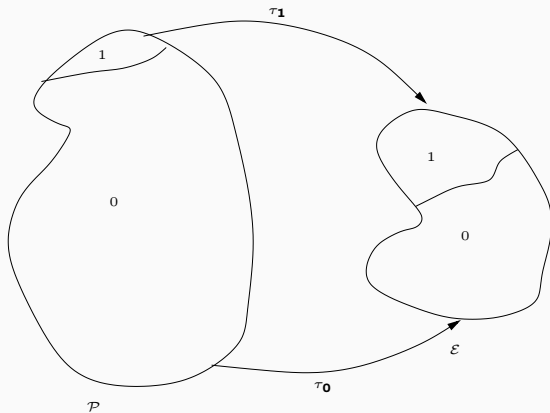
Idée

On va tenter d'obtenir un échantillon avec plus de 1.

- On cherche à **augmenter les 1** dans l'échantillon.



- On cherche à **augmenter les 1** dans l'échantillon.



Variable d'échantillonnage

Soit S_i une **variable aléatoire** à valeurs dans $\{0, 1\}$ telle que

$$S_i = \begin{cases} 1 & \text{on garde l'individu } (x_i, Y_i) \text{ dans l'échantillon} \\ 0 & \text{on le supprime.} \end{cases}$$

- La variable S_i va servir à **rééquilibrer** les 1 et les 0 dans l'échantillon.
- On définit (et on suppose)

$$\tau_0 = \mathbf{P}(S_i = 1 | Y_i = 0) \quad \text{et} \quad \tau_1 = \mathbf{P}(S_i = 1 | Y_i = 1).$$

- La variable S_i va servir à **rééquilibrer** les 1 et les 0 dans l'échantillon.
- On définit (et on suppose)

$$\tau_0 = \mathbf{P}(S_i = 1 | Y_i = 0) \quad \text{et} \quad \tau_1 = \mathbf{P}(S_i = 1 | Y_i = 1).$$

Interprétation

- Ces deux probabilités peuvent être vues comme des "poids".
- Comparées à des **taux de sondage**.

- La variable S_i va servir à **rééquilibrer** les 1 et les 0 dans l'échantillon.
- On définit (et on suppose)

$$\tau_0 = \mathbf{P}(S_i = 1 | Y_i = 0) \quad \text{et} \quad \tau_1 = \mathbf{P}(S_i = 1 | Y_i = 1).$$

Interprétation

- Ces deux probabilités peuvent être vues comme des "poids".
- Comparées à des **taux de sondage**.
- Si $\tau_1 > \tau_0$ alors on va **augmenter la proportion de 1** dans l'échantillon et on peut penser que les **estimateurs seront performants**.

- La variable S_i va servir à **rééquilibrer** les 1 et les 0 dans l'échantillon.
- On définit (et on suppose)

$$\tau_0 = \mathbf{P}(S_i = 1 | Y_i = 0) \quad \text{et} \quad \tau_1 = \mathbf{P}(S_i = 1 | Y_i = 1).$$

Interprétation

- Ces deux probabilités peuvent être vues comme des "poids".
- Comparées à des **taux de sondage**.
- Si $\tau_1 > \tau_0$ alors on va **augmenter la proportion de 1** dans l'échantillon et on peut penser que les **estimateurs seront performants**.
- **Hypothèse sous-jacente** : $S_i, i = 1, \dots, n$ sont, conditionnellement à Y_i , indépendants.

2 modèles

1. **Modèle initial** : $\mathcal{L}(Y_i) = \mathcal{B}(p_\beta(x_i))$ avec

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

2. **Modèle rééquilibré** : $\mathcal{L}(Y_i|S_i = 1) = \mathcal{B}(p_\gamma(x_i))$ avec

$$\text{logit } p_\gamma(x_i) = \text{logit } \mathbf{P}_\gamma(Y_i = 1|S_i = 1) = \gamma_0 + \gamma_1 x_i.$$

- Il faut garder à l'esprit que les **quantités d'intérêt** restent les $p_\beta(x_i)$.

2 modèles

1. **Modèle initial** : $\mathcal{L}(Y_i) = \mathcal{B}(p_\beta(x_i))$ avec

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

2. **Modèle rééquilibré** : $\mathcal{L}(Y_i|S_i = 1) = \mathcal{B}(p_\gamma(x_i))$ avec

$$\text{logit } p_\gamma(x_i) = \text{logit } \mathbf{P}_\gamma(Y_i = 1|S_i = 1) = \gamma_0 + \gamma_1 x_i.$$

- Il faut garder à l'esprit que les **quantités d'intérêt** restent les $p_\beta(x_i)$.
- Le **modèle rééquilibré** devrait permettre de résoudre le problème de déséquilibre si τ_1 et τ_0 sont bien "choisis".

2 modèles

1. **Modèle initial** : $\mathcal{L}(Y_i) = \mathcal{B}(p_\beta(x_i))$ avec

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

2. **Modèle rééquilibré** : $\mathcal{L}(Y_i|S_i = 1) = \mathcal{B}(p_\gamma(x_i))$ avec

$$\text{logit } p_\gamma(x_i) = \text{logit } \mathbf{P}_\gamma(Y_i = 1|S_i = 1) = \gamma_0 + \gamma_1 x_i.$$

- Il faut garder à l'esprit que les **quantités d'intérêt** restent les $p_\beta(x_i)$.
- Le **modèle rééquilibré** devrait permettre de résoudre le problème de déséquilibre si τ_1 et τ_0 sont bien "choisis".
- Néanmoins ce **modèle rééquilibré**, on va **estimer les probabilités** $p_\gamma(x_i)$.

2 modèles

1. **Modèle initial** : $\mathcal{L}(Y_i) = \mathcal{B}(p_\beta(x_i))$ avec

$$\text{logit } p_\beta(x_i) = \beta_0 + \beta_1 x_i.$$

2. **Modèle rééquilibré** : $\mathcal{L}(Y_i|S_i = 1) = \mathcal{B}(p_\gamma(x_i))$ avec

$$\text{logit } p_\gamma(x_i) = \text{logit } \mathbf{P}_\gamma(Y_i = 1|S_i = 1) = \gamma_0 + \gamma_1 x_i.$$

- Il faut garder à l'esprit que les **quantités d'intérêt** restent les $p_\beta(x_i)$.
- Le **modèle rééquilibré** devrait permettre de résoudre le problème de déséquilibre si τ_1 et τ_0 sont bien "choisis".
- Néanmoins ce **modèle rééquilibré**, on va **estimer les probabilités** $p_\gamma(x_i)$.

Question

Quel est le **lien** entre $p_\beta(x_i)$ et $p_\gamma(x_i)$?

Propriété

On a

$$\text{logit } p_{\gamma}(x_i) = \text{logit } p_{\beta}(x_i) + \log \left(\frac{\tau_1}{\tau_0} \right).$$

Par conséquent

$$\text{logit } p_{\beta}(x_i) = \gamma_0 - \log \left(\frac{\tau_1}{\tau_0} \right) + \gamma_1 x_i.$$

Propriété

On a

$$\text{logit } p_{\gamma}(x_i) = \text{logit } p_{\beta}(x_i) + \log \left(\frac{\tau_1}{\tau_0} \right).$$

Par conséquent

$$\text{logit } p_{\beta}(x_i) = \gamma_0 - \log \left(\frac{\tau_1}{\tau_0} \right) + \gamma_1 x_i.$$

- On déduit

$$\beta_0 = \gamma_0 - \log \left(\frac{\tau_1}{\tau_0} \right) \quad \text{et} \quad \beta_1 = \gamma_1.$$

Propriété

On a

$$\text{logit } p_{\gamma}(x_i) = \text{logit } p_{\beta}(x_i) + \log \left(\frac{\tau_1}{\tau_0} \right).$$

Par conséquent

$$\text{logit } p_{\beta}(x_i) = \gamma_0 - \log \left(\frac{\tau_1}{\tau_0} \right) + \gamma_1 x_i.$$

- On déduit

$$\beta_0 = \gamma_0 - \log \left(\frac{\tau_1}{\tau_0} \right) \quad \text{et} \quad \beta_1 = \gamma_1.$$

Commentaires

- Seule la **constante** est affectée par le biais du au rééquilibrage. On peut de plus la corriger si on connaît les **taux de sondage** τ_0 et τ_1 .

Propriété

On a

$$\text{logit } p_{\gamma}(x_i) = \text{logit } p_{\beta}(x_i) + \log \left(\frac{\tau_1}{\tau_0} \right).$$

Par conséquent

$$\text{logit } p_{\beta}(x_i) = \gamma_0 - \log \left(\frac{\tau_1}{\tau_0} \right) + \gamma_1 x_i.$$

- On déduit

$$\beta_0 = \gamma_0 - \log \left(\frac{\tau_1}{\tau_0} \right) \quad \text{et} \quad \beta_1 = \gamma_1.$$

Commentaires

- Seule la **constante** est affectée par le biais du au rééquilibrage. On peut de plus la corriger si on connaît les **taux de sondage** τ_0 et τ_1 .
- L'emv $\hat{\gamma}_1$ est un **estimateur consistant** de β_1 avec a priori **moins de variance** que β_1 .

Exemple

- On utilise l'échantillonnage rétrospectif pour estimer les paramètres du modèle

$$\text{logit } p_{\beta}(x_i) = \beta_0 + \beta_1 x_i, \quad i = 1, \dots, 100$$

où les x_i sont uniformes sur $[0, 1]$, $\beta_0 = -1$ et $\beta_1 = -6$.

Exemple

- On utilise l'échantillonnage rétrospectif pour estimer les paramètres du modèle

$$\text{logit } p_{\beta}(x_i) = \beta_0 + \beta_1 x_i, \quad i = 1, \dots, 100$$

où les x_i sont uniformes sur $[0, 1]$, $\beta_0 = -1$ et $\beta_1 = -6$.

- On calcule les estimateurs du maximum de vraisemblance pour un échantillon :
 - "classique" : $(x_1, y_1), \dots, (x_n, y_n)$ tel que y_i sont générés selon des lois de Bernoulli $p_{\beta}(x_i)$. On note $\hat{\beta}_0$ et $\hat{\beta}_1$ les emv calculés.

Exemple

- On utilise l'échantillonnage rétrospectif pour estimer les paramètres du modèle

$$\text{logit } p_{\beta}(x_i) = \beta_0 + \beta_1 x_i, \quad i = 1, \dots, 100$$

où les x_i sont uniformes sur $[0, 1]$, $\beta_0 = -1$ et $\beta_1 = -6$.

- On calcule les estimateurs du maximum de vraisemblance pour un échantillon :
 - "classique" : $(x_1, y_1), \dots, (x_n, y_n)$ tel que y_i sont générés selon des lois de Bernoulli $p_{\beta}(x_i)$. On note $\hat{\beta}_0$ et $\hat{\beta}_1$ les emv calculés.
 - "rééquilibré" : $(x_1, y_1, 1), \dots, (x_n, y_n, 1)$ issues de (x_i, Y_i, S_i) avec

$$\tau_0 = \mathbf{P}(S = 1 | Y = 0) = 0.95 \quad \text{et} \quad \tau_1 = \mathbf{P}(S = 1 | Y = 1) = 0.05.$$

On note $\hat{\gamma}_0$ et $\hat{\gamma}_1$ les emv calculés.

Exemple

- On utilise l'échantillonnage rétrospectif pour estimer les paramètres du modèle

$$\text{logit } p_{\beta}(x_i) = \beta_0 + \beta_1 x_i, \quad i = 1, \dots, 100$$

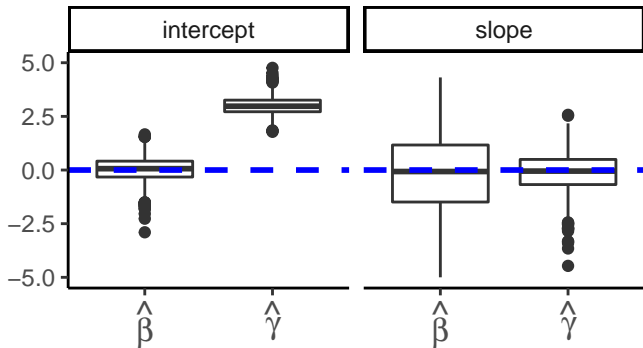
où les x_i sont uniformes sur $[0, 1]$, $\beta_0 = -1$ et $\beta_1 = -6$.

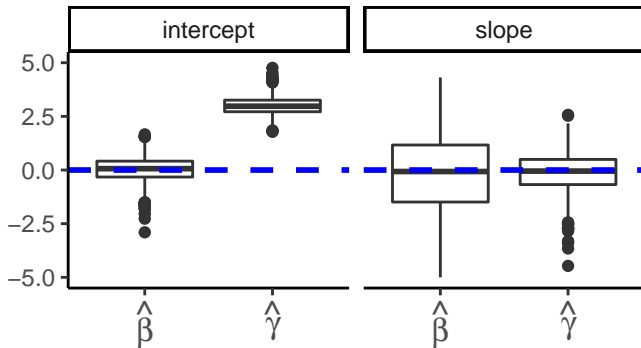
- On calcule les estimateurs du maximum de vraisemblance pour un échantillon :
 - "classique" : $(x_1, y_1), \dots, (x_n, y_n)$ tel que y_i sont générés selon des lois de Bernoulli $p_{\beta}(x_i)$. On note $\hat{\beta}_0$ et $\hat{\beta}_1$ les emv calculés.
 - "rééquilibré" : $(x_1, y_1, 1), \dots, (x_n, y_n, 1)$ issues de (x_i, Y_i, S_i) avec

$$\tau_0 = \mathbf{P}(S = 1 | Y = 0) = 0.95 \quad \text{et} \quad \tau_1 = \mathbf{P}(S = 1 | Y = 1) = 0.05.$$

On note $\hat{\gamma}_0$ et $\hat{\gamma}_1$ les emv calculés.

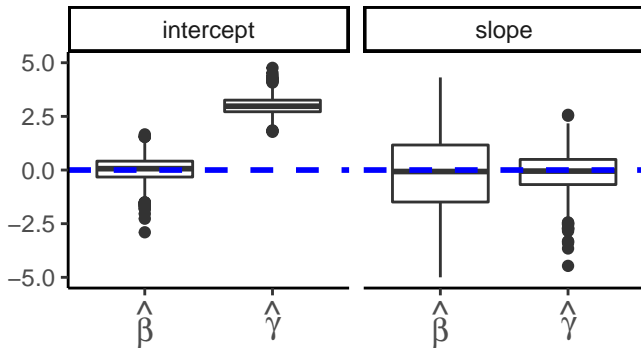
- On répète l'expérience 1000 fois.





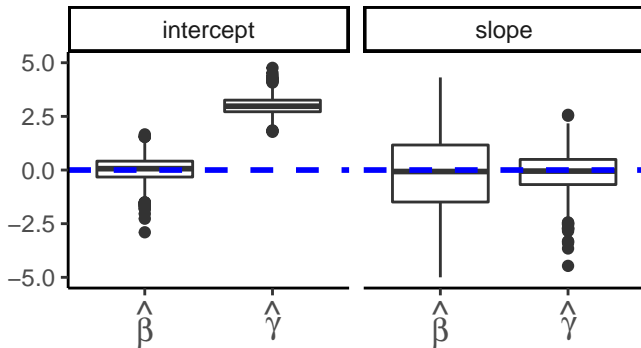
Remarque

- La variance de $\hat{\gamma}_1$ est nettement plus faible que celle de $\hat{\beta}_1$.



Remarque

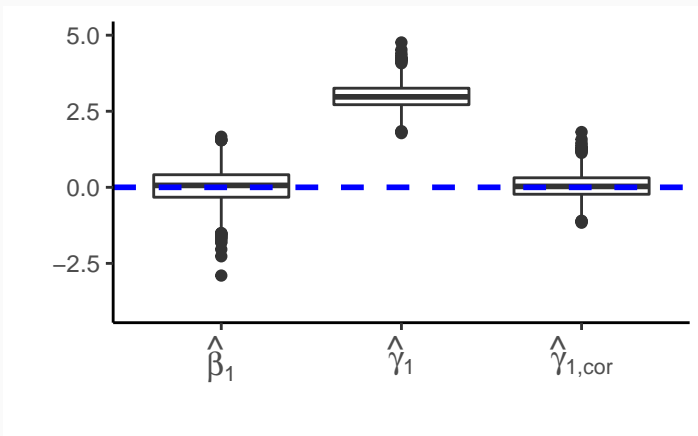
- La **variance** de $\hat{\gamma}_1$ est **nettement plus faible** que celle de $\hat{\beta}_1$.
- En revanche, $\hat{\gamma}_0$ est **biaisé** en tant qu'estimateur de β_0 .



Remarque

- La **variance** de $\hat{\gamma}_1$ est **nettement plus faible** que celle de $\hat{\beta}_1$.
- En revanche, $\hat{\gamma}_0$ est **biaisé** en tant qu'estimateur de β_0 .
- On peut **corriger** ce biais en prenant en considération les **taux de sondage** τ_0 et τ_1 .

Correction de la constante



Remarque

La correction a permis de **débiaiser** l'estimateur de la constante.

Cette **propriété remarquable** du modèle logistique dans le cadre d'un **échantillonnage rétrospectif** peut être appliquée dans (au moins) deux cas.

1. Les **études cas-témoins** (très utilisées en épidémiologie). On souhaite par exemple mesurer l'importance d'un caractère sur une pathologie. On construit alors l'échantillon en sélectionnant
 - un nombre n_1 fixé de patients atteints (**cas**) ;
 - un nombre n_0 fixé de patients sains (**témoins**).

Cette **propriété remarquable** du modèle logistique dans le cadre d'un **échantillonnage rétrospectif** peut être appliquée dans (au moins) deux cas.

1. Les **études cas-témoins** (très utilisées en épidémiologie). On souhaite par exemple mesurer l'importance d'un caractère sur une pathologie. On construit alors l'échantillon en sélectionnant
 - un nombre n_1 fixé de patients atteints (**cas**) ;
 - un nombre n_0 fixé de patients sains (**témoïn**).
2. Lorsque que l'on dispose d'une grande base de données dans lesquels les **individus 1 sont sous représentés**. On construit alors une **deuxième base de données** en donnant un **poids plus élevé aux individus 1** pour être dans la seconde base ($\tau_1 > \tau_0$).

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

- Tout comme pour le modèle logistique, les données déséquilibrées peuvent mettre en défaut les performances des estimateurs dans de nombreux modèles.
- Nécessité de trouver des stratégies pour pallier à cette difficulté.

- Tout comme pour le modèle logistique, les **données déséquilibrées** peuvent **mettre en défaut les performances des estimateurs** dans de nombreux modèles.
- Nécessité de trouver des **stratégies** pour pallier à cette difficulté.

Stratégies classiques

- **Critères de performances** : prendre en compte le déséquilibre dans la mesure de la performance des algorithmes.

- Tout comme pour le modèle logistique, les **données déséquilibrées** peuvent **mettre en défaut les performances des estimateurs** dans de nombreux modèles.
- Nécessité de trouver des **stratégies** pour pallier à cette difficulté.

Stratégies classiques

- **Critères de performances** : prendre en compte le déséquilibre dans la mesure de la performance des algorithmes.
- **Ré-échantillonnage** :
 1. augmenter le poids de la classe sous-représentée dans l'échantillon.
 2. diminuer le poids de la classe sur-représentée dans l'échantillon.

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

Classification binaire : rappel

- (X, Y) à valeurs dans $\mathbb{R}^p \times \{-1, 1\}$.

Classification binaire : rappel

- (X, Y) à valeurs dans $\mathbb{R}^p \times \{-1, 1\}$.
- Règle de classification $\implies g : \mathbb{R}^p \rightarrow \{-1, 1\}$.

Classification binaire : rappel

- (X, Y) à valeurs dans $\mathbb{R}^p \times \{-1, 1\}$.
- Règle de classification $\implies g : \mathbb{R}^p \rightarrow \{-1, 1\}$.
- Fonction de score $\implies S : \mathbb{R}^p \rightarrow \mathbb{R}$.

Classification binaire : rappel

- (X, Y) à valeurs dans $\mathbb{R}^p \times \{-1, 1\}$.
- Règle de classification $\implies g : \mathbb{R}^p \rightarrow \{-1, 1\}$.
- Fonction de score $\implies S : \mathbb{R}^p \rightarrow \mathbb{R}$.

Lien

Une règle peut se déduire d'un score en fixant un seuil $s \in \mathbb{R}$:

$$g_s(x) = \begin{cases} 1 & \text{si } S(x) \geq s. \\ 0 & \text{sinon.} \end{cases}$$

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

Critères standards

- Erreur de classification : $EC = P(g(X) \neq Y)$;
- Accuracy : $Acc = P(g(X) = Y) = 1 - EC$;

Critères standards

- Erreur de classification : $EC = \mathbf{P}(g(X) \neq Y)$;
- Accuracy : $Acc = \mathbf{P}(g(X) = Y) = 1 - EC$;
- Faux négatifs : $FNR = \mathbf{P}(g(X) \neq Y | Y = 1)$
- Sensibilité, recall ou vrais positifs :
 $TPR = \mathbf{P}(g(X) = Y | Y = 1) = 1 - FNR$;

Critères standards

- Erreur de classification : $EC = P(g(X) \neq Y)$;
- Accuracy : $Acc = P(g(X) = Y) = 1 - EC$;
- Faux négatifs : $FNR = P(g(X) \neq Y | Y = 1)$
- Sensibilité, recall ou vrais positifs :
 $TPR = P(g(X) = Y | Y = 1) = 1 - FNR$;
- Faux positifs : $FPR = P(g(X) \neq Y | Y = -1)$;
- Spécificité ou vrais négatifs :
 $TNR = P(g(X) = Y | Y = -1) = 1 - FPR$;

Critères standards

- Erreur de classification : $EC = P(g(X) \neq Y)$;
- Accuracy : $Acc = P(g(X) = Y) = 1 - EC$;
- Faux négatifs : $FNR = P(g(X) \neq Y | Y = 1)$
- Sensibilité, recall ou vrais positifs :
 $TPR = P(g(X) = Y | Y = 1) = 1 - FNR$;
- Faux positifs : $FPR = P(g(X) \neq Y | Y = -1)$;
- Spécificité ou vrais négatifs :
 $TNR = P(g(X) = Y | Y = -1) = 1 - FPR$;
- False discovery rate : $FDR = P(g(X) \neq Y | g(X) = 1)$;
- Précision ou positive predicted value :
 $PPV = P(g(X) = Y | g(X) = 1) = 1 - FDR$;

Critères standards

- Erreur de classification : $EC = P(g(X) \neq Y)$;
- Accuracy : $Acc = P(g(X) = Y) = 1 - EC$;
- Faux négatifs : $FNR = P(g(X) \neq Y | Y = 1)$
- Sensibilité, recall ou vrais positifs :
 $TPR = P(g(X) = Y | Y = 1) = 1 - FNR$;
- Faux positifs : $FPR = P(g(X) \neq Y | Y = -1)$;
- Spécificité ou vrais négatifs :
 $TNR = P(g(X) = Y | Y = -1) = 1 - FPR$;
- False discovery rate : $FDR = P(g(X) \neq Y | g(X) = 1)$;
- Précision ou positive predicted value :
 $PPV = P(g(X) = Y | g(X) = 1) = 1 - FDR$;
- ...

Estimation

- Ces critères sont **inconnus** et généralement estimés à partir de la **table de confusion** :

	$g(X) = 1$	$g(X) = -1$
$Y = 1$	TP	FP
$Y = -1$	FN	TN

Estimation

- Ces critères sont **inconnus** et généralement estimés à partir de la **table de confusion** :

	$g(X) = 1$	$g(X) = -1$
$Y = 1$	TP	FP
$Y = -1$	FN	TN

- Par exemple :

$$\widehat{EC} = \frac{FP+FN}{TP+FP+FN+TN} \quad \text{ou} \quad \widehat{TPR} = \frac{TP}{TP+FN}.$$

Estimation

- Ces critères sont **inconnus** et généralement estimés à partir de la **table de confusion** :

	$g(X) = 1$	$g(X) = -1$
$Y = 1$	TP	FP
$Y = -1$	FN	TN

- Par exemple :

$$\widehat{EC} = \frac{FP+FN}{TP+FP+FN+TN} \quad \text{ou} \quad \widehat{TPR} = \frac{TP}{TP+FN}.$$

Attention

Les prévisions doivent être faites à l'aide de techniques de **validation hold out** ou **validation croisée**.

- Dans le cas de **données déséquilibrées**, la difficulté consiste à choisir un **bon critère** pour mesurer la performance des algorithmes.
- Par exemple, l'**erreur de classification** ou l'**accuracy** se révéleront généralement **peu performants** dans cette situation.

- Dans le cas de **données déséquilibrées**, la difficulté consiste à choisir un **bon critère** pour mesurer la performance des algorithmes.
- Par exemple, l'**erreur de classification** ou l'**accuracy** se révéleront généralement **peu performants** dans cette situation.

Comment ?

Prendre en compte des "**critères conditionnels**" pour éviter de donner un **poids trop important** à la classe sur-représentée.

Balanced accuracy

- On part de l'accuracy :

$$\begin{aligned} \mathbf{P}(g(X) = Y) = & \mathbf{P}(g(X) = 1|Y = 1)\mathbf{P}(Y = 1) \\ & + \mathbf{P}(g(X) = -1|Y = -1)\mathbf{P}(Y = -1); \end{aligned}$$

- Si une classe est **sur-représentée**, l'erreur dans cette classe sera privilégiée.
- Critère mal adapté pour mesurer la **capacité à détecter la petite classe**.

Balanced accuracy

Il donne le **même poids** aux deux erreurs

$$\text{Bal Acc} = \frac{1}{2}\mathbf{P}(g(X) = 1|Y = 1) + \frac{1}{2}\mathbf{P}(g(X) = -1|Y = -1) = \frac{\text{TPR} + \text{TNR}}{2}.$$

Balanced accuracy

- On part de l'accuracy :

$$\begin{aligned} \mathbf{P}(g(X) = Y) = & \mathbf{P}(g(X) = 1|Y = 1)\mathbf{P}(Y = 1) \\ & + \mathbf{P}(g(X) = -1|Y = -1)\mathbf{P}(Y = -1); \end{aligned}$$

Balanced accuracy

- On part de l'accuracy :

$$\begin{aligned} \mathbf{P}(g(X) = Y) = & \mathbf{P}(g(X) = 1|Y = 1)\mathbf{P}(Y = 1) \\ & + \mathbf{P}(g(X) = -1|Y = -1)\mathbf{P}(Y = -1); \end{aligned}$$

- Si une classe est **sur-représentée**, l'erreur dans cette classe sera privilégiée.

Balanced accuracy

- On part de l'accuracy :

$$\begin{aligned} \mathbf{P}(g(X) = Y) = & \mathbf{P}(g(X) = 1|Y = 1)\mathbf{P}(Y = 1) \\ & + \mathbf{P}(g(X) = -1|Y = -1)\mathbf{P}(Y = -1); \end{aligned}$$

- Si une classe est **sur-représentée**, l'erreur dans cette classe sera privilégiée.
- Critère mal adapté pour mesurer la **capacité à détecter la petite classe**.

Balanced accuracy

- On part de l'accuracy :

$$\begin{aligned} \mathbf{P}(g(X) = Y) = & \mathbf{P}(g(X) = 1|Y = 1)\mathbf{P}(Y = 1) \\ & + \mathbf{P}(g(X) = -1|Y = -1)\mathbf{P}(Y = -1); \end{aligned}$$

- Si une classe est **sur-représentée**, l'erreur dans cette classe sera privilégiée.
- Critère mal adapté pour mesurer la **capacité à détecter la petite classe**.

Balanced accuracy

Il donne le **même poids** aux deux erreurs

$$\text{Bal Acc} = \frac{1}{2}\mathbf{P}(g(X) = 1|Y = 1) + \frac{1}{2}\mathbf{P}(g(X) = -1|Y = -1) = \frac{\text{TPR} + \text{TNR}}{2}.$$

- Le **balanced accuracy** est la **moyenne arithmétique** des vrais positifs et négatifs :

- Le **balanced accuracy** est la **moyenne arithmétique** des vrais positifs et négatifs :
- Le **F_1 -score** est la **moyenne harmonique** entre
 1. la **précision** $\mathbf{P}(Y = 1|g(X) = 1)$ (capacité à identifier les positifs parmi les prédits positifs) ;
 2. et le **recall** : $\mathbf{P}(g(X) = 1|Y = 1)$ (capacité à bien prédire les positifs)

F_1 score

- Le **balanced accuracy** est la **moyenne arithmétique** des vrais positifs et négatifs :
- Le **F_1 -score** est la **moyenne harmonique** entre
 1. la **précision** $\mathbf{P}(Y = 1|g(X) = 1)$ (capacité à identifier les positifs parmi les prédits positifs) ;
 2. et le **recall** : $\mathbf{P}(g(X) = 1|Y = 1)$ (capacité à bien prédire les positifs)

F_1 score

$$F_1 = 2 \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

F_1 score

- Le **balanced accuracy** est la **moyenne arithmétique** des vrais positifs et négatifs :
- Le **F_1 -score** est la **moyenne harmonique** entre
 1. la **précision** $\mathbf{P}(Y = 1|g(X) = 1)$ (capacité à identifier les positifs parmi les prédicts positifs) ;
 2. et le **recall** : $\mathbf{P}(g(X) = 1|Y = 1)$ (capacité à bien prédire les positifs)

F_1 score

$$F_1 = 2 \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Interprétation

- **Bal Acc** et F_1 varient entre 0 et 1.
- Plus ils sont **proches de 1**, **meilleure** sera la règle.
- **Attention** : le F_1 score n'est pas symétrique !

- Il consiste à comparer
 - la probabilité de bien classer (d'accord), c'est-à-dire l'**accuracy** $P(a)$;

- Il consiste à comparer
 - la probabilité de bien classer (d'accord), c'est-à-dire l'**accuracy** $\mathbf{P}(a)$;
 - à une probabilité de bien classer de façon **aléatoire** $\mathbf{P}(e)$, c'est-à-dire en supposant que la règle $g(X)$ est indépendante de Y :

- Il consiste à comparer
 - la probabilité de bien classer (d'accord), c'est-à-dire l'**accuracy** $\mathbf{P}(a)$;
 - à une probabilité de bien classer de façon **aléatoire** $\mathbf{P}(e)$, c'est-à-dire en supposant que la règle $g(X)$ est indépendante de Y :

$$\begin{aligned}\mathbf{P}(e) &= \mathbf{P}_{\text{al}}(g(X) = Y) \\ &= \mathbf{P}(g(X) = -1)\mathbf{P}(Y = -1) + \mathbf{P}(g(X) = 1)\mathbf{P}(Y = 1).\end{aligned}$$

- Il consiste à comparer
 - la probabilité de bien classer (d'accord), c'est-à-dire l'**accuracy** $\mathbf{P}(a)$;
 - à une probabilité de bien classer de façon **aléatoire** $\mathbf{P}(e)$, c'est-à-dire en supposant que la règle $g(X)$ est indépendante de Y :

$$\begin{aligned}\mathbf{P}(e) &= \mathbf{P}_{\text{al}}(g(X) = Y) \\ &= \mathbf{P}(g(X) = -1)\mathbf{P}(Y = -1) + \mathbf{P}(g(X) = 1)\mathbf{P}(Y = 1).\end{aligned}$$

Définition

Le κ de Cohen est défini par

$$\kappa = \frac{\mathbf{P}(a) - \mathbf{P}(e)}{1 - \mathbf{P}(e)}.$$

- Il consiste à comparer
 - la probabilité de bien classer (d'accord), c'est-à-dire l'**accuracy** $P(a)$;
 - à une probabilité de bien classer de façon **aléatoire** $P(e)$, c'est-à-dire en supposant que la règle $g(X)$ est indépendante de Y :

$$\begin{aligned}P(e) &= P_{\text{al}}(g(X) = Y) \\ &= P(g(X) = -1)P(Y = -1) + P(g(X) = 1)P(Y = 1).\end{aligned}$$

Définition

Le **κ de Cohen** est défini par

$$\kappa = \frac{P(a) - P(e)}{1 - P(e)}.$$

Plus ce coefficient est **proche de 1**, meilleure sera la règle.

Sensibilité aux données déséquilibrées

- Cas de données déséquilibrées : $P(Y = -1)$ grand.
- Règle qui classe (presque) toujours -1.

- Cas de données déséquilibrées : $P(Y = -1)$ grand.
- Règle qui classe (presque) toujours -1.
- On a alors :

$$P(a) \approx P(Y = -1) \quad \text{et} \quad P(e) \approx P(Y = -1).$$

Sensibilité aux données déséquilibrées

- Cas de données déséquilibrées : $P(Y = -1)$ grand.
- Règle qui classe (presque) toujours -1.
- On a alors :

$$P(a) \approx P(Y = -1) \quad \text{et} \quad P(e) \approx P(Y = -1).$$

Conclusion

Le κ de Cohen va prendre des petites valeurs dans ce cas là alors que l'accuracy sera proche de 1.

Exemple

- On souhaite **comparer deux algorithmes** qui ont fourni les prévisions P_1 et P_2 dont voici les **tables de contingence** :

```
> table(P1,Y)
```

```
##      Y
## P1    0    1
##    0 468  31
##    1   0   1
```

```
> table(P2,Y)
```

```
##      Y
## P2    0    1
##    0 407   4
##    1  61  28
```

Exemple

- On souhaite **comparer deux algorithmes** qui ont fourni les prévisions P_1 et P_2 dont voici les **tables de contingence** :

```
> table(P1,Y)
```

```
##      Y
## P1    0    1
##    0 468  31
##    1   0   1
```

```
> table(P2,Y)
```

```
##      Y
## P2    0    1
##    0 407   4
##    1  61  28
```

Remarque

- La classe 0 est **sur-représentée**.

Exemple

- On souhaite **comparer deux algorithmes** qui ont fourni les prévisions P_1 et P_2 dont voici les **tables de contingence** :

```
> table(P1,Y)
```

```
##      Y
## P1    0    1
##    0 468  31
##    1   0   1
```

```
> table(P2,Y)
```

```
##      Y
## P2    0    1
##    0 407   4
##    1  61  28
```

Remarque

- La classe 0 est **sur-représentée**.
- En terme d'**erreur de classification**, P_1 semble meilleur.

Exemple

- On souhaite **comparer deux algorithmes** qui ont fourni les prévisions P_1 et P_2 dont voici les **tables de contingence** :

```
> table(P1,Y)
```

```
##      Y
## P1    0    1
##    0 468  31
##    1   0   1
```

```
> table(P2,Y)
```

```
##      Y
## P2    0    1
##    0 407   4
##    1  61  28
```

Remarque

- La classe 0 est **sur-représentée**.
- En terme d'**erreur de classification**, P_1 semble meilleur.
- P_2 semble plus à même de **détecter les 1**.

Exemple

- On souhaite **comparer deux algorithmes** qui ont fourni les prévisions P_1 et P_2 dont voici les **tables de contingence** :

```
> table(P1,Y)
```

```
##      Y
## P1    0    1
##    0 468  31
##    1   0   1
```

```
> table(P2,Y)
```

```
##      Y
## P2    0    1
##    0 407   4
##    1  61  28
```

Remarque

- La classe 0 est **sur-représentée**.
 - En terme d'**erreur de classification**, P_1 semble meilleur.
 - P_2 semble plus à même de **détecter les 1**.
-
- La fonction **confusionMatrix** du package **caret** permet de calculer la plupart des indicateurs proposés.

```

> library(caret)
> confusionMatrix(data=P1,reference=Y,mode="everything",positive="1")
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    0    1
##      0 468   31
##      1   0    1
##
##
##              Accuracy : 0.938
##              95% CI : (0.9131, 0.9575)
##      No Information Rate : 0.936
##      P-Value [Acc > NIR] : 0.4741
##
##
##              Kappa : 0.0569
##
##
##      McNemar's Test P-Value : 7.118e-08
##
##
##              Sensitivity : 0.03125
##              Specificity : 1.00000
##              Pos Pred Value : 1.00000
##              Neg Pred Value : 0.93788
##              Precision : 1.00000
##              Recall : 0.03125
##              F1 : 0.06061
##              Prevalence : 0.06400
##              Detection Rate : 0.00200
##              Detection Prevalence : 0.00200
##              Balanced Accuracy : 0.51562
##
##
##      'Positive' Class : 1
##

```



```

> confusionMatrix(data=P2,reference=Y,mode="everything",positive="1")
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    0    1
##      0 407    4
##      1  61   28
##
##              Accuracy : 0.87
##              95% CI : (0.8373, 0.8982)
##      No Information Rate : 0.936
##      P-Value [Acc > NIR] : 1
##
##              Kappa : 0.407
##
##      McNemar's Test P-Value : 3.759e-12
##
##              Sensitivity : 0.8750
##              Specificity : 0.8697
##      Pos Pred Value : 0.3146
##      Neg Pred Value : 0.9903
##              Precision : 0.3146
##              Recall : 0.8750
##              F1 : 0.4628
##              Prevalence : 0.0640
##      Detection Rate : 0.0560
##      Detection Prevalence : 0.1780
##      Balanced Accuracy : 0.8723
##
##      'Positive' Class : 1
##

```

- Sans surprise, l'Accuracy va privilégier P_1 ;

- Sans surprise, l'**Accuracy** va privilégier P_1 ;
- Des critères tels que le **F_1 -score** et le **κ de Cohen** vont quand à eux sélectionner P_2 (qui semble **mieux approprié** sur cet exemple).

Connaissances a priori

- Les critères présentés précédemment sont des **critères généraux** qui permettent de comparer des prévisions.
- Dans certains cas, l'**expertise métier** peut nous apporter une **information supplémentaire** sur le **bon critère** à considérer.

Connaissances a priori

- Les critères présentés précédemment sont des **critères généraux** qui permettent de comparer des prévisions.
- Dans certains cas, l'**expertise métier** peut nous apporter une **information supplémentaire** sur le **bon critère** à considérer.

Exemple

- **German credit dataset** de l'UCI machine learning repository.
- On dispose de l'information suivante :

This dataset requires use of a cost matrix (see below)

```
..... 1 2
-----
1 0 1
-----
2 5 0
```

(1 = Good, 2 = Bad)

The rows represent the actual classification and the columns the predicted classification.

- Une erreur est 5 fois plus importante que l'autre : prédire 1 alors qu'on a observé 2 est 5 fois plus important que prédire 2 alors qu'on a observé 1.

- Une erreur est 5 fois plus importante que l'autre : prédire 1 alors qu'on a observé 2 est 5 fois plus important que prédire 2 alors qu'on a observé 1.
- Cette information doit être prise en compte à la fois pour :

- Une erreur est 5 fois plus importante que l'autre : prédire 1 alors qu'on a observé 2 est 5 fois plus important que prédire 2 alors qu'on a observé 1.
- Cette information doit être prise en compte à la fois pour :
 1. Comparer des algorithmes.

- Une erreur est 5 fois plus importante que l'autre : prédire 1 alors qu'on a observé 2 est 5 fois plus important que prédire 2 alors qu'on a observé 1.
- Cette information doit être prise en compte à la fois pour :
 1. Comparer des algorithmes.
 2. Entraîner les algorithmes.

- Une erreur est 5 fois plus importante que l'autre : prédire 1 alors qu'on a observé 2 est 5 fois plus important que prédire 2 alors qu'on a observé 1.
- Cette information doit être prise en compte à la fois pour :
 1. Comparer des algorithmes.
 2. Entraîner les algorithmes.

Exemple

On pourra par exemple définir un nouveau critère qui donne un poids 5 fois plus important à l'erreur la plus grave.

- On rappelle que l'**erreur de classification** d'une règle g est définie par

$$\mathbf{P}(g(X) \neq Y) = \mathbf{P}(g(X) \neq Y \cap Y = 0) + \mathbf{P}(g(X) \neq Y \cap Y = 1).$$

Pondérer un risque

- On rappelle que l'**erreur de classification** d'une règle g est définie par

$$\mathbf{P}(g(X) \neq Y) = \mathbf{P}(g(X) \neq Y \cap Y = 0) + \mathbf{P}(g(X) \neq Y \cap Y = 1).$$

- Ce critère sous-entend donc que les 2 erreurs possibles ont le **même poids**.

Pondérer un risque

- On rappelle que l'**erreur de classification** d'une règle g est définie par

$$\mathbf{P}(g(X) \neq Y) = \mathbf{P}(g(X) \neq Y \cap Y = 0) + \mathbf{P}(g(X) \neq Y \cap Y = 1).$$

- Ce critère sous-entend donc que les 2 erreurs possibles ont le **même poids**.
- On pourra donner des **poids différents** α_1 et α_2 en considérant l'erreur pondérée

$$\alpha_1 \mathbf{P}(g(X) \neq Y \cap Y = 0) + \alpha_2 \mathbf{P}(g(X) \neq Y \cap Y = 1).$$

Pondérer les erreurs pour entrainer un modèle

- De nombreux algorithmes sont entraînés en minimisant des fonctions de perte.
- Par défaut, ces fonctions de perte donnent généralement le même poids aux deux types d'erreurs.

Pondérer les erreurs pour entrainer un modèle

- De nombreux algorithmes sont entraînés en minimisant des fonctions de perte.
- Par défaut, ces fonctions de perte donnent généralement le même poids aux deux types d'erreurs.
- Dans le cas où les deux erreurs ne sont pas symétriques, il est important de prendre en compte les poids dans le calibrage de la méthode.
- Nous illustrons cela avec les arbres CART.

- L'algorithme produit une suite d'**arbres emboîtés** qui optimisent un critère **coût/complexité**.

Arbres CART : rappels

- L'algorithme produit une suite d'**arbres emboîtés** qui optimisent un critère **coût/complexité**.
- La **sélection de l'arbre final** s'effectue ensuite en **minimisant l'erreur de classification** calculée par validation croisée.

Arbres CART : rappels

- L'algorithme produit une suite d'**arbres emboîtés** qui optimisent un critère **coût/complexité**.
- La **sélection de l'arbre final** s'effectue ensuite en **minimisant l'erreur de classification calculée par validation croisée**.
- Sur **R**, on peut utiliser **rpart** pour construire la suite et **printcp** pour la visualiser.

```

> set.seed(1234)
> tree1 <- rpart(High~., data=data1, cp=0.0001, minsplit=5)
> printcp(tree1)
##
## Classification tree:
## rpart(formula = High ~ ., data = data1, cp = 1e-04, minsplit = 5)
##
## Variables actually used in tree construction:
## [1] Advertising Age          CompPrice  Education  Income      Population
## [7] Price          ShelveLoc   US
##
## Root node error: 164/400 = 0.41
##
## n= 400
##
##      CP nsplit rel error  xerror    xstd
## 1  0.2865854      0  1.000000  1.00000  0.059980
## 2  0.1097561      1  0.713415  0.71341  0.055477
## 3  0.0457317      2  0.603659  0.74390  0.056147
## 4  0.0365854      4  0.512195  0.78049  0.056887
## 5  0.0274390      5  0.475610  0.71951  0.055615
## 6  0.0243902      8  0.390244  0.71951  0.055615
## 7  0.0182927      9  0.365854  0.69512  0.055051
## 8  0.0152439     10  0.347561  0.67073  0.054453
## 9  0.0121951     14  0.286585  0.67073  0.054453
## 10 0.0091463     18  0.231707  0.64024  0.053658
## 11 0.0081301     20  0.213415  0.60976  0.052806
## 12 0.0060976     29  0.140244  0.60976  0.052806
## 13 0.0001000     38  0.085366  0.64634  0.053821

```

- Le même poids est ici donné aux deux erreurs pour :
 1. construire la suite d'arbres à travers le critère coût/complexité.
 2. sélectionner l'arbre dans la suite à travers l'erreur de classification.

- Le même poids est ici donné aux deux erreurs pour :
 1. construire la suite d'arbres à travers le critère coût/complexité.
 2. sélectionner l'arbre dans la suite à travers l'erreur de classification.
- On peut donner un poids différent aux erreurs en modifiant l'argument `parms` dans `rpart`.

- Le même poids est ici donné aux deux erreurs pour :
 1. construire la suite d'arbres à travers le critère coût/complexité.
 2. sélectionner l'arbre dans la suite à travers l'erreur de classification.
- On peut donner un poids différent aux erreurs en modifiant l'argument `parms` dans `rpart`.
- On propose ici par exemple d'utiliser comme critère

$$1\mathbf{P}(g(X) \neq Y \cap Y = 0) + 2\mathbf{P}(g(X) \neq Y \cap Y = 1).$$

```

> tree2 <- rpart(High ~ ., data=data1, params=list(loss=matrix(c(0,2,1,0), ncol=2)), cp=0.0001, minsplit=5)
> printcp(tree2)
##
## Classification tree:
## rpart(formula = High ~ ., data = data1, params = list(loss = matrix(c(0,
##      2, 1, 0), ncol = 2)), cp = 1e-04, minsplit = 5)
##
## Variables actually used in tree construction:
## [1] Advertising Age          CompPrice  Education  Income
## [6] Population Price         ShelveLoc  Urban      US
##
## Root node error: 236/400 = 0.59
##
## n= 400
##
##      CP nsplit rel error  xerror    xstd
## 1  0.1716102      0  1.000000 2.00000 0.083362
## 2  0.0762712      2  0.656780 0.91525 0.067446
## 3  0.0402542      3  0.580508 0.91102 0.069553
## 4  0.0268362      5  0.500000 0.88983 0.069335
## 5  0.0211864      8  0.419492 0.83475 0.067528
## 6  0.0169492     10  0.377119 0.80508 0.067231
## 7  0.0148305     11  0.360169 0.74153 0.065155
## 8  0.0127119     15  0.300847 0.73305 0.064843
## 9  0.0112994     16  0.288136 0.73305 0.064843
## 10 0.0105932     19  0.254237 0.67797 0.062849
## 11 0.0084746     23  0.211864 0.67797 0.062849
## 12 0.0056497     32  0.135593 0.61017 0.059346
## 13 0.0042373     35  0.118644 0.60169 0.058956
## 14 0.0021186     40  0.097458 0.59746 0.058299
## 15 0.0001000     42  0.093220 0.60593 0.058695

```

- On remarque que les suites d'arbres ajustées sont différentes.

- On remarque que les suites d'arbres ajustées sont différentes.
- De plus, les arbres de la seconde suite auront tendance à mieux détecter les yes (1), ce qui est logique vu le critère utilisé.

- On remarque que les suites d'arbres ajustées sont différentes.
- De plus, les arbres de la seconde suite auront tendance à mieux détecter les yes (1), ce qui est logique vu le critère utilisé.
- On peut par exemple comparer les valeurs ajustées des derniers arbres des deux suites.

```
> table(P1,Y)
```

##	Y		
##	P1	No	Yes
##	No	228	6
##	Yes	8	158

```
> table(P2,Y)
```

##	Y		
##	P2	No	Yes
##	No	218	2
##	Yes	18	162

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

- La courbe ROC (et l'AUC) peut se révéler être un critère pertinent dans le cas de données déséquilibrées.

- La courbe ROC (et l'AUC) peut se révéler être un critère pertinent dans le cas de données déséquilibrées.
- Ce critère n'est en effet pas basé sur la prédiction de classes mais sur la manière dont le score ordonne les individus.

- La **courbe ROC** (et l'**AUC**) peut se révéler être un **critère pertinent** dans le cas de **données déséquilibrées**.
- Ce critère n'est en effet **pas basé sur la prédiction de classes** mais sur la manière dont le **score ordonne les individus**.

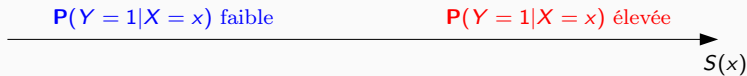
Attention

Si l'objectif final est de **prédire des classes**, il faudra

- se pencher sur le **choix du seuil** ;
- et étudier les **performances de la règle associée au seuil choisi**.

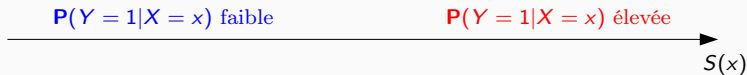
Rappel : courbe ROC

- On reste dans un cadre de **classification binaire** ($\mathcal{Y} = \{-1, 1\}$).
- Mais... plutôt que de chercher une règle de prévision $g : \mathcal{X} \rightarrow \{-1, 1\}$, on **cherche une fonction** $S : \mathcal{X} \rightarrow \mathbb{R}$ telle que



Rappel : courbe ROC

- On reste dans un cadre de **classification binaire** ($\mathcal{Y} = \{-1, 1\}$).
- Mais... plutôt que de chercher une règle de prévision $g : \mathcal{X} \rightarrow \{-1, 1\}$, on **cherche une fonction** $S : \mathcal{X} \rightarrow \mathbb{R}$ telle que



- Une telle fonction est appelée **fonction de score** : plutôt que de prédire directement le groupe d'un nouvel individu $x \in \mathcal{X}$, on lui donne une **note** $S(x)$
 - **élevée** si il a des "chances" d'être dans le groupe 1 ;
 - **faible** si il a des "chances" d'être dans le groupe -1 ;

Lien score/règle de prévision

- Etant donné un score S , on peut déduire une règle de prévision en fixant un seuil s (la réciproque n'est pas vraie) :

$$g_s(x) = \begin{cases} 1 & \text{si } S(x) \geq s \\ -1 & \text{sinon.} \end{cases}$$

- Cette règle définit la table de confusion

	$g_s(X) = -1$	$g_s(X) = 1$
$Y = -1$	OK	E_1
$Y = 1$	E_2	OK

Lien score/règle de prévision

- Etant donné un score S , on peut déduire une **règle de prévision** en **fixant un seuil** s (la réciproque n'est pas vraie) :

$$g_s(x) = \begin{cases} 1 & \text{si } S(x) \geq s \\ -1 & \text{sinon.} \end{cases}$$

- Cette règle définit la **table de confusion**

	$g_s(X) = -1$	$g_s(X) = 1$
$Y = -1$	OK	E_1
$Y = 1$	E_2	OK

- Pour chaque seuil s , on distingue deux types d'**erreur**

$$\alpha(s) = \mathbf{P}(g_s(X) = 1 | Y = -1) = \mathbf{P}(S(X) \geq s | Y = -1)$$

et

$$\beta(s) = \mathbf{P}(g_s(X) = -1 | Y = 1) = \mathbf{P}(S(X) < s | Y = 1).$$

On définit également

- **Spécificité** : $sp(s) = \mathbf{P}(S(X) < s | Y = -1) = 1 - \alpha(s)$
- **Sensibilité** : $se(s) = \mathbf{P}(S(X) \geq s | Y = 1) = 1 - \beta(s)$

On définit également

- **Spécificité** : $sp(s) = P(S(X) < s | Y = -1) = 1 - \alpha(s)$
- **Sensibilité** : $se(s) = P(S(X) \geq s | Y = 1) = 1 - \beta(s)$

Performance d'un score

Elle se mesure généralement en **visualisant** les erreurs $\alpha(s)$ et $\beta(s)$ et/ou la spécificité et la sensibilité pour **tous les seuils s** .

- **Idée** : représenter sur un graphe 2d les deux types d'erreur pour **tous les seuils** s .

Définition

C'est une **courbe paramétrée** par le seuil :

$$\begin{cases} x(s) = \alpha(s) = 1 - sp(s) = \mathbf{P}(S(X) > s | Y = -1) \\ y(s) = 1 - \beta(s) = se(s) = \mathbf{P}(S(X) \geq s | Y = 1) \end{cases}$$

Courbe ROC

- **Idée** : représenter sur un graphe 2d les deux types d'erreur pour **tous les seuils** s .

Définition

C'est une **courbe paramétrée** par le seuil :

$$\begin{cases} x(s) = \alpha(s) = 1 - sp(s) = \mathbf{P}(S(X) > s | Y = -1) \\ y(s) = 1 - \beta(s) = se(s) = \mathbf{P}(S(X) \geq s | Y = 1) \end{cases}$$

Remarque

Pour tout score S on a

- $x(-\infty) = y(-\infty) = 1$;

Courbe ROC

- **Idée** : représenter sur un graphe 2d les deux types d'erreur pour **tous les seuils** s .

Définition

C'est une **courbe paramétrée** par le seuil :

$$\begin{cases} x(s) = \alpha(s) = 1 - sp(s) = \mathbf{P}(S(X) > s | Y = -1) \\ y(s) = 1 - \beta(s) = se(s) = \mathbf{P}(S(X) \geq s | Y = 1) \end{cases}$$

Remarque

Pour tout score S on a

- $x(-\infty) = y(-\infty) = 1$;
- $x(+\infty) = y(+\infty) = 0$;

Courbe ROC

- **Idée** : représenter sur un graphe 2d les deux types d'erreur pour **tous les seuils** s .

Définition

C'est une **courbe paramétrée** par le seuil :

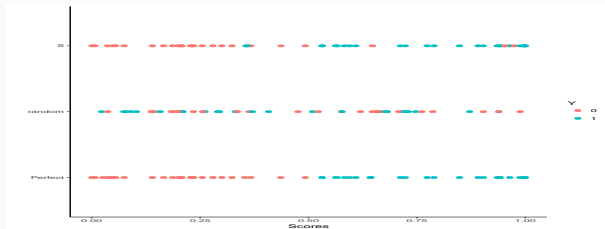
$$\begin{cases} x(s) = \alpha(s) = 1 - sp(s) = \mathbf{P}(S(X) > s | Y = -1) \\ y(s) = 1 - \beta(s) = se(s) = \mathbf{P}(S(X) \geq s | Y = 1) \end{cases}$$

Remarque

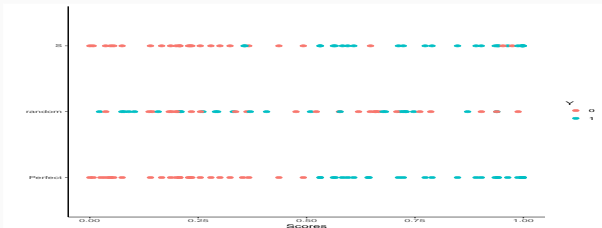
Pour tout score S on a

- $x(-\infty) = y(-\infty) = 1$;
- $x(+\infty) = y(+\infty) = 0$;
- La courbe ROC part de $(1, 1)$ pour finir à $(0, 0)$.

Scores parfait et aléatoire



Scores parfait et aléatoire

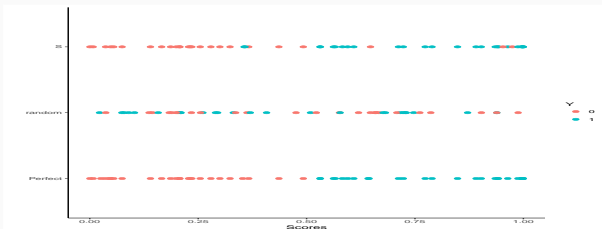


Définition

- **Score parfait** : il est tel qu'il existe un seuil s^* tel que

$$P(Y = 1 | S(X) \geq s^*) = 1 \quad \text{et} \quad P(Y = -1 | S(X) < s^*) = 1.$$

Scores parfait et aléatoire



Définition

- **Score parfait** : il est tel qu'il existe un seuil s^* tel que

$$P(Y = 1 | S(X) \geq s^*) = 1 \quad \text{et} \quad P(Y = -1 | S(X) < s^*) = 1.$$

- **Score aléatoire** : il est tel que $S(X)$ et Y sont indépendantes.

Analyse de la courbe ROC

- Pour un **score parfait** on a $x(s^*) = 0$ et $y(s^*) = 1 \implies$

Analyse de la courbe ROC

- Pour un **score parfait** on a $x(s^*) = 0$ et $y(s^*) = 1 \implies$ une courbe ROC parfaite est définie par les **deux segments**

$$[(1, 1); (0, 1)] \quad \text{et} \quad [(0, 1); (0, 0)].$$

Analyse de la courbe ROC

- Pour un **score parfait** on a $x(s^*) = 0$ et $y(s^*) = 1 \implies$ une courbe ROC parfaite est définie par les **deux segments**

$$[(1, 1); (0, 1)] \quad \text{et} \quad [(0, 1); (0, 0)].$$

- Pour un **score aléatoire** on a $\forall s, x(s) = y(s) \implies$

Analyse de la courbe ROC

- Pour un **score parfait** on a $x(s^*) = 0$ et $y(s^*) = 1 \implies$ une courbe ROC parfaite est définie par les **deux segments**

$$[(1, 1); (0, 1)] \quad \text{et} \quad [(0, 1); (0, 0)].$$

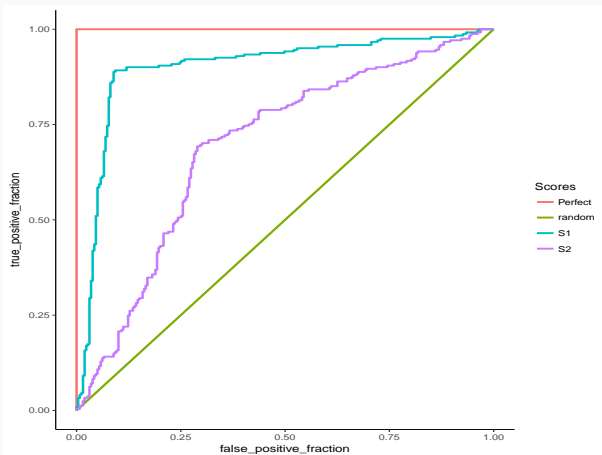
- Pour un **score aléatoire** on a $\forall s, x(s) = y(s) \implies$ la "pire" courbe ROC est donc la **première bissectrice**.

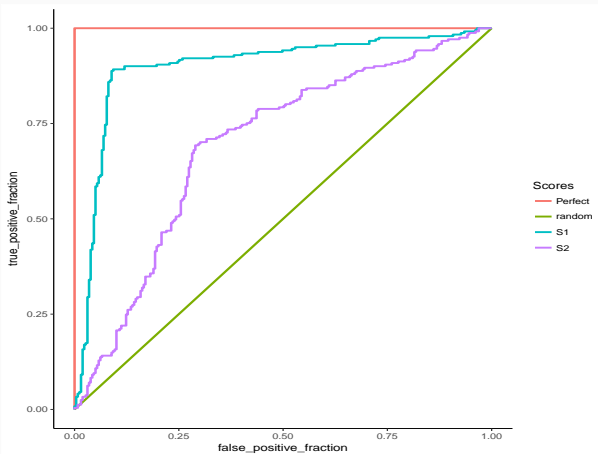
Analyse de la courbe ROC

- Pour un **score parfait** on a $x(s^*) = 0$ et $y(s^*) = 1 \implies$ une courbe ROC parfaite est définie par les **deux segments**

$$[(1, 1); (0, 1)] \quad \text{et} \quad [(0, 1); (0, 0)].$$

- Pour un **score aléatoire** on a $\forall s, x(s) = y(s) \implies$ la "pire" courbe ROC est donc la **première bissectrice**.
- Généralement la **courbe ROC** d'un score classique se situe **entre ces deux courbes**.





Interprétation

On mesurera la performance d'un score par sa **capacité à se rapprocher de la droite d'équation $y = 1$** le plus vite possible.

Définition

- L'aire sous la courbe ROC d'un score S , notée $AUC(S)$ est souvent utilisée pour mesurer sa performance.
- Pour un score parfait on a $AUC(S) = 1$, pour un score aléatoire $AUC(S) = 1/2$.

Définition

- L'aire sous la courbe ROC d'un score S , notée $AUC(S)$ est souvent utilisée pour mesurer sa performance.
- Pour un score parfait on a $AUC(S) = 1$, pour un score aléatoire $AUC(S) = 1/2$.

Proposition

- Etant données deux observations (X_1, Y_1) et (X_2, Y_2) indépendantes et de même loi que (X, Y) , on a

$$AUC(S) = \mathbf{P}(S(X_1) \geq S(X_2) | (Y_1, Y_2) = (1, -1)).$$

```
> library(pROC)
> df1 %>% group_by(Scores) %>% summarize(auc(D,M))
## # A tibble: 4 x 2
##   Scores   'auc(D, M)'
##   <chr>      <dbl>
## 1 Perfect      1
## 2 random    0.5
## 3 S1        0.896
## 4 S2        0.699
```

Score optimal

- Le critère $AUC(S)$ peut être interprété comme une **fonction de perte** pour un score S ;
- Se pose donc la question d'existence d'un **score optimal** S^* vis-à-vis de ce critère.

Score optimal

- Le critère $AUC(S)$ peut être interprété comme une **fonction de perte** pour un score S ;
- Se pose donc la question d'existence d'un **score optimal** S^* vis-à-vis de ce critère.

Théorème [Clémentçon et al., 2008]

Soit $S^*(x) = \mathbf{P}(Y = 1|X = x)$, on a alors pour toutes fonctions de score S

$$AUC(S^*) \geq AUC(S).$$

Score optimal

- Le critère $AUC(S)$ peut être interprété comme une **fonction de perte** pour un score S ;
- Se pose donc la question d'existence d'un **score optimal** S^* vis-à-vis de ce critère.

Théorème [Cléménçon et al., 2008]

Soit $S^*(x) = \mathbf{P}(Y = 1|X = x)$, on a alors pour toutes fonctions de score S

$$AUC(S^*) \geq AUC(S).$$

Conséquence

Le problème pratique consistera à trouver un **"bon" estimateur**

$S_n(x) = S_n(x, \mathcal{D}_n)$ de

$$S^*(x) = \mathbf{P}(Y = 1|X = x).$$

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

- Comme nous l'avons vu dans la première partie, une manière de répondre au problème de données déséquilibrées est de ré-échantillonner en tentant de combler (au moins en partie) le déséquilibre.

- Comme nous l'avons vu dans la première partie, une manière de répondre au problème de données déséquilibrées est de **ré-échantillonner** en tentant de **combler (au moins en partie) le déséquilibre**.
- Ces approches consistent le plus souvent à :
 1. **sur-échantillonner (oversampling)** la classe minoritaire ;

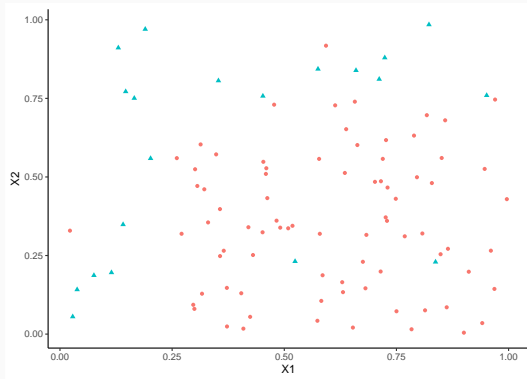
- Comme nous l'avons vu dans la première partie, une manière de répondre au problème de données déséquilibrées est de **ré-échantillonner** en tentant de **combler (au moins en partie) le déséquilibre**.
- Ces approches consistent le plus souvent à :
 1. **sur-échantillonner (oversampling)** la classe minoritaire ;
 2. et/ou **sous-échantillonner (undersampling)** la classe majoritaire.

Différentes techniques pour ré-échantillonner

- Nous en présentons quelques unes mais on pourra consulter le **blog de Jeremy Jourdan**.
- 2 packages R : **unbalanced** et **UBL**

Un exemple

- Nous illustrons les différentes techniques sur les données jouets suivantes.



- $\text{Card}\{i : y_i = 0\} = 80$ et $\text{Card}\{i : y_i = 20\} = 80$

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

Random oversampling

- Méthode la plus naïve : dupliquer aléatoirement des observations dans la classe minoritaire.

```
> library(UBL)
> over1 <- RandOverClassif(Y~.,dat=df)
> over2 <- RandOverClassif(Y~.,dat=df,C.perc=list("0"=1,"1"=2))
> summary(over1$Y)
##  0  1
## 80 80
> summary(over2$Y)
##  0  1
## 80 40
```

- L'approche précédente **diminue** de façon artificielle la **variabilité** dans les données.

- L'approche précédente **diminue** de façon artificielle la **variabilité** dans les données.
- Il existe d'autres approches qui permettent de **générer de nouvelles observations** de la **classe minoritaire**.

- L'approche précédente **diminue** de façon artificielle la **variabilité** dans les données.
- Il existe d'autres approches qui permettent de **générer de nouvelles observations** de la **classe minoritaire**.
- L'algorithme le plus utilisé est **SMOTE** : **S**ynthetic **M**inority **O**ver-sampling **T**Echnique [Chawla et al., 2002].

- L'approche précédente **diminue** de façon artificielle la **variabilité** dans les données.
- Il existe d'autres approches qui permettent de **générer de nouvelles observations** de la **classe minoritaire**.
- L'algorithme le plus utilisé est **SMOTE** : **S**ynthetic **M**inority **O**ver-sampling **T**Echnique [Chawla et al., 2002].
- L'idée est de générer de nouvelles observations **entre des individus de la plus petite classe**.

Pour une observation x_m de la plus petite classe, on **génèrera une nouvelle observation** selon l'algorithme :

1. Calculer les **k plus proches voisins de x_m** parmi les $x_i, i = 1, \dots, n$ privés de x_m qui sont dans la même classe que x_m ;

Pour une observation x_m de la plus petite classe, on **génèrera une nouvelle observation** selon l'algorithme :

1. Calculer les **k plus proches voisins de x_m** parmi les $x_i, i = 1, \dots, n$ privés de x_m qui sont dans le même classe que x_m ;
2. Choisir au **hasard un des plus proches voisins** calculés précédemment, on le note $x_{(m)}$;

Pour une observation x_m de la plus petite classe, on **génèrera une nouvelle observation** selon l'algorithme :

1. Calculer les **k plus proches voisins de x_m** parmi les $x_i, i = 1, \dots, n$ privés de x_m qui sont dans le même classe que x_m ;
2. Choisir au **hasard un des plus proches voisins** calculés précédemment, on le note $x_{(m)}$;
3. Générer aléatoirement un nouveau point x sur le **segment reliant x_m à $x_{(m)}$** .

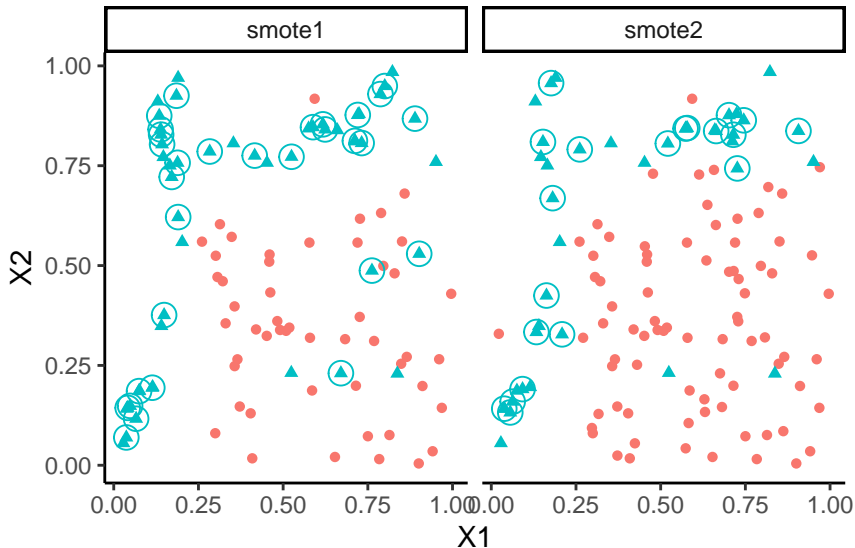
Example

```
> set.seed(1234)
> smote1 <- SmoteClassif(Y~.,dat=df,k=4)
> smote2 <- SmoteClassif(Y~.,dat=df,k=4,C.perc=list("0"=1,"1"=2))
> summary(smote1$Y)
##    0    1
## 50 50
> summary(smote2$Y)
##    0    1
## 80 40
```

Example

```
> set.seed(1234)
> smote1 <- SmoteClassif(Y~.,dat=df,k=4)
> smote2 <- SmoteClassif(Y~.,dat=df,k=4,C.perc=list("0"=1,"1"=2))
> summary(smote1$Y)
##    0    1
## 50 50
> summary(smote2$Y)
##    0    1
## 80 40
```


Résultats SMOTE



- **C.perc** : contrôle le niveau de ré-équilibrage. Par défaut il est parfait :
 1. on **génère 30 nouvelles observations de la classe minoritaire** en appliquant l'algorithme aux 20 individus de la classe 1 puis en le ré-appliquant à 10 individus choisis au hasard ;
 2. on **diminue la classe majoritaire** en choisissant aléatoirement 50 individus parmi les 80.

Paramètres SMOTE

- **C.perc** : contrôle le niveau de ré-équilibrage. Par défaut il est parfait :
 1. on **génère 30 nouvelles observations de la classe minoritaire** en appliquant l'algorithme aux 20 individus de la classe 1 puis en le ré-appliquant à 10 individus choisis au hasard ;
 2. on **diminue la classe majoritaire** en choisissant aléatoirement 50 individus parmi les 80.
- Dans le second cas, **20 nouvelles observations** de la classe minoritaire sont générées en appliquant l'algorithme aux 20 individus de cette classe, la **classe majoritaire reste inchangée**.

Paramètres SMOTE

- **C.perc** : contrôle le niveau de ré-équilibrage. Par défaut il est parfait :
 1. on **génère 30 nouvelles observations de la classe minoritaire** en appliquant l'algorithme aux 20 individus de la classe 1 puis en le ré-appliquant à 10 individus choisis au hasard ;
 2. on **diminue la classe majoritaire** en choisissant aléatoirement 50 individus parmi les 80.
- Dans le second cas, **20 nouvelles observations** de la classe minoritaire sont générées en appliquant l'algorithme aux 20 individus de cette classe, la **classe majoritaire reste inchangée**.
- **Nombre de voisins k** : à choisir par l'utilisateur (ou en utilisant les données...).

- Adaptive synthetic sampling approach for imbalanced learning (voir [He et al., 2008])
- Proche de SMOTE mais

- Adaptive synthetic sampling approach for imbalanced learning (voir [He et al., 2008])
- Proche de SMOTE mais le nombre d'observations générés pour un x_i est proportionnel à la densité des observations du groupe majoritaire au voisinage de x_i .

- Adaptive synthetic sampling approach for imbalanced learning (voir [He et al., 2008])
- Proche de SMOTE mais le nombre d'observations générés pour un x_i est proportionnel à la densité des observations du groupe majoritaire au voisinage de x_i .
- Plus d'observations générées aux voisinages des cas isolés.

Algorithme Adasyn

Entrées : $dth \leq 1$, k plus petit que n_1 , $\beta \in \mathbb{R}^+$.

1. Calculer $d = n_0/n_1$. Si $d \leq dth$ stop.

Algorithme Adasyn

Entrées : $dth \leq 1$, k plus petit que n_1 , $\beta \in \mathbb{R}^+$.

1. Calculer $d = n_0/n_1$. Si $d \leq dth$ stop.
2. Calculer G le nombre d'observations à générer :

$$G = (n_0 - n_1)\beta.$$

Algorithme Adasyn

Entrées : $dth \leq 1$, k plus petit que n_1 , $\beta \in \mathbb{R}^+$.

1. Calculer $d = n_0/n_1$. Si $d \leq dth$ stop.
2. Calculer G le nombre d'observations à générer :

$$G = (n_0 - n_1)\beta.$$

3. Calculer les k -ppv de chaque individu $x_i, i \in \mathcal{X}_1$ de la classe minoritaire et en déduire pour chacun

$$r_i = \frac{\text{card}\{j : y_j = 0 \text{ et } x_j \in kppv(x_i)\}}{k}.$$

Algorithme Adasyn

Entrées : $dth \leq 1$, k plus petit que n_1 , $\beta \in \mathbb{R}^+$.

1. Calculer $d = n_0/n_1$. Si $d \leq dth$ stop.
2. Calculer G le nombre d'observations à générer :

$$G = (n_0 - n_1)\beta.$$

3. Calculer les k -ppv de chaque individu $x_i, i \in \mathcal{X}_1$ de la classe minoritaire et en déduire pour chacun

$$r_i = \frac{\text{card}\{j : y_j = 0 \text{ et } x_j \in \text{kppv}(x_i)\}}{k}.$$

4. Normalisation :

$$\tilde{r}_i = \frac{r_i}{\sum_{i \in \mathcal{X}_1} r_i}.$$

5. Calculer le nombre d'individus à générer pour chaque x_i de \mathcal{X}_1 :

$$G_i = G\check{r}_i.$$

5. Calculer le nombre d'individus à générer pour chaque x_i de \mathcal{X}_1 :

$$G_i = G\check{r}_i.$$

6. Pour chaque x_i de \mathcal{X}_1 répéter G_i fois :

5. Calculer le nombre d'individus à générer pour chaque x_i de \mathcal{X}_1 :

$$G_i = G\check{r}_i.$$

6. Pour chaque x_i de \mathcal{X}_1 répéter G_i fois :

6.1 Choisir au hasard un de ces $kppv$ du groupe 1 $\Rightarrow x_i^{(1)}$

5. Calculer le nombre d'individus à générer pour chaque x_i de \mathcal{X}_1 :

$$G_i = G\check{r}_i.$$

6. Pour chaque x_i de \mathcal{X}_1 répéter G_i fois :

6.1 Choisir au hasard un de ces $kppv$ du groupe 1 $\Rightarrow x_i^{(1)}$

6.2 Générer un point au hasard entre x_i et $x_i^{(1)}$

$$x_{i,j} = x_i + \lambda(x_i^{(1)} - x_i)$$

où λ est générée selon une loi uniforme sur $[0, 1]$.

5. Calculer le nombre d'individus à générer pour chaque x_i de \mathcal{X}_1 :

$$G_i = G\check{r}_i.$$

6. Pour chaque x_i de \mathcal{X}_1 répéter G_i fois :

6.1 Choisir au hasard un de ces $kppv$ du groupe 1 $\implies x_i^{(1)}$

6.2 Générer un point au hasard entre x_i et $x_i^{(1)}$

$$x_{i,j} = x_i + \lambda(x_i^{(1)} - x_i)$$

où λ est générée selon une loi uniforme sur $[0, 1]$.

Sorties : les nouvelles données $x_{i,j}, i \in \mathcal{X}_1, j \in \{1, \dots, G_i\}$.

Exemple

```
> adasyn <- AdasynClassif(Y~X1+X2,dat=df,beta=1,k=5,dth = 0.95)
> summary(adasyn$Y)
##    0    1
## 80 78
```

Ici encore il faudra choisir les paramètres

- **beta** pour contrôler le niveau de ré-équilibrage ;

Exemple

```
> adasyn <- AdasynClassif(Y~X1+X2,dat=df,beta=1,k=5,dth = 0.95)
> summary(adasyn$Y)
##    0    1
## 80 78
```

Ici encore il faudra choisir les paramètres

- **beta** pour contrôler le niveau de ré-équilibrage ;
- **k** pour la taille des voisinages ;

Exemple

```
> adasyn <- AdasynClassif(Y~X1+X2,dat=df,beta=1,k=5,dth = 0.95)
> summary(adasyn$Y)
##    0    1
## 80 78
```

Ici encore il faudra choisir les paramètres

- **beta** pour contrôler le niveau de ré-équilibrage ;
- **k** pour la taille des voisinages ;
- la **distance** pour calculer les k ppv...

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

Random undersampling

- Comme le random oversampling mais dans l'autre sens :
ré-échantillonner la classe majoritaire de manière à obtenir un effectif proche de la classe minoritaire.

```
> under1 <- RandUnderClassif(Y~.,dat=df)
> under2 <- RandUnderClassif(Y~.,dat=df,C.perc=list("0"=0.5,"1"=1))
> summary(under1$Y)
##  0  1
## 20 20
> summary(under2$Y)
##  0  1
## 40 20
```

- **Idée** : supprimer les observations de la classe majoritaire qui se trouvent proches d'observations de la classe minoritaire [Tomek, 1976].

- **Idée** : supprimer les observations de la classe majoritaire qui se trouvent proches d'observations de la classe minoritaire [Tomek, 1976].

Tomek link

Soit x dans le groupe majo et y dans le groupe mino. (x, y) est un **T-link** si pour toute autre observation z

$$d(x, y) \leq d(x, z) \quad \text{et} \quad d(x, y) \leq d(y, z).$$

- **Idée** : supprimer les observations de la classe majoritaire qui se trouvent proches d'observations de la classe minoritaire [Tomek, 1976].

Tomek link

Soit x dans le groupe majo et y dans le groupe mino. (x, y) est un **T-link** si pour toute autre observation z

$$d(x, y) \leq d(x, z) \quad \text{et} \quad d(x, y) \leq d(y, z).$$

L'approche consiste à **supprimer les x qui ont un T-link.**

- **Idée** : supprimer les observations de la classe majoritaire qui se trouvent proches d'observations de la classe minoritaire [Tomek, 1976].

Tomek link

Soit x dans le groupe majo et y dans le groupe mino. (x, y) est un **T-link** si pour toute autre observation z

$$d(x, y) \leq d(x, z) \quad \text{et} \quad d(x, y) \leq d(y, z).$$

L'approche consiste à **supprimer les x qui ont un T-link**.

- Cette méthode ne permet **pas forcément de rééquilibrer** les deux classes.

- **Idée** : supprimer les observations de la classe majoritaire qui se trouvent proches d'observations de la classe minoritaire [Tomek, 1976].

Tomek link

Soit x dans le groupe majo et y dans le groupe mino. (x, y) est un **T-link** si pour toute autre observation z

$$d(x, y) \leq d(x, z) \quad \text{et} \quad d(x, y) \leq d(y, z).$$

L'approche consiste à **supprimer les x qui ont un T-link**.

- Cette méthode ne permet **pas forcément de rééquilibrer** les deux classes.
- Elle peut néanmoins permettre de **clarifier une future classification** en supprimant des **1 isolés au milieu des 0**, et donc de mieux détecter la classe minoritaire.

Exemple

- Il suffit d'utiliser **TomekClassif** :

```
> tomek1 <- TomekClassif(Y~.,dat=df)
> tomek2 <- TomekClassif(Y~.,dat=df,rem="maj")
```

Exemple

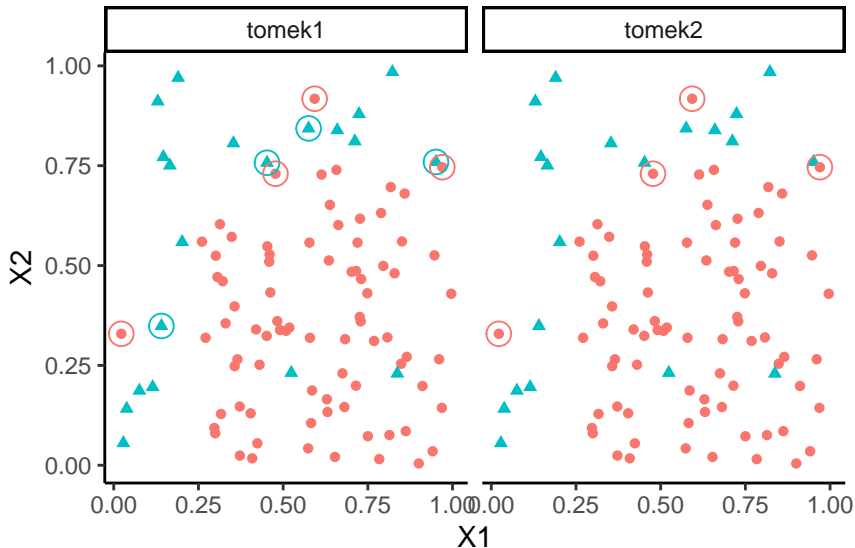
- Il suffit d'utiliser **TomekClassif** :

```
> tomek1 <- TomekClassif(Y~.,dat=df)
> tomek2 <- TomekClassif(Y~.,dat=df,rem="maj")
```

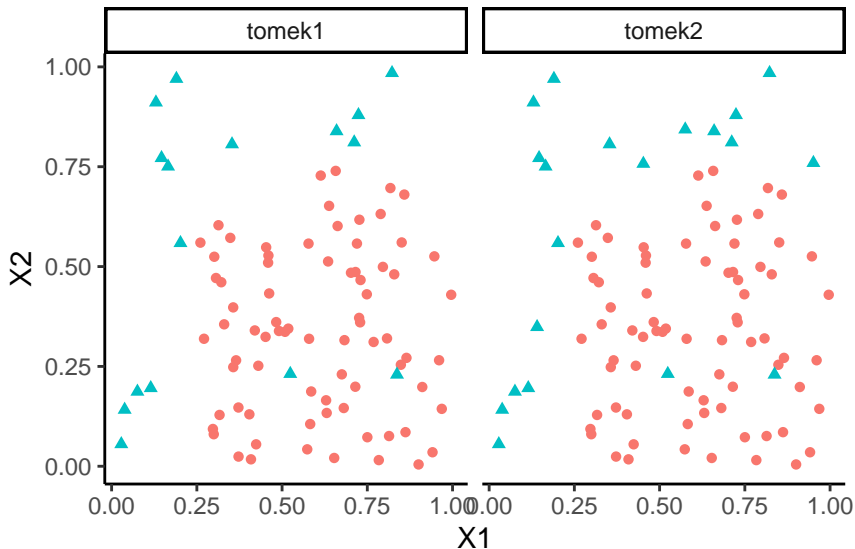
- On peut récupérer les **observations supprimées** avec

```
> tomek1[[2]]
## [1] 1 7 12 69 14 100 16 17
> tomek2[[2]]
## [1] 1 69 100 16
```

Visualisation des liens de Tomek



Suppression des liens



- Plusieurs algorithmes permettant de ré-équilibrer ont été présentés, il en existe d'autres **Condensed Nearest Neighbors** (CNN), **Edited Nearest Neighbors** (ENN)...
- Les idées sont proches, on pourra consulter la **vignette du package UBL**.

- Plusieurs algorithmes permettant de ré-équilibrer ont été présentés, il en existe d'autres **Condensed Nearest Neighbors** (CNN), **Edited Nearest Neighbors** (ENN)...
- Les idées sont proches, on pourra consulter la **vignette du package UBL**.
- Ces méthodes dépendent de **paramètres** : nombre de voisins, distances...

- Plusieurs algorithmes permettant de ré-équilibrer ont été présentés, il en existe d'autres **Condensed Nearest Neighbors** (CNN), **Edited Nearest Neighbors** (ENN)...
- Les idées sont proches, on pourra consulter la **vignette du package UBL**.
- Ces méthodes dépendent de **paramètres** : nombre de voisins, distances...
- Nécessité de définir des **stratégies qui permettent de choisir une méthode**.

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

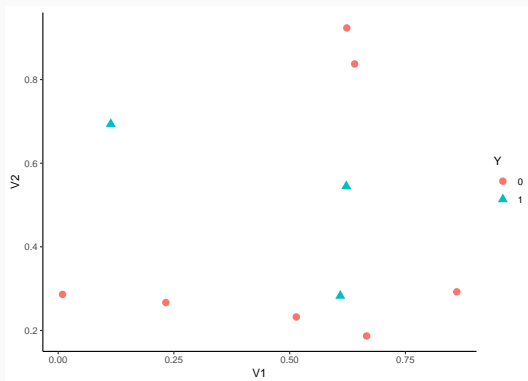
Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

- On illustre les fonctions du package sur le jeu de données suivants :



Random oversampling

```
> library(unbalanced)
> rand.over <- ubOver(X,Y)
> rand.over
## $X
##           V1           V2
## 1  0.609274733 0.2827336
## 1.1 0.609274733 0.2827336
## 1.2 0.609274733 0.2827336
## 2   0.514251141 0.2322259
## 3   0.622299405 0.5449748
## 4   0.232550506 0.2668208
## 5   0.640310605 0.8372956
## 6   0.666083758 0.1867228
## 7   0.113703411 0.6935913
## 7.1 0.113703411 0.6935913
## 7.2 0.113703411 0.6935913
## 8   0.009495756 0.2862233
## 9   0.860915384 0.2923158
## 10  0.623379442 0.9234335
##
## $Y
## [1] 1 1 1 0 1 0 0 0 1 1 1 0 0 0
## Levels: 0 1
```

Entrées :

1. **perc.over** : **perc.over**/100 est le nombre d'observations générés pour chaque individu de la classe minoritaire.

Entrées :

1. **perc.over** : **perc.over**/100 est le nombre d'observations générés pour chaque individu de la classe minoritaire.
2. **perc.under** : **perc.under**/100 est le nombre d'observations choisis aléatoirement dans la classe majoritaire pour chaque smoted (nouvelles) observations.

Entrées :

1. **perc.over** : **perc.over**/100 est le nombre d'observations générés pour chaque individu de la classe minoritaire.
2. **perc.under** : **perc.under**/100 est le nombre d'observations choisis aléatoirement dans la classe majoritaire pour chaque smoted (nouvelles) observations.
3. **k** : nombre de voisins à considérer pour générer les nouveaux individus (plus petit que le nombre d'observations de la classe minoritaire).

- Pour chaque individu x de la classe minoritaire,
 1. Calculer ses k plus proches voisins ;
 2. Répéter **perc.over**/100 fois
 - 2.1 Choisir au hasard un de ses k plus proches voisins : \tilde{x} ;

- Pour chaque individu x de la classe minoritaire,
 1. Calculer ses k plus proches voisins ;
 2. Répéter **perc.over**/100 fois
 - 2.1 Choisir au hasard un de ses k plus proches voisins : \tilde{x} ;
 - 2.2 Générer un nouveau point de la classe minoritaire entre x et \tilde{x} ;

- Pour chaque individu x de la classe minoritaire,
 1. Calculer ses k plus proches voisins ;
 2. Répéter **perc.over**/100 fois
 - 2.1 Choisir au hasard un de ses k plus proches voisins : \tilde{x} ;
 - 2.2 Générer un nouveau point de la classe minoritaire entre x et \tilde{x} ;
 - 2.3 Choisir au hasard **perc.under**/100 individus dans la classe majoritaire.

- Pour chaque individu x de la classe minoritaire,
 1. Calculer ses k plus proches voisins ;
 2. Répéter **perc.over**/100 fois
 - 2.1 Choisir au hasard un de ses k plus proches voisins : \tilde{x} ;
 - 2.2 Générer un nouveau point de la classe minoritaire entre x et \tilde{x} ;
 - 2.3 Choisir au hasard **perc.under**/100 individus dans la classe majoritaire.
- **Sortie** : les individus de la classe minoritaire + ceux créés en 2.2 (classe mino.) + ceux créés en 2.3 (classe majo).

Exemple

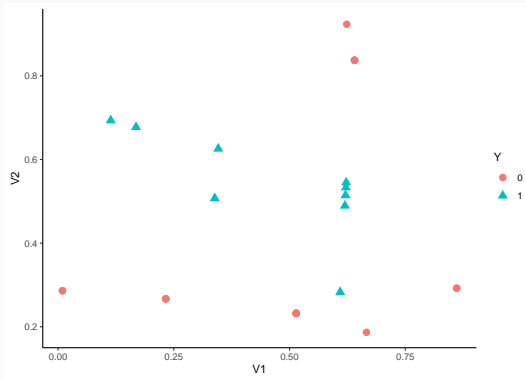
```
> set.seed(123)
> rand.smote <- ubSMOTE(X,Y,k=2,perc.over = 200,perc.under = 300)
> summary(rand.smote$Y)
```

Exemple

```
> set.seed(123)
> rand.smote <- ubSMOTE(X,Y,k=2,perc.over = 200,perc.under = 300)
> summary(rand.smote$Y)
```

```
##  0  1
```

```
## 18  9
```



Random undersampling

```
> set.seed(123)
> rand.under <- ubUnder(X,Y)
> rand.under
## $X
##           V1           V2
## 1  0.609274733 0.2827336
## 3  0.622299405 0.5449748
## 6  0.666083758 0.1867228
## 7  0.113703411 0.6935913
## 8  0.009495756 0.2862233
## 10 0.623379442 0.9234335
##
## $Y
## [1] 1 1 0 1 0 0
## Levels: 0 1
##
## $id.rm
## [1] 2 4 5 9
```

```

> ubTomek(X,Y)
## Instances removed 2 : 28.57 % of 0 class ; 20 % of training ; Time needed 0
## $X
##           V1           V2
## 1  0.6092747 0.2827336
## 3  0.6222994 0.5449748
## 4  0.2325505 0.2668208
## 5  0.6403106 0.8372956
## 6  0.6660838 0.1867228
## 7  0.1137034 0.6935913
## 9  0.8609154 0.2923158
## 10 0.6233794 0.9234335
##
## $Y
## [1] 1 1 0 0 0 1 0 0
## Levels: 0 1
##
## $id.rm
## [1] 2 8

```

Données déséquilibrées en logistique

Le problème du déséquilibre

Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

Critères de performance

Critères basés sur des règles

Critères basés sur des scores

Ré-échantillonnage

Oversampling

Undersampling

Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

Approche "classique" : minimisation de risque empirique

Racing

- Plusieurs façons d'appréhender des données déséquilibrées.

Questions

- Faut-il ré-équilibrer ?
- Quelle méthode pour ré-équilibrer ?
- Quel algorithme utiliser ensuite ? (une méthode de ré-équilibrage peut être pertinente pour un algo et ne pas l'être pour un autre)

- Plusieurs façons d'appréhender des données déséquilibrées.

Questions

- Faut-il ré-équilibrer ?
- Quelle méthode pour ré-équilibrer ?
- Quel algorithme utiliser ensuite ? (une méthode de ré-équilibrage peut être pertinente pour un algo et ne pas l'être pour un autre)
- Comment choisir la meilleure approche ?

- Plusieurs façons d'appréhender des données déséquilibrées.

Questions

- Faut-il ré-équilibrer ?
 - Quelle méthode pour ré-équilibrer ?
 - Quel algorithme utiliser ensuite ? (une méthode de ré-équilibrage peut être pertinente pour un algo et ne pas l'être pour un autre)
 - Comment choisir la meilleure approche ?
-
- Comme souvent en machine learning, il n'existe pas de méthode universelle pour choisir un algorithme, il va falloir faire des choix !

Données déséquilibrées en logistique

- Le problème du déséquilibre

- Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

- Critères de performance

 - Critères basés sur des règles

 - Critères basés sur des scores

- Ré-échantillonnage

 - Oversampling

 - Undersampling

 - Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

- Approche "classique" : minimisation de risque empirique

- Racing

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).
- Un algorithme est une **fonction de prévision** $g_n(x) = g_n(x, \mathcal{D}_n)$ qui fournit une prévision pour un individu x .

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).
- Un algorithme est une **fonction de prévision** $g_n(x) = g_n(x, \mathcal{D}_n)$ qui fournit une prévision pour un individu x .
- L'algorithme g_n peut inclure tout un **tas d'étapes** :

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).
- Un algorithme est une **fonction de prévision** $g_n(x) = g_n(x, \mathcal{D}_n)$ qui fournit une prévision pour un individu x .
- L'algorithme g_n peut inclure tout un **tas d'étapes** :
 1. gestion des **données manquantes** ;

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).
- Un algorithme est une **fonction de prévision** $g_n(x) = g_n(x, \mathcal{D}_n)$ qui fournit une prévision pour un individu x .
- L'algorithme g_n peut inclure tout un **tas d'étapes** :
 1. gestion des **données manquantes** ;
 2. procédure de **choix de variables** ;

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).
- Un algorithme est une **fonction de prévision** $g_n(x) = g_n(x, \mathcal{D}_n)$ qui fournit une prévision pour un individu x .
- L'algorithme g_n peut inclure tout un **tas d'étapes** :
 1. gestion des **données manquantes** ;
 2. procédure de **choix de variables** ;
 3. **ré-équilibrage** de \mathcal{D}_n ;

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).
- Un algorithme est une **fonction de prévision** $g_n(x) = g_n(x, \mathcal{D}_n)$ qui fournit une prévision pour un individu x .
- L'algorithme g_n peut inclure tout un **tas d'étapes** :
 1. gestion des **données manquantes** ;
 2. procédure de **choix de variables** ;
 3. **ré-équilibrage** de \mathcal{D}_n ;
 4. **entraînement d'un modèle** (ridge, lasso, arbres...) avec **sélection automatique des paramètres** (validation croisée, oob...) ;
 5. ...

- Les techniques vues précédemment ne sortent pas du cadre classique du **machine learning**.
- **Données** : $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ (équilibrées ou non).
- Un algorithme est une **fonction de prévision** $g_n(x) = g_n(x, \mathcal{D}_n)$ qui fournit une prévision pour un individu x .
- L'algorithme g_n peut inclure tout un **tas d'étapes** :
 1. gestion des **données manquantes** ;
 2. procédure de **choix de variables** ;
 3. **ré-équilibrage** de \mathcal{D}_n ;
 4. **entraînement d'un modèle** (ridge, lasso, arbres...) avec **sélection automatique des paramètres** (validation croisée, oob...) ;
 5. ...

Performance de g_n

Elle peut s'évaluer à l'aide d'un (bon) **critère de prévision** calculé à partir de procédure de **ré-échantillonnage** (validation hold out, validation croisée).

- Un **critère de prévision** ou **fonction de perte** (voire **risque empirique**) est une fonction qui mesure une **perte** (ou erreur) entre n prévisions $\hat{y}_i, i = 1 \dots, n$ et n observations $y_i, i = 1, \dots, n$.
- Mathématiquement, c'est donc une fonction :

$$\begin{aligned}\mathcal{R} : \mathbb{R}^n \times \mathbb{R}^n &\rightarrow \mathbb{R} \\ (y_1^n, \hat{y}_1^n) &\mapsto \mathcal{R}(y_1^n, \hat{y}_1^n).\end{aligned}$$

Critère de prévision

- Un **critère de prévision** ou **fonction de perte** (voire **risque empirique**) est une fonction qui mesure une **perte** (ou erreur) entre n prévisions $\hat{y}_i, i = 1 \dots, n$ et n observations $y_i, i = 1, \dots, n$.
- Mathématiquement, c'est donc une fonction :

$$\mathcal{R} : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$$
$$(y_1^n, \hat{y}_1^n) \mapsto \mathcal{R}(y_1^n, \hat{y}_1^n).$$

Exemples

Erreur de classification, AUC, tous les critères vus dans la section "critères de performance".

- Afin d'éviter des problèmes de **biais**, il faut évaluer ces critères sur des **données non utilisées pour construire l'algorithme**.

- Afin d'éviter des problèmes de **biais**, il faut évaluer ces critères sur des **données non utilisées pour construire l'algorithme**.
- On utilise souvent des procédures de **ré-échantillonnage** qui consistent à **séparer les données en plusieurs blocs de manière** :
 - **construire l'algorithme** sur certains blocs ;
 - **calculer les prévisions** sur les autres.

- Afin d'éviter des problèmes de **biais**, il faut évaluer ces critères sur des **données non utilisées pour construire l'algorithme**.
- On utilise souvent des procédures de **ré-échantillonnage** qui consistent à **séparer les données en plusieurs blocs de manière** :
 - **construire l'algorithme** sur certains blocs ;
 - **calculer les prévisions** sur les autres.
- Nous rappelons la **validation croisée K -folds**.

Algorithme - CV

Entrées. \mathcal{D}_n : données, K un entier qui divise n , un algorithme g_n , un critère de prévision \mathcal{R} ;

1. Construire une partition $\{\mathcal{I}_1, \dots, \mathcal{I}_K\}$ de $\{1, \dots, n\}$;
2. Pour $k = 1, \dots, K$
 - 2.1 $\mathcal{I}_{\text{train}} = \{1, \dots, n\} \setminus \mathcal{I}_k$ et $\mathcal{I}_{\text{test}} = \mathcal{I}_k$;
 - 2.2 Construire l'algorithme de prédiction sur $\mathcal{D}_{n,\text{train}} = \{(X_i, Y_i) : i \in \mathcal{I}_{\text{train}}\}$, on le note $g_{n,k}(\cdot) = g_n(\cdot, \mathcal{D}_{n,\text{train}})$;
 - 2.3 En déduire $\hat{Y}_i = g_{n,k}(X_i)$ pour $i \in \mathcal{I}_{\text{test}}$;
3. **Retourner**

$$\mathcal{R}(g_n) = \mathcal{R}(Y_1^n, \hat{Y}_1^n).$$

- On remarque que l'algorithme g_n s'applique **uniquement aux données d'apprentissage**.

- On remarque que l'algorithme g_n s'applique **uniquement aux données d'apprentissage**.
- Si on utilise une **technique de ré-équilibrage** dans l'algorithme, il faut donc prendre garde dans la validation croisée à ne **ré-équilibrer que l'échantillon d'apprentissage**, pas le test !

- On remarque que l'algorithme g_n s'applique **uniquement aux données d'apprentissage**.
- Si on utilise une **technique de ré-équilibrage** dans l'algorithme, il faut donc prendre garde dans la validation croisée à ne **ré-équilibrer que l'échantillon d'apprentissage**, pas le test !

Attention

Il ne faut surtout **pas ré-équilibrer tout l'échantillon** avant de faire la **validation croisée** !

Choisir un algorithme

- On peut utiliser cette méthode pour **comparer plusieurs algorithmes** g_1, \dots, g_ℓ :
 1. On calcule $\mathcal{R}(g_1), \dots, \mathcal{R}(g_\ell)$;
 2. On choisit celui qui **optimise** le critère.

Choisir un algorithme

- On peut utiliser cette méthode pour comparer plusieurs algorithmes g_1, \dots, g_ℓ :
 1. On calcule $\mathcal{R}(g_1), \dots, \mathcal{R}(g_\ell)$;
 2. On choisit celui qui optimise le critère.
- Ces algorithmes peuvent inclure (ou non) des méthodes de ré-équilibrage.

Choisir un algorithme

- On peut utiliser cette méthode pour **comparer plusieurs algorithmes** g_1, \dots, g_ℓ :
 1. On calcule $\mathcal{R}(g_1), \dots, \mathcal{R}(g_\ell)$;
 2. On choisit celui qui **optimise** le critère.
- Ces algorithmes peuvent inclure (ou non) des **méthodes de ré-équilibrage**.
- Pour justifier de l'intérêt de ré-équilibrer, il conviendra de toujours **comparer ces méthodes au cas non ré-équilibré**.

Choisir un algorithme

- On peut utiliser cette méthode pour **comparer plusieurs algorithmes** g_1, \dots, g_ℓ :
 1. On calcule $\mathcal{R}(g_1), \dots, \mathcal{R}(g_\ell)$;
 2. On choisit celui qui **optimise** le critère.
- Ces algorithmes peuvent inclure (ou non) des **méthodes de ré-équilibrage**.
- Pour justifier de l'intérêt de ré-équilibrer, il conviendra de toujours **comparer ces méthodes au cas non ré-équilibré**.

Règle finale

Une fois l'algorithme g_k choisit, on le **recalcule sur toutes les données** : $\hat{g}_k(\cdot) = \hat{g}_k(\cdot, \mathcal{D}_n)$ et on calcule les prévisions sur de nouveaux individus...

Données déséquilibrées en logistique

- Le problème du déséquilibre

- Le schéma d'échantillonnage rétrospectif

Stratégies classiques des données déséquilibrées

- Critères de performance

 - Critères basés sur des règles

 - Critères basés sur des scores

- Ré-échantillonnage

 - Oversampling

 - Undersampling

 - Annexe : le package unbalanced

Choisir une méthode pour des données déséquilibrées

- Approche "classique" : minimisation de risque empirique

- Racing

- Les techniques de validation croisée présentées précédemment peuvent se révéler **couteuses en temps de calcul**, notamment avec des **données déséquilibrées**.

- Les techniques de validation croisée présentées précédemment peuvent se révéler **couteuses en temps de calcul**, notamment avec des **données déséquilibrées**.
- Elles nécessitent en effet de tester **tous les algorithmes** avec toutes les **techniques de ré-équilibrage** !

- Les techniques de validation croisée présentées précédemment peuvent se révéler **couteuses en temps de calcul**, notamment avec des **données déséquilibrées**.
- Elles nécessitent en effet de tester **tous les algorithmes** avec toutes les **techniques de ré-équilibrage** !
- Il peut être préférable de **choisir au préalable une méthode de ré-équilibrage** pour chaque méthode.

- Les techniques de validation croisée présentées précédemment peuvent se révéler **couteuses en temps de calcul**, notamment avec des **données déséquilibrées**.
- Elles nécessitent en effet de tester **tous les algorithmes** avec toutes les **techniques de ré-équilibrage** !
- Il peut être préférable de **choisir au préalable une méthode de ré-équilibrage** pour chaque méthode.
- Il existe un algorithme, appelé **racing** [Birattari et al., 2002], qui permet de choisir une méthode de ré-équilibrage appropriée **sans forcément balayer toutes les données**.

Idée

- Tester en parallèle plusieurs méthodes de ré-équilibrage sur une partie des données.

Idée

- Tester en parallèle plusieurs méthodes de ré-équilibrage sur une partie des données.
- Utiliser un test statistique pour déterminer si une (ou plusieurs) méthode est significativement pire que les autres.

Idée

- Tester en parallèle plusieurs méthodes de ré-équilibrage sur une partie des données.
- Utiliser un test statistique pour déterminer si une (ou plusieurs) méthode est significativement pire que les autres.
- Si c'est le cas, ces méthodes sont supprimées de la course.

Idée

- Tester en parallèle plusieurs méthodes de ré-équilibrage sur une partie des données.
- Utiliser un test statistique pour déterminer si une (ou plusieurs) méthode est significativement pire que les autres.
- Si c'est le cas, ces méthodes sont supprimées de la course.
- On continue avec les méthodes restantes sur une autre partie des données.

Idée

- Tester en parallèle plusieurs méthodes de ré-équilibrage sur une partie des données.
- Utiliser un test statistique pour déterminer si une (ou plusieurs) méthode est significativement pire que les autres.
- Si c'est le cas, ces méthodes sont supprimées de la course.
- On continue avec les méthodes restantes sur une autre partie des données.
- On s'arrête si il reste une seule méthode ou si on a balayé toutes les données.

Quelques notations

- On considère une **méthode particulière** (par exemple une forêt aléatoire) ;

Quelques notations

- On considère une **méthode particulière** (par exemple une forêt aléatoire) ;
- Un **critère de performance** (par exemple AUC).

Quelques notations

- On considère une **méthode particulière** (par exemple une forêt aléatoire);
- Un **critère de performance** (par exemple AUC).
- B **algos de ré-équilibrage** $\implies g_j, j = 1, \dots, B$ la méthode appliquée avec l'algo j ;

Quelques notations

- On considère une **méthode particulière** (par exemple une forêt aléatoire);
- Un **critère de performance** (par exemple AUC).
- B **algos de ré-équilibrage** $\implies g_j, j = 1, \dots, B$ la méthode appliquée avec l'algo j ;
- $\mathcal{I}_1, \dots, \mathcal{I}_K$ une **partition** de $\{1, \dots, n\}$ and K **blocs**.

Etape 1

1. Entraîner les algorithmes sur les données du premier bloc.

Etape 1

1. **Entraîner les algorithmes** sur les données du premier bloc.
2. Prédire sur les autres blocs et en déduire la **valeur du critère** pour chaque algorithme.

Etape 1

1. **Entraîner les algorithmes** sur les données du premier bloc.
2. Prédire sur les autres blocs et en déduire la **valeur du critère** pour chaque algorithme.
3. On note R_{1j} le **rang** de l'algorithme j .

Etape 2

1. **Entraîner les algorithmes** sur les données du second bloc.

Etape 2

1. **Entraîner les algorithmes** sur les données du second bloc.
2. Prédire sur les autres blocs et en déduire la **valeur du critère** pour chaque algorithme.

Etape 2

1. **Entraîner les algorithmes** sur les données du second bloc.
2. Prédire sur les autres blocs et en déduire la **valeur du critère** pour chaque algorithme.
3. On note R_{2j} le **rang** de l'algorithme j .

Etape 2

1. **Entraîner les algorithmes** sur les données du second bloc.
2. Prédire sur les autres blocs et en déduire la **valeur du critère** pour chaque algorithme.
3. On note R_{2j} le **rang** de l'algorithme j .
4. Tester si les **rangs des algorithmes** dans les deux premiers blocs sont **indépendants**.

Etape 2

1. **Entraîner les algorithmes** sur les données du second bloc.
2. Prédire sur les autres blocs et en déduire la **valeur du critère** pour chaque algorithme.
3. On note R_{2j} le **rang** de l'algorithme j .
4. Tester si les **rangs des algorithmes** dans les deux premiers blocs sont **indépendants**.
5. Si oui, aller à l'étape 3, si non **enlever les algorithmes significativement inférieurs au meilleur**.

Etape 2

1. **Entraîner les algorithmes** sur les données du second bloc.
2. Prédire sur les autres blocs et en déduire la **valeur du critère** pour chaque algorithme.
3. On note R_{2j} le **rang** de l'algorithme j .
4. Tester si les **rangs des algorithmes** dans les deux premiers blocs sont **indépendants**.
5. Si oui, aller à l'étape 3, si non **enlever les algorithmes significativement inférieurs au meilleur**.

Itération

Cette étape est **répétée sur les blocs suivants** jusqu'à ce qu'il reste un **seul algorithme** ou que **tous les blocs soient balayés**.

- Deux tests sont effectués à chaque étape.

- Deux tests sont effectués à chaque étape.
- Il existe plusieurs procédures

- Deux tests sont effectués à chaque étape.
- Il existe plusieurs procédures
- Nous détaillons ici le test de Friedman basés sur les rangs des algorithmes dans chaque bloc.

Test de Friedman

- On se place à l'étape k et on note $R_j = \sum_{\ell=1}^k R_{\ell j}$ la somme des rangs de l'algorithme j .
- La statistique de test est donnée par :

$$T = \frac{(B-1) \sum_{j=1}^B \left(R_j - \frac{k(B+1)}{2} \right)^2}{\sum_{\ell=1}^k \sum_{j=1}^B R_{\ell j}^2 - \frac{kB(B+1)^2}{4}}.$$

Test de Friedman

- On se place à l'étape k et on note $R_j = \sum_{\ell=1}^k R_{\ell j}$ la somme des rangs de l'algorithme j .
- La statistique de test est donnée par :

$$T = \frac{(B-1) \sum_{j=1}^B \left(R_j - \frac{k(B+1)}{2} \right)^2}{\sum_{\ell=1}^k \sum_{j=1}^B R_{\ell j}^2 - \frac{kB(B+1)^2}{4}}.$$

- Sous l'hypothèse nulle que les rangs des méthodes sont identiques à chaque étape, cette statistique suit approximativement une loi du χ^2 à $B-1$ ddl.

Test de Friedman

- On se place à l'étape k et on note $R_j = \sum_{\ell=1}^k R_{\ell j}$ la somme des rangs de l'algorithme j .
- La statistique de test est donnée par :

$$T = \frac{(B-1) \sum_{j=1}^B \left(R_j - \frac{k(B+1)}{2} \right)^2}{\sum_{\ell=1}^k \sum_{j=1}^B R_{\ell j}^2 - \frac{kB(B+1)^2}{4}}.$$

- Sous l'hypothèse nulle que les rangs des méthodes sont identiques à chaque étape, cette statistique suit approximativement une loi du χ^2 à $B-1$ ddl.
- On rejettera donc l'hypothèse nulle si cette statistique dépasse le quantile d'ordre $1-\alpha$ de cette loi du χ^2 .

- Lorsqu'au cours d'une étape l'hypothèse nulle du test précédent est **rejetée**, il faut décider des **méthodes à supprimer**.

- Lorsqu'au cours d'une étape l'hypothèse nulle du test précédent est **rejetée**, il faut décider des **méthodes à supprimer**.
- On teste alors **chaque méthode contre la meilleure** à l'aide d'un nouveau test toujours basé sur les **rangs**.

- Lorsqu'au cours d'une étape l'hypothèse nulle du test précédent est **rejetée**, il faut décider des **méthodes à supprimer**.
- On teste alors **chaque méthode contre la meilleure** à l'aide d'un nouveau test toujours basé sur les **rangs**.
- La statistique est basé sur la **différence entre les R_j et R_{best}** (voir [Birattari et al., 2002]).

- Lorsqu'au cours d'une étape l'hypothèse nulle du test précédent est **rejetée**, il faut décider des **méthodes à supprimer**.
- On teste alors **chaque méthode contre la meilleure** à l'aide d'un nouveau test toujours basé sur les **rangs**.
- La statistique est basé sur la **différence entre les R_j et R_{best}** (voir [Birattari et al., 2002]).
- Les méthodes qui ont des classement **significativement inférieurs au meilleur** sont **supprimées**.

- Lorsqu'au cours d'une étape l'hypothèse nulle du test précédent est **rejetée**, il faut décider des **méthodes à supprimer**.
- On teste alors **chaque méthode contre la meilleure** à l'aide d'un nouveau test toujours basé sur les **rangs**.
- La statistique est basé sur la **différence entre les R_j et R_{best}** (voir [Birattari et al., 2002]).
- Les méthodes qui ont des classement **significativement inférieurs au meilleur** sont **supprimées**.
- Sur **R**, on peut utiliser la fonction **ubRacing**.

```

> set.seed(1234)
> ubConf <- list(percOver=200, percUnder=200, k=2, perc=50, method="percPos", w=NULL)
> library(randomForest)
> results <- ubRacing(Class ~., ubIonosphere, "randomForest", positive=1, metric="auc", ubConf=ubConf,
+                      ntree=500)

## Racing for unbalanced methods selection in 10 fold CV
## Number of candidates.....9
## Max number of folds in the CV.....10
## Max number of experiments.....100
## Statistical test.....Friedman test
##
##                               Markers:
##                               x No test is performed.
##                               - The test is performed and
##                               some candidates are discarded.
##                               = The test is performed but
##                               no candidate is discarded.
##
## +-----+-----+-----+-----+-----+
## | |      Fold|      Alive|      Best| Mean best| Exp so far|
## +-----+-----+-----+-----+-----+
## |x|        1|        9|        1|    0.9301|        9|
## |=|        2|        9|        1|    0.9395|       18|
## |-|        3|        5|        7|    0.9472|       27|
## |=|        4|        5|        7|    0.9405|       32|
## |=|        5|        5|        7|    0.9441|       37|
## |=|        6|        5|        7|    0.9354|       42|
## |=|        7|        5|        7|    0.9401|       47|
## |=|        8|        5|        7|    0.9401|       52|
## |-|        9|        2|        7|    0.9404|       57|
## |=|       10|        2|        7|    0.9379|       59|
## +-----+-----+-----+-----+-----+
## Selected candidate: ubENN      metric: auc      mean value: 0.9379

```

- On peut visualiser les valeurs d'AUC sur chaque bloc avec

```
> results$Race
##          unbal      ubOver      ubUnder      ubSMOTE      ubOSS      ubCNN
## [1,] 0.9301235 0.9105844 0.8177736 0.9063130 0.9106501 0.9110006
## [2,] 0.9488862 0.9483848 0.9378787 0.9451807 0.7160077 0.8298313
## [3,] 0.9608072 0.9623628 0.9310777 0.9636375 0.9200156 0.5111053
## [4,] 0.9143868 0.9299766          NA 0.8972772          NA          NA
## [5,] 0.9551633 0.9535131          NA 0.9466302          NA          NA
## [6,] 0.9357521 0.9266111          NA 0.9456531          NA          NA
## [7,] 0.9670648 0.9489298          NA 0.9577793          NA          NA
## [8,] 0.9415188 0.9362875          NA 0.9282881          NA          NA
## [9,] 0.9398186 0.9321243          NA 0.9106325          NA          NA
## [10,] 0.9226540          NA          NA          NA          NA          NA
##          ubENN      ubNCL      ubTomek
## [1,] 0.9280426 0.9059406 0.9223473
## [2,] 0.9489080 0.9107633 0.9233402
## [3,] 0.9646314 0.9419238 0.9553842
## [4,] 0.9205098          NA 0.9140177
## [5,] 0.9584419          NA 0.9605046
## [6,] 0.8919793          NA 0.9337546
## [7,] 0.9684816          NA 0.9569946
## [8,] 0.9401892          NA 0.9380749
## [9,] 0.9421509          NA 0.9031126
## [10,] 0.9159949          NA          NA
> as_tibble(results$Race) %>% select(unbal,ubENN) %>% summarize_all(mean)
## # A tibble: 1 x 2
##   unbal ubENN
##   <dbl> <dbl>
## 1 0.942 0.938
```

- On peut remarquer qu'en terme de moyenne d'AUC, ce n'est pas l'algorithme **ubENN** qui est le meilleur.

- On peut remarquer qu'en terme de moyenne d'AUC, ce n'est pas l'algorithme **ubENN** qui est le meilleur.
- La décision se fait en effet sur les **rangs**.

```
> df <- results$Race
> df[is.na(df)] <- 0
> df <- 1-df
> a <- df %>% apply(1,rank) %>% apply(1,sum)
> a[c(1,7)]
## unbal ubENN
##      21      19
```

Conclusion

- Problème difficile !

Conclusion

- Problème difficile !
- Nécessite de bien connaître les données et la problématique (pour définir le (ou les) bon(s) critère(s)).

Conclusion

- Problème **difficile** !
- Nécessite de **bien connaître** les **données** et la **problématique** (pour définir le (ou les) bon(s) critère(s)).
- Nécessite une bonne **maîtrise machine learning** afin de ne pas partir dans tous les sens !

Conclusion

- Problème **difficile** !
- Nécessite de **bien connaître** les **données** et la **problématique** (pour définir le (ou les) bon(s) critère(s)).
- Nécessite une bonne **maitrise machine learning** afin de ne pas partir dans tous les sens !
- Il faut bien structurer la démarche et **faire des (bons) choix**...

Conclusion

- Problème **difficile** !
- Nécessite de **bien connaître** les **données** et la **problématique** (pour définir le (ou les) bon(s) critère(s)).
- Nécessite une bonne **maitrise machine learning** afin de ne pas partir dans tous les sens !
- Il faut bien structurer la démarche et **faire des (bons) choix**...

En effet

Même si on optimise souvent, **tout n'est pas automatique**...

-  Birattari, M., Stützle, T., Paquete, L., and Varrentrapp, K. (2002).
A racing algorithm for configuring metaheuristics.
In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, pages 11–18.
-  Chawla, V., Bowyer, K., Hall, L., and Kegelmeyer, W. (2002).
Smote : Synthetic minority over-sampling technique.
Journal of Artificial Intelligence Research, 16 :321–357.
-  Cléménçon, S., Lugosi, G., and Vayatis, N. (2008).
Ranking and empirical minimization of u-statistics.
The Annals of Statistics, 36(2) :844–874.



He, H., Bai, Y., Garcia, E., and Li, S. (2008).

Adasyn : Adaptive synthetic sampling approach for imbalanced learning.

In 008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pages 1322–1328.



Tomek, I. (1976).

Two modifications of cnn.

IEEE Transactions on Systems, Man, and Cybernetics, 6 :769–772.