Label tree algorithms for extreme classification

Krzysztof Dembczyński

Poznan University of Technology, Poland

UGent Data Science Seminars, Ghent, June 13, 2019

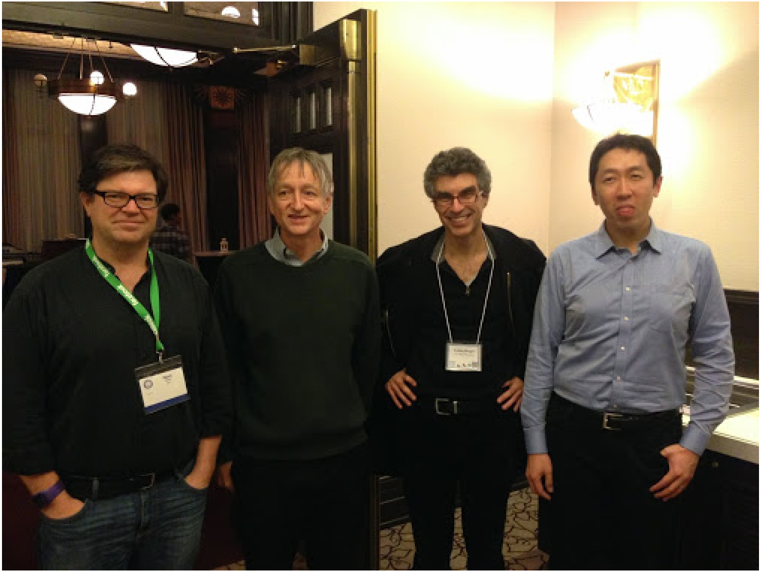
Outline

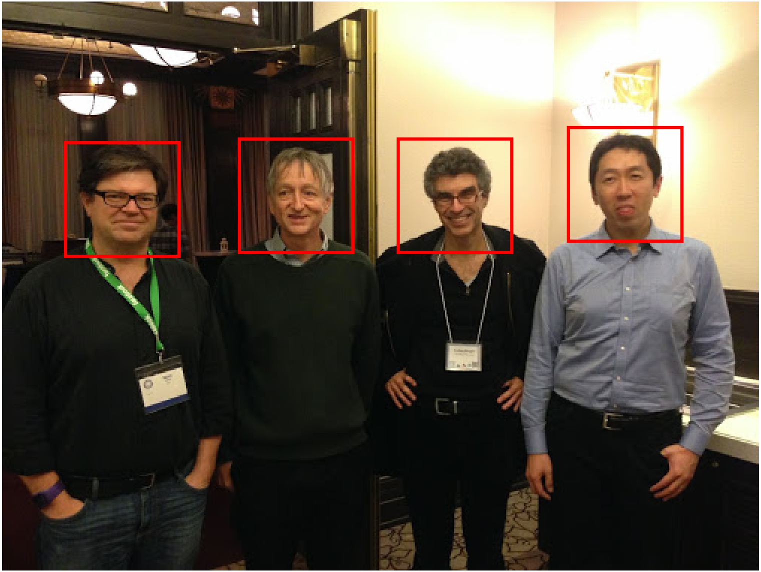
1. Extreme multi-label classification: applications and challenges
2. Theoretical framework
3. Tree-based algorithms: decision and label trees
4. Take-away message

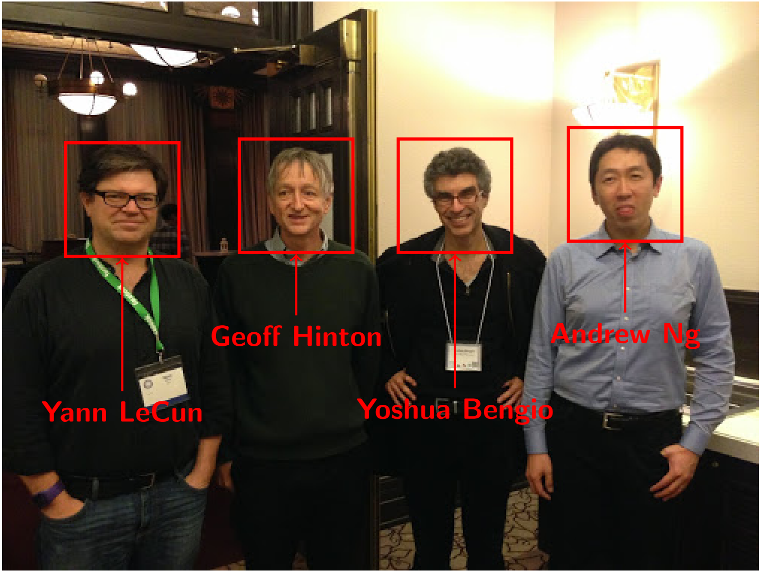
Outline

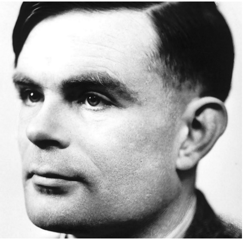
1. Extreme multi-label classification: applications and challenges
2. Theoretical framework
3. Tree-based algorithms: decision and label trees
4. Take-away message

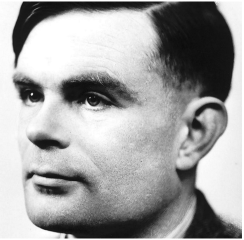
Extreme multi-label classification is a problem of **labeling** an item with a **small** set of tags out of an **extremely large** number of potential tags



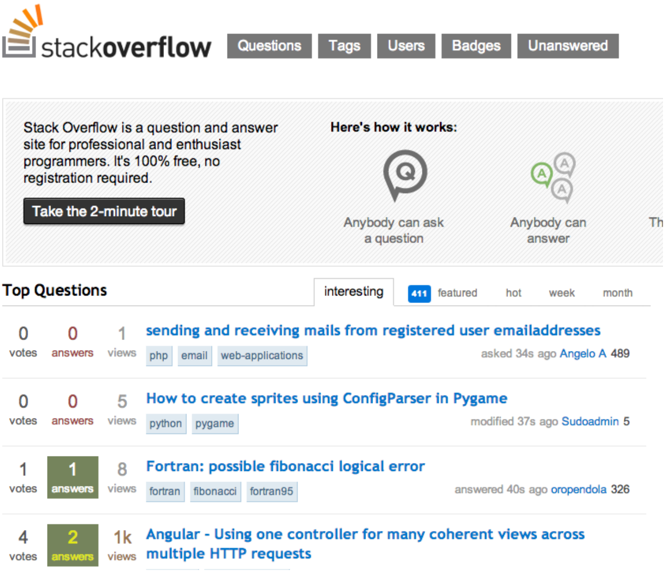



Yann LeCun
Geoff Hinton
Yoshua Bengio
Andrew Ng

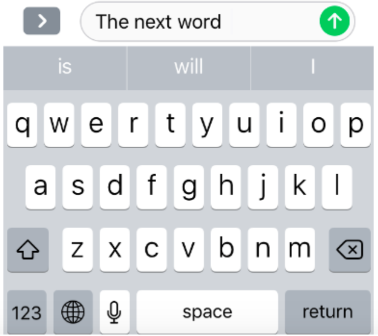




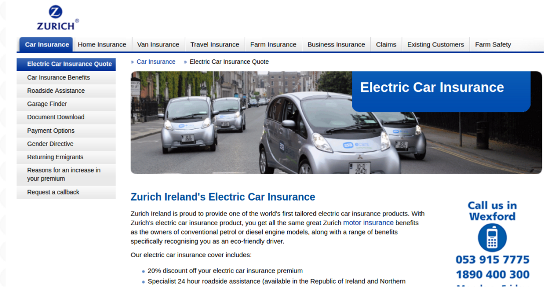
Alan Turing, 1912 births, 1954 deaths
20th-century mathematicians, 20th-century philosophers
Academics of the University of Manchester Institute of Science and Technology
Alumni of King's College, Cambridge Artificial intelligence researchers
Atheist philosophers, Bayesian statisticians, British cryptographers, British logicians
British long-distance runners, British male athletes, British people of World War II
Computability theorists, Computer designers, English atheists
English computer scientists, English inventors, English logicians
English long-distance runners, English mathematicians
English people of Scottish descent, English philosophers, Former Protestants
Fellows of the Royal Society, Gay men
Government Communications Headquarters people, History of artificial intelligence
Inventors who committed suicide, LGBT scientists
LGBT scientists from the United Kingdom, Male long-distance runners
Mathematicians who committed suicide, Officers of the Order of the British Empire
People associated with Bletchley Park, People educated at Sherborne School
People from Maida Vale, People from Wilmslow
People prosecuted under anti-homosexuality laws, Philosophers of mind
Philosophers who committed suicide, Princeton University alumni, 1930-39
Programmers who committed suicide, People who have received posthumous pardons
Recipients of British royal pardons, Academics of the University of Manchester
Suicides by cyanide poisoning, Suicides in England, Theoretical computer scientists



New question ⇒ Assignment/recommendation of users



Sequence of words $\Rightarrow$ Recommendation of the next word



Possible bid phrases:

- Zurich car insurance
- Car insurance
- Auto insurance
- Vehicle insurance
- Electric car insurance

On-line ad $\Rightarrow$ Recommendation of queries to an advertiser

Setting

- **Multi-class classification**:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d \xrightarrow{h(\boldsymbol{x})} y \in \{1, \ldots, m\}$$

	x_1	x_2	$\ldots$	x_d	y
$\boldsymbol{x}$	4.0	2.5		-1.5	5

Setting

- **Multi-class classification**:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d \xrightarrow{h(\boldsymbol{x})} y \in \{1, \ldots, m\}$$

	x_1	x_2	$\ldots$	x_d	y
$\boldsymbol{x}$	4.0	2.5		-1.5	5

- **Multi-label classification**:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d \xrightarrow{\boldsymbol{h}(\boldsymbol{x})} \boldsymbol{y} = (y_1, y_2, \ldots, y_m) \in \{0, 1\}^m$$

	x_1	x_2	$\ldots$	x_d	y_1	y_2	$\ldots$	y_m
$\boldsymbol{x}$	4.0	2.5		-1.5	1	1		0

Extreme classification

$\boxed{\textbf{Extreme classification} \Rightarrow \text{a } \textbf{large} \text{ number of } \textbf{labels } m \ (\geq 10^5)}$

Extreme classification

Extreme classification $\Rightarrow$ a **large** number of **labels** m $(\geq 10^5)$

$\Downarrow$

Computational and statistical challenges

Extreme classification: Computational challenges

- **Computational complexity**:

Extreme classification: Computational challenges

- **Computational complexity**:
  - Naive one-vs-all approach (a dense linear model for each label):

$$\hat{\boldsymbol{y}} = [\![\mathbf{W}\boldsymbol{x} > 0]\!]$$

Problem size:	$n > 10^6$, $d > 10^6$, $m > 10^5$
	$\Downarrow$
Complexity:	training time $> 10^{17}$
	space $> 10^{11}$
	test time $> 10^{11}$

Extreme classification: Computational challenges

- **Computational complexity**:
  - Naive one-vs-all approach (a dense linear model for each label):

$$\hat{\boldsymbol{y}} = [\![\mathbf{W}\boldsymbol{x} > 0]\!]$$

Problem size:	$n > 10^6$, $d > 10^6$, $m > 10^5$
	$\Downarrow$
Complexity:	training time $> 10^{17}$
	space $> 10^{11}$
	test time $> 10^{11}$

  - time vs. space

Extreme classification: Computational challenges

- **Computational complexity**:
  - Naive one-vs-all approach (a dense linear model for each label):

$$\hat{\boldsymbol{y}} = [\![\mathbf{W}\boldsymbol{x} > 0]\!]$$

Problem size:	$n > 10^6$, $d > 10^6$, $m > 10^5$
	$\Downarrow$
Complexity:	training time $> 10^{17}$
	space $> 10^{11}$
	test time $> 10^{11}$

  - time vs. space
  - #examples vs. #features vs. #labels

Extreme classification: Computational challenges

- **Computational complexity**:
  - Naive one-vs-all approach (a dense linear model for each label):

$$\hat{\boldsymbol{y}} = [\![\mathbf{W}\boldsymbol{x} > 0]\!]$$

Problem size:	$n > 10^6$, $d > 10^6$, $m > 10^5$
	$\Downarrow$
Complexity:	training time $> 10^{17}$
	space $> 10^{11}$
	test time $> 10^{11}$

  - time vs. space
  - #examples vs. #features vs. #labels
  - training vs. validation vs. prediction

Extreme classification: Computational challenges

- **It does not have to be so hard**:

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available
  - Large data $\longrightarrow$ sparse data (sparse features and labels)

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available
  - Large data $\longrightarrow$ sparse data (sparse features and labels)
  - Fast learning algorithms for standard learning problems exist

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available
  - Large data $\longrightarrow$ sparse data (sparse features and labels)
  - Fast learning algorithms for standard learning problems exist
- **New algorithmic solutions**:

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available
  - Large data $\longrightarrow$ sparse data (sparse features and labels)
  - Fast learning algorithms for standard learning problems exist
- **New algorithmic solutions**:
  - Smart 1-vs-All approaches

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available
  - Large data $\longrightarrow$ sparse data (sparse features and labels)
  - Fast learning algorithms for standard learning problems exist
- **New algorithmic solutions**:
  - Smart 1-vs-All approaches
  - Label filtering/maximum inner product search (MIPS)

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available
  - Large data $\longrightarrow$ sparse data (sparse features and labels)
  - Fast learning algorithms for standard learning problems exist
- **New algorithmic solutions**:
  - Smart 1-vs-All approaches
  - Label filtering/maximum inner product search (MIPS)
  - Embeddings

Extreme classification: Computational challenges

- **It does not have to be so hard**:
  - High performance computing resources available
  - Large data $\longrightarrow$ sparse data (sparse features and labels)
  - Fast learning algorithms for standard learning problems exist
- **New algorithmic solutions**:
  - Smart 1-vs-All approaches
  - Label filtering/maximum inner product search (MIPS)
  - Embeddings
  - **Tree-based methods**

Extreme classification: Statistical challenges

- **Predictive performance**:

Extreme classification: Statistical challenges

- **Predictive performance**:
  - Performance measures: Hamming loss, prec@k, NDCG@k, Macro F?

Extreme classification: Statistical challenges

- **Predictive performance**:
  - Performance measures: Hamming loss, prec@k, NDCG@k, Macro F?
  - Learning theory for large m

Extreme classification: Statistical challenges

- **Predictive performance**:
  - Performance measures: Hamming loss, prec$@k$, NDCG$@k$, Macro F?
  - Learning theory for large m
  - Training and prediction under limited time and space budged

Extreme classification: Statistical challenges

- **Predictive performance**:
  - Performance measures: Hamming loss, prec@k, NDCG@k, Macro F?
  - Learning theory for large m
  - Training and prediction under limited time and space budged
  - Unreliable learning information: weak, missing, or only positive labels

Extreme classification: Statistical challenges

- **Predictive performance**:
  - Performance measures: Hamming loss, prec@k, NDCG@k, Macro F?
  - Learning theory for large m
  - Training and prediction under limited time and space budged
  - Unreliable learning information: weak, missing, or only positive labels
  - Long-tail label distributions and zero-shot learning

Extreme classification: Statistical challenges

- Unreliable learning information:

Extreme classification: Statistical challenges

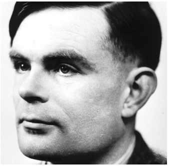
- Unreliable learning information:
  - We **cannot** expect that all labels will be properly **checked** and **assigned** to training examples

Extreme classification: Statistical challenges

- Unreliable learning information:
  - We **cannot** expect that all labels will be properly **checked** and **assigned** to training examples
  - This leads to learning with **weak**, **missing** or only **positive** labels.

Extreme classification: Statistical challenges

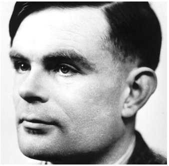
- Unreliable learning information:
  - We **cannot** expect that all labels will be properly **checked** and **assigned** to training examples
  - This leads to learning with **weak**, **missing** or only **positive** labels.



Alan Turing, 1912 births, 1954 deaths,
20th-century mathematicians, 20th-century philosophers
Academics of the University of Manchester Institute of Science and Technology
Alumni of King's College, Cambridge Artificial intelligence researchers
Atheist philosophers, Bayesian statisticians, British cryptographers, British logicians
British long-distance runners, British male athletes, British people of World War II
Computability theorists, Computer designers, English atheists
...

Extreme classification: Statistical challenges

- Unreliable learning information:
  - We **cannot** expect that all labels will be properly **checked** and **assigned** to training examples
  - This leads to learning with **weak**, **missing** or only **positive** labels.

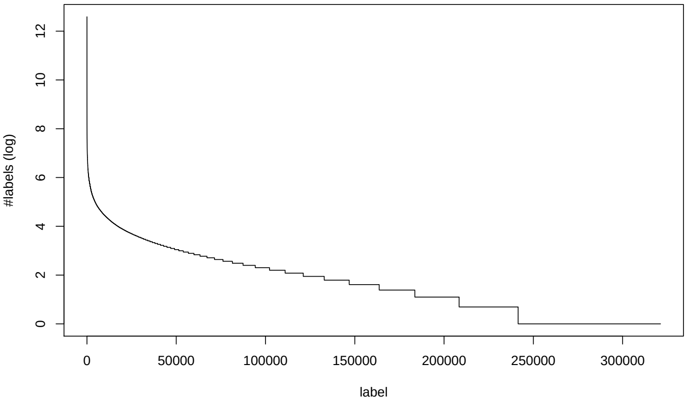


Alan Turing, 1912 births, 1954 deaths,
20th-century mathematicians, 20th-century philosophers
Academics of the University of Manchester Institute of Science and Technology
Alumni of King's College, Cambridge Artificial intelligence researchers
Atheist philosophers, Bayesian statisticians, British cryptographers, British logicians
British long-distance runners, British male athletes, British people of World War II
Computability theorists, Computer designers, English atheists
...

Category **Enigma machine** not assigned!

Extreme classification: Statistical challenges

- Long-tail label distributions and zero-shot learning:
  - Frequency of labels in the WikiLSHTC dataset:[1]



  - Many labels with only few examples ($\Rightarrow$ one- and zero-shot learning)

[1] http://manikvarma.org/downloads/XC/XMLRepository.html

Outline

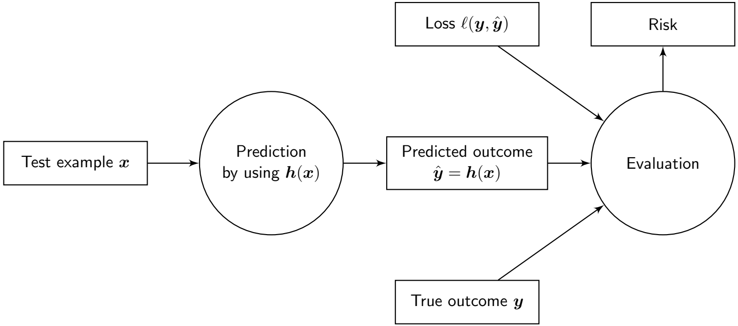
1 Extreme multi-label classification: applications and challenges

2 Theoretical framework

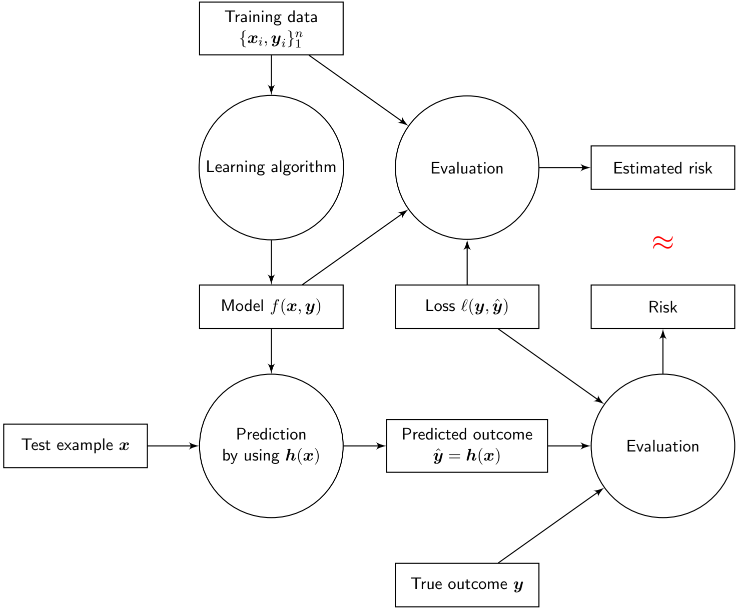
3 Tree-based algorithms: decision and label trees

4 Take-away message

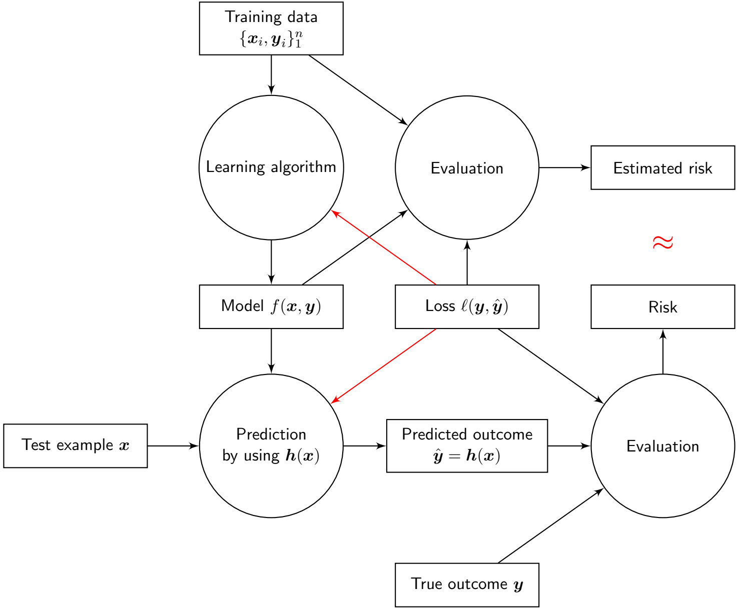
Formal setting



Formal setting



Formal setting



Formal setting

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $\mathbf{P}(\boldsymbol{x})$
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$

Formal setting

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $\mathbf{P}(\boldsymbol{x})$
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$
- **Outcome** $\boldsymbol{y} \in \mathcal{Y}$ drawn from a distribution $\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$
  - a vector of labels $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$

Formal setting

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $\mathbf{P}(\boldsymbol{x})$
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$
- **Outcome** $\boldsymbol{y} \in \mathcal{Y}$ drawn from a distribution $\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$
  - a vector of labels $\boldsymbol{y} = (y_1, y_2, \dots, y_m)$
- **Prediction** $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$ by means of **prediction function** $\boldsymbol{h} \colon \mathcal{X} \to \mathcal{Y}$
  - $\boldsymbol{h}$ returns prediction $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$ for every input $\boldsymbol{x}$

Formal setting

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $\mathbf{P}(\boldsymbol{x})$
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$
- **Outcome** $\boldsymbol{y} \in \mathcal{Y}$ drawn from a distribution $\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$
  - a vector of labels $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$
- **Prediction** $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$ by means of **prediction function** $\boldsymbol{h}\colon \mathcal{X} \to \mathcal{Y}$
  - $\boldsymbol{h}$ returns prediction $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$ for every input $\boldsymbol{x}$
- **Loss** of our prediction: $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})$
  - $\ell\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a task-specific **loss function**

Formal setting

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $\mathbf{P}(\boldsymbol{x})$
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$
- **Outcome** $\boldsymbol{y} \in \mathcal{Y}$ drawn from a distribution $\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$
  - a vector of labels $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$
- **Prediction** $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$ by means of **prediction function** $\boldsymbol{h} \colon \mathcal{X} \to \mathcal{Y}$
  - $\boldsymbol{h}$ returns prediction $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$ for every input $\boldsymbol{x}$
- **Loss** of our prediction: $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})$
  - $\ell \colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a task-specific **loss function**
- **Goal**: find a prediction function with small loss

Formal setting

- **Goal**: minimize the **expected** loss over all examples (**risk**):

$$L_\ell(\boldsymbol{h}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathbf{P}}\left[\ell(\boldsymbol{y},\boldsymbol{h}(\boldsymbol{x}))\right] = \mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}\left[\ell(\boldsymbol{y},\boldsymbol{h}(\boldsymbol{x}))\right]$$

Formal setting

- **Goal**: minimize the **expected** loss over all examples (**risk**):

$$L_\ell(\boldsymbol{h}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim \mathbf{P}}\left[\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))\right] = \mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}\left[\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))\right]$$

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

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

Formal setting

- **Goal**: minimize the **expected** loss over all examples (**risk**):

$$L_\ell(\boldsymbol{h}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim \mathbf{P}}\left[\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))\right] = \mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}\left[\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))\right]$$

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

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

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

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

Decision-theoretic recipe for learning algorithms

1. For the loss function of interest derive the form of the Bayes classifier

Decision-theoretic recipe for learning algorithms

1. For the loss function of interest derive the form of the Bayes classifier
2. Identify the quantities needed to compute the Bayes classifier

Decision-theoretic recipe for learning algorithms

1. For the loss function of interest derive the form of the Bayes classifier
2. Identify the quantities needed to compute the Bayes classifier
3. Design an algorithm for estimating this quantities

Decision-theoretic recipe for learning algorithms

1. For the loss function of interest derive the form of the Bayes classifier
2. Identify the quantities needed to compute the Bayes classifier
3. Design an algorithm for estimating this quantities
4. For a test example, compute the estimates and plug-in into the Bayes classifier

Hamming loss

- **Hamming loss**:

$$\ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{1}{m} \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!]$$

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

Hamming loss

- **Hamming loss**:

$$\ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{1}{m} \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!]$$

- Sparse labels $\Rightarrow$ Hamming loss of an **all-zero** classifier close to **0**

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

Hamming loss

- **Hamming loss**:

$$\ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{1}{m}\sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!]$$

- Sparse labels $\Rightarrow$ Hamming loss of an **all-zero** classifier close to **0**
- **The optimal strategy**:[2]

$$h_j^*(\boldsymbol{x}) = [\![\eta_j(\boldsymbol{x}) > 0.5]\!] \,,$$

where $\eta_j(\boldsymbol{x}) = \mathbf{P}(y_j = 1 \,|\, \boldsymbol{x})$

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

Hamming loss

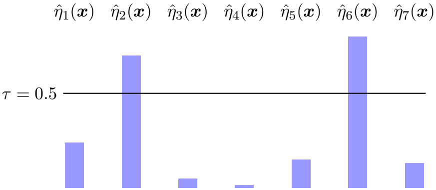
- **Hamming loss**:

$$\ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{1}{m} \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!]$$

- Sparse labels $\Rightarrow$ Hamming loss of an **all-zero** classifier close to **0**
- **The optimal strategy**:[2]

$$h_j^*(\boldsymbol{x}) = [\![\eta_j(\boldsymbol{x}) > 0.5]\!] \,,$$

where $\eta_j(\boldsymbol{x}) = \mathbf{P}(y_j = 1 \,|\, \boldsymbol{x})$



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

Hamming loss

- **Proof**:

$$L_H(\boldsymbol{h} \,|\, \boldsymbol{x}) \quad = \quad \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x}))$$

Hamming loss

- **Proof**:

$$
\begin{aligned}
L_H(\boldsymbol{h} \,|\, \boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) \\
&= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!]
\end{aligned}
$$

Hamming loss

- **Proof**:

$$\begin{aligned} L_H(\boldsymbol{h} \,|\, \boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) \\ &= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!] \\ &= \sum_{j=1}^{m} \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) [\![y_j \neq h_j(\boldsymbol{x})]\!] \end{aligned}$$

Swapping sums

Hamming loss

- **Proof**:

$$
\begin{aligned}
L_H(\boldsymbol{h} \,|\, \boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) \\
&= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!] \\
&= \sum_{j=1}^{m} \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) [\![y_j \neq h_j(\boldsymbol{x})]\!] \qquad \boxed{\text{Swapping sums}} \\
&= \sum_{j=1}^{m} \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \left(y_j (1 - h_j(\boldsymbol{x})) + (1 - y_j) h_j(\boldsymbol{x}) \right)
\end{aligned}
$$

Hamming loss

- **Proof**:

$$
\begin{aligned}
L_H(\boldsymbol{h} \,|\, \boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) \\
&= \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!] \\
&= \sum_{j=1}^{m} \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) [\![y_j \neq h_j(\boldsymbol{x})]\!] \qquad \text{Swapping sums} \\
&= \sum_{j=1}^{m} \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \left(y_j (1 - h_j(\boldsymbol{x})) + (1 - y_j) h_j(\boldsymbol{x}) \right) \\
\text{Marginalization} \qquad &= \sum_{j=1}^{m} \mathbf{P}(y_j = 1 \,|\, \boldsymbol{x})(1 - h_j(\boldsymbol{x})) + \mathbf{P}(y_j = 0 \,|\, \boldsymbol{x}) h_j(\boldsymbol{x})
\end{aligned}
$$

Hamming loss

- **Proof**:

$$
\begin{aligned}
L_H(\boldsymbol{h}\,|\,\boldsymbol{x}) &= \sum_{\boldsymbol{y}\in\mathcal{Y}} \mathbf{P}(\boldsymbol{y}\,|\,\boldsymbol{x})\ell_H(\boldsymbol{y},\boldsymbol{h}(\boldsymbol{x})) \\
&= \sum_{\boldsymbol{y}\in\mathcal{Y}} \mathbf{P}(\boldsymbol{y}\,|\,\boldsymbol{x}) \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{x})]\!] \\
&= \sum_{j=1}^{m} \sum_{\boldsymbol{y}\in\mathcal{Y}} \mathbf{P}(\boldsymbol{y}\,|\,\boldsymbol{x}) [\![y_j \neq h_j(\boldsymbol{x})]\!] \qquad \text{Swapping sums} \\
&= \sum_{j=1}^{m} \sum_{\boldsymbol{y}\in\mathcal{Y}} \mathbf{P}(\boldsymbol{y}\,|\,\boldsymbol{x}) \left(y_j(1-h_j(\boldsymbol{x})) + (1-y_j)h_j(\boldsymbol{x}) \right) \\
\text{Marginalization} \qquad &= \sum_{j=1}^{m} \mathbf{P}(y_j = 1\,|\,\boldsymbol{x})(1-h_j(\boldsymbol{x})) + \mathbf{P}(y_j = 0\,|\,\boldsymbol{x})h_j(\boldsymbol{x}) \\
&= \sum_{j=1}^{m} L_{0/1}(y_j, h_j(\boldsymbol{x}))
\end{aligned}
$$

The result follows from the well-known fact about the risk minimization in binary classification.

Precision@k

- **Precision at position k**:

$$\operatorname{prec}@k(\boldsymbol{y}, \boldsymbol{h}, \boldsymbol{x}) = \frac{1}{k} \sum_{j \in \hat{\mathcal{Y}}_k} [\![y_j = 1]\!] \,,$$

where $\hat{\mathcal{Y}}_k$ is a set of k labels predicted by $\boldsymbol{h}$.

Precision@k

- **Precision at position k**:

$$\text{prec@}k(\boldsymbol{y}, \boldsymbol{h}, \boldsymbol{x}) = \frac{1}{k} \sum_{j \in \hat{\mathcal{Y}}_k} [\![y_j = 1]\!] \,,$$

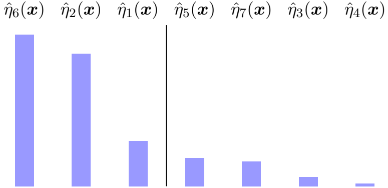
  where $\hat{\mathcal{Y}}_k$ is a set of k labels predicted by $\boldsymbol{h}$.
- **The optimal strategy**: select top k labels according to $\eta_j(\boldsymbol{x})$

Precision@k

- **Precision at position k**:

$$\text{prec@}k(\boldsymbol{y}, \boldsymbol{h}, \boldsymbol{x}) = \frac{1}{k} \sum_{j \in \hat{\mathcal{Y}}_k} [\![y_j = 1]\!] \,,$$

  where $\hat{\mathcal{Y}}_k$ is a set of k labels predicted by $\boldsymbol{h}$.
- **The optimal strategy**: select top k labels according to $\eta_j(\boldsymbol{x})$



- **Normalized Discounted Cumulative Gain at position k**:

$$\mathrm{NDCG@}k(\boldsymbol{y}, f, \boldsymbol{x}) = N_k(\boldsymbol{y}) \sum_{r=1}^{k} \frac{y_{\pi(r)}}{\log(1+r)},$$

where π is a permutation of labels for $\boldsymbol{x}$ returned by ranker f, and $N_k(\boldsymbol{y})$ normalizes NDCG@k to the interval $[0, 1]$:

$$N_k(\boldsymbol{y}) = \left(\sum_{r=1}^{\max(k, \sum_{i=1}^{m} y_i)} \frac{1}{\log(1+r)} \right)^{-1}$$

Normalized Discounted Cumulative Gain

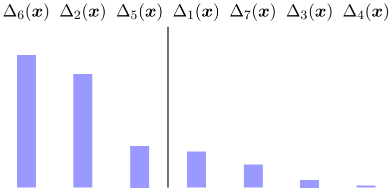
- **The optimal strategy**: rank labels according to the following marginal quantities:

$$\Delta_j(\boldsymbol{x}) = \sum_{\boldsymbol{y}: y_j = 1} N_k(\boldsymbol{y}) \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$$

Normalized Discounted Cumulative Gain

- **The optimal strategy**: rank labels according to the following marginal quantities:

$$\Delta_j(\boldsymbol{x}) = \sum_{\boldsymbol{y}: y_j = 1} N_k(\boldsymbol{y}) \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$$



Macro-averaging of the F-measure

- The **macro** F-measure (F-score):

$$F_M(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) = \frac{1}{m}\sum_{j=1}^{m} F(\boldsymbol{y}_{\cdot j}, \hat{\boldsymbol{y}}_{\cdot j}) = \frac{1}{m}\sum_{j=1}^{m} \frac{2\sum_{i=1}^{n} y_{ij}\hat{y}_{ij}}{\sum_{i=1}^{n} y_{ij} + \sum_{i=1}^{n} \hat{y}_{ij}}$$

True labels

y_{11}	y_{12}	y_{13}	y_{14}
y_{21}	y_{22}	y_{23}	y_{24}
y_{31}	y_{32}	y_{33}	y_{34}
y_{41}	y_{42}	y_{43}	y_{44}
y_{51}	y_{52}	y_{53}	y_{54}
y_{61}	y_{62}	y_{63}	y_{64}

Predicted labels

$\hat{y}_{11}$	$\hat{y}_{12}$	$\hat{y}_{13}$	$\hat{y}_{14}$
$\hat{y}_{21}$	$\hat{y}_{22}$	$\hat{y}_{23}$	$\hat{y}_{24}$
$\hat{y}_{31}$	$\hat{y}_{32}$	$\hat{y}_{33}$	$\hat{y}_{34}$
$\hat{y}_{41}$	$\hat{y}_{42}$	$\hat{y}_{43}$	$\hat{y}_{44}$
$\hat{y}_{51}$	$\hat{y}_{52}$	$\hat{y}_{53}$	$\hat{y}_{54}$
$\hat{y}_{61}$	$\hat{y}_{62}$	$\hat{y}_{63}$	$\hat{y}_{64}$

Macro-averaging of the F-measure

- The **macro** F-measure (F-score):

$$F_M(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) = \frac{1}{m}\sum_{j=1}^{m} F(\boldsymbol{y}_{\cdot j}, \hat{\boldsymbol{y}}_{\cdot j}) = \frac{1}{m}\sum_{j=1}^{m} \frac{2\sum_{i=1}^{n} y_{ij}\hat{y}_{ij}}{\sum_{i=1}^{n} y_{ij} + \sum_{i=1}^{n} \hat{y}_{ij}}$$

True labels

y_{11}	y_{12}	y_{13}	y_{14}
y_{21}	y_{22}	y_{23}	y_{24}
y_{31}	y_{32}	y_{33}	y_{34}
y_{41}	y_{42}	y_{43}	y_{44}
y_{51}	y_{52}	y_{53}	y_{54}
y_{61}	y_{62}	y_{63}	y_{64}

Predicted labels

$\hat{y}_{11}$	$\hat{y}_{12}$	$\hat{y}_{13}$	$\hat{y}_{14}$
$\hat{y}_{21}$	$\hat{y}_{22}$	$\hat{y}_{23}$	$\hat{y}_{24}$
$\hat{y}_{31}$	$\hat{y}_{32}$	$\hat{y}_{33}$	$\hat{y}_{34}$
$\hat{y}_{41}$	$\hat{y}_{42}$	$\hat{y}_{43}$	$\hat{y}_{44}$
$\hat{y}_{51}$	$\hat{y}_{52}$	$\hat{y}_{53}$	$\hat{y}_{54}$
$\hat{y}_{61}$	$\hat{y}_{62}$	$\hat{y}_{63}$	$\hat{y}_{64}$

Macro-averaging of the F-measure

- The **macro** F-measure (F-score):

$$F_M(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) = \frac{1}{m}\sum_{j=1}^{m} F(\boldsymbol{y}_{\cdot j}, \hat{\boldsymbol{y}}_{\cdot j}) = \frac{1}{m}\sum_{j=1}^{m} \frac{2\sum_{i=1}^{n} y_{ij}\hat{y}_{ij}}{\sum_{i=1}^{n} y_{ij} + \sum_{i=1}^{n} \hat{y}_{ij}}$$

True labels

y_{11}	y_{12}	y_{13}	y_{14}
y_{21}	y_{22}	y_{23}	y_{24}
y_{31}	y_{32}	y_{33}	y_{34}
y_{41}	y_{42}	y_{43}	y_{44}
y_{51}	y_{52}	y_{53}	y_{54}
y_{61}	y_{62}	y_{63}	y_{64}

Predicted labels

$\hat{y}_{11}$	$\hat{y}_{12}$	$\hat{y}_{13}$	$\hat{y}_{14}$
$\hat{y}_{21}$	$\hat{y}_{22}$	$\hat{y}_{23}$	$\hat{y}_{24}$
$\hat{y}_{31}$	$\hat{y}_{32}$	$\hat{y}_{33}$	$\hat{y}_{34}$
$\hat{y}_{41}$	$\hat{y}_{42}$	$\hat{y}_{43}$	$\hat{y}_{44}$
$\hat{y}_{51}$	$\hat{y}_{52}$	$\hat{y}_{53}$	$\hat{y}_{54}$
$\hat{y}_{61}$	$\hat{y}_{62}$	$\hat{y}_{63}$	$\hat{y}_{64}$

Macro-averaging of the F-measure

- The **macro** F-measure (F-score):

$$F_M(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) = \frac{1}{m}\sum_{j=1}^{m} F(\boldsymbol{y}_{\cdot j}, \hat{\boldsymbol{y}}_{\cdot j}) = \frac{1}{m}\sum_{j=1}^{m} \frac{2\sum_{i=1}^{n} y_{ij}\hat{y}_{ij}}{\sum_{i=1}^{n} y_{ij} + \sum_{i=1}^{n} \hat{y}_{ij}}$$

True labels

y_{11}	y_{12}	y_{13}	y_{14}
y_{21}	y_{22}	y_{23}	y_{24}
y_{31}	y_{32}	y_{33}	y_{34}
y_{41}	y_{42}	y_{43}	y_{44}
y_{51}	y_{52}	y_{53}	y_{54}
y_{61}	y_{62}	y_{63}	y_{64}

Predicted labels

$\hat{y}_{11}$	$\hat{y}_{12}$	$\hat{y}_{13}$	$\hat{y}_{14}$
$\hat{y}_{21}$	$\hat{y}_{22}$	$\hat{y}_{23}$	$\hat{y}_{24}$
$\hat{y}_{31}$	$\hat{y}_{32}$	$\hat{y}_{33}$	$\hat{y}_{34}$
$\hat{y}_{41}$	$\hat{y}_{42}$	$\hat{y}_{43}$	$\hat{y}_{44}$
$\hat{y}_{51}$	$\hat{y}_{52}$	$\hat{y}_{53}$	$\hat{y}_{54}$
$\hat{y}_{61}$	$\hat{y}_{62}$	$\hat{y}_{63}$	$\hat{y}_{64}$

Macro-averaging of the F-measure

- The **macro** F-measure (F-score):

$$F_M(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) = \frac{1}{m}\sum_{j=1}^{m} F(\boldsymbol{y}_{\cdot j}, \hat{\boldsymbol{y}}_{\cdot j}) = \frac{1}{m}\sum_{j=1}^{m} \frac{2\sum_{i=1}^{n} y_{ij}\hat{y}_{ij}}{\sum_{i=1}^{n} y_{ij} + \sum_{i=1}^{n} \hat{y}_{ij}}$$

True labels

y_{11}	y_{12}	y_{13}	y_{14}
y_{21}	y_{22}	y_{23}	y_{24}
y_{31}	y_{32}	y_{33}	y_{34}
y_{41}	y_{42}	y_{43}	y_{44}
y_{51}	y_{52}	y_{53}	y_{54}
y_{61}	y_{62}	y_{63}	y_{64}

Predicted labels

$\hat{y}_{11}$	$\hat{y}_{12}$	$\hat{y}_{13}$	$\hat{y}_{14}$
$\hat{y}_{21}$	$\hat{y}_{22}$	$\hat{y}_{23}$	$\hat{y}_{24}$
$\hat{y}_{31}$	$\hat{y}_{32}$	$\hat{y}_{33}$	$\hat{y}_{34}$
$\hat{y}_{41}$	$\hat{y}_{42}$	$\hat{y}_{43}$	$\hat{y}_{44}$
$\hat{y}_{51}$	$\hat{y}_{52}$	$\hat{y}_{53}$	$\hat{y}_{54}$
$\hat{y}_{61}$	$\hat{y}_{62}$	$\hat{y}_{63}$	$\hat{y}_{64}$

Macro-averaging of the F-measure

- Can be **solved** by **reduction** to m independent **binary** problems[3]

[3] O. Koyejo, N. Natarajan, P. Ravikumar, and I. Dhillon. Consistent multilabel classification. In *NIPS*, 2015

Macro-averaging of the F-measure

- Can be **solved** by **reduction** to m independent **binary** problems[3]
- **Thresholding** the conditional probabilities:

$$F(\tau) = \frac{2 \int_{\mathcal{X}} \eta(\boldsymbol{x}) [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}{\int_{\mathcal{X}} \eta(\boldsymbol{x}) \, \mathrm{d}\mu(\boldsymbol{x}) + \int_{\mathcal{X}} [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}$$

[3] O. Koyejo, N. Natarajan, P. Ravikumar, and I. Dhillon. Consistent multilabel classification. In *NIPS*, 2015

Macro-averaging of the F-measure

- Can be **solved** by **reduction** to m independent **binary** problems[3]
- **Thresholding** the conditional probabilities:

$$F(\tau) = \frac{2\int_{\mathcal{X}} \eta(\boldsymbol{x}) [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}{\int_{\mathcal{X}} \eta(\boldsymbol{x}) \, \mathrm{d}\mu(\boldsymbol{x}) + \int_{\mathcal{X}} [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}$$

- The **optimal F-measure** is $F(\tau^*)$: no binary classifier can be better

[3] O. Koyejo, N. Natarajan, P. Ravikumar, and I. Dhillon. Consistent multilabel classification. In *NIPS*, 2015

Macro-averaging of the F-measure

- Can be **solved** by **reduction** to m independent **binary** problems[3]
- **Thresholding** the conditional probabilities:

$$F(\tau) = \frac{2\int_{\mathcal{X}} \eta(\boldsymbol{x}) [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}{\int_{\mathcal{X}} \eta(\boldsymbol{x}) \, \mathrm{d}\mu(\boldsymbol{x}) + \int_{\mathcal{X}} [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}$$

- The **optimal F-measure** is $F(\tau^*)$: no binary classifier can be better
- The optimal solution **satisfies** the following **condition**: $F(\tau^*) = 2\tau^*$

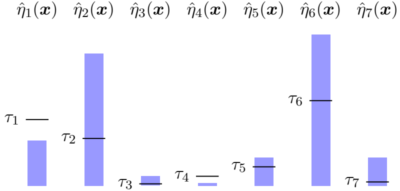
[3] O. Koyejo, N. Natarajan, P. Ravikumar, and I. Dhillon. Consistent multilabel classification. In *NIPS*, 2015

Macro-averaging of the F-measure

- Can be **solved** by **reduction** to m independent **binary** problems[3]
- **Thresholding** the conditional probabilities:

$$F(\tau) = \frac{2\int_{\mathcal{X}} \eta(\boldsymbol{x}) [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}{\int_{\mathcal{X}} \eta(\boldsymbol{x}) \, \mathrm{d}\mu(\boldsymbol{x}) + \int_{\mathcal{X}} [\![\eta(\boldsymbol{x}) \geq \tau]\!] \, \mathrm{d}\mu(\boldsymbol{x})}$$

- The **optimal F-measure** is $F(\tau^*)$: no binary classifier can be better
- The optimal solution **satisfies** the following **condition**: $F(\tau^*) = 2\tau^*$



[3] O. Koyejo, N. Natarajan, P. Ravikumar, and I. Dhillon. Consistent multilabel classification. In *NIPS*, 2015

Predictive models

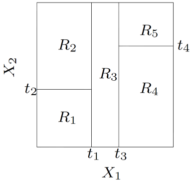
- **Lesson learned**: Train models that estimate marginal probabilities or other related marginal quantities

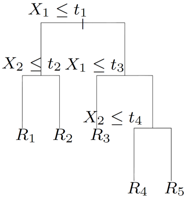
Outline

Decision trees

- Decision trees:

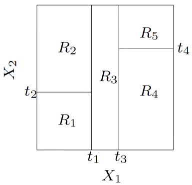
- Decision trees:
  - Partition of the feature space to small subregions:

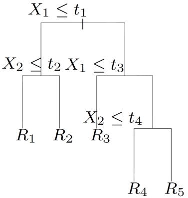




Decision trees

- Decision trees:
  - Partition of the feature space to small subregions:

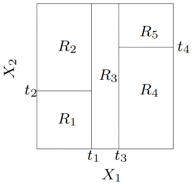


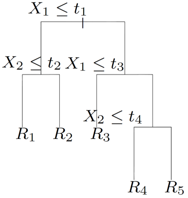


  - **Algorithm**: recursive splits of the regions

Decision trees

- Decision trees:
  - Partition of the feature space to small subregions:

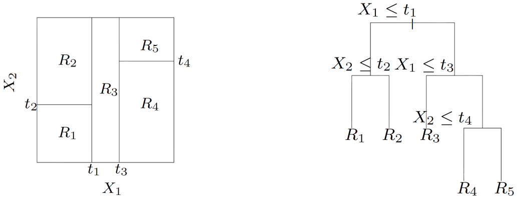




  - **Algorithm**: recursive splits of the regions
  - **Typical splits**: parallel to the axes (large number of possible splits)

Decision trees

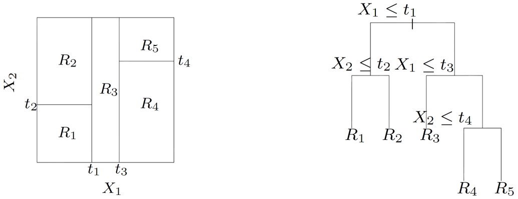
- Decision trees:
  - Partition of the feature space to small subregions:



  - **Algorithm**: recursive splits of the regions
  - **Typical splits**: parallel to the axes (large number of possible splits)
  - **Splitting criterion**: loss over partitions (can be costly)

Decision trees

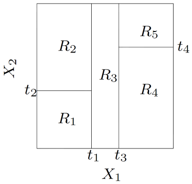
- Decision trees:
  - Partition of the feature space to small subregions:

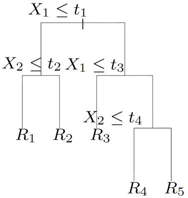


  - **Algorithm**: recursive splits of the regions
  - **Typical splits**: parallel to the axes (large number of possible splits)
  - **Splitting criterion**: loss over partitions (can be costly)
  - **Fast prediction**: logarithmic in n

Decision trees

- Decision trees:
  - Partition of the feature space to small subregions:





  - **Algorithm**: recursive splits of the regions
  - **Typical splits**: parallel to the axes (large number of possible splits)
  - **Splitting criterion**: loss over partitions (can be costly)
  - **Fast prediction**: logarithmic in n
  - Decision trees for extreme classification?

Decision trees

- Decision trees for extreme classification:

[4] Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. In *NIPS 29*, 2015

[5] Yashoteja Prabhu and Manik Varma. FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014

[6] Maryam Majzoubi and Anna Choromanska. LdSM: Logarithm-depth streaming multi-label decision trees, 2019

Decision trees

- Decision trees for extreme classification:
  - Limited number of potential splits

---

[4] Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. In *NIPS 29*, 2015

[5] Yashoteja Prabhu and Manik Varma. FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014

[6] Maryam Majzoubi and Anna Choromanska. LdSM: Logarithm-depth streaming multi-label decision trees, 2019

Decision trees

- Decision trees for extreme classification:
  - Limited number of potential splits ⇒ linear splits?

---

[4] Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. In *NIPS 29*, 2015

[5] Yashoteja Prabhu and Manik Varma. FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014

[6] Maryam Majzoubi and Anna Choromanska. LdSM: Logarithm-depth streaming multi-label decision trees, 2019

Decision trees

- Decision trees for extreme classification:
  - Limited number of potential splits $\Rightarrow$ linear splits?
  - Adjusting of linear splits

[4] Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. In *NIPS 29*, 2015

[5] Yashoteja Prabhu and Manik Varma. FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014

[6] Maryam Majzoubi and Anna Choromanska. LdSM: Logarithm-depth streaming multi-label decision trees, 2019

Decision trees

- Decision trees for extreme classification:
  - Limited number of potential splits ⇒ linear splits?
  - Adjusting of linear splits ⇒ Assignment of examples to partitions?

[4] Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. In *NIPS 29*, 2015

[5] Yashoteja Prabhu and Manik Varma. FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014

[6] Maryam Majzoubi and Anna Choromanska. LdSM: Logarithm-depth streaming multi-label decision trees, 2019

Decision trees

- Decision trees for extreme classification:
  - Limited number of potential splits $\Rightarrow$ linear splits?
  - Adjusting of linear splits $\Rightarrow$ Assignment of examples to partitions?
  - Efficient splitting criterion

[4] Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. In *NIPS 29*, 2015

[5] Yashoteja Prabhu and Manik Varma. FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014

[6] Maryam Majzoubi and Anna Choromanska. LdSM: Logarithm-depth streaming multi-label decision trees, 2019

Decision trees

- Decision trees for extreme classification:
  - Limited number of potential splits $\Rightarrow$ linear splits?
  - Adjusting of linear splits $\Rightarrow$ Assignment of examples to partitions?
  - Efficient splitting criterion
  - New algorithms: LomTree[4], **FastXML**[5], LdSM[6]

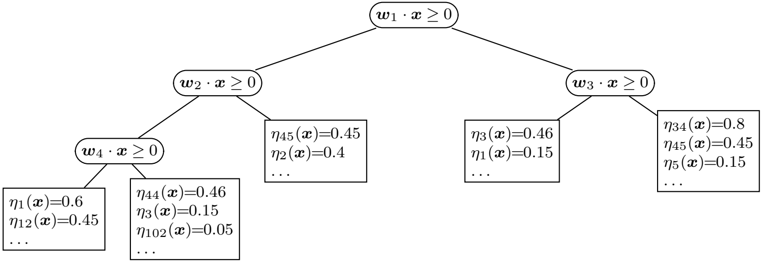
[4] Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. In *NIPS 29*, 2015

[5] Yashoteja Prabhu and Manik Varma. FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014

[6] Maryam Majzoubi and Anna Choromanska. LdSM: Logarithm-depth streaming multi-label decision trees, 2019

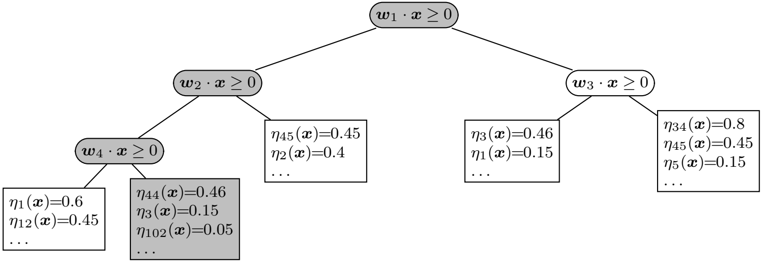
FastXML

- Uses an **ensemble** of decision trees
- **Sparse linear** classifiers trained in internal nodes
- Very **efficient** training procedure
- **Empirical distributions** in leaves
- A test example passes **one path** from the root to a leaf



FastXML

- Uses an **ensemble** of decision trees
- **Sparse linear** classifiers trained in internal nodes
- Very **efficient** training procedure
- **Empirical distributions** in leaves
- A test example passes **one path** from the root to a leaf



Optimization in FastXML

- In each internal node FastXML solves:

$$
\begin{aligned}
\min \quad & \|\boldsymbol{w}\|_1 + \sum_{i=1}^{n} C_{\delta_i} \log(1 + \exp(-\delta_i \boldsymbol{w}^\top \boldsymbol{x})) \\
& - C_r \sum_{i=1}^{n} \frac{1}{2}(1 + \delta_i)\mathrm{NDCG@}m(\boldsymbol{y}_i, \pi^+) \\
& - C_r \sum_{i=1}^{n} \frac{1}{2}(1 - \delta_i)\mathrm{NDCG@}m(\boldsymbol{y}_i, \pi^-) \\
\text{w.r.t.} \quad & \boldsymbol{w} \in \mathbb{R}^d, \boldsymbol{\delta} \in \{-1, 1\}^n, \pi^+, \pi^- \in \Pi(1, m)
\end{aligned}
$$

Optimization in FastXML

- In each internal node FastXML solves:

$$
\begin{aligned}
\min \quad & \|\boldsymbol{w}\|_1 + \sum_{i=1}^{n} C_{\delta_i} \log(1 + \exp(-\delta_i \boldsymbol{w}^\top \boldsymbol{x})) \\
& - C_r \sum_{i=1}^{n} \frac{1}{2}(1 + \delta_i) \mathrm{NDCG}@m(\boldsymbol{y}_i, \pi^+) \\
& - C_r \sum_{i=1}^{n} \frac{1}{2}(1 - \delta_i) \mathrm{NDCG}@m(\boldsymbol{y}_i, \pi^-) \\
\text{w.r.t.} \quad & \boldsymbol{w} \in \mathbb{R}^d, \boldsymbol{\delta} \in \{-1, 1\}^n, \pi^+, \pi^- \in \Pi(1, m)
\end{aligned}
$$

linear split

partitioning of training examples

label ranking in positive and negative partition

Optimization in FastXML

- In each internal node FastXML solves:

$$
\begin{aligned}
\min \quad & \|\boldsymbol{w}\|_1 + \sum_{i=1}^{n} C_{\delta_i} \log(1 + \exp(-\delta_i \boldsymbol{w}^\top \boldsymbol{x})) \\
& - C_r \sum_{i=1}^{n} \frac{1}{2}(1 + \delta_i)\mathrm{NDCG}@m(\boldsymbol{y}_i, \pi^+) \\
& - C_r \sum_{i=1}^{n} \frac{1}{2}(1 - \delta_i)\mathrm{NDCG}@m(\boldsymbol{y}_i, \pi^-) \\
\text{w.r.t.} \quad & \boldsymbol{w} \in \mathbb{R}^d, \boldsymbol{\delta} \in \{-1, 1\}^n, \pi^+, \pi^- \in \Pi(1, m)
\end{aligned}
$$

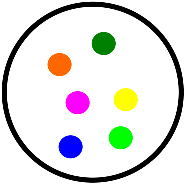
1. Bernoulli sampling of $\boldsymbol{\delta}$
2. Optimization of $\pi^{\pm}$
3. Optimization of $\boldsymbol{\delta}$
4. Optimization of $\boldsymbol{w}$
5. Repeat 2-4

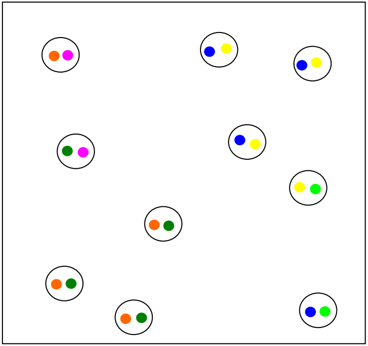
linear split

partitioning of training examples

label ranking in positive and negative partition

FastXML[7]

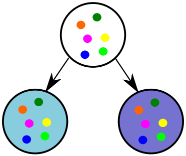


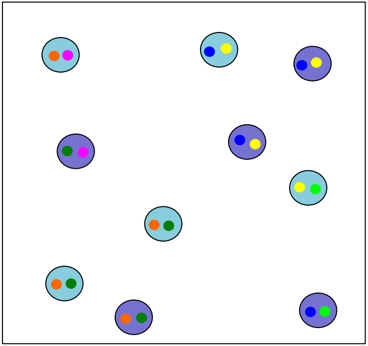


[7] https://www.youtube.com/watch?v=1X71fTx1LKA

FastXML[7]

Bernoulli sampling of δ
(with parameter $p = 0.5$)

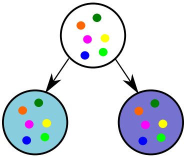


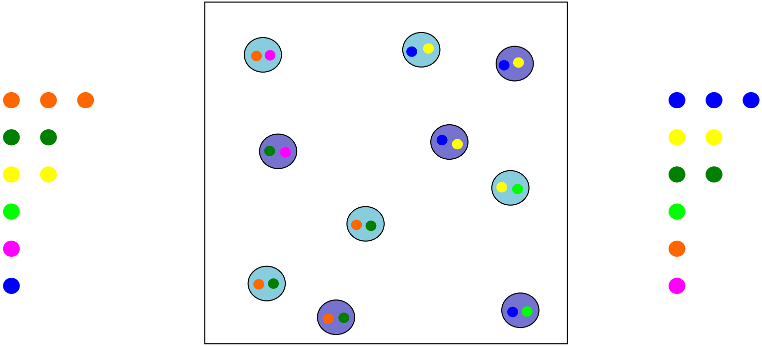


[7] https://www.youtube.com/watch?v=1X71fTx1LKA

FastXML[7]

Optimization of $\pi^{\pm}$
(rank labels according to $\hat{\Delta}_j^{\pm} = \sum_{i:\delta_i=\pm 1} N_m(\boldsymbol{y}_i) y_{i,j}$)

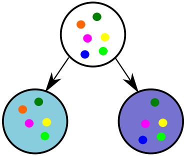


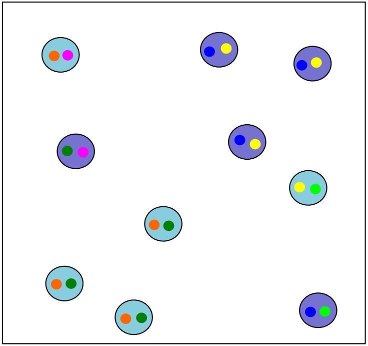


[7] https://www.youtube.com/watch?v=1X71fTx1LKA

FastXML[7]

Optimization of δ
($\delta_i^* = \mathrm{sign}(v_i^- - v_i^+)$ with $v_i^\pm = C_{\delta_\pm} \log(1{+}e^{\mp\delta_i \boldsymbol{w}^\top \boldsymbol{x}}) + C_r \mathrm{NDCG@}m(\boldsymbol{y}_i, \pi^\pm)$)

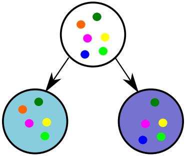


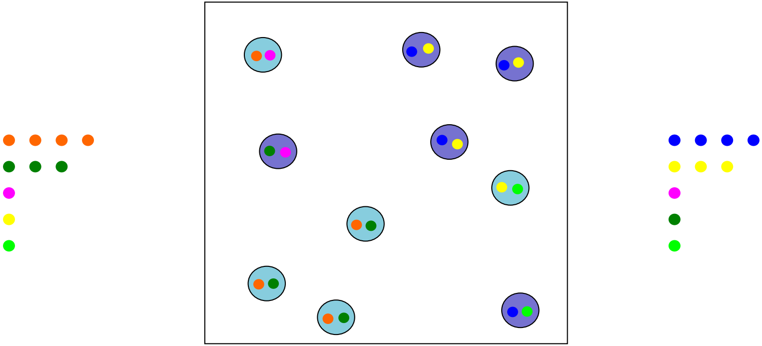


[7] https://www.youtube.com/watch?v=1X71fTx1LKA

FastXML[7]

Optimization of $\pi^{\pm}$
(rank labels according to $\hat{\Delta}_j^{\pm} = \sum_{i:\delta_i=\pm1} N_m(\boldsymbol{y}_i) y_{i,j}$)

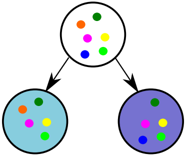


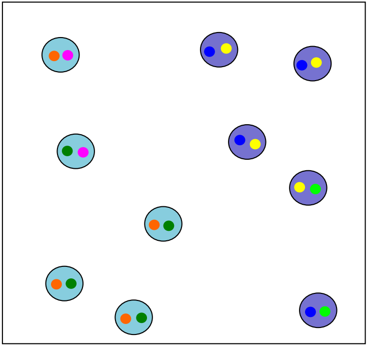


7 https://www.youtube.com/watch?v=1X71fTx1LKA

FastXML[7]

Optimization of δ
($\delta_i^* = \mathrm{sign}(v_i^- - v_i^+)$ with
$v_i^\pm = C_{\delta_\pm} \log(1+e^{\mp\delta_i \boldsymbol{w}^\top \boldsymbol{x}}) + C_r \mathrm{NDCG@}m(\boldsymbol{y}_i, \pi^\pm))$



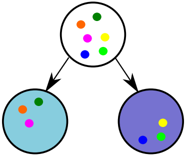


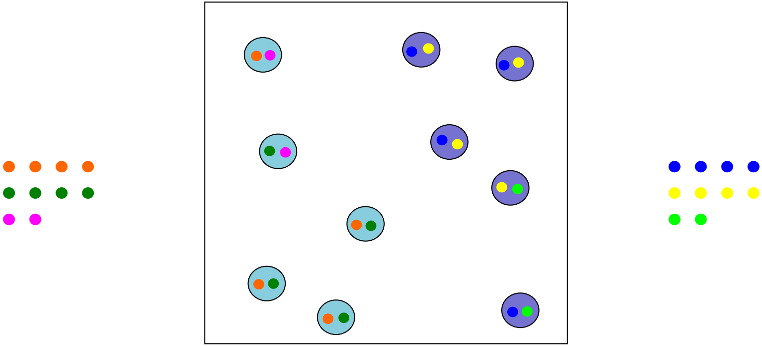
[7] https://www.youtube.com/watch?v=1X71fTx1LKA

FastXML[7]

Optimization of $\pi^{\pm}$

(rank labels according to $\hat{\Delta}_j^{\pm} = \sum_{i:\delta_i=\pm 1} N_m(\boldsymbol{y}_i) y_{i,j}$)

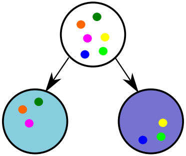


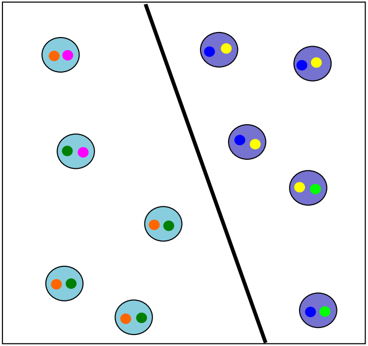


7 `https://www.youtube.com/watch?v=1X71fTx1LKA`

FastXML[7]

Optimization of $\boldsymbol{w}$
$(\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \|\boldsymbol{w}\|_1 + \sum_{i=1}^{n} C_{\delta_i} \log(1 + e^{-\delta_i \boldsymbol{w}^\top \boldsymbol{x}}))$

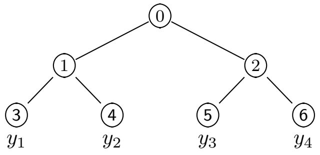




[7] https://www.youtube.com/watch?v=1X71fTx1LKA

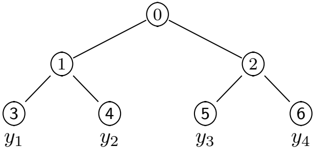
Label trees

- Label trees:
  - Organize classifiers in a tree structure (one leaf $\Leftrightarrow$ one label):



Label trees

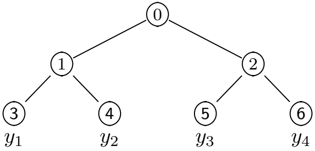
- Label trees:
  - Organize classifiers in a tree structure (one leaf $\Leftrightarrow$ one label):



  - **Tree structure**: partitioning of labels (predefined or trained)

Label trees

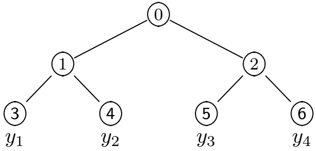
- Label trees:
  - Organize classifiers in a tree structure (one leaf $\Leftrightarrow$ one label):



  - **Tree structure**: partitioning of labels (predefined or trained)
  - **Node classifiers**: output variables defined by the label partitioning

Label trees

- Label trees:
  - Organize classifiers in a tree structure (one leaf $\Leftrightarrow$ one label):



  - **Tree structure**: partitioning of labels (predefined or trained)
  - **Node classifiers**: output variables defined by the label partitioning
  - **Fast prediction**: almost logarithmic in m

Label trees with probabilistic classifiers

Nested dichotomies,[8] Conditional probability trees,[9] Hierarchical softmax,[10] fastText,[11] Probabilistic classifier chains[12]

$\Downarrow$

Probability classifier trees[13]

$\Downarrow$

Hierarchical softmax

---

[8] J. Fox. *Applied regression analysis, linear models, and related methods*. Sage, 1997
E. Frank and S. Kramer. Ensembles of nested dichotomies for multi-class problems. In *ICML*, 2004

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

[10] Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *AISTATS*, pages 246–252, 2005

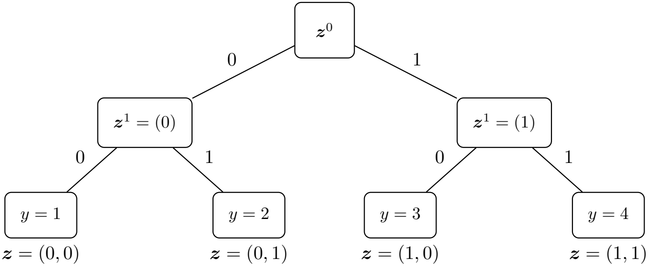
[11] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016

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

[13] Krzysztof Dembczyński, Wojciech Kotłowski, Willem Waegeman, Róbert Busa-Fekete, and Eyke Hüllermeier. Consistency of probabilistic classifier trees. In *ECMLPKDD*. Springer, 2016

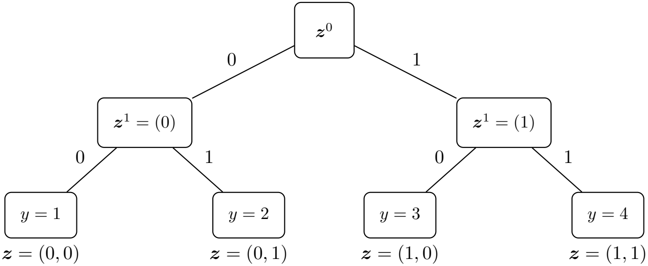
Hierarchical softmax (HSM)

- Encode the labels by a **prefix code** ($\Rightarrow$ tree structure)



Hierarchical softmax (HSM)

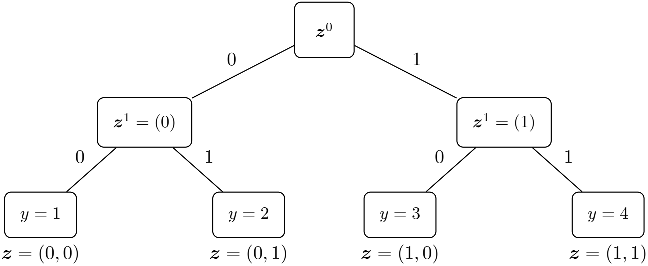
- Encode the labels by a **prefix code** ($\Rightarrow$ tree structure)



- Each label y **coded** by $\boldsymbol{z} = (z_1, \ldots, z_l) \in \mathcal{C}$

Hierarchical softmax (HSM)

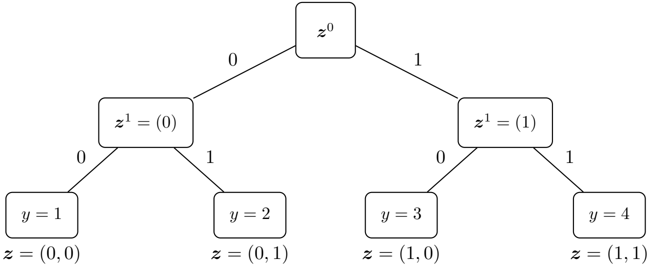
- Encode the labels by a **prefix code** ($\Rightarrow$ tree structure)



- Each label y **coded** by $\boldsymbol{z} = (z_1, \ldots, z_l) \in \mathcal{C}$
- An internal node identified by a **partial** code $\boldsymbol{z}^j = (z_1, \ldots, z_j)$

Hierarchical softmax (HSM)

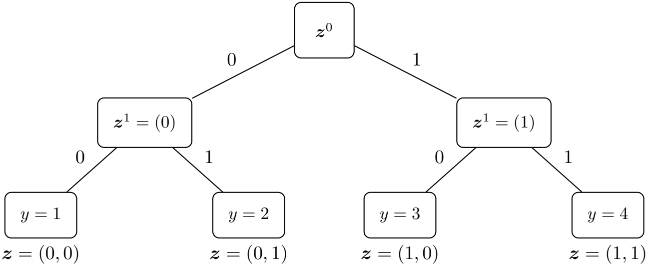
- Encode the labels by a **prefix code** ($\Rightarrow$ tree structure)



- Each label y **coded** by $\boldsymbol{z} = (z_1, \ldots, z_l) \in \mathcal{C}$
- An internal node identified by a **partial** code $\boldsymbol{z}^j = (z_1, \ldots, z_j)$
- The code does **not** have to be binary

Hierarchical softmax (HSM)

- Encode the labels by a **prefix code** ($\Rightarrow$ tree structure)

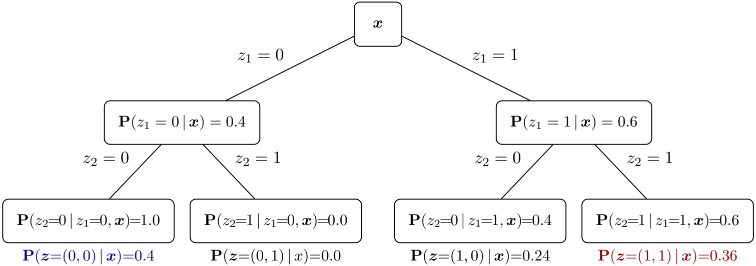


- Each label y **coded** by $\boldsymbol{z} = (z_1, \ldots, z_l) \in \mathcal{C}$
- An internal node identified by a **partial** code $\boldsymbol{z}^j = (z_1, \ldots, z_j)$
- The code does **not** have to be binary
- Different **structures** possible: random tree, Huffman tree, trained structure

Hierarchical softmax (HSM)

- HSM estimates $\mathbf{P}(y \,|\, \boldsymbol{x})$ by following a **path** from the root to a leaf:

$$\mathbf{P}(y \,|\, \boldsymbol{x}) = \mathbf{P}(\boldsymbol{z} \,|\, \boldsymbol{x}) = \prod_{j=1}^{l} \mathbf{P}(z_j | \boldsymbol{z}^{j-1}, \boldsymbol{x})$$

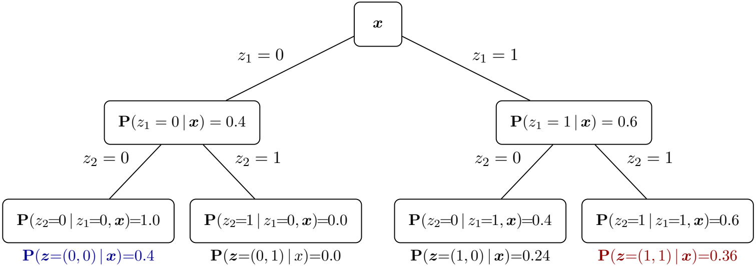


- Training: **separate** learning problems in the **internal** nodes
- Prediction: **depth first search**/**beam search**

Hierarchical softmax (HSM)

- HSM estimates $\mathbf{P}(y \,|\, \boldsymbol{x})$ by following a **path** from the root to a leaf:

$$\mathbf{P}(z_1 \,|\, \boldsymbol{x})\mathbf{P}(z_2 \,|\, z_1, \boldsymbol{x}) = \frac{\mathbf{P}(z_1, \boldsymbol{x})}{\mathbf{P}(\boldsymbol{x})}\frac{\mathbf{P}(z_2, z_1, \boldsymbol{x})}{\mathbf{P}(z_1, \boldsymbol{x})} = \mathbf{P}(z_1, z_2 \,|\, \boldsymbol{x})$$



- Training: **separate** learning problems in the **internal** nodes
- Prediction: **depth first search**/**beam search**

HSM for XMLC

- **Pick-one-label** heuristic used, for example, in fastText:

$$\eta'_j(\boldsymbol{x}) = \mathbf{P}'(y_j = 1 \,|\, \boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{Y}} y_j \frac{\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})}{\sum_{j'=1}^{m} y_{j'}}$$

HSM for XMLC

- **Pick-one-label** heuristic used, for example, in fastText:

$$\eta'_j(\boldsymbol{x}) = \mathbf{P}'(y_j = 1 \,|\, \boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{Y}} y_j \frac{\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})}{\sum_{j'=1}^{m} y_{j'}}$$

- **Theorem**: inconsistent for label-wise logistic loss and precision@k

labels $\boldsymbol{y}$	probability $\mathbf{P}(\boldsymbol{y} \,\vert\, \boldsymbol{x})$
$\{1\}$	0.15
$\{2\}$	0.10
$\{1,2\}$	0.25
$\{3\}$	0.30
$\{4\}$	0.20

True marg. probs	P-o-l marg. probs
$\eta_1(\boldsymbol{x}) = 0.4$	$\eta'_3(\boldsymbol{x}) = 0.3$
$\eta_2(\boldsymbol{x}) = 0.35$	$\eta'_1(\boldsymbol{x}) = 0.275$
$\eta_3(\boldsymbol{x}) = 0.3$	$\eta'_2(\boldsymbol{x}) = 0.225$
$\eta_4(\boldsymbol{x}) = 0.2$	$\eta'_4(\boldsymbol{x}) = 0.2$

HSM for XMLC

- **Pick-one-label** heuristic used, for example, in fastText:

$$\eta'_j(\boldsymbol{x}) = \mathbf{P}'(y_j = 1 \,|\, \boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{Y}} y_j \frac{\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})}{\sum_{j'=1}^{m} y_{j'}}$$

- **Theorem**: inconsistent for label-wise logistic loss and precision$@k$

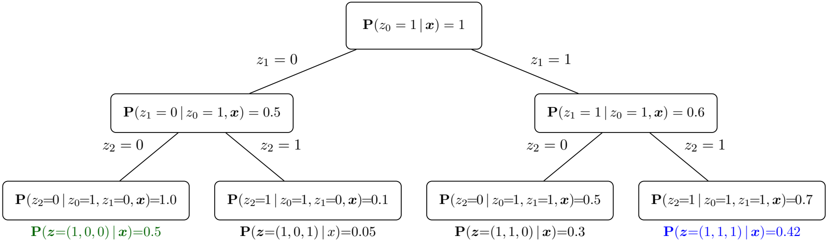
labels $\boldsymbol{y}$	probability $\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$
$\{1\}$	0.15
$\{2\}$	0.10
$\{1,2\}$	0.25
$\{3\}$	0.30
$\{4\}$	0.20

True marg. probs	P-o-l marg. probs
$\eta_1(\boldsymbol{x}) = 0.4$	$\eta'_3(\boldsymbol{x}) = 0.3$
$\eta_2(\boldsymbol{x}) = 0.35$	$\eta'_1(\boldsymbol{x}) = 0.275$
$\eta_3(\boldsymbol{x}) = 0.3$	$\eta'_2(\boldsymbol{x}) = 0.225$
$\eta_4(\boldsymbol{x}) = 0.2$	$\eta'_4(\boldsymbol{x}) = 0.2$

- **Theorem**: consistent for precision$@k$ for independent labels

Probabilistic label trees (PLTs)[14]

- **Similar** tree structure and encoding of $y_j = 1$ by $\boldsymbol{z} = (1, z_1, \ldots, z_l)$

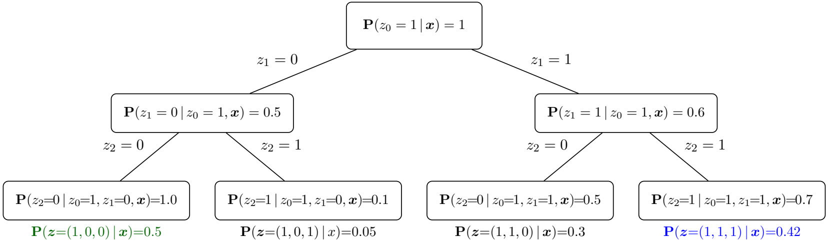


[14] K. Jasinska, K. Dembczyński, R. Busa-Fekete, K. Pfannschmidt, T. Klerx, and E. Hüllermeier. Extreme F-measure maximization using sparse probability estimates. In *ICML*, pages 1435–1444, 2016

Marek Wydmuch, Kalina Jasinska, Mikhail Kuznetsov, Róbert Busa-Fekete, and Krzysztof Dembczyński. A no-regret generalization of hierarchical softmax to extreme multi-label classification. In *NeurIPS*, pages 6355–6366. 2018

Probabilistic label trees (PLTs)[14]

- **Similar** tree structure and encoding of $y_j = 1$ by $\boldsymbol{z} = (1, z_1, \ldots, z_l)$



- **Marginal probabilities** $\eta_j(\boldsymbol{x})$ obtained by:

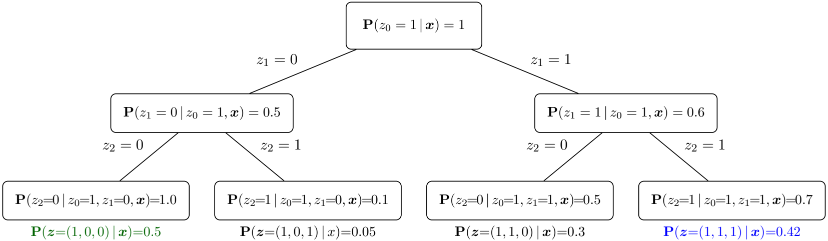
$$\eta_j(\boldsymbol{x}) = \mathbf{P}(\boldsymbol{z} \,|\, \boldsymbol{x}) = \prod_{i=0}^{l} \mathbf{P}(z_j | \boldsymbol{z}^{j-1}, \boldsymbol{x})$$

[14] K. Jasinska, K. Dembczyński, R. Busa-Fekete, K. Pfannschmidt, T. Klerx, and E. Hüllermeier. Extreme F-measure maximization using sparse probability estimates. In *ICML*, pages 1435–1444, 2016

Marek Wydmuch, Kalina Jasinska, Mikhail Kuznetsov, Róbert Busa-Fekete, and Krzysztof Dembczyński. A no-regret generalization of hierarchical softmax to extreme multi-label classification. In *NeurIPS*, pages 6355–6366. 2018

Probabilistic label trees (PLTs)[14]

- **Similar** tree structure and encoding of $y_j = 1$ by $\boldsymbol{z} = (1, z_1, \ldots, z_l)$



- **Marginal probabilities** $\eta_j(\boldsymbol{x})$ obtained by:

$$\eta_j(\boldsymbol{x}) = \mathbf{P}(\boldsymbol{z} \,|\, \boldsymbol{x}) = \prod_{i=0}^{l} \mathbf{P}(z_j | \boldsymbol{z}^{j-1}, \boldsymbol{x})$$

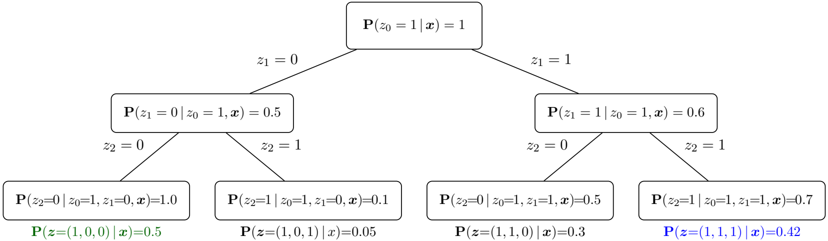
- $\boldsymbol{z}^j \Leftrightarrow$ **at least one** positive label in the corresponding subtree

[14] K. Jasinska, K. Dembczyński, R. Busa-Fekete, K. Pfannschmidt, T. Klerx, and E. Hüllermeier. Extreme F-measure maximization using sparse probability estimates. In *ICML*, pages 1435–1444, 2016

Marek Wydmuch, Kalina Jasinska, Mikhail Kuznetsov, Róbert Busa-Fekete, and Krzysztof Dembczyński. A no-regret generalization of hierarchical softmax to extreme multi-label classification. In *NeurIPS*, pages 6355–6366. 2018

Probabilistic label trees (PLTs)[14]

- **Similar** tree structure and encoding of $y_j = 1$ by $\boldsymbol{z} = (1, z_1, \ldots, z_l)$



- **Marginal probabilities** $\eta_j(\boldsymbol{x})$ obtained by:

$$\eta_j(\boldsymbol{x}) = \mathbf{P}(\boldsymbol{z} \,|\, \boldsymbol{x}) = \prod_{i=0}^{l} \mathbf{P}(z_j | \boldsymbol{z}^{j-1}, \boldsymbol{x})$$

- $\boldsymbol{z}^j \Leftrightarrow$ **at least one** positive label in the corresponding subtree
- $\sum_{z_j} \mathbf{P}(z_j \,|\, \boldsymbol{z}^{j-1}) \geq 1 \Rightarrow$ separate classifiers in **all** nodes of the tree

[14] K. Jasinska, K. Dembczyński, R. Busa-Fekete, K. Pfannschmidt, T. Klerx, and E. Hüllermeier. Extreme F-measure maximization using sparse probability estimates. In *ICML*, pages 1435–1444, 2016

Marek Wydmuch, Kalina Jasinska, Mikhail Kuznetsov, Róbert Busa-Fekete, and Krzysztof Dembczyński. A no-regret generalization of hierarchical softmax to extreme multi-label classification. In *NeurIPS*, pages 6355–6366. 2018

Probabilistic label trees (PLTs) for multi-class distribution

- For multi-class distribution it always holds that:

$$\mathbf{P}(z_0 = 1 \,|\, \boldsymbol{x}) = 1 \quad \text{and} \quad \sum_{z_j} P(z_j | \boldsymbol{z}^{j-1}, \boldsymbol{x}) = 1$$

- All classifiers can be moved one level up $\Rightarrow$ no classifiers in leaves
- **PLTs boil down to HSM**

Probabilistic label trees (PLTs)

- **Training**:
  - independent training of all node classifiers
  - reduced complexity by the conditions used in the nodes
  - batch or online learning of node classifiers
  - sparse representation: small number of active features in lower nodes, feature hashing
  - dense representation: hidden representation of features (strong compression)

Probabilistic label trees (PLTs)

- **Training**:
  - independent training of all node classifiers
  - reduced complexity by the conditions used in the nodes
  - batch or online learning of node classifiers
  - sparse representation: small number of active features in lower nodes, feature hashing
  - dense representation: hidden representation of features (strong compression)
- **Tree structure**:
  - Random, complete-sorted tree, Huffman tree, trained (e.g., hierarchical clustering of labels), online training

Probabilistic label trees (PLTs)

- **Training**:
  - independent training of all node classifiers
  - reduced complexity by the conditions used in the nodes
  - batch or online learning of node classifiers
  - sparse representation: small number of active features in lower nodes, feature hashing
  - dense representation: hidden representation of features (strong compression)
- **Tree structure**:
  - Random, complete-sorted tree, Huffman tree, trained (e.g., hierarchical clustering of labels), online training
- **Prediction**:
  - depth first search/beam search

Theoretical guarantees

- **Theorem**: For any distribution $\mathbf{P}$ and internal node classifiers $f_{\boldsymbol{z}^i}$, the following holds:

$$|\eta_j(\boldsymbol{x}) - \hat{\eta}_j(\boldsymbol{x})| \leq \sum_{i=0}^{l} \mathbf{P}(\boldsymbol{z}^{i-1} \,|\, \boldsymbol{x}) \sqrt{\frac{2}{\lambda}} \sqrt{\mathrm{reg}_\ell(f_{\boldsymbol{z}^i} \,|\, \boldsymbol{z}^{i-1}, \boldsymbol{x})} \,,$$

  where $\mathrm{reg}_\ell(f_{\boldsymbol{z}^i} \,|\, \boldsymbol{z}^{i-1}, \boldsymbol{x})$ is a binary classification regret for a strongly proper composite loss ℓ and λ is a constant specific for loss ℓ.

Theoretical guarantees

- **Theorem**: For any distribution $\mathbf{P}$ and classifier $\boldsymbol{h}$ delivering estimates $\hat{\eta}_j(\boldsymbol{x})$ of the marginal probabilities of labels, the following holds:

$$\mathrm{reg}_{p@k}(\boldsymbol{h} \,|\, \boldsymbol{x}) = \frac{1}{k} \sum_{i \in \mathcal{Y}_k} \eta_i(\boldsymbol{x}) - \frac{1}{k} \sum_{j \in \hat{\mathcal{Y}}_k} \eta_j(\boldsymbol{x}) \leq 2 \max_l |\eta_l(\boldsymbol{x}) - \hat{\eta}_l(\boldsymbol{x})|$$

Theoretical guarantees

- PLTs are **no-regret generalization** of HSM to multi-label problems.

Empirical studies[15]

Dataset	Metrics	FastXML	PPDSparse	DiSMEC	FT	LT	**XT**	Parabel	XML-CNN
Wiki-30K	P@1	82.03	73.80	85.20	80.78	80.85	**85.23**	83.77	82.78
$N_{train} = 14146$	P@3	67.47	60.90	**74.60**	50.46	50.59	73.18	71.96	66.34
$N_{test} = 6616$	P@5	57.76	50.40	**65.90**	36.79	37.68	63.39	62.44	56.23
$d = 101938$	T_{train}	16m	†	†	10m	12m	18m	**5m**	88m⋆
$m = 30938$	T_{test}/N_{test}	3.00ms	†	†	1.88ms	10.09ms	**0.83ms**	1.63ms∗	1.39ms⋆
	model size	354M	†	†	513M	513M	**259M**	**109M**∗	⋆
Delicious-200K	P@1	42.81	45.05	44.71	42.22	42.71	**47.85**	43.32	‡
$N_{train} = 196606$	P@3	38.76	38.34	38.08	37.90	36.27	**42.08**	38.49	‡
$N_{test} = 100095$	P@5	36.34	34.90	34.7	35.05	33.43	**39.13**	35.83	‡
$d = 782585$	T_{train}	458m	4781m	1080h	271m	563m	502m	105m	‡
$m = 205443$	T_{test}/N_{test}	4.86ms	275ms	300ms	1.97ms	1.98ms	**1.41ms**	**1.31ms**∗	‡
	model size	15.4G	9.4G	18.0G	9.0G	9.0G	**1.9G**	**1.8G**∗	‡
WikiLSHTC	P@1	49.35	64.08	**64.94**	41.13	50.15	58.73	61.53	‡
$N_{train} = 1778351$	P@3	32.69	41.26	**42.71**	24.09	31.95	39.24	40.07	‡
$N_{test} = 587084$	P@5	24.03	30.12	**31.5**	17.44	23.59	29.26	29.25	‡
$d = 617899$	T_{train}	724m	236m	750h	207	212m	550m	**34m**	‡
$m = 325056$	T_{test}/N_{test}	2.17ms	37.76ms	2580ms	1.25ms	4.76ms	**0.81ms**	0.92ms∗	‡
	model size	9.3G	5.2G	3.8G	6.5G	6.5G	**3.3G**	**1.1G**∗	‡
Wiki-500K	P@1	54.10	70.16	**70.20**	32.73	37.18	64.48	66.12	59.85
$N_{train} = 1813391$	P@3	29.45	50.57	**50.60**	19.02	21.62	45.84	47.02	39.28
$N_{test} = 783743$	P@5	21.21	39.66	**39.70**	14.46	16.01	35.46	36.45	29.81
$d = 2381304$	T_{train}	3214m	1771m	7495h	496m	531m	1253m	**168m**	7032m⋆
$m = 501070$	T_{test}/N_{test}	8.03ms	113.70ms	9300ms	2.05ms	6.43ms	**1.07ms**	4.68ms∗	21.06ms⋆
	model size	63G	3.4G	14.7G	11G	11G	**5.5G**	**2.0G**∗	**3.7G**⋆
Amazon-670K	P@1	34.24	45.32	**45.37**	25.47	27.67	39.90	41.59	35.39
$N_{train} = 490449$	P@3	29.30	40.37	**40.40**	21.47	20.96	35.36	37.18	33.74
$N_{test} = 153025$	P@5	26.12	36.92	**36.96**	18.61	17.72	32.04	33.85	32.64
$d = 135909$	T_{train}	422m	102m	373h	162m	182m	241m	**8m**	3134m⋆
$m = 670091$	T_{test}/N_{test}	3.39ms	66.09ms	1380ms	7.84ms	5.13ms	**1.72ms**	**0.68ms**∗	16.18ms⋆
	model size	10G	6.0G	3.8G	3.2G	3.2G	**1.5G**	**0.7G**∗	**1.5G**⋆

[15] XMLC benchmarks from `http://manikvarma.org/downloads/XC/XMLRepository.html`

Empirical studies

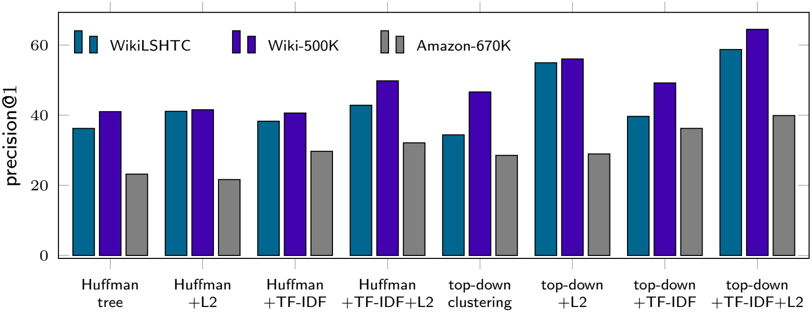
- Selected results for precision@1

Method	WikiLSHTC	Amazon670K	Delicious200K
HSM-vw	36.90	33.64	41.58
PLT-vw	**41.63**	**36.85**	**45.27**
FastText	41.13	25.47	42.22
ExtremeText	58.73	39.90	**47.85**
Parabel[16]	**61.53**	**41.59**	43.32
FastXML	49.75	34.24	42.81

[16] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *WWW*. ACM, 2018

Empirical studies

- The ablation analysis of different variants of XT.



Outline

1. Extreme multi-label classification: applications and challenges
2. Theoretical framework
3. Tree-based algorithms: decision and label trees
4. Take-away message

Take-away message

- **Extreme classification**: #examples, #features, **#labels**

Take-away message

- **Extreme classification**: #examples, #features, **#labels**
- **Complexity**: time vs. space, training vs. validation vs. **prediction**

Take-away message

- **Extreme classification**: #examples, #features, **#labels**
- **Complexity**: time vs. space, training vs. validation vs. **prediction**
- **Computational and statistical challenges**

Take-away message

- **Extreme classification**: #examples, #features, **#labels**
- **Complexity**: time vs. space, training vs. validation vs. **prediction**
- **Computational and statistical challenges**
- **Tree-based algorithms**:

Take-away message

- **Extreme classification**: #examples, #features, **#labels**
- **Complexity**: time vs. space, training vs. validation vs. **prediction**
- **Computational and statistical challenges**
- **Tree-based algorithms**:
  - Decision trees

Take-away message

- **Extreme classification**: #examples, #features, **#labels**
- **Complexity**: time vs. space, training vs. validation vs. **prediction**
- **Computational and statistical challenges**
- **Tree-based algorithms**:
  - Decision trees
  - **Label trees $\Rightarrow$ Probabilistic label trees**

Take-away message

- **Extreme classification**: #examples, #features, **#labels**
- **Complexity**: time vs. space, training vs. validation vs. **prediction**
- **Computational and statistical challenges**
- **Tree-based algorithms**:
  - Decision trees
  - **Label trees** $\Rightarrow$ **Probabilistic label trees**

https://www.cs.put.poznan.pl/kdembczynski