Multi-label Classification

Jesse Read
https://users.ics.aalto.fi/jesse/

Aalto University
Department of Information and Computer Science
Helsinki, Finland

Summer School on Data Sciences for Big Data
September 5, 2015

Multi-label Classification

Binary classification: Is this a picture of the sea?

$$\in \{\texttt{yes}, \texttt{no}\}$$

Multi-label Classification

Multi-*class* classification: What is this a picture of?

$$\in \{\texttt{sea}, \texttt{sunset}, \texttt{trees}, \texttt{people}, \texttt{mountain}, \texttt{urban}\}$$

Multi-label Classification

Multi-label classification: Which labels are relevant to this picture?

$$\subseteq \{\texttt{sea}, \texttt{sunset}, \texttt{trees}, \texttt{people}, \texttt{mountain}, \texttt{urban}\}$$

i.e., multiple labels per instance instead of a single label!

Multi-label Classification

	$K = 2$	$K > 2$
$L = 1$	**binary**	**multi-class**
$L > 1$	**multi-label**	**multi-output**†

† also known as multi-target, multi-dimensional.

Figure: For L target variables (labels), each of K values.

- multi-output can be cast to multi-label, just as multi-class can be cast to binary.
- tagging / keyword assignment: set of labels (L) is not predefined

Increasing Interest

year	in text	in title
1996-2000	23	1
2001-2005	188	18
2006-2010	1470	164
2011-2015	4550	485

Table: Academic articles containing the phrase '*multi-label classification*' (Google Scholar)

Single-label vs. Multi-label

Table: Single-label $Y \in \{0, 1\}$

X_1	X_2	X_3	X_4	X_5	Y
1	0.1	3	1	0	0
0	0.9	1	0	1	1
0	0.0	1	1	0	0
1	0.8	2	0	1	1
1	0.0	2	0	1	0
0	0.0	3	1	1	?

Table: Multi-label $Y \subseteq \{\lambda_1, \ldots, \lambda_L\}$

X_1	X_2	X_3	X_4	X_5	Y
1	0.1	3	1	0	$\{\lambda_2, \lambda_3\}$
0	0.9	1	0	1	$\{\lambda_1\}$
0	0.0	1	1	0	$\{\lambda_2\}$
1	0.8	2	0	1	$\{\lambda_1, \lambda_4\}$
1	0.0	2	0	1	$\{\lambda_4\}$
0	0.0	3	1	1	?

Single-label vs. Multi-label

Table: Single-label $Y \in \{0, 1\}$

X_1	X_2	X_3	X_4	X_5	Y
1	0.1	3	1	0	0
0	0.9	1	0	1	1
0	0.0	1	1	0	0
1	0.8	2	0	1	1
1	0.0	2	0	1	0
0	0.0	3	1	1	?

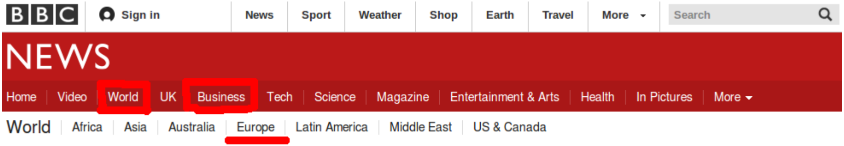
Table: Multi-label $[Y_1, \ldots, Y_L] \in 2^L$

X_1	X_2	X_3	X_4	X_5	Y_1	Y_2	Y_3	Y_4
1	0.1	3	1	0	0	1	1	0
0	0.9	1	0	1	1	0	0	0
0	0.0	1	1	0	0	1	0	0
1	0.8	2	0	1	1	0	0	1
1	0.0	2	0	1	0	0	0	1
0	0.0	3	1	1	?	?	?	?

Outline

Text Categorization

For example, the news . . .



Novo Banco: Portugal bank sell-off hits snag

Portugal's central bank has missed its deadline to sell Novo Banco, a bank created after the collapse of the country's second-biggest lender.

- Reuters collection, newswire stories into 103 topic codes

Text Categorization

For example, the IMDb dataset: Textual movie plot summaries associated with genres (labels).

Text Categorization

For example, the IMDb dataset: Textual movie plot summaries associated with genres (labels).

	abandoned	*accident*	⋮	*violent*	*wedding*	horror	romance	…	comedy	action
i	X_1	X_2	…	X_{1000}	X_{1001}	Y_1	Y_2	…	Y_{27}	Y_{28}
1	1	0	…	0	1	0	1	…	0	0
2	0	1	…	1	0	1	0	…	0	0
3	0	0	…	0	1	0	1	…	0	0
4	1	1	…	0	1	1	0	…	0	1
5	1	1	…	0	1	0	1	…	0	1
⋮	⋮	⋮	⋱	⋮	⋮	⋮	⋮	⋱	⋮	⋮
120919	1	1	…	0	0	0	0	…	0	1

Labelling E-mails

- For example, the *Enron* e-mails multi-labelled to 53 categories by the *UC Berkeley Enron Email Analysis Project*

Company Business, Strategy, etc.
Purely Personal
Empty Message
Forwarded email(s)
. . .
company image – current
. . .
Jokes, humor (related to business)
. . .
Emotional tone: worry / anxiety
Emotional tone: sarcasm
. . .
Emotional tone: shame
Company Business, Strategy, etc.

Labelling Images

Images are labelled to indicate

- multiple concepts
- multiple objects
- multiple people

e.g., Scene data with concept labels
$\subseteq \{\texttt{beach}, \texttt{sunset}, \texttt{foliage}, \texttt{field}, \texttt{mountain}, \texttt{urban}\}$

Applications: Audio

Labelling music/tracks with genres / voices, concepts, etc.

e.g., Music dataset, audio tracks labelled with different moods, among: {

- `amazed-surprised`,
- `happy-pleased`,
- `relaxing-calm`,
- `quiet-still`,
- `sad-lonely`,
- `angry-aggressive`

}

Medical

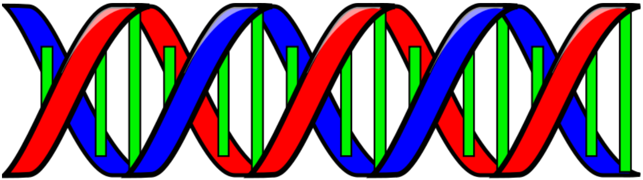
Medical Diagnosis

- medical history, symptoms $\rightarrow$ diseases / ailments

e.g., Medical dataset,

- clinical free text reports by radiologists
- label assignment out of 45 ICD-9-CM codes

Bioinformatics



- Genes are associated with biological functions.
- E.g. the Yeast dataset: 2, 417 genes, described by 103 attributes, labeled into 14 groups of the FunCAt functional catalogue.

Related Tasks

- multi-output[1] classification: outputs are nominal

$$y_j \in \{1, \ldots, K\}, \mathbf{y} \in \mathbb{N}^L$$

- multi-output regression: outputs are real-valued

$$y_j \in \mathbb{R}, \mathbf{y} \in \mathbb{R}^L$$

X_1	X_2	X_3	X_4	X_5	price	age	percent
x_1	x_2	x_3	x_4	x_5	37.00	25	0.88
x_1	x_2	x_3	x_4	x_5	22.88	22	0.22
x_1	x_2	x_3	x_4	x_5	88.23	11	0.77

- label ranking, i.e., preference learning

$$\lambda_3 \succ \lambda_1 \succ \lambda_4 \succ \ldots \succ \lambda_2$$

[1] aka multi-target, multi-dimensional

Related Areas

- multi-task learning: multiple tasks, shared representation, data may come from different sources e.g., learn to recognise speech for different speakers, classify text from different corpora
- sequential learning: predict across time indices instead of across label indices
- structured output prediction: assume particular structure amoung outputs, e.g., pixels

Advanced Applications

Figure: Image Segmentation: Foreground $y_j = 1$

Advanced Applications

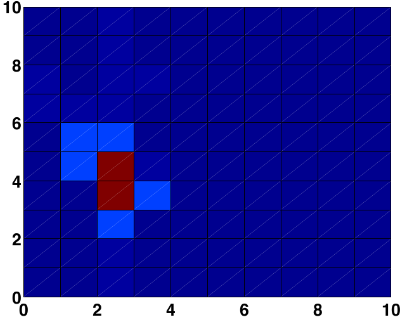


Figure: Localization: $y_j = 1$ if j-th tile occupied.

Advanced Applications

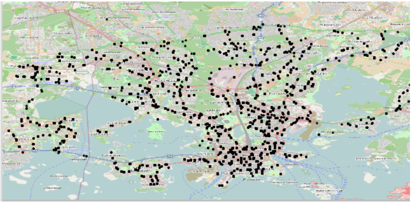
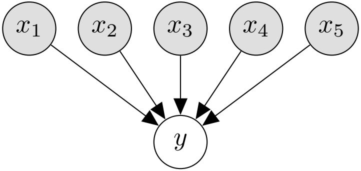


Figure: Demand prediction: $y_j = 1$ if high demand at j-th node.

Outline

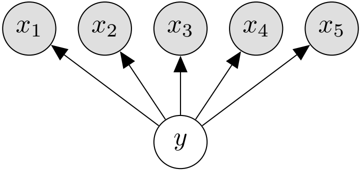
Single-label Classification



$$\hat{y} = h(\mathbf{x})$$ • classifier h

$$= \underset{y \in \{0,1\}}{\operatorname{argmax}}\, p(y|\mathbf{x})$$ • MAP Estimate

Example: Naive Bayes



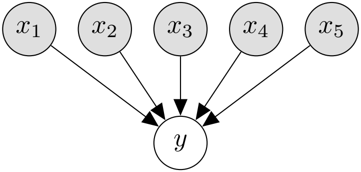
$$\hat{y} = \operatorname*{argmax}_{y \in \{0,1\}} p(y) p(\mathbf{x}|y)$$

- Generative, $p(y|\mathbf{x}) \propto p(\mathbf{x}|y) p(y)$

$$= \operatorname*{argmax}_{y \in \{0,1\}} p(y) \prod_{d=1}^{D} p(x_d|y)$$

- Naive Bayes

Example: Logistic Regression

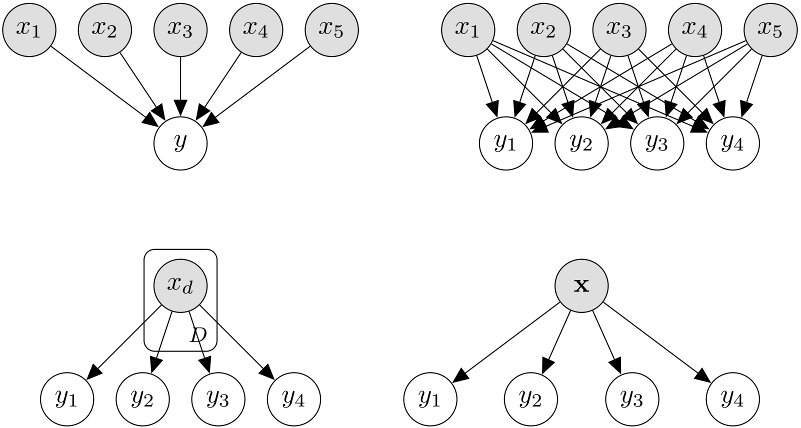


$$\hat{y} = \operatorname*{argmax}_{y \in \{0,1\}} p(y|\mathbf{x})$$ • MAP Estimate

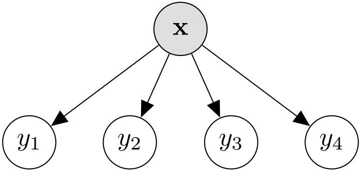
$$p(y = 1|\mathbf{x}) = f_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})}$$ • Logistic Regression

and find $\mathbf{w}$ to minimize $E(\mathbf{w})$

Focus on the Labels



Multi-label Classification



$$\hat{y}_j = h_j(\mathbf{x}) = \operatorname*{argmax}_{y_j \in \{0,1\}} p(y_j|\mathbf{x})$$

- for index, $j = 1, \ldots, L$

and then,

$$\begin{aligned}\hat{\mathbf{y}} = \mathbf{h}(\mathbf{x}) &= [\hat{y}_1, \ldots, \hat{y}_4] \\ &= \left[\operatorname*{argmax}_{y_1 \in \{0,1\}} p(y_1|\mathbf{x}), \cdots, \operatorname*{argmax}_{y_4 \in \{0,1\}} p(y_4|\mathbf{x})\right] \\ &= \left[f_1(\mathbf{x}), \cdots, f_4(\mathbf{x})\right] = f(\mathbf{W}^\top \mathbf{x})\end{aligned}$$

This is the Binary Relevance method (BR).

Outline

BR *Transformation*

1. Transform dataset ...

$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

... into L separate binary problems (one for each label)

$\mathbf{X}$	Y_1
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	Y_2
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(2)}$	0
$\mathbf{x}^{(3)}$	1
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	Y_3
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(2)}$	0
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	Y_4
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	0
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1
$\mathbf{x}^{(5)}$	1

2. and train with any off-the-shelf binary base classifier.

Why Not Binary Relevance?

BR ignores label dependence, i.e.,

$$p(\mathbf{y}|\mathbf{x}) \propto p(\mathbf{x}) \prod_{j=1}^{L} p(y_j|\mathbf{x})$$

which may not always hold!

Example (Film Genre Classification)

$$p(y_{\texttt{romance}}|\mathbf{x}) \neq p(y_{\texttt{romance}}|\mathbf{x}, y_{\texttt{horror}})$$

Why Not Binary Relevance?

BR ignores label dependence, i.e.,

$$p(\mathbf{y}|\mathbf{x}) \propto p(\mathbf{x}) \prod_{j=1}^{L} p(y_j|\mathbf{x})$$

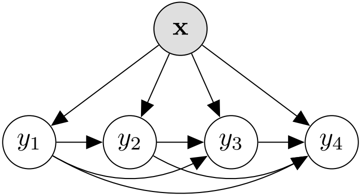
which may not always hold!

Table: Average predictive performance (5 fold CV, EXACT MATCH)

	L	BR	MCC
Music	6	0.30	**0.37**
Scene	6	0.54	**0.68**
Yeast	14	0.14	**0.23**
Genbase	27	0.94	**0.96**
Medical	45	0.58	**0.62**
Enron	53	0.07	**0.09**
Reuters	101	0.29	**0.37**

Classifier Chains

Modelling label dependence,



$$p(\mathbf{y}|\mathbf{x}) \propto p(\mathbf{x}) \prod_{j=1}^{L} p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1})$$

and,

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y} \in \{0,1\}^L} p(\mathbf{y}|\mathbf{x})$$

CC Transformation

Similar to BR: make L binary problems, but include previous predictions as feature attributes,

$\mathbf{X}$	Y_1
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1
$\mathbf{x}^{(5)}$	0

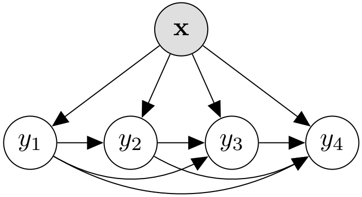
$\mathbf{X}$	Y_1	Y_2
$\mathbf{x}^{(1)}$	0	1
$\mathbf{x}^{(2)}$	1	0
$\mathbf{x}^{(3)}$	0	1
$\mathbf{x}^{(4)}$	1	0
$\mathbf{x}^{(5)}$	0	0

$\mathbf{X}$	Y_1	Y_2	Y_3
$\mathbf{x}^{(1)}$	0	1	1
$\mathbf{x}^{(2)}$	1	0	0
$\mathbf{x}^{(3)}$	0	1	0
$\mathbf{x}^{(4)}$	1	0	0
$\mathbf{x}^{(5)}$	0	0	0

$\mathbf{X}$	Y_1	Y_3	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

and, again, apply any classifier (not necessarily a probabilistic one)!

Greedy CC



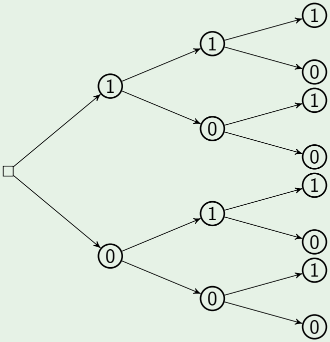
L classifiers for L labels. For test instance $\tilde{\mathbf{x}}$, classify [22],

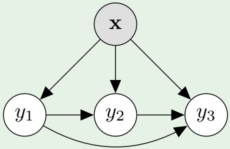
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}})$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1)$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2)$
4. $\hat{y}_4 = h_4(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2, \hat{y}_3)$

and return

$$\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_L]$$

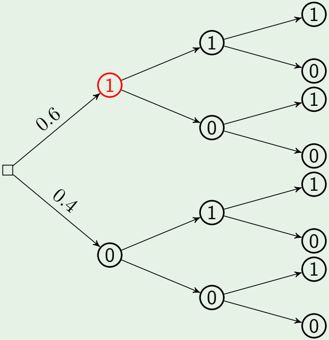
Example

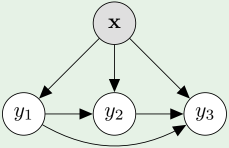




$$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [?, ?, ?]$$

Example

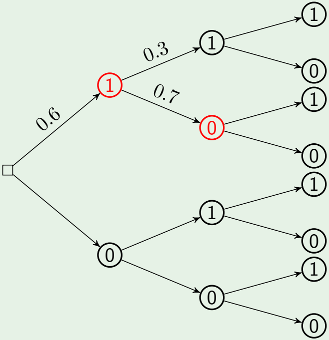


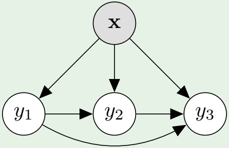


1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \operatorname{argmax}_{y_1} p(y_1|\tilde{\mathbf{x}}) = 1$

$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, ?, ?]$

Example

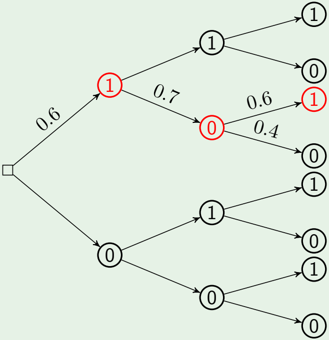




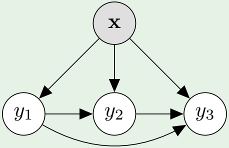
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \operatorname{argmax}_{y_1} p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1) = \ldots = 0$

$$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, \mathbf{0}, ?]$$

Example

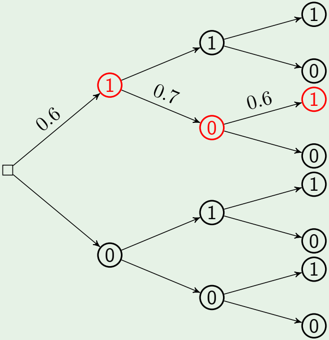


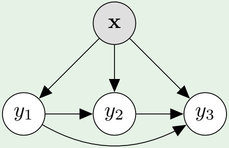
$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, 0, 1]$



1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \operatorname{argmax}_{y_1} p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1) = \ldots = 0$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2) = \ldots = 1$

Example





1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \text{argmax}_{y_1} p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1) = \ldots = 0$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2) = \ldots = 1$

$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, 0, 1]$

- Improves over BR; similar build time (if $L < D$);
- able to use any off-the-shelf classifier for h_j; parralelizable
- But, errors may be propagated down the chain

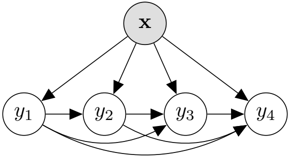
Bayes Optimal CC

Bayes-optimal Probabilistic CC [4] (PCC)

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}\in\{0,1\}^L} p(\mathbf{y}|\mathbf{x})$$

$$= \operatorname*{argmax}_{\mathbf{y}\in\{0,1\}^L} \left\{ p(y_1|\mathbf{x}) \prod_{j=2}^{L} p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1}) \right\}$$

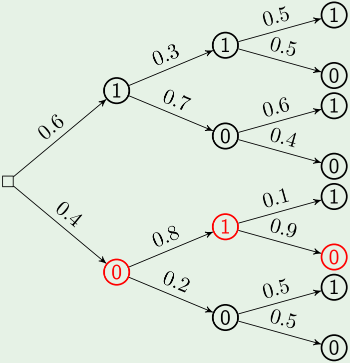
• chain rule



Test all possible paths ($\mathbf{y} = [y_1, \ldots, y_L] \in 2^L$ in total)

Bayes Optimal CC

Example



1. $p(\mathbf{y} = [0, 0, 0]) = 0.040$
2. $p(\mathbf{y} = [0, 0, 1]) = 0.040$
3. $p(\mathbf{y} = [0, 1, 0]) = 0.288$
4. $\ldots$
6. $p(\mathbf{y} = [1, 0, 1]) = 0.252$
7. $\ldots$
8. $p(\mathbf{y} = [1, 1, 1]) = 0.090$

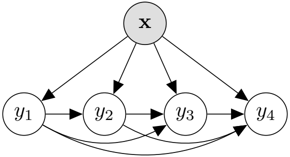
return $\text{argmax}_{\mathbf{y}}\, p(\mathbf{y}|\tilde{\mathbf{x}})$

- Better accuracy than greedy CC but computationally limited to $L \lesssim 15$

Monte-Carlo search for CC

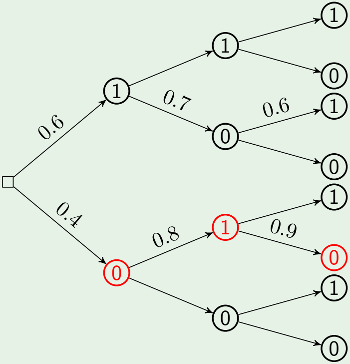
1. For $t = 1, \ldots, T$ iterations:
   - Sample $\boxed{\mathbf{y}_t \sim p(\mathbf{y}|\mathbf{x})}$ the chain [20]
     1. $y_1 \sim p(y_1|\mathbf{x})$ $//y_1 = 1$ with probability $p(y_1|\mathbf{x})$
     2. $y_2 \sim p(y_2|\mathbf{x}, y_1, y_2)$
     3. $\ldots$
     4. $y_L \sim p(y_L|\mathbf{x}, y_1, \ldots, y_{L-1})$
2. Predict

$$\hat{\mathbf{y}} = \underset{\mathbf{y}_t \in \{\mathbf{y}_1, \ldots, \mathbf{y}_T\}}{\operatorname{argmax}} p(\mathbf{y}_t|\mathbf{x})$$



Monte-Carlo search for CC

Example



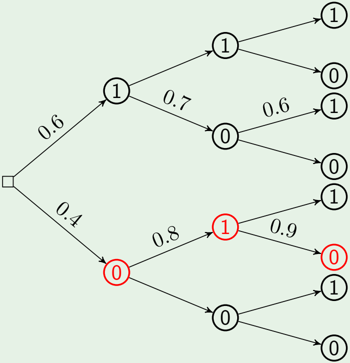
Sample T times . . .

- $p([1, 0, 1]) = 0.6 \cdot 0.7 \cdot 0.6 = 0.252$
- $p([0, 1, 0]) = 0.4 \cdot 0.8 \cdot 0.9 = 0.288$

return $\text{argmax}_{\mathbf{y}_t}\ p(\mathbf{y}_t|\mathbf{x})$

Monte-Carlo search for CC

Example



Sample T times . . .

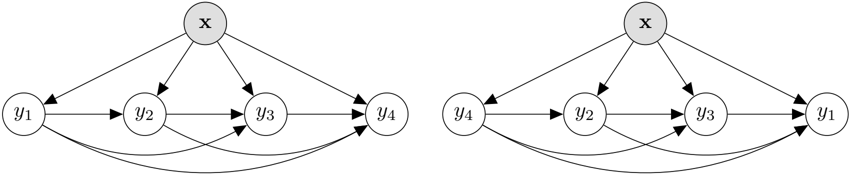
- $p([1, 0, 1]) = 0.6 \cdot 0.7 \cdot 0.6 = 0.252$
- $p([0, 1, 0]) = 0.4 \cdot 0.8 \cdot 0.9 = 0.288$

return $\mathrm{argmax}_{\mathbf{y}_t}\ p(\mathbf{y}_t|\mathbf{x})$

- Tractable, with similar accuracy to (Bayes Optimal) PCC
- Can use other search algorithms, e.g., beam search [13]

Does Label-*order* Matter?

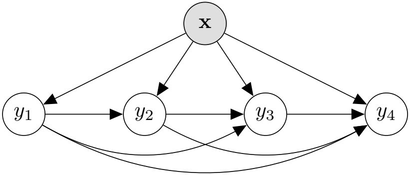
Are these models equivalent?



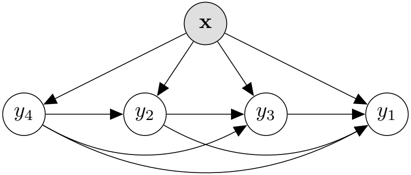
vs

Does Label-*order* Matter?

Are these models equivalent?



vs



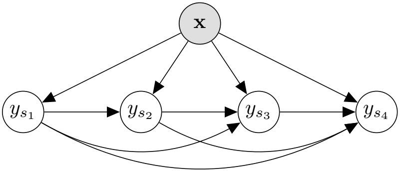
$$p(\mathbf{x}, \mathbf{y}) = p(y_1|\mathbf{x})p(y_2|y_1, \mathbf{x}) = p(y_2|\mathbf{x})p(y_1|y_2, \mathbf{x})$$

but we are estimating p from finite and noisy data (and possibly doing a greedy search); thus

$$\hat{p}(y_1|\mathbf{x})\hat{p}(y_2|\hat{y}_1, \mathbf{x}) \neq \hat{p}(y_2|\mathbf{x})\hat{p}(y_1|\hat{y}_2, \mathbf{x})$$

Searching the Chain Space

Can search the space of possible chain orderings [20] with, e.g., Monte Carlo walk



For $u = 1, \ldots, U$:

1. propose $\boldsymbol{s}_u = [s_1, \ldots, s_L] = \text{permute}([1, \ldots, L])$
2. build model on sequence $\boldsymbol{s}_u$
3. evaluate; accept if better (if $\mathcal{J}(\boldsymbol{s}_u) > \mathcal{J}(\boldsymbol{s}_{u-1})$)

Use $\mathbf{h}_{\boldsymbol{s}_U}$ as the final model.

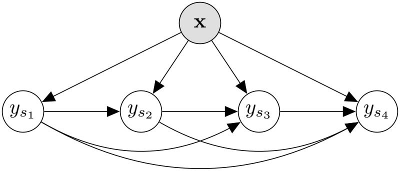
Example

Scene data

u	$\boldsymbol{s}_u = [s_1, \ldots, s_L]$	$\mathcal{J}(\boldsymbol{s}_u)$
0	[4, 2, 0, 1, 3, 5]	0.623
1	[4, 2, 0, 3, 1, 5]	0.628
2	[4, 2, 0, 3, 5, 1]	0.638
3	[4, 0, 2, 3, 5, 1]	0.647
5	[4, 0, 5, 2, 3, 1]	0.653
18	[5, 1, 4, 3, 2, 0]	0.654
23	[5, 4, 0, 1, 2, 3]	0.664
128	[3, 5, 1, 0, 2, 4]	0.668
176	[5, 3, 1, 0, 4, 2]	0.669
225	[5, 3, 1, 4, 0, 2]	0.670

$\mathcal{J}(\boldsymbol{s}) := \text{EXACTMATCH}(\mathbf{Y}, \mathbf{h}_{\boldsymbol{s}}(\mathbf{X}))$ (higher is better)

Searching the Chain Space



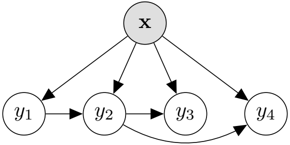
- The space is of combinational proportions, ... but a little search can go a long way.
- Many other options:
  - add temperature to freeze $\boldsymbol{s}_u$ from left to right over time
  - use a population of chain sequences: $\boldsymbol{s}_u^{(1)}, \ldots, \boldsymbol{s}_u^{(M)}$
  - use beam search

Chain Structure

We can formulate any structure,

$$y_j = h_j(\mathbf{x}, \mathsf{pa}(y_j))$$

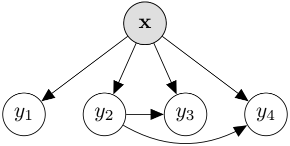
where $\mathsf{pa}(y_j)$ = parents of node j.



- If $\mathsf{pa}(y_j) := \{y_1, \ldots, y_{j-1}\}$ we recover CC
- 'partial' models are more efficient and interpretable

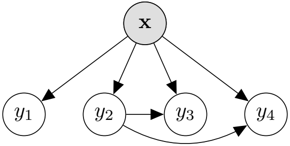
Structured Classifiers Chains

1. Measure some heuristic
   - marginal dependence [30]
   - conditional dependence [31]
2. Find a structure
3. Plug in base classifiers and run some CC inference



Structured Classifiers Chains

1. Measure some heuristic
   - marginal dependence [30]
   - conditional dependence [31]
2. Find a structure
3. Plug in base classifiers and run some CC inference



Related to Bayesian networks, [1, 2]:

$$p(\mathbf{y}, \tilde{\mathbf{x}}) = \prod_{j=1}^{L} p(y_j | \mathsf{pa}(y_j), \tilde{\mathbf{x}})$$

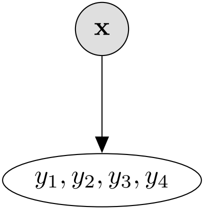
Label Powerset (LP)

One multi-class problem (taking many values),

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}\in\{0,1\}^L} p(\mathbf{x}) \prod_{j=1}^{L} p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1}) \quad \bullet \text{ PCC}$$

$$= \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} p(\mathbf{y}|\mathbf{x}) \quad \bullet \text{ LP, where } \mathcal{Y} \subset \{0,1\}^L$$

$$\equiv \operatorname*{argmax}_{y\in\{0,\ldots,2^L-1\}} p(y|\mathbf{x}) \quad \bullet \text{ a multi-class problem!}$$



Label Powerset (LP)

One multi-class problem (taking many values),

$$\hat{\mathbf{y}} = \underset{\mathbf{y}\in\{0,1\}^L}{\operatorname{argmax}}\, p(\mathbf{x}) \prod_{j=1}^{L} p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1}) \quad \bullet \text{ PCC}$$

$$= \underset{\mathbf{y}\in\mathcal{Y}}{\operatorname{argmax}}\, p(\mathbf{y}|\mathbf{x}) \quad \bullet \text{ LP, where } \mathcal{Y} \subset \{0,1\}^L$$

$$\equiv \underset{y\in\{0,\ldots,2^L-1\}}{\operatorname{argmax}}\, p(y|\mathbf{x}) \quad \bullet \text{ a multi-class problem!}$$

- Each value is a label vector,
- typically, the occurrences of the training set.
- In practice, $|\mathcal{Y}| \leq N$, and $|\mathcal{Y}| \ll 2^L$

Label Powerset Method (LP)

1. Transform dataset . . .

$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	1	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

. . . into a multi-*class* problem, taking 2^L possible values:

$\mathbf{X}$	$Y \in 2^L$
$\mathbf{x}^{(1)}$	0110
$\mathbf{x}^{(2)}$	1000
$\mathbf{x}^{(3)}$	0110
$\mathbf{x}^{(4)}$	1001
$\mathbf{x}^{(5)}$	0001

2. . . . and train any off-the-shelf multi-*class* classifier.

Issues with LP

- complexity: there is no greedy label-by-label option
- imbalance: few examples per class label
- overfitting: how to predict new value?

Example

In the Enron dataset, 44% of labelsets are unique (a single training example or test instance). In del.icio.us dataset, 98% are unique.

RAkEL

$\mathbf{X}$	$Y \in 2^L$
$\mathbf{x}^{(1)}$	0110
$\mathbf{x}^{(2)}$	1000
$\mathbf{x}^{(3)}$	0110
$\mathbf{x}^{(4)}$	1001
$\mathbf{x}^{(5)}$	0001

Ensembles of RAndom k-labEL subsets (`RA`k`EL`) [27]

- Do LP on M subsets $\subset \{1, \ldots, L\}$ of size k

$\mathbf{X}$	$Y_{123} \in 2^k$
$\mathbf{x}^{(1)}$	011
$\mathbf{x}^{(2)}$	100
$\mathbf{x}^{(3)}$	011
$\mathbf{x}^{(4)}$	100
$\mathbf{x}^{(5)}$	000

$\mathbf{X}$	$Y_{124} \in 2^k$
$\mathbf{x}^{(1)}$	010
$\mathbf{x}^{(2)}$	100
$\mathbf{x}^{(3)}$	010
$\mathbf{x}^{(4)}$	101
$\mathbf{x}^{(5)}$	001

$\mathbf{X}$	$Y_{234} \in 2^k$
$\mathbf{x}^{(1)}$	110
$\mathbf{x}^{(2)}$	000
$\mathbf{x}^{(3)}$	110
$\mathbf{x}^{(4)}$	001
$\mathbf{x}^{(5)}$	001

Pruned Sets

$\mathbf{X}$	$Y \in 2^L$
$\mathbf{x}^{(1)}$	0110
$\mathbf{x}^{(2)}$	1000
$\mathbf{x}^{(3)}$	0110
$\mathbf{x}^{(4)}$	1001
$\mathbf{x}^{(5)}$	0001

Ensembles of Pruned label Sets (EPS) [21]

- Do LP on M *pruned* subsets (wrt class *values*)
- Can flip bits to reduce ratio of classes to examples

$\mathbf{X}$	$Y \in 2^L$
$\mathbf{x}^{(1)}$	0110
$\mathbf{x}^{(3)}$	0110
$\mathbf{x}^{(4)}$	0001
$\mathbf{x}^{(5)}$	0001

$\mathbf{X}$	$Y \in 2^L$
$\mathbf{x}^{(1)}$	0110
$\mathbf{x}^{(2)}$	1000
$\mathbf{x}^{(3)}$	0110
$\mathbf{x}^{(4)}$	0001
$\mathbf{x}^{(4)}$	1000

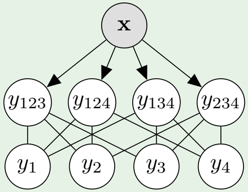
$\mathbf{X}$	$Y \in 2^L$
$\mathbf{x}^{(1)}$	0110
$\mathbf{x}^{(2)}$	1000
$\mathbf{x}^{(3)}$	0110
$\mathbf{x}^{(4)}$	1000
$\mathbf{x}^{(5)}$	0001

Ensemble-based Voting

Most problem-transformation methods are ensemble-based, e.g., ECC, EPS, RA*k*EL.

Ensemble Voting

	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$	$\hat{y}_4$
$\mathbf{h}^1(\tilde{\mathbf{x}})$	1	1	1	
$\mathbf{h}^2(\tilde{\mathbf{x}})$		0	1	0
$\mathbf{h}^3(\tilde{\mathbf{x}})$	1		0	0
$\mathbf{h}^4(\tilde{\mathbf{x}})$	1	0		0
score	0.75	0.25	0.75	0
$\hat{\mathbf{y}}$	1	0	1	0



- more predictive power (ensemble effect)
- LP can predict novel label combinations

Scaling Up

LSHTC4: Large Scale Hierarchal Text Classification

A wikipedia-scale problem

- $325,056$ labels
- $2.4M$ examples

- Even with only $1,000$ features, have to learn over $300M$ parameters with BR (linear models)
- ... plus $52,831M$ more with CC
- ... plus ensembles ($\times 10$, $\times 50$?)
- LP transformation generates around $1.47M$ classes

Scaling Up

Our approach [16, 23]:

1. Ignore the predefined hierarchy
2. work with subsets of the labelset (RAkEL)
3. prune them (pruned sets)
4. chain these sets together (classifier chains)
5. mix of base classifiers (centroid, decision trees, SVMs)
6. ensemble with sample features and instances (random subspace)
7. randomization: splits, pruning, reintroduction, chain links, base classifier parameters
8. train models in parallel, weight according to score on hold-out sets (avoid overfitting!)

Pairwise Multi-label Classification

$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

- Create a pairwise transformation, of up to $\frac{L(L-1)}{2}$ binary classifiers (*all-vs-all*), but smaller than in BR

$\mathbf{X}$	Y_{1v2}
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1

$\mathbf{X}$	Y_{1v3}
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(4)}$	1

$\mathbf{X}$	Y_{1v4}
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(5)}$	0

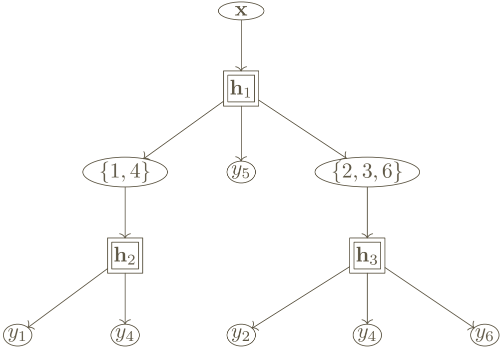
$\mathbf{X}$	Y_{2v3}
$\mathbf{x}^{(3)}$	1

$\mathbf{X}$	Y_{2v4}
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(3)}$	1
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	Y_{3v4}
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

- Ensemble voting, or calibrated label ranking [7]
- Can also model four classes (related to LP)

Hierarchy of MLC (HOMER)



1. Cluster labels (randomly, *k*-means) [28], or use pre-defined hierarchy
2. Apply problem transformation

Multi-label Regularization

Regularization

$$\hat{\mathbf{y}} = \mathbf{b}(\mathbf{h}(\mathbf{x})), \text{ or } \hat{\mathbf{y}} = \mathbf{b}(\mathbf{h}(\mathbf{x}), \mathbf{x})$$

where

- $\tilde{\mathbf{y}} = \mathbf{h}(\mathbf{x})$ is an initial classification; and
- $\mathbf{b}$ is some regularizer

Examples:

- Meta BR: A second (meta) BR ($\mathbf{b}$) takes as input the output from an initial BR ($\mathbf{h}$) [9]
- Error Correcting Output Codes: bit vector $\tilde{\mathbf{y}}$ has been distorted by noise; attempt to correct it [6]
- Subset matching: if $\tilde{\mathbf{y}}$ does not exist in training set, match it to the closest one that does

Problem Transformation Summary

Two ways of viewing a multi-label problem of L labels:

1. L binary problems (BR),
2. a multi-class problem with 2^L classes (LP)

or a combination of these.

General method:

1. Transform data into subproblems (binary or multi-class)
2. Apply some off-the-shelf base classifier
3. (*Optional*) Regularize
4. (*Optional*) Ensemble

Outline

Algorithm Adaptation

1. Take your favourite (most suitable) classifier
2. Modify it for multi-label classification

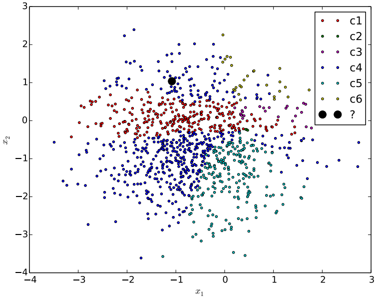
- Advantage: a single model, usually very scalable
- Disadvantage: predictive performance depends on the problem domain

k Nearest Neighbours (kNN)

Assign to $\tilde{\mathbf{x}}$ the majority class of the k 'nearest neighbours'

$$\hat{y} = \underset{y}{\operatorname{argmax}} \sum_{i \in N_k} y^{(i)}$$

where N_k contains the training pairs with $\mathbf{x}^{(i)}$ closest to $\tilde{\mathbf{x}}$.

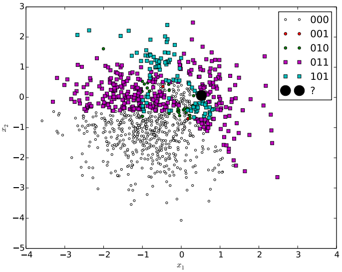


Multi-label *k*NN

Assigns the most common *labels* of the k nearest neighbours

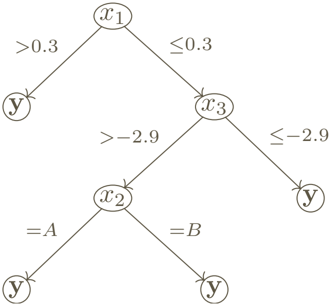
$$p(y_j = 1|\mathbf{x}) = \frac{1}{k} \sum_{i \in N_k} y_j^{(i)}$$

$$\hat{y}_j = \underset{y_j \in \{0,1\}}{\operatorname{argmax}} [p(y_j|\mathbf{x}) > 0.5]$$



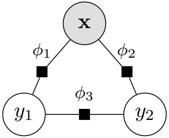
For example, [32]. Related to ensemble voting.

Decision Trees



- construct like `C4.5` (multi-label entropy [3])
- multiple labels at the leaves
- predictive clustering trees [12] are highly competitive in an random forest/ensemble

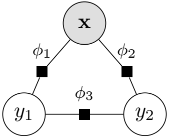
Conditional Random Fields



$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_c \phi_c(\mathbf{x}, \mathbf{y})$$
$$= \frac{1}{Z(\mathbf{x})} \exp\{\sum_c w_c f_c(\mathbf{x}, \mathbf{y})\}$$

where, e.g., $\phi_3(\mathbf{x}, \mathbf{y}) = \phi_3(y_1, y_2) \propto p(y_2|y_1)$. Factors can be modelled with, e.g., with a problem transformation

Conditional Random Fields



→

$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

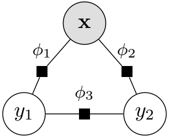
$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

where, e.g., $\phi_3(\mathbf{x}, \mathbf{y}) = \phi_3(y_1, y_2) \propto p(y_2|y_1)$. Factors can be modelled with, e.g., with a problem transformation

Conditional Random Fields

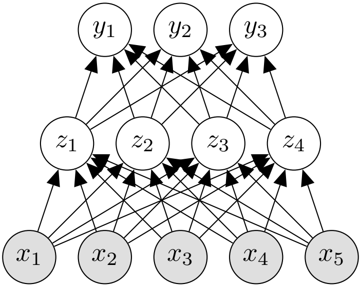


where, e.g., $\phi_3(\mathbf{x}, \mathbf{y}) = \phi_3(y_1, y_2) \propto p(y_2|y_1)$. Factors can be modelled with, e.g., with a problem transformation, but computational burden is shifted to inference, e.g.,

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \{0,1\}^L}{\operatorname{argmax}} f_1(\mathbf{x}, y_1) f_2(\mathbf{x}, y_2) f_3(y_2, y_1)$$

- Gibbs simpling [10] (like an undirected PCC)
- Supported combinations [8] (i.e., $\mathcal{Y}$ in LP)

Neural Network



- Just include an output node for each label.
- train with, e.g., gradient descent + error back-propagation

Other Algorithm Adaptations

- Max-margin methods / SVMs [29]
- Association rules [25]
- Boosting [24]
- Generative (Bayesian) [15]

Outline

Label Dependence in MLC

Common approach: Present methods to

1. measure label dependence
2. find a structure that best represents this

and then apply classifiers, compare results to BR.

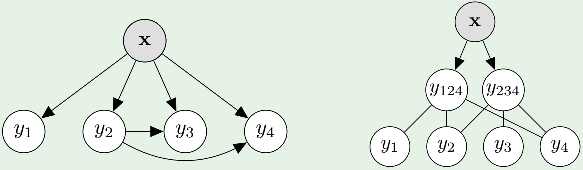
Label Dependence in MLC

Common approach: Present methods to

1. measure **label dependence**
2. find a **structure** that best represents this

and then apply classifiers, compare results to BR.

Example



- Which labels (nodes) to link together? (CC, PGMs)
- Which subsets to form from the labelset? (RA*k*EL)

Label Dependence in MLC

Common approach: Present methods to

1. measure label dependence
2. find a structure that best represents this

and then apply classifiers, compare results to BR.

Problem

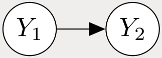
Accuracy often indistinguishable to that of random ensembles, or slow! (although, may be more compact and/or interpretable)

Marginal label dependence

Marginal dependence

When the joint is **not** the product of the marginals, i.e.,

$$p(y_2) \neq p(y_2|y_1)$$
$$p(y_1)p(y_2) \neq p(y_1, y_2)$$



- Estimate from co-occurrence frequencies in training data

Marginal label dependence

Example

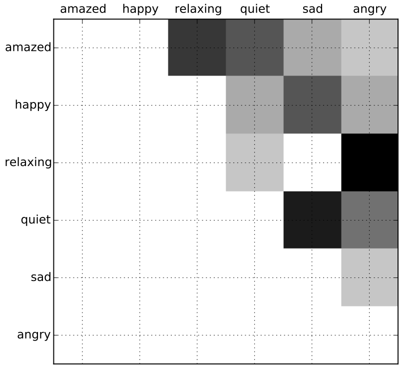


Figure: Music dataset - Mutual Information

Marginal label dependence

Example

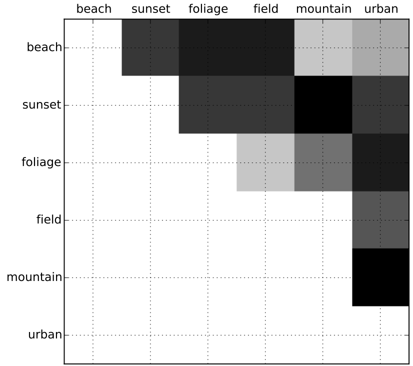


Figure: Scene dataset - Mutual Information

Exploiting marginal dependence

A Toy Dataset

X_1	X_2	Y_1	Y_2	Y_3
0	0	0	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	0

Measure marginal label dependence (i.e., do labels co-occur frequently, or does one exclude the other?).

Exploiting marginal dependence

A Toy Dataset

X_1	X_2	Y_1	Y_2	Y_3
0	0	0	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	0

Measure marginal label dependence (i.e., do labels co-occur frequently, or does one exclude the other?).

- But all labels are interdependent! For example,

$$\hat{p}(y_2 = 1 | y_1 = 1) \neq \hat{p}(y_2 = 1)$$
$$1/3 > 1/4$$

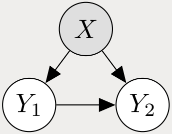
- Could use a threshold, or statistical significance, ...
- But how does this relate to classification, $p(y_j|\mathbf{x})$?

Conditional label dependence

Conditional dependence

...conditioned on input observation **x**.

$$p(y_2|y_1, \mathbf{x}) \neq p(y_2|\mathbf{x})$$



- Have to build and measure models

Indication of conditional dependence if

- the performance of LP/CC exeeds that of BR
- errors among the binary models are correlated

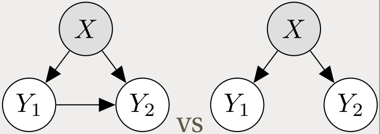
Conditional label dependence

Conditional *in*dependence

...conditioned on input observation $\mathbf{x}$.

$$p(y_2) \neq p(y_2|y_1)$$

, but $p(y_2|\mathbf{x}) = p(y_2|, y_1, \mathbf{x})$



- Have to build and measure models

Indication of conditional dependence if

- the performance of LP/CC exeeds that of BR
- errors among the binary models are correlated

Exploiting conditional dependence

A Toy Dataset

X_1	X_2	Y_1	Y_2	Y_3
0	0	0	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	0

Measure *conditional* label dependence (build models, measure the difference in error rate).

Exploiting conditional dependence

A Toy Dataset

X_1	X_2	Y_1	Y_2	Y_3
0	0	0	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	0

Measure *conditional* label dependence (build models, measure the difference in error rate).

- But building models is expensive!
- Which structure to construct?

Exploiting conditional dependence

A Toy Dataset

X_1	X_2	Y_1 (OR)	Y_2 (AND)	Y_3 (XOR)
0	0	0	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	0

Oracle: complete conditional independence!

Complete conditional independence,

$$p(Y_j|Y_k, X_1, X_2) = p(Y_j|X_1, X_2), \forall j, k : 0 < j < k \leq L$$

Then the binary relevance (BR) classifier should suffice?

The LOGICAL Problem

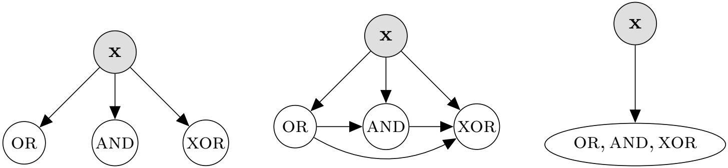


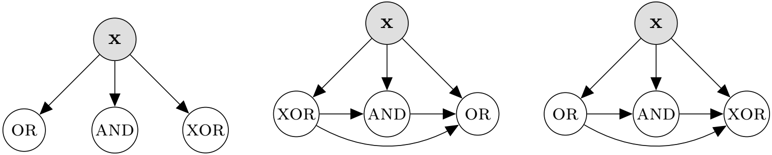
Figure: BR (left), CC (middle), LP (right)

Table: The LOGICAL problem, base classifier logistic regression.

Metric	BR	CC	LP
HAMMING SCORE	0.83	1.00	1.00
EXACT MATCH	0.50	1.00	1.00

- Why didn't BR work?

XOR Solution



- Only one of these works (with greedy inference)!
- The ground truth (oracle) is

$$p(y_{\text{XOR}} | y_{\text{AND}}, \mathbf{x}) = p(y_{\text{XOR}} | \mathbf{x})$$

but, recall: we have an *estimation* of this,

$$\hat{f}(y_{\text{XOR}} | y_{\text{AND}}, \mathbf{x}) \neq \hat{f}(y_{\text{XOR}} | \mathbf{x})$$

(finite data, finite training time, limited class of model $\hat{f}$, i.e., linear): dependence depends on the model!

Solutions

1. Use a suitable *structure*
2. Use a suitable *base classifier*
3. Ensure that labels are conditionally *independent*.

Solutions

1. Use a suitable *structure* How to find it?
2. Use a suitable *base classifier* Which one is suitable?
3. Ensure that labels are conditionally *independent*. How to do that?

Main limiting factor: computational complexity.

The LOGICAL Problem

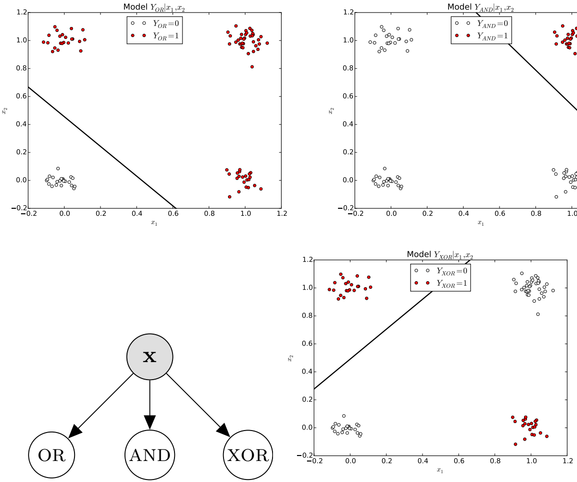


Figure: Binary Relevance (BR): linear decision boundary (solid line, estimated with logistic regression) not viable for Y_{XOR} label

Solution via Structure

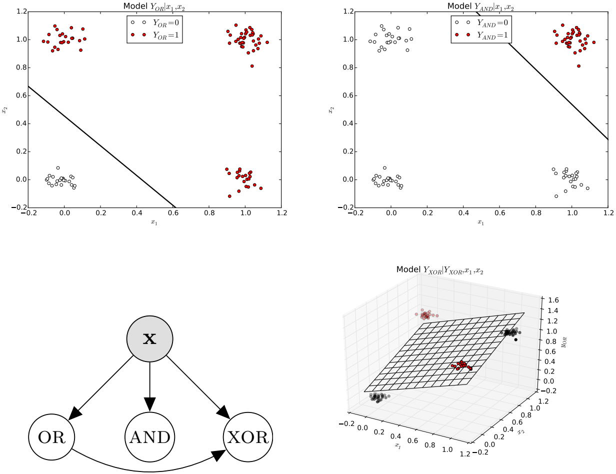


Figure: Solution via structure: linear model now applicable to Y_{XOR}

Solution via Structure

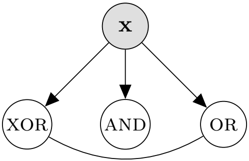
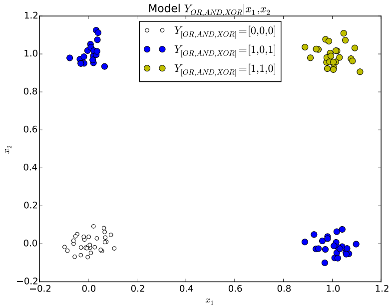


Figure: Solution via structure: two labels have augmented decision space

Can also use undirected connections

- directionality not an issue,
- but implies greater computational burden (≈ LP)
- ... possibly shifted to inference (≈ PCC, CDN)

Solution via Multi-class Decomposition



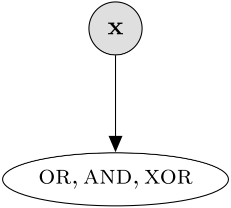
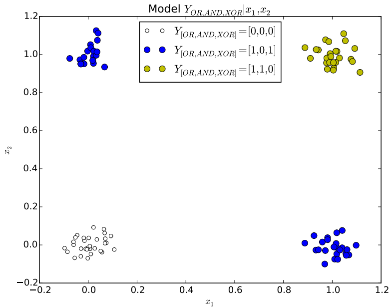


Figure: Label Powerset (LP): solvable with one-vs-one multi-class decomposition for any (e.g., linear) base classifier

Solution via Multi-class Decomposition



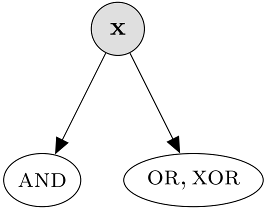


Figure: Label Powerset (LP): solvable with one-vs-one multi-class decomposition for any (e.g., linear) base classifier. Also possible with RA*k*EL subsets $Y_{\mathrm{OR,XOR}}$ and Y_{AND}

Solution via Con. Independence

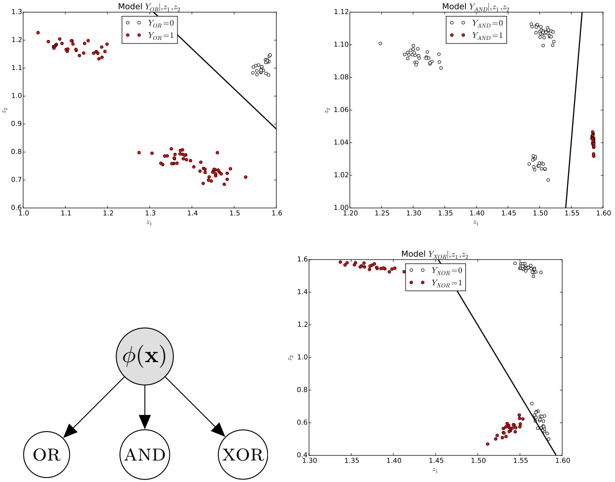


Figure: Solution via non-linear (e.g., random RBF) transformation on input to new space $\mathbf{z}$ (creating independence).

Solution via Suitable Base-classifier

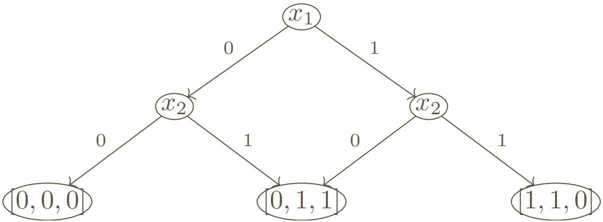


Figure: Solution via non-linear classifier (e.g., Decision Tree). Leaves hold examples, where $\mathbf{y} = [y_{\text{OR}}, y_{\text{AND}}, y_{\text{XOR}}]$

On Real World Problems . . .

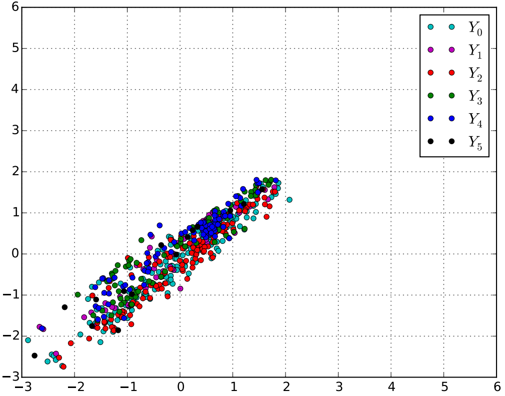
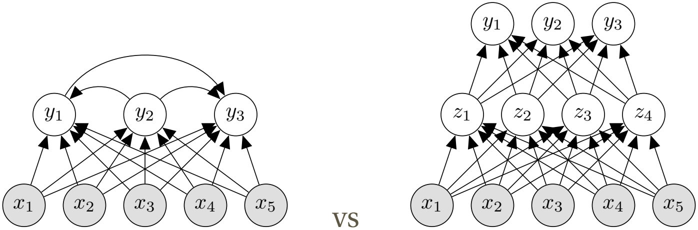


Figure: Music dataset, kernel PCA

Latent Variables

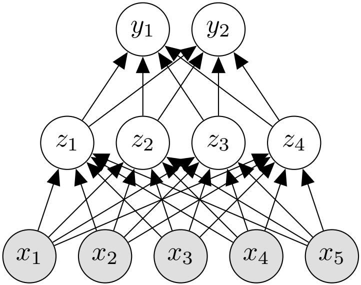
$$p(\mathbf{y}|\mathbf{x}) = \frac{\sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{y}, \mathbf{z})}{p(\mathbf{x})}$$
$$= \frac{1}{Z} \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{y}, \mathbf{z})$$



- Can view label dependencies as having marginalized out latent variables

Inner Layer Methods

1. Use an inner layer
   $\mathbf{z} = \mathbf{f}(\mathbf{x}), \mathbf{z} \in \mathbb{R}^H$
2. Apply a classifier
   $\mathbf{y} = \mathbf{h}(\mathbf{z})$



- PCA, CCA [17]
- Kernel PCA [29]
- Mixture models [15]
- Clustering [28]
- Compressive Sensing [11]
- Deep Learning [18]
- Auto Encoders

Another look: Problem Transformation

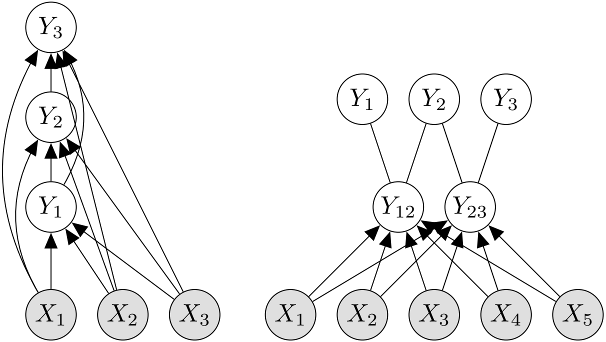


Figure: Methods CC and RA*k*EL (among others) can be viewed as using an inner layer [18].

What about marginal dependence?

- Can be seen as a kind of constraint
- used for regularization
  (recall: e.g., ECOC, subset matching)

Label Dependence: Summary

- Marginal dependence for regularization
- Conditional dependence
  - ... depends on the model
  - ... may be introduced
- Should consider together:
  - base classifier
  - structure
  - inner layer
- An open problem
- Much existing research is relevant

Outline

Multi-label Evaluation

In single-label classification, simply compare true label y with predicted label $\hat{y}$ [or $p(y|\tilde{\mathbf{x}})$].What about in multi-label classification?

Example

If true label vector is $\mathbf{y} = [1, 0, 0, 0]$, then $\hat{\mathbf{y}} = ?$

urban	mountain	beach	foliage
1	0	0	0
1	1	0	0
0	0	0	0
0	1	1	1

- compare bit-wise? too lenient?
- compare vector-wise? too strict?

Hamming Loss

Example

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0 1 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0 1 0 0]
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1 0 0 1]

$$\text{Hamming loss} = \frac{1}{NL} \sum_{i=1}^{N} \sum_{j=1}^{L} \mathbb{I}[\hat{y}_j^{(i)} \neq y_j^{(i)}]$$

$$= 0.20$$

0/1 Loss

Example

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0 1 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0 1 0 0]
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1 0 0 1]

$$\text{0/1 LOSS} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(\hat{\mathbf{y}}^{(i)} \neq \mathbf{y}^{(i)})$$
$$= 0.60$$

Other Metrics

- JACCARD INDEX – often called multi-label ACCURACY
- RANK LOSS – average fraction of pairs not correctly ordered
- ONE ERROR – if top ranked label is not in set of true labels
- COVERAGE – average “depth” to cover all true labels
- LOG LOSS – i.e., cross entropy
- PRECISION – predicted positive labels that are relevant
- RECALL – relevant labels which were predicted
- PRECISION vs. RECALL curves
- F-MEASURE
    - *micro-averaged* (‘global’ view)
    - *macro-averaged* by label (ordinary averaging of a binary measure, changes in infrequent labels have a big impact)
    - *macro-averaged* by example (one example at a time, average across examples)

For general evaluation, ***use multiple and contrasting evaluation measures!***

Hamming loss vs. 0/1 loss

Hamming loss

- *evaluation by example*, suitable for evaluating

$$\hat{y}_j = \operatorname*{argmax}_{y_j \in \{0,1\}} p(y_j|\mathbf{x})$$

  i.e., BR
- favours sparse labelling
- does not benefit directly from modelling label dependence

0/1 loss

- *evaluation by label*, suitable for evaluating

$$\mathbf{y} = \operatorname*{argmax}_{\mathbf{y} \in \{0,1\}^L} p(\mathbf{y}|\mathbf{x})$$

  i.e., PCC, LP
- does not favour sparse labelling
- benefits from models of label dependence

Hamming loss vs. 0/1 loss

Example: 0/1 loss vs. Hamming loss

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(2)}$	[1 0 0 1]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[0 1 1 0]	[0 1 0 0]
$\tilde{\mathbf{x}}^{(4)}$	[1 0 0 0]	[1 0 1 1]
$\tilde{\mathbf{x}}^{(5)}$	[0 1 0 1]	[0 1 0 1]

- Ham. Loss 0.3
- 0/1 Loss 0.6

HAMMING LOSS VS. 0/1 LOSS

Example: 0/1 LOSS VS. HAMMING LOSS

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[1 0 **1** 1]
$\tilde{\mathbf{x}}^{(2)}$	[1 0 0 1]	[1 **1** 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[0 1 1 0]	[0 1 **1** 0]
$\tilde{\mathbf{x}}^{(4)}$	[1 0 0 0]	[1 0 1 **0**]
$\tilde{\mathbf{x}}^{(5)}$	[0 1 0 1]	[0 1 0 1]

Optimize HAMMING LOSS

...

- HAM. LOSS 0.2
- 0/1 LOSS 0.8

...0/1 LOSS goes up

Hamming loss vs. 0/1 loss

Example: 0/1 loss vs. Hamming loss

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[**0** **1** **0** 1]
$\tilde{\mathbf{x}}^{(2)}$	[1 0 0 1]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[0 1 1 0]	[0 **0** **1** 0]
$\tilde{\mathbf{x}}^{(4)}$	[1 0 0 0]	[**0** **1** 1 1]
$\tilde{\mathbf{x}}^{(5)}$	[0 1 0 1]	[0 1 0 1]

Optimize 0/1 Loss ...

- Ham. Loss 0.4
- 0/1 Loss 0.4

... Hamming loss goes up

Hamming loss vs. 0/1 loss

Example: 0/1 loss vs. Hamming loss

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[**0 1** 0 1]
$\tilde{\mathbf{x}}^{(2)}$	[1 0 0 1]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[0 1 1 0]	[0 **0 1** 0]
$\tilde{\mathbf{x}}^{(4)}$	[1 0 0 0]	[**0 1** 1 1]
$\tilde{\mathbf{x}}^{(5)}$	[0 1 0 1]	[0 1 0 1]

- Usually cannot minimize both at the same time …
- … unless: labels are independent of each other! [5]

Threshold Selection

Methods often return a posterior probability, or ensemble votes $\mathbf{p}(\tilde{\mathbf{x}})$. Use a threshold of 0.5 ?

$$\hat{y}_j = \begin{cases} 1, & p_j(\tilde{\mathbf{x}}) \geq 0.5 \\ 0, & \text{otherwise} \end{cases}$$

Example with threshold of 0.5

$\tilde{\mathbf{x}}^{(i)}$	$\mathbf{y}^{(i)}$	$\mathbf{p}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathbb{I}[\mathbf{p}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[0.9 0.0 0.4 0.6]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0.1 0.8 0.0 0.8]	[0 1 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[0.8 0.0 0.1 0.7]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0.1 0.7 0.4 0.2]	[0 1 0 0]
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1.0 0.0 0.0 1.0]	[1 0 0 1]

Threshold Selection

Methods often return a posterior probability, or ensemble votes $\mathbf{p}(\tilde{\mathbf{x}})$. Use a threshold of 0.5 ?

$$\hat{y}_j = \begin{cases} 1, & p_j(\tilde{\mathbf{x}}) \geq 0.5 \\ 0, & \text{otherwise} \end{cases}$$

Example with threshold of 0.5

$\tilde{\mathbf{x}}^{(i)}$	$\mathbf{y}^{(i)}$	$\mathbf{p}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathbb{I}[\mathbf{p}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[0.9 0.0 0.4 0.6]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0.1 0.8 0.0 0.8]	[0 1 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[0.8 0.0 0.1 0.7]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0.1 0.7 0.4 0.2]	[0 1 0 0]
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1.0 0.0 0.0 1.0]	[1 0 0 1]

…but would eliminate two errors with a threshold of 0.4 !

Threshold Selection

Threshold calibration strategies:

- Ad-hoc, e.g., $t = 0.5$

Threshold Selection

Threshold calibration strategies:

- Ad-hoc, e.g., $t = 0.5$
- Internal validation, e.g., $t \in \{0.1, 0.2, \ldots, 0.9\}$

Threshold Selection

Threshold calibration strategies:

- Ad-hoc, e.g., $t = 0.5$
- Internal validation, e.g., $t \in \{0.1, 0.2, \ldots, 0.9\}$
- PCut: such that the training data and test data have similar average number of labels/example

$$\hat{t} = \operatorname*{argmin}_t \Big| \underbrace{\frac{1}{N} \sum_{i,j} \mathbb{I}(p_j^{(i)} > t)}_{\text{test data}} - \underbrace{\frac{1}{N} \sum_{i,j} y_j^{(i)}}_{\text{train data}} \Big|$$

  - Can be done efficiently.
  - Can also calibrate t_j for each label individually.
  - Assumes training set similar to test set (i.e., not ideal for data streams)
- Can be viewed as another form of regularization

$$\hat{\mathbf{y}} = \mathbf{b}(\mathbf{h}(\tilde{\mathbf{x}}))$$

Various Real-World Concerns

- In data streams, label dependence (and therefore, appropriate structures/transformations/base classifiers)
  - may not be known in advance
  - must learn it incrementally
  - and adapt to change over time (concept drift)
  - New labels must be incorporated, old labels phased out
- Labels may be missing from training data,
  - but *we don't know when they're missing* (non-relevance $\neq$ missing)
  - Labelling is more intensive per example (affects both manual labelling and active learning)

Outline

Summary

Multi-label classification

- Can be approached via problem transformation or algorithm adaptation
- Label dependence and scalability are the main themes
- An active area of research and a gateway to many related areas

Resources

- Overview [26]
- Review/Survey of Algorithms [33]
- Extensive empirical comparison [14]
- Some slides: A, B, C
- `http://users.ics.aalto.fi/jesse/`

Software & Datasets

- Mulan (Java)
- Meka (Java)
- Scikit-Learn (Python) offers some multi-label support
- Clus (Java)
- LAMDA (Matlab)

Datasets

- `http://mulan.sourceforge.net/datasets.html`
- `http://meka.sourceforge.net/#datasets`

MEKA

- A WEKA-based framework for multi-label classification and evaluation
- support for data-stream, semi-supervised classification

MEKA

`http://meka.sourceforge.net`

A MEKA Classifier

```
package weka.classifiers.multilabel;
import weka.core.*;

public class  DumbClassifier extends MultilabelClassifier {

   /**
    * BuildClassifier
    */
   public void buildClassifier (Instances D) throws Exception {
      // the first L attributes are the labels
      int L = D.classIndex();
   }

   /**
    * DistributionForInstance - return the distribution p(y[j]|x)
    */
   public double[] distributionForInstance(Instance x) throws Exception {
      int L = x.classIndex();
      // predict 0 for each label
      return new double[L];
   }
}
```

References

Antonucci Alessandro, Giorgio Corani, Denis Mauá, and Sandra Gabaglio.
An ensemble of Bayesian networks for multilabel classification.
In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI'13, pages 1220–1225. AAAI Press, 2013.

Hanen Borchani.
Multi-dimensional classification using Bayesian networks for stationary and evolving streaming data.
PhD thesis, Departamento de Inteligencia Artificial, Facultad de Informática, Universidad Politécnica de Madrid, 2013.

Amanda Clare and Ross D. King.
Knowledge discovery in multi-label phenotype data.
Lecture Notes in Computer Science, 2168, 2001.

Krzysztof Dembczyński, Weiwei Cheng, and Eyke Hüllermeier.
Bayes optimal multilabel classification via probabilistic classifier chains.
In *ICML '10: 27th International Conference on Machine Learning*, pages 279–286, Haifa, Israel, June 2010. Omnipress.

Krzysztof Dembczyński, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier.
On label dependence and loss minimization in multi-label classification.
Mach. Learn., 88(1-2):5–45, July 2012.

Chun-Sung Ferng and Hsuan-Tien Lin.
Multi-label classification with error-correcting codes.
In *Proceedings of the 3rd Asian Conference on Machine Learning, ACML 2011, Taoyuan, Taiwan, November 13-15, 2011*, pages 281–295, 2011.

Johannes Fürnkranz, Eyke Hüllermeier, Eneldo Loza Mencía, and Klaus Brinker.
Multilabel classification via calibrated label ranking.
Machine Learning, 73(2):133–153, November 2008.

Nadia Ghamrawi and Andrew McCallum.
Collective multi-label classification.

In *CIKM '05: 14th ACM international Conference on Information and Knowledge Management*, pages 195–200, New York, NY, USA, 2005. ACM Press.

Shantanu Godbole and Sunita Sarawagi.
Discriminative methods for multi-labeled classification.
In *PAKDD '04: Eighth Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 22–30. Springer, 2004.

Yuhong Guo and Suicheng Gu.
Multi-label classification using conditional dependency networks.
In *IJCAI '11: 24th International Conference on Artificial Intelligence*, pages 1300–1305. IJCAI/AAAI, 2011.

Daniel Hsu, Sham M. Kakade, John Langford, and Tong Zhang.
Multi-label prediction via compressed sensing.
In *NIPS '09: Neural Information Processing Systems 2009*, 2009.

Dragi Kocev, Celine Vens, Jan Struyf, and Sašo Deroski.
Tree ensembles for predicting structured outputs.
Pattern Recognition, 46(3):817–833, March 2013.

Abhishek Kumar, Shankar Vembu, AdityaKrishna Menon, and Charles Elkan.
Beam search algorithms for multilabel learning.
Machine Learning, 92(1):65–89, 2013.

Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Sašo Džeroski.
An extensive experimental comparison of methods for multi-label learning.
Pattern Recognition, 45(9):3084–3104, September 2012.

Andrew Kachites McCallum.
Multi-label text classification with a mixture model trained by em.
In *AAAI 99 Workshop on Text Learning*, 1999.

Antti Puurula, Jesse Read, and Albert Bifet.
Kaggle LSHTC4 winning solution.
Technical report, Kaggle LSHTC4 Winning Solution, 2014.

Piyush Rai and Hal Daume.

Multi-label prediction via sparse infinite CCA.
In *NIPS 2009: Advances in Neural Information Processing Systems 22*, pages 1518–1526. 2009.

Jesse Read and Jaakko Hollmén.
A deep interpretation of classifier chains.
In *Advances in Intelligent Data Analysis XIII - 13th International Symposium, IDA 2014*, pages 251–262, October 2014.

Jesse Read and Jaakko Hollmén.
Multi-label classification using labels as hidden nodes.
ArXiv.org, stats.ML(1503.09022v1), 2015.

Jesse Read, Luca Martino, and David Luengo.
Efficient monte carlo methods for multi-dimensional learning with classifier chains.
Pattern Recognition, 47(3):15351546, 2014.

Jesse Read, Bernhard Pfahringer, and Geoff Holmes.
Multi-label classification using ensembles of pruned sets.
In *ICDM 2008: Eighth IEEE International Conference on Data Mining*, pages 995–1000. IEEE, 2008.

Jesse Read, Bernhard Pfahringer, Geoffrey Holmes, and Eibe Frank.
Classifier chains for multi-label classification.
Machine Learning, 85(3):333–359, 2011.

Jesse Read, Antti Puurula, and Albert Bifet.
Multi-label classification with meta labels.
In *ICDM'14: IEEE International Conference on Data Mining (ICDM 2014)*, pages 941–946. IEEE, December 2014.

Robert E. Schapire and Yoram Singer.
Boostexter: A boosting-based system for text categorization.
Machine Learning, 39(2/3):135–168, 2000.

F. A. Thabtah, P. Cowling, and Yonghong Peng.
MMAC: A new multi-class, multi-label associative classification approach.
In *ICDM '04: Fourth IEEE International Conference on Data Mining*, pages 217–224, 2004.

Grigorios Tsoumakas and Ioannis Katakis.
Multi label classification: An overview.
International Journal of Data Warehousing and Mining, 3(3):1–13, 2007.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas.
Random k-labelsets for multi-label classification.
IEEE Transactions on Knowledge and Data Engineering, 23(7):1079–1089, 2011.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis P. Vlahavas.
Effective and efficient multilabel classification in domains with large number of labels.
In *ECML/PKDD Workshop on Mining Multidimensional Data*, 2008.

Jason Weston, Olivier Chapelle, André Elisseeff, Bernhard Schölkopf, and Vladimir Vapnik.
Kernel dependency estimation.
In *NIPS*, pages 897–904, 2003.

Julio H. Zaragoza, Luis Enrique Sucar, Eduardo F. Morales, Concha Bielza, and Pedro Larrañaga.
Bayesian chain classifiers for multidimensional classification.
In *24th International Joint Conference on Artificial Intelligence (IJCAI '11)*, pages 2192–2197, 2011.

Min-Ling Zhang and Kun Zhang.
Multi-label learning by exploiting label dependency.
In *KDD '10: 16th ACM SIGKDD International conference on Knowledge Discovery and Data mining*, pages 999–1008. ACM, 2010.

Min-Ling Zhang and Zhi-Hua Zhou.
ML-KNN: A lazy learning approach to multi-label learning.
Pattern Recognition, 40(7):2038–2048, 2007.

Min-Ling Zhang and Zhi-Hua Zhou.
A review on multi-label learning algorithms.
IEEE Transactions on Knowledge and Data Engineering, 99(PrePrints):1, 2013.

Multi-label Classification

Jesse Read
https://users.ics.aalto.fi/jesse/

Aalto University
Department of Information and Computer Science
Helsinki, Finland

Summer School on Data Sciences for Big Data
September 5, 2015