

# Multi-label Classification



**Charmgil Hong**  
**cs3750**

*(Presented on Nov 11, 2014)*

# Goals of the talk

- 1.To understand the **geometry of different approaches** for multi-label classification
- 2.To appreciate **how the Machine Learning techniques further improve** the multi-label classification methods
- 3.To learn how to **evaluate** the multi-label classification methods

# Agenda

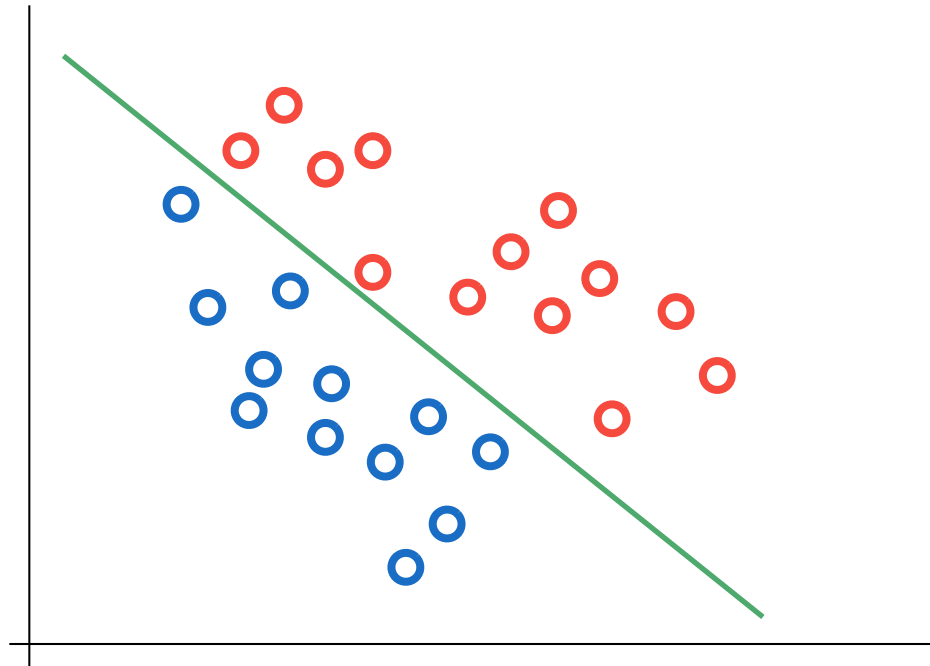
- Motivation & Problem definition
- Solutions
- Advanced solutions
- Evaluation metrics
- Toolboxes
- Summary

# Notation

- $X \in \mathbb{R}^m$  : feature vector variable (input)
- $Y \in \mathbb{R}^d$  : class vector variable (output)
  - $\mathbf{x} = \{x_1, \dots, x_m\}$ : feature vector instance
  - $\mathbf{y} = \{y_1, \dots, y_d\}$ : class vector instance
  - In a shorthand,  $P(Y=\mathbf{y}|X=\mathbf{x}) = P(\mathbf{y}|\mathbf{x})$
- $D_{train}$  : training dataset;  $D_{test}$  : test dataset

# Motivation

- Traditional classification
  - Each data instance is associated with a single class variable



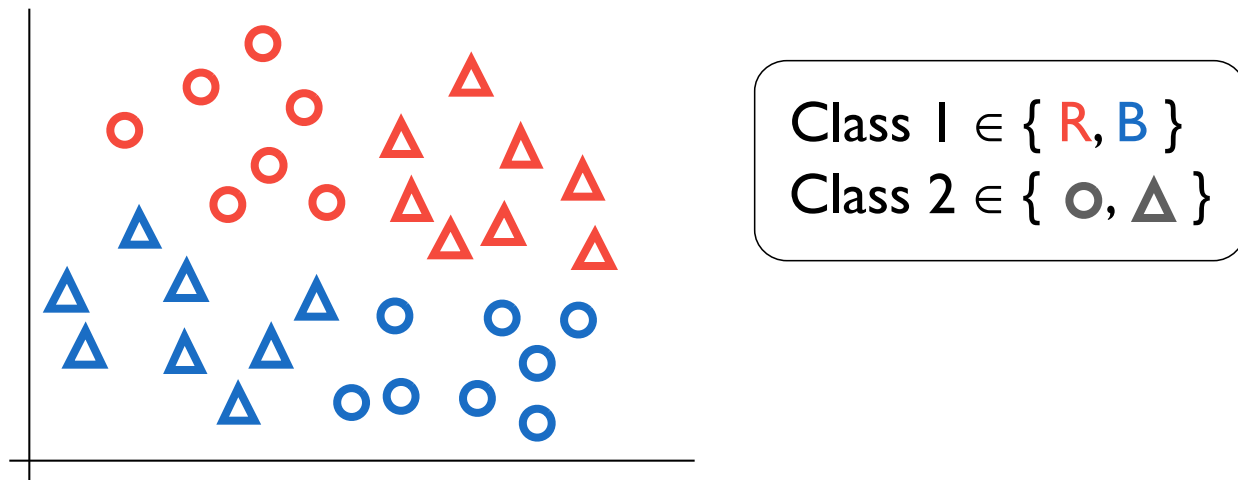
# Motivation

- An issue with traditional classification
  - In many real-world applications, each data instance can be associated with **multiple class variables**
  - Examples
    - A news article may cover multiple topics, such as *politics* and *economics*
    - An image may include multiple objects as *building*, *road*, and *car*
    - A gene may be associated with several biological functions

# Problem Definition

- Multi-label classification (MLC)
  - Each data instance is associated with **multiple binary class variables**
  - Objective: assign each instance the **most probable assignment** of the class variables

$$h : \mathbf{X} \in \mathbb{R}^m \rightarrow \mathbf{Y} \in \{0, 1\}^d$$



# A simple solution

- Idea
  - Transform a multi-label classification problem to multiple single-label classification problems
  - Learn  $d$  independent classifiers for  $d$  class variables



# Binary Relevance (BR) [Clare and King, 2001; Boutell et al, 2004]

- Idea
  - Transform a multi-label classification problem to multiple single-label classification problems
  - Learn  $d$  independent classifiers for  $d$  class variables
- Illustration

$D_{train}$	$X_1$	$X_2$	$Y_1$	$Y_2$	$Y_3$
$n=1$	0.7	0.4	1	1	0
$n=2$	0.6	0.2	1	1	0
$n=3$	0.1	0.9	0	0	1
$n=4$	0.3	0.1	0	0	0
$n=5$	0.8	0.9	1	0	1

$$h_1 : X \rightarrow Y_1$$

$$h_2 : X \rightarrow Y_2$$

$$h_3 : X \rightarrow Y_3$$

# Binary Relevance (BR) [Clare and King, 2001; Boutell et al, 2004]

- Advantages
  - Computationally efficient
- Disadvantages
  - Does not capture the dependence relations among the class variables
  - Not suitable for the objective of MLC
    - Does not find the most probable assignment
    - Instead, it maximizes the marginal distribution of each class variable

# Binary Relevance (BR) [Clare and King, 2001; Boutell et al, 2004]

- Marginal vs. Joint: a motivating example
- Question: find the most probable assignment (**MAP**: maximum a posteriori) of  $\mathbf{Y} = (Y_1, Y_2)$

$P(Y_1, Y_2   \mathbf{X} = \mathbf{x})$	$Y_1 = 0$	$Y_1 = 1$	$P(Y_2   \mathbf{X} = \mathbf{x})$
$Y_2 = 0$	0.2	<b>0.45</b>	<b>0.65</b>
$Y_2 = 1$	0.35	0	0.35
$P(Y_1   \mathbf{X} = \mathbf{x})$	<b>0.55</b>	0.45	

➡ Prediction on the joint (MAP):  $Y_1 = 1, Y_2 = 0$

➡ Prediction on the marginals:  $Y_1 = 0, Y_2 = 0$

- We want to **maximize the joint distribution of  $\mathbf{Y}$**  given observation  $\mathbf{X} = \mathbf{x}$ ; i.e.,

$$h^*(\mathbf{x}) = \arg \max_{\mathbf{y}} P(\mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x})$$

# Another simple solution

- Idea
  - Transform each label combination to a class value
  - Learn a multi-class classifier with the new class values

# Label Powerset (LP) [Tsoumakas and Vlahavas, 2007]

- Idea
  - Transform each label combination to a class value
  - Learn a multi-class classifier with the new class values
- Illustration

$D_{train}$	$X_1$	$X_2$		$Y_1$	$Y_2$	$Y_3$
$n=1$	0.7	0.4		0	0	1
$n=2$	0.6	0.2		0	0	1
$n=3$	0.1	0.9		0	1	0
$n=4$	0.3	0.1		0	1	1
$n=5$	0.8	0.9		1	0	1



$Y_{LP}$
1
1
2
3
4

$$h_{LP} : \mathbf{X} \rightarrow Y_{LP}$$

# Label Powerset (LP) [Tsoumakas and Vlahavas, 2007]

- Advantages
  - Learns the **full joint** of the class variables
    - Each of the new class values maps to a label combination
- Disadvantages
  - **The number of choices in the new class** can be **exponential**  
( $|Y_{LP}| = O(2^d)$ )
    - Learning a multi-class classifier on exponential choices is expensive
    - The resulting class distribution would be **sparse** and **imbalanced**
  - Only predicts **the label combinations that are seen in the training set**

# BR vs. LP



Independent  
classifiers



All possible  
label combo.

- BR and LP are two extreme MLC approaches
  - *BR* maximizes the marginals on each class variable; while *LP* directly models the joint of all class variables
  - BR is computationally more efficient; but does not consider the relationship among the class variables
  - LP considers the relationship among the class variables by modeling the full joint of the class variables; but can be computationally very expensive

# Agenda

✓ Motivation

- Solutions
- Advanced solutions
- Evaluation metrics
- Toolboxes
- Summary



# Solutions

- Section agenda
  - Solutions *rooted on BR*
  - Solutions *rooted on LP*
  - Other solutions

# Solutions rooted on BR

- BR: Binary Relevance [Clare and King, 2001; Boutell et al, 2004]
  - Models independent classifiers  $P(y_i|\mathbf{x})$  on each class variable
  - Does **not learn the class dependences**
- Key extensions from BR
  - Learn the class dependence relations **by adding new class-dependent features** :  $P(y_i|\mathbf{x}, \{\textit{new\_features}\})$

# Solutions rooted on BR

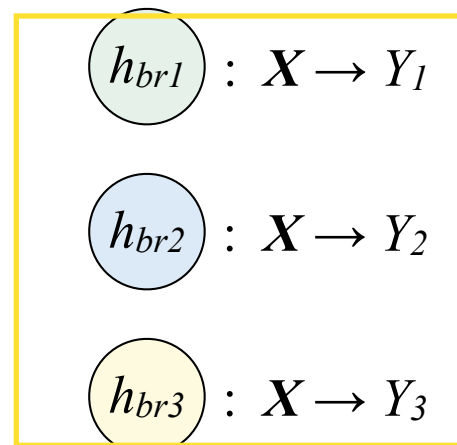
- Idea: **layered approach**
  - *Layer-1*: Learn and predict on  $D_{train}$ , using the *BR* approach
  - *Layer-2*: Learn  $d$  classifiers on **the original features and the output of *layer-1***
- Existing methods
  - Classification with Heterogeneous Features (*CHF*) [Godbole et al, 2004]
  - Instance-based Logistic Regression (*IBLR*) [Cheng et al, 2009]

# Classification with Heterogeneous Features (CHF)

## • Illustration

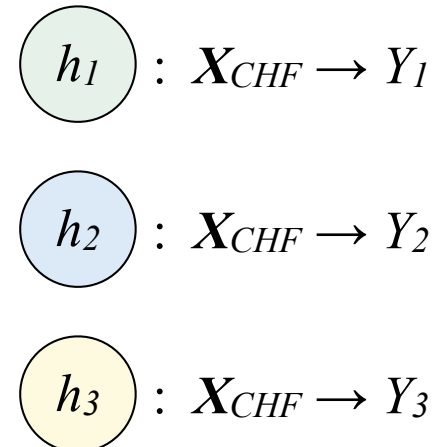
Layer-1

$D_{train}$	$X_1$	$X_2$	$Y_1$	$Y_2$	$Y_3$
$n=1$	0.7	0.4	1	1	0
$n=2$	0.6	0.2	1	1	0
$n=3$	0.1	0.9	0	0	1
$n=4$	0.3	0.1	0	0	0
$n=5$	0.8	0.9	1	0	1



Layer-2

	$X_{CHF}$		$Y_1$	$Y_2$	$Y_3$
	$X_1$	$X_2$	$h_{br1}(X)$	$h_{br2}(X)$	$h_{br3}(X)$
$n=1$	0.7	0.4	.xx	.xx	.xx
$n=2$	0.6	0.2	.xx	.xx	.xx
$n=3$	0.1	0.9	.xx	.xx	.xx
$n=4$	0.3	0.1	.xx	.xx	.xx
$n=5$	0.8	0.9	.xx	.xx	.xx



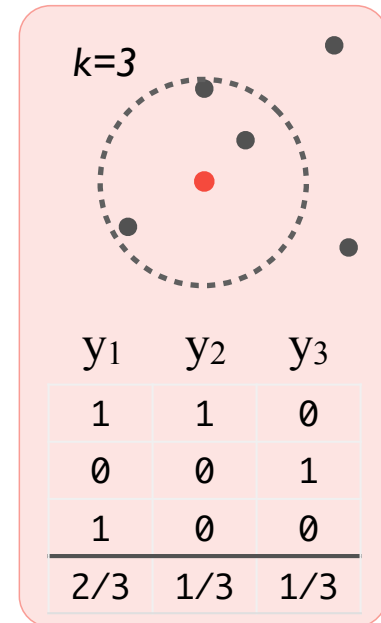
# Instance-based Logistic Regression (IBLR)

- Illustration

$D_{train}$	$X_1$	$X_2$		$Y_1$	$Y_2$	$Y_3$
$n=1$	0.7	0.4		1	1	0
$n=2$	0.6	0.2		1	1	0
$n=3$	0.1	0.9		0	0	1
$n=4$	0.3	0.1		0	0	0
$n=5$	0.8	0.9		1	0	1



$KNN\ Score$		
$\lambda_1$	$\lambda_2$	$\lambda_3$
.xx	.xx	.xx
.xx	.xx	.xx
.xx	.xx	.xx
.xx	.xx	.xx
.xx	.xx	.xx



	$X_{IBLR}$						<div><div></div><div><math>Y_1</math></div></div>	<div><div></div><div><math>Y_2</math></div></div>	<div><div></div><div><math>Y_3</math></div></div>
	$X_1$	$X_2$	$\lambda_1$	$\lambda_2$	$\lambda_3$				
$n=1$	0.7	0.4	.xx	.xx	.xx		1	1	0
$n=2$	0.6	0.2	.xx	.xx	.xx		1	1	0
$n=3$	0.1	0.9	.xx	.xx	.xx		0	0	1
$n=4$	0.3	0.1	.xx	.xx	.xx		0	0	0
$n=5$	0.8	0.9	.xx	.xx	.xx		1	0	1

$h_1 : X_{IBLR} \rightarrow Y_1$

$h_2 : X_{IBLR} \rightarrow Y_2$

$h_3 : X_{IBLR} \rightarrow Y_3$

# Solutions rooted on BR: *CHF* & *IBLR*

- Advantages
  - Model the class dependences by enriching the feature space using the *layer-l* classifiers
- Disadvantages
  - Learn the dependence relations in an **indirect way**
  - The predictions are not stable

# Solutions rooted on LP

- LP: Label Powerset [\[Tsoumakas and Vlahavas, 2007\]](#)
  - Models a multi-class classifier on the enumeration of all possible class assignment
  - Can **create exponentially many classes** and computationally very expensive
- Key extensions from LP
  - **Prune the infrequent class assignments** from the consideration to reduce the size of the class assignment space
  - Represent the joint distribution more compactly

# Pruned problem transformation (PPT) [Read et al, 2008]

- Class assignment conversion in PPT
  - Prune infrequent class assignment sets
  - User specifies the threshold for “infrequency”

$D_{train}$	$X_1$	$X_2$		$Y_1$	$Y_2$	$Y_3$
$n=1$	0.7	0.4		0	0	1
$n=2$	0.6	0.2		0	0	1
$n=3$	0.1	0.9		0	0	0
$n=4$	0.3	0.1		0	1	0
$n=5$	0.1	0.8		0	0	0
$n=6$	0.2	0.1		0	1	0
$n=7$	0.2	0.2		0	1	0
$n=8$	0.2	0.9		0	0	0
$n=9$	0.7	0.3		0	0	1
$n=10$	0.9	0.9		0	1	1



$D_{train-LP}$	$Y_{LP}$
$n=1$	1
$n=2$	1
$n=3$	0
$n=4$	2
$n=5$	0
$n=6$	2
$n=7$	2
$n=8$	0
$n=9$	1
$n=10$	3

}  $|Y_{LP}| = 4$



# Pruned problem transformation (PPT) [Read et al, 2008]

- Class assignment conversion in PPT
  - Prune infrequent class assignment sets
  - User specifies the threshold for “infrequency”

$D_{train}$	$X_1$	$X_2$		$Y_1$	$Y_2$	$Y_3$
$n=1$	0.7	0.4		0	0	1
$n=2$	0.6	0.2		0	0	1
$n=3$	0.1	0.9		0	0	0
$n=4$	0.3	0.1		0	1	0
$n=5$	0.1	0.8		0	0	0
$n=6$	0.2	0.1		0	1	0
$n=7$	0.2	0.2		0	1	0
$n=8$	0.2	0.9		0	0	0
$n=9$	0.7	0.3		0	0	1
$n=10$	0.9	0.9		0	0	1
$n=11$	0.9	0.9		0	1	0



$D_{train-PPT}$	$Y_{PPT}$
$n=1$	1
$n=2$	1
$n=3$	0
$n=4$	2
$n=5$	0
$n=6$	2
$n=7$	2
$n=8$	0
$n=9$	1
$n=10$	1
$n=11$	2

}  $|Y_{PPT}| = 3$

# Solutions rooted on LP: *PPT*

- Advantages

- Simple add-on to the LP method that focuses on key relationships
- Models the full joint more efficiently

- Disadvantages

- Based on an ad-hoc pruning heuristic
  - Mapping to lower dimensional label space is not clear
- (As LP) Only predicts the label combinations that are seen in the training set

## Other solution: MLKNN [Zhang and Zhou, 2007]

- Multi-label k-Nearest Neighbor (MLKNN) [Zhang and Zhou, 2007]
  - Learn a classifier for each class (as BR) by combining k-nearest neighbor with Bayesian inference
  - Application is limited as KNN
    - Does not produce a model
    - Does not work well on high-dimensional data

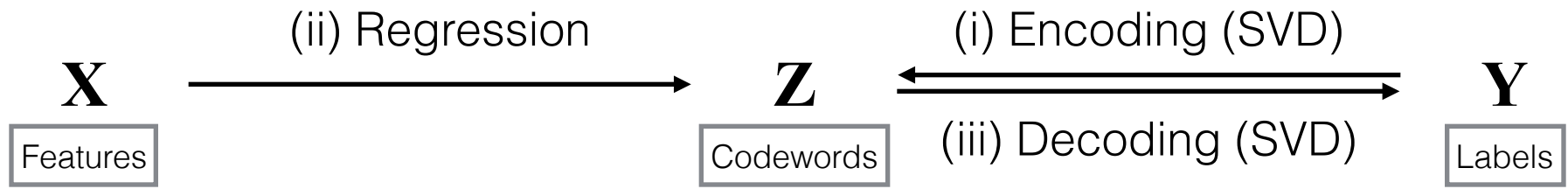
# Multi-label output coding

- Key idea
  - Motivated by the error-correcting output coding (ECOC) scheme [Dietterich 1995; Bose & Ray-Chaudhuri 1960] in communication
  - Solve the MLC problems using lower dimensional codewords
- An output coding MLC method usually consists of three parts:
  - Encoding: Convert output vectors  $\mathbf{Y}$  into codewords  $\mathbf{Z}$
  - Prediction: Perform regression from  $\mathbf{X}$  to  $\mathbf{Z}$ ; say  $\mathbf{R}$
  - Decoding: Recover the class assignments  $\mathbf{Y}$  from  $\mathbf{R}$

# Multi-label output coding

- Existing methods
  - OC (Output Coding) with Compressed Sensing (OCCS) [Hsu et al, 2009]
  - Principle Label Space Transformation (PLST) [Tai and Lin, 2010]
  - OC with Canonical Correlation Analysis (CCAOC) [Zhang and Schneider, 2011]
  - Maximum Margin Output Coding (MMOC) [Zhang and Schneider, 2012]

# Principle Label Space Transformation (PLST) [Tai and Lin, 2010]



- *Encoding:* Convert **output vectors  $\mathbf{Y}$  into codewords  $\mathbf{Z}$** , using the singular vector decomposition (SVD)
- $\mathbf{Z} = \mathbf{V}^T \mathbf{Y} = (V_1^T \mathbf{Y}, \dots, V_q^T \mathbf{Y})$ , where  $\mathbf{V}$  is a  $d \times q$  projection vector ( $d > q$ )
- *Prediction:* Perform **regression from  $\mathbf{X}$  to  $\mathbf{Z}$** ; say  $\mathbf{R}$
- *Decoding:* **Recover the class labels  $\mathbf{Y}$  from  $\mathbf{R}$**  using SVD
- Achieved by optimizing a combinatorial loss function

# Multi-label output coding

- Existing methods are differentiated from one to another mainly by the encoding/decoding schemes they apply

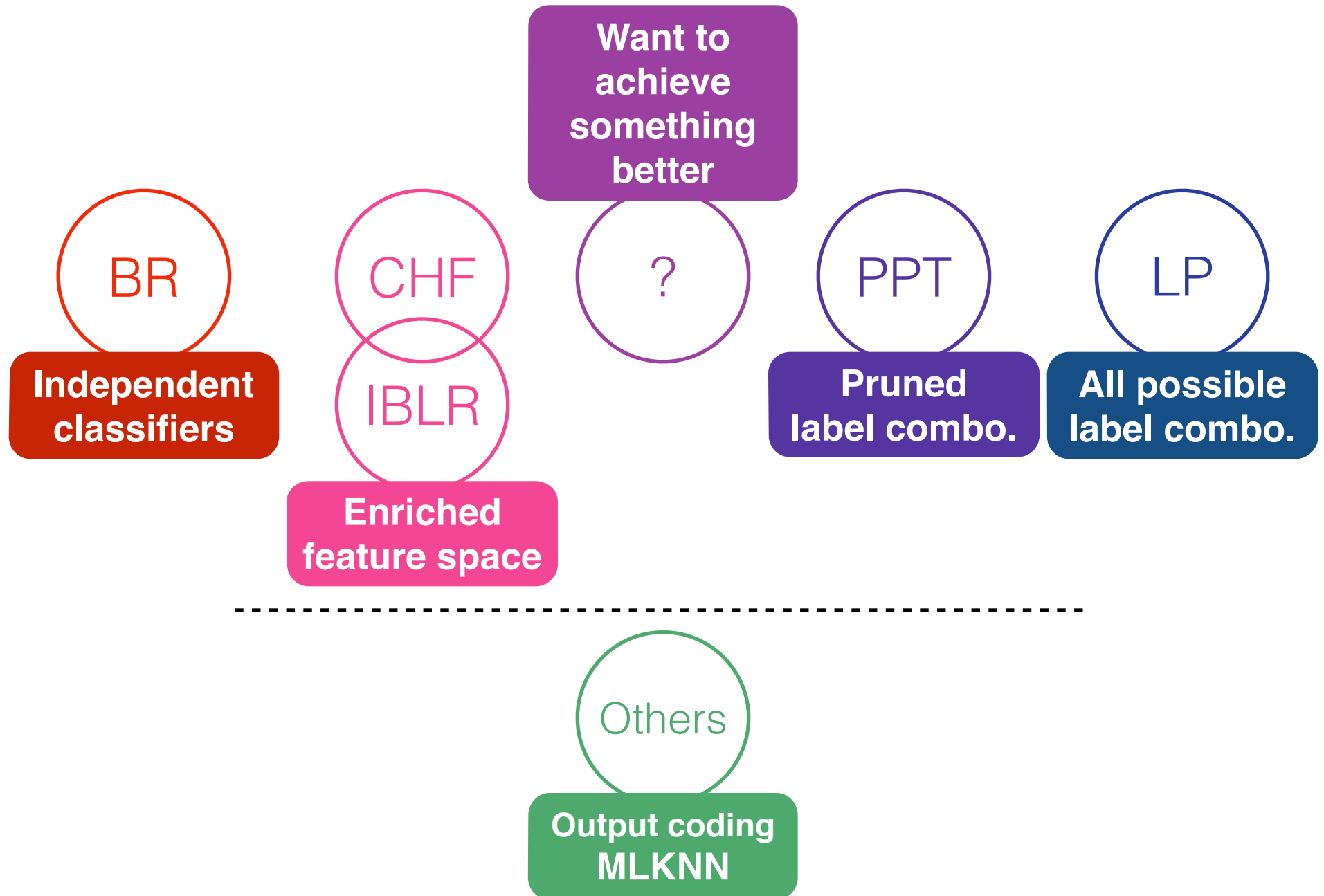
Method	Key difference
<i>OCCS</i>	Uses <b>compressed sensing</b> [Donoho 2006] for encoding and decoding
<i>PLST</i>	Uses <b>singular vector decomposition</b> (SVD) [Johnson & Wichern 2002] for encoding and decoding
<i>CCAOC</i>	Uses <b>canonical correlation analysis</b> (CCA) [Johnson & Wichern 2002] for encoding and <b>mean-field approximation</b> for decoding
<i>MMOC</i>	Uses <b>SVD</b> for encoding and <b>maximum margin formulation</b> for decoding

# Multi-label output coding

- Advantages
  - Show excellent prediction performances
- Disadvantages
  - Only able to predict the single best output for a given input
    - Cannot estimate probabilities for different input-output pairs
  - Not scalable
    - Encoding and decoding steps rely on matrix decomposition, whose complexities are sensitive to  $d$  and  $N$
  - Cannot be generalized to non-binary cases



# Section summary



# Agenda

✓ Motivation

✓ Solutions

- Advanced solutions
- Evaluation metrics
- Toolboxes
- Summary

# Advanced solutions

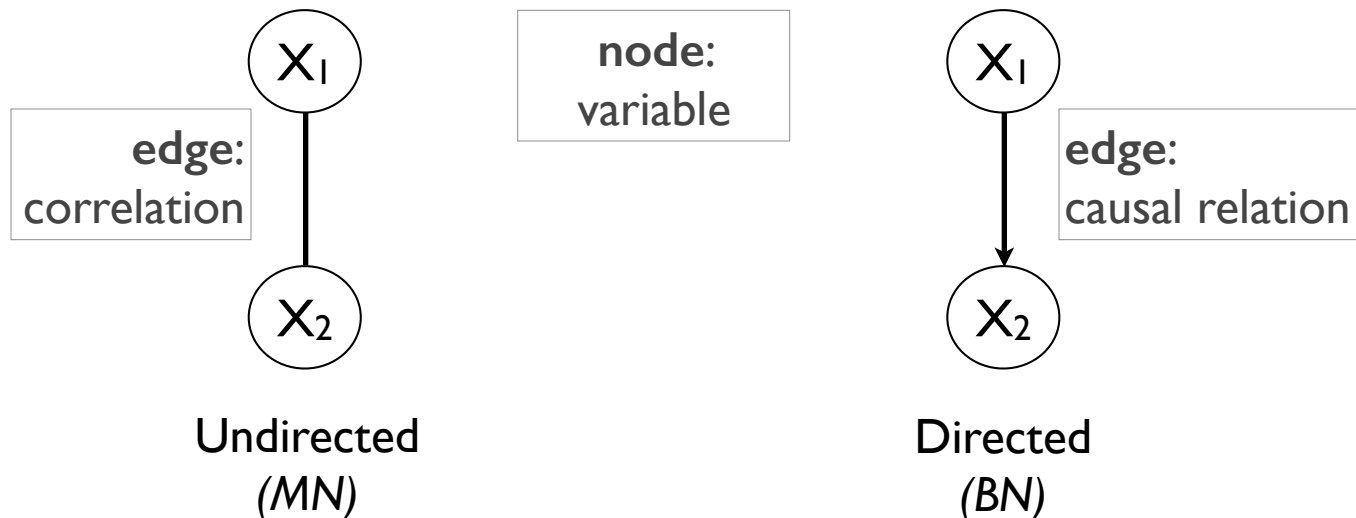
- Section agenda
  - Extensions using probabilistic graphical models (PGMs)
  - Extensions using ensemble techniques

# Extensions using PGMs

- Probabilistic Graphical Models (PGMs)
  - PGM refers to a family of distributions on a set of **random variables** that are compatible with all the **probabilistic independence propositions** encoded in **a graph**
  - A smart way to formulate exponentially large probability distribution **without paying an exponential cost**
    - **Using PGMs, we can reduce the model complexity**
  - **PGM = Multivariate statistics + Graphical structure**

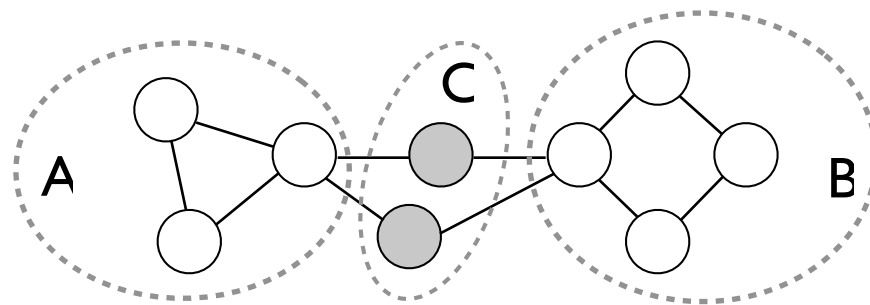
# Extensions using PGMs

- Representation: Two types
  - Undirected graphical models (UGMs)
    - Also known as Markov networks (MNs)
  - Directed graphical models (DGMs)
    - Also known as Bayesian networks (BNs)



# Extensions using PGMs

- How PGMs reduce the model complexity?
- Key idea: Exploit the **conditional independence (CI)** relations among variables!!
- Conditional independence (CI): Random variables  $A$ ,  $B$  are conditionally independent given  $C$ , if  $P(A, B | C) = P(A | C)P(B | C)$
- *UGM* and *DGM* offer a set of graphical notations for CI

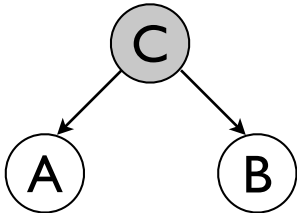
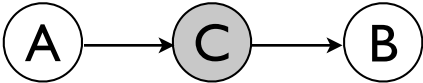
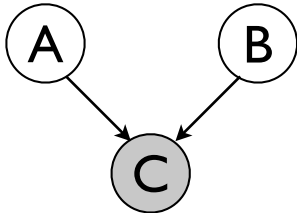


$$A \perp B \mid C$$

CI representation in *UGM*

# Extensions using PGMs

- How PGMs reduce the model complexity?
- Key idea: Exploit the **conditional independence (CI)** relations among variables!!
- Conditional independence (CI): Random variables  $A$ ,  $B$  are conditionally independent given  $C$ , if  $P(A, B | C) = P(A | C)P(B | C)$
- *UGM* and *DGM* offer a set of graphical notations for CI

		
$A \perp B \mid C$		$A \not\perp B \mid C$
CI representations in <i>DGM</i>		

## Extensions using PGMs

- PGMs have been an excellent representation / formulation tool for the MLC problems
- The dependences among features ( $\mathbf{X}$ ) and class variables ( $\mathbf{Y}$ ) can be represented easily with PGMs
- By exploiting the conditional independence, we can make the computation simpler

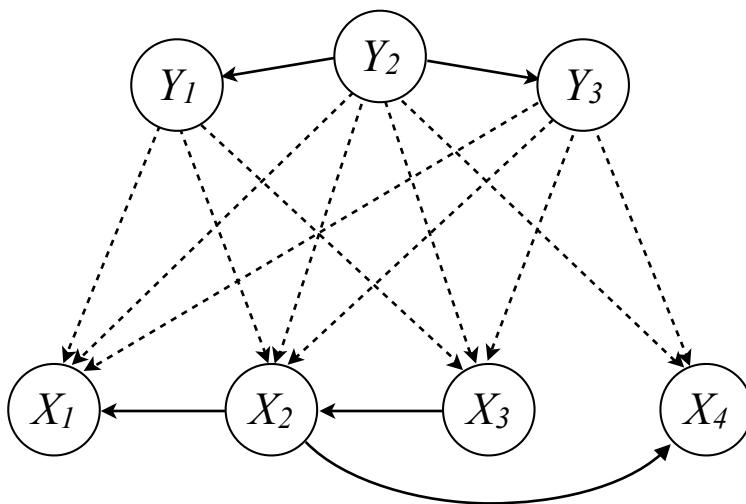


# Extensions using PGMs

- Existing methods
  - Undirected models (Markov networks)
    - Multi-label Conditional Random Field (ML-CRF) [Ghamrawi and McCallum, 2005; Pakdaman et al, 2014]
    - Composite Marginal Models (CMM) [Zhang and Schneider, 2012]
  - Directed models (Bayesian networks)
    - Multi-dimensional Bayesian Classifiers (MBC) [van der Gaag and de Waal, 2006]
    - Classifier Chains (CC) [Read et al, 2009]
    - Conditional Tree-structured Bayesian Networks (CTBN) [Batal et al, 2013]

# Multi-dimensional Bayesian Networks (MBC) [van der Gaag and de Waal, 2006]

- Key idea
  - Model the **full joint of input and output** using a **Bayesian network**
  - Use graphical structures to **represent the dependence relations** among the input and output variables
- Example MBC ( $d = 3, m = 4$ )



The joint distribution  $P(\mathbf{X}, \mathbf{Y})$  is represented by the decomposition

$$\mathbf{X} = X_1|X_2 \cdot X_2|X_3 \cdot X_3 \cdot X_4|X_2$$

and

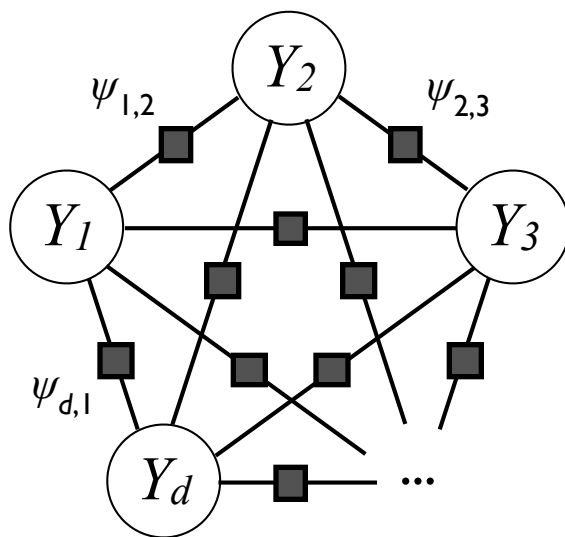
$$\mathbf{Y} = Y_1|Y_2 \cdot X_2 \cdot Y_3|Y_2$$

## Multi-dimensional Bayesian Networks (MBC) [\[van der Gaag and de Waal, 2006\]](#)

- Advantages
  - The full joint distribution of the feature and class variables can be represented efficiently using the Bayesian network
- Disadvantages
  - Models the relations among the feature variables which do not carry much information in modeling the multi-label relations

# Multi-label Conditional Random Fields (MLCRF) [Pakdaman et al, 2014]

- Key idea
  - **Model the conditionals  $P(\mathbf{Y}|\mathbf{X})$**  to capture the relations among the class variables conditioned on the feature variables
  - Learn a **pairwise Markov network** to model the relations between the input and output variables
- Representation



$$P(\mathbf{Y}|\mathbf{X}) = \frac{\prod_{i=1}^d \prod_{j=1}^d \psi_{i,j}(Y_i, Y_j, \mathbf{X}) \phi_i(Y_i, \mathbf{X})}{Z}$$

( $\psi_{i,j}$  and  $\phi_i$  are the potentials of  $Y_i, Y_j, \mathbf{X}$ ; and  $Z$  is the normalization term)

# Multi-label Conditional Random Fields (MLCRF) [Pakdaman et al, 2014]

- Advantages

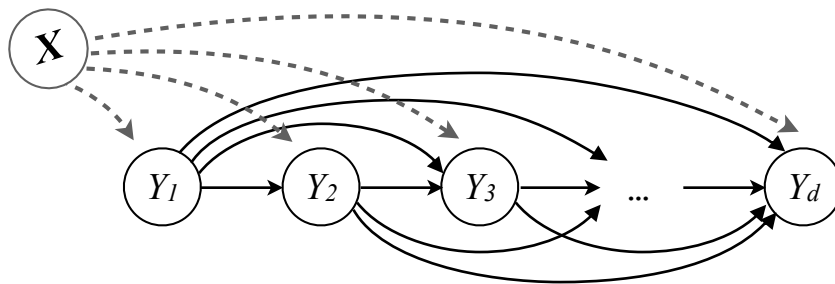
- Directly models the conditional joint distribution  $P(\mathbf{Y}|\mathbf{X})$

- Disadvantages

- Learning and prediction is computationally very demanding
  - To perform an inference, the normalization term  $Z$  should be computed, which is usually very costly
  - The iterative parameter learning process requires inference at each step whose computational cost is even more expensive
    - In practice, approximate inference techniques are applied to make the model usable

# Classifier Chains (CC) [Read et al, 2009]

- Key idea
  - Model  $P(\mathbf{Y}|\mathbf{X})$  using a directed chain network, where **all preceding classes in the chain are conditioning** the following class variables
- Representation

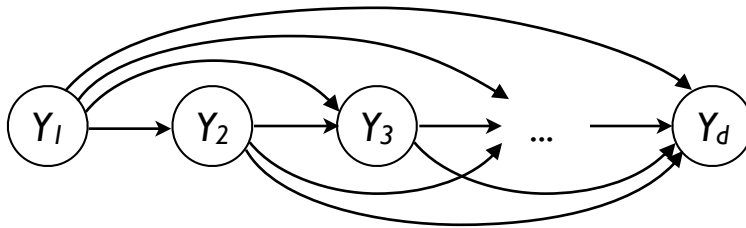


$$\begin{aligned} P(\mathbf{Y}|\mathbf{X}) &= \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i)) \\ &= \prod_{i=1}^d P(Y_i|\mathbf{X}, Y_1, \dots, Y_{i-1}) \\ &= P(Y_1|\mathbf{X}) \cdot P(Y_2|\mathbf{X}, Y_1) \cdot \dots \cdot P(Y_d|\mathbf{X}, Y_1, \dots, Y_{d-1}) \end{aligned}$$

*all preceding labels*

# Classifier Chains (CC) [Read et al, 2009]

- Learning



$$P(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i))$$

- No structure learning (random chain order)
- Parameter learning is performed on the decomposed CPDs:

$$\operatorname{argmax}_{\theta} P(Y_i|\mathbf{X}, \pi(Y_i); \theta)$$

- Prediction

- Performed by greedy maximization of each factors (CPDs):

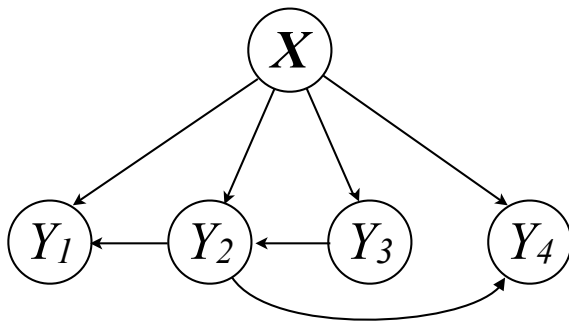
$$\operatorname{argmax}_{Y_i} P(Y_i|\mathbf{X}, \pi(Y_i); \theta)$$

# Conditional Tree-structured Bayesian Networks (CTBNs) [Batal et al, 2013]

- Key idea

- Learn  $P(\mathbf{Y}|\mathbf{X})$  using a **tree-structured Bayesian network** of the class labels
- **Tree-structures can be seen as restricted chains**, where each class variable has at most one parent class variable

- Example CTBN



$$P(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i))$$

*at most one parent label*

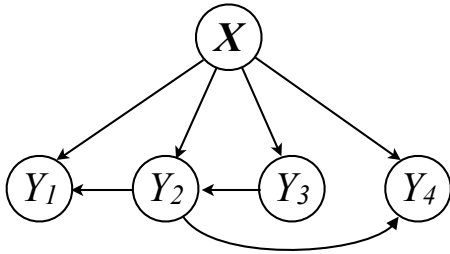
This network represents:

$$P(y_1, y_2, y_3, y_4|\mathbf{x}) = P(y_3|\mathbf{x}) \cdot P(y_2|\mathbf{x}, y_3) \cdot P(y_1|\mathbf{x}, y_2) \cdot P(y_4|\mathbf{x}, y_2)$$



# Conditional Tree-structured Bayesian Networks (CTBNs) [Batal et al, 2013]

- Learning

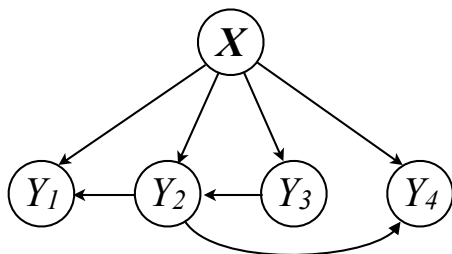


$$P(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i))$$

- Structure learning by **optimizing conditional log-likelihood**
  1. Define a **complete weighted directed** graph, whose edge weights is equal to **conditional log-likelihood**
  2. Find the **maximum branching tree** from the graph  
(\* Maximum branching tree = maximum weighted directed spanning tree)

# Conditional Tree-structured Bayesian Networks (CTBNs) [Batal et al, 2013]

- Learning

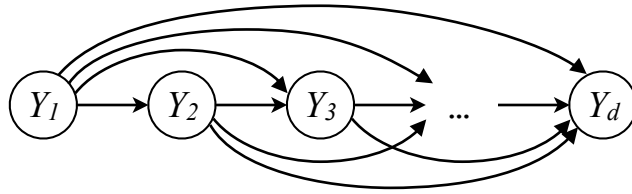


$$P(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i))$$

- Structure learning by **optimizing conditional log-likelihood**
- Parameter learning is performed on the decomposed CPDs
- Prediction
  - **Exact MAP prediction** is performed by a **belief propagation (max-product)** algorithm

# CC vs. CTBN

CC

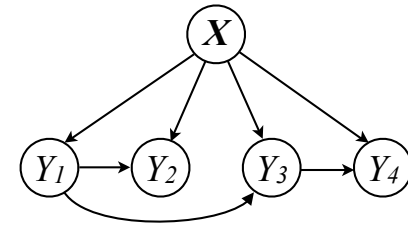


$$P(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i))$$

*all preceding labels*

- Decomposes the joint probability along with the chain structure
- No structure learning (label ordering is given at random)
- Maximizes the marginals along the chain (suboptimal solution)
- Errors in prediction propagate to the following label prediction

CTBN



$$P(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i))$$

*at most one parent label*

- Decomposes the joint probability along with the tree structure
- Tree structure is learned using a score-based algorithm
- Performs exact MAP prediction (linear time optimal solution)
- The tree-structure assumption may restrict its modeling ability

# Advanced solutions

- Section agenda
  - ✓ Extensions using probabilistic graphical models (PGMs)
- Extensions using ensemble techniques

# Extensions using ensemble techniques

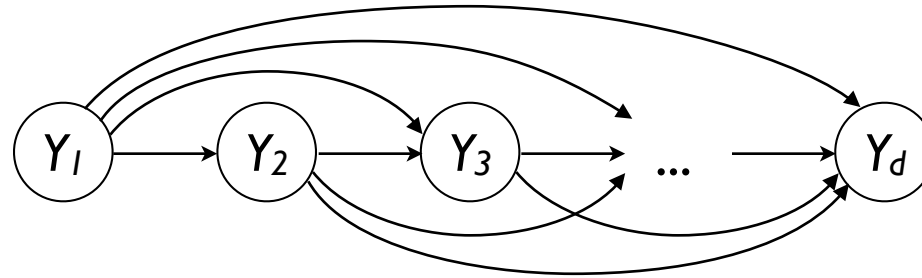
- Ensemble techniques
  - Techniques of **training multiple classifiers** and **combining their predictions** to produce a single classifier
- Ensemble techniques can further improve the performance of MLC classifiers
- Objective: Use a combination of simpler classifiers to improve predictions

# Extensions using ensemble techniques

- Existing methods
  - Ensemble of CCs (ECC) [\[Read et al, 2009\]](#)
  - Mixture of CTBNs (MC) [\[Hong et al, 2014\]](#)

# Ensemble of Classifier Chains (ECC) [Read et al, 2009]

- Recall CC



$$P(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^d P(Y_i|\mathbf{X}, \pi(Y_i))$$

- Key Idea

- Create user-specified number of CC's on **random subsets of data** with **random orderings** of the class labels
- Predict by **majority vote** over all base classifiers

# Ensemble of Classifier Chains (ECC) [Read et al, 2009]

- Advantages
  - Often times, the performance improves
- Disadvantages
  - Ad-hoc ensemble implementation
    - Learns base classifiers on random subsets of data with random label ordering
    - Ensemble decisions are made by simple averaging over the base models and often inaccurate



# Mixture of CTBNs (MC) [Hong et al, 2014]

- Motivation
  - If the underlying dependency structure in data is **more complex than a tree structure**, a single CTBN cannot model the data properly
- Key idea
  - Use the *Mixtures-of-Trees* [Meila and Jordan, 2000] framework to learn multiple CTBNs and use them for prediction

# Mixture of CTBNs (MC) [Hong et al, 2014]

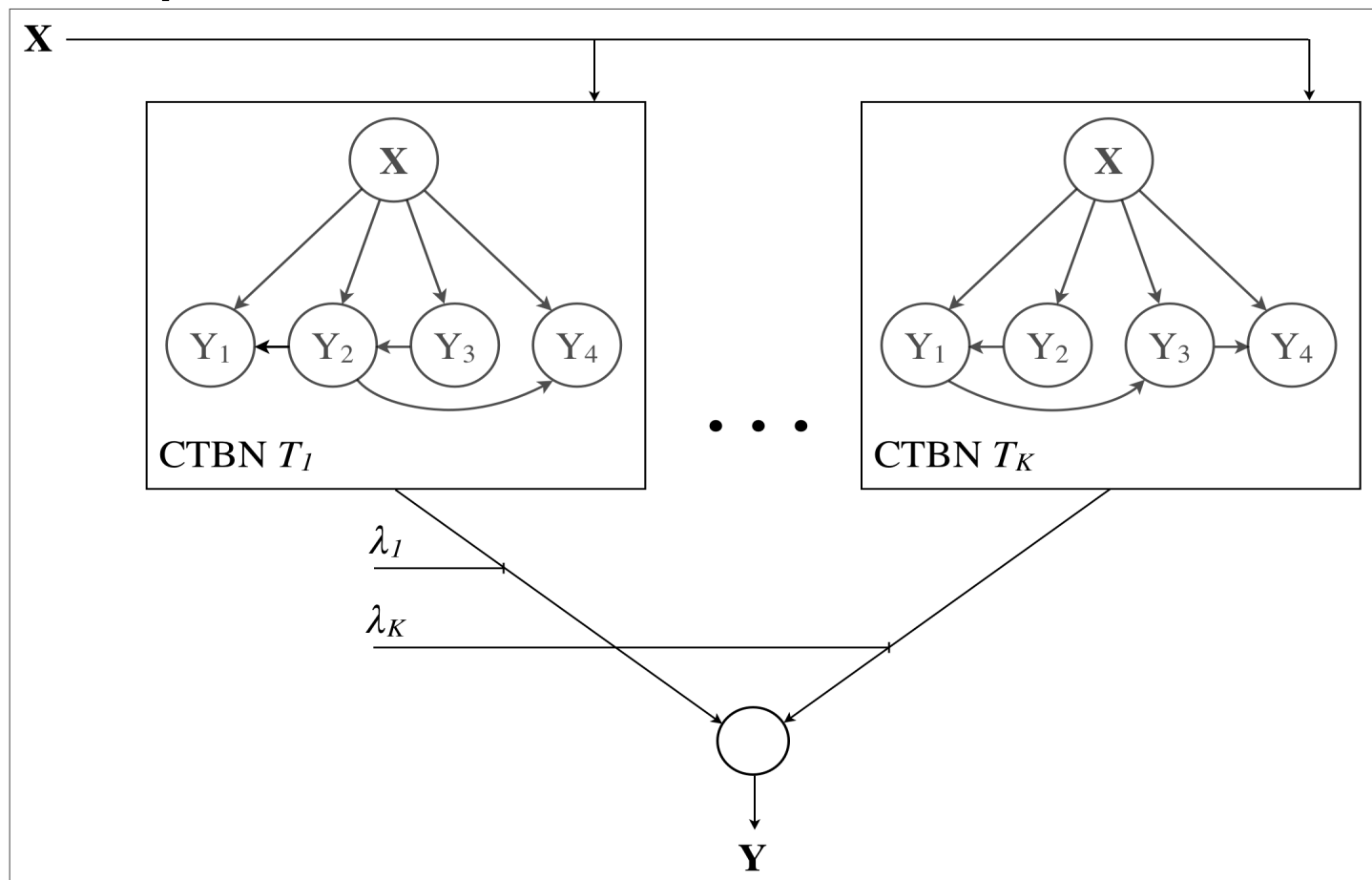
- MC defines the multivariate posterior distribution of class vector  $P(\mathbf{y}|\mathbf{x}) = P(y_1, \dots, y_d|\mathbf{x})$  as

$$\begin{aligned} P(\mathbf{y}|\mathbf{x}) &= \sum_{k=1}^K \lambda_k P(\mathbf{y}|\mathbf{x}, T_k) \\ &= \sum_{k=1}^K \lambda_k \prod_{i=1}^d P(y_i|\mathbf{x}, y_{\pi(i,T)}) \end{aligned}$$

- $P(\mathbf{y}|\mathbf{x}, T_k)$  is the  $k$ -th **mixture component** defined by a CTBN  $T_k$
- $\lambda_k$  is the **mixture coefficient** representing the weight of the  $k$ -th component (influence of the  $k$ -th CTBN model  $T_k$  to the mixture)

# Mixture of CTBNs (MC) [Hong et al, 2014]

- An example MC



$$P(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^K \lambda_k P(\mathbf{y}|\mathbf{x}, T_k)$$

# Mixture of CTBNs (MC) [Hong et al, 2014]

- Parameter learning
  - Objective: Optimize the **model parameters** (CTBN parameters  $\{\theta_1, \dots, \theta_K\}$  and mixture coefficients  $\{\lambda_1, \dots, \lambda_K\}$ )
  - Idea (**apply EM**)
    1. Associate each instance  $(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})$  with a **hidden variable**  $z^{(n)} \in \{1, \dots, K\}$  indicating **which CTBN it belongs to**.
    2. Iteratively optimize the **expected complete log-likelihood**:

$$E \left[ \sum_{n=1}^N \log P(\mathbf{y}^{(n)}, z^{(n)} | \mathbf{x}^{(n)}) \right]$$
$$= E \left[ \sum_{n=1}^N \sum_{k=1}^K 1[z^{(n)} = k] [\log \lambda_k + \log P(\mathbf{y}^{(n)} | \mathbf{x}^{(n)}, T_k)] \right]$$

# Mixture of CTBNs (MC) [Hong et al, 2014]

- Structure learning
  - Objective: Find **multiple CTBN structures** from data
  - Idea (**boosting-like heuristic**)
    1. On each addition of a new structure to the mixture, **recalculate the weight of each data instance ( $\omega$ )** such that it represents the relative “hardness” of the instance
    2. Learn the best tree structure by **optimizing the weighted conditional log-likelihood**:

$$\sum_{n=1}^N \sum_{i=1}^d \omega^{(n)} \log P(y_i^{(n)} | \mathbf{x}^{(n)}, y_{\pi(i,T)}^{(n)})$$

# Mixture of CTBNs (MC) [Hong et al, 2014]

- Prediction
  - Objective: Find the **maximum a posteriori (MAP)** prediction for a new instance  $\mathbf{x}$
  - Idea
    1. Search the space of all class assignments by defining a Markov chain
    2. Use an annealed version of exploration procedure to speed up the search

# Mixture of CTBNs (MC) [Hong et al, 2014]

- Advantages
  - Learns an ensemble model for MLC in a principled way
  - Produces accurate and reliable results
- Disadvantages
  - Iterative optimization process in learning requires a large amount of time

# Agenda

- ✓ Motivation
- ✓ Solutions
- ✓ Advanced solutions
- Evaluation metrics
- Toolboxes
- Summary



# Evaluation metrics

- Evaluation of MLC methods is more difficult than that of single-label classification
- Measuring the **Hamming accuracy is not sufficient** for the goal of MLC

- Hamming accuracy (HA) =  $\frac{1}{N} \sum_{n=1}^N \llbracket h(\mathbf{x}^{(n)}) \Delta \mathbf{y}^{(n)} \rrbracket$

- HA measures the individual accuracy on each class variable, which can be optimized by the binary relevance (BR) model

- We want to find “**jointly accurate**” class assignments

- We want to measure if the model predicts all the labels correctly

- Exact match accuracy (EMA) =  $\frac{1}{N} \sum_{n=1}^N \llbracket h(\mathbf{x}^{(n)}) = \mathbf{y}^{(n)} \rrbracket$

# Evaluation metrics

- Exact match accuracy (EMA) =  $\frac{1}{N} \sum_{n=1}^N \mathbb{I}[h(\mathbf{x}^{(n)}) = \mathbf{y}^{(n)}]$ 
  - EMA evaluate if the prediction is correct on all class variables
  - Most appropriate metric for MLC
    - We are looking for the most probable assignment of classes
  - It can be too strict
- Multi-label accuracy (MLA) =  $\frac{1}{N} \sum_{n=1}^N \frac{h(\mathbf{x}^{(n)}) \cap \mathbf{y}^{(n)}}{h(\mathbf{x}^{(n)}) \cup \mathbf{y}^{(n)}}$ 
  - MLA evaluate the Jaccard index between prediction and true class assignments
  - It is less strict than EMA; overestimates the model accuracy

# Evaluation metrics

- Conditional log-likelihood loss (CLL-loss)

- $\text{CLL-loss} = \sum_{n=1}^N -\log P(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}; M)$

- Reflects the model fitness

- F1-scores: harmonics mean of precision and recall

- $\text{Micro F1} = \frac{1}{N} \sum_{n=1}^N \frac{2 \times TP^{(n)}}{2 \times TP^{(n)} + FP^{(n)} + FN^{(n)}}$

- Computes the F1-score on each instance and then average out

- $\text{Macro F1} = \frac{1}{d} \sum_{i=1}^d \frac{2 \times TP^{(i)}}{2 \times TP^{(i)} + FP^{(i)} + FN^{(i)}}$

- Computes the F1-score on each class and then average out

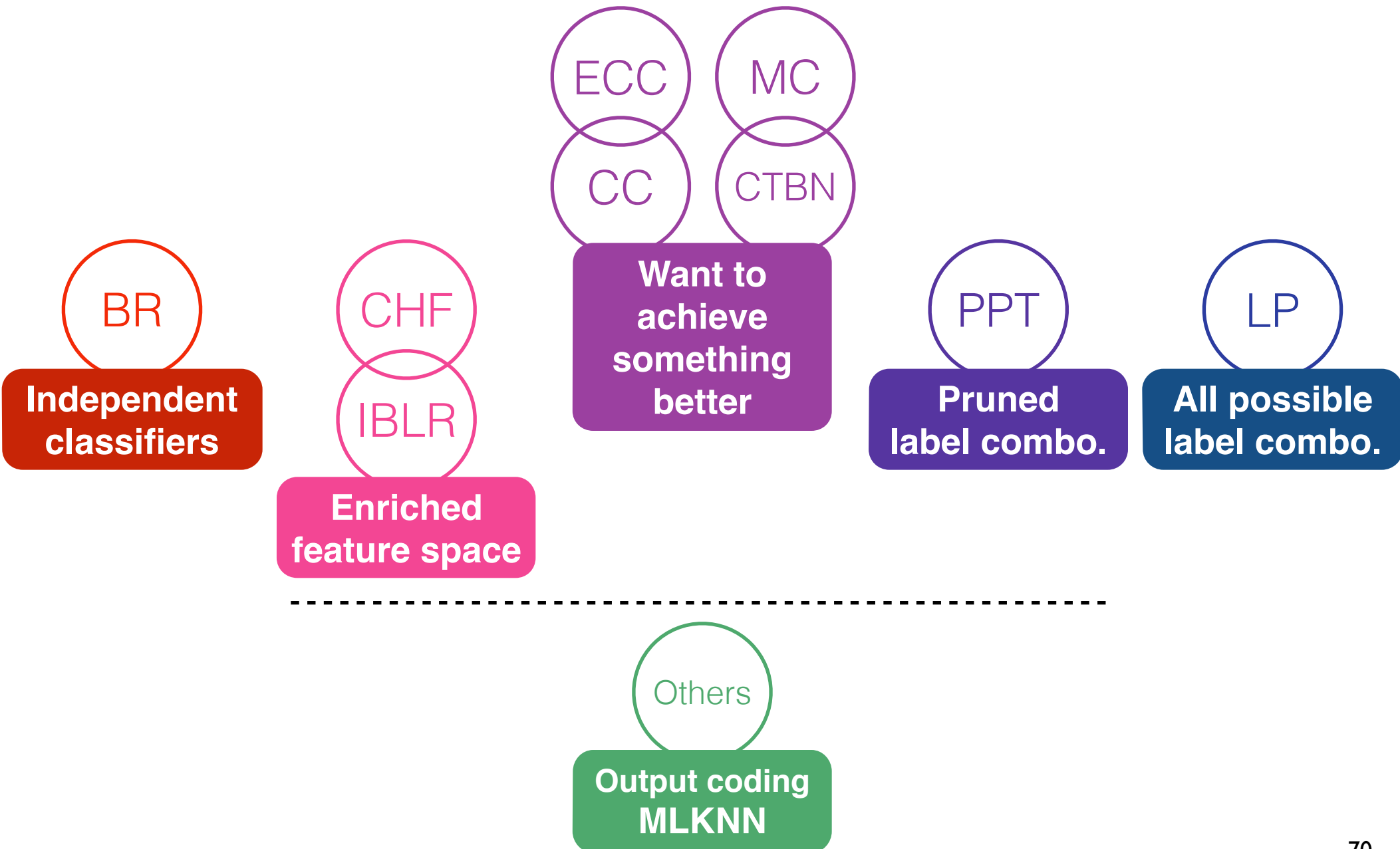
# Agenda

- ✓ Motivation
- ✓ Solutions
- ✓ Advanced solutions
- ✓ Evaluation metrics
- **Toolboxes**
- **Summary**

# Toolboxes

- **MEKA**: a Multi-label Extension to WEKA  
<http://meka.sourceforge.net/>
- **Mulan**: a Java library for Multi-label Learning  
<http://mulan.sourceforge.net/>
- **LibSVM MLC Extension** (BR and LP)  
<http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/multilabel/>
- **LAMDA Lab** (Nanjing Univ., China) Code Repository  
[http://lamda.nju.edu.cn/Default.aspx?  
Page=Data&NS=&AspxAutoDetectCookieSupport=1](http://lamda.nju.edu.cn/Default.aspx?Page=Data&NS=&AspxAutoDetectCookieSupport=1)
- **Prof. Min-Ling Zhang** (Southeast Univ., China)  
<http://cse.seu.edu.cn/old/people/zhangml/Resources.htm#codes>

# Summary



# References

- [Batal et al, 2013] I. Batal, C. Hong, and M. Hauskrecht. “An efficient probabilistic framework for multi-dimensional classification”. In: Proceedings of the 22nd ACM international conference on Conference on information and knowledge management (CIKM). 2013, pp. 2417–2422.
- [Bose & Ray-Chaudhuri 1960] R.C. Bose, D.K. Ray-Chaudhuri. On a class of error correcting binary group codes. In: Inform and Control, 3. 1960, pp. 68–79.
- [Boutell et al, 2004] M. R. Boutell et al. “Learning Multi-label Scene Classification”. In: Pattern Recognition 37.9 (2004)
- [Cheng and Hüllermeier, 2009] W. Cheng and E. Hüllermeier. “Combining instance-based learning and logistic regression for multilabel classification”. In: Machine Learning 76.2-3 (2009)
- [Clare and King, 2001] A. Clare and R. D. King. “Knowledge Discovery in Multi-Label Pheno- type Data”. In: Lecture Notes in Computer Science. Springer, 2001.
- [Dietterich, 1995] T. G. Dietterich and G. Bakiri. "Solving Multiclass Learning Problems via Error-Correcting Output Codes", In: Journal of Artificial Intelligence Research. 1995. Volume 2, pages 263-286.
- [Donoho, 2006] D. Donoho, “Compressed sensing,” IEEE Trans. Inform. Theory, vol. 52, no. 4, pp. 1289–1306, April 2006.

# References

- [Ghamrawi and McCallum, 2005] N. Ghamrawi and A. McCallum. “Collective multi-label classification”. In: Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM). 2005, pp. 195–200.
- [Godbole et al, 2004] S. Godbole and S. Sarawagi. “Discriminative Methods for Multi-labeled Classification”. In: PAKDD’04. 2004, pp. 22–30.
- [Hong et al, 2014] C. Hong, I. Batal, and M. Hauskrecht. “A mixtures-of-trees framework for multi-label classification”. In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management (CIKM), 2014. ACM.
- [Hsu et al, 2009] [Hsu et al, 2009] D. Hsu et al. “Multi-Label Prediction via Compressed Sensing”. In: NIPS. 2009, pp. 772– 780.
- [Johnson & Wichern, 2002] R. A. Johnson and D. W. Wichern. “Applied Multivariate Statistical Analysis” (5th Ed.). 2002. Upper Saddle River, N.J., Prentice-Hall.
- [Meila and Jordan, 2000] M. Meila and M. I. Jordan. “Learning with mixtures of trees”. Journal of Machine Learning Research, 1:1–48, 2000.
- [Pakdaman et al, 2014] M. Pakdaman, I. Batal, Z. Liu, C. Hong, and M. Hauskrecht. “An optimization-based framework to learn conditional random fields for multi-label classification”. In SDM. SIAM, 2014.



# References

- [Read et al, 2008] J. Read, B. Pfahringer, and G. Holmes. “Multi-label Classification Using Ensembles of Pruned Sets”. In: ICDM. IEEE Computer Society, 2008, pp. 995–1000.
- [Read et al, 2009] J. Read et al. “Classifier Chains for Multi-label Classification”. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II. ECML PKDD '09. Bled, Slovenia: Springer-Verlag, 2009, pp. 254–269.
- [Tai and Lin, 2010] Farbound Tai and Hsuan-Tien Lin. “Multi-label Classification with Principle Label Space Transformation”. In: Proceedings of the 2nd International Workshop on Multi-Label Learning. 2010.
- [van der Gaag and de Waal, 2006] L. C. van der Gaag and P. R. de Waal. “Multi-dimensional Bayesian Network Classifiers”. In: Probabilistic Graphical Models. 2006, pp. 107–114
- [Zhang and Schneider, 2012a] Y. Zhang and J. Schneider. “A Composite Likelihood View for Multi-Label Classification”. In: AISTATS (2012).
- [Zhang and Schneider, 2012b] Y. Zhang and J. Schneider. “Maximum Margin Output Coding”. In: Proceedings of the 29th International Conference on Machine Learning (ICML-12). ICML '12. Edinburgh, Scotland, UK: Omnipress, 2012, pp. 1575–1582.
- [Zhang and Zhou, 2007] Min-Ling Zhang and Zhi-Hua Zhou. “ML-KNN: A lazy learning approach to multi-label learning”. In: Pattern Recogn. 40.7 (July 2007), pp. 2038–2048.

**Thanks!**