Decision-theoretic Machine Learning

Krzysztof Dembczyński and Wojciech Kotłowski

Intelligent Decision Support Systems Laboratory (IDSS)
Poznań University of Technology, Poland

Poznań University of Technology, Summer 2019

Agenda

1. Introduction to Machine Learning
2. Binary Classification
3. Bipartite Ranking
4. **Multi-Label Classification**

Outline

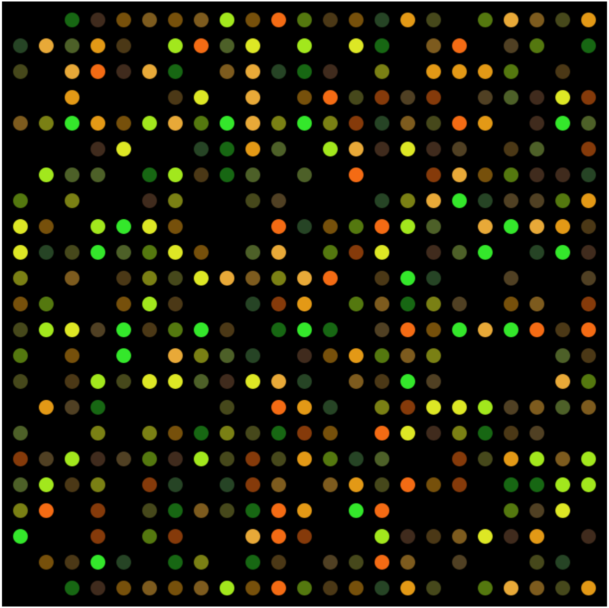
Outline

Multi-label classification

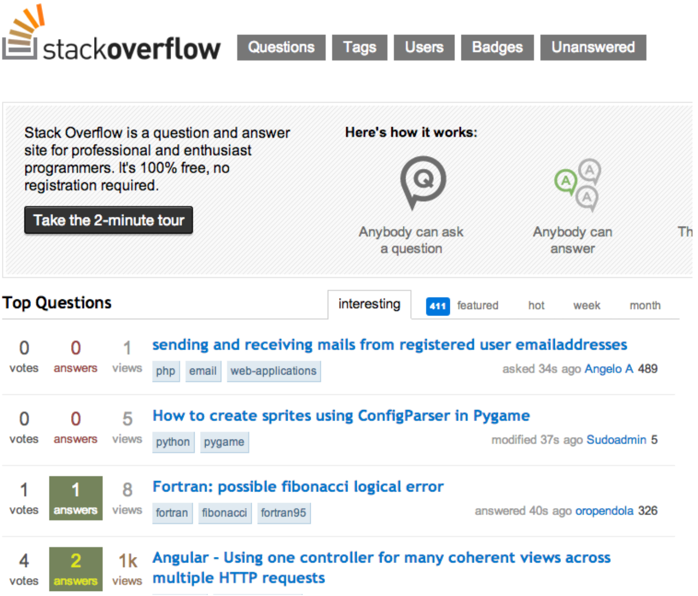
- A classification problem in which we consider **more than one** binary output variables.

Image annotation: cloud? sky? tree?

Ecology: Prediction of the presence or absence of species



Gene function prediction



Document tagging

Multi-label classification

- **Multi-label classification:** For a feature vector $\boldsymbol{x}$ predict accurately a vector of responses $\boldsymbol{y}$ using a function $\boldsymbol{h}(\boldsymbol{x})$:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_p) \xrightarrow{\boldsymbol{h}(\boldsymbol{x})} \boldsymbol{y} = (y_1, y_2, \ldots, y_m) \in \mathcal{Y} = \{0, 1\}^m$$

Multi-label classification

- Training data: $\{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$.
- **Predict** a vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$ for a given $\boldsymbol{x}$.

	x_1	x_2	y_1	y_2	$\ldots$	y_m
$\boldsymbol{x}_1$	5.0	4.5	1	1		0
$\boldsymbol{x}_2$	2.0	2.5	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
$\boldsymbol{x}_n$	3.0	3.5	0	1		1
$\boldsymbol{x}$	4.0	2.5	?	?		?

Multi-label classification

- Training data: $\{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$.
- **Predict** a vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$ for a given $\boldsymbol{x}$.

	x_1	x_2	y_1	y_2	$\ldots$	y_m
$\boldsymbol{x}_1$	5.0	4.5	1	1		0
$\boldsymbol{x}_2$	2.0	2.5	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
$\boldsymbol{x}_n$	3.0	3.5	0	1		1
$\boldsymbol{x}$	4.0	2.5	1	1		0

Multi-label classification

- **Example** $\boldsymbol{x}$ is coming from an unknown input distribution $P(\boldsymbol{x})$.
- **True outcome** $\boldsymbol{y}$ is generated from $P(\boldsymbol{y} \,|\, \boldsymbol{x})$.
- **Predicted outcome** is given by $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$.
- The (**task**) **loss** of a single prediction is $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})$.

Multi-label classification

- The overall goal is to minimize the **risk**:

$$L_\ell(\boldsymbol{h}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}(\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})))$$

- The optimal prediction function, the so-called **Bayes classifier**, is:

$$\boldsymbol{h}^*_\ell = \arg\min_{\boldsymbol{h}} L_\ell(\boldsymbol{h})$$

- The **regret** of a classifier $\boldsymbol{h}$ with respect to ℓ is defined as:

$$\mathrm{Reg}_\ell(\boldsymbol{h}) = L_\ell(\boldsymbol{h}) - L_\ell(\boldsymbol{h}^*_\ell) = L_\ell(\boldsymbol{h}) - L^*_\ell$$

Multi-label classification

- We use training examples $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_1^n$ to find either:
  - a good approximation of $\boldsymbol{h}^*$, or
  - a good estimation of $P(\boldsymbol{y} \,|\, \boldsymbol{x})$ (or a function of it).
- In the second case, we need to apply an inference procedure to approximate $\boldsymbol{h}^*$.

Main challenges

- Appropriate modeling of dependencies between labels

$$y_1, y_2, \ldots, y_m$$

- A multitude of multivariate loss functions defined over the output vector

$$\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))$$

Label interdependence

- **Marginal** and **conditional dependence**:

$$P(\boldsymbol{y}) \neq \prod_{i=1}^{m} P(y_i) \qquad P(\boldsymbol{y} \,|\, \boldsymbol{x}) \neq \prod_{i=1}^{m} P(y_i \,|\, \boldsymbol{x})$$

marginal (in)dependence $\not\rightleftarrows$ conditional (in)dependence

Label interdependence

- **Marginal dependence $\not\rightarrow$ conditional dependence**

Label interdependence

- **Marginal dependence $\not\rightarrow$ conditional dependence**
  - Consider two independent labels y_1 and y_2 generated by the same logistic model:

    $$P(y_i = 1|\boldsymbol{x}) = (1 + \exp(-\phi f(\boldsymbol{x})))^{-1},$$

    where ϕ controls the Bayes error rate.

Label interdependence

- **Marginal dependence $\not\to$ conditional dependence**
  - Consider two independent labels y_1 and y_2 generated by the same logistic model:

    $$P(y_i = 1|\boldsymbol{x}) = (1 + \exp(-\phi f(\boldsymbol{x})))^{-1},$$

    where ϕ controls the Bayes error rate.
  - Thus, the two labels are conditionally independent, having the conditional distribution:

    $$P(\boldsymbol{y}|\boldsymbol{x}) = P(y_1|\boldsymbol{x})P(y_2|\boldsymbol{x}).$$

Label interdependence

- **Marginal dependence $\not\to$ conditional dependence**
  - Consider two independent labels y_1 and y_2 generated by the same logistic model:

    $$P(y_i = 1|\boldsymbol{x}) = (1 + \exp(-\phi f(\boldsymbol{x})))^{-1},$$

    where ϕ controls the Bayes error rate.
  - Thus, the two labels are conditionally independent, having the conditional distribution:

    $$P(\boldsymbol{y}|\boldsymbol{x}) = P(y_1|\boldsymbol{x})P(y_2|\boldsymbol{x}).$$

  - Depending on the value of ϕ, however, they will be stronger or weaker marginally dependent.

Label interdependence

- **Marginal dependence $\not\rightarrow$ conditional dependence**
  - Consider two independent labels y_1 and y_2 generated by the same logistic model:

    $$P(y_i = 1|\boldsymbol{x}) = (1 + \exp(-\phi f(\boldsymbol{x})))^{-1},$$

    where ϕ controls the Bayes error rate.
  - Thus, the two labels are conditionally independent, having the conditional distribution:

    $$P(\boldsymbol{y}|\boldsymbol{x}) = P(y_1|\boldsymbol{x})P(y_2|\boldsymbol{x}).$$

  - Depending on the value of ϕ, however, they will be stronger or weaker marginally dependent.
  - For $\phi \to \infty$ (Bayes error rate tends to 0), the marginal dependence increases towards the deterministic one ($y_1 = y_2$).

Label interdependence

- **Marginal dependence $\not\leftarrow$ conditional dependence**

Label interdependence

- **Marginal dependence $\not\leftarrow$ conditional dependence**
  - Consider two labels y_1 and y_2 and a single binary feature x_1 with the joint distribution $P(x_1, y_1, y_2)$ given as:

x_1	y_1	y_2	P	x_1	y_1	y_2	P
0	0	0	0.25	1	0	0	0
0	0	1	0	1	0	1	0.25
0	1	0	0	1	1	0	0.25
0	1	1	0.25	1	1	1	0

Label interdependence

- **Marginal dependence $\not\leftarrow$ conditional dependence**
  - Consider two labels y_1 and y_2 and a single binary feature x_1 with the joint distribution $P(x_1, y_1, y_2)$ given as:

x_1	y_1	y_2	P	x_1	y_1	y_2	P
0	0	0	0.25	1	0	0	0
0	0	1	0	1	0	1	0.25
0	1	0	0	1	1	0	0.25
0	1	1	0.25	1	1	1	0

  - The labels are conditionally dependent, since:

$$P(y_1 = 0|x_1 = 1)P(y_2 = 0|x_1 = 1) = 0.5 \times 0.5 = 0.25,$$

but the joint probability is

$$P(y_1 = 0, y_2 = 0|x_1 = 1) = 0.$$

In fact $y_1 = y_2$ for $x_1 = 0$ and $y_2 = 1 - y_1$ for $x_1 = 1$.

Label interdependence

- **Marginal dependence $\not\Leftarrow$ conditional dependence**
  - Consider two labels y_1 and y_2 and a single binary feature x_1 with the joint distribution $P(x_1, y_1, y_2)$ given as:

x_1	y_1	y_2	P	x_1	y_1	y_2	P
0	0	0	0.25	1	0	0	0
0	0	1	0	1	0	1	0.25
0	1	0	0	1	1	0	0.25
0	1	1	0.25	1	1	1	0

  - The labels are conditionally dependent, since:

$$P(y_1 = 0|x_1 = 1)P(y_2 = 0|x_1 = 1) = 0.5 \times 0.5 = 0.25,$$

but the joint probability is

$$P(y_1 = 0, y_2 = 0|x_1 = 1) = 0.$$

In fact $y_1 = y_2$ for $x_1 = 0$ and $y_2 = 1 - y_1$ for $x_1 = 1$.
  - However, the labels are marginally independent, since

$$P(y_1) = P(y_2) = 0.5, \text{ and } P(y_1, y_2) = P(y_1)P(y_2).$$

Label interdependence

- **Deterministic dependencies**:
  - Consider labels y_1 and y_2 with the following conditional distribution for a given $\boldsymbol{x}$:

$$
\begin{aligned}
P(y_1 = 1, y_2 = 1 \,|\, \boldsymbol{x}) &= 1, \\
P(y_1 = 0, y_2 = 1 \,|\, \boldsymbol{x}) &= 0, \\
P(y_1 = 1, y_2 = 0 \,|\, \boldsymbol{x}) &= 0, \\
P(y_1 = 0, y_2 = 0 \,|\, \boldsymbol{x}) &= 0.
\end{aligned}
$$

  - Are y_1 and y_2 conditionally dependent?

- **Deterministic dependencies**:
  - Consider labels y_1 and y_2 with the following conditional distribution for a given $\boldsymbol{x}$:

$$
\begin{aligned}
P(y_1 = 1, y_2 = 1 \,|\, \boldsymbol{x}) &= 1, \\
P(y_1 = 0, y_2 = 1 \,|\, \boldsymbol{x}) &= 0, \\
P(y_1 = 1, y_2 = 0 \,|\, \boldsymbol{x}) &= 0, \\
P(y_1 = 0, y_2 = 0 \,|\, \boldsymbol{x}) &= 0.
\end{aligned}
$$

  - Are y_1 and y_2 conditionally dependent?
  - **No**, since it holds:

$$P(y_1, y_2 \,|\, \boldsymbol{x}) = P(y_1 \,|\, \boldsymbol{x}) P(y_2 \,|\, \boldsymbol{x})$$

Label interdependence

- **Model similarities**:
  - Similarities in the structural parts $g_i(\boldsymbol{x})$ of the models:

$$f_i(\boldsymbol{x}) = g_i(\boldsymbol{x}) + \epsilon_i, \text{ for } i = 1, \ldots, m$$

- **Structure** imposed (domain knowledge) on targets
  - Chains,
  - Hierarchies,
  - General graphs,
  - ...

Label interdependence

- **Interdependence** vs. **hypothesis** and **feature space**:
  - Regularization constraints the hypothesis space.
  - Modeling dependencies may increase the expressiveness of the model.
  - Using a more complex model on individual labels may also help.
  - Comparison of models is difficult in general, as they differ in many respects (e.g., complexity)!

Multivariate loss functions

- **Decomposable** and **non-decomposable** losses over **examples**

$$L = \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, \boldsymbol{h}(\boldsymbol{x}_i)) \quad L \neq \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, \boldsymbol{h}(\boldsymbol{x}_i))$$

- **Decomposable** and **non-decomposable** losses over **labels**

$$\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \sum_{i=1}^{m} \ell(y_i, h_i(\boldsymbol{x})) \quad \ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) \neq \sum_{i=1}^{m} \ell(y_i, h_i(\boldsymbol{x}))$$

- Different formulations of loss functions possible:
  - Set-based losses.
  - Ranking-based losses.

Multi-label loss functions

- **Subset 0/1 loss**: $\ell_{0/1}(\boldsymbol{y}, \boldsymbol{h}) = [\![\boldsymbol{y} \neq \boldsymbol{h}]\!]$
- **Hamming loss**: $\ell_H(\boldsymbol{y}, \boldsymbol{h}) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq h_i]\!]$
- **F-measure-based loss**: $\ell_F(\boldsymbol{y}, \boldsymbol{h}) = 1 - \frac{2 \sum_{i=1}^{m} y_i h_i}{\sum_{i=1}^{m} y_i + \sum_{i=1}^{m} h_i}$
- **Rank loss**: $\ell_{\mathrm{rnk}}(\boldsymbol{y}, \boldsymbol{h}) = w(\boldsymbol{y}) \sum_{y_i > y_j} \left([\![h_i < h_j]\!] + \frac{1}{2} [\![h_i = h_j]\!] \right)$
- ...

Relations between losses

- The set-based loss function $\ell(\boldsymbol{y}, \boldsymbol{h})$ should fulfill some basic conditions:
  - $\ell(\boldsymbol{y}, \boldsymbol{h}) = 0$ if and only if $\boldsymbol{y} = \boldsymbol{h}$.
  - $\ell(\boldsymbol{y}, \boldsymbol{h})$ is maximal when $y_i \neq h_i$ for every $i = 1, \ldots, m$.
  - Should be monotonically non-decreasing with respect to the number of $y_i \neq h_i$.

Relations between losses

- The set-based loss function $\ell(\boldsymbol{y}, \boldsymbol{h})$ should fulfill some basic conditions:
  - $\ell(\boldsymbol{y}, \boldsymbol{h}) = 0$ if and only if $\boldsymbol{y} = \boldsymbol{h}$.
  - $\ell(\boldsymbol{y}, \boldsymbol{h})$ is maximal when $y_i \neq h_i$ for every $i = 1, \ldots, m$.
  - Should be monotonically non-decreasing with respect to the number of $y_i \neq h_i$.
- **In case of deterministic data (no-noise)**: the optimal prediction should have the same form for all loss functions and the risk for this prediction should be 0.

Relations between losses

- The set-based loss function $\ell(\boldsymbol{y}, \boldsymbol{h})$ should fulfill some basic conditions:
  - $\ell(\boldsymbol{y}, \boldsymbol{h}) = 0$ if and only if $\boldsymbol{y} = \boldsymbol{h}$.
  - $\ell(\boldsymbol{y}, \boldsymbol{h})$ is maximal when $y_i \neq h_i$ for every $i = 1, \ldots, m$.
  - Should be monotonically non-decreasing with respect to the number of $y_i \neq h_i$.
- **In case of deterministic data (no-noise)**: the optimal prediction should have the same form for all loss functions and the risk for this prediction should be 0.
- **In case of non-deterministic data (noise)**: the optimal prediction and its risk can be different for different losses.

Learning and inference with multi-label losses

- The loss functions, like Hamming loss or subset $0/1$ loss, often referred to as **task losses**, are usually neither convex nor differentiable.

- The loss functions, like Hamming loss or subset $0/1$ loss, often referred to as **task losses**, are usually neither convex nor differentiable.
- Therefore learning is a hard optimization problem.

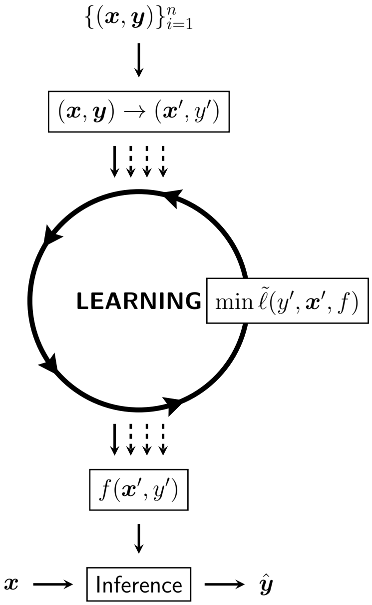
Learning and inference with multi-label losses

- The loss functions, like Hamming loss or subset $0/1$ loss, often referred to as **task losses**, are usually neither convex nor differentiable.
- Therefore learning is a hard optimization problem.
- Two approaches try to make this task easier
  - Reduction.
  - Surrogate loss minimization.

Learning and inference with multi-label losses

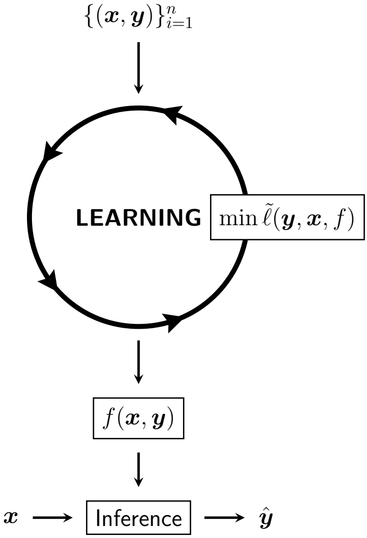
- The loss functions, like Hamming loss or subset $0/1$ loss, often referred to as **task losses**, are usually neither convex nor differentiable.
- Therefore learning is a hard optimization problem.
- Two approaches try to make this task easier
  - Reduction.
  - Surrogate loss minimization.
- Two phases in solving multi-label problems:
  - Learning: Estimate parameters of a scoring function $f(\boldsymbol{x}, \boldsymbol{y})$.
  - Inference: Use the scoring function $f(\boldsymbol{x}, \boldsymbol{y})$ to classify new instances by finding the best $\boldsymbol{y}$ for a given $\boldsymbol{x}$.

Reduction



- **Reduce** the original problem into simple problems, for which efficient algorithmic solutions are available.
- Reduction to one or a sequence of problems.
- Plug-in rule classifiers.

Structured loss minimization



- Replace the task loss by a **surrogate loss** that is easier to cope with.
- Surrogate loss is typically a differentiable approximation of the task loss or a convex upper bound of it.

Statistical consistency

- Analysis of algorithms in terms of their infinite sample performance.[1]
- We say that a proxy loss $\tilde{\ell}$ is **consistent** (**calibrated**) with the task loss ℓ when the following holds:

$$\mathrm{Reg}_{\tilde{\ell}}(\boldsymbol{h}) \to 0 \Rightarrow \mathrm{Reg}_{\ell}(\boldsymbol{h}) \to 0\,.$$

[1] A. Tewari and P.L. Bartlett. On the consistency of multiclass classification methods. *JMLR*, 8:1007–1025, 2007

D. McAllester and J. Keshet. Generalization bounds and consistency for latent structural probit and ramp loss. In *NIPS*, pages 2205–2212, 2011

W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013

Statistical consistency

- Analysis of algorithms in terms of their infinite sample performance.[1]
- We say that a proxy loss $\tilde{\ell}$ is **consistent** (**calibrated**) with the task loss ℓ when the following holds:

$$\mathrm{Reg}_{\tilde{\ell}}(\boldsymbol{h}) \to 0 \Rightarrow \mathrm{Reg}_{\ell}(\boldsymbol{h}) \to 0\,.$$

- The definition concerns both surrogate loss minimization and reduction:
  - Surrogate loss minimization: $\tilde{\ell} =$ surrogate loss.
  - Reduction: $\tilde{\ell} =$ loss used in the reduced problem.

[1] A. Tewari and P.L. Bartlett. On the consistency of multiclass classification methods. *JMLR*, 8:1007–1025, 2007

D. McAllester and J. Keshet. Generalization bounds and consistency for latent structural probit and ramp loss. In *NIPS*, pages 2205–2212, 2011

W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013

Outline

Basic reductions: Binary relevance

- **Binary relevance**: Decomposes the problem to m binary classification problems:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = y_i), \quad i = 1, \ldots, m$$

	X_1	X_2	Y_1	Y_2	$\ldots$	Y_m
$\boldsymbol{x}_1$	5.0	4.5	1	1		0
$\boldsymbol{x}_2$	2.0	2.5	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
$\boldsymbol{x}_n$	3.0	3.5	0	1		1

Basic reductions: Binary relevance

- **Binary relevance**: Decomposes the problem to m binary classification problems:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = y_i), \quad i = 1, \ldots, m$$

	X_1	X_2	Y_1	Y_2	$\ldots$	Y_m
$\boldsymbol{x}_1$	5.0	4.5	1	1		0
$\boldsymbol{x}_2$	2.0	2.5	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
$\boldsymbol{x}_n$	3.0	3.5	0	1		1

- Seems to be very simplistic.
- Ignores any dependencies.
- Is it good for any loss function?

Basic reductions: Label powerset

- **Label powerset**: Treats each label combination as a new meta-class in multi-class classification:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y}))$$

	X_1	X_2	Y_1	Y_2	$\dots$	Y_m
$\boldsymbol{x}_1$	5.0	4.5	1	1		0
$\boldsymbol{x}_2$	2.0	2.5	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
$\boldsymbol{x}_n$	3.0	3.5	0	1		1

Basic reductions: Label powerset

- **Label powerset**: Treats each label combination as a new meta-class in multi-class classification:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y}))$$

	X_1	X_2	Y_1	Y_2	$\ldots$	Y_m
$\boldsymbol{x}_1$	5.0	4.5	1	1		0
$\boldsymbol{x}_2$	2.0	2.5	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
$\boldsymbol{x}_n$	3.0	3.5	0	1		1

- Any multi-class classification algorithm can be used, but the number of classes is huge.
- Takes other labels into account, but ignores internal structure of classes (label vectors).

What about task losses minimized by BR and LP?

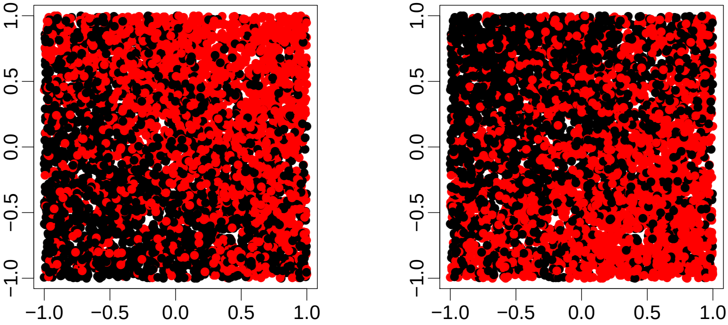
Synthetic data

- Two independent models:

$$f_1(\boldsymbol{x}) = \frac{1}{2}x_1 + \frac{1}{2}x_2, \quad f_2(\boldsymbol{x}) = \frac{1}{2}x_1 - \frac{1}{2}x_2$$

- Logistic model to get labels:

$$P(y_i = 1) = \frac{1}{1 + \exp(-2f_i)}$$



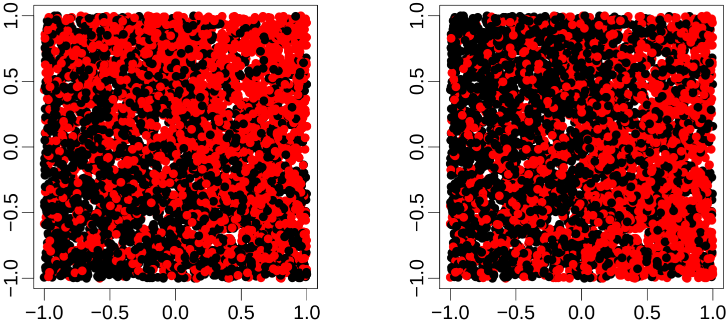
Synthetic data

- Two dependent models:

$$f_1(\boldsymbol{x}) = \frac{1}{2}x_1 + \frac{1}{2}x_2 \quad f_2(y_1, \boldsymbol{x}) = y_1 + \frac{1}{2}x_1 - \frac{1}{2}x_2 - \frac{2}{3}$$

- Logistic model to get labels:

$$P(y_i = 1) = \frac{1}{1 + \exp(-2f_i)}$$



Results for two performance measures

- Hamming loss: $\ell_H(\boldsymbol{y}, \boldsymbol{h}) = \frac{1}{m}\sum_{i=1}^{m} [\![y_i \neq h_i]\!]$,
- Subset 0/1 loss: $\ell_{0/1}(\boldsymbol{y}, \boldsymbol{h}) = [\![\boldsymbol{y} \neq \boldsymbol{h}]\!]$.

Conditional independence		
classifier	Hamming loss	subset 0/1 loss
BR LR		
LP LR		

Conditional dependence		
classifier	Hamming loss	subset 0/1 loss
BR LR		
LP LR		

Results for two performance measures

- Hamming loss: $\ell_H(\boldsymbol{y}, \boldsymbol{h}) = \frac{1}{m}\sum_{i=1}^{m} [\![y_i \neq h_i]\!]$,
- Subset 0/1 loss: $\ell_{0/1}(\boldsymbol{y}, \boldsymbol{h}) = [\![\boldsymbol{y} \neq \boldsymbol{h}]\!]$.

Conditional independence		
classifier	Hamming loss	subset 0/1 loss
BR LR	0.4232	0.6723
LP LR	0.4232	0.6725

Conditional dependence		
classifier	Hamming loss	subset 0/1 loss
BR LR		
LP LR		

Results for two performance measures

- Hamming loss: $\ell_H(\boldsymbol{y}, \boldsymbol{h}) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq h_i]\!]$,
- Subset 0/1 loss: $\ell_{0/1}(\boldsymbol{y}, \boldsymbol{h}) = [\![\boldsymbol{y} \neq \boldsymbol{h}]\!]$.

Conditional independence		
classifier	Hamming loss	subset 0/1 loss
BR LR	0.4232	0.6723
LP LR	0.4232	0.6725

Conditional dependence		
classifier	Hamming loss	subset 0/1 loss
BR LR	0.3470	0.5499
LP LR	0.3610	0.5146

Linear + XOR synthetic data

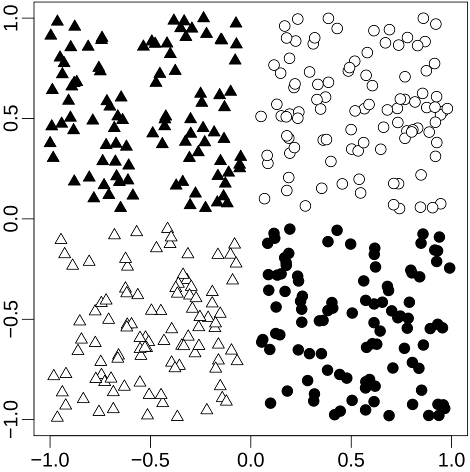


Figure: Problem with two targets: shapes (△ vs. ∘) and colors (□ vs. ■).

Linear + XOR synthetic data

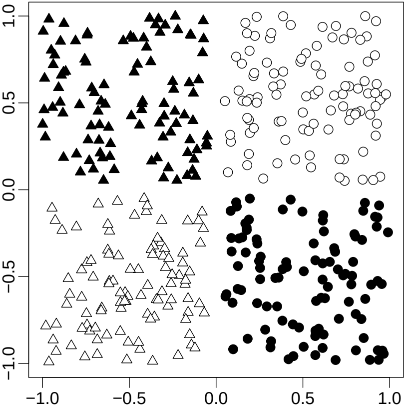
classifier	Hamming loss	subset 0/1 loss
BR LR	$0.2399(\pm.0097)$	$0.4751(\pm.0196)$
LP LR	$0.0143(\pm.0020)$	$0.0195(\pm.0011)$
Bayes Optimal	0	0

Linear + XOR synthetic data

CLASSIFIER	HAMMING LOSS	SUBSET 0/1 LOSS
BR LR	$0.2399(\pm.0097)$	$0.4751(\pm.0196)$
LP LR	$0.0143(\pm.0020)$	$0.0195(\pm.0011)$
BR MLRules	**$0.0011(\pm.0002)$**	**$0.0020(\pm.0003)$**
BAYES OPTIMAL	0	0

Linear + XOR synthetic data

- BR LR uses two linear classifiers: cannot handle the label color (□ vs. ■) – the XOR problem.
- LP LR uses four linear classifiers to solve 4-class problem (△, ▲, ○, •): extends the hypothesis space.
- BR MLRules uses two non-linear classifiers (based on decision rules): XOR problem is not a problem.
- There is no noise in the data.
- Easy to perform unfair comparison.



Multi-label loss functions

- The conditional risk in multi-label classification of $\boldsymbol{h}$ at $\boldsymbol{x}$:

$$L_\ell(\boldsymbol{h} \,|\, \boldsymbol{x}) = \mathbb{E}_{\boldsymbol{y}}\left[\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))\right] = \sum_{\boldsymbol{y} \in \mathcal{Y}} P(\boldsymbol{y} \,|\, \boldsymbol{x}) \ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))$$

- The risk-minimizing classifier for a given $\boldsymbol{x}$:

$$\boldsymbol{h}^*(\boldsymbol{x}) = \arg\min_{\boldsymbol{h}} L_\ell(\boldsymbol{h} \,|\, \boldsymbol{x})$$

- Let us start with Hamming loss and subset $0/1$ loss ...[2]

[2] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. On loss minimization and label dependence in multi-label classification. *Machine Learning*, 88:5–45, 2012

Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is

Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is the **marginal mode**:

$$h_i^*(\boldsymbol{x}) = \arg\max_{y_i \in \{0,1\}} P(y_i \,|\, \boldsymbol{x})\,, \quad i = 1, \ldots, m,$$

Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is the **marginal mode**:

$$h_i^*(\boldsymbol{x}) = \arg\max_{y_i \in \{0,1\}} P(y_i \,|\, \boldsymbol{x}), \quad i = 1, \ldots, m,$$

while for the subset 0/1 loss is

Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is the **marginal mode**:

$$h_i^*(\boldsymbol{x}) = \arg\max_{y_i \in \{0,1\}} P(y_i \,|\, \boldsymbol{x})\,, \quad i = 1, \ldots, m,$$

while for the subset 0/1 loss is the **joint mode**:

$$\boldsymbol{h}^*(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} P(\boldsymbol{y} \,|\, \boldsymbol{x})\,.$$

Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is the **marginal mode**:

$$h_i^*(\boldsymbol{x}) = \arg\max_{y_i \in \{0,1\}} P(y_i \,|\, \boldsymbol{x}) \,, \quad i = 1, \ldots, m,$$

while for the subset 0/1 loss is the **joint mode**:

$$\boldsymbol{h}^*(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} P(\boldsymbol{y} \,|\, \boldsymbol{x}) \,.$$

- Marginal mode vs. joint mode.

$\boldsymbol{y}$	$P(\boldsymbol{y})$
0 0 0 0	0.30
0 1 1 1	0.17
1 0 1 1	0.18
1 1 0 1	0.17
1 1 1 0	0.18

Marginal mode: 1 1 1 1
Joint mode: 0 0 0 0

Equivalence of risk minimizers and mutual risk bounds

- The risk minimizers for ℓ_H and $\ell_{0/1}$ are **equivalent**,

$$\boldsymbol{h}_H^*(\boldsymbol{x}) = \boldsymbol{h}_{0/1}^*(\boldsymbol{x})\,,$$

under specific conditions, for example, when:

Equivalence of risk minimizers and mutual risk bounds

- The risk minimizers for ℓ_H and $\ell_{0/1}$ are **equivalent**,

$$\boldsymbol{h}_H^*(\boldsymbol{x}) = \boldsymbol{h}_{0/1}^*(\boldsymbol{x}) \,,$$

  under specific conditions, for example, when:
  - Targets $y_1, \ldots, y_m$ are conditionally independent, i.e,

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{m} P(y_i|\boldsymbol{x}) \,.$$

Equivalence of risk minimizers and mutual risk bounds

- The risk minimizers for ℓ_H and $\ell_{0/1}$ are **equivalent**,

$$\boldsymbol{h}_H^*(\boldsymbol{x}) = \boldsymbol{h}_{0/1}^*(\boldsymbol{x}),$$

under specific conditions, for example, when:

  - Targets $y_1, \ldots, y_m$ are conditionally independent, i.e,

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{m} P(y_i|\boldsymbol{x}).$$

  - The probability of the joint mode satisfies

$$P(\boldsymbol{h}_{0/1}^*(\boldsymbol{x})|\boldsymbol{x}) > 0.5.$$

Equivalence of risk minimizers and mutual risk bounds

- The risk minimizers for ℓ_H and $\ell_{0/1}$ are **equivalent**,

$$\boldsymbol{h}^*_H(\boldsymbol{x}) = \boldsymbol{h}^*_{0/1}(\boldsymbol{x})\,,$$

  under specific conditions, for example, when:
  - Targets $y_1, \ldots, y_m$ are conditionally independent, i.e,

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{m} P(y_i|\boldsymbol{x})\,.$$

  - The probability of the joint mode satisfies

$$P(\boldsymbol{h}^*_{0/1}(\boldsymbol{x})|\boldsymbol{x}) > 0.5\,.$$

- The following bounds hold for any $P(\boldsymbol{y}\,|\,\boldsymbol{x})$ and $\boldsymbol{h}$:

$$\frac{1}{m} L_{0/1}(\boldsymbol{h}\,|\,\boldsymbol{x}) \leq L_H(\boldsymbol{h}\,|\,\boldsymbol{x}) \leq L_{0/1}(\boldsymbol{h}\,|\,\boldsymbol{x})$$

Regret analysis

- The previous results may suggest that one of the loss functions can be used as a proxy (surrogate) for the other:
  - For some situations both risk minimizers coincide.
  - One can provide mutual bounds for both loss functions.

Regret analysis

- The previous results may suggest that one of the loss functions can be used as a proxy (surrogate) for the other:
  - For some situations both risk minimizers coincide.
  - One can provide mutual bounds for both loss functions.
- **However, the regret analysis of the worst case shows that minimization of the subset 0/1 loss may result in a large error for the Hamming loss and vice versa.**

Regret analysis

- The **regret** of a classifier with respect to ℓ is defined as:

$$\mathrm{Reg}_\ell(\boldsymbol{h}) = L_\ell(\boldsymbol{h}) - L_\ell(\boldsymbol{h}^*_\ell)\,,$$

  where $\boldsymbol{h}^*_\ell$ is the Bayes classifier for a given loss ℓ.
- Regret measures how worse is $\boldsymbol{h}$ by comparison with the optimal classifier for a given loss.
- To simplify the analysis we will consider the conditional regret:

$$\mathrm{Reg}_\ell(\boldsymbol{h} \,|\, \boldsymbol{x}) = L_\ell(\boldsymbol{h} \,|\, \boldsymbol{x}) - L_\ell(\boldsymbol{h}^*_\ell \,|\, \boldsymbol{x})\,.$$

- We will analyze the regret between:
  - the Bayes classifier for Hamming loss $\boldsymbol{h}^*_H$
  - the Bayes classifier for subset 0/1 loss $\boldsymbol{h}^*_{0/1}$

  with respect to both functions.
- It is a bit an unusual analysis.

Regret analysis

- The following **upper bound** holds:

$$\mathrm{Reg}_{0/1}(\boldsymbol{h}^*_H \,|\, \boldsymbol{x}) = L_{0/1}(\boldsymbol{h}^*_H \,|\, \boldsymbol{x}) - L_{0/1}(\boldsymbol{h}^*_{0/1} \,|\, \boldsymbol{x}) < 0.5$$

- Moreover, this **bound is tight**.
- **Example**:

$\boldsymbol{y}$	$P(\boldsymbol{y})$
0 0 0 0	0.02
0 0 1 1	0.49
1 1 0 0	0.49

Marginal mode: 0 0 0 0
Joint mode: 0 0 1 1 or 1 1 0 0

Regret analysis

- The following **upper bound** holds $m > 3$:

$$\mathrm{Reg}_H(\boldsymbol{h}^*_{0/1} \,|\, \boldsymbol{x}) = L_H(\boldsymbol{h}^*_{0/1} \,|\, \boldsymbol{x}) - L_H(\boldsymbol{h}^*_H \,|\, \boldsymbol{x}) < \frac{m-2}{m+2}$$

- Moreover, this **bound is tight**.
- **Example**:

$\boldsymbol{y}$	$P(\boldsymbol{y})$
0 0 0 0	0.170
0 1 1 1	0.166
1 0 1 1	0.166
1 1 0 1	0.166
1 1 1 0	0.166
1 1 1 1	0.166

Marginal mode: 1 1 1 1
Joint mode: 0 0 0 0

Relations between losses

- The risk minimizers of Hamming and subset $0/1$ loss are different: marginal mode vs. joint mode.

Relations between losses

- The risk minimizers of Hamming and subset $0/1$ loss are different: marginal mode vs. joint mode.
- Under specific conditions, like label independence or high probability of the joint mode (> 0.5), these two risk minimizers are equivalent.

Relations between losses

- The risk minimizers of Hamming and subset $0/1$ loss are different: marginal mode vs. joint mode.
- Under specific conditions, like label independence or high probability of the joint mode (> 0.5), these two risk minimizers are equivalent.
- The risks of these loss functions are mutually upper bounded.

Relations between losses

- The risk minimizers of Hamming and subset $0/1$ loss are different: marginal mode vs. joint mode.
- Under specific conditions, like label independence or high probability of the joint mode (> 0.5), these two risk minimizers are equivalent.
- The risks of these loss functions are mutually upper bounded.
- Minimization of the subset $0/1$ loss may cause a high regret for the Hamming loss and vice versa.

BR vs. LP

BR vs. LP

- Binary relevance (BR)

BR vs. LP

- Binary relevance (BR)
  - BR is **consistent** for Hamming loss **without** any additional assumptions on **label (in)dependence**.

BR vs. LP

- Binary relevance (BR)
  - BR is **consistent** for Hamming loss **without** any additional assumptions on **label (in)dependence**.
  - If this would not be true, then **we could not optimally solve binary classification problems!!!**

BR vs. LP

- Binary relevance (BR)
  - BR is **consistent** for Hamming loss **without** any additional assumptions on **label (in)dependence**.
  - If this would not be true, then **we could not optimally solve binary classification problems!!!**
  - For other losses, one should take **additional assumptions**:
    - For subset $0/1$ loss: label independence, high probability of the joint mode (> 0.5), ...

BR vs. LP

- Binary relevance (BR)
  - BR is **consistent** for Hamming loss **without** any additional assumptions on **label (in)dependence**.
  - If this would not be true, then **we could not optimally solve binary classification problems!!!**
  - For other losses, one should take **additional assumptions**:
    - For subset $0/1$ loss: label independence, high probability of the joint mode (> 0.5), ...
  - Learning and inference is **linear** in m (however, faster algorithms exist).

- Label powerset (LP)

BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset 0/1 loss.

BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset $0/1$ loss.
  - In its basic formulation it is **not consistent** for Hamming loss.

BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset $0/1$ loss.
  - In its basic formulation it is **not consistent** for Hamming loss.
  - However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given x: inference for **any loss** would be then possible.

BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset $0/1$ loss.
  - In its basic formulation it is **not consistent** for Hamming loss.
  - However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given x: inference for **any loss** would be then possible.
  - Similarly, by reducing to cost-sensitive multi-class classification LP can be used with **almost any loss function**.

BR vs. LP

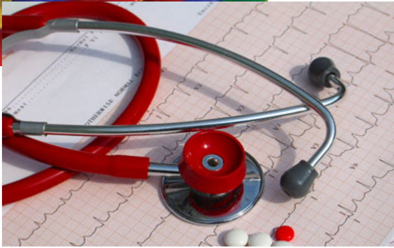
- Label powerset (LP)
  - LP is **consistent** for the subset $0/1$ loss.
  - In its basic formulation it is **not consistent** for Hamming loss.
  - However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given x: inference for **any loss** would be then possible.
  - Similarly, by reducing to cost-sensitive multi-class classification LP can be used with **almost any loss function**.
  - LP may gain from the implicit expansion of the **feature** or **hypothesis space**.

BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset $0/1$ loss.
  - In its basic formulation it is **not consistent** for Hamming loss.
  - However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given x: inference for **any loss** would be then possible.
  - Similarly, by reducing to cost-sensitive multi-class classification LP can be used with **almost any loss function**.
  - LP may gain from the implicit expansion of the **feature** or **hypothesis space**.
  - Unfortunately, learning and inference is basically **exponential** in m (however, this complexity is constrained by the number of training examples).

Relations between losses

- Both are commonly used.
- Hamming loss:
  - Not too many labels.
  - Well-balanced labels.
  - **Application**: Gene function prediction.
- Subset 0/1 loss:
  - Very restrictive.
  - Small number of labels.
  - Low noise problems.
  - **Application**: Prediction of diseases of a patient.



Outline

Beyond LP

- **Classical multi-class classification algorithms**:
  - k-nearest neighbors,
  - Decision trees,
  - Logistic regression,
  - Multi-class SVMs,
  - ...
- **Reduction algorithms**:
  - 1 vs All,
  - 1 vs 1 and Weighted All-Pairs (WAP),
  - Directed acyclic graphs (DAG),
  - ECOC, PECOC, SECOC,
  - Filter Trees,
  - Conditional Probability Trees,
  - ...
- **Can we adapt these algorithms to multi-label classification and different task losses in a more direct way?**

Beyond LP

- Naive reduction to 1 vs. All:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y}))$$

Beyond LP

- Naive reduction to 1 vs. All:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y}))$$

- Reduction of **multi-class classification** to **binary classification**:

$$(\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y})) \longrightarrow \{(\boldsymbol{x}, y, 1)\} \cup \{(\boldsymbol{x}, y', 0) : \forall y' \neq y\}$$

Beyond LP

- Naive reduction to 1 vs. All:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y}))$$

- Reduction of **multi-class classification** to **binary classification**:

$$(\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y})) \longrightarrow \{(\boldsymbol{x}, y, 1)\} \cup \{(\boldsymbol{x}, y', 0) : \forall y' \neq y\}$$

- But we can reduce directly **multi-label classification** to **binary classification**:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow \{(\boldsymbol{x}, \boldsymbol{y}, 1)\} \cup \{(\boldsymbol{x}, \boldsymbol{y}', 0) : \forall \boldsymbol{y}' \neq \boldsymbol{y}\}$$

Beyond LP

- Naive reduction to 1 vs. All:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y}))$$

- Reduction of **multi-class classification** to **binary classification**:

$$(\boldsymbol{x}, y = \mathrm{metaclass}(\boldsymbol{y})) \longrightarrow \{(\boldsymbol{x}, y, 1)\} \cup \{(\boldsymbol{x}, y', 0) : \forall y' \neq y\}$$

- But we can reduce directly **multi-label classification** to **binary classification**:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow \{(\boldsymbol{x}, \boldsymbol{y}, 1)\} \cup \{(\boldsymbol{x}, \boldsymbol{y}', 0) : \forall \boldsymbol{y}' \neq \boldsymbol{y}\}$$

- **We can exploit now the internal structure of label vectors!!!**

Internal structure of classes

- The model can be given by a **scoring function** $f(\boldsymbol{x}, \boldsymbol{y})$.

Internal structure of classes

- The model can be given by a **scoring function** $f(\boldsymbol{x}, \boldsymbol{y})$.
- Different forms of $f(\boldsymbol{x}, \boldsymbol{y})$ are possible, for example:

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i)$$

Internal structure of classes

- The model can be given by a **scoring function** $f(\boldsymbol{x}, \boldsymbol{y})$.
- Different forms of $f(\boldsymbol{x}, \boldsymbol{y})$ are possible, for example:

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l) \,,$$

  where the second term models pairwise interactions.

Internal structure of classes

- The model can be given by a **scoring function** $f(\boldsymbol{x}, \boldsymbol{y})$.
- Different forms of $f(\boldsymbol{x}, \boldsymbol{y})$ are possible, for example:

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l)\,,$$

  where the second term models pairwise interactions.
- Prediction is given by:

$$\boldsymbol{h}(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} f(\boldsymbol{x}, \boldsymbol{y})$$

Internal structure of classes

- Generalization of logistic regression and SVMs for $f(\boldsymbol{x}, \boldsymbol{y})$:
  - Conditional random fields (CRFs),[3]
  - Structured support vector machines (SSVMs).[4]

[3] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001

[4] Y. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, 2005

CRFs and SSVMs

- CRFs use **logistic loss** as a surrogate:

$$\tilde{\ell}_{\log}(\boldsymbol{y}, \boldsymbol{x}, f) = -\log P(\boldsymbol{y}|\boldsymbol{x}) = \log\left(\sum_{\boldsymbol{y}\in\mathcal{Y}} \exp(f(\boldsymbol{x}, \boldsymbol{y}))\right) - f(\boldsymbol{x}, \boldsymbol{y})\,.$$

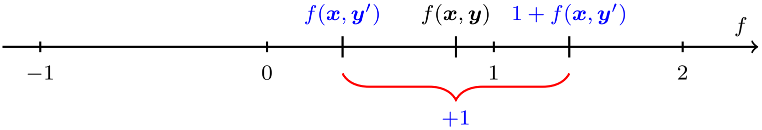
CRFs and SSVMs

- CRFs use **logistic loss** as a surrogate:

$$\tilde{\ell}_{\log}(\boldsymbol{y}, \boldsymbol{x}, f) = -\log P(\boldsymbol{y}|\boldsymbol{x}) = \log\left(\sum_{\boldsymbol{y}\in\mathcal{Y}} \exp(f(\boldsymbol{x}, \boldsymbol{y}))\right) - f(\boldsymbol{x}, \boldsymbol{y})\,.$$

- SSVMs minimize the **structured hinge loss**:

$$\tilde{\ell}_h(\boldsymbol{y}, \boldsymbol{x}, f) = \max_{\boldsymbol{y}'\in\mathcal{Y}}\{[\![\boldsymbol{y}' \neq \boldsymbol{y}]\!] + f(\boldsymbol{x}, \boldsymbol{y}')\} - f(\boldsymbol{x}, \boldsymbol{y})\,.$$



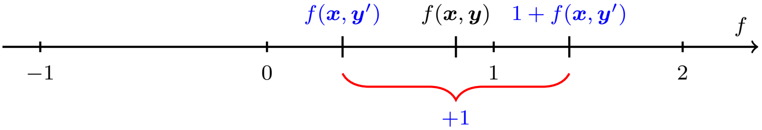
CRFs and SSVMs

- CRFs use **logistic loss** as a surrogate:

$$\tilde{\ell}_{\log}(\boldsymbol{y}, \boldsymbol{x}, f) = -\log P(\boldsymbol{y}|\boldsymbol{x}) = \log \left(\sum_{\boldsymbol{y} \in \mathcal{Y}} \exp(f(\boldsymbol{x}, \boldsymbol{y})) \right) - f(\boldsymbol{x}, \boldsymbol{y}) \, .$$

- SSVMs minimize the **structured hinge loss**:

$$\tilde{\ell}_{h}(\boldsymbol{y}, \boldsymbol{x}, f) = \max_{\boldsymbol{y}' \in \mathcal{Y}} \{ [\![\boldsymbol{y}' \neq \boldsymbol{y}]\!] + f(\boldsymbol{x}, \boldsymbol{y}') \} - f(\boldsymbol{x}, \boldsymbol{y}) \, .$$



- SSVMs and CRFs are quite similar to each other:
  - max vs. soft-max
  - margin vs. no-margin

CRFs and SSVMs

- Follow the general LP strategy, but can exploit the internal structure of classes within scoring function $f(\boldsymbol{x}, \boldsymbol{y})$.
- Convex optimization problem, but its hardness depends on the structure of $f(\boldsymbol{x}, \boldsymbol{y})$.
- Similarly, the inference (also known as decoding problem) is hard in the general case.
- For sequence and tree structures, the problem can be solved in polynomial time.

CRFs and SSVMs for different task losses

- In SSVMs, task loss $\ell(\boldsymbol{y}, \boldsymbol{y}')$ can be used for **margin rescaling**:

$$\tilde{\ell}_h(\boldsymbol{y}, \boldsymbol{x}, f) = \max_{\boldsymbol{y}' \in \mathcal{Y}} \{\ell(\boldsymbol{y}, \boldsymbol{y}') + f(\boldsymbol{x}, \boldsymbol{y}')\} - f(\boldsymbol{x}, \boldsymbol{y})\,.$$

[5] W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013

A. Tewari and P.L. Bartlett. On the consistency of multiclass classification methods. *JMLR*, 8:1007–1025, 2007

D. McAllester. *Generalization Bounds and Consistency for Structured Labeling in Predicting Structured Data*. MIT Press, 2007

CRFs and SSVMs for different task losses

- In SSVMs, task loss $\ell(\boldsymbol{y}, \boldsymbol{y}')$ can be used for **margin rescaling**:

$$\tilde{\ell}_h(\boldsymbol{y}, \boldsymbol{x}, f) = \max_{\boldsymbol{y}' \in \mathcal{Y}} \{\ell(\boldsymbol{y}, \boldsymbol{y}') + f(\boldsymbol{x}, \boldsymbol{y}')\} - f(\boldsymbol{x}, \boldsymbol{y})\,.$$

- SSVMs with Hamming loss and

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i)$$

  decompose to BR with SVMs.

[5] W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013

A. Tewari and P.L. Bartlett. On the consistency of multiclass classification methods. *JMLR*, 8:1007–1025, 2007

D. McAllester. *Generalization Bounds and Consistency for Structured Labeling in Predicting Structured Data*. MIT Press, 2007

CRFs and SSVMs for different task losses

- In SSVMs, task loss $\ell(\boldsymbol{y}, \boldsymbol{y}')$ can be used for **margin rescaling**:

$$\tilde{\ell}_h(\boldsymbol{y}, \boldsymbol{x}, f) = \max_{\boldsymbol{y}' \in \mathcal{Y}} \{\ell(\boldsymbol{y}, \boldsymbol{y}') + f(\boldsymbol{x}, \boldsymbol{y}')\} - f(\boldsymbol{x}, \boldsymbol{y})\,.$$

- SSVMs with Hamming loss and

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i)$$

decompose to BR with SVMs.

Question

Prove why this is true.

[5] W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013

A. Tewari and P.L. Bartlett. On the consistency of multiclass classification methods. *JMLR*, 8:1007–1025, 2007

D. McAllester. *Generalization Bounds and Consistency for Structured Labeling in Predicting Structured Data*. MIT Press, 2007

CRFs and SSVMs for different task losses

- In SSVMs, task loss $\ell(\boldsymbol{y}, \boldsymbol{y}')$ can be used for **margin rescaling**:

$$\tilde{\ell}_h(\boldsymbol{y}, \boldsymbol{x}, f) = \max_{\boldsymbol{y}' \in \mathcal{Y}} \{\ell(\boldsymbol{y}, \boldsymbol{y}') + f(\boldsymbol{x}, \boldsymbol{y}')\} - f(\boldsymbol{x}, \boldsymbol{y}) \,.$$

- SSVMs with Hamming loss and

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i)$$

  decompose to BR with SVMs.

Question

Prove why this is true.

- In general SSVMs are inconsistent.[5]

[5] W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013

A. Tewari and P.L. Bartlett. On the consistency of multiclass classification methods. *JMLR*, 8:1007–1025, 2007

D. McAllester. *Generalization Bounds and Consistency for Structured Labeling in Predicting Structured Data*. MIT Press, 2007

CRFs and SSVMs for different task losses

- CRFs are tailored for the subset 0/1 loss and cannot directly take other task losses into account.

[6] P. Pletscher, C.S. Ong, and J.M. Buhmann. Entropy and margin maximization for structured output learning. In *ECML/PKDD*. Springer, 2010

Q. Shi, M. Reid, and T. Caetano. Hybrid model of conditional random field and support vector machine. In *Workshop at NIPS*, 2009

K. Gimpel and N. Smith. Softmax-margin crfs: Training log-linear models with cost functions. In *HLT*, pages 733–736, 2010

CRFs and SSVMs for different task losses

- CRFs are tailored for the subset $0/1$ loss and cannot directly take other task losses into account.
- CRFs with the scoring function of the form

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i)$$

minimize Hamming loss ($\rightarrow$ BR with logistic regression).

[6] P. Pletscher, C.S. Ong, and J.M. Buhmann. Entropy and margin maximization for structured output learning. In *ECML/PKDD*. Springer, 2010

Q. Shi, M. Reid, and T. Caetano. Hybrid model of conditional random field and support vector machine. In *Workshop at NIPS*, 2009

K. Gimpel and N. Smith. Softmax-margin crfs: Training log-linear models with cost functions. In *HLT*, pages 733–736, 2010

CRFs and SSVMs for different task losses

- CRFs are tailored for the subset 0/1 loss and cannot directly take other task losses into account.
- CRFs with the scoring function of the form

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i)$$

minimize Hamming loss ($\rightarrow$ BR with logistic regression).

Question

Prove why this is true.

[6] P. Pletscher, C.S. Ong, and J.M. Buhmann. Entropy and margin maximization for structured output learning. In *ECML/PKDD*. Springer, 2010

Q. Shi, M. Reid, and T. Caetano. Hybrid model of conditional random field and support vector machine. In *Workshop at NIPS*, 2009

K. Gimpel and N. Smith. Softmax-margin crfs: Training log-linear models with cost functions. In *HLT*, pages 733–736, 2010

CRFs and SSVMs for different task losses

- CRFs are tailored for the subset 0/1 loss and cannot directly take other task losses into account.
- CRFs with the scoring function of the form

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i)$$

minimize Hamming loss ($\rightarrow$ BR with logistic regression).

Question

Prove why this is true.

- Some works on incorporating margin into CRFs.[6]

[6] P. Pletscher, C.S. Ong, and J.M. Buhmann. Entropy and margin maximization for structured output learning. In *ECML/PKDD*. Springer, 2010

Q. Shi, M. Reid, and T. Caetano. Hybrid model of conditional random field and support vector machine. In *Workshop at NIPS*, 2009

K. Gimpel and N. Smith. Softmax-margin crfs: Training log-linear models with cost functions. In *HLT*, pages 733–736, 2010

SSVMs vs. BR

Table: SSVMs with pairwise term[7] vs. BR with LR[8].

Dataset	SSVM Best	BR LR
Scene	0.101±.003	0.102±.003
Yeast	0.202±.005	0.199±.005
Synth1	0.069±.001	0.067±.002
Synth2	0.058±.001	0.084±.001

- There is almost no difference between both algorithms.

7 Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In *ICML*. Omnipress, 2008

8 K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An analysis of chaining in multi-label classification. In *ECAI*, 2012

Probabilistic classifier chains

- Probabilistic classifier chains (PCCs)[9] are an efficient reduction method similar to conditional probability trees.[10]
- They estimate the joint conditional distribution $P(\boldsymbol{y} \,|\, \boldsymbol{x})$ as CRFs.
- Their idea is to repeatedly apply the **product rule of probability**:

$$P(\boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{i=1}^{m} P(y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1}) \,.$$

[9] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning Journal*, 85:333–359, 2011

K. Dembczyński, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *ICML*, pages 279–286. Omnipress, 2010

[10] A. Beygelzimer, J. Langford, Y. Lifshits, G. B. Sorkin, and A. L. Strehl. Conditional probability tree estimation analysis and algorithms. In *UAI*, pages 51–58, 2009

Probabilistic classifier chains

- Probabilistic classifier chains (PCCs)[9] are an efficient reduction method similar to conditional probability trees.[10]
- They estimate the joint conditional distribution $P(\boldsymbol{y} \mid \boldsymbol{x})$ as CRFs.
- Their idea is to repeatedly apply the **product rule of probability**:

$$P(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{i=1}^{m} P(y_i \mid \boldsymbol{x}, y_1, \ldots, y_{i-1}) \,.$$

- **Example**:

$$P(y_1, y_2 \mid \boldsymbol{x}) = \frac{P(y_1, \boldsymbol{x})}{P(\boldsymbol{x})} \frac{P(y_1, y_2, \boldsymbol{x})}{P(y_1, \boldsymbol{x})} = P(y_1 \mid \boldsymbol{x}) P(y_2 \mid y_1, \boldsymbol{x}) \,.$$

---

[9] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning Journal*, 85:333–359, 2011

K. Dembczyński, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *ICML*, pages 279–286. Omnipress, 2010

[10] A. Beygelzimer, J. Langford, Y. Lifshits, G. B. Sorkin, and A. L. Strehl. Conditional probability tree estimation analysis and algorithms. In *UAI*, pages 51–58, 2009

Probabilistic classifier chains

- PCCs follow a reduction to a sequence of subproblems:

$$(\boldsymbol{x}, \boldsymbol{y}) \longrightarrow (\boldsymbol{x}' = (\boldsymbol{x}, y_1, \ldots, y_{i-1}), y = y_i), \quad i = 1, \ldots, m$$

- Learning of PCCs relies on constructing probabilistic classifiers for estimating

$$P(y_i | \boldsymbol{x}, y_1, \ldots, y_{i-1}) \,,$$

  independently for each $i = 1, \ldots, m$.
- Let us denote these estimates by

$$Q(y_i | \boldsymbol{x}, y_1, \ldots, y_{i-1}) \,.$$

- The final model is:

$$Q(\boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{i=1}^{m} Q(y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1}) \,.$$

Probabilistic classifier chains

- We can use scoring functions of the form $f_i(\boldsymbol{x}', y_i)$ and train logistic regression (or any probabilistic classifier) to get $Q(y_i|\boldsymbol{x}')$.
- By using the linear models, the overall scoring function takes the form:

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}, y_i) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l)$$

- Theoretically the order of labels does not matter, but practically it may.

Probabilistic classifier chains

- PCCs enable estimation of probability of any label vector $\boldsymbol{y}$.
- To get such an estimate it is enough to compute:

$$Q(\boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{i=1}^{m} Q(y_i | \boldsymbol{x}, y_1, \ldots, y_{i-1})$$

- There is, however, a problem how to compute the optimal decision $\boldsymbol{h}(\boldsymbol{x})$ (with respect to Q) for a given loss function.

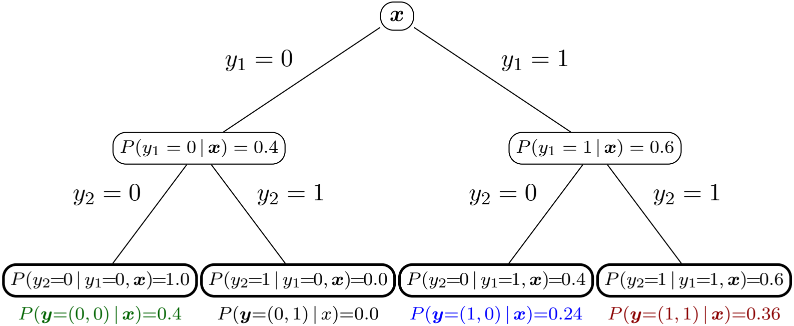
Probabilistic classifier chains

- Inference in PCCs:
  - Greedy search,
  - Advanced search techniques: beam search, uniform-cost search,
  - Exhaustive search,
  - Sampling + inference.

Greedy search

- Greedy search follows the chain by using predictions from previous steps as inputs in the consecutive steps:
  - $f_1 : \boldsymbol{x} \mapsto \hat{y}_1$
  - $f_2 : \boldsymbol{x}, \hat{y}_1 \mapsto \hat{y}_2$
  - $f_3 : \boldsymbol{x}, \hat{y}_1, y_2 \mapsto \hat{y}_3$
  - $\ldots$
  - $f_m : \boldsymbol{x}, \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{m-1} \mapsto \hat{y}_m$
- Greedy search is fast ($O(m)$).
- Does not require probabilistic classifiers.
- The resulting $\hat{\boldsymbol{y}}$ is neither the joint nor the marginal mode.
- Optimal if labels are independent or the probability of the joint mode > 0.5.

- Greedy search fails for the joint mode and the marginal mode:



Advanced search techniques

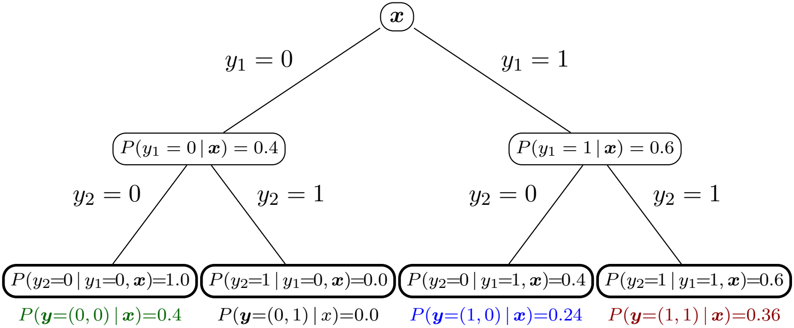
- Advanced search techniques: beam search,[11] a variant of uniform-cost search.[12]
- Finding the joint mode relies on finding the most probable path in the tree.
- The use of a priority queue and a cut point gives a fast algorithm with provable guarantees.

[11] A. Kumar, S. Vembu, A.K. Menon, and C. Elkan. Beam search algorithms for multilabel learning. In *Machine Learning*, 2013

[12] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An analysis of chaining in multi-label classification. In *ECAI*, 2012

Advanced search techniques

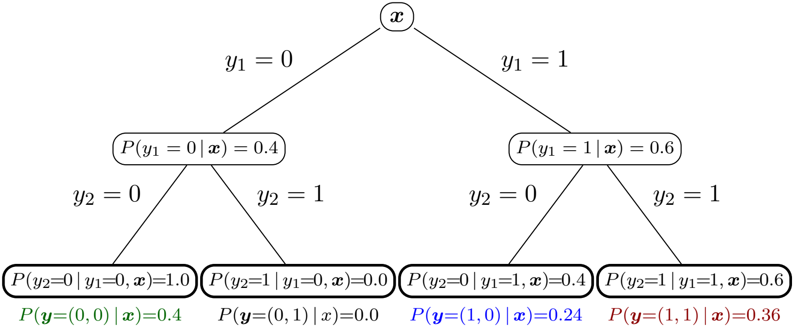
- Uniform-cost search



- Priority list $\mathcal{Q}$:

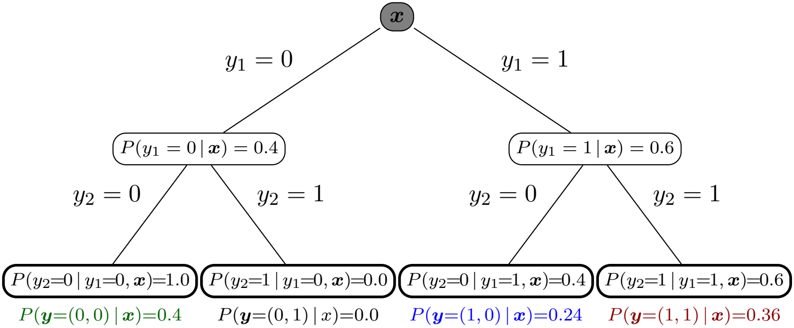
Advanced search techniques

- Uniform-cost search



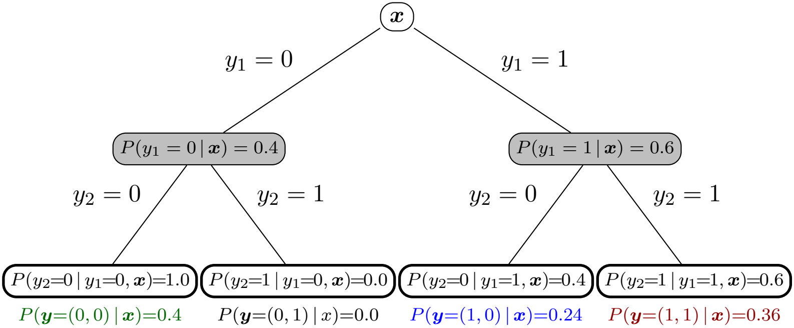
- Priority list $\mathcal{Q}$: root

- Uniform-cost search



- Priority list $\mathcal{Q}$:

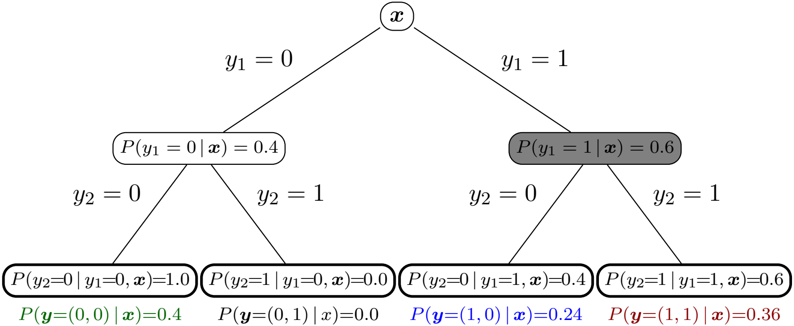
- Uniform-cost search



- Priority list $\mathcal{Q}$: [(1),0.6], [(0),0.4]

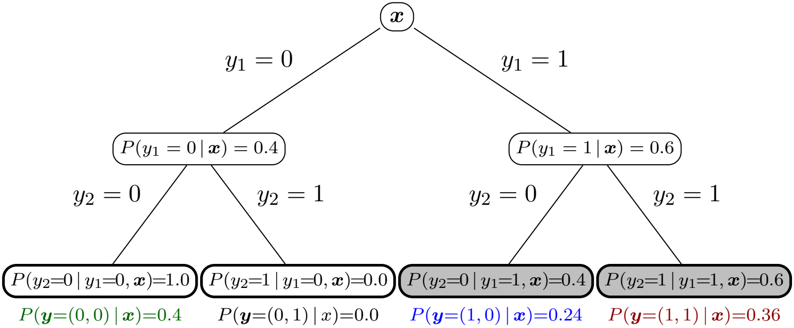
Advanced search techniques

- Uniform-cost search



- Priority list $\mathcal{Q}$: [(0),0.4]

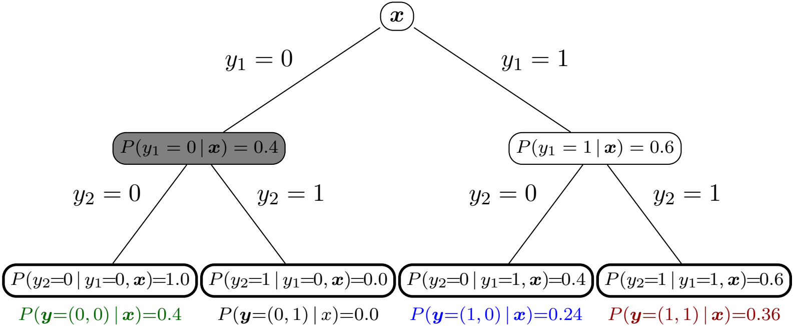
- Uniform-cost search



- Priority list $\mathcal{Q}$: [(0),0.4], [(1,1),0.36], [(1,0),0.24]

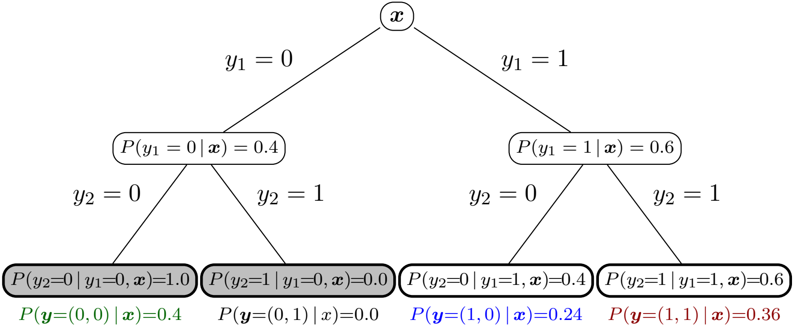
Advanced search techniques

- Uniform-cost search



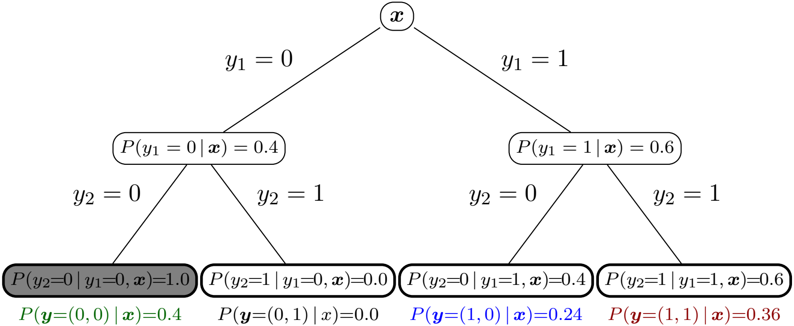
- Priority list $\mathcal{Q}$: [(1,1),0.36], [(1,0),0.24]

- Uniform-cost search



- Priority list $\mathcal{Q}$: [(0,0),0.4], [(1,1),0.36], [(1,0),0.24], [(0,1),0.0]

- Uniform-cost search



- Priority list $\mathcal{Q}$: Solution is found

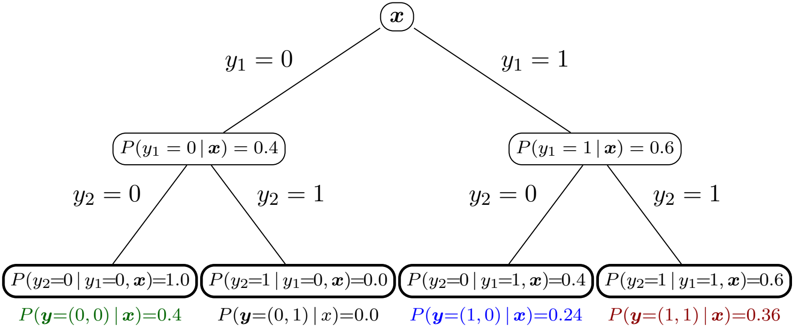
Advanced search techniques

- ϵ-approximation inference:[13]
  - Insert items to priority queue $\mathcal{Q}$ with partial probabilities $> \epsilon$.
  - If solution has not been found, then perform greedy search from nodes without survived children.

[13] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An analysis of chaining in multi-label classification. In *ECAI*, 2012

ϵ-approximation inference

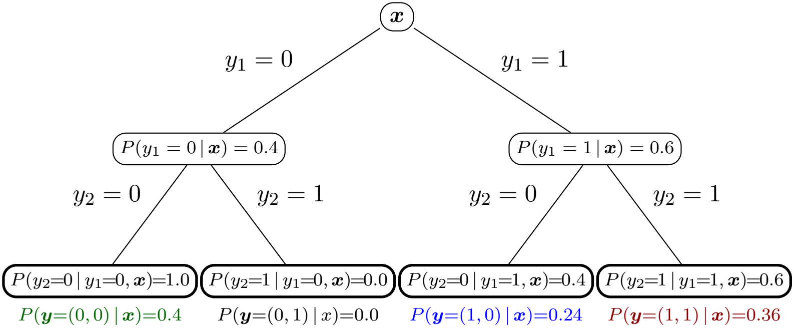
- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$:

ϵ-approximation inference

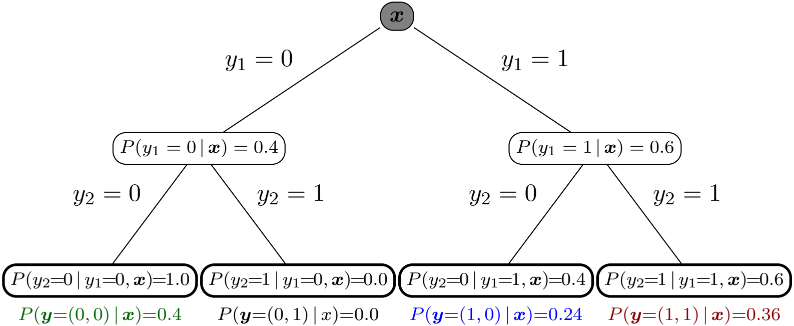
- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$: root

ϵ-approximation inference

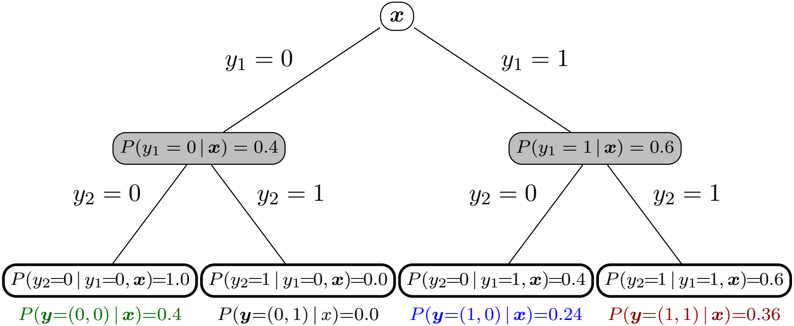
- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$: $\epsilon = 0.5$

ϵ-approximation inference

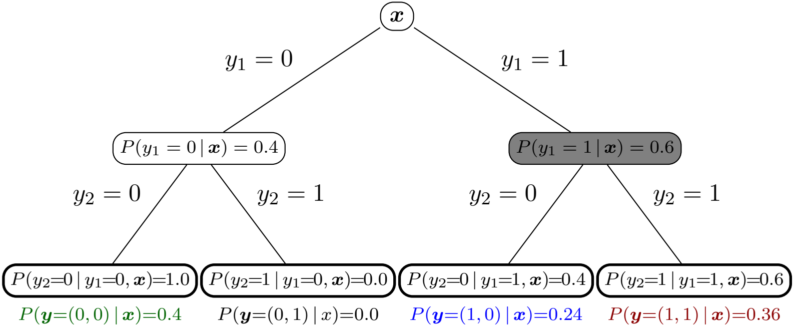
- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$: [(1),0.6], $\epsilon = 0.5$, [(0),0.4]

ϵ-approximation inference

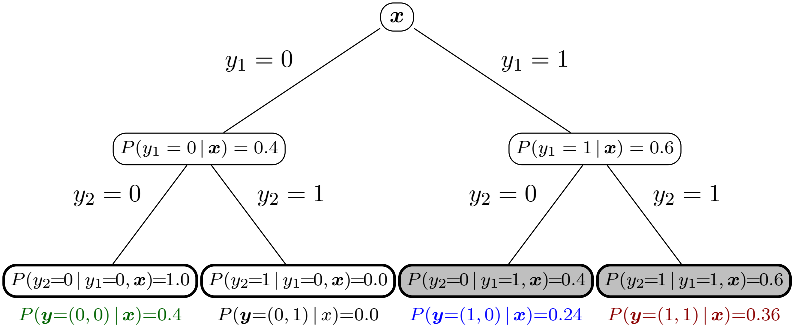
- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$: $\epsilon = 0.5$, [(0),0.4]

ϵ-approximation inference

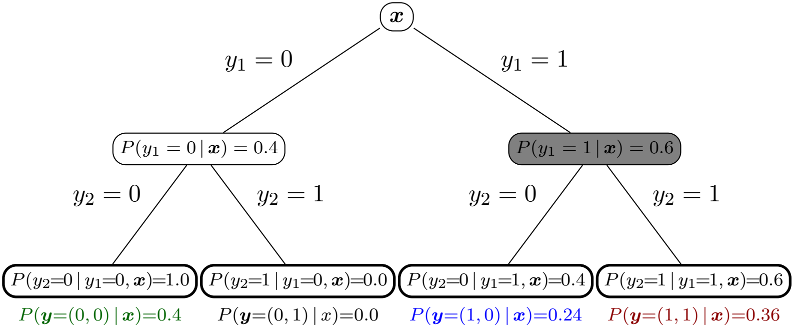
- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$: $\epsilon = 0.5$, [(0),0.4], [(1,1),0.36], [(1,0),0.24]

ϵ-approximation inference

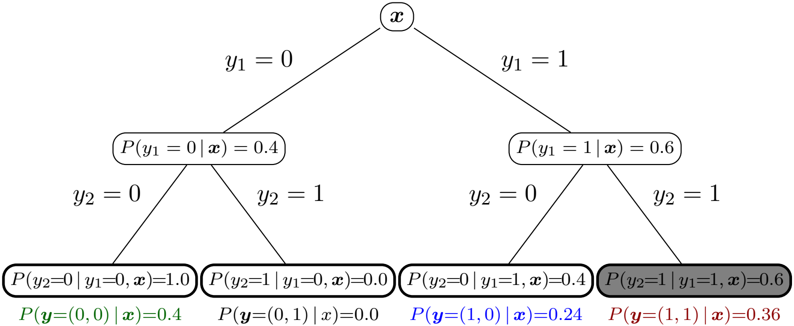
- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$: Start the greedy search from (1).

ϵ-approximation inference

- $\epsilon = 0.5$



- Priority list $\mathcal{Q}$: Suboptimal solution (1,1) is found.

ϵ-approximation inference

- For $\epsilon = 0.5$, it is equivalent to greedy search.
- For $\epsilon = 0.0$, it is equivalent to uniform-cost search.
- For a given ϵ, the following guarantees can be given:

  Theorem: Let $\epsilon = 2^{-c}$, where $1 \leq c \leq m$. To get the label vector $\hat{\boldsymbol{y}}$ the algorithm needs $\mathcal{O}(m2^c)$ calls to node classifiers with a guarantee that:

  $$Q(\boldsymbol{y}^* \,|\, \boldsymbol{x}) - Q(\hat{\boldsymbol{y}} \,|\, \boldsymbol{x}) \leq \epsilon - 2^{-m}$$

Question

Prove this result.

ϵ-approximation inference

- The ϵ-approximate inference will always find the joint mode if its probability mass $\geq \epsilon$.
- In other words, the algorithm with $\epsilon = 0$ finds the solution in a linear time of $1/p_{\max}$, where $p_{\max}$ is the probability mass of the joint mode.
- For problems with low noise (high values of $p_{\max}$), this method should work very fast.
- Greedy search has very bad guarantees:

$$Q(\boldsymbol{y}^* \,|\, \boldsymbol{x}) - Q(\hat{\boldsymbol{y}} \,|\, \boldsymbol{x}) \leq 0.5 - 2^{-m} \,.$$

Regret bound for PCC

- The typical approach for estimating probabilities of $\boldsymbol{y}$ is minimization of the logistic loss:

$$\ell_{\log}(\boldsymbol{y}, \boldsymbol{x}, f) = -\log Q(\boldsymbol{y} \,|\, \boldsymbol{x}) \,,$$

  where f is a model that delivers estimate $Q(\boldsymbol{y} \,|\, \boldsymbol{x})$ of $P(\boldsymbol{y} \,|\, \boldsymbol{x})$.
- By using the chain rule of probability, we get:

$$\begin{aligned} \ell_{\log}(\boldsymbol{y}, \boldsymbol{x}, f) &= -\log \prod_{i=1}^{m} Q(y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1}) \\ &= -\sum_{i=1}^{m} \log Q(y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1}) = -\sum_{i=1}^{m} \log Q_i(\boldsymbol{y}) \,, \end{aligned}$$

  where we use the notation $Q_i(\boldsymbol{y}) = Q(y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1})$.
- This is a sum of univariate log losses on a path from the root to the leaf corresponding to $\boldsymbol{y}$.

Regret bound for PCC

- **Theorem**: For all distributions and all PCCs trained with logistic regression f and used with the ϵ-approximate inference algorithm,

$$\mathrm{Reg}_{0/1}(\mathrm{PCC}_\epsilon(f)) \leq \sqrt{2m\overline{\mathrm{Reg}_{\log}}(f)} + \epsilon$$

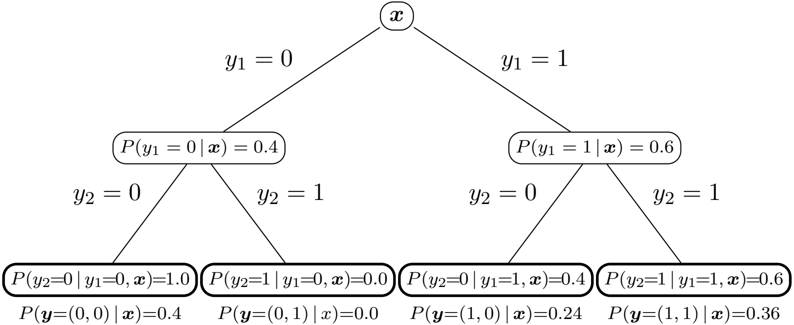
where $\overline{\mathrm{Reg}_{\log}}(f)$ is the average logistic regret over the paths from the root to the leafs.

PCC for other losses

- **Exhaustive search**:
  - Compute the entire distribution $Q(\boldsymbol{y} \,|\, \boldsymbol{x})$ by traversing the probability tree.
  - Use an appropriate inference for a given loss ℓ on the estimated joint distribution:
$$\hat{y} = \arg\max_{\boldsymbol{h} \in \mathcal{Y}} \sum_{y \in \mathcal{Y}} Q(\boldsymbol{y} \,|\, \boldsymbol{x}) \ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))$$
  - This approach is extremely costly.
- **Ancestral sampling**:
  - Sampling can be easily performed by using the probability tree.
  - Make inference based on the empirical distribution.
  - Hamming loss: estimate marginal probabilities.

Probabilistic classifier chains

- Exhaustive search and ancestral sampling:



- Sample: (1,1), (1,0), (0,0), (0,0), (1,1), (0,0), (1,0), (1,1), (0,0) . . .

Probabilistic classifier chains

Table: PCC vs. SSVMs on Hamming loss and PCC vs. BR on subset 0/1 loss.

Dataset	PCC	SSVM Best	PCC	BR
	Hamming loss		subset 0/1 loss	
Scene	0.104±.004	0.101±.003	0.385±.014	0.509±.014
Yeast	0.203±.005	0.202±.005	0.761±.014	0.842±.012
Synth1	0.067±.001	0.069±.001	0.239±.006	0.240±.006
Synth2	0.000±.000	0.058±.001	0.000±.000	0.832±.004
Reuters	0.060±.002	0.045±.001	0.598±.009	0.689±.008
Mediamill	0.172±.001	0.182±.001	0.885±.003	0.902±.003

Recurrent classifiers

- PCCs are similar to Maximum Entropy Markov Models (MEMMs)[14] introduced for sequence learning:
  - One logistic classifier that takes dependences up to the k-th degree.
  - Inference by dynamic programming.
- Searn[15] is another approach that is based on recurrent classifiers:
  - Linear inference.
  - Learning is performed in the iterative way to solve the egg and the chicken problem: output of the classifier is also used as input to the classifier.

[14] A. K. McCallum, D. Freitag, and F. (2000) Pereira. Maximum entropy markov models for information extraction and segmentation. In *ICML*, 2000

[15] H. Daumé III, J. Langford, and D. Marcu. Search-based structured prediction. *Machine Learning*, 75:297–325, 2009

Output search space

- More advanced search techniques.
- Popular topic in structured output prediction.
- Search techniques for different task losses.[16]

[16] J.R. Doppa, A. Fern, and P. Tadepalli. Structured prediction via output space search. *JMLR*, 15:1317–1350, 2014

PCC for multi-class classification

- PCC can be used for multi-class classification:
  - Map each class label to a label vector: binary coding, hierarchical clustering, ...
  - The same idea as in conditional probability trees (CPT).[17]
  - Label tree classifiers for efficient multi-class classification.[18]

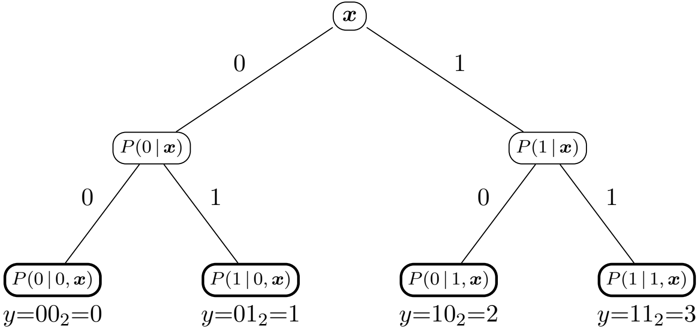
[17] A. Beygelzimer, J. Langford, Y. Lifshits, G. B. Sorkin, and A. L. Strehl. Conditional probability tree estimation analysis and algorithms. In *UAI*, pages 51–58, 2009

[18] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *NIPS*, pages 163–171. Curran Associates, Inc., 2010

J. Deng, S. Satheesh, A. C. Berg, and Fei Fei F. Li. Fast and balanced: Efficient label tree learning for large scale object recognition. In *NIPS*, pages 567–575. 2011

PCC for multi-class classification

- We assign each class an integer from 0 to $k-1$ and code it by its binary representation on m bits.
- Example: $k=4$, $\mathcal{Y}=\{0,1,2,3\}$.
- k leaves, one for each class.



Consistent and efficient label tree classifiers

- PCC: fast learning but inference can be costly.
- Greedy search is the most efficient, but is not consistent.
- How to ensure a linear inference in m for any loss?

Filter trees

- Filter trees (FT)[19] have been originally introduced for cost-sensitive multi-class classification, but can be easily adapted to multi-label classification.

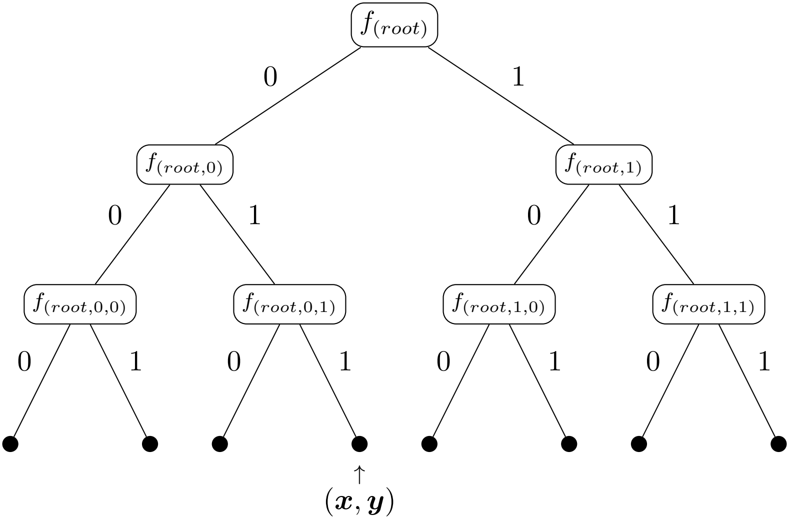
[19] A. Beygelzimer, J. Langford, and P. D. Ravikumar. Error-correcting tournaments. In *ALT*, pages 247–262, 2009

Filter trees

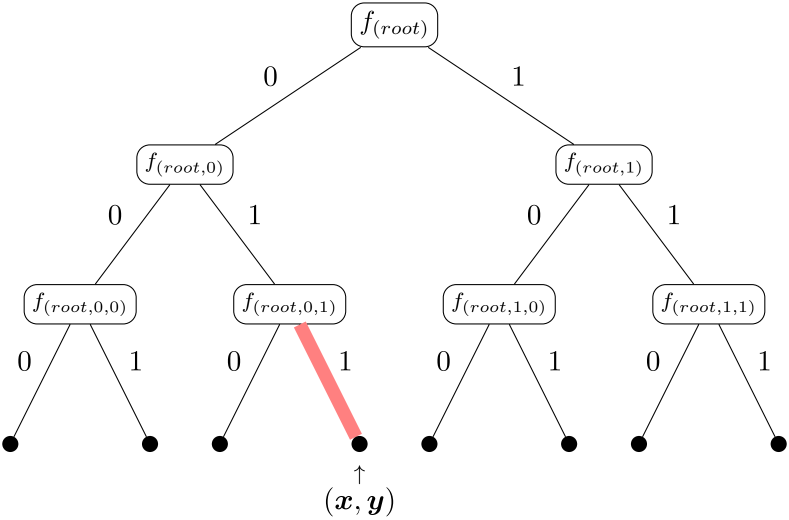
- Filter trees (FT)[19] have been originally introduced for cost-sensitive multi-class classification, but can be easily adapted to multi-label classification.
- They use a **bottom-up** learning algorithm to train the label tree.
- Based on a single **elimination tournament** on the set of classes/label combinations.

[19] A. Beygelzimer, J. Langford, and P. D. Ravikumar. Error-correcting tournaments. In *ALT*, pages 247–262, 2009

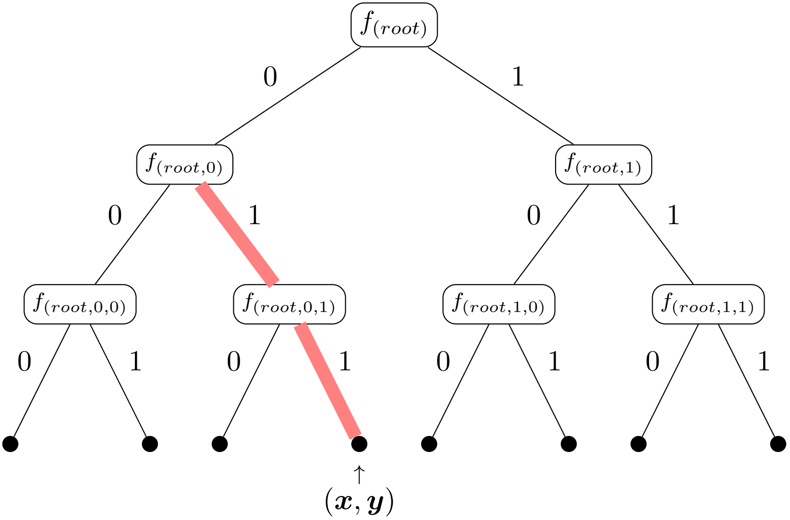
Filter trees : Example



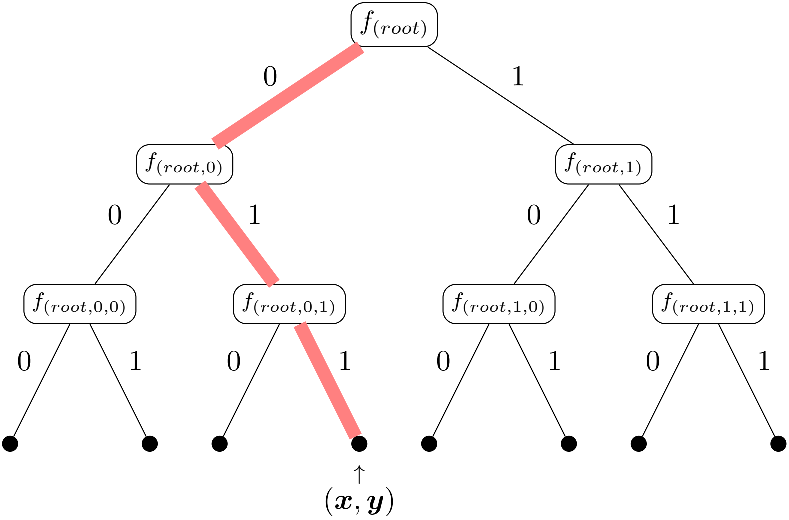
Filter trees : Example



Filter trees : Example



Filter trees : Example



Filter trees

- FT are trained to predict y_{i+1} based on previous labels.
- FT implicitly transforms the underlying distribution P over multi-class/multi-label examples into a **specific distribution** P^{FT} over weighted binary examples.
- The inference procedure of FT is straight-forward and uses the **greedy search**.
- FT are **consistent** for any cost function.

Filter trees

- Filter tree training:

Input: training set $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$, importance-weighted binary learner $Learn$
for each non-leaf node $\boldsymbol{v} = (root, y_1, \ldots, y_{i-1})$ in the order from leaves to root **do**
  $S_{\boldsymbol{v}} = \emptyset$
  for each traning example $(\boldsymbol{x}, \boldsymbol{y})$ **do**
    Let $\boldsymbol{y}_l$ and $\boldsymbol{y}_r$ be the two label vectors input to $\boldsymbol{v}$
    $y_i \leftarrow \arg\min_{l,r}\{\ell(\boldsymbol{y}, \boldsymbol{y}_l), \ell(\boldsymbol{y}, \boldsymbol{y}_r)\}$
    $w = |\ell(\boldsymbol{y}, \boldsymbol{y}_l) - \ell(\boldsymbol{y}, \boldsymbol{y}_r)|$
    $S_{\boldsymbol{v}} \leftarrow S_{\boldsymbol{v}} \cup (\boldsymbol{x}, y_i, w)$
  end for
  $f_{\boldsymbol{v}} = Learn(S_{\boldsymbol{v}})$
end for
return $f = \{f_{\boldsymbol{v}}\}$

Filter trees

- Different training schemes possible:
  - Train a classifier in each node,
  - Train a classifier on each level,
  - Train one global binary classifier (in several loops).
- The tree in multi-label classification is given naturally, but the order of labels may influence the performance.
- In general case, training can be costly ($O(2^m)$), but efficient variants for multi-label classification exist.[20]
- Prediction is always linear in the number of labels ($O(m)$).

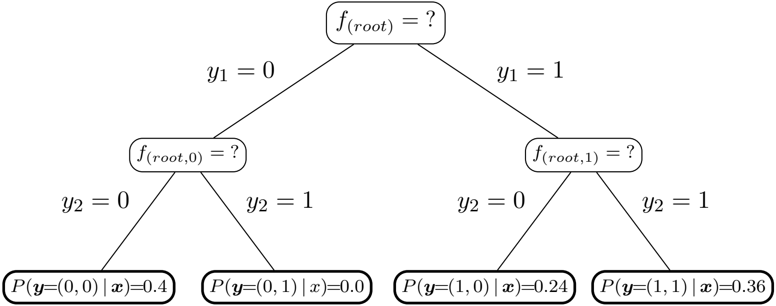
[20] Chun-Liang Li and Hsuan-Tien Lin. Condensed filter tree for cost-sensitive multi-label classification. In *ICML*, pages 423–431, 2014

Filter trees

- Filter trees for the subset 0/1 loss use a training example only on one path from a leaf to the root.
- Therefore, training in this case is also linear in the number of labels ($O(m)$).
- Moreover, all misclassified examples are filter out, i.e., $f_{(root,y_1,\dots y_i)}(\boldsymbol{x})$ predicts y_{i+1} given that all classifiers below predict the subsequent labels correctly:

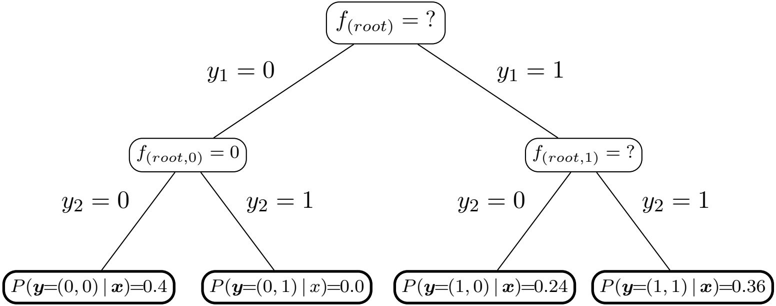
$$f_{(root,y_1,\dots y_i)} : \boldsymbol{x} \mapsto (y_{i+1} \,|\, y_{j+1} = f_{(root,y_1,\dots y_j)} : j = i+1, \dots, m-1)$$

- Consistency of FT for a single $\boldsymbol{x}$:

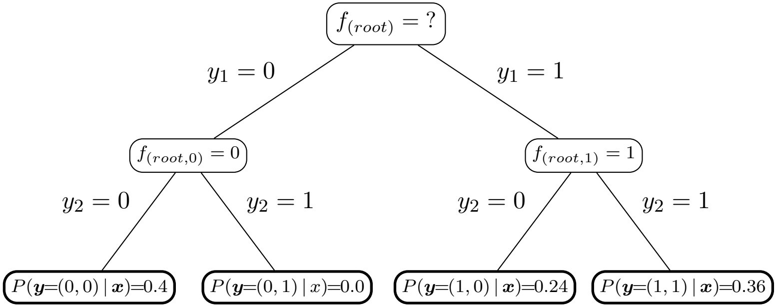


Filter trees: Consistency

- Consistency of FT for a single $\boldsymbol{x}$:

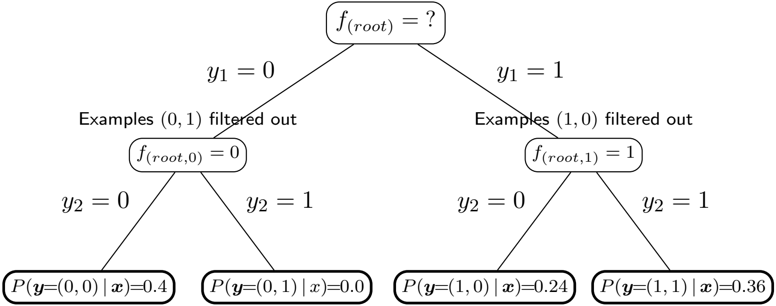


- Consistency of FT for a single $\boldsymbol{x}$:



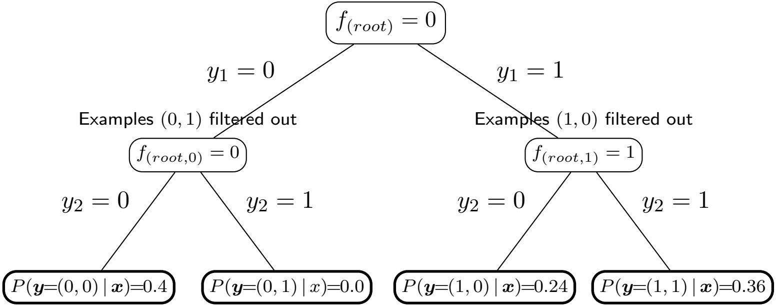
Filter trees: Consistency

- Consistency of FT for a single $\boldsymbol{x}$:



Filter trees: Consistency

- Consistency of FT for a single $\boldsymbol{x}$:



Regret bound for filter trees

- Let $f_{\boldsymbol{v}}$ be a classifier for the binary classification problem induced at node $\boldsymbol{v}$.
- The average binary regret is defined as:

$$\overline{\mathrm{Reg}}_{0/1}(f, P^{\mathrm{FT}}) = \frac{1}{\sum_{\boldsymbol{v}} W_{\boldsymbol{v}}} \sum_{\boldsymbol{v}} \mathrm{Reg}_{0/1}(f_{\boldsymbol{v}}, P^{\mathrm{FT}}_{\boldsymbol{v}}) W_{\boldsymbol{v}},$$

where

$$W_{\boldsymbol{v}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})} w_{\boldsymbol{v}}(\boldsymbol{x}, \boldsymbol{y}).$$

- **Theorem**:[21] For all distributions and all FT classifiers trained with a binary classifier f, and any cost-matrix-based task loss ℓ,

$$\mathrm{Reg}_{\ell}(\mathrm{FT}(f)) \leq \overline{\mathrm{Reg}}_{0/1}(f, P^{\mathrm{FT}}) \sum_{\boldsymbol{v}} W_{\boldsymbol{v}} \,.$$

[21] A. Beygelzimer, J. Langford, and P. D. Ravikumar. Error-correcting tournaments. In *ALT*, pages 247–262, 2009

Regret bound for filter trees

- For subset 0/1 loss, we have

$$\sum_{\boldsymbol{v}} w_{\boldsymbol{v}}(\boldsymbol{x}, \boldsymbol{y}) \leq m \,,$$

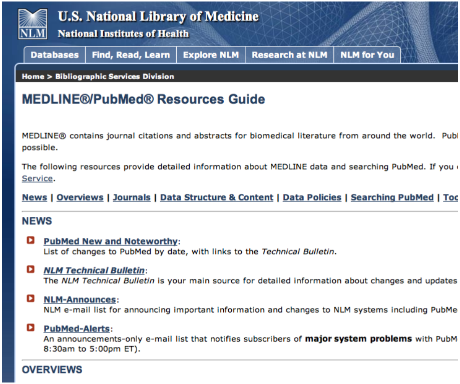
  since each training example $(\boldsymbol{x}, \boldsymbol{y})$ will appear in training at most once per level with importance weight 1.
- The regret bound has then the form:

$$\mathrm{Reg}_{\ell}(\mathrm{FT}(f)) \leq m \overline{\mathrm{Reg}}_{0/1}(f, P^{\mathrm{FT}}) \,.$$

Outline

Maximization of the F-measure

- Applications: Information retrieval, document tagging, and NLP.
  - JRS 2012 Data Mining Competition: Indexing documents from MEDLINE or PubMed Central databases with concepts from the Medical Subject Headings ontology.



- The F_β-measure-based loss function (F_β-loss):

$$\begin{aligned}\ell_{F_\beta}(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) &= 1 - F_\beta(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) \\ &= 1 - \frac{(1+\beta^2)\sum_{i=1}^m y_i h_i(\boldsymbol{x})}{\beta^2 \sum_{i=1}^m y_i + \sum_{i=1}^m h_i(\boldsymbol{x})} \in [0,1] \,.\end{aligned}$$

- Provides a **better balance** between relevant and irrelevant labels.
- However, it **is not easy** to optimize.

SSVMs for F_β-based loss

- SSVMs can be used to minimize F_β-based loss.
- Rescale the margin by $\ell_F(\boldsymbol{y}, \boldsymbol{y}')$.
- Two algorithms:[22]

RML

No label interactions:

$$f(\boldsymbol{y}, \boldsymbol{x}) = \sum_{i=1}^{m} f_i(y_i, \boldsymbol{x})$$

Quadratic learning and linear prediction

SML

Submodular interactions:

$$f(\boldsymbol{y}, \boldsymbol{x}) = \sum_{i=1}^{m} f_i(y_i, \boldsymbol{x}) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l)$$

More complex (graph-cut and approximate algorithms)

- Both are inconsistent.

[22] J. Petterson and T. S. Caetano. Reverse multi-label learning. In *NIPS*, pages 1912–1920, 2010
J. Petterson and T. S. Caetano. Submodular multi-label learning. In *NIPS*, pages 1512–1520, 2011

Plug-in rule approach

- Plug estimates of required parameters into the Bayes classifier:[23]

$$
\begin{aligned}
\boldsymbol{h}^* &= \operatorname*{arg\,min}_{\boldsymbol{h}\in\mathcal{Y}} \mathbb{E}\left[\ell_{F_\beta}(\boldsymbol{Y},\boldsymbol{h})\right] \\
&= \operatorname*{arg\,max}_{\boldsymbol{h}\in\mathcal{Y}} \sum_{\boldsymbol{y}\in\mathcal{Y}} P(\boldsymbol{y}) \frac{(\beta+1)\sum_{i=1}^m y_i h_i}{\beta^2\sum_{i=1}^m y_i + \sum_{i=1}^m h_i}
\end{aligned}
$$

- **No closed form** solution for this optimization problem.
- The problem **cannot** be solved **naively** by brute-force search:
  - This would require to check all possible combinations of labels (2^m)
  - To sum over 2^m number of elements for computing the expected value.
  - The number of parameters to be estimated $(P(\boldsymbol{y}))$ is 2^m.

[23] W. Waegeman, K. Dembczynski, W. Cheng A. Jachnik, and E. Hüllermeier. On the Bayes-optimality of F-measure maximizers. *Minor revision*, 2014

Plug-in rule approach

- Approximation needed?

[24] N. Ye, K. Chai, W. Lee, and H. Chieu. Optimizing F-measures: a tale of two approaches. In *ICML*, 2012

[25] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An exact algorithm for F-measure maximization. In *NIPS*, volume 25, 2011

[26] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman, and E. Hüllermeier. Optimizing the F-measure in multi-label classification: Plug-in rule approach versus structured loss minimization. In *ICML*, 2013

Plug-in rule approach

- Approximation needed? Not really. The exact solution is tractable!

---

[24] N. Ye, K. Chai, W. Lee, and H. Chieu. Optimizing F-measures: a tale of two approaches. In *ICML*, 2012

[25] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An exact algorithm for F-measure maximization. In *NIPS*, volume 25, 2011

[26] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman, and E. Hüllermeier. Optimizing the F-measure in multi-label classification: Plug-in rule approach versus structured loss minimization. In *ICML*, 2013

Plug-in rule approach

- Approximation needed? Not really. The exact solution is tractable!

LFP:

Assumes label independence.

Linear number of parameters: $P(y_i = 1)$.

Inference based on dynamic programming.[24]

Reduction to LR for each label.

EFP:

No assumptions.

Quadratic number of parameters: $P(y_i = 1, s = \sum_i y_i)$.

Inference based on matrix multiplication and top k selection.[25]

Reduction to multinomial LR for each label.

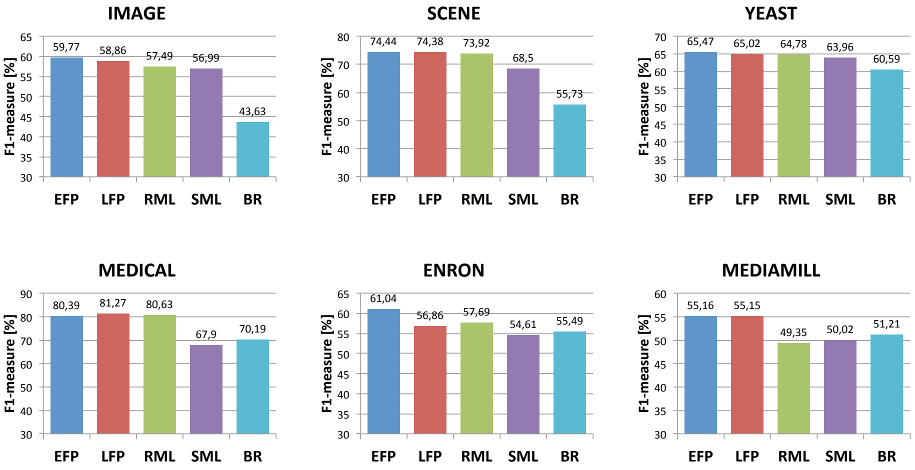
- EFP is consistent.[26]

[24] N. Ye, K. Chai, W. Lee, and H. Chieu. Optimizing F-measures: a tale of two approaches. In *ICML*, 2012

[25] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An exact algorithm for F-measure maximization. In *NIPS*, volume 25, 2011

[26] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman, and E. Hüllermeier. Optimizing the F-measure in multi-label classification: Plug-in rule approach versus structured loss minimization. In *ICML*, 2013

Maximization of the F-measure



Outline

Multi-label ranking

Serena romps to fifth Wimbledon title against brave Radwanska

By **Paul Gittings**, CNN
July 7, 2012 -- Updated 2220 GMT (0620 HKT)

STORY HIGHLIGHTS

- Serena Williams wins fifth Wimbledon crown
- American beats Agnieszka Radwanska of Poland 6-1 5-7 6-2
- Radwanska battles respiratory

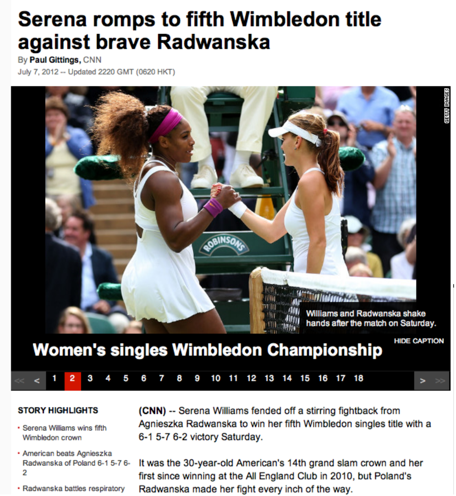
(CNN) -- Serena Williams fended off a stirring fightback from Agnieszka Radwanska to win her fifth Wimbledon singles title with a 6-1 5-7 6-2 victory Saturday.

It was the 30-year-old American's 14th grand slam crown and her first since winning at the All England Club in 2010, but Poland's Radwanska made her fight every inch of the way.

Multi-label classification

politics	0
economy	0
business	0
sport	1
tennis	1
soccer	0
show-business	0
celebrities	1
⋮	
England	1
USA	1
Poland	1
Lithuania	0

Multi-label ranking



Serena romps to fifth Wimbledon title against brave Radwanska

By **Paul Gittings,** CNN

July 7, 2012 -- Updated 2220 GMT (0620 HKT)

Williams and Radwanska shake hands after the match on Saturday.

HIDE CAPTION

Women's singles Wimbledon Championship

<< < 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 > >>

STORY HIGHLIGHTS

- Serena Williams wins fifth Wimbledon crown
- American beats Agnieszka Radwanska of Poland 6-1 5-7 6-2
- Radwanska battles respiratory

(CNN) -- Serena Williams fended off a stirring fightback from Agnieszka Radwanska to win her fifth Wimbledon singles title with a 6-1 5-7 6-2 victory Saturday.

It was the 30-year-old American's 14th grand slam crown and her first since winning at the All England Club in 2010, but Poland's Radwanska made her fight every inch of the way.

Multi-label ranking

tennis

$\curlyvee$

sport

$\curlyvee$

England

$\curlyvee$

Poland

$\curlyvee$

USA

$\curlyvee$

$\vdots$

$\curlyvee$

politics

- **Ranking loss**:

$$\ell_{\text{rnk}}(\boldsymbol{y}, \boldsymbol{f}) = w(\boldsymbol{y}) \sum_{(i,j)\,:\, y_i > y_j} \left([\![f_i(\boldsymbol{x}) < f_j(\boldsymbol{x})]\!] + \frac{1}{2} [\![f_i(\boldsymbol{x}) = f_j(\boldsymbol{x})]\!] \right) ,$$

where $w(\boldsymbol{y}) < w_{max}$ is a weight function.

	X_1	X_2	Y_1		Y_2		$\ldots$		Y_m
$\boldsymbol{x}$	4.0	2.5	1		0				0
			h_2	$>$	h_1	$>$	$\ldots$	$>$	h_m

- **Ranking loss**:

$$\ell_{\mathrm{rnk}}(\boldsymbol{y}, \boldsymbol{f}) = w(\boldsymbol{y}) \sum_{(i,j)\,:\,y_i > y_j} \left([\![f_i(\boldsymbol{x}) < f_j(\boldsymbol{x})]\!] + \frac{1}{2} [\![f_i(\boldsymbol{x}) = f_j(\boldsymbol{x})]\!] \right) \,,$$

  where $w(\boldsymbol{y}) < w_{max}$ is a weight function.

  The weight function $w(\boldsymbol{y})$ is usually used to normalize the range of rank loss to $[0, 1]$:

$$w(\boldsymbol{y}) = \frac{1}{n_+ n_-},$$

  i.e., it is equal to the inverse of the total number of pairwise comparisons between labels.

Pairwise surrogate losses

- The most intuitive approach is to use pairwise **convex surrogate** losses of the form

$$\tilde{\ell}_\phi(\boldsymbol{y}, \boldsymbol{f}) = \sum_{(i,j):\, y_i > y_j} w(\boldsymbol{y})\phi(f_i - f_j)\,,$$

  where ϕ is
  - an exponential function (BoosTexter)[27]: $\phi(f) = e^{-f}\,,$
  - logistic function (LLLR)[28]: $\phi(f) = \log(1 + e^{-f})\,,$
  - or hinge function (RankSVM)[29]: $\phi(f) = \max(0, 1 - f)\,.$

---

[27] R. E. Schapire and Y. Singer. BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning*, 39(2/3):135–168, 2000

[28] O. Dekel, Ch. Manning, and Y. Singer. Log-linear models for label ranking. In *NIPS*. MIT Press, 2004

[29] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *NIPS*, pages 681–687, 2001

Multi-label ranking

- This approach is, however, **inconsistent** for the most commonly used convex surrogates.[30]
- The **consistent** classifier can be, however, obtained by using univariate loss functions[31] ...

[30] J. Duchi, L. Mackey, and M. Jordan. On the consistency of ranking algorithms. In *ICML*, pages 327–334, 2010

W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013

[31] K. Dembczynski, W. Kotlowski, and E. Hüllermeier. Consistent multilabel ranking through univariate losses. In *ICML*, 2012

Reduction to weighted binary relevance

- The Bayes ranker can be obtained by sorting labels according to:

$$\Delta_i^1 = \sum_{\boldsymbol{y} :\, y_i = 1} w(\boldsymbol{y}) P(\boldsymbol{y} \,|\, \boldsymbol{x})\,.$$

- For $w(\boldsymbol{y}) \equiv 1$, Δ_i^u reduces to **marginal probabilities** $P(y_i = u \,|\, \boldsymbol{x})$.
- The solution can be obtained with BR or its weighted variant in a general case.

Reduction to weighted binary relevance

- Consider the sum of **univariate (weighted)** losses:

$$\tilde{\ell}_{\exp}(\boldsymbol{y}, \boldsymbol{f}) = w(\boldsymbol{y}) \sum_{i=1}^{m} e^{-y' f_i} ,$$

$$\tilde{\ell}_{\log}(\boldsymbol{y}, \boldsymbol{f}) = w(\boldsymbol{y}) \sum_{i=1}^{m} \log\left(1 + e^{-y' f_i}\right) .$$

where $y' = 2y_i - 1$.

- The risk minimizer of these losses is:

$$f_i^*(\boldsymbol{x}) = \frac{1}{c} \log \frac{\Delta_i^1}{\Delta_i^0} = \frac{1}{c} \log \frac{\Delta_i^1}{W - \Delta_i^1} ,$$

which is a strictly increasing transformation of Δ_i^1, where

$$W = \mathbb{E}_{\boldsymbol{y}}[w(\boldsymbol{y}) \,|\, \boldsymbol{x}] = \sum_{\boldsymbol{y}} w(\boldsymbol{y}) P(\boldsymbol{y} \,|\, \boldsymbol{x}) .$$

Reduction to weighted binary relevance

- **Vertical reduction**: Solving m independent classification problems.
- Standard algorithms, like AdaBoost and logistic regression, can be adapted to this setting.
- AdaBoost.MH follows this approach for $w = 1$.[32]
- Besides its **simplicity** and **efficiency**, this approach is **consistent** (regret bounds have also been derived).[33]

[32] R. E. Schapire and Y. Singer. BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning*, 39(2/3):135–168, 2000

[33] K. Dembczynski, W. Kotlowski, and E. Hüllermeier. Consistent multilabel ranking through univariate losses. In *ICML*, 2012

Weighted binary relevance

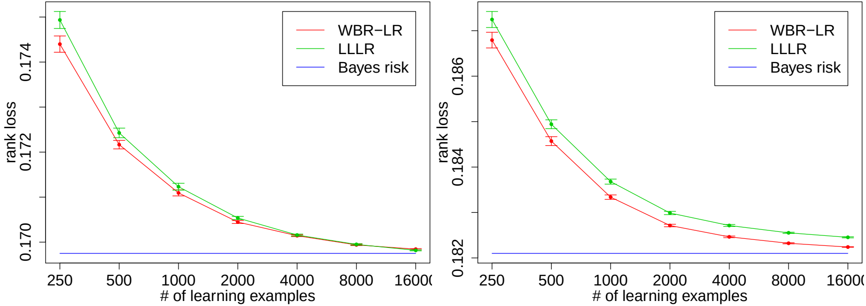


Figure: WBR LR vs. LLLR. Left: independent data. Right: dependent data.

- **Label independence**: the methods perform more or less en par.
- **Label dependence**: WBR shows small but consistent improvements.

Benchmark data

Table: WBR-AdaBoost vs. AdaBoost.MR (left) and WBR-LR vs LLLR (right).

DATASET	AB.MR	WBR-AB	LLLR	WBR-LR
IMAGE	0.2081	0.2041	0.2047	0.2065
EMOTIONS	0.1703	0.1699	0.1743	0.1657
SCENE	0.0720	0.0792	0.0861	0.0793
YEAST	0.2072	0.1820	0.1728	0.1736
MEDIAMILL	0.0665	0.0609	0.0614	0.0472

- WBR is at least competitive to state-of-the-art algorithms defined on pairwise surrogates.

Outline

1. Multi-label classification
2. Simple approaches to multi-label classification
3. Beyond simple approaches
4. Other task losses
5. Rank loss minimization
6. Summary

Summary

- Multi-label classification.
- Simple approaches to multi-label classification.
- Task losses minimized by BR and LP.
- CRFs and SSVMs.
- PCC and Filter trees.
- Approaches for other loss functions: F-measure and rank loss.

Open challenges

- Learning and inference algorithms for any task loss and output structure.
- Consistency of the algorithms.
- Large-scale datasets: number of instances, features, and labels.

Conclusions

- Take-away message:
  - Two main issues: loss minimization and label dependence.
  - Two main approaches: surrogate loss minimization and reduction.
  - Consistency of algorithms.
  - High regret between solutions for different losses.
  - Proper modeling of label dependence for different loss functions.
  - Be careful with empirical evaluations.
  - Independent models can perform quite well.
- For more check:

  `http://www.cs.put.poznan.pl/kdembczynski`