Apprentissage de modèles à base de règles pour la prédiction de la résistance aux antibiotiques

Alexandre Drouin

Département d'informatique et de génie logiciel, Université Laval
Département de médecine moléculaire, Université Laval

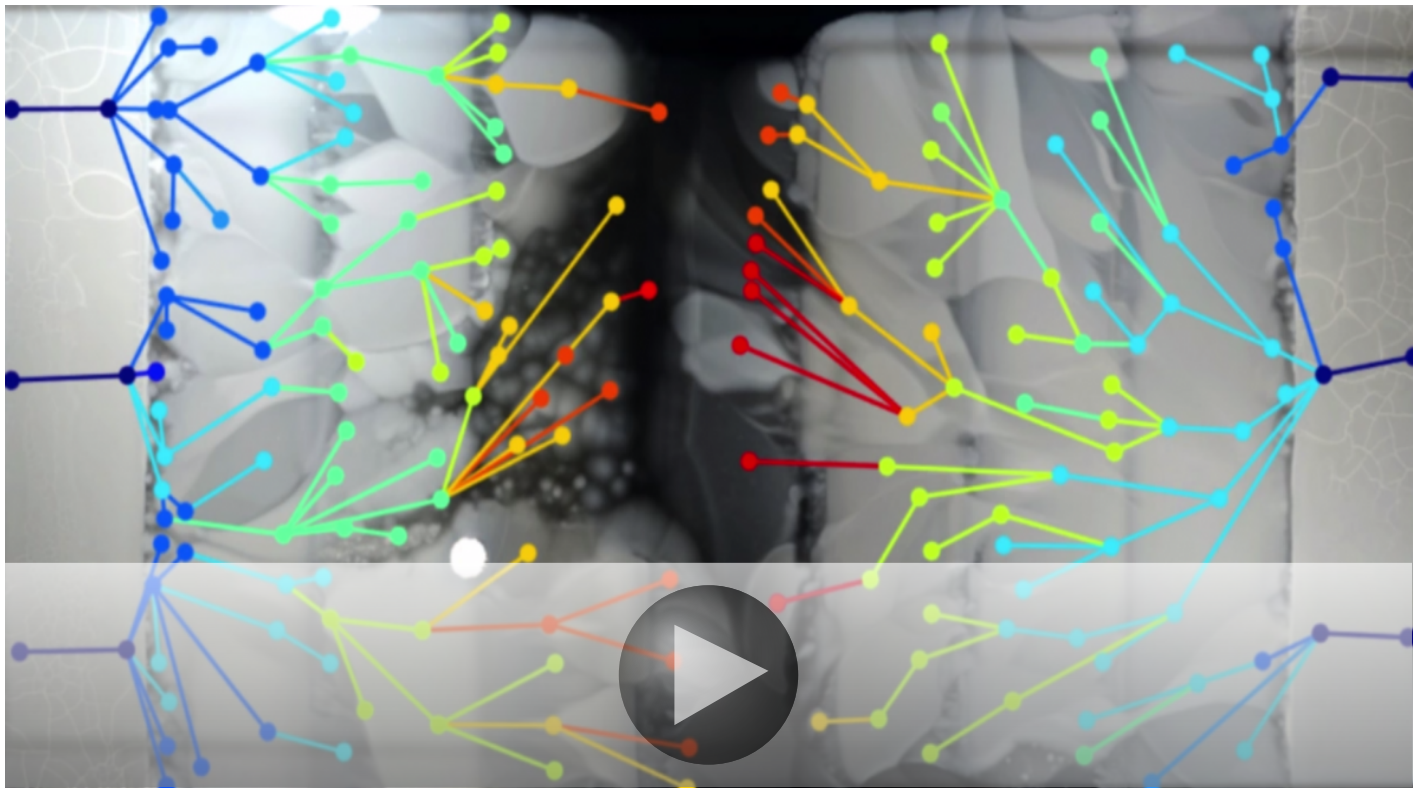
Sainte-Foy, Québec
17 mars 2017

Plan

Introduction

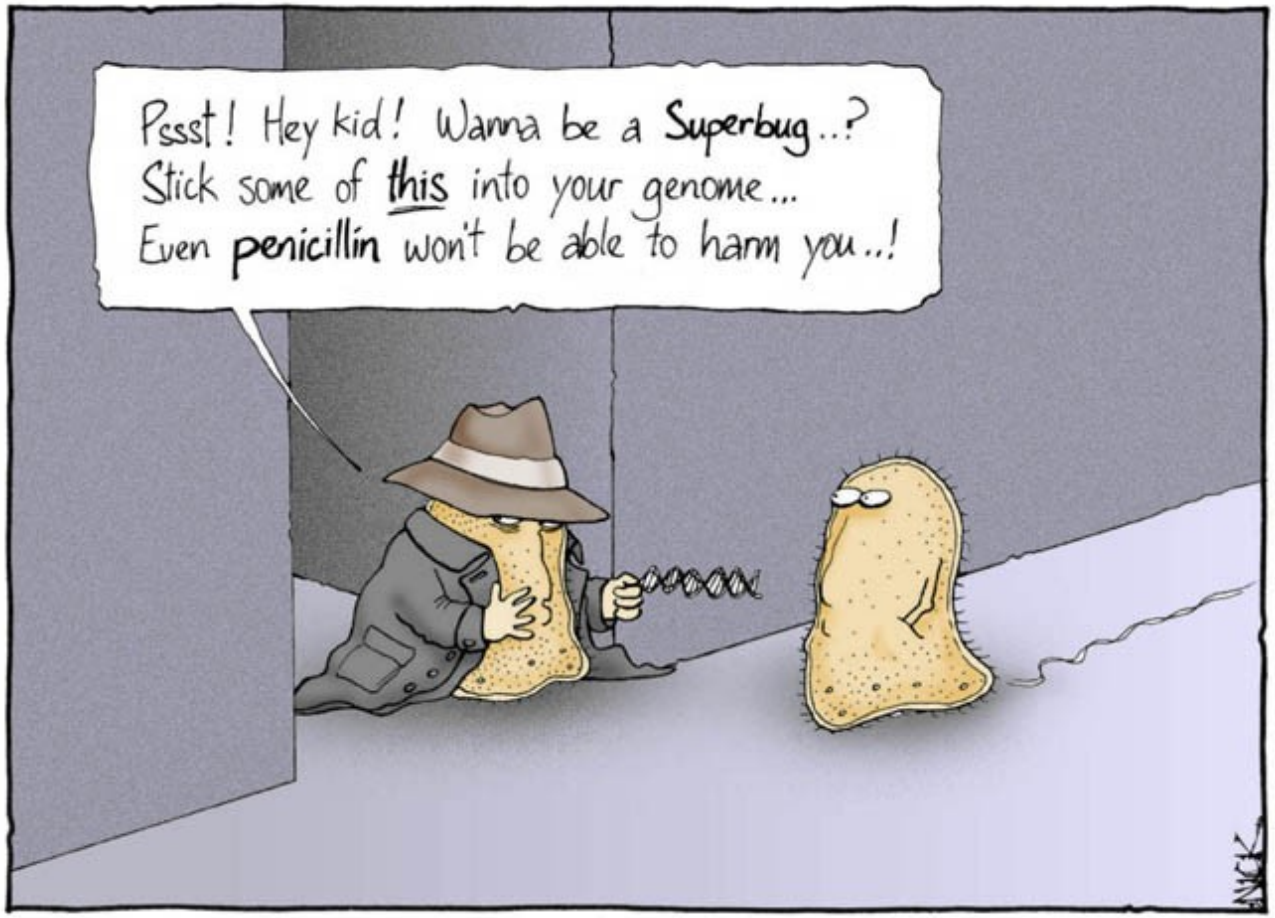
Qu'est-ce que la résistance aux antibiotiques ?

- État dans lequel une bactérie est capable de survivre en présence d'un antibiotique
- Généralement atteint via des modifications génomiques :
  - ex. : mutations, transfert horizontal de gènes



Credits: Harvard Medical School (Kishony Lab) and Technion, 2016

Qu'est-ce que la résistance aux antibiotiques ?



It was on a short-cut through the hospital kitchens that Albert was first approached by a member of the Antibiotic Resistance.

Source : `http://www.med.uottawa.ca/sim/data/AntibioticResistance_e.htm`

Pourquoi prédire la résistance aux antibiotiques ?

- Profilage rapide des antibiotiques pouvant être utilisés pour traiter une infection
- Plan de traitement personnalisé pour chaque patient, raccourcir/éviter la thérapie empirique (médecine personnalisée)
- Extraire des nouvelles connaissances biologiques des modèles obtenus

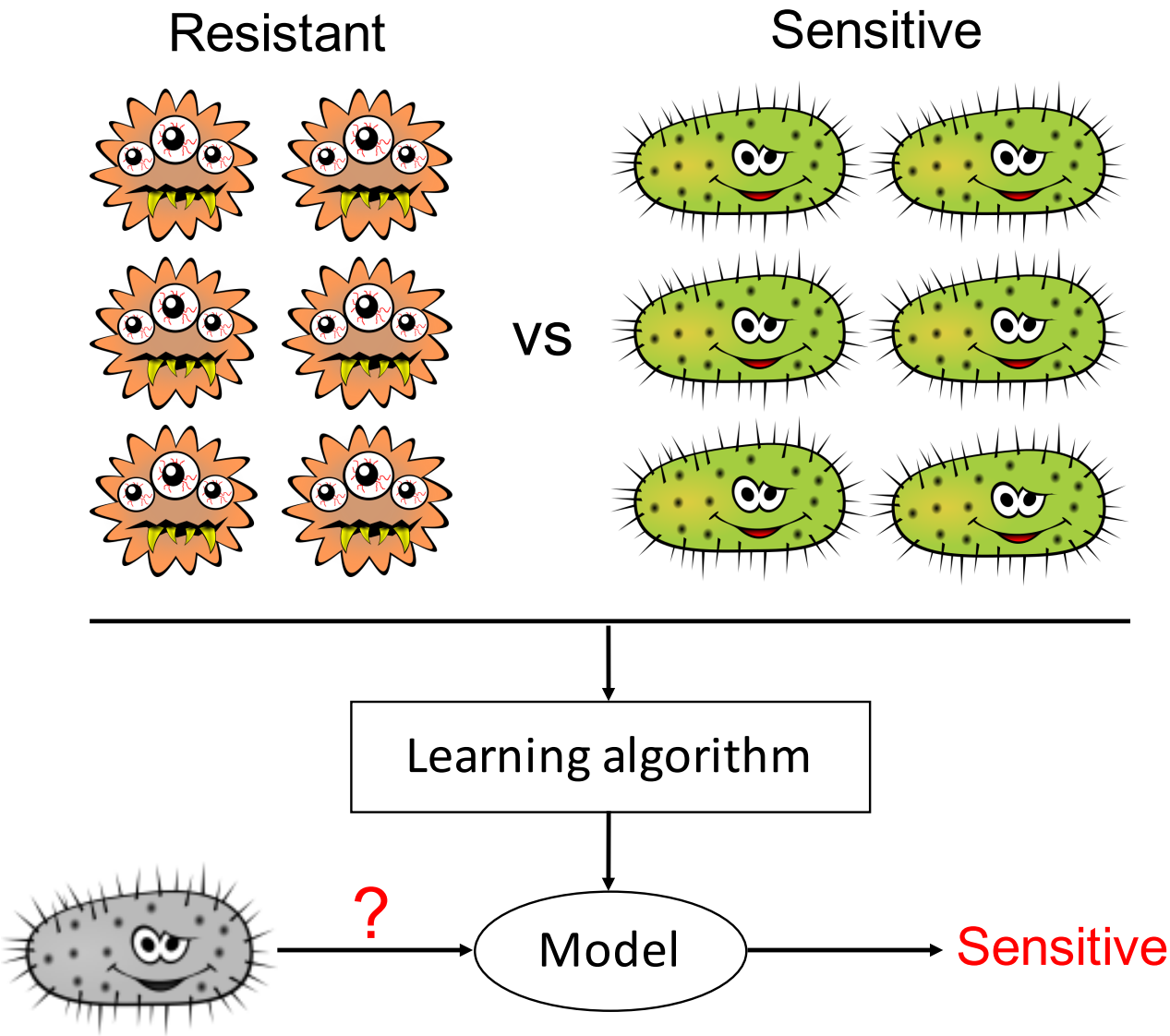
Pourquoi prédire la résistance aux antibiotiques ?

- Profilage rapide des antibiotiques pouvant être utilisés pour traiter une infection
- Plan de traitement personnalisé pour chaque patient, raccourcir/éviter la thérapie empirique (médecine personnalisée)
- Extraire des nouvelles connaissances biologiques des modèles obtenus

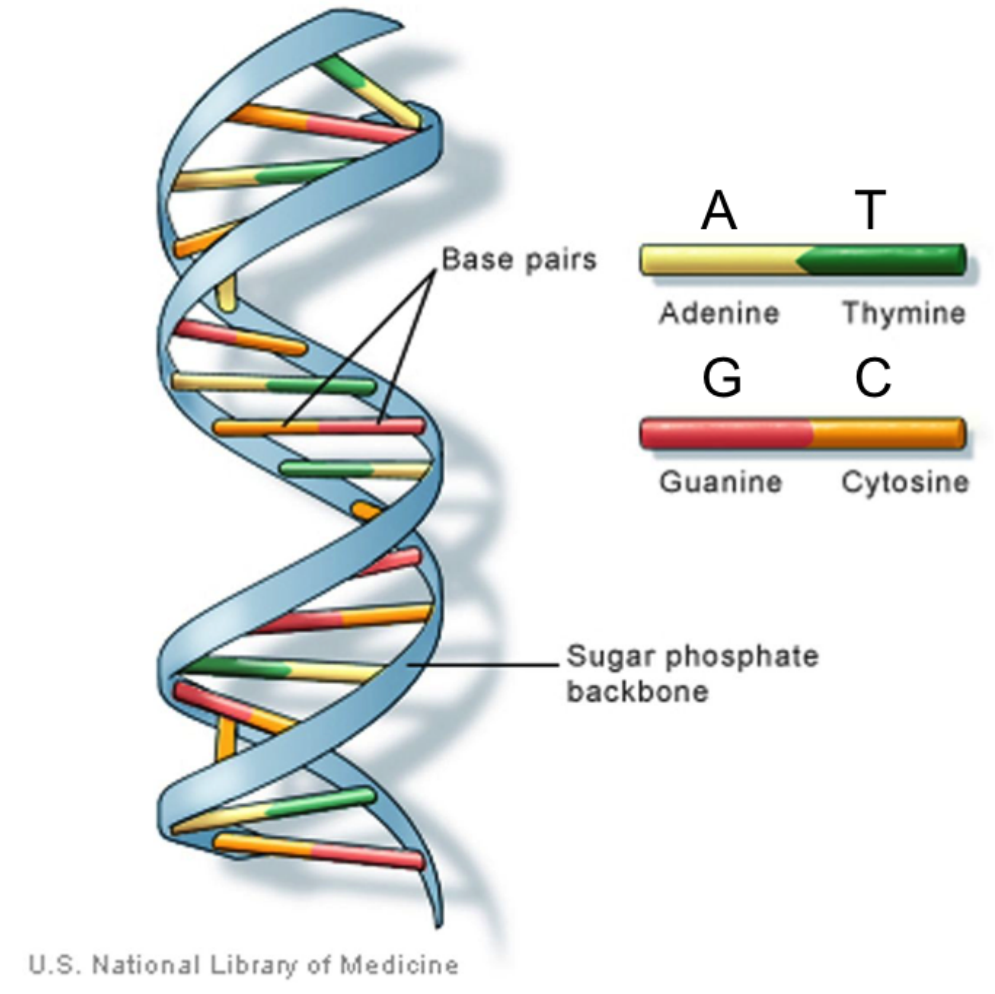
Pourquoi prédire la résistance aux antibiotiques ?

- Profilage rapide des antibiotiques pouvant être utilisés pour traiter une infection
- Plan de traitement personnalisé pour chaque patient, raccourcir/éviter la thérapie empirique (médecine personnalisée)
- Extraire des nouvelles connaissances biologiques des modèles obtenus

“Genome-wide association study” bactérien



Génomique



Étude de l'ensemble de l'ADN d'un ou plusieurs individus

Formalisation : un problème d'apprentissage supervisé

Ensemble de données

$$\mathcal{S} \stackrel{\text{def}}{=} \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_m, y_m))\} \sim D^m$$

$\mathbf{x} \in \mathcal{X} \stackrel{\text{def}}{=} \{A, C, G, T\}^*$ est un génome
$y \in \{0, 1\}$ est une étiquette (résistant, sensible)
D est une distribution inconnue qui génère les données

Objectif

1. Définir une représentation vectorielle des génomes $\phi : \mathcal{X} \to \mathbb{R}^d$
2. Trouver un prédicteur $h : \mathbb{R}^d \to \{0, 1\}$ qui a une bonne performance de généralisation, c.-à-d. qui minimise :

$$R(h) \stackrel{\text{def}}{=} \Pr_{(\mathbf{x},y)\sim D}[h(\phi(\mathbf{x})) \neq y]$$

Formalisation : un problème d'apprentissage supervisé

Ensemble de données

$$\mathcal{S} \stackrel{\text{def}}{=} \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_m, y_m))\} \sim D^m$$

$\mathbf{x} \in \mathcal{X} \stackrel{\text{def}}{=} \{A, C, G, T\}^*$ est un génome
$y \in \{0, 1\}$ est une étiquette (résistant, sensible)
D est une distribution inconnue qui génère les données

Objectif

1. Définir une représentation vectorielle des génomes $\phi : \mathcal{X} \to \mathbb{R}^d$
2. Trouver un prédicteur $h : \mathbb{R}^d \to \{0, 1\}$ qui a une bonne performance de généralisation, c.-à-d. qui minimise :

$$R(h) \stackrel{\text{def}}{=} \Pr_{(\mathbf{x},y)\sim D}[h(\phi(\mathbf{x})) \neq y]$$

Formalisation : un problème d'apprentissage supervisé

Ensemble de données

$$\mathcal{S} \stackrel{\text{def}}{=} \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_m, y_m))\} \sim D^m$$

$\mathbf{x} \in \mathcal{X} \stackrel{\text{def}}{=} \{A, C, G, T\}^*$ est un génome
$y \in \{0, 1\}$ est une étiquette (résistant, sensible)
D est une distribution inconnue qui génère les données

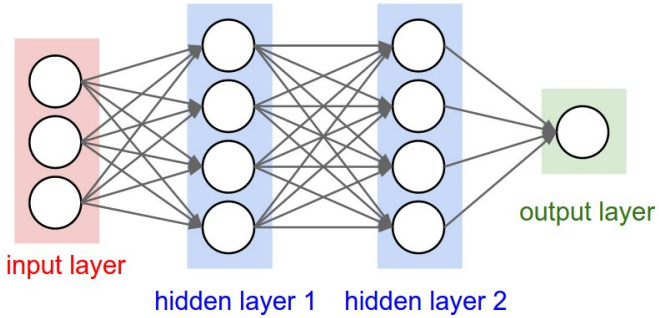
Objectif

1. Définir une représentation vectorielle des génomes $\phi : \mathcal{X} \to \mathbb{R}^d$
2. Trouver un prédicteur $h : \mathbb{R}^d \to \{0, 1\}$ qui a une bonne performance de généralisation, c.-à-d. qui minimise :

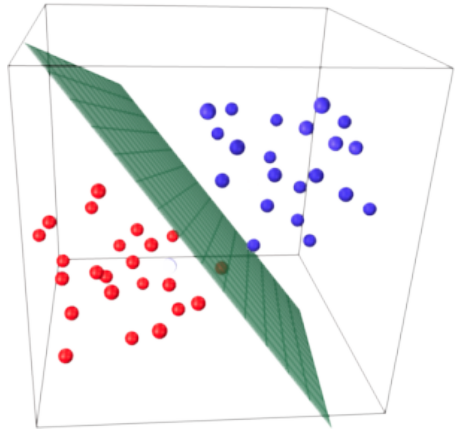
$$R(h) \stackrel{\text{def}}{=} \Pr_{(\mathbf{x},y)\sim D}[h(\phi(\mathbf{x})) \neq y]$$

Les modèles doivent être interprétables

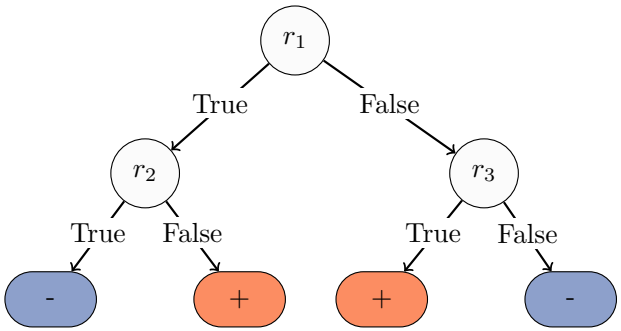
Réseau de neurones



Support Vector Machine



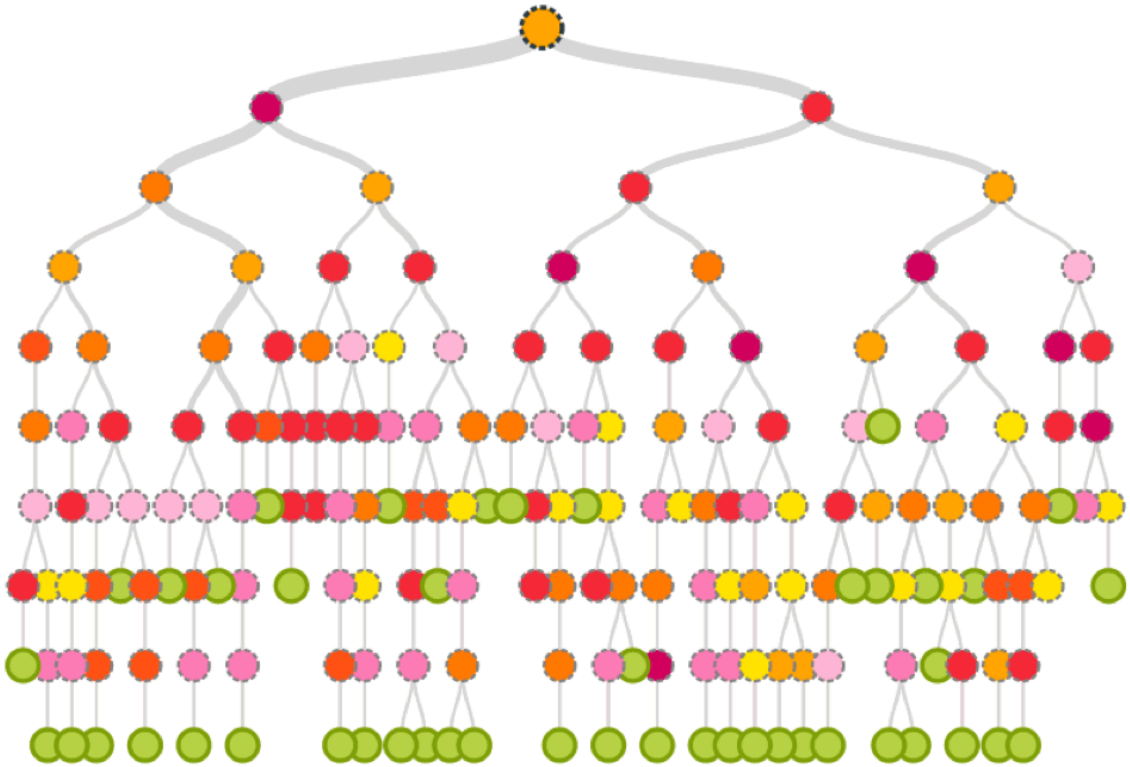
Arbre de décision



Les modèles doivent être interprétables :

- Validation et acceptation
- Permet de détecter les biais

Les modèles doivent être parcimonieux



Source : `https://blog.bigml.com/2013/05/07/a-sunburst-of-insight/`

Attention au surapprentissage

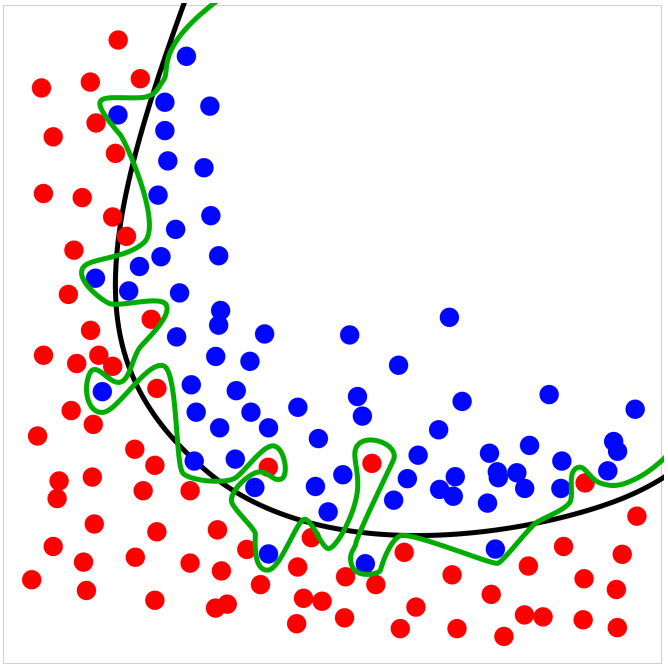
Représentation de très haute dimensionnalité $\phi(\mathbf{x})$

$+$

Peu d'exemples d'apprentissage ($m \ll d$)

$=$

Risque de surapprentissage !



Méthode

Représentation des génomes : profils de k-mers

Definition

k-mer : une séquence de k nucléotides

Note : Il y a 4^k séquences possibles

Definition

$\mathcal{K}$: l'ensemble de tous les k-mers qui sont dans au moins 1 génome de $\mathcal{S}$

Note : $|\mathcal{K}|$ peut QUAND MÊME être ÉNORME (dizaines de millions)

Nous représentons chaque génome $\mathbf{x}$ par un vecteur binaire $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{B}^{|\mathcal{K}|}$, tel que

$$\phi(\mathbf{x})_j = \begin{cases} 1, & \text{si } k_j \in \mathcal{K} \text{ est une sous-séquence de } \mathbf{x} \\ 0, & \text{sinon.} \end{cases}$$

Représentation des génomes : profils de k-mers

Definition

k-mer : une séquence de k nucléotides

Note : Il y a 4^k séquences possibles ($k = 31$, $4^{31} \approx 4 \cdot 10^{18}$)

Definition

$\mathcal{K}$: l'ensemble de tous les k-mers qui sont dans au moins 1 génome de $\mathcal{S}$

Note : $|\mathcal{K}|$ peut QUAND MÊME être ÉNORME (dizaines de millions)

Nous représentons chaque génome $\mathbf{x}$ par un vecteur binaire $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{B}^{|\mathcal{K}|}$, tel que

$$\phi(\mathbf{x})_j = \begin{cases} 1, & \text{si } k_j \in \mathcal{K} \text{ est une sous-séquence de } \mathbf{x} \\ 0, & \text{sinon.} \end{cases}$$

Représentation des génomes : profils de k-mers

Definition

k-mer : une séquence de k nucléotides

Note : Il y a 4^k séquences possibles ($k = 31$, $4^{31} \approx 4 \cdot 10^{18}$)

Definition

$\mathcal{K}$: l'ensemble de tous les k-mers qui sont dans au moins 1 génome de $\mathcal{S}$

Note : $|\mathcal{K}|$ peut QUAND MÊME être ÉNORME (dizaines de millions)

Nous représentons chaque génome $\mathbf{x}$ par un vecteur binaire $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{B}^{|\mathcal{K}|}$, tel que

$$\phi(\mathbf{x})_j = \begin{cases} 1, & \text{si } k_j \in \mathcal{K} \text{ est une sous-séquence de } \mathbf{x} \\ 0, & \text{sinon.} \end{cases}$$

Représentation des génomes : profils de k-mers

Definition

k-mer : une séquence de k nucléotides

Note : Il y a 4^k séquences possibles ($k = 31$, $4^{31} \approx 4 \cdot 10^{18}$)

Definition

$\mathcal{K}$: l'ensemble de tous les k-mers qui sont dans au moins 1 génome de $\mathcal{S}$

Note : $|\mathcal{K}|$ peut QUAND MÊME être ÉNORME (dizaines de millions)

Nous représentons chaque génome $\mathbf{x}$ par un vecteur binaire $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{B}^{|\mathcal{K}|}$, tel que

$$\phi(\mathbf{x})_j = \begin{cases} 1, & \text{si } k_j \in \mathcal{K} \text{ est une sous-séquence de } \mathbf{x} \\ 0, & \text{sinon.} \end{cases}$$

Représentation des génomes (exemple)

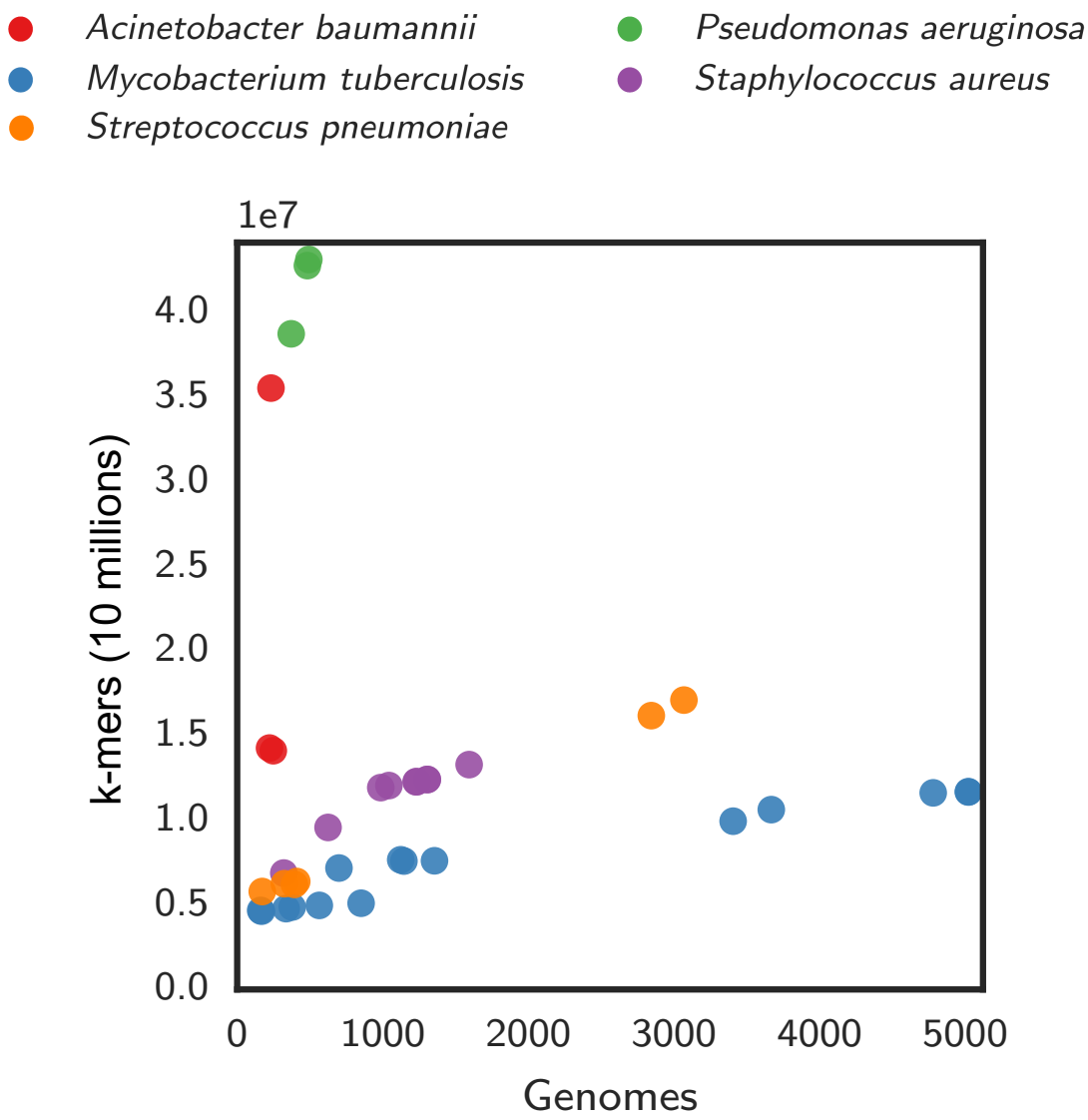
$$\mathcal{K} = \left\{ \begin{array}{cccc} \text{CAGATA} & \text{GATAGA} & \text{GAACAG} & \text{CGATGA} \\ \text{AGATAG} & \text{AGAACA} & \text{ATAGAA} & \text{CCGGCT} \\ \text{AACAGC} & \text{TAGAAC} & \text{TTTCGG} & \text{AAATAC} \end{array} \right\}$$

$$\mathbf{x} = \text{CAGATAGAACAGC}$$

$$\phi(\mathbf{x}) =$$

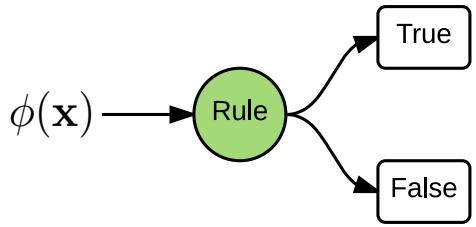
1	0	1	1	0	1	1	0	1	1	1	0
CAGATA	TTTCGG	AGATAG	GATAGA	CGATGA	AACAGC	ATAGAA	CCGGCT	TAGAAC	GAACAG	AGAACA	AAATAC

Exemple : taille des ensembles de données



L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2002)

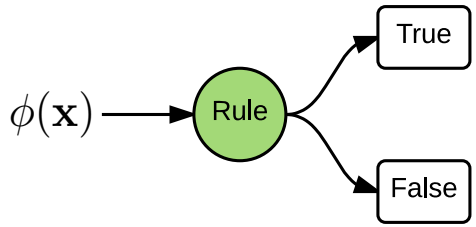
- **Entrée :** Un ensemble de règles à valeur booléenne (présence/absence de k-mers) et un ensemble de données (exemples, étiquettes)



- **Objective :** Trouver la plus courte conjonction (ET-logique) ou disjonction (OU-logique) de règles qui prédit le plus précisément possible les étiquettes

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2002)

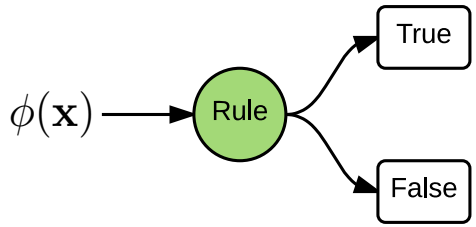
- **Entrée :** Un ensemble de règles à valeur booléenne (présence/absence de k-mers) et un ensemble de données (exemples, étiquettes)



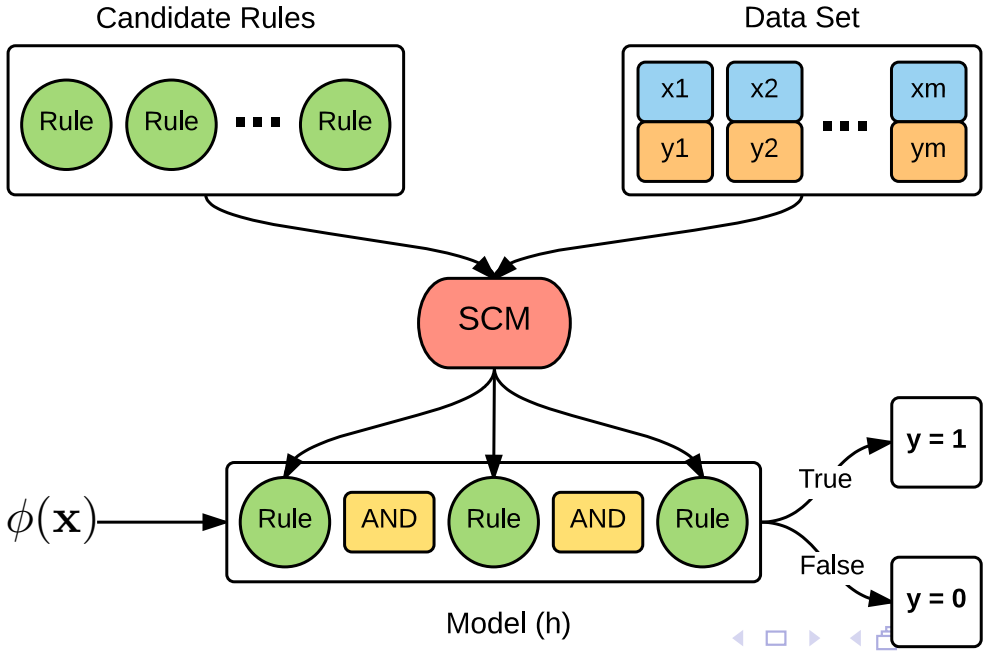
- **Objective :** Trouver la plus courte conjonction (ET-logique) ou disjonction (OU-logique) de règles qui prédit le plus précisément possible les étiquettes

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2002)

- **Entrée :** Un ensemble de règles à valeur booléenne (présence/absence de k-mers) et un ensemble de données (exemples, étiquettes)



- **Objective :** Trouver la plus courte conjonction (ET-logique) ou disjonction (OU-logique) de règles qui prédit le plus précisément possible les étiquettes



L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Objectif

Étant donné un ensemble de règles à valeur booléenne $\mathcal{R}$, trouver le prédicteur qui minimise le risque empirique :

$$R_S \stackrel{\text{def}}{=} \sum_{i=1}^{m} \frac{1}{m} I[h(\phi(\mathbf{x}_i)) \neq y_i],$$

en utilisant le plus petit sous-ensemble de $\mathcal{R}$.

- Ce problème est *NP*-difficile (minimum set cover)
- Solution : Utilisé un algorithme glouton inspiré de celui de Chvátal (1979)

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Objectif

Étant donné un ensemble de règles à valeur booléenne $\mathcal{R}$, trouver le prédicteur qui minimise le risque empirique :

$$R_S \stackrel{\text{def}}{=} \sum_{i=1}^{m} \frac{1}{m} I[h(\phi(\mathbf{x}_i)) \neq y_i],$$

en utilisant le plus petit sous-ensemble de $\mathcal{R}$.

- Ce problème est *NP*-difficile (minimum set cover)
- Solution : Utilisé un algorithme glouton inspiré de celui de Chvátal (1979)

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Algorithme glouton (Conjonction)

1. Commencer avec une conjonction vide
2. Calculer un score d'utilité pour chaque règle de $\mathcal{R}$
3. Sélectionner la règle avec la plus grande utilité (r^*)
4. Supprimer tous les exemples pour lesquels $r^*(\phi(\mathbf{x})) = \text{Faux}$
5. Aller à l'étape 2 jusqu'à ce que l'une de ces conditions soit satisfaite :
   - Tous les exemples négatifs ont été retirés
   - s itérations complétées (hyperparamètre)

Justification de l'étape 4

- La conjonction prend la valeur Faux pour tout exemple pour qui au moins une règle retourne Faux.
- Inutile de considérer ces exemples dans les itérations futures

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Algorithme glouton (Conjonction)

1. Commencer avec une conjonction vide
2. Calculer un score d'utilité pour chaque règle de $\mathcal{R}$
3. Sélectionner la règle avec la plus grande utilité ($r^\star$)
4. Supprimer tous les exemples pour lesquels $r^\star(\phi(\mathbf{x})) = \text{Faux}$
5. Aller à l'étape 2 jusqu'à ce que l'une de ces conditions soit satisfaite :
   - Tous les exemples négatifs ont été retirés
   - s itérations complétées (hyperparamètre)

Justification de l'étape 4

- La conjonction prend la valeur Faux pour tout exemple pour qui au moins une règle retourne Faux.
- Inutile de considérer ces exemples dans les itérations futures

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Algorithme glouton (Conjonction)

1. Commencer avec une conjonction vide
2. Calculer un score d'utilité pour chaque règle de $\mathcal{R}$
3. Sélectionner la règle avec la plus grande utilité ($r^\star$)
4. Supprimer tous les exemples pour lesquels $r^\star(\phi(\mathbf{x})) = \text{Faux}$
5. Aller à l'étape 2 jusqu'à ce que l'une de ces conditions soit satisfaite :
   - Tous les exemples négatifs ont été retirés
   - s itérations complétées (hyperparamètre)

Justification de l'étape 4

- La conjonction prend la valeur Faux pour tout exemple pour qui au moins une règle retourne Faux.
- Inutile de considérer ces exemples dans les itérations futures

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Algorithme glouton (Conjonction)

1. Commencer avec une conjonction vide
2. Calculer un score d'utilité pour chaque règle de $\mathcal{R}$
3. Sélectionner la règle avec la plus grande utilité ($r^\star$)
4. Supprimer tous les exemples pour lesquels $r^\star(\boldsymbol{\phi}(\mathbf{x})) = \text{Faux}$
5. Aller à l'étape 2 jusqu'à ce que l'une de ces conditions soit satisfaite :
   - Tous les exemples négatifs ont été retirés
   - s itérations complétées (hyperparamètre)

Justification de l'étape 4

- La conjonction prend la valeur Faux pour tout exemple pour qui au moins une règle retourne Faux.
- Inutile de considérer ces exemples dans les itérations futures

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Algorithme glouton (Conjonction)

1. Commencer avec une conjonction vide
2. Calculer un score d'utilité pour chaque règle de $\mathcal{R}$
3. Sélectionner la règle avec la plus grande utilité ($r^\star$)
4. Supprimer tous les exemples pour lesquels $r^\star(\boldsymbol{\phi}(\mathbf{x})) = \text{Faux}$
5. Aller à l'étape 2 jusqu'à ce que l'une de ces conditions soit satisfaite :
   - Tous les exemples négatifs ont été retirés
   - s itérations complétées (hyperparamètre)

Justification de l'étape 4

- La conjonction prend la valeur Faux pour tout exemple pour qui au moins une règle retourne Faux.
- Inutile de considérer ces exemples dans les itérations futures

L'algorithme Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Algorithme glouton (Conjonction)

1. Commencer avec une conjonction vide
2. Calculer un score d'utilité pour chaque règle de $\mathcal{R}$
3. Sélectionner la règle avec la plus grande utilité ($r^\star$)
4. Supprimer tous les exemples pour lesquels $r^\star(\boldsymbol{\phi}(\mathbf{x})) = \text{Faux}$
5. Aller à l'étape 2 jusqu'à ce que l'une de ces conditions soit satisfaite :
   - Tous les exemples négatifs ont été retirés
   - s itérations complétées (hyperparamètre)

Justification de l'étape 4

- La conjonction prend la valeur Faux pour tout exemple pour qui au moins une règle retourne Faux.
- Inutile de considérer ces exemples dans les itérations futures

The Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Score d'utilité

Pour toute règle r_i, l'utilité est donnée par :

$$U_i \stackrel{\text{def}}{=} \mathcal{N}_i - p \cdot \overline{\mathcal{P}_i},$$

où $\mathcal{N}_i$ est le nombre d'exemples négatifs correctement classifiés par r_i,
$\overline{\mathcal{P}_i}$ est le nombre d'exemples positifs mal classifiés par r_i,
p est un hyperparamètre.

Gros volumes de données

- La complexité de calcul de SCM est $O(m \cdot |\mathcal{R}| \cdot s)$, donc linéaire en fonction du nombre d'exemples et de règles
- Implémentation *out-of-core* : les données sont analysées par blocs

The Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Score d'utilité

Pour toute règle r_i, l'utilité est donnée par :

$$U_i \stackrel{\text{def}}{=} \mathcal{N}_i - p \cdot \overline{\mathcal{P}_i},$$

où $\mathcal{N}_i$ est le nombre d'exemples négatifs correctement classifiés par r_i,
$\overline{\mathcal{P}_i}$ est le nombre d'exemples positifs mal classifiés par r_i,
p est un hyperparamètre.

Gros volumes de données

- La complexité de calcul de SCM est $O(m \cdot |\mathcal{R}| \cdot s)$, donc linéaire en fonction du nombre d'exemples et de règles
- Implémentation *out-of-core* : les données sont analysées par blocs

The Set Covering Machine (Marchand and Shawe-Taylor, 2003)

Score d'utilité

Pour toute règle r_i, l'utilité est donnée par :

$$U_i \stackrel{\text{def}}{=} \mathcal{N}_i - p \cdot \overline{\mathcal{P}_i},$$

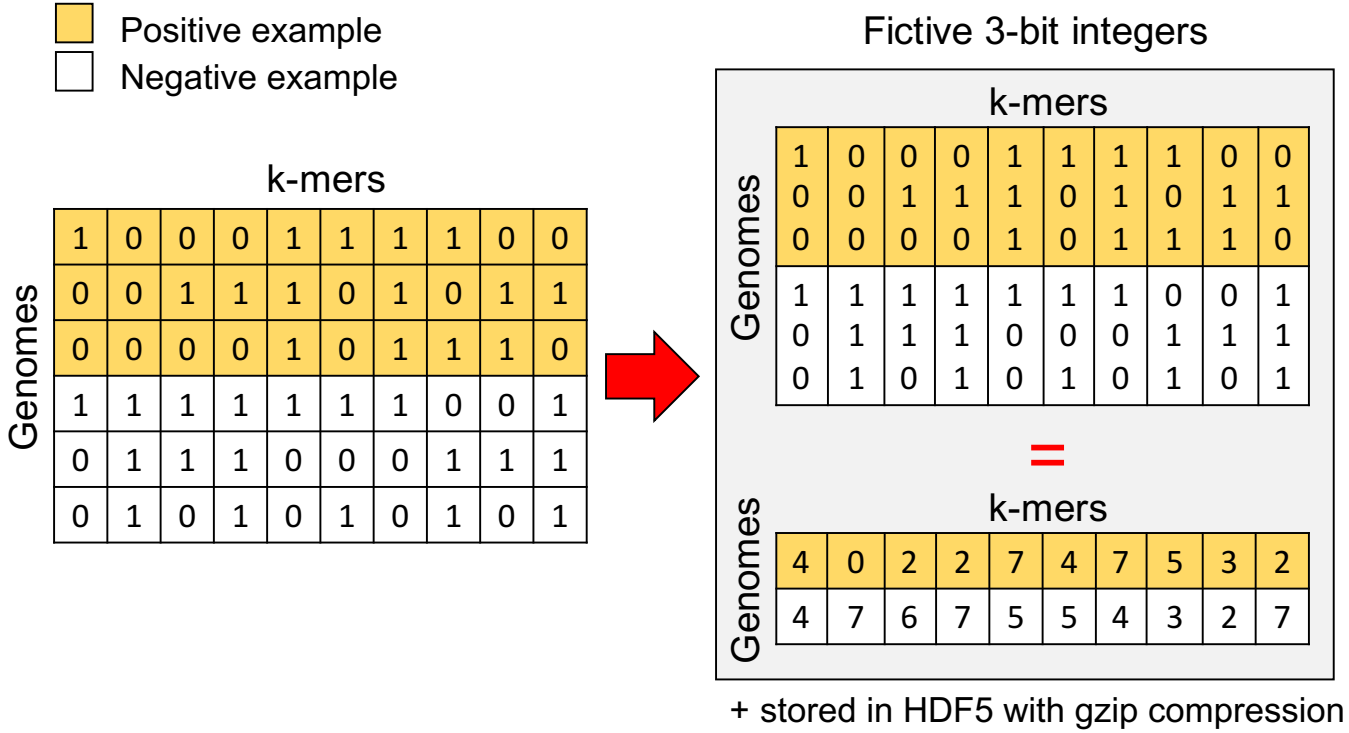
où $\mathcal{N}_i$ est le nombre d'exemples négatifs correctement classifiés par r_i,
$\overline{\mathcal{P}_i}$ est le nombre d'exemples positifs mal classifiés par r_i,
p est un hyperparamètre.

Gros volumes de données

- La complexité de calcul de SCM est $O(m \cdot |\mathcal{R}| \cdot s)$, donc linéaire en fonction du nombre d'exemples et de règles
- Implémentation *out-of-core* : les données sont analysées par blocs

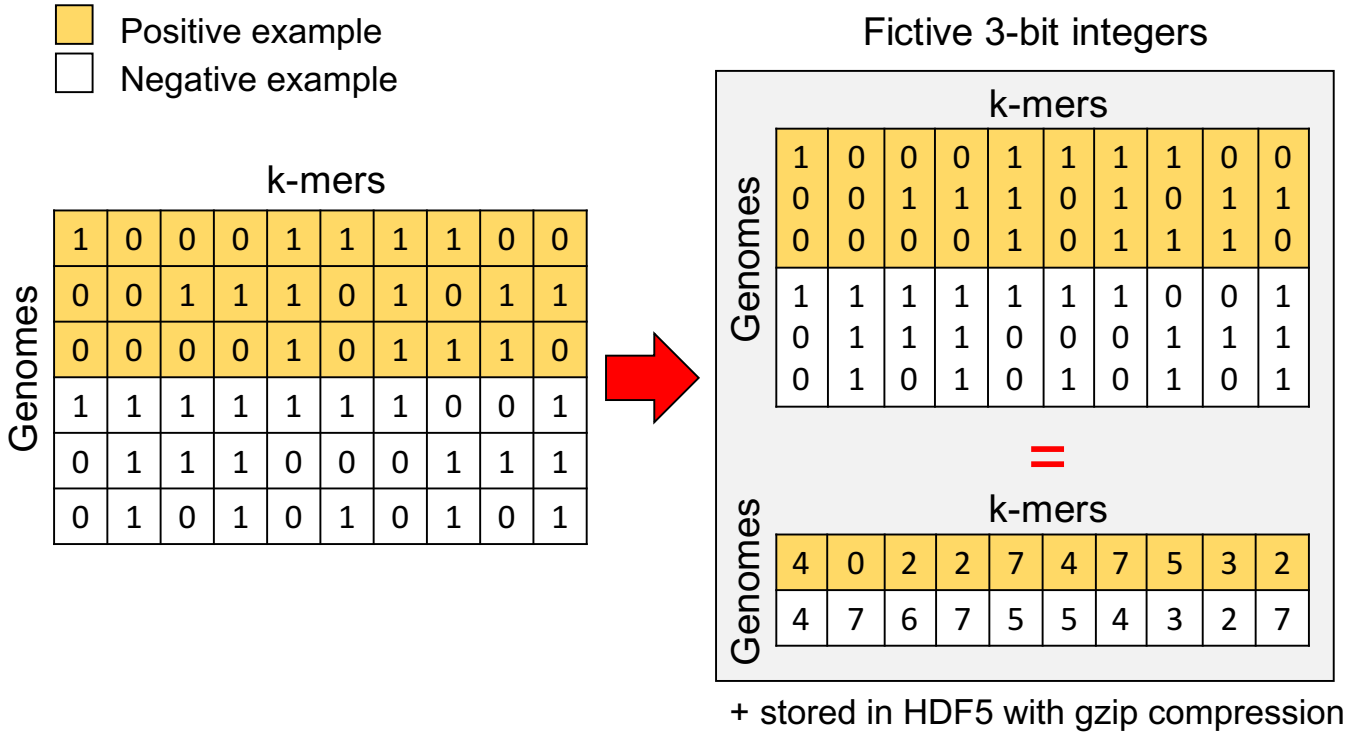
Kover : une implémentation *out-of-core* pour la génomique

1) Représentation compressée



Kover : une implémentation *out-of-core* pour la génomique

1) Représentation compressée

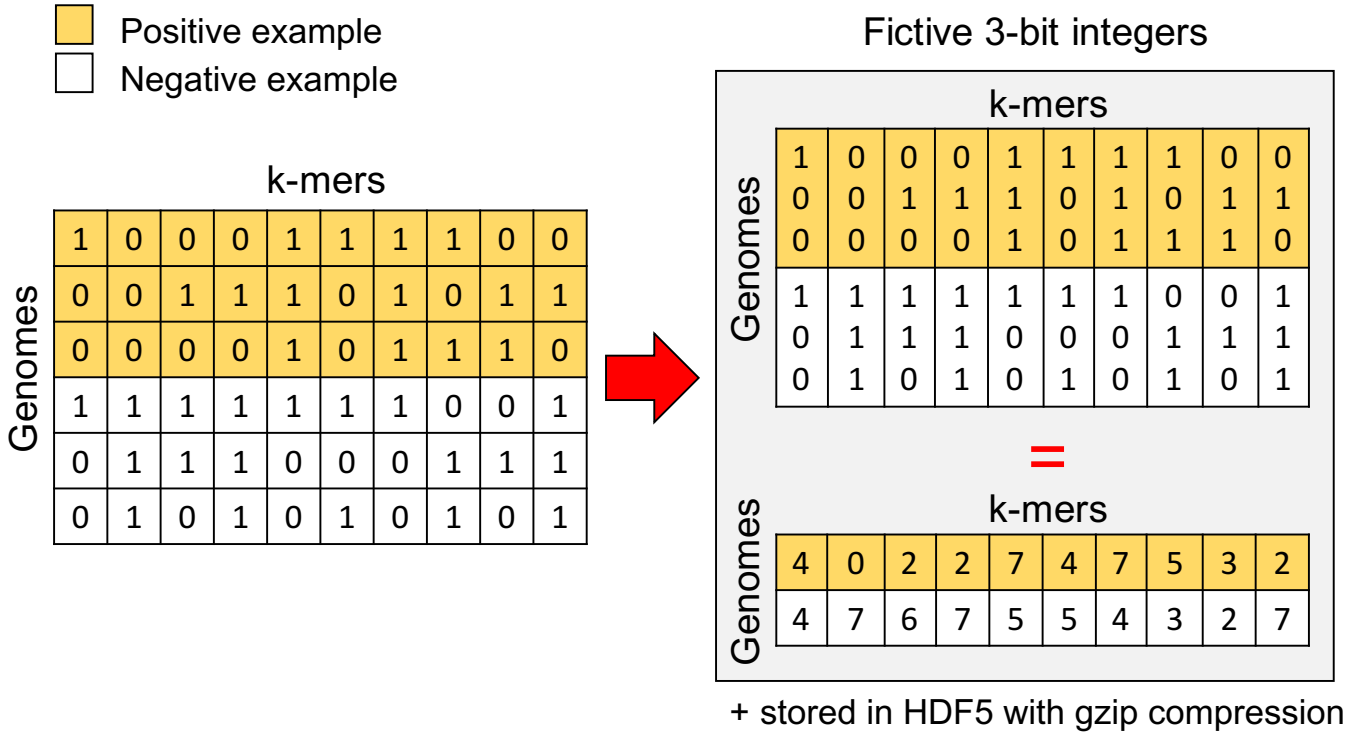


+ stored in HDF5 with gzip compression

2) Apprentissage à partir des données compressées

Kover : une implémentation *out-of-core* pour la génomique

1) Représentation compressée



+ stored in HDF5 with gzip compression

2) Apprentissage à partir des données compressées

$\overline{\mathcal{P}_i}$ = Number of positive examples − Number of correctly classified positive examples

popcount [4 0 2 2 7 4 7 5 3 2] = 1 0 1 1 3 1 3 2 2 1

Pouvons nous espérer une bonne performance de généralisation ?

Nous pouvons borner le risque d'un prédicteur en fonction de sa performance sur les données d'entraînement. Le terme suivant borne le risque de toute conjonction h de règles dans $\mathcal{R}$ avec probabilité $\geq 1 - \delta$.

Borne du rasoir d'Occam

$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m-r}\left[\ln\binom{m}{r} + \ln\binom{2\cdot 4^k}{|h|} - \ln(\zeta(r)\cdot\zeta(|h|)\cdot\delta)\right],$$

où r est le nombre d'erreurs sur l'ensemble d'entraînement,

$|h|$ est le nombre de règles dans le modèle,

ζ est une fonction telle que $\sum_{b\in\mathbb{N}}\zeta(b) \leq 1$

- Le terme combinatoire domine cette borne, même pour les prédicteurs faisant peu d'erreurs
- La borne suggère une mauvaise performance de généralisation.

Pouvons nous espérer une bonne performance de généralisation ?

Nous pouvons borner le risque d'un prédicteur en fonction de sa performance sur les données d'entraînement. Le terme suivant borne le risque de toute conjonction h de règles dans $\mathcal{R}$ avec probabilité $\geq 1 - \delta$.

Borne du rasoir d'Occam

$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m-r}\left[\ln\binom{m}{r} + \ln\binom{2\cdot 4^k}{|h|} - \ln(\zeta(r)\cdot\zeta(|h|)\cdot\delta)\right],$$

où r est le nombre d'erreurs sur l'ensemble d'entraînement,

$|h|$ est le nombre de règles dans le modèle,

ζ est une fonction telle que $\sum_{b\in\mathbb{N}}\zeta(b) \leq 1$

- Le terme combinatoire domine cette borne, même pour les prédicteurs faisant peu d'erreurs
- La borne suggère une mauvaise performance de généralisation.

Pouvons nous espérer une bonne performance de généralisation ?

Nous pouvons borner le risque d'un prédicteur en fonction de sa performance sur les données d'entraînement. Le terme suivant borne le risque de toute conjonction h de règles dans $\mathcal{R}$ avec probabilité $\geq 1 - \delta$.

Borne du rasoir d'Occam

$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m - r} \left[\ln \binom{m}{r} + \ln \binom{2 \cdot 4^k}{|h|} - \ln(\zeta(r) \cdot \zeta(|h|) \cdot \delta) \right],$$

où r est le nombre d'erreurs sur l'ensemble d'entraînement,

$|h|$ est le nombre de règles dans le modèle,

ζ est une fonction telle que $\sum_{b \in \mathbb{N}} \zeta(b) \leq 1$

- Le terme combinatoire domine cette borne, même pour les prédicteurs faisant peu d'erreurs
- La borne suggère une mauvaise performance de généralisation.

Pouvons nous espérer une bonne performance de généralisation ?

Nous pouvons borner le risque d'un prédicteur en fonction de sa performance sur les données d'entraînement. Le terme suivant borne le risque de toute conjonction h de règles dans $\mathcal{R}$ avec probabilité $\geq 1 - \delta$.

Borne du rasoir d'Occam

$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m-r}\left[\ln\binom{m}{r} + \ln\binom{2\cdot 4^k}{|h|} - \ln(\zeta(r)\cdot\zeta(|h|)\cdot\delta)\right],$$

où r est le nombre d'erreurs sur l'ensemble d'entraînement,

$|h|$ est le nombre de règles dans le modèle,

ζ est une fonction telle que $\sum_{b\in\mathbb{N}}\zeta(b) \leq 1$

- Le terme combinatoire domine cette borne, même pour les prédicteurs faisant peu d'erreurs
- La borne suggère une mauvaise performance de généralisation.

Pouvons nous espérer une bonne performance de généralisation ?

Dans le contexte de la compression d'échantillons, le prédicteur h est spécifié en utilisant un petit ensemble d'exemples d'entraînement $(\mathcal{Z}_\mathbf{i})$:

Borne de compression d'échantillons

$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m - |h| - r} \left[\ln \binom{m}{|h|} + \ln \binom{m - |h|}{r} + \sum_{\mathbf{x} \in \mathcal{Z}_\mathbf{i}} \ln(2 \cdot |\mathbf{x}|) - \ln(\zeta(|h|) \cdot \zeta(r) \cdot \delta) \right],$$

où r est le nombre d'erreurs faites sur $\mathcal{S} \setminus \mathcal{Z}_\mathbf{i}$.

- La borne ne dépend plus de k
- Considérer des représentations exponentiellement plus complexes, sans pénalité au niveau de la généralisation

Pouvons nous espérer une bonne performance de généralisation ?

Dans le contexte de la compression d'échantillons, le prédicteur h est spécifié en utilisant un petit ensemble d'exemples d'entraînement $(\mathcal{Z}_\mathbf{i})$:

Borne de compression d'échantillons

$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m - |h| - r} \left[\ln \binom{m}{|h|} + \ln \binom{m - |h|}{r} + \sum_{\mathbf{x} \in \mathcal{Z}_\mathbf{i}} \ln(2 \cdot |\mathbf{x}|) - \ln(\zeta(|h|) \cdot \zeta(r) \cdot \delta) \right],$$

où r est le nombre d'erreurs faites sur $\mathcal{S} \setminus \mathcal{Z}_\mathbf{i}$.

- La borne ne dépend plus de k
- Considérer des représentations exponentiellement plus complexes, sans pénalité au niveau de la généralisation

Pouvons nous espérer une bonne performance de généralisation ?

Dans le contexte de la compression d'échantillons, le prédicteur h est spécifié en utilisant un petit ensemble d'exemples d'entraînement $(\mathcal{Z}_\mathbf{i})$:

Borne de compression d'échantillons

$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m - |h| - r} \left[\ln \binom{m}{|h|} + \ln \binom{m - |h|}{r} + \sum_{\mathbf{x} \in \mathcal{Z}_\mathbf{i}} \ln(2 \cdot |\mathbf{x}|) - \ln(\zeta(|h|) \cdot \zeta(r) \cdot \delta) \right],$$

où r est le nombre d'erreurs faites sur $\mathcal{S} \setminus \mathcal{Z}_\mathbf{i}$.

- La borne ne dépend plus de k
- Considérer des représentations exponentiellement plus complexes, sans pénalité au niveau de la généralisation

Pouvons nous espérer une bonne performance de généralisation ?

Dans le contexte de la compression d'échantillons, le prédicteur h est spécifié en utilisant un petit ensemble d'exemples d'entraînement $(\mathcal{Z}_\mathbf{i})$:

Borne de compression d'échantillons

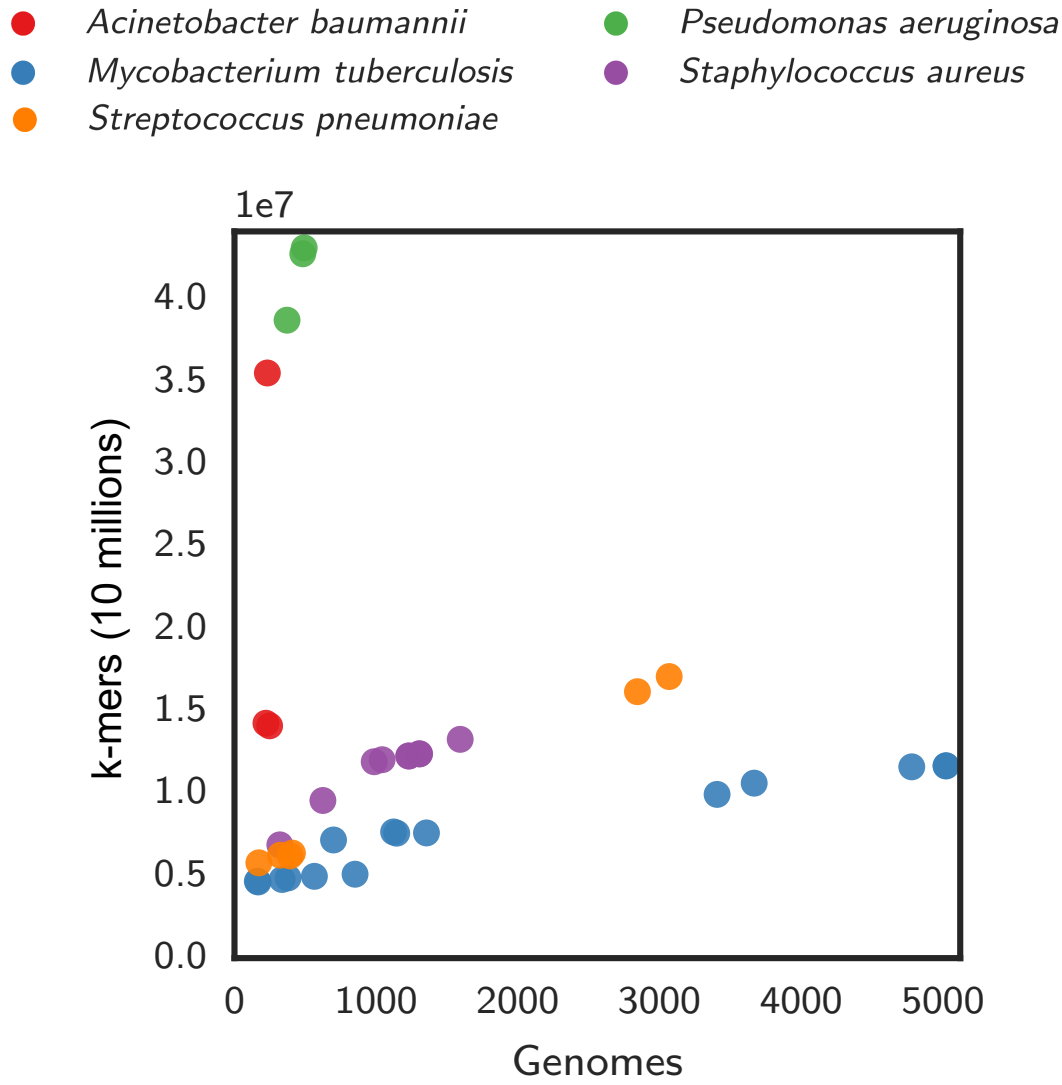
$$\epsilon \stackrel{\text{def}}{=} \frac{1}{m - |h| - r} \left[\ln \binom{m}{|h|} + \ln \binom{m - |h|}{r} + \sum_{\mathbf{x} \in \mathcal{Z}_\mathbf{i}} \ln(2 \cdot |\mathbf{x}|) - \ln(\zeta(|h|) \cdot \zeta(r) \cdot \delta) \right],$$

où r est le nombre d'erreurs faites sur $\mathcal{S} \setminus \mathcal{Z}_\mathbf{i}$.

- La borne ne dépend plus de k
- Considérer des représentations exponentiellement plus complexes, sans pénalité au niveau de la généralisation

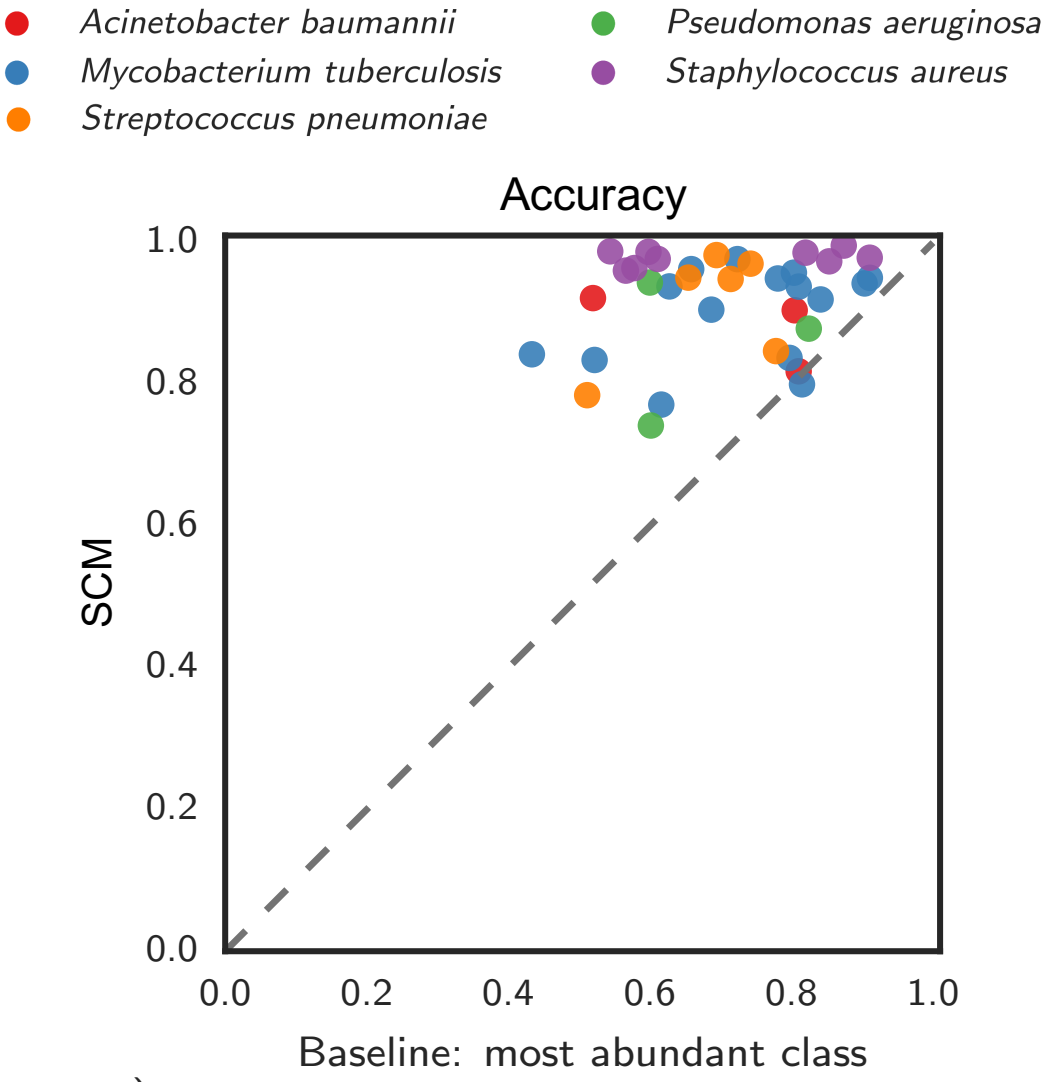
Résultats

La base de données PATRIC (Wattam et al, 2013)



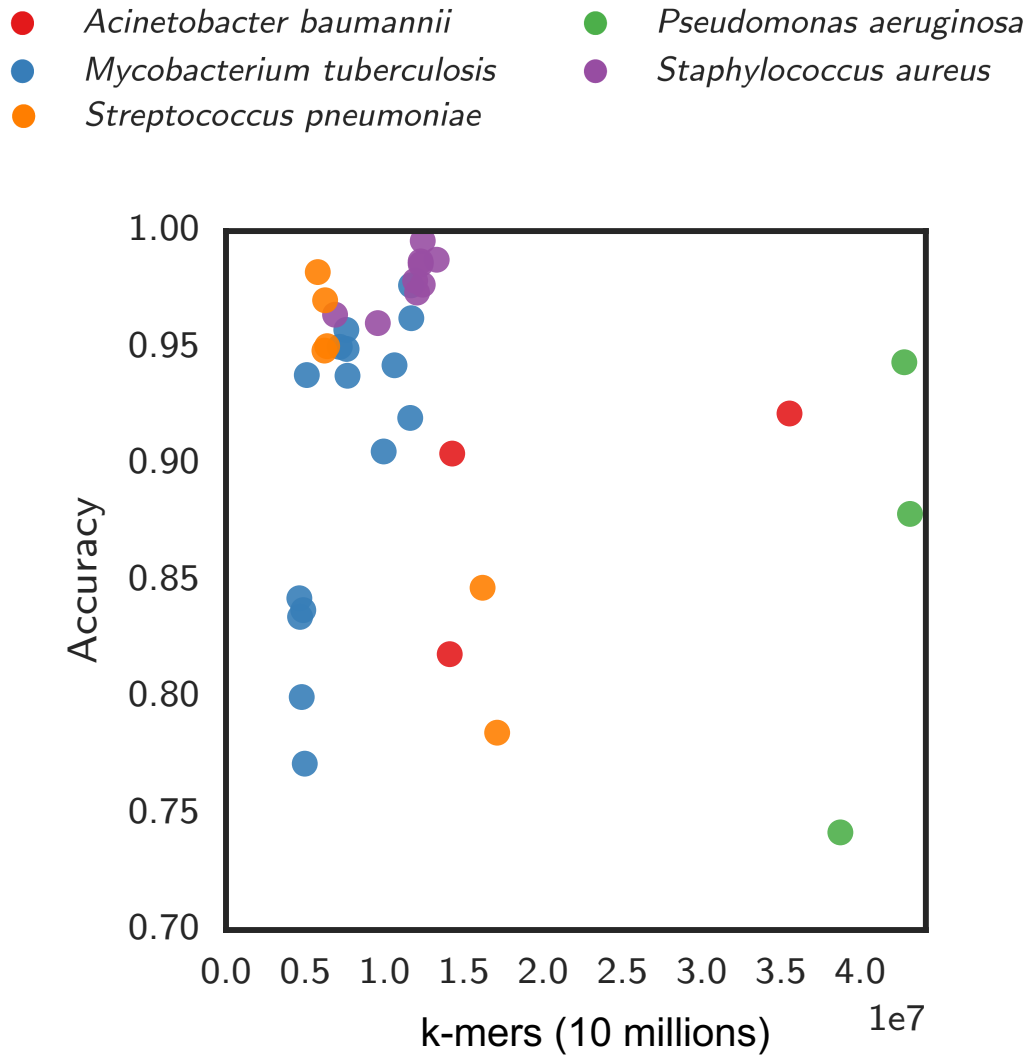
5 espèces bactériennes, 36 ensembles de données, $\sim$ 12000 génomes

La base de données PATRIC (Wattam et al, 2013)



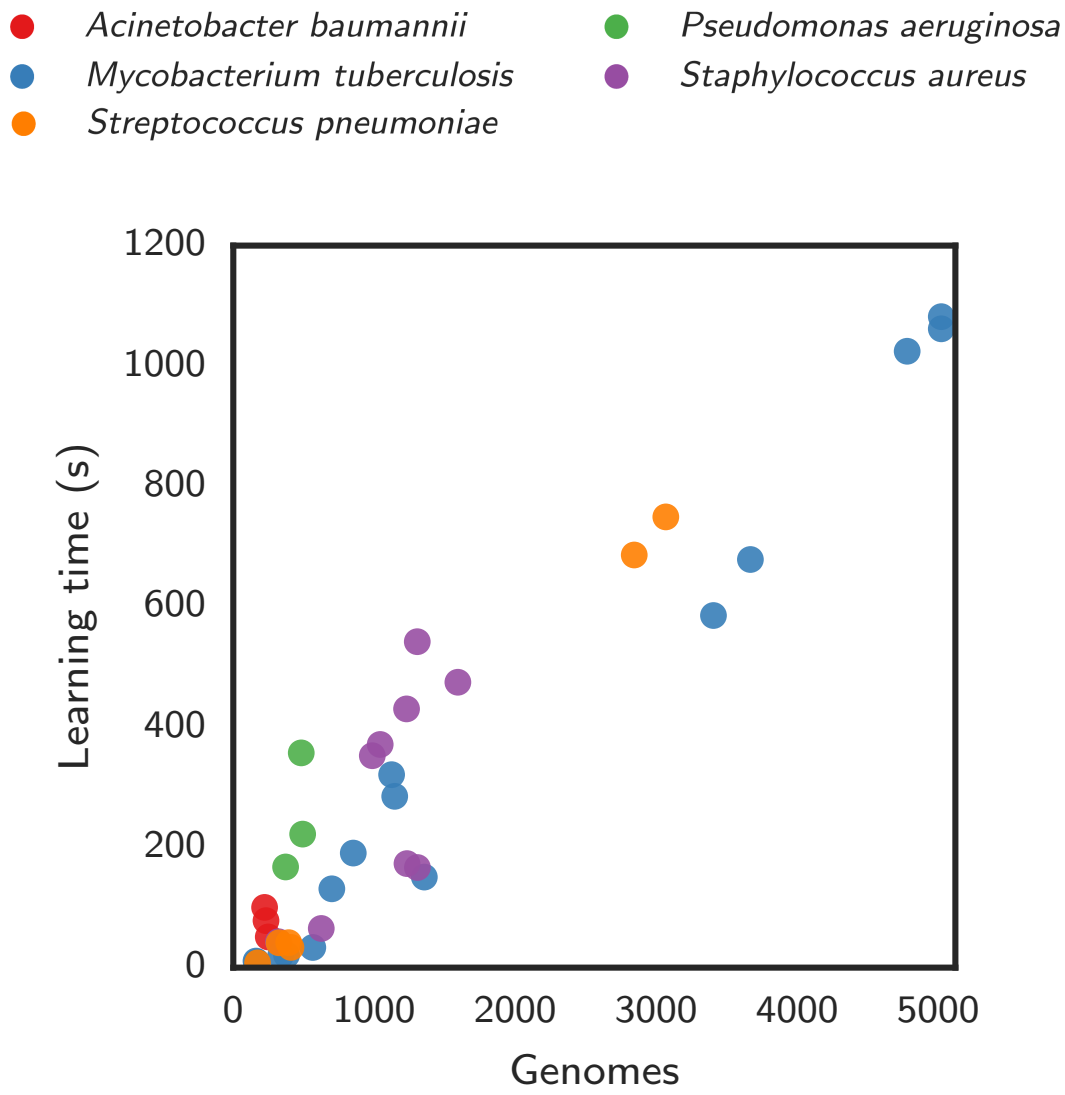
Taux de bonnes prédictions $\geq 90\%$ pour 26/36 ensembles de données

La base de données PATRIC (Wattam et al, 2013)



Le taux de bonnes prédictions ne dépend pas du nombre de k-mers

La base de données PATRIC (Wattam et al, 2013)

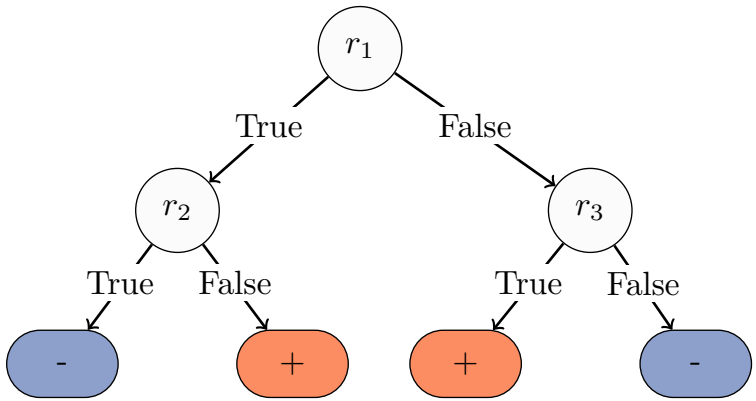


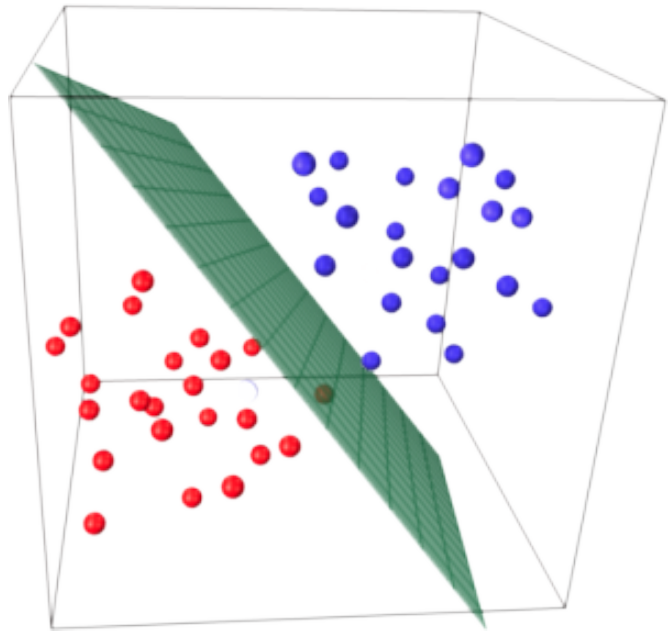
Temps de calcul linéaire en fonction du nombre de génomes (8 sec. - 18 min.)

Comparaison à d'autres algorithmes

Nous avons comparé SCM à :

- Arbres de décision (CART)
- SVM parcimonieux (L_1-SVM)
- SVM denses (L_2-SVM)
- SVM avec noyau polynomial (PolySVM)

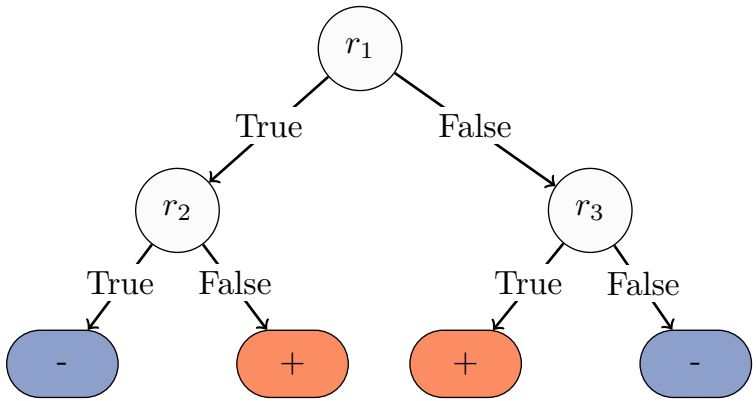


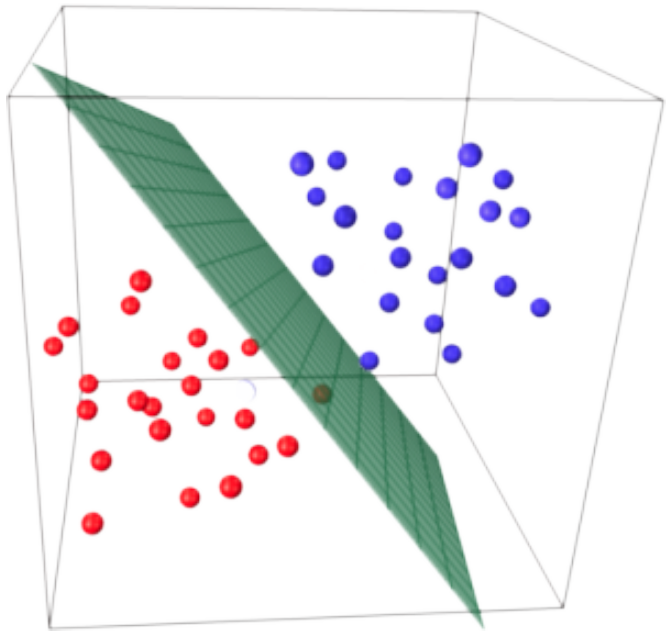


Comparaison à d'autres algorithmes

Nous avons comparé SCM à :

- Arbres de décision (CART) + χ^2 sélection d'attributs
- SVM parcimonieux (L_1-SVM) + χ^2 sélection d'attributs
- SVM denses (L_2-SVM) + χ^2 sélection d'attributs
- SVM avec noyau polynomial (PolySVM)





Comparaison à d'autres algorithmes

Dataset	SCM	χ^2 + CART	χ^2 + L1SVM	χ^2 + L2SVM	PolySVM
C. difficile					
Azithromycin	**0.030 (3.3)**	0.086 (7.2)	0.064 (20326.0)	0.056 (10^6)	0.048 (all)
Ceftriaxone	**0.073 (2.6)**	0.117 (6.8)	0.087 (8114.1)	0.102 (10^6)	0.076 (all)
Clarithromycin	**0.011 (3.0)**	0.070 (8.0)	0.062 (36686.1)	0.059 (10^6)	0.053 (all)
Clindamycin	0.021 **(1.4)**	0.011 (2.0)	**0.009** (598.2)	0.021 (10^6)	0.039 (all)
Moxifloxacin	**0.020 (1.0)**	**0.020** (1.3)	**0.020** (25.6)	0.048 (10^6)	0.048 (all)
M. tuberculosis					
Ethambutol	0.179 **(1.4)**	0.185 (1.9)	**0.153** (201.3)	0.221 (10^6)	0.221 (all)
Isoniazid	0.021 **(1.0)**	0.021 (1.1)	**0.017** (104.7)	0.125 (10^6)	0.119 (all)
Pyrazinamide	**0.318 (3.1)**	0.371 (4.4)	0.353 (481.2)	0.342 (10^6)	0.382 (all)
Rifampicin	**0.031 (1.4)**	**0.031** (1.5)	**0.031** (130.0)	0.196 (10^6)	0.204 (all)
Streptomycin	0.050 **(1.0)**	0.052 (1.6)	**0.043** (98.8)	0.137 (10^6)	0.148 (all)
P. aeruginosa					
Amikacin	0.175 **(4.9)**	0.206 (14.1)	0.187 (11514.6)	**0.164** (10^6)	0.179 (all)
Doripenem	0.270 **(1.4)**	**0.261** (1.9)	**0.261** (950.0)	0.275 (10^6)	0.281 (all)
Levofloxacin	**0.072** (1.2)	0.076 (1.0)	0.085 (148.9)	0.212 (10^6)	0.225 (all)
Meropenem	0.267 (1.6)	**0.261** (1.0)	0.328 (5368.5)	0.327 (10^6)	0.331 (all)
S. pneumoniae					
Benzylpenicillin	0.013 **(1.1)**	0.012 (2.3)	**0.011** (124.9)	0.013 (10^6)	0.015 (all)
Erythromycin	**0.037 (2.0)**	0.047 (3.8)	0.041 (328.8)	0.042 (10^6)	0.047 (all)
Tetracycline	0.031 **(1.1)**	**0.029** (1.2)	0.032 (1108.5)	0.037 (10^6)	0.037 (all)

- SCM apprend les modèles les plus parcimonieux
- Pour la plupart des datasets, SCM généralise bien et surpasse le baseline

Comparaison à d'autres algorithmes

Dataset	SCM	χ^2 + CART	χ^2 + L1SVM	χ^2 + L2SVM	PolySVM
C. difficile					
Azithromycin	0.030 **(3.3)**	0.086 (7.2)	0.064 (20326.0)	0.056 (10^6)	0.048 (all)
Ceftriaxone	0.073 **(2.6)**	0.117 (6.8)	0.087 (8114.1)	0.102 (10^6)	0.076 (all)
Clarithromycin	0.011 **(3.0)**	0.070 (8.0)	0.062 (36686.1)	0.059 (10^6)	0.053 (all)
Clindamycin	0.021 **(1.4)**	0.011 (2.0)	0.009 (598.2)	0.021 (10^6)	0.039 (all)
Moxifloxacin	0.020 **(1.0)**	0.020 (1.3)	0.020 (25.6)	0.048 (10^6)	0.048 (all)
M. tuberculosis					
Ethambutol	0.179 **(1.4)**	0.185 (1.9)	0.153 (201.3)	0.221 (10^6)	0.221 (all)
Isoniazid	0.021 **(1.0)**	0.021 (1.1)	0.017 (104.7)	0.125 (10^6)	0.119 (all)
Pyrazinamide	0.318 **(3.1)**	0.371 (4.4)	0.353 (481.2)	0.342 (10^6)	0.382 (all)
Rifampicin	0.031 **(1.4)**	0.031 (1.5)	0.031 (130.0)	0.196 (10^6)	0.204 (all)
Streptomycin	0.050 **(1.0)**	0.052 (1.6)	0.043 (98.8)	0.137 (10^6)	0.148 (all)
P. aeruginosa					
Amikacin	0.175 **(4.9)**	0.206 (14.1)	0.187 (11514.6)	0.164 (10^6)	0.179 (all)
Doripenem	0.270 **(1.4)**	0.261 (1.9)	0.261 (950.0)	0.275 (10^6)	0.281 (all)
Levofloxacin	0.072 (1.2)	0.076 (1.0)	0.085 (148.9)	0.212 (10^6)	0.225 (all)
Meropenem	0.267 (1.6)	0.261 (1.0)	0.328 (5368.5)	0.327 (10^6)	0.331 (all)
S. pneumoniae					
Benzylpenicillin	0.013 **(1.1)**	0.012 (2.3)	0.011 (124.9)	0.013 (10^6)	0.015 (all)
Erythromycin	0.037 **(2.0)**	0.047 (3.8)	0.041 (328.8)	0.042 (10^6)	0.047 (all)
Tetracycline	0.031 **(1.1)**	0.029 (1.2)	0.032 (1108.5)	0.037 (10^6)	0.037 (all)

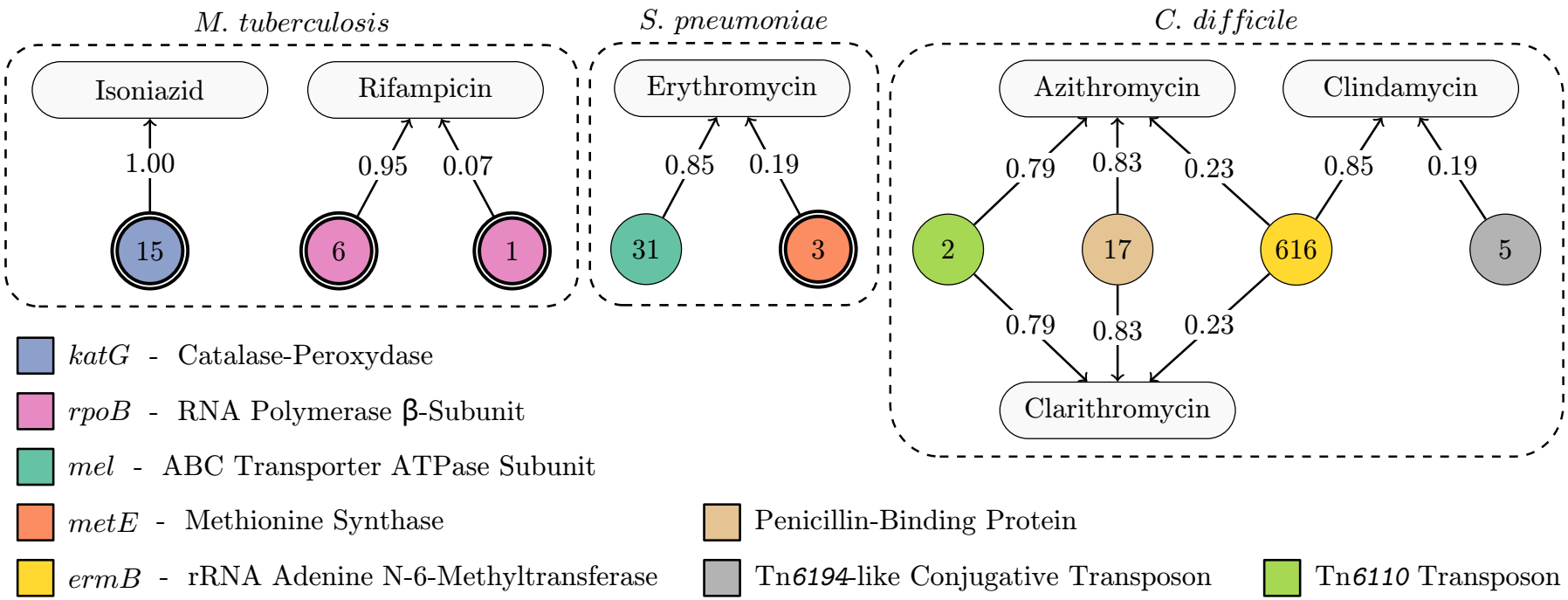
- SCM apprend les modèles les plus parcimonieux
- Pour la plupart des datasets, SCM généralise bien et surpasse le baseline

Comparaison à d'autres algorithmes

Dataset	SCM	χ^2 + CART	χ^2 + L1SVM	χ^2 + L2SVM	PolySVM
C. difficile					
Azithromycin	**0.030** (3.3)	0.086 (7.2)	0.064 (20326.0)	0.056 (10^6)	0.048 (all)
Ceftriaxone	**0.073** (2.6)	0.117 (6.8)	0.087 (8114.1)	0.102 (10^6)	0.076 (all)
Clarithromycin	**0.011** (3.0)	0.070 (8.0)	0.062 (36686.1)	0.059 (10^6)	0.053 (all)
Clindamycin	0.021 (1.4)	0.011 (2.0)	**0.009** (598.2)	0.021 (10^6)	0.039 (all)
Moxifloxacin	**0.020** (1.0)	**0.020** (1.3)	**0.020** (25.6)	0.048 (10^6)	0.048 (all)
M. tuberculosis					
Ethambutol	0.179 (1.4)	0.185 (1.9)	**0.153** (201.3)	0.221 (10^6)	0.221 (all)
Isoniazid	0.021 (1.0)	0.021 (1.1)	**0.017** (104.7)	0.125 (10^6)	0.119 (all)
Pyrazinamide	**0.318** (3.1)	0.371 (4.4)	0.353 (481.2)	0.342 (10^6)	0.382 (all)
Rifampicin	**0.031** (1.4)	**0.031** (1.5)	**0.031** (130.0)	0.196 (10^6)	0.204 (all)
Streptomycin	0.050 (1.0)	0.052 (1.6)	**0.043** (98.8)	0.137 (10^6)	0.148 (all)
P. aeruginosa					
Amikacin	0.175 (4.9)	0.206 (14.1)	0.187 (11514.6)	**0.164** (10^6)	0.179 (all)
Doripenem	0.270 (1.4)	**0.261** (1.9)	**0.261** (950.0)	0.275 (10^6)	0.281 (all)
Levofloxacin	**0.072** (1.2)	0.076 (1.0)	0.085 (148.9)	0.212 (10^6)	0.225 (all)
Meropenem	0.267 (1.6)	**0.261** (1.0)	0.328 (5368.5)	0.327 (10^6)	0.331 (all)
S. pneumoniae					
Benzylpenicillin	0.013 (1.1)	0.012 (2.3)	**0.011** (124.9)	0.013 (10^6)	0.015 (all)
Erythromycin	**0.037** (2.0)	0.047 (3.8)	0.041 (328.8)	0.042 (10^6)	0.047 (all)
Tetracycline	0.031 (1.1)	**0.029** (1.2)	0.032 (1108.5)	0.037 (10^6)	0.037 (all)

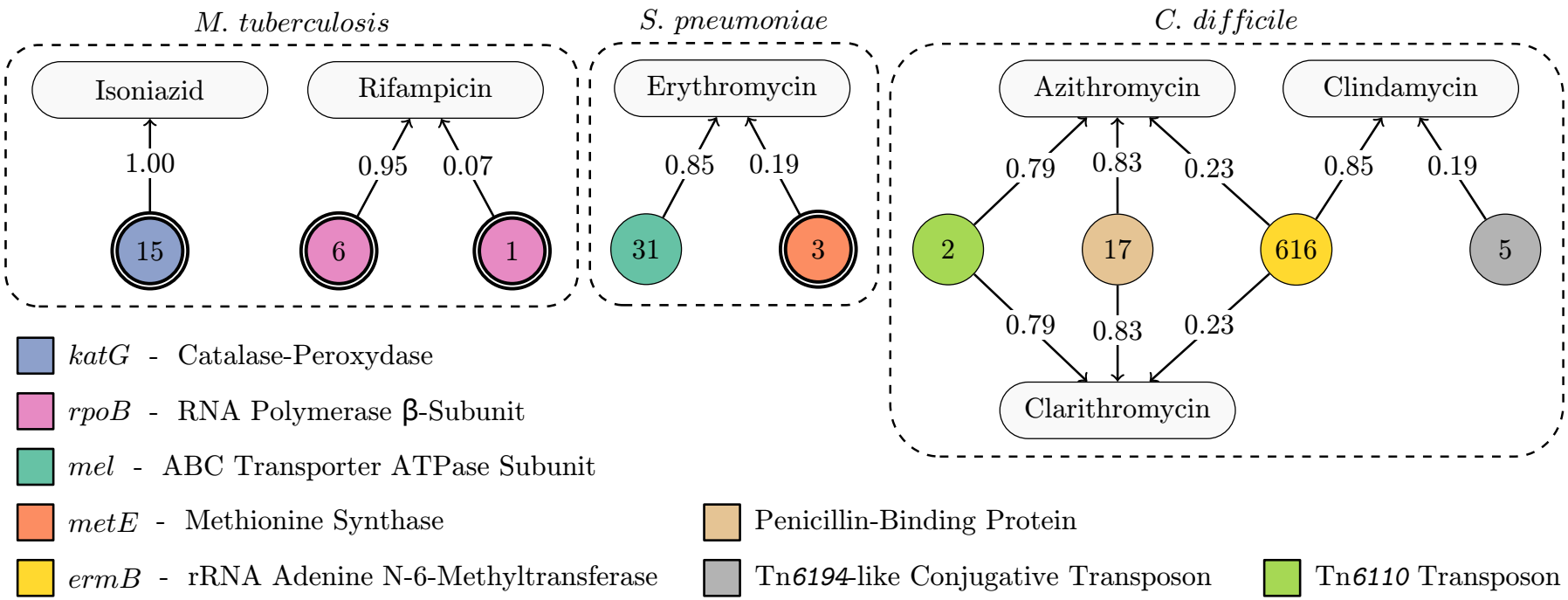
- SCM apprend les modèles les plus parcimonieux
- Pour la plupart des datasets, SCM généralise bien et surpasse le baseline

Les modèles obtenus sont interprétables



En quelques heures de calcul and sans connaissances *a priori*, nous avons retrouvé des mécanismes de résistance aux antibiotiques connus.

Les modèles obtenus sont interprétables

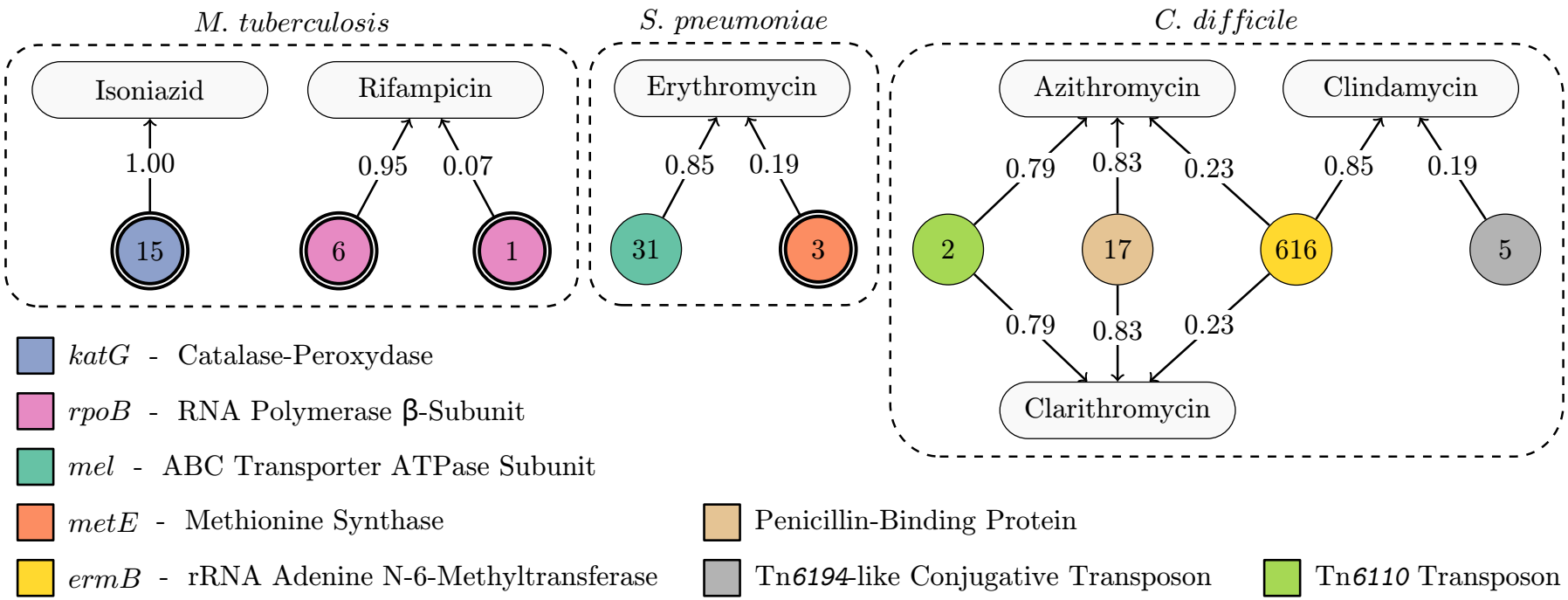


Protein Sci. 2010 Mar;19(3):458-74. doi: 10.1002/pro.324.

Isoniazid-resistance conferring mutations in Mycobacterium tuberculosis KatG: catalase, peroxidase, and INH-NADH adduct formation activities.

Cade CE[1], Dlouhy AC, Medzihradszky KF, Salas-Castillo SP, Ghiladi RA.

Les modèles obtenus sont interprétables

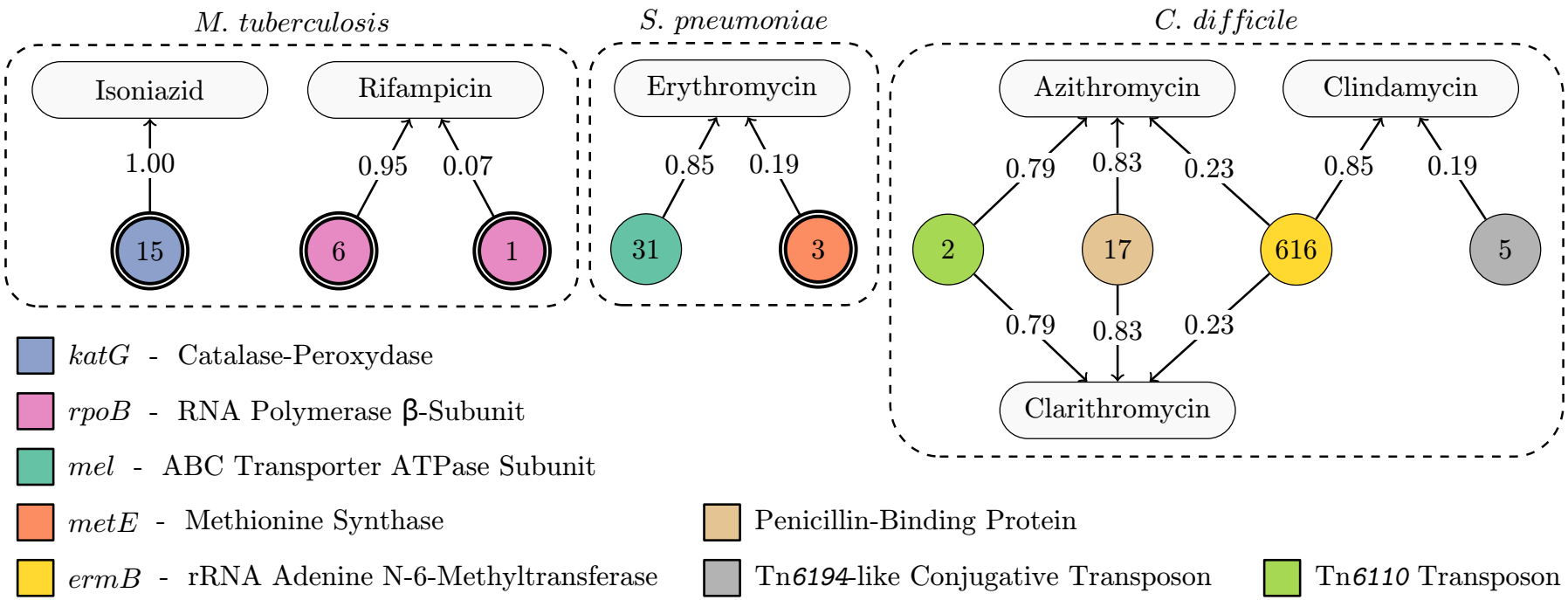


FEMS Microbiol Lett. 1996 Oct 15;144(1):103-8.

Rifampicin resistance and mutation of the rpoB gene in Mycobacterium tuberculosis.

Taniguchi H[1], Aramaki H, Nikaido Y, Mizuguchi Y, Nakamura M, Koga T, Yoshida S.

Les modèles obtenus sont interprétables

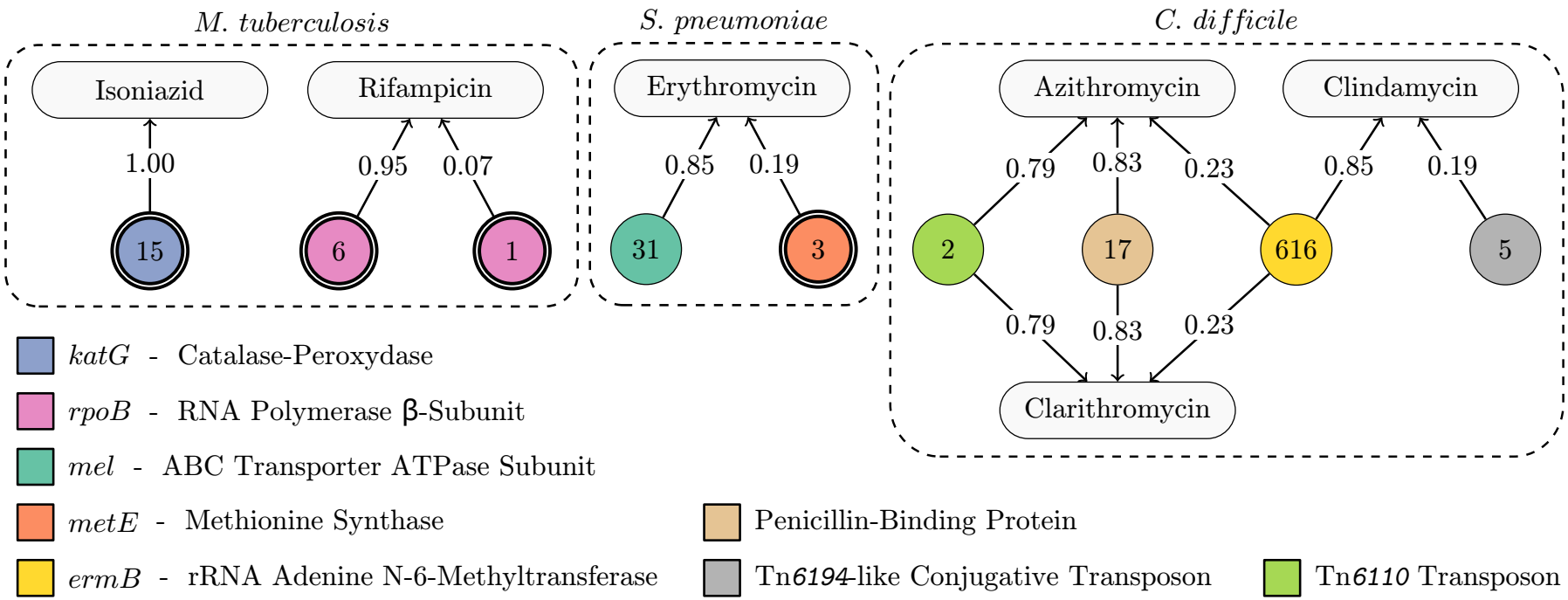


Antimicrob Agents Chemother. 2005 Oct;49(10):4203-9.

Macrolide efflux in Streptococcus pneumoniae is mediated by a dual efflux pump (mel and mef) and is erythromycin inducible.

Ambrose KD[1], Nisbet R, Stephens DS.

Les modèles obtenus sont interprétables

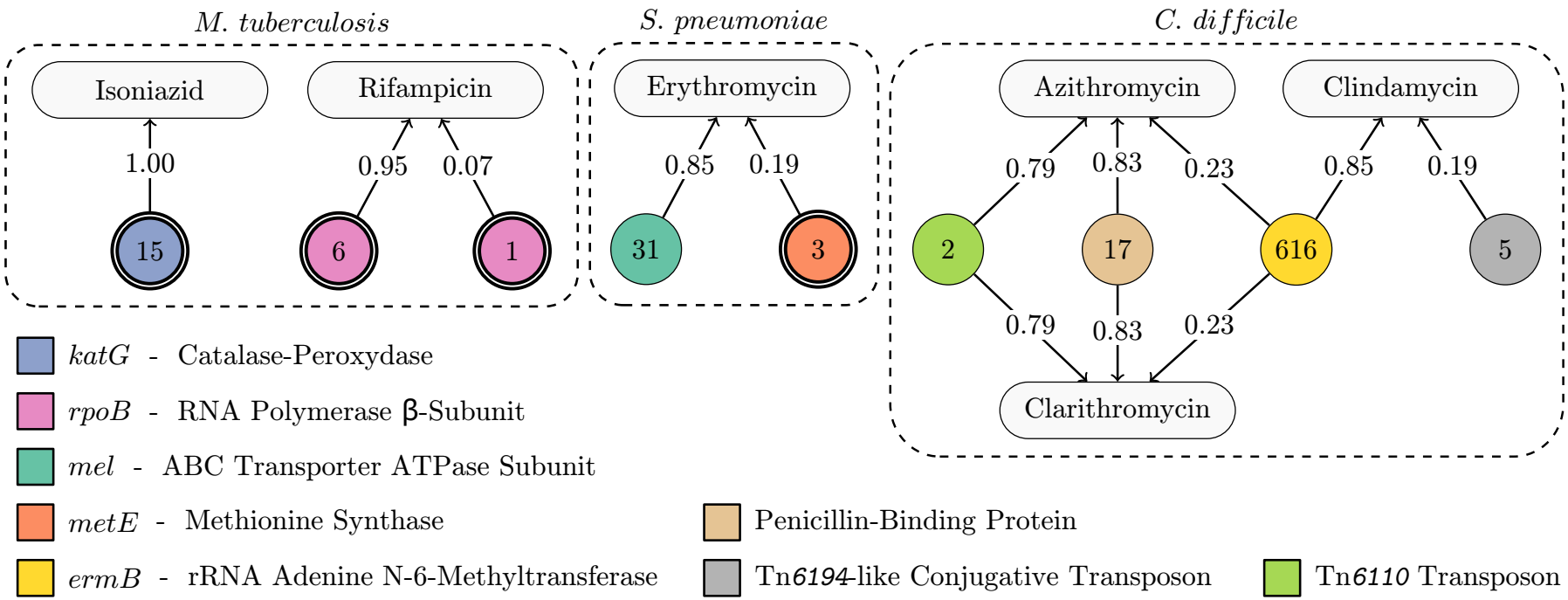


PLoS One. 2013;8(1):e49638. doi: 10.1371/journal.pone.0049638. Epub 2013 Jan 22.

The effects of methionine acquisition and synthesis on Streptococcus pneumoniae growth and virulence.

Basavanna S[1], Chimalapati S, Maqbool A, Rubbo B, Yuste J, Wilson RJ, Hosie A, Ogunniyi AD, Paton JC, Thomas G, Brown JS.

Les modèles obtenus sont interprétables

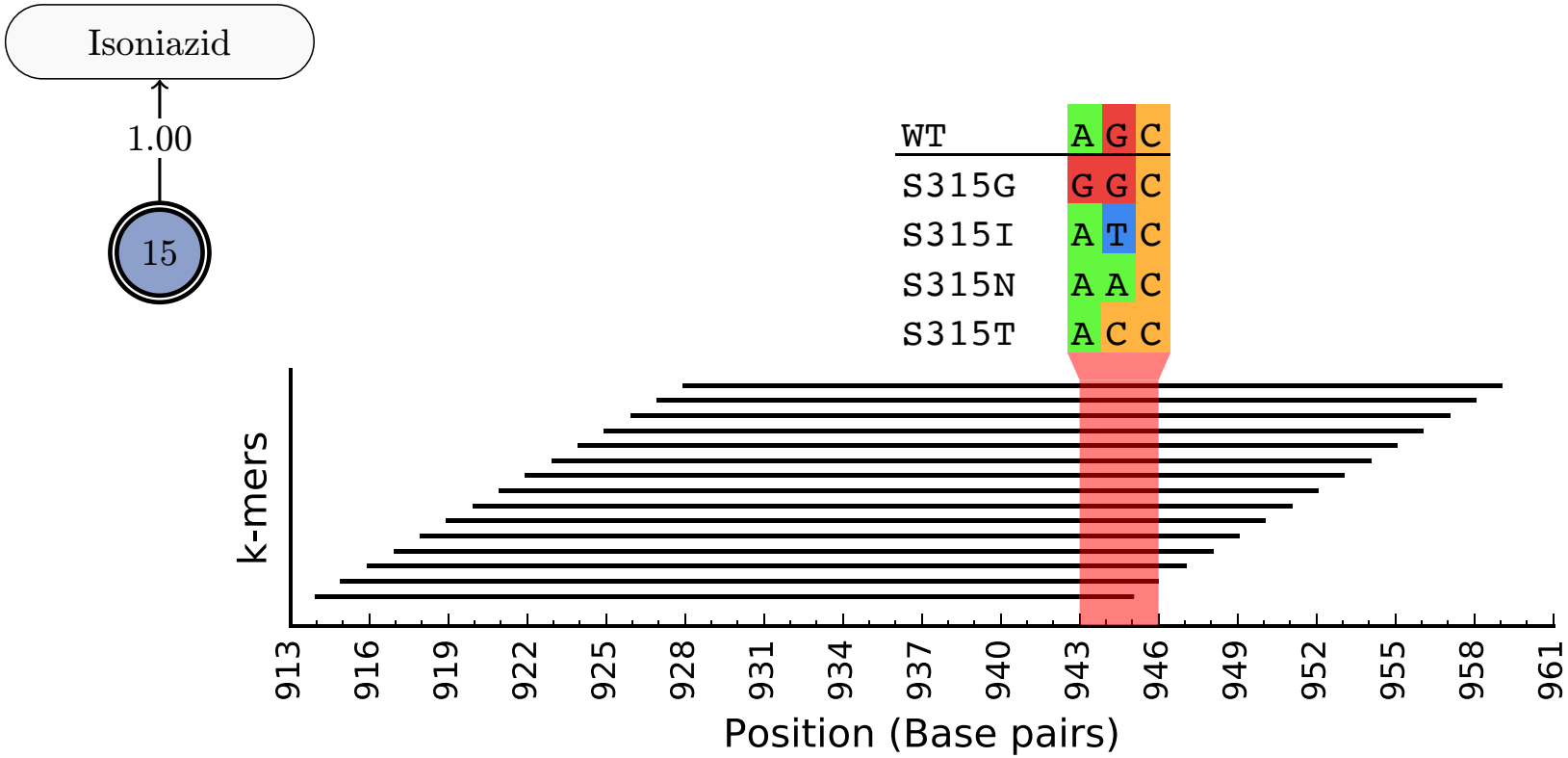


Antimicrob Agents Chemother. 2000 Feb;44(2):411-3.

The macrolide-lincosamide-streptogramin B resistance determinant from Clostridium difficile 630 contains two erm(B) genes.

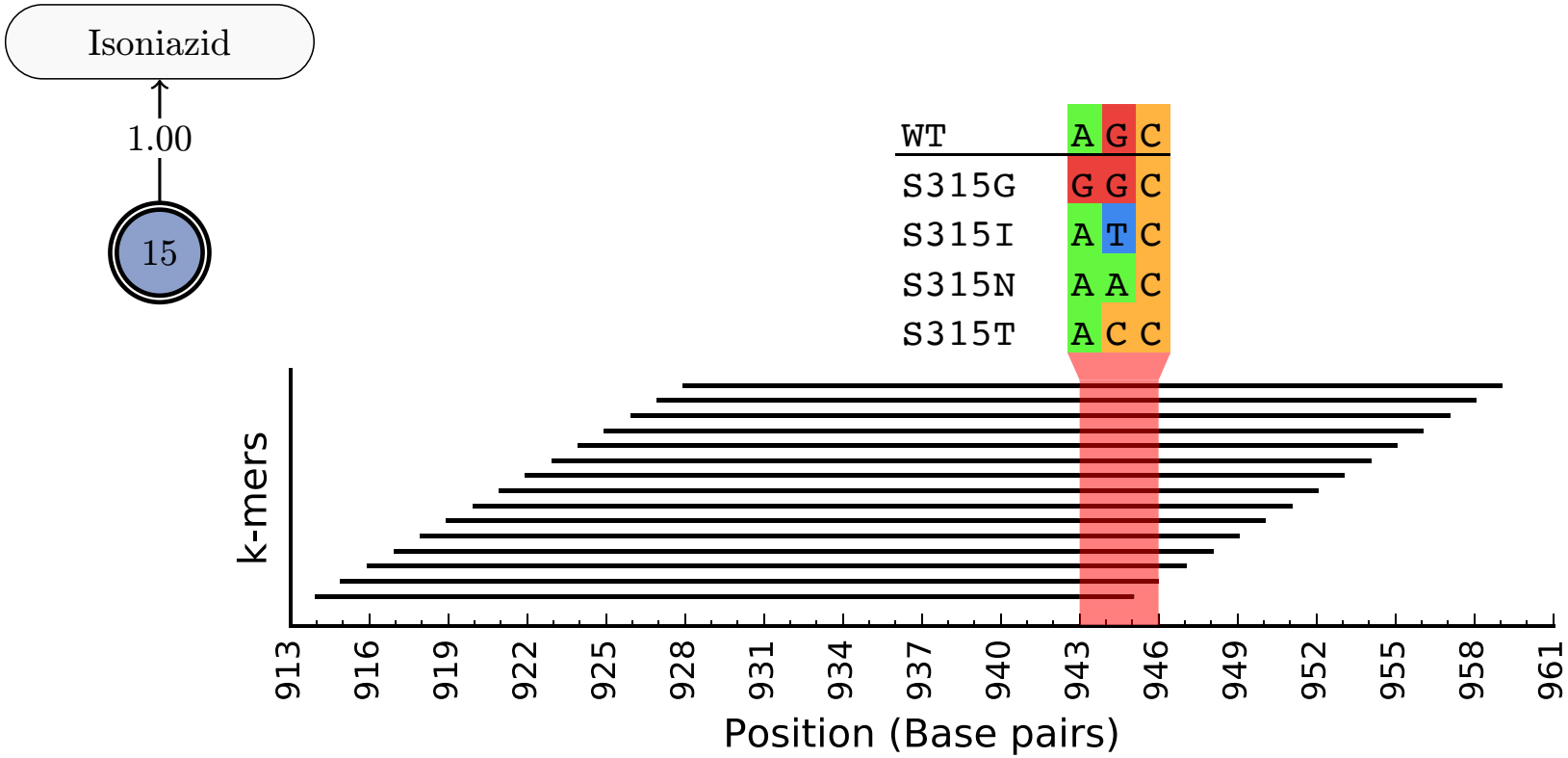
Farrow KA[1], Lyras D, Rood JI.

Au delà des *k*-mers



- Des mutations connues pour causer la résistance ont été retrouvées (S315G, S315I, S315N et S315T)
- Le modèle s'appuie sur l'absence de la séquence de type sauvage (WT), capturant efficacement la présence de ces quatres variantes

Au delà des *k*-mers



- Des mutations connues pour causer la résistance ont été retrouvées (S315G, S315I, S315N et S315T)
- Le modèle s'appuie sur l'absence de la séquence de type sauvage (WT), capturant efficacement la présence de ces quatres variantes

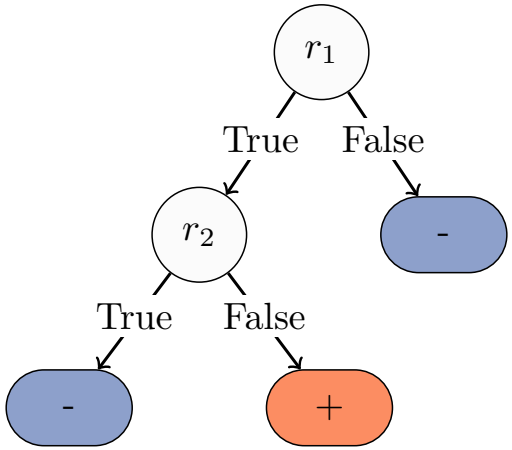
Nouveaux résultats

Apprentissage d'arbres de décision

- Classification and regression trees (Breiman, 1984)
- Nous avons réussi à obtenir une implémentation *out-of-core* de CART
- Mêmes propriétés computationnelles que SCM

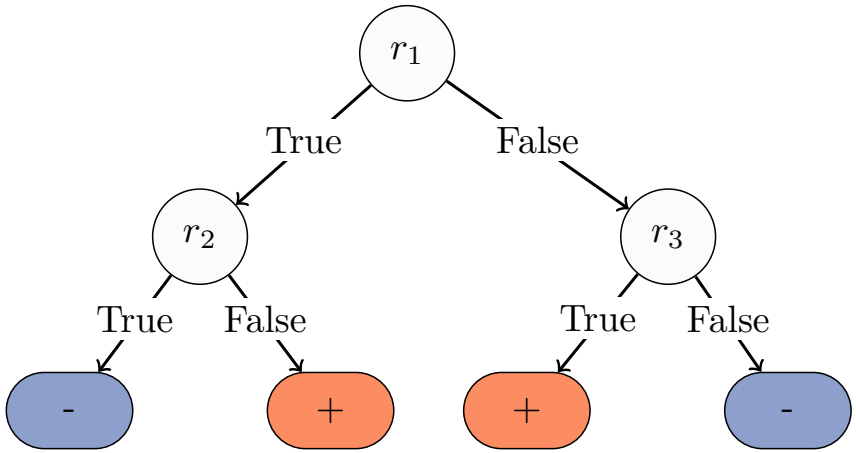
Arbre de décision vs conjonction

Conjunction



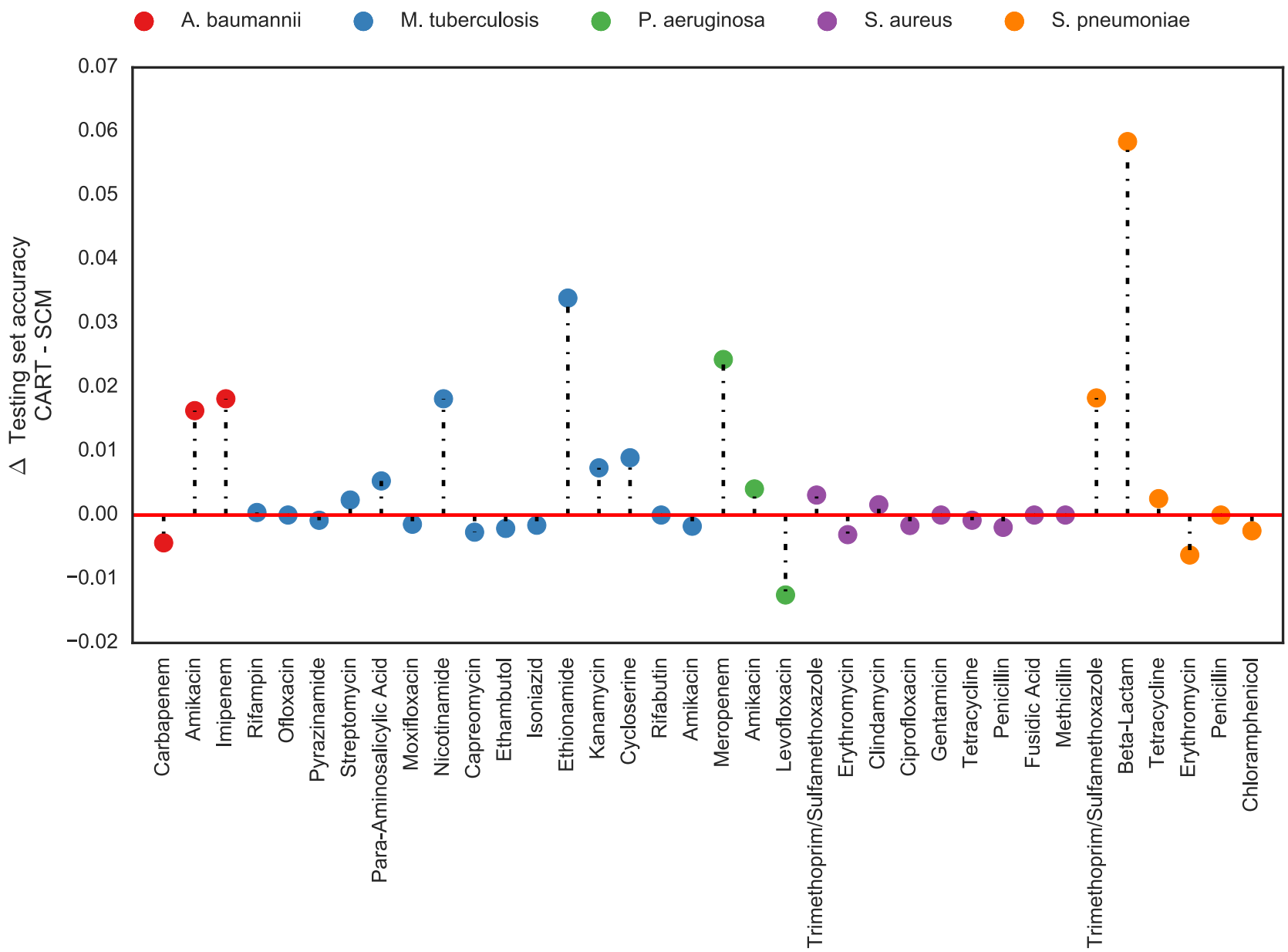
$$r_1 \wedge \neg r_2$$

Decision tree

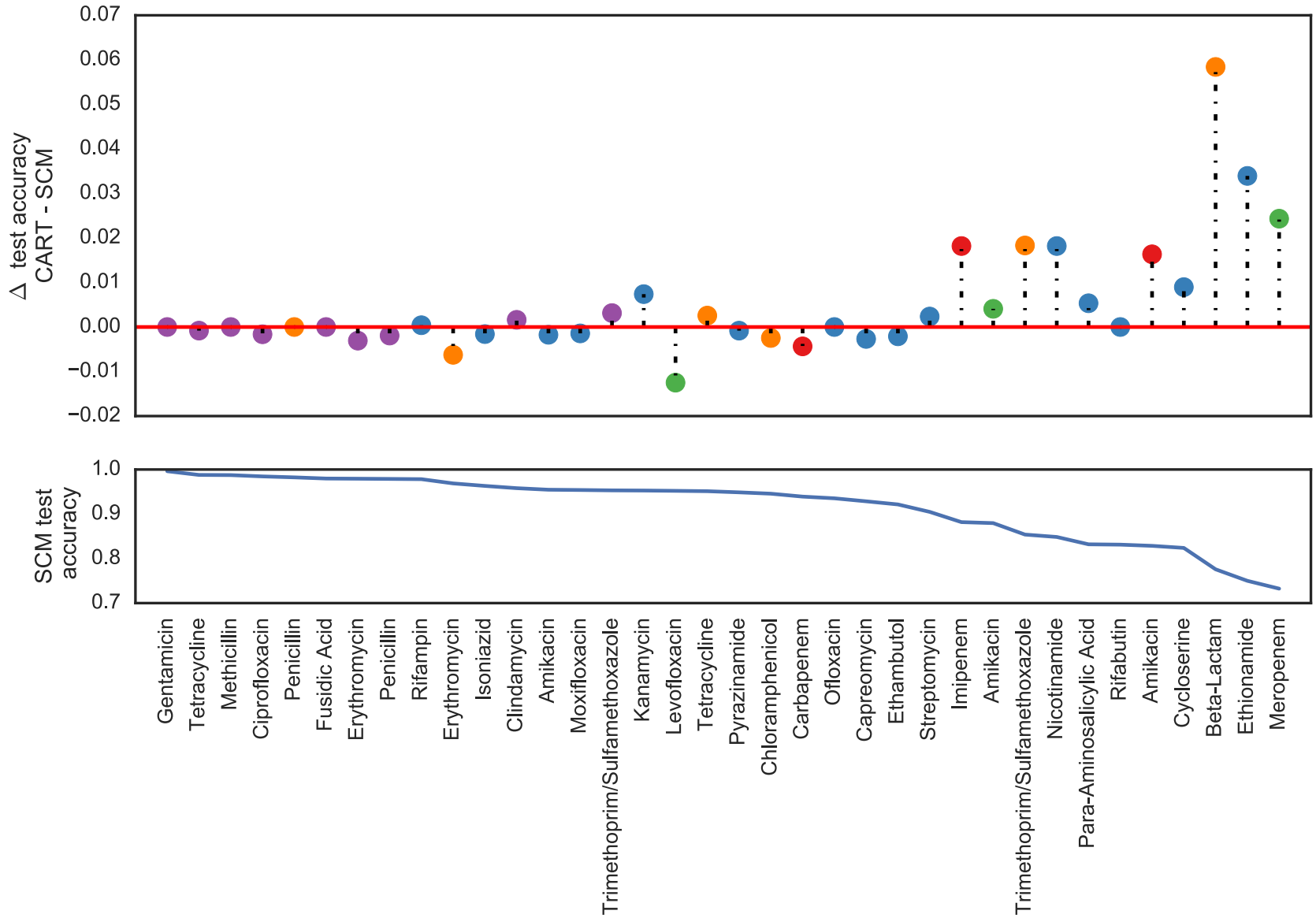


$$(r_1 \wedge \neg r_2) \vee (\neg r_1 \wedge r_3)$$

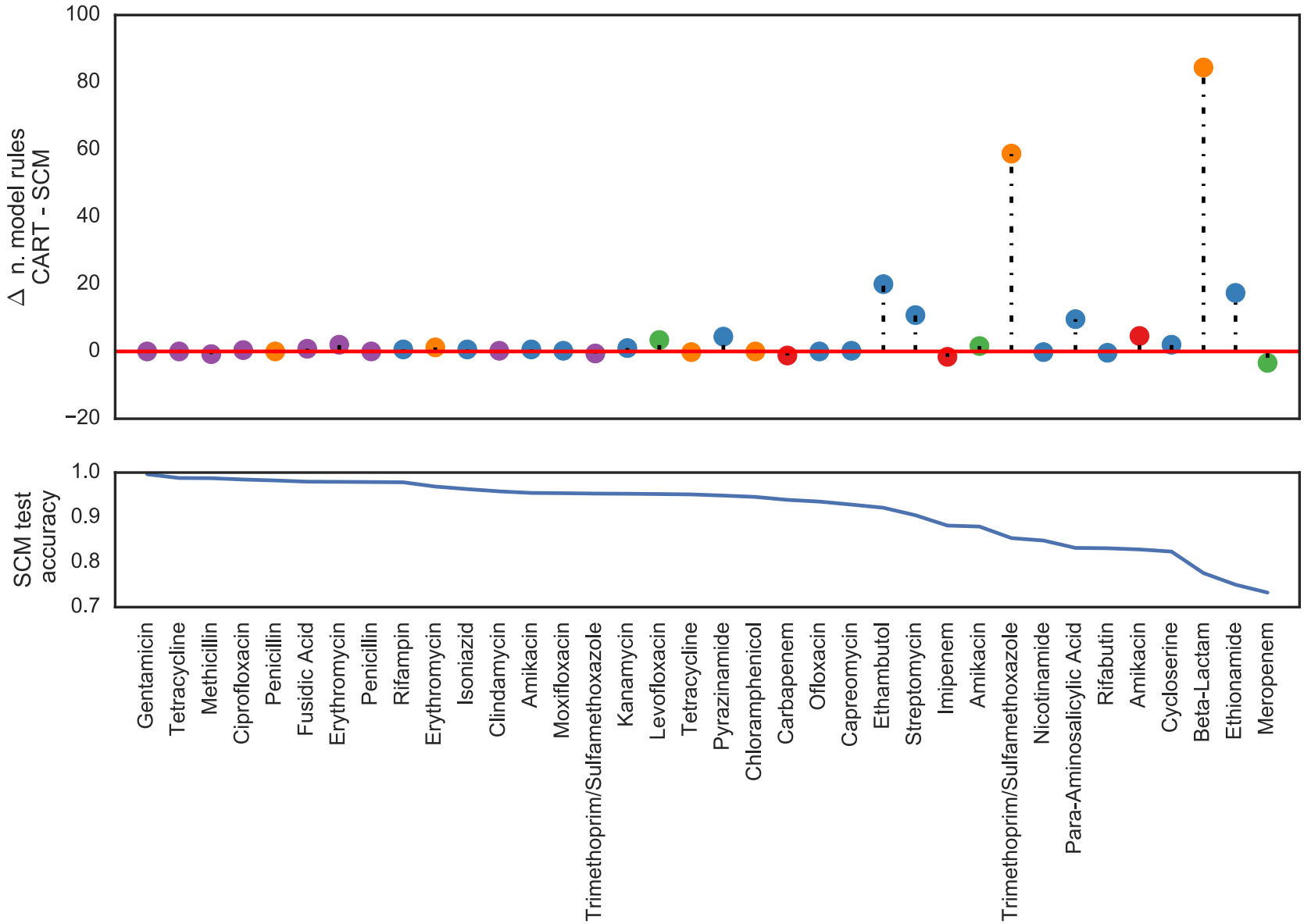
CART surpasse SCM pour certains ensembles de données



CART est meilleur où SCM est moins bon



CART utilise plus de règles que SCM



Conclusion

Conclusion

- Modèles exacts : modèles de la résistance aux antibiotiques faisant très peu d'erreurs de prédiction
- Interprétables : les modèles soulignent l'importance d'un ensemble concis de k-mers
- Largement applicable : autres organismes et phénotypes
- Gros volumes de données : implémentation basée sur le disque
- Garanties mathématiques sur l'exactitude des modèles (à venir pour CART)

Conclusion

- Modèles exacts : modèles de la résistance aux antibiotiques faisant très peu d'erreurs de prédiction
- Interprétables : les modèles soulignent l'importance d'un ensemble concis de k-mers
- Largement applicable : autres organismes et phénotypes
- Gros volumes de données : implémentation basée sur le disque
- Garanties mathématiques sur l'exactitude des modèles (à venir pour CART)

Conclusion

- Modèles exacts : modèles de la résistance aux antibiotiques faisant très peu d'erreurs de prédiction
- Interprétables : les modèles soulignent l'importance d'un ensemble concis de k-mers
- Largement applicable : autres organismes et phénotypes
- Gros volumes de données : implémentation basée sur le disque
- Garanties mathématiques sur l'exactitude des modèles (à venir pour CART)

Conclusion

- Modèles exacts : modèles de la résistance aux antibiotiques faisant très peu d'erreurs de prédiction
- Interprétables : les modèles soulignent l'importance d'un ensemble concis de k-mers
- Largement applicable : autres organismes et phénotypes
- Gros volumes de données : implémentation basée sur le disque
- Garanties mathématiques sur l'exactitude des modèles (à venir pour CART)

Conclusion

- Modèles exacts : modèles de la résistance aux antibiotiques faisant très peu d'erreurs de prédiction
- Interprétables : les modèles soulignent l'importance d'un ensemble concis de k-mers
- Largement applicable : autres organismes et phénotypes
- Gros volumes de données : implémentation basée sur le disque
- Garanties mathématiques sur l'exactitude des modèles (à venir pour CART)

Travaux futurs

- Abondance des k-mers au lieu de la présence/absence
- Apprentissage semi-supervisé exploiter les données non-étiquetées (phylogénie)
- Application au génome humain

Merci !


Frédéric Raymond


Sébastien Giguère


Maxime Déraspe


Gaël L. St-Pierre


François Laviolette


Mario Marchand


Jacques Corbeil

KOVER

`http://github.com/aldro61/kover`

Drouin et al. (2016). **Predictive computational phenotyping and biomarker discovery using reference-free genome comparisons**. BMC genomics, 17(1), 754.

Contact: alexandre.drouin.8@ulaval.ca

NSERC CRSNG