Multi-label Classification

Jesse Read

Universidad Carlos III de Madrid.
Department of Signal Theory and Communications
Madrid, Spain

II MLKDD
São Carlos, Brazil. July 15, 2013

Multi-label Classification

Single-label classification: Is this a picture of a beach?

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

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

$$\subseteq \{\mathtt{beach}, \mathtt{sunset}, \mathtt{foliage}, \mathtt{field}, \mathtt{mountain}, \mathtt{urban}\}$$

i.e., each instance can have multiple labels instead of a single one!

Multi-label Classification. Part I.

Outline

Introduction: 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	?

Introduction: 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	?	?	?	?

Applications: Text Categorization

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

Applications: Text Categorization

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

example	*abandoned* X_1	*accident* X_2	$\vdots$ $\ldots$	*violent* X_{1000}	*wedding* X_{1001}	horror Y_1	romance Y_2	$\ldots$ $\ldots$	comedy Y_{27}	action Y_{28}
1	1	0	$\ldots$	0	1	0	1	$\ldots$	0	0
2	0	1	$\ldots$	1	0	1	0	$\ldots$	0	0
3	0	0	$\ldots$	0	1	0	1	$\ldots$	0	0
4	1	1	$\ldots$	0	1	1	0	$\ldots$	0	1
5	1	1	$\ldots$	0	1	0	1	$\ldots$	0	1
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
120919	1	1	$\ldots$	0	0	0	0	$\ldots$	0	1

(binary bag-of-words representation)

Applications: Text Categorization

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

example	*abandoned* X_1	*accident* X_2	$\vdots$ $\ldots$	*violent* X_{1000}	*wedding* X_{1001}	Y
1	1	0	...	1	0	{romance, comedy }
2	0	1	...	0	1	{horror}
3	0	0	...	1	0	{romance}
4	1	1	...	0	1	{horror, action}
5	1	0	...	0	1	{action}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
120919	1	0	...	0	1	{action}

(binary bag-of-words representation)

Applications: Text Categorization

For example, the news . . .

Brazil challenges US on 'espionage'

Brazil request clarifications from the US government over allegations that its intelligence agencies spied on Brazilian citizens and companies.

For example, REUTERS

- Reuters collection, newswire stories into 103 topic codes

Applications: E-mail

Enron, e-mails messages made public from the Enron corporation.

"a few beers after work?" $\boxed{\text{work}}$ $\boxed{\text{personal}}$ $\boxed{\text{important}}$

For example, the **UC Berkeley Enron Email Analysis Project** multi-labeled 1702 *Enron* e-mails into 53 categories:

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.

Applications: Image

Images are labeled 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., Emotions dataset, audio tracks labelled with different moods, among: {

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

}

Applications: Medical

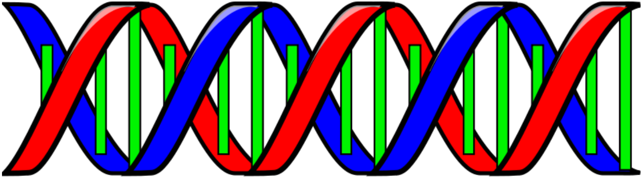
Medical Diagnosis

- medical history, symptoms → diseases / ailments

e.g., Medical dataset,

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

Applications: 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.

Introduction: Notation / Labels as Items in a Set

- Input $\mathcal{X} = \mathbb{R}^D$, Labelset $\mathcal{Y} = \{\lambda_1, \ldots, \lambda_L\}$, label assignment $Y \subseteq \mathcal{Y}$.
- We have set of training examples $\mathcal{D} = \{(\mathbf{x}^{(i)}, Y^{(i)})\}_{i=1}^N =$

$$\underbrace{\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_D^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_D^{(N)} \end{bmatrix}}_{\mathbf{X} \in \mathcal{X}^N} \underbrace{\begin{bmatrix} Y^{(1)} \\ Y^{(2)} \\ \vdots \\ Y^{(N)} \end{bmatrix}}_{\mathbf{Y} \in \mathcal{Y}^N}$$

where

- $\mathbf{x}^{(i)} = [x_1, \ldots, x_D] \in \mathcal{X}$ is the representation of a *data instance*
- $Y^{(i)} \subset \mathcal{Y}$ is some *label set*, where
  for example, $Y^{(1)} = \{\lambda_1, \lambda_4, \lambda_8\}$ are the labels relevant to $\mathbf{x}^{(1)}$.

Introduction: Notation / Labels as Variables

- Input $\mathcal{X} = \mathbb{R}^D$, Output $\mathcal{Y} = \{0,1\}^L$
- We have set of training examples $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N =$

$$\underbrace{\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_D^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_D^{(N)} \end{bmatrix}}_{\mathbf{X} \in \mathcal{X}^N} \underbrace{\begin{bmatrix} y_1^{(1)} & y_2^{(1)} & \cdots & y_L^{(1)} \\ y_1^{(2)} & y_2^{(2)} & \cdots & y_L^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ y_1^{(N)} & y_2^{(N)} & \cdots & y_L^{(N)} \end{bmatrix}}_{\mathbf{Y} \in \mathcal{Y}^N}$$

where
- $\mathbf{x}^{(i)} = [x_1, \ldots, x_D] \in \mathcal{X}$ is the representation of a *data instance*
- $\mathbf{y}^{(i)} = [y_1, \ldots, y_L] \in \mathcal{Y}$ is some *label vector*, where

$$y_j = \begin{cases} 1, & \text{if label } j \text{ is relevant to this instance} \\ 0, & \text{otherwise} \end{cases}$$

Equivalent notation (for $L = 10$):

$$Y^{(i)} = \{\lambda_1, \lambda_4, \lambda_8\} \Leftrightarrow \mathbf{y}^{(i)} = [1,0,0,1,0,0,0,1,0,0]$$

Introduction: Notation / Labels as Variables

Training / Building a model

Use training set $\mathcal{D}\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ to build function / classifier

$$h : \mathcal{X} \to \mathcal{Y}$$

Testing / Prediction

For a test instances $\tilde{\mathbf{x}}$, we obtain the prediction

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

Evaluation

If we have the true classification $\mathbf{y}$ available, we then compare it to $\hat{\mathbf{y}}$ and gauge *accuracy* (more on this later).

Multi-label Data: Datasets

	$\mathcal{X}$ (data inst.)	$\mathcal{Y}$ (labels)	L	N	D	LC
Music	audio data	emotions	6	593	72	1.87
Scene	image data	scene labels	6	2407	294	1.07
Yeast	genes	biological fns	14	2417	103	4.24
Genbase	genes	biological fns	27	661	1185	1.25
Medical	medical text	diagnoses	45	978	1449	1.25
Enron	e-mails	labels, tags	53	1702	1001	3.38
Reuters	news articles	categories	103	6000	500	1.46
TMC07	textual reports	errors	22	28596	500	2.16
Ohsumed	medical articles	disease cats.	23	13929	1002	1.66
IMDB	plot summaries	genres	28	120919	1001	2.00
20NG	posts	news groups	20	19300	1006	1.03
MediaMill	video data	annotations	101	43907	120	4.38
Del.icio.us	bookmarks	tags	983	16105	500	19.02

- L number of labels
- N number of examples
- D number of input feature attributes
- Label Cardinality (LC) $\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{L} y_j^{(i)}$ (Average number of labels per example)

Multi-label Data: Statistics

- L number of labels
- N number of examples
- D number of input feature attributes
- Label Cardinality (LC) $\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{L} y_j^{(i)}$ (Average number of labels per example)
- Label Density $\frac{\text{LC}}{L}$ (LC divided by the number of labels)
- Diversity: $\text{LC} \cdot N$
- Distinct labelsets: proportion of labelsets that are distinct
- Most frequent labelset: proportion of instances that have most frequent labelset

Multi-label Data

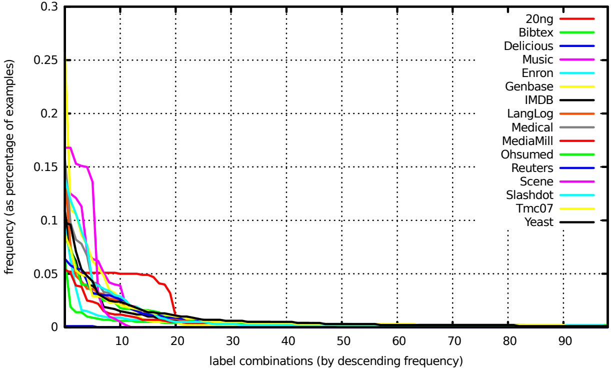


Figure : The proportion of instances assigned the top 100 most frequent labelsets (in descending order of proportion). Zipf's law: a combination ≈ twice as frequent as next-most-frequent.

Multi-label Data

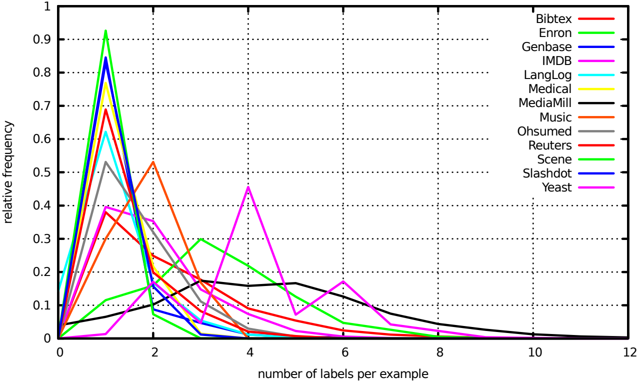


Figure : The proportion of instances in each dataset relevant to $0, 1, 2, \ldots, 12$ of L possible labels; most are relevant to only a few! I.e., Label Cardinality $\ll L$.

Multi-label Data

There are dependencies (i.e., *correlations*, *relationships*, *co-occurences*) among labels

- e.g., $\{\texttt{relaxing-calm}, \texttt{quiet-still}\}$ vs. $\{\texttt{relaxing-calm}, \texttt{angry-aggressive}\}$
- e.g., $\{\texttt{beach}, \texttt{sunset}\}$ vs. $\{\texttt{beach}, \texttt{field}\}$

From the IMDb dataset:

- $P(\texttt{family})P(\texttt{adult}) = 0.068 \cdot 0.015 = 0.001$ ($\approx$ 121 movies)
- $P(\texttt{family}, \texttt{adult}) = 0.0$ (0 movies!)

On most datasets:

- $P(\mathbf{y} = [1, 1, 1, 1, 1, 1]) = 0$

Multi-label Data

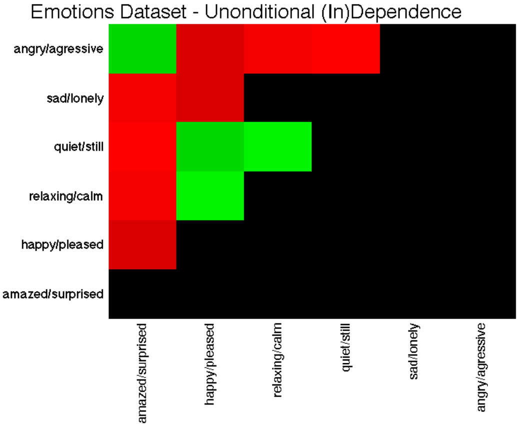


Figure : Person's correlation coefficient $P_{Y_j,Y_k} = \frac{\mathrm{cov}(Y_j, Y_k)}{\sigma_{Y_j}\sigma_{Y_k}}$ on Music.

Main Challenges in Multi-label Classification

The main challenges are to

- model label dependencies; and
- do this efficiently.

Related Tasks

- multi-dimensional / multi-objective learning; $y_j \in \{1, \ldots, K\}$

X_1	X_2	X_3	X_4	X_5	sex	cat.	type
x_1	x_2	x_3	x_4	x_5	F	4	A
x_1	x_2	x_3	x_4	x_5	M	2	B
x_1	x_2	x_3	x_4	x_5	F	3	C

- multi-target regression; $y_j \in \mathbb{R}$

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

- multi-task; data may come from different sources, e.g., different text corpora
- label ranking; interested in label preferences e.g., $\lambda_3 \succ \lambda_1 \succ \lambda_4 \succ \ldots \succ \lambda_2$

Outline

Introduction: Methods for Multi-label Classification

Problem Transformation Methods

- Transforms the multi-label problem into single-label problem(s)
- Use any off-the-shelf single-label classifier to suit requirements
- i.e., **Adapt your data to the algorithm**

Algorithm Adaptation Methods

- Adapt a single-label algorithm to produce multi-label outputs
- Benefit from specific classifier advantages (e.g., efficiency)
- i.e., **Adapt your algorithm to the data**

Many methods involve a mix of both approaches.

Problem Transformation

For example,

- Binary Relevance: L binary problems (one vs. all)
- Label Powerset: one multi-class problem of 2^L class-values
- Pairwise: $\frac{L(L-1)}{2}$ binary problems (all vs. all)
- Copy-Weight: one multi-class problem of L class values

At training time, with $\mathcal{D}$:

1. Transform the multi-label training data to single-label data
2. Learn from the single-label transformed data

At testing time, for $\tilde{\mathbf{x}}$:

1. Make single-label predictions
2. Translate these into multi-label predictions

Binary Relevance (BR)

In the old days . . .

$\mathbf{X}$	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

. . . just make 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

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

Binary Relevance (BR)

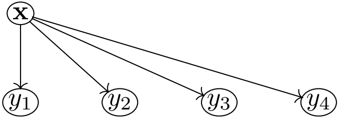
$\mathbf{X}$	Y_1
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	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

Prediction: $\hat{\mathbf{y}} = [h_1(\tilde{\mathbf{x}}), \ldots, h_L(\tilde{\mathbf{x}})]$



Binary Relevance (BR)

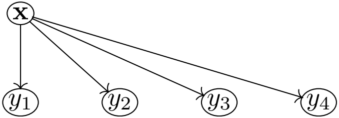
$\mathbf{X}$	Y_1
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	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

Prediction: $\hat{\mathbf{y}} = [h_1(\tilde{\mathbf{x}}), \ldots, h_L(\tilde{\mathbf{x}})]$



Disadvantages:

- Does not model label dependency, $\{\texttt{adult}, \texttt{family}\}$ possible
- Class imbalance, e.g., $P(\neg\texttt{family}) \gg P(\texttt{family})$

BR-improvements

Stacked BR (2BR) [Godbole and Sarawagi, 2004]: stack another BR on top, predict

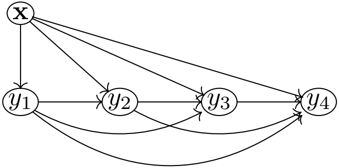
$$\hat{\mathbf{y}} = \mathbf{h}^2(\mathbf{h}^1(\tilde{\mathbf{x}}))$$

For example, given $\tilde{\mathbf{x}}$,

	$\hat{Y}_1$	$\hat{Y}_2$	$\hat{Y}_3$	$\hat{Y}_4$
$\mathbf{h}^1(\tilde{\mathbf{x}})$	1	0	0	1
$\hat{\mathbf{y}} = \mathbf{h}^2(\mathbf{h}^1(\tilde{\mathbf{x}}))$	1	0	0	0

BR-improvements

Chain Classifier (CC) [Cheng et al., 2010, Read et al., 2011]



Like 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

(more on this tomorrow)

Label Powerset Method (LP)

To model label correlations, we can ...

$\mathbf{X}$	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

... make a single multi-*class* problem with 2^L possible class 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

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

Issues with LP

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

- complexity: many class labels
- imbalance: not many examples per class label
- overfitting: how to predict new value?

LP Improvements

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

Ensembles of RAndom k-labEL subsets (RAkEL) [Tsoumakas and Vlahavas, 2007]

- Do LP on M subsets $\subset \{\lambda_1, \ldots, \lambda_L\}$ of size k

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

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

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

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

LP Improvements

Ensembles of RAndom k-labEL subsets (RAkEL) [Tsoumakas and Vlahavas, 2007]

- Do LP on M subsets $\subset \{\lambda_1, \ldots, \lambda_L\}$ of size k

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

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

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

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

- 2^k problems much easier to deal with than 2^L (but still models label dependencies)
- use k and M (number of models) to trade-off dependency modelling and scalability

LP Improvements

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

Ensembles of Pruned Sets (EPS) [Read et al., 2008]

- ‘prune’ out infrequent labelsets, replace with sampled frequent sets

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

LP Improvements

Ensembles of Pruned Sets (EPS) [Read et al., 2008]

- 'prune' out infrequent labelsets, replace with sampled frequent sets

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

- best used in an ensemble (of M models), parameterised by
  - p: a combination occurring $\leq p$ is *infrequent*
  - n: replace them with n subsampled frequent sets (if available)
- keep (most) label dependency information, reduce complexity and other LP issues

Ensemble-based Voting

Ensemble methods (e.g., RA*k*EL, EPS) make prediction via a voting scheme. For some test instance $\tilde{\mathbf{x}}$:

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

(majority vote; can also use weighted vote, *threshold*)

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

Pairwise Binary (PW)

Another binary transformation, but this time . . .

$\mathbf{X}$	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

. . . make $\frac{L(L-1)}{2}$ binary classifiers (*all-vs-all*) . . .

$\mathbf{X}$	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

where each model is trained based on examples annotated by at least one of the labels, but not both.

Pairwise Binary (PW)

$\mathbf{X}$	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

Predict $y_{j,k} = \mathbf{h}_{j,k}(\tilde{\mathbf{x}})$ *for all* $1 \leq j < k \leq L$

$$y_{j,k} = \begin{cases} 0, & \lambda_j \succ \lambda_k \\ 1, & \lambda_k \succ \lambda_j \end{cases}$$

Pairwise Binary (PW)

$\mathbf{X}$	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

Predict $y_{j,k} = \mathbf{h}_{j,k}(\tilde{\mathbf{x}})$ *for all* $1 \leq j < k \leq L$

$$y_{j,k} = \begin{cases} 0, & \lambda_j \succ \lambda_k \\ 1, & \lambda_k \succ \lambda_j \end{cases}$$

Issues:

- this produces pairwise rankings, how to get a labelset?
- how much sense does it make to find a decision boundary between overlapping labels?
- can be expensive in terms of numbers of classifiers ($\frac{L(L-1)}{2}$)

PW Improvements

- Calibrated Label Ranking CLR ([Fürnkranz et al., 2008]): Calibrate a 'virtual label' λ_0 to split the ranking:

$$\lambda_1 \succ \lambda_3 \succ \lambda_0 \succ \lambda_4 \succ \lambda_2 \ldots$$

- Can also have a four-class problem:

$$Y_{j,k} \in \{00, 01, 10, 11\}$$

  - like pairwise 'LP'
  - larger subproblems than PW

Copy-Weight Classifier (CW)

$\mathbf{X}$	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

... make a single multi-class problem with L possible class values:

$\mathbf{X}$	$Y \in \{1, \ldots, L\}$	w
$\mathbf{x}^{(1)}$	2	0.5
$\mathbf{x}^{(1)}$	3	0.5
$\mathbf{x}^{(2)}$	1	1.0
$\mathbf{x}^{(3)}$	2	1.0
$\mathbf{x}^{(4)}$	1	0.5
$\mathbf{x}^{(4)}$	4	0.5
$\mathbf{x}^{(5)}$	4	1.0

each example duplicated $|Y^{(i)}|$ times, weighted as $\frac{1}{|Y^{(i)}|}$.

Copy-Weight Classifier (CW)

$\mathbf{X}$	$Y \in \{1, \ldots, L\}$	w
$\mathbf{x}^{(1)}$	2	0.5
$\mathbf{x}^{(1)}$	3	0.5
$\mathbf{x}^{(2)}$	1	1.0
$\mathbf{x}^{(3)}$	2	1.0
$\mathbf{x}^{(4)}$	1	0.5
$\mathbf{x}^{(4)}$	4	0.5
$\mathbf{x}^{(5)}$	4	1.0

Predict $\hat{\mathbf{y}} = \left[\mathcal{I}[h(y_1|\tilde{\mathbf{x}}) > 0.5], \ldots, \mathcal{I}[h(y_L|\tilde{\mathbf{x}}) > 0.5]\right]$ *where*
$h(y_j|\tilde{\mathbf{x}}) \approx p(y_j = 1|\tilde{\mathbf{x}})$

Issues / Disadvantages:

- decision boundary for identical instances / different classes?
- transformed dataset grows large (N) with high label cardinality
- no obvious way to model dependencies (like BR)

Algorithm Adaptation

Take your favourite classifier, make it multi-label capable.

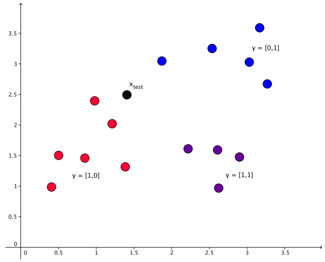
Adapting, e.g.,

- k-Nearest Neighbours
- Decision Trees
- Neural Networks
- Support Vector Machines

to learn from multi-label data and make multi-label predictions.

k Nearest Neighbours

- kNN assigns to $\tilde{\mathbf{x}}$ the majority class of the k 'nearest neighbours'
- MLkNN [Zhang and Zhou, 2007] assigns to $\tilde{\mathbf{x}}$ the most common *labels* of the k nearest neighbours neighbours



- ... combined with Bayesian inference (MAP principle):

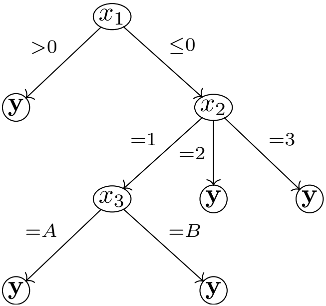
$$y_j = \begin{cases} 1, & \text{if } P(c_{j,\mathbf{x}}|y_j = 1)P(y_j = 1) \geq P(c_{j,\mathbf{x}}|y_j = 0)P(y_j = 0) \\ 0, & \text{otherwise} \end{cases}$$

($c_{j,\mathbf{x}} :=$ number of examples in neighbourhood of $\mathbf{x}$ with $y_j = 1$; *P*robabilities estimated from training data).

Decision Tree

- Multi-label C4.5 [Clare and King, 2001]: Extension of the popular C4.5 decision tree algorithm; with multi-label entropy:

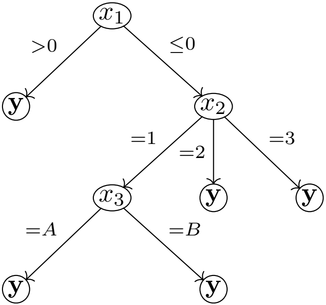
$$H_{\mathrm{ML}}(S) = \sum_{j=1}^{L} P(y_j)\log(P(y_j)) + (1 - P(y_j))\log(1 - P(y_j))$$



- constructed just like C4.5
- allows multiple labels at the leaves

Decision Tree

- Multi-label C4.5 [Clare and King, 2001]: Extension of the popular C4.5 decision tree algorithm



- constructed just like C4.5
- allows multiple labels at the leaves
- works well in an ensemble / random forest

Maximum Margin Method

RankSVM, a Maximum Margin approach [Elisseeff and Weston, 2002]:

- one classifier for each label

$$h_j(\mathbf{x}) = \mathbf{w}_j^\top \mathbf{x} + b_j$$

- use kernel trick for non-linearity
- define multi-label margin, for each $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ in training set $\mathcal{D}$:

$$\min_{(\mathbf{x}^{(i)},\mathbf{y}^{(i)})\in\mathcal{D}} \min_{j,k} \frac{\mathbf{w}_j^\top \mathbf{x} + b_j - \mathbf{w}_k^\top \mathbf{x} - b_k}{\|\mathbf{w}_j - \mathbf{w}_k\|}$$

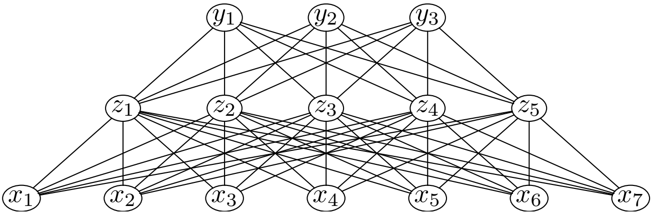
- solve with quadratic programming
- improved performance over BR with SVMs

Neural Networks

BPMLL [Zhang and Zhou, 2006] is

- a regular back-prop. neural network with multiple outputs
- trained with gradient descent + error back-propagation
- with an error function based on ranking (relevant labels should be ranked higher than non-relevant labels)

$$E = \sum_{i=1}^{N} E_i = \sum_{i=1}^{N} \frac{1}{|Y_i||\bar{Y}_i|} \sum_{(j,k)\in Y_i \times \bar{Y}_i} \exp(-(y_k^{(i)} - y_j^{(i)}))$$



- one hidden layer
- one output per label

Which method is best?

Unsurprisingly, this depends on the problem.

- For efficiency / speed: Decision Tree-based
- For flexibility: problem transformation methods, esp. BR-based
- For predictive power? Use ensembles (most modern methods)

Which method is best?

Unsurprisingly, this depends on the problem.

- For efficiency / speed: Decision Tree-based
- For flexibility: problem transformation methods, esp. BR-based
- For predictive power? Use ensembles (most modern methods)

An extensive empirical study by [Madjarov et al., 2012] recommends:

- **RT-PCT**: Random Forest of Predictive Clustering Trees (Algorithm Adaptation, Decision Tree based)
- **HOMER**: Hierarchy Of Multilabel ClassifiERs (Problem Transformation, LP-based (original presentation))
- **CC**: Classifier Chains (Problem Transformation, BR-based)

(More on these later)
But what do we mean by '**best**'?

Outline

Multi-label Evaluation

In single-label classification, accuracy is just:

$$= \frac{1}{N}\sum_{i=1}^{N}\mathcal{I}[\hat{y}^{(i)} = y^{(i)}]$$

($\mathcal{I}[c]$ returns 1 if condition c holds, 0 otherwise)

In multi-label classification, e.g., :

$$\hat{\mathbf{y}} = [0, 0, 0, 0, 1, 0, 0]$$

$$\mathbf{y} = [0, 0, 0, 0, 1, 1, 0]$$

- compare each bit? too lenient?
- treat as a single label? too strict?

Multi-label Evaluation Metrics

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[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]

Hamming loss

$$= \frac{1}{NL} \sum_{i=1}^{N} \sum_{i=1}^{L} \mathcal{I}[\hat{y}_j^{(i)} \neq y_j^{(i)}]$$

$$= 0.20$$

Multi-label Evaluation Metrics

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[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]

0/1 LOSS

$$= \frac{1}{N} \sum_{i=1}^{N} \mathcal{I}(\hat{\mathbf{y}}^{(i)} \neq \mathbf{y}^{(i)})$$

$$= 0.60$$

Often used as EXACT MATCH (1−0/1 LOSS)

Multi-label Evaluation Metrics

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[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]

ACCURACY

$$= \frac{1}{N} \sum_{i=1}^{N} \frac{|\hat{\mathbf{y}}^{(i)} \wedge \mathbf{y}^{(i)}|}{|\hat{\mathbf{y}}^{(i)} \vee \mathbf{y}^{(i)}|}$$

$$= \frac{1}{5}(\frac{1}{3} + 1 + 1 + \frac{1}{2} + \frac{1}{2})$$

$$= 0.67$$

(Where $\vee$ and $\wedge$ are the logical OR and AND operations, applied vector-wise)

Multi-label Evaluation Metrics

Sometimes we want to evaluate probabilities / confidences directly.

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[0.9 0.0 0.4 0.6]
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0.1 0.8 0.0 0.8]
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[0.8 0.0 0.1 0.7]
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0.1 0.7 0.1 0.2]
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1.0 0.0 0.0 1.0]

where $\mathbf{h}(\tilde{\mathbf{x}}) \approx [p(y_1 = 1|\tilde{\mathbf{x}}), \ldots, p(y_L = 1|\tilde{\mathbf{x}})]$

LOG LOSS – like HAMMING LOSS, to encourage good 'confidence',

- $y_j = 1,\ h_j(\tilde{\mathbf{x}}) = 0.4$ incurs loss of $-\log(0.4) = 0.92$
- $y_j = 1,\ h_j(\tilde{\mathbf{x}}) = 0.1$ incurs loss of $-\log(0.1) = 2.30$

Multi-label Evaluation Metrics

Or we may want to evaluate the label ranking.

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[0.9 0.0 0.4 0.6]
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0.1 0.8 0.0 0.8]
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[0.8 0.0 0.1 0.7]
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0.1 0.7 0.1 0.2]
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1.0 0.0 0.0 1.0]

where $\mathbf{h}(\tilde{\mathbf{x}}) \approx [p(y_1 = 1|\tilde{\mathbf{x}}), \ldots, p(y_L = 1|\tilde{\mathbf{x}})]$

RANKING LOSS – to encourage good ranking;
evaluates the average fraction of label pairs miss-ordered for $\tilde{\mathbf{x}}$:

$$= \frac{1}{N} \sum_{i=1}^{N} \sum_{(j,k):y_j > y_k} \left(\mathcal{I}[r_i(j) < r_i(k)] + \frac{1}{2} \mathcal{I}[r_i(j) = r_i(k)] \right)$$

where $r_i(j) :=$ ranking of label j for instance $\tilde{\mathbf{x}}^{(i)}$

Multi-label Evaluation Metrics

Or we may want to evaluate the label ranking.

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$	
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[0.9 0.0 0.4 0.6]	$r(1) < r(3) < r(4) < r(2)$
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0.1 0.8 0.0 0.8]	$r(2) = r(4) < r(1) < r(3)$
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[0.8 0.0 0.1 0.7]	$r(1) < r(4) < r(3) < r(2)$
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0.1 0.7 0.1 0.2]	$r(2) < r(4) < r(3) = r(1)$
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1.0 0.0 0.0 1.0]	$r(1) = r(5) < r(2) < r(3)$

where $\mathbf{h}(\tilde{\mathbf{x}}) \approx [p(y_1 = 1|\tilde{\mathbf{x}}), \ldots, p(y_L = 1|\tilde{\mathbf{x}})]$

RANKING LOSS – to encourage good ranking;
evaluates the average fraction of label pairs miss-ordered for $\tilde{\mathbf{x}}$:

$$\frac{1}{5}(\frac{1}{4} + \frac{0}{4} + \frac{0}{4} + \frac{1.5}{4} + \frac{1}{4})$$

Multi-label Evaluation Metrics

Other metrics used in the literature:

- ONE ERROR – if top ranked label is not in set of true labels
- COVERAGE – average "depth" to cover all true labels
- PRECISION
- RECALL
- macro-averaged $\mathrm{F1}$ (ordinary averaging of a binary measure)
- micro-averaged $\mathrm{F1}$ (labels as different instances of a 'global' label)
- PRECISION vs. RECALL curves

Multi-label Evaluation: Which Metric to Use?

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

Multi-label Evaluation: Which Metric to Use?

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]

Optimizing HAMMING LOSS ...

- HAM. LOSS 0.2
- 0/1 LOSS 0.8

Multi-label Evaluation: Which Metric to Use?

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]

Optimizing 0/1 LOSS ...

- HAM. LOSS 0.4
- 0/1 LOSS 0.4

Multi-label Evaluation: Which Metric to Use?

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]

- HAMMING LOSS can in principal be minimized without taking label dependence into account.
- For 0/1 LOSS label dependence must be taken into account.
- Usually not be possible to minimize both at the same time!

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

Methods that output real values

Many methods return real values $\mathbf{h}(\tilde{\mathbf{x}}) \in \mathbb{R}^L$, which may be, e.g.,

- probabilistic information; or
- votes from an ensemble process

Example: Prediction from ensemble of 3 multi-label models

For some test instance $\tilde{\mathbf{x}}$ ...

	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$	$\hat{y}_4$
$\mathbf{h}^1(\tilde{\mathbf{x}})$	1	0	1	0
$\mathbf{h}^2(\tilde{\mathbf{x}})$	0	1	1	0
$\mathbf{h}^3(\tilde{\mathbf{x}})$	1	0	1	0
$\mathbf{h}(\tilde{\mathbf{x}})$	2	1	3	0
$\equiv$	0.67	0.33	1.00	0.00
$\hat{\mathbf{y}} \in \{0,1\}^L$	?	?	?	?

We may want to evaluate these directly (e.g., LOG LOSS); but we usually need to convert them to binary values ($\hat{\mathbf{y}}$).

Threshold Selection

Use a threshold of 0.5 ?

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

Example with threshold of 0.5

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathcal{I}[\mathbf{h}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[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

Example with threshold of 0.5

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathcal{I}[\mathbf{h}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[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

Example with threshold of 0.5

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathcal{I}[\mathbf{h}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[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]

Possible thresholding strategies:

- Use *ad-hoc* threshold, e.g., 0.5
  - how to know which threshold to use?

Threshold Selection

Example with threshold of 0.5

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathcal{I}[\mathbf{h}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[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]

Possible thresholding strategies:

- Select a threshold from an internal validation test, e.g., $\in \{0.1, 0.2, \ldots, 0.9\}$
  - slow

Threshold Selection

Example with threshold of 0.5

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathcal{I}[\mathbf{h}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[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]

Possible thresholding strategies:

- Calibrate a threshold such that $\text{LCARD}(\mathbf{Y}) \approx \text{LCARD}(\hat{\mathbf{Y}})$
  - e.g., *training data* has label cardinality of 1.7;
  - set a threshold t such that the label cardinality of the *test* data is as close as possible to 1.7

Threshold Selection

Example with threshold of 0.5

	$\mathbf{y}^{(i)}$	$\mathbf{h}(\tilde{\mathbf{x}}^{(i)})$	$\hat{\mathbf{y}}^{(i)} := \mathcal{I}[\mathbf{h}(\tilde{\mathbf{x}}^{(i)}) \geq 0.5]$
$\tilde{\mathbf{x}}^{(1)}$	[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]

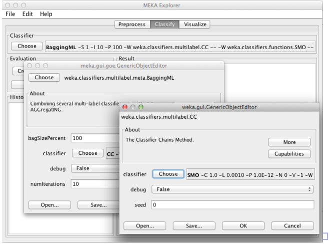
Possible thresholding strategies:

- Calibrate L thresholds such that each $\text{LCard}(\mathbf{Y}_j) \approx \text{LCard}(\hat{\mathbf{Y}}_j)$
  - e.g., the frequency of label $y_j = 1$ is 0.3,
  - set a threshold t_j such that $h_j(\tilde{\mathbf{x}}) \geq t_j \Leftrightarrow \hat{y}_j = 1$ with frequency as close as possible to 0.3

Outline

MEKA

- A WEKA-based framework for multi-label classification and evaluation
- BR, LP, PW, CC, and many others, implemented
  - can be used from the command line or GUI in any ensemble scheme
  - can be used with any single-label base (WEKA) classifier
- many evaluation metrics
- thresholds calibrated automatically (or optionally, set *ad-hoc*)
- `http://meka.sourceforge.net`



MEKA: A Multi-label Classifier

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

public class DumbClassifier extends MultilabelClassifier {

  /**
   * BuildClassifier - build a model h from training data D.
   */
  public void buildClassifier(Instances D) throws Exception {
    // the first L attributes are the labels
    int L = D.classIndex();
  }

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

MEKA: Multi-label datasets

A multi-label dataset with L labels (indexed at the front):

```
@relation  Example_Dataset: -L 3

@attribute Y1 {0,1}
@attribute Y2 {0,1}
@attribute Y3 {0,1}
@attribute X1 {A,B,C}
@attribute X2 {0,1}
@attribute X3 numeric
@attribute X4 numeric

@data
1,0,1,B,1,0.3,0.1
0,1,1,C,0,0.8,0.5
...
```

MEKA: Running experiments

```
# Our dumb classifier, 5-fold CV on Music.arff (randomized)
$ java weka.classifier.multilabel.DumbClassifier -t Music.
    arff -R -x 5
...
          Threshold : 1.0E-5
                  N : 118.4 +/- 0.548
                  L : 6     +/- 0
           Accuracy : 0     +/- 0
             H_loss : 0.312 +/- 0.014
       ZeroOne_loss : 1     +/- 0
...
        LCard_train : 1.87  +/- 0.021
         LCard_pred : 0     +/- 0
         LCard_real : 1.87  +/- 0.084
         Build_time : 0     +/- 0
          Test_time : 0.002 +/- 0.001
         Total_time : 0.002 +/- 0.001
```

MEKA: BR

```
...
  /**
   * BuildClassifier - build L models h[0] .. h[L-1], from training data D.
   */
    public void buildClassifier(Instances D) throws Exception {
        int L = D.classIndex();

        // m_Classifier is part of MultilabelClassifier, and supplied at runtime
        h = AbstractClassifier.makeCopies(m_Classifier,L);
        m_InstancesTemplates = new Instances[L];

        for(int j = 0; j < L; j++) {

            //Select only class attribute 'j'
            Instances D_j = MLUtils.keepAttributesAt(new Instances(D),new int[]{j},L);
            D_j.setClassIndex(0);

            //Build the classifier for that class
            h[j].buildClassifier(D_j);

            m_InstancesTemplates[j] = new Instances(D_j, 0);
        }
    }
...
```

MEKA: Running experiments

```
# BR, with SVMs as the single-label base classifier,
   threshold 0.5
$ java weka.classifier.multilabel.BR -threshold 0.5 -t Music.
   arff -x 5 -R -W weka.classifiers.functions.SMO
...
          Threshold : 0.5
                  N : 118.4 +/- 0.548
                  L : 6     +/- 0
           Accuracy : 0.517 +/- 0.03
             H_loss : 0.191 +/- 0.014
       ZeroOne_loss : 0.73  +/- 0.054
...
        LCard_train : 1.87  +/- 0.021
         LCard_pred : 1.483 +/- 0.084
         LCard_real : 1.87  +/- 0.084
         Build_time : 0.351 +/- 0.15
          Test_time : 0.017 +/- 0.011
         Total_time : 0.369 +/- 0.16
```

MEKA: Running experiments

```
# EPS: Ensembles of PS, SVMs as the base classifier
$ java weka.classifiers.multilabel.meta.EnsembleML -t Music.
    arff -W weka.classifier.multilabel.PS -- -W weka.
    classifiers.functions.SMO
...
          Threshold : 0.6
                  N : 118.4 +/- 0.548
                  L : 6     +/- 0
           Accuracy : 0.587 +/- 0.021
             H_loss : 0.191 +/- 0.012
       ZeroOne_loss : 0.661 +/- 0.03
...
         LCard_train : 1.87  +/- 0.021
          LCard_pred : 1.938 +/- 0.042
          LCard_real : 1.87  +/- 0.084
          Build_time : 26.181 +/- 2.179
           Test_time : 0.099 +/- 0.023
          Total_time : 26.281 +/- 2.197
```

End of Part 1

In Part 2:

- More on Label Dependency
- Advanced Methods for Multi-label Classification:
  from Classifier Chains to Structured Output Learning
- Advanced Topics
- Open Questions and Future Directions
- Summary, Conclusions, References and Resources