Multi-label learning from batch and streaming data

Jesse Read

Télécom ParisTech

École Polytechnique

Summer School on Mining Big and Complex Data
5 September 2016 — Ohrid, Macedonia

Introduction

$\mathbf{x} =$ 

Binary classification

$$y \in \{\texttt{sunset}, \texttt{non\_sunset}\}$$

Introduction

$\mathbf{x} =$ 

Multi-*class* classification

$$y \in \{\texttt{sunset}, \texttt{people}, \texttt{foliage}, \texttt{beach}, \texttt{urban}\}$$

Introduction

$\mathbf{x} =$ 

Multi-label classification

$$\mathbf{y} \subseteq \{\texttt{sunset}, \texttt{people}, \texttt{foliage}, \texttt{beach}, \texttt{urban}\}$$
$$\in \{0,1\}^5 \qquad = [1, 0, 1, 0, 0, 0]$$

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

Introduction

	$K = 2$	$K > 2$
$L = 1$	**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
- set of labels (L) is predefined (contrast to *tagging*/*keyword* assignment)

Introduction

Table: Academic articles containing the phrase "*multi-label classification*" (Google Scholar, 2016)

year	in text	in title
1996-2000	23	1
2001-2005	188	18
2006-2010	1470	164
2011-2015	5280	629

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 and Tag Recommendation

For example, the IMDb dataset: Textual movie plot summaries associated with genres (labels). Also: Bookmarks, Bibtex, del.icio.us datasets.

Text Categorization and Tag Recommendation

For example, the IMDb dataset: Textual movie plot summaries associated with genres (labels). Also: Bookmarks, Bibtex, del.icio.us datasets.

	abandoned	*accident*	$\vdots$	*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
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
120919	1	1	...	0	0	0	0	...	0	1

Labelling Images

Images are labelled to indicate

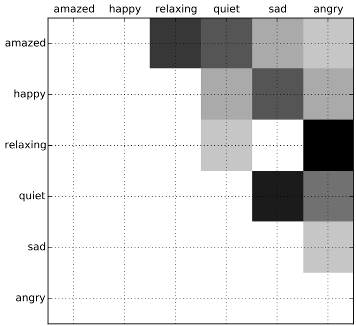
- multiple concepts
- multiple objects
- multiple people

e.g., Associating Scenes with concepts
$\subseteq \{\texttt{beach}, \texttt{sunset}, \texttt{foliage}, \texttt{field}, \texttt{mountain}, \texttt{urban}\}$

Labelling Audio

For example, labelling music with emotions, concepts, etc.

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



Related Tasks

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

X_1	X_2	X_3	X_4	X_5	rank	gender	group
x_1	x_2	x_3	x_4	x_5	1	M	2
x_1	x_2	x_3	x_4	x_5	4	F	2
x_1	x_2	x_3	x_4	x_5	2	M	1

- multi-output regression: outputs are real-valued

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

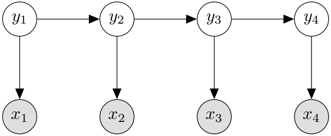
- label ranking, i.e., preference learning

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

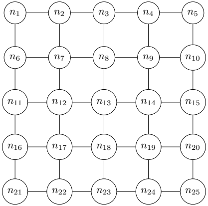
[1] aka multi-target, multi-dimensional

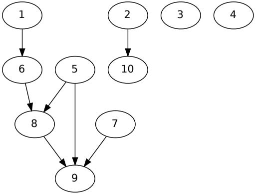
Related Areas

- multi-task learning: multiple tasks, shared representation
- sequential learning: predict across time indices instead of across label indices; each label may have a different input



- structured output prediction: assume particular structure amoung outputs, e.g., pixels, hierarchy





Streaming Multi-label Data

Many advanced applications must deal with data streams:

- Data arrives continuously, potentially infinitely
- Prediction must be made immediately
- Expect concept drift

For example,

- Demand prediction
- Intrusion detection
- Pollution detection

Demand Prediction

Outputs (labels) represent the demand at multiple points.

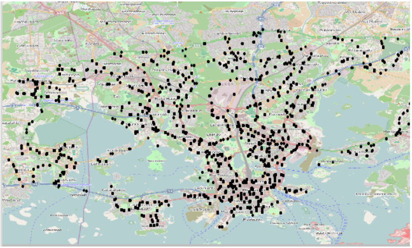


Figure: Stops in the greater Helsinki region. The *Kutsuplus* taxi service could be called to any of these.

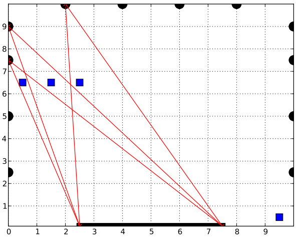
We are interested in predicting, for each label $[y_1, \ldots, y_L]$,

$$p(y_j = 1 | \mathbf{x})$$

• probability of demand at j-th node

Localization and Tracking

Outputs represent points in space which may contain an object ($y_j = 1$) or not ($y_j = 0$). Observations are given as $\mathbf{x}$.



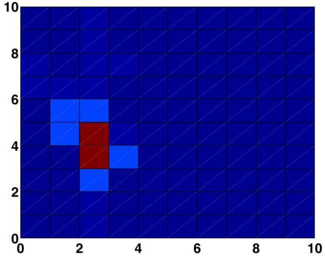


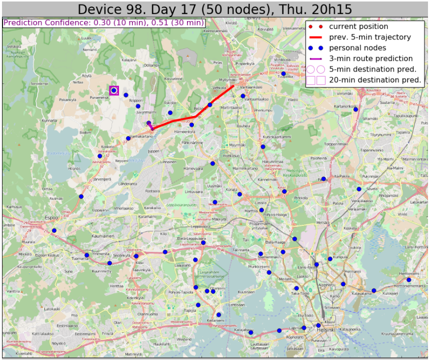
Figure: Modelled on a real-world scenario; a room with a single light source and a number of light sensors.

We are interested in predicting, for each label $[y_1, \ldots, y_L]$,

$$y_j = 1 \quad \bullet \text{ if } j\text{-th tile occupied}$$

Route/Destination Forecasting

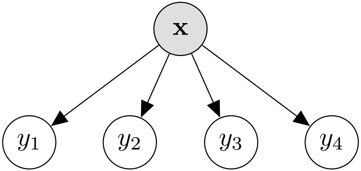
Personal nodes of a traveller and predicted trajectory



- L • number of geographic points of interest
- $\mathbf{x}$ • observed data (e.g., GPS, sensor activity, time of day)
- $p(y_j = 1|\mathbf{x})$ • probability an object is present at the j-th node
- $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N}$ • training data

Outline

Multi-label Classification



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

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

and then,

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

This is the Binary Relevance method (BR).

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.,

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

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.,

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

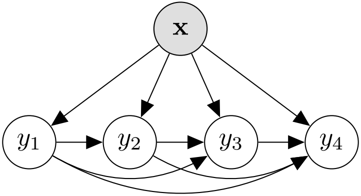
which may not always hold!

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,



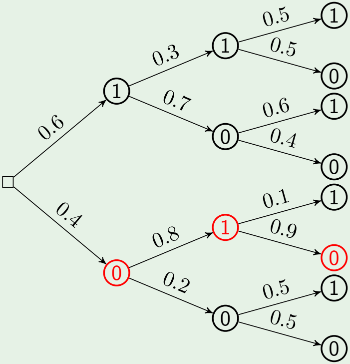
$$p(\mathbf{y}|\mathbf{x}) = p(y_1|\mathbf{x}) \prod_{j=2}^{L} p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1})$$

and,

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

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

- Search space of $\{0, 1\}^L$ paths is too much

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

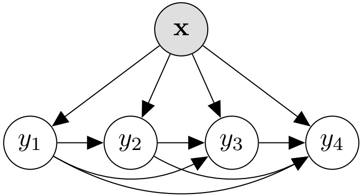
$\mathbf{X}$	Y_1	Y_2
$\mathbf{x}^{(1)}$	0	1
$\mathbf{x}^{(2)}$	1	0
$\mathbf{x}^{(3)}$	0	1
$\mathbf{x}^{(4)}$	1	0
$\mathbf{x}^{(5)}$	0	0

$\mathbf{X}$	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



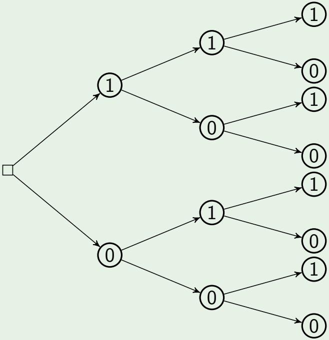
L classifiers for L labels. For test instance $\tilde{\mathbf{x}}$, classify

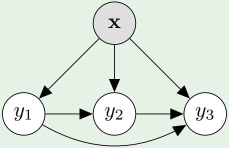
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}})$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1)$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2)$
4. $\hat{y}_4 = h_4(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2, \hat{y}_3)$

and return

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

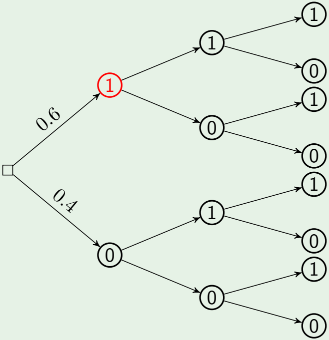
Example

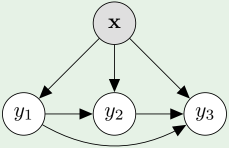




$$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [?, ?, ?]$$

Example

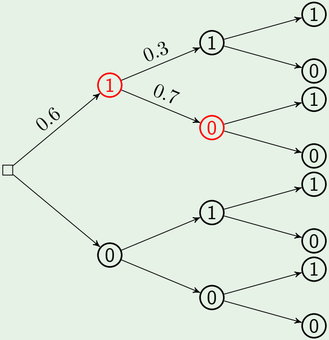


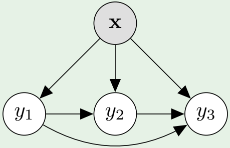


1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \operatorname{argmax}_{y_1} p(y_1|\tilde{\mathbf{x}}) = 1$

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

Example

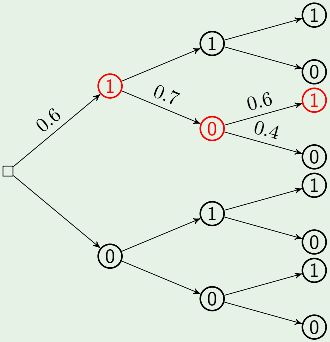




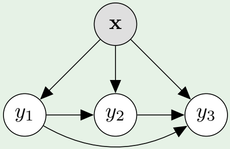
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \operatorname{argmax}_{y_1} p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1) = \ldots = 0$

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

Example

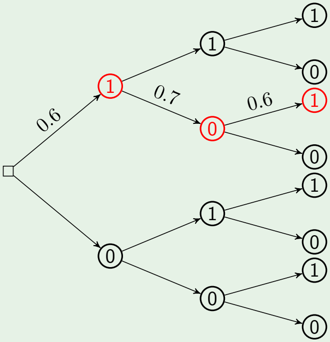


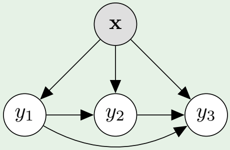
$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, 0, 1]$



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





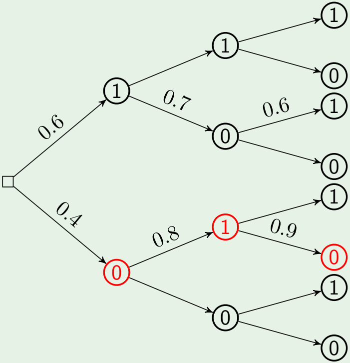
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \operatorname{argmax}_{y_1} p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1) = \ldots = 0$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2) = \ldots = 1$

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

- 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

Monte-Carlo search for CC

Example



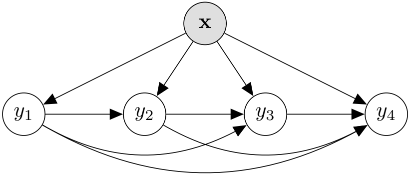
Sample T times ...

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

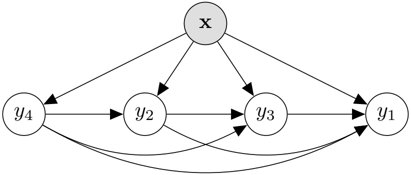
return $\operatorname{argmax}_{\mathbf{y}_t} p(\mathbf{y}_t|\mathbf{x})$

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

Does Label-*order* Matter?



vs



In theory, models are equivalent, since

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

- **but** we are estimating p from finite and noisy data; thus

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

- and in the greedy case,

$$\hat{p}(y_2|y_1,\mathbf{x}) \approx \hat{p}(y_2|\hat{y}_1,\mathbf{x}) = \hat{p}(y_2|y_1 = \underset{y_1}{\operatorname{argmax}}\, \hat{p}(y_1|\mathbf{x})|\mathbf{x})$$

Does Label-*order* Matter?

In theory, models are equivalent, since

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

- **but** we are estimating p from finite and noisy data; thus

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

- and in the greedy case,

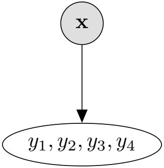
$$\hat{p}(y_2|y_1,\mathbf{x}) \approx \hat{p}(y_2|\hat{y}_1,\mathbf{x}) = \hat{p}(y_2|y_1 = \operatorname*{argmax}_{y_1} \hat{p}(y_1|\mathbf{x})|\mathbf{x})$$

The approximations cause high variance on account of error propagation. We can

1. can reduce variance with an *ensemble* of classifier chains
2. we can search space of chain orders (huge space, but a little search makes a difference)

Label Powerset Method (LP)

One multi-class problem (taking many values),



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

- Each value is a label vector, 2^L in total, but
- typically, only the occurrences of the training set.
- (in practice, $|\mathcal{Y}| \leq$size of training set, 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 (up to 2^L combinations)
- imbalance: few examples per class label
- overfitting: how to predict new value?

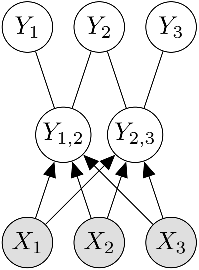
Example

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

Meta Labels

Improving the label-powerset approach:

- decomposition of label set into M subsets of size k ($k < L$)
- pruning, such that, e.g., $Y_{1,2} \in \{[0,0],[0,1],[1,1]\}$
- combine together with random subspace method with a voting scheme



Meta Labels

Improving the label-powerset approach:

- decomposition of label set into M subsets of size k ($k < L$)
- pruning, such that, e.g., $Y_{1,2} \in \{[0,0],[0,1],[1,1]\}$
- combine together with random subspace method with a voting scheme

Method	Inference Complexity
Label Powerset	$O(2^L \cdot D)$
Pruned Sets	$O(P \cdot D)$
Decomposition / RAkEL	$O(M \cdot 2^k \cdot D)$
Meta Labels	$O(M \cdot P \cdot D')$

where $P < 2^L$ and $P < 2^k$, $D' < D$.

Summary of Mehtods

Two views of a multi-label problem of L labels:

1. L binary problems
2. a multi-class problem with up to 2^L classes

Problem Transformation:

1. Transform data into subproblems (binary or multi-class)
2. Apply some off-the-shelf base classifier

or, **Algorithm Adaptation**:

1. Take a suitable single-label classifier (kNN, neural networks, decision trees . . .)
2. Adapt it (if necessary) for multi-label classification

Outline

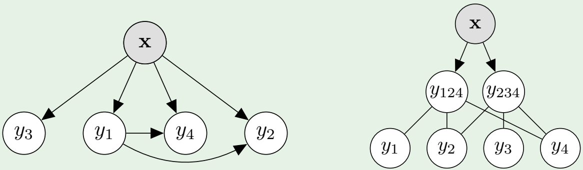
Label Dependence in MLC

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.

Example



- Link particular labels (nodes) together (CC-based methods)
- Form particular label subsets (LP-based methods)

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

Measuring label dependence is expensive, models built on it often do not improve over models built on random dependence!

Problem

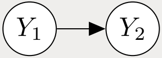
For some metrics (such as Hamming-loss / label accuracy), knowledge of label dependence is theoretically unnecessary!

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

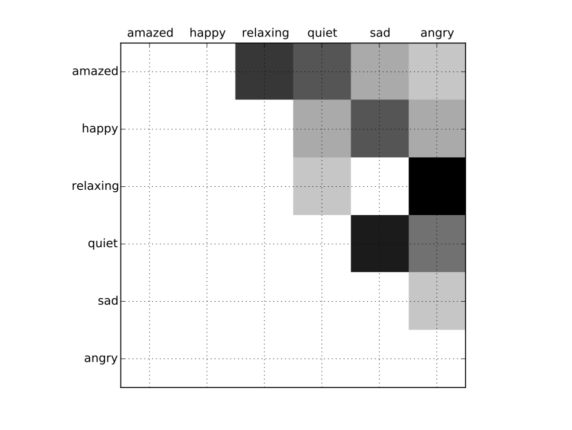


Figure: Music dataset - Mutual Information

Marginal label dependence

Example

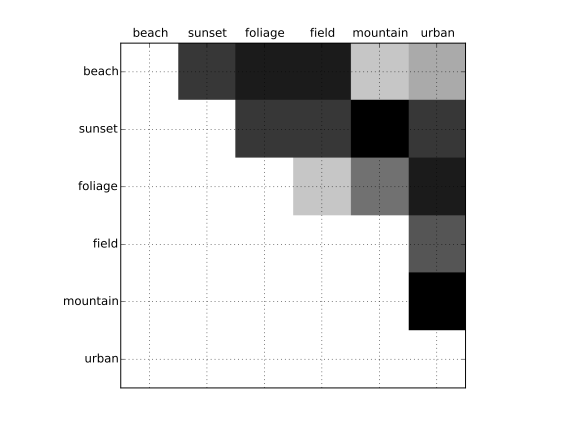


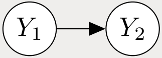
Figure: Scene dataset - Mutual Information

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

Used for regularization/constraints:

1. $\hat{\mathbf{y}} = h(\mathbf{x})$ makes a prediction
2. $\hat{\mathbf{y}}' = g(\hat{\mathbf{y}})$ regularizes the prediction

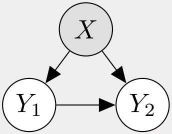
Conditional label dependence

But at classification time, we condition on the input!

Conditional dependence

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

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



- 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

But what does this mean?

Conditional label dependence

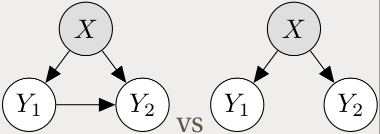
But at classification time, we condition on the input!

Conditional *in*dependence

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

For example,

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



- 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

But what does this mean?

The LOGICAL Problem

Example (The LOGICAL Toy Problem)

		OR	AND	XOR
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

- Each label is a logical operation (independent of the others!)

The LOGICAL Problem

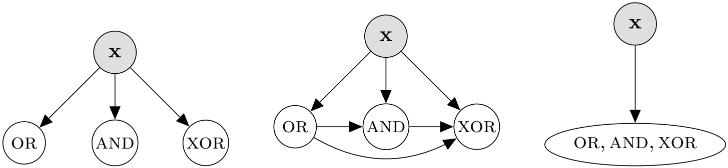


Figure: BR (left), CC (middle), LP (right)

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

- Dependence is introduced by an inadequate model!
- Dependence depends on the model.

The LOGICAL Problem

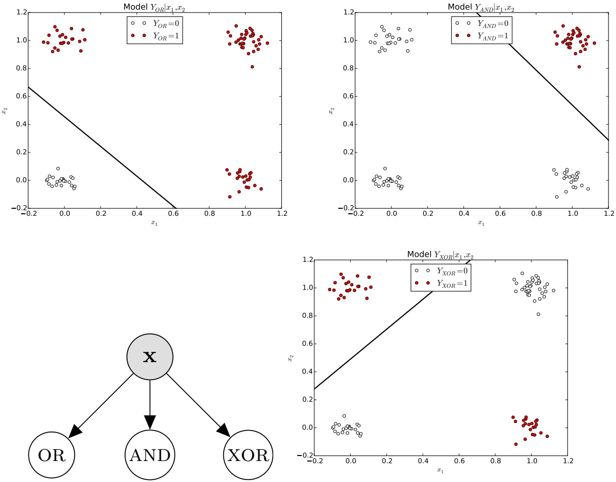


Figure: Binary Relevance (BR): linear decision boundary (solid line, estimated with logistic regression) not viable for Y_{XOR} label

Solution via Structure

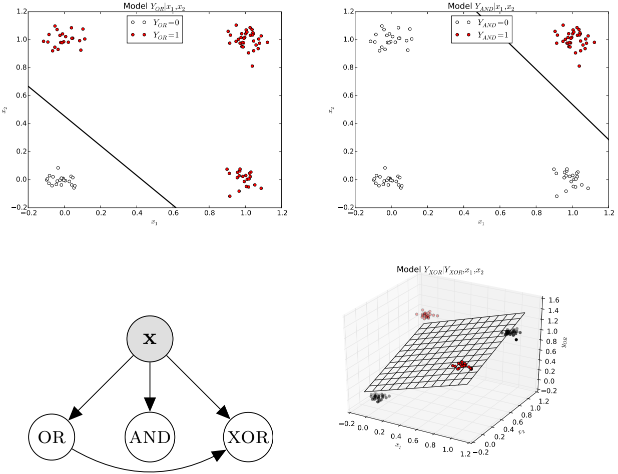


Figure: Classifier chains (CC): linear model now applicable to Y_{XOR}

Solution via Multi-class Decomposition

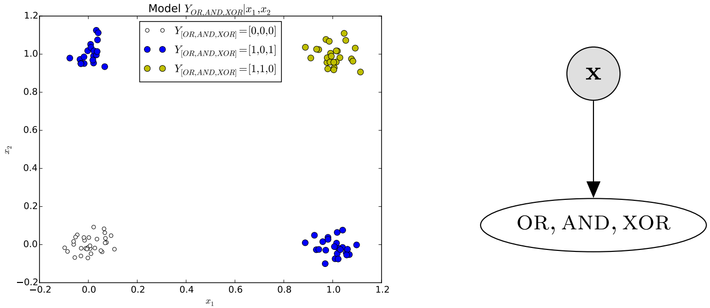


Figure: Label Powerset (LP): solvable with one-vs-one multi-class decomposition for any (e.g., linear) base classifier.

Solution via Con. Independence

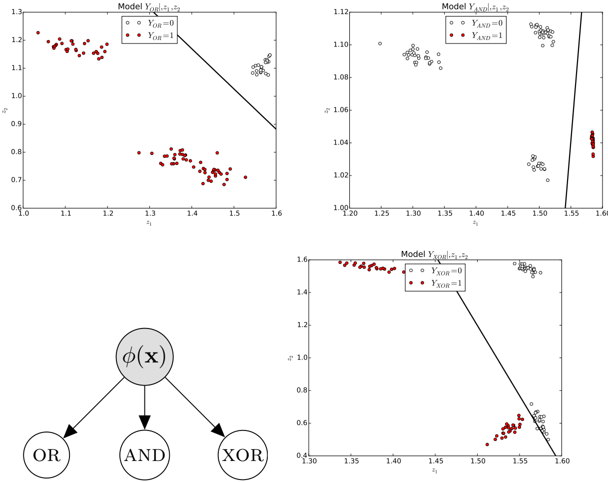


Figure: Solution via latent structure (e.g., random RBF) to new input space $\mathbf{z}$; creating independence: $p(y_{\text{XOR}}|\mathbf{z}, y_{\text{OR}}, y_{\text{AND}}) \approx p(y_{\text{XOR}}|\mathbf{z})$.

Solution via Suitable Base-classifier

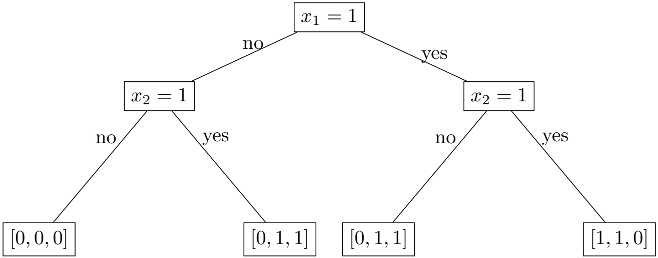


Figure: Solution via non-linear classifier (e.g., Decision Tree). Leaves hold examples, where $\mathbf{y} = [y_{\text{AND}}, y_{\text{OR}}, y_{\text{XOR}}]$

Detecting Dependence

Conditional label dependence and the choice of base model are inseperable.

$$y_j = h_j(\mathbf{x}) + \epsilon_j$$
$$y_k = h_k(\mathbf{x}) + \epsilon_k$$

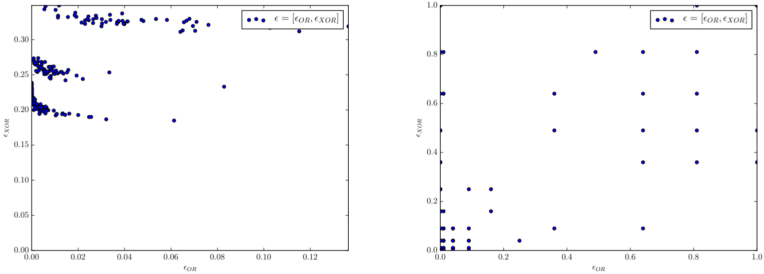


Figure: Errors from logistic regression (left) and decision tree (right).

A fresh look at Problem Transformation

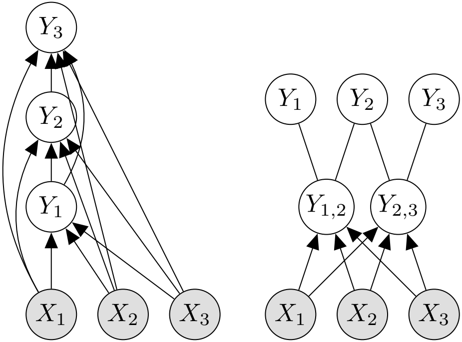


Figure: Standard methods can be viewed as ('deep'/cascaded) basis functions on the label space.

Label Dependence: Summary

- Marginal dependence for regularization
- Conditional dependence
  - . . . depends on the model
  - . . . may be introduced
- Should consider together:
  - base classifier
  - label structure
  - inner-layer structure
- An open problem
- Much existing research is relevant (latent-variable models, neural networks, deep learning, . . .)

Outline

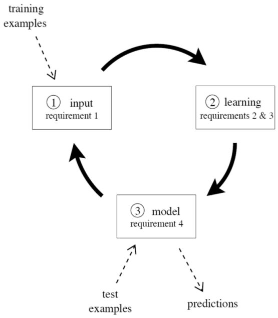
Classification in Data Streams

Setting:

- sequence is potentially infinite
- high speed of arrival
- stream is one-way

Implications

- work in limited memory
- adapt to concept drift



Multi-label Streams Methods

1. Batch-incremental Ensemble
2. Problem transformation with an incremental base learner
3. Multi-label kNN
4. Multi-label incremental decision trees
5. Neural networks

Batch-Incremental Ensemble

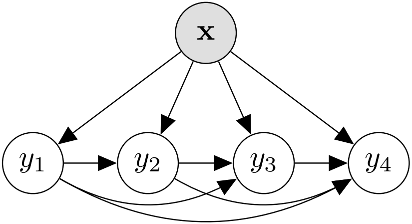
Build regular multi-label models on batches/windows of instances (typically in a [weighted] ensemble).

$$\underbrace{\begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_1, \begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_2, \begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_3, \begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_4}_{\mathbf{h}_1}, \underbrace{\begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_5, \begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_6, \begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_7, \begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_8}_{\mathbf{h}_2}, \begin{bmatrix}\mathbf{x}\\\mathbf{y}\end{bmatrix}_9, \begin{bmatrix}\mathbf{x}\\\hat{\mathbf{y}}\end{bmatrix}_{10}$$

- A common approach in the literature, and can be surprisingly effective
- Free choice of base classifier (e.g., C4.5, SVM)
- What batch size to use?
  - Too small = models insufficient
  - Too large = slow to adapt
  - Too many batches = too slow

Problem Transformation with Incremental Base Learner

Use an incremental learner (Naive Bayes, SGD, Hoeffding trees) with any problem transformation method (BR, LP, CC, ...)

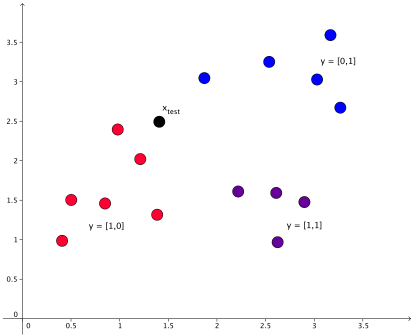


- Simple implementation,
- Risk of overfitting (e.g., with classifier chains)
- Concept drift may invalidate structure
- Limited choice of base learner
  (must be incremental)

Multi-label kNN

Maintain a dynamic buffer of instances, compare each test instance $\tilde{\mathbf{x}}$ to the k neighbouring instances,

$$\hat{y}_j = \begin{cases} 1 & \left(\frac{1}{k}\sum_{i|\mathbf{x}^{(i)} \in \mathsf{Ne}(\tilde{\mathbf{x}})} y_j^{(i)} > 0.5\right) \\ 0 & \text{otherwise} \end{cases}$$



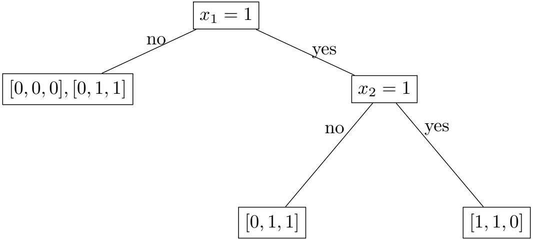
- efficient wrt L
- …but not wrt D
- limited buffer size,
- not suitable for all problems

ML Incremental Decision Trees

- A small sample can suffice to choose a splitting attribute (Hoeffding bound gives guarantees)
- As in regular tree, with modified splitting criteria, e.g.,

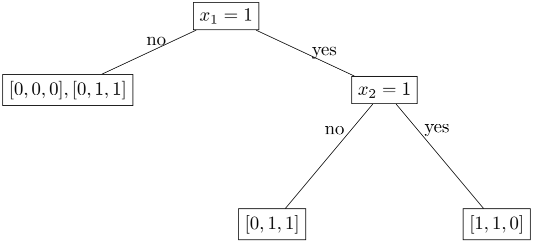
$$H_{\mathsf{ML}}(S) = -\sum_{j=1}^{L} \sum_{c \in \{0,1\}} P(y_j = c) \log_2 P(y_j = c)$$

- Examples with *multiple labels* collect at the leaves.



ML Incremental Decision Trees

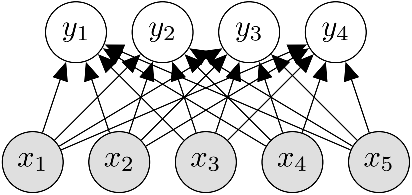
- *Fast*, and usually *competitive*,
- But tree may grow very conservatively,
- ...and need to replace it (or part of it) when concept changes.



- Place multi-label classifiers at the leaves of the tree
- and wrap it in an ensemble.

Neural networks

Each label is a node. Trained with SGD



$$\hat{\mathbf{y}} = \mathbf{W}^\top \tilde{\mathbf{x}}$$

$$\mathbf{g} = \nabla E(\mathbf{W})$$

$$W_{j,k}^{(t+1)} = W_{j,k}^{(t)} + \lambda \mathbf{g}_{j,k}$$

- Can be applied natively
- One layer = BR, should use hidden layers to model label dependence / improve performance
- Hyper-parameter tuning can be tedious
- Relatively *poor performance* in empirical comparisons on standard data streams (improving now with recent advances in SGD, more common use of basis expansion)

Multi-label Data Streams: Issues

- Overfitting
- Class imbalance
- Multi-dimensional concept drift
- Labelled examples difficult to obtain (semi-supervised)
- Dynamic label set
- Time dependence

Multi-label Concept Drift

Consider the relative frequencies of the j-th and k-th labels:

$$\begin{bmatrix} p_j & p_{j,k} \\ & p_k \end{bmatrix}$$

(if marginal independence then $p_{j,k} = p_j p_k$).

Possible drift:

- p_j increases (j-th label relatively more frequent)
- p_j and p_k both decrease (label cardinality decreasing)
- $p_{j,k}$ changes relative to $p_j p_k$ (change in marginal dependence relation between the labels)

Multi-label Concept Drift

And when conditioned on input **x**, we consider the relative frequencies/values of the j-th and k-th errors:

$$\begin{bmatrix} \epsilon_j & \epsilon_{j,k} \\ & \epsilon_k \end{bmatrix}$$

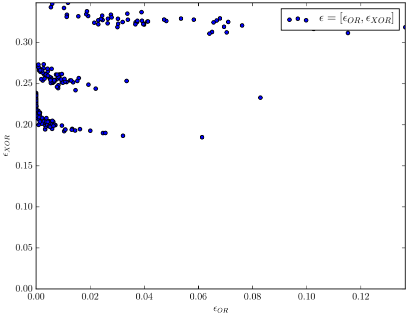
(if conditional independence, then $\epsilon_{j,k} \approx \epsilon_j \cdot \epsilon_k$).

Possible drift:

- ϵ_j increases (more errors on j-th label)
- ϵ_j and ϵ_k both increase (more errors)
- $\epsilon_{j,k}$ changes relative to ϵ_j, ϵ_k (change in conditional dependence relation)

Example

Recall the distribution of errors



This shape may change over time – and structures may need to be adjusted to cope

Dealing with Concept Drift

Possible approaches

- Just ignore it – batch models must be replaced anyway, kNN and SGD adapt; in other cases can use weighted ensembles/fading factor
- Monitor a predictive performance statistic with a change detector (e.g., window based-detection, ADWIN) and reset models
- Monitor the distribution with a change detector (e.g., window based, KL divergence) and reset/recalibrate models

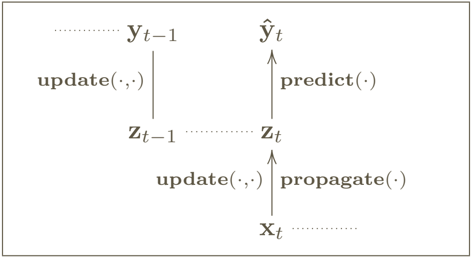
(similar to single-labelled data, except more complex measurement)

Dealing with Unlabelled Instances

- Ignore instances with no label
- Use active learning to get good labels
- Use predicted labels (self-training)
- Use an unsupervised process for example clustering, latent-variable representations.

Dealing with Unlabelled Instances

- Use an unsupervised process for example clustering, latent-variable representations.
  1. $\mathbf{z}_t = g(\mathbf{x}_t)$
  2. $\hat{\mathbf{y}}_t = h(\mathbf{z}_t)$
  3. update g with $(\mathbf{x}_t, \mathbf{z}_t)$
  4. update h with $(\mathbf{z}_{t-1}, \mathbf{y}_{t-1})$ (*if* y_{t-1} is available)



Can also be as one single model

Summary

- Multi-label classification is an active area of research, relevant to many real-world problems
- Methods that deal appropriately wiith label dependence can achieve significant gains over a naive approach
- Many multi-label problems come in the form of a data stream, incurring particular challenges

Multi-label learning from batch and streaming data

Jesse Read

Télécom ParisTech

École Polytechnique

Summer School on Mining Big and Complex Data

5 September 2016 — Ohrid, Macedonia