Rule-based machine learning algorithms for antibiotic resistance prediction

Alexandre Drouin

Department of Computer Science and Software Engineering, Université Laval
Department of Molecular Medicine, Université Laval

Great Lakes Bioinformatics Conference
Chicago, Illinois
May 16, 2017
Predicting antibiotic resistance

Resistant vs Sensitive

Learning algorithm

Model

? 

Sensitive
### Data Set

A data set is a collection of genomes and associated labels:

\[
S \overset{\text{def}}{=} \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \sim D^m
\]

\(x \in \mathcal{X} \overset{\text{def}}{=} \{A, C, G, T\}^*\) is a genome

\(y \in \{0, 1\}\) is a label (resistant or sensitive)

\(D\) is a data generating distribution

### Objective

1. Define a suitable (vectorial) representation for genomes \(\phi : \mathcal{X} \rightarrow \mathbb{R}^d\)

2. Find a predictor \(h : \mathbb{R}^d \rightarrow \{0, 1\}\) that has a good generalization performance, i.e., that minimizes:

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

3. The predictor must be interpretable!
Formalization as a Supervised Learning Problem

Data Set

A data set is a collection of genomes and associated labels:

\[ S \overset{\text{def}}{=} \{ (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \} \sim D^m \]

- \( x \in \mathcal{X} \overset{\text{def}}{=} \{ A, C, G, T \}^* \) is a genome
- \( y \in \{ 0, 1 \} \) is a label (resistant or sensitive)
- \( D \) is a data generating distribution

Objective

1. Define a suitable (vectorial) representation for genomes \( \phi : \mathcal{X} \rightarrow \mathbb{R}^d \)
2. Find a predictor \( h : \mathbb{R}^d \rightarrow \{ 0, 1 \} \) that has a good generalization performance, i.e. that minimizes:

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

3. The predictor must be interpretable!
Formalization as a Supervised Learning Problem

Data Set

A data set is a collection of genomes and associated labels:

\[ S \overset{\text{def}}{=} \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \sim D^m \]

\( x \in \mathcal{X} \overset{\text{def}}{=} \{A, C, G, T\}^* \) is a genome

\( y \in \{0, 1\} \) is a label (resistant or sensitive)

\( D \) is a data generating distribution

Objective

1. Define a suitable (vectorial) representation for genomes \( \phi : \mathcal{X} \rightarrow \mathbb{R}^d \)

2. Find a predictor \( h : \mathbb{R}^d \rightarrow \{0, 1\} \) that has a good generalization performance, i.e. that minimizes:

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

3. The predictor must be interpretable!
Formalization as a Supervised Learning Problem

**Data Set**

A data set is a collection of genomes and associated labels:

\[ S \overset{\text{def}}{=} \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \sim D^m \]

- \( x \in \mathcal{X} \overset{\text{def}}{=} \{A, C, G, T\}^* \) is a genome
- \( y \in \{0, 1\} \) is a label (resistant or sensitive)
- \( D \) is a data generating distribution

**Objective**

1. Define a suitable (vectorial) representation for genomes \( \phi : \mathcal{X} \rightarrow \mathbb{R}^d \)
2. Find a predictor \( h : \mathbb{R}^d \rightarrow \{0, 1\} \) that has a good generalization performance, i.e. that minimizes:

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

3. The predictor must be interpretable!
Genome Representation: the $k$-mer profile

**Definition**

- $k$-mer: a sequence of $k$ nucleotides
- $\mathcal{K}$: the set of all $k$-mers that are in at least one genome of $S$

Each genome is represented by its **$k$-mer profile**, which is a binary vector indicating the presence (1) or absence (0) of each $k$-mer.

\[
\mathcal{K} = \{ \text{CAGATA, GATAGA, GAACAG, CGATGA, AGATAG, AGAACA, ATAGAA, CCGGCT, AACAGC, TAGAAC, TTTCGG, AAATAC} \}
\]

\[
x = \text{CAGATAGAACAGC}
\]

\[
\phi(x) = \begin{bmatrix}
1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0
\end{bmatrix}
\]
Rule-based models

- **Input:** A set of boolean-valued rules (presence/absence of $k$-mers) and a data set (examples, labels)

\[ \phi(x) \rightarrow \text{Rule} \rightarrow \text{True} \text{ or False} \]

- **Output:**
Rule-based models

- **Input:** A set of boolean-valued rules (presence/absence of $k$-mers) and a data set (examples, labels)

- **Output:**

Conjunction

$$r_1 \land \neg r_2$$

Decision tree

$$(r_1 \land \neg r_2) \lor (\neg r_1 \land r_3)$$
Objective: Find the shortest conjunction (logical-AND) or disjunction (logical-OR) of rules that most accurately predicts the labels
**Objective:** Find the **shortest** conjunction (logical-AND) or disjunction (logical-OR) of rules that **most accurately predicts** the labels.
Previous work on Set Covering Machines (Drouin et al., 2016)

![KOver Logo](http://github.com/aldro61/kover)

Key findings:

- The SCM algorithm can be implemented out-of-core
- Theoretical results stressing the importance of sparsity
- Interpretable models for a variety of antibiotics and species
- Machine learning approaches recover true resistance mechanisms
Previous work on Set Covering Machines (Drouin et al., 2016)

Key findings:

- The SCM algorithm can be implemented out-of-core
- Theoretical results stressing the importance of sparsity
- Interpretable models for a variety of antibiotics and species
- Machine learning approaches recover true resistance mechanisms
Previous work on Set Covering Machines (Drouin et al., 2016)

Key findings:

- The SCM algorithm can be implemented out-of-core
- Theoretical results stressing the importance of sparsity
- **Interpretable models** for a variety of antibiotics and species
- Machine learning approaches recover **true resistance mechanisms**

http://github.com/aldro61/kover
Key findings:

- The SCM algorithm can be implemented *out-of-core*
- Theoretical results stressing the importance of *sparsity*
- Interpretable models for a variety of antibiotics and species
- Machine learning approaches recover *true resistance mechanisms*
Within hours of computation and without prior knowledge, known antibiotic resistance mechanisms are recovered.
Interpretable models of antibiotic resistance

M. tuberculosis

- Isoniazid
  - 1.00
  - 15

- Rifampicin
  - 0.95
  - 6
  - 1

S. pneumoniae

- Erythromycin
  - 0.79
  - 31
  - 3

- Azithromycin
  - 0.85
  - 0.19

- Clarithromycin
  - 0.83
  - 0.79
  - 0.23

- Clindamycin
  - 0.85
  - 0.19

C. difficile

- Azithromycin
  - 0.79
  - 0.83
  - 0.23

- Clindamycin
  - 0.85
  - 0.19

- Penicillin-Binding Protein

- katG - Catalase-Peroxidase
- rpoB - RNA Polymerase β-Subunit
- mel - ABC Transporter ATPase Subunit
- metE - Methionine Synthase
- ermB - rRNA Adenine N-6-Methyltransferase

- Tn6194-like Conjugative Transposon
- Tn6110 Transposon


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

Cade CE¹, Dlouhy AC, Medzihradszky KF, Salas-Castillo SP, Ghiladi RA.
Interpretable models of antibiotic resistance

*M. tuberculosis*
- Isoniazid
  - 1.00
  - 15
- Rifampicin
  - 0.95
  - 6
  - 1

*S. pneumoniae*
- Erythromycin
  - 0.85
  - 31
  - 3

*C. difficile*
- Azithromycin
  - 0.79
  - 2
  - 0.83
  - 17
  - 0.23
  - 616
- Clindamycin
  - 0.85
  - 5

**Gene Variations**
- *katG* - Catalase-Peroxydase
- *rpoB* - RNA Polymerase β-Subunit
- *mel* - ABC Transporter ATPase Subunit
- *metE* - Methionine Synthase
- *ermB* - rRNA Adenine N-6-Methyltransferase

**Transposons**
- Tn6194-like Conjugative Transposon
- Tn6110 Transposon

**References**

Rifampicin resistance and mutation of the rpoB gene in Mycobacterium tuberculosis.
Taniguchi H¹, Aramaki H, Nikaido Y, Mizuguchi Y, Nakamura M, Koga T, Yoshida S.
Interpretable models of antibiotic resistance

M. tuberculosis

<table>
<thead>
<tr>
<th>Antibiotic</th>
<th>Resistance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isoniazid</td>
<td>1.00</td>
</tr>
<tr>
<td>Rifampicin</td>
<td>0.95 0.07</td>
</tr>
<tr>
<td>Erythromycin</td>
<td>0.85 0.19</td>
</tr>
</tbody>
</table>

S. pneumoniae

<table>
<thead>
<tr>
<th>Antibiotic</th>
<th>Resistance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azithromycin</td>
<td>0.79 0.23</td>
</tr>
<tr>
<td>Clarithromycin</td>
<td>0.85 0.19</td>
</tr>
</tbody>
</table>

C. difficile

<table>
<thead>
<tr>
<th>Antibiotic</th>
<th>Resistance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azithromycin</td>
<td>0.79 0.23</td>
</tr>
<tr>
<td>Clarithromycin</td>
<td>0.85 0.19</td>
</tr>
</tbody>
</table>

**KatG** - Catalase-Peroxydase  
**rpoB** - RNA Polymerase β-Subunit  
**mel** - ABC Transporter ATPase Subunit  
**metE** - Methionine Synthase  
**ermB** - rRNA Adenine N-6-Methyltransferase  
**Penicillin-Binding Protein**  
**Tn6194-like Conjugative Transposon**  
**Tn6110 Transposon**


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

Ambrose KD¹, Nisbet R, Stephens DS.

Alexandre Drouin (Université Laval)  
Predicting antibiotic resistance  
March 23, 2017  
7 / 17
**Interpretable models of antibiotic resistance**

*M. tuberculosis*

- Isoniazid
  - 1.00 (15)
- Rifampicin
  - 0.95 (6)
  - 0.07 (1)

*S. pneumoniae*

- Erythromycin
  - 0.85 (31)
  - 0.19 (3)

*C. difficile*

- Azithromycin
  - 0.79 (2)
  - 0.83 (17)
  - 0.23 (616)
- Clindamycin
  - 0.85 (5)

**Genetic markers**

- *katG* - Catalase-Peroxydase
- *rpoB* - RNA Polymerase β-Subunit
- *mel* - ABC Transporter ATPase Subunit
- *metE* - Methionine Synthase
- *ermB* - rRNA Adenine N-6-Methyltransferase
- Penicillin-Binding Protein
- Tn6194-like Conjugative Transposon
- Tn6110 Transposon

**References**


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

Interpretable models of antibiotic resistance

M. tuberculosis

<table>
<thead>
<tr>
<th>Antibiotic</th>
<th>Resistance Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isoniazid</td>
<td>1.00</td>
</tr>
<tr>
<td>Rifampicin</td>
<td>0.95 0.07</td>
</tr>
<tr>
<td>Erythromycin</td>
<td>0.85 0.19</td>
</tr>
</tbody>
</table>

S. pneumoniae

<table>
<thead>
<tr>
<th>Antibiotic</th>
<th>Resistance Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azithromycin</td>
<td>0.79 0.31</td>
</tr>
<tr>
<td>Clarithromycin</td>
<td>0.83 0.23</td>
</tr>
</tbody>
</table>

C. difficile

<table>
<thead>
<tr>
<th>Antibiotic</th>
<th>Resistance Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azithromycin</td>
<td>0.85 0.19</td>
</tr>
<tr>
<td>Clindamycin</td>
<td>0.79 0.31</td>
</tr>
</tbody>
</table>

katG - Catalase-Peroxidase
rpoB - RNA Polymerase β-Subunit
mel - ABC Transporter ATPase Subunit
metE - Methionine Synthase
ermB - rRNA Adenine N-6-Methyltransferase

Penicillin-Binding Protein
Tn6194-like Conjugative Transposon
Tn6110 Transposon

The macrolide-lincosamide-streptogramin B resistance determinant from Clostridium difficile 630 contains two erm(B) genes.
Farrow KA¹, Lyras D, Rood JI.
Classification and Regression Trees


- SCM and CART share many similarities:
  - Rule-based models
  - Greedy algorithm that uses an heuristic to select rules
  - Both try to use to fewest possible rules

- We created an out-of-core implementation of CART with the same computational properties as our SCM implementation.
SCM and CART share many similarities:

- Rule-based models
- Greedy algorithm that uses an heuristic to select rules
- Both try to use the fewest possible rules

We created an out-of-core implementation of CART with the same computational properties as our SCM implementation.

Great book!
Classification and Regression Trees


- SCM and CART share many similarities:
  - Rule-based models
  - Greedy algorithm that uses an heuristic to select rules
  - Both try to use to fewest possible rules

- We created an out-of-core implementation of CART with the same computational properties as our SCM implementation
An application to the PATRIC database (Wattam et al, 2013)

Extracted all combinations of species/antibiotic for which 25 resistant/sensitive isolates were available (73 data sets)
Metrics for comparison

- **Accuracy:** proportion of correct predictions
- **Sensitivity:** proportion of resistant isolates predicted as resistant
- **Specificity:** proportion of sensitive isolates predicted as sensitive
Metrics for comparison

- **Accuracy**: proportion of correct predictions
- **Sensitivity**: proportion of resistant isolates predicted as resistant
- **Specificity**: proportion of sensitive isolates predicted as sensitive
Metrics for comparison

- **Accuracy:** proportion of correct predictions
- **Sensitivity:** proportion of resistant isolates predicted as resistant
- **Specificity:** proportion of sensitive isolates predicted as sensitive
Both algorithms learn highly accurate models.
Both algorithms learn highly accurate models

Classification Trees

Sensitivity

Specificity

A. baumannii
C. difficile
E. faecium
K. pneumoniae
M. tuberculosis
N. gonorrhoeae
P. aeruginosa
P. difficile
S. aureus
S. enterica
S. haemolyticus
S. pneumoniae
Comparison of the accuracy of both algorithms

Accuracy on unseen data

No clear winner! (p=0.234)
Comparison of the accuracy of both algorithms

... but SCM models rely on much fewer k-mers ($p=2.452e^{-15}$)
The accuracy/interpretability trade-off

Accuracy: 77.5 %
Sensitivity: 89.3%
Specificity: 64.9%
k-mers: 10
The accuracy/interpretability trade-off
The accuracy/interpretability trade-off

Accuracy: 82.1% (↑ 4.6%)
Sensitivity: 83.9% (↓ 5.4%)
Specificity: 80.1% (↑ 15.2%)

\(k\)-mers: 180
Kover AMR Platform - Web-based model analysis

https://aldro61.github.io/kover-amr-platform
Conclusion

- **Interpretability**: rule-based models that highlight the importance of specific $k$-mers

- **Accuracy**: both algorithms generated highly accurate predictive models of antibiotic resistance

- **Scalability**: disk-based implementations for both algorithms

- **Applicability**: applicable to other phenotypes and species

- **Availability**: open-source
Conclusion

- **Interpretability**: rule-based models that highlight the importance of specific $k$-mers

- **Accuracy**: both algorithms generated highly accurate predictive models of antibiotic resistance

- **Scalability**: disk-based implementations for both algorithms

- **Applicability**: applicable to other phenotypes and species

- **Availability**: open-source
Conclusion

- **Interpretability**: rule-based models that highlight the importance of specific $k$-mers

- **Accuracy**: both algorithms generated highly accurate predictive models of antibiotic resistance

- **Scalability**: disk-based implementations for both algorithms

- **Applicability**: applicable to other phenotypes and species

- **Availability**: open-source
Conclusion

- **Interpretability**: rule-based models that highlight the importance of specific \( k \)-mers

- **Accuracy**: both algorithms generated highly accurate predictive models of antibiotic resistance

- **Scalability**: disk-based implementations for both algorithms

- **Applicability**: applicable to other phenotypes and species

- **Availability**: open-source
Conclusion

- **Interpretability**: rule-based models that highlight the importance of specific $k$-mers

- **Accuracy**: both algorithms generated highly accurate predictive models of antibiotic resistance

- **Scalability**: disk-based implementations for both algorithms

- **Applicability**: applicable to other phenotypes and species

- **Availability**: open-source
Future works

- Embedding phylogenetic prior knowledge in the learning algorithm
- Sample-compression theory to reduce overfitting in CART
- Extensions to classification and regression
- Application to human phenotypes
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
Comparison to other learning algorithms

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

- SCM tends to learn the sparsest models
- On most datasets, the SCM generalizes well and outperforms the baseline
### Comparison to other learning algorithms

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

- SCM tends to learn the **sparsest** models
- On most datasets, the SCM generalizes well and outperforms the baseline
Comparison to other learning algorithms

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

- SCM tends to learn the **sparsest** models
- On most datasets, the SCM **generalizes well** and outperforms the baseline
## Choosing the $k$-mer Length

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SCM-31</th>
<th>SCM-CV</th>
<th>$k$-mer length</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C. difficile</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Azithromycin</td>
<td>0.030 (3.3)</td>
<td>0.032 (3.2)</td>
<td>40.0 ± 23.0</td>
</tr>
<tr>
<td>Ceftriaxone</td>
<td>0.073 (2.6)</td>
<td>0.085 (3.0)</td>
<td>41.0 ± 31.3</td>
</tr>
<tr>
<td>Clarithromycin</td>
<td>0.011 (3.0)</td>
<td>0.018 (2.9)</td>
<td>26.0 ± 12.0</td>
</tr>
<tr>
<td>Clindamycin</td>
<td>0.021 (1.4)</td>
<td><strong>0.011</strong> (1.9)</td>
<td>56.0 ± 21.6</td>
</tr>
<tr>
<td>Moxifloxacin</td>
<td><strong>0.020</strong> (1.0)</td>
<td><strong>0.020</strong> (1.0)</td>
<td>39.0 ± 16.0</td>
</tr>
<tr>
<td><strong>M. tuberculosis</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ethambutol</td>
<td>0.179 (1.4)</td>
<td><strong>0.164</strong> (1.2)</td>
<td>49.0 ± 22.7</td>
</tr>
<tr>
<td>Isoniazid</td>
<td><strong>0.021</strong> (1.0)</td>
<td><strong>0.021</strong> (1.0)</td>
<td>24.0 ± 4.6</td>
</tr>
<tr>
<td>Pyrazinamide</td>
<td><strong>0.318</strong> (3.1)</td>
<td>0.353 (2.3)</td>
<td>35.0 ± 24.6</td>
</tr>
<tr>
<td>Rifampicin</td>
<td>0.031 (1.4)</td>
<td><strong>0.006</strong> (1.0)</td>
<td>49.0 ± 6.0</td>
</tr>
<tr>
<td>Streptomycin</td>
<td>0.050 (1.0)</td>
<td><strong>0.048</strong> (1.0)</td>
<td>25.0 ± 4.9</td>
</tr>
<tr>
<td><strong>P. aeruginosa</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amikacin</td>
<td><strong>0.175</strong> (4.9)</td>
<td>0.188 (4.9)</td>
<td>43.0 ± 28.2</td>
</tr>
<tr>
<td>Doripenem</td>
<td><strong>0.270</strong> (1.4)</td>
<td><strong>0.270</strong> (1.8)</td>
<td>39.0 ± 28.6</td>
</tr>
<tr>
<td>Levofloxacin</td>
<td><strong>0.072</strong> (1.2)</td>
<td>0.073 (1.4)</td>
<td>34.0 ± 12.7</td>
</tr>
<tr>
<td>Meropenem</td>
<td><strong>0.267</strong> (1.6)</td>
<td>0.276 (2.0)</td>
<td>32.0 ± 14.5</td>
</tr>
<tr>
<td><strong>S. pneumoniae</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Benzylpenicillin</td>
<td>0.013 (1.1)</td>
<td><strong>0.011</strong> (1.0)</td>
<td>50.0 ± 28.8</td>
</tr>
<tr>
<td>Erythromycin</td>
<td><strong>0.037</strong> (2.0)</td>
<td>0.038 (1.9)</td>
<td>39.0 ± 29.6</td>
</tr>
<tr>
<td>Tetracycline</td>
<td><strong>0.031</strong> (1.1)</td>
<td><strong>0.031</strong> (1.1)</td>
<td>55.0 ± 24.6</td>
</tr>
</tbody>
</table>
Can we expect good generalization?

We can bound the risk of a predictor based on its performance on the training set. The following term bounds the risk of every conjunction $h$ of rules in $R$ with probability $\geq 1 - \delta$.

**Occam’s Razor Bound**

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

where $r$ is the number of errors on the training set, $|h|$ is the number of rules in the conjunction, $\zeta$ is any function such that $\sum_{b \in \mathbb{N}} \zeta(b) \leq 1$.

- The combinatorial term dominates the bound even for classifiers that make few errors.
- The bound seems to indicate bad generalization performance.
Can we expect good generalization?

We can bound the risk of a predictor based on its performance on the training set. The following term bounds the risk of every conjunction $h$ of rules in $\mathcal{R}$ with probability $\geq 1 - \delta$.

**Occam’s Razor Bound**

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

where $r$ is the number of errors on the training set,
$|h|$ is the number of rules in the conjunction,
$\zeta$ is any function such that $\sum_{b \in \mathbb{N}} \zeta(b) \leq 1$

- The combinatorial term dominates the bound even for classifiers that make few errors
- The bound seems to indicate bad generalization performance.
Can we expect good generalization?

We can bound the risk of a predictor based on its performance on the training set. The following term bounds the risk of every conjunction $h$ of rules in $\mathcal{R}$ with probability $\geq 1 - \delta$.

**Occam’s Razor Bound**

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

where $r$ is the number of errors on the training set, $|h|$ is the number of rules in the conjunction, $\zeta$ is any function such that $\sum_{b \in \mathbb{N}} \zeta(b) \leq 1$.

- The combinatorial term dominates the bound even for classifiers that make few errors.
- The bound seems to indicate bad generalization performance.
Can we expect good generalization?

We can bound the risk of a predictor based on its performance on the training set. The following term bounds the risk of every conjunction $h$ of rules in $\mathcal{R}$ with probability $\geq 1 - \delta$.

**Occam’s Razor Bound**

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

where $r$ is the number of errors on the training set,

$|h|$ is the number of rules in the conjunction,

$\zeta$ is any function such that $\sum_{b \in \mathbb{N}} \zeta(b) \leq 1$

- The combinatorial term dominates the bound even for classifiers that make few errors
- The bound seems to indicate bad generalization performance.
Can we expect good generalization?

In the sample compression framework, the predictor $h$ is specified using a small set of training examples ($\mathcal{Z}_i$):

\[
\epsilon \overset{\text{def}}{=} \frac{1}{m - |h| - r} \left[ \ln \left( \frac{m}{|h|} \right) + \ln \left( \frac{m - |h|}{r} \right) \right. \\
\left. + \sum_{x \in \mathcal{Z}_i} \ln(2 \cdot |x|) - \ln(\zeta(|h|) \cdot \zeta(r) \cdot \delta) \right],
\]

where $r$ is the number of errors made on $S \setminus \mathcal{Z}_i$.

- The bound does not depend on $k$ anymore.
- We can consider exponentially more complex feature spaces without any penalty on the generalization error.
Can we expect good generalization?

In the sample compression framework, the predictor $h$ is specified using a small set of training examples ($Z_i$):

**Sample Compression Bound**

$$
\epsilon \overset{\text{def}}{=} \frac{1}{m - |h| - r} \left[ \ln \left( \frac{m}{|h|} \right) + \ln \left( \frac{m - |h|}{r} \right) \\
+ \sum_{x \in Z_i} \ln(2 \cdot |x|) - \ln(\zeta(|h|) \cdot \zeta(r) \cdot \delta) \right],
$$

where $r$ is the number of errors made on $S \setminus Z_i$.

- The bound does not depend on $k$ anymore
- We can consider exponentially more complex feature spaces without any penalty on the generalization error.
Can we expect good generalization?

In the sample compression framework, the predictor $h$ is specified using a small set of training examples ($\mathcal{Z}_i$):

**Sample Compression Bound**

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

where $r$ is the number of errors made on $S \setminus \mathcal{Z}_i$.

- The bound does not depend on $k$ anymore.
- We can consider exponentially more complex feature spaces without any penalty on the generalization error.
Can we expect good generalization?

In the sample compression framework, the predictor $h$ is specified using a small set of training examples ($Z_i$):

\[
\epsilon \defeq \frac{1}{m - |h| - r} \left[ \ln \left( \frac{m}{|h|} \right) + \ln \left( \frac{m - |h|}{r} \right) \right. \\
\left. + \sum_{x \in Z_i} \ln(2 \cdot |x|) - \ln(\zeta(|h|) \cdot \zeta(r) \cdot \delta) \right],
\]

where $r$ is the number of errors made on $S \setminus Z_i$.

- The bound does not depend on $k$ anymore.
- We can consider exponentially more complex feature spaces without any penalty on the generalization error.
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Cross-validation</th>
<th>Bound Selection</th>
<th>Bound Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C. difficile</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Azithromycin</td>
<td>0.030 (3.3)</td>
<td><strong>0.025</strong> (2.6)</td>
<td>0.251</td>
</tr>
<tr>
<td>Ceftriaxone</td>
<td><strong>0.073</strong> (2.6)</td>
<td>0.089 (1.4)</td>
<td>0.379</td>
</tr>
<tr>
<td>Clarithromycin</td>
<td><strong>0.011</strong> (3.0)</td>
<td>0.026 (2.6)</td>
<td>0.256</td>
</tr>
<tr>
<td>Clindamycin</td>
<td>0.021 (1.4)</td>
<td><strong>0.018</strong> (1.8)</td>
<td>0.181</td>
</tr>
<tr>
<td>Moxifloxacin</td>
<td><strong>0.020</strong> (1.0)</td>
<td><strong>0.020</strong> (1.0)</td>
<td>0.174</td>
</tr>
<tr>
<td><strong>M. tuberculosis</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ethambutol</td>
<td>0.179 (1.4)</td>
<td><strong>0.172</strong> (1.1)</td>
<td>0.564</td>
</tr>
<tr>
<td>Isoniazid</td>
<td><strong>0.021</strong> (1.0)</td>
<td><strong>0.021</strong> (1.0)</td>
<td>0.327</td>
</tr>
<tr>
<td>Pyrazinamide</td>
<td><strong>0.318</strong> (3.1)</td>
<td>0.392 (1.4)</td>
<td>0.695</td>
</tr>
<tr>
<td>Rifampicin</td>
<td>0.031 (1.4)</td>
<td><strong>0.027</strong> (1.0)</td>
<td>0.358</td>
</tr>
<tr>
<td>Streptomycin</td>
<td><strong>0.050</strong> (1.0)</td>
<td><strong>0.050</strong> (1.0)</td>
<td>0.393</td>
</tr>
<tr>
<td><strong>P. aeruginosa</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amikacin</td>
<td><strong>0.175</strong> (4.9)</td>
<td>0.177 (2.7)</td>
<td>0.493</td>
</tr>
<tr>
<td>Doripenem</td>
<td><strong>0.270</strong> (1.4)</td>
<td>0.275 (1.3)</td>
<td>0.562</td>
</tr>
<tr>
<td>Levofloxacin</td>
<td><strong>0.072</strong> (1.2)</td>
<td><strong>0.072</strong> (1.0)</td>
<td>0.324</td>
</tr>
<tr>
<td>Meropenem</td>
<td><strong>0.267</strong> (1.6)</td>
<td>0.284 (1.7)</td>
<td>0.596</td>
</tr>
<tr>
<td><strong>S. pneumoniae</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Benzylpenicillin</td>
<td><strong>0.013</strong> (1.1)</td>
<td>0.011 (1.0)</td>
<td>0.142</td>
</tr>
<tr>
<td>Erythromycin</td>
<td>0.037 (2.0)</td>
<td><strong>0.035</strong> (2.0)</td>
<td>0.210</td>
</tr>
<tr>
<td>Tetracycline</td>
<td>0.031 (1.1)</td>
<td><strong>0.027</strong> (1.0)</td>
<td>0.164</td>
</tr>
</tbody>
</table>
Spurious Correlations can be Overcome

a) Ethambutol
   | Isoniazid | Pyrazinamide | Rifampicin | Streptomycin |
---|-----------|--------------|------------|-------------|
Ethambutol | 1.00      |             |            |             |
Isoniazid   | 1.00      | 1.00        |            |             |
Pyrazinamide| 1.00      | 1.00        | 1.00       |             |
Rifampicin  | 15        | 15          | 1          |             |
Streptomycin|           |             |            | 0.76        |

Round 1 (0.050)
Round 2 (0.059)
Round 3 (0.059)
Round 4 (0.061)
Round 5 (0.061)
Round 6 (0.072)

b) katG - Catalase-Peroxydase
    rpoB - RNA Polymerase β-Subunit
    rpsL - 30S Ribosomal Protein S12

Alexandre Drouin (Université Laval)