Multi-Target Prediction: A Unifying View on Problems and Methods

Willem Waegeman, Eyke Hüllermeier and Krzysztof Dembczyński

Department of Data Analysis and Mathematical Modelling, Ghent University, Belgium
Department of Computer Science, University of Paderborn, Germany
Institute of Computing Science, Poznan University of Technology, Poland

September 10 ECML/PKDD 2018

UEFA Nations League: Rode Duivels ontmoeten Zwitserland en IJsland

De wedstrijden worden gespeeld in het najaar.

WO 24/01/2018 - 12:47

In Zwitserland is vandaag geloot voor de UEFA Nations League, een gloednieuw toernooi met alle Europese voetballanden. De Rode Duivels spelen tegen Zwitserland en IJsland. De vier groepswinnaars uit divisie A, de divisie van de Belgen, strijden in 2019 in een Final Four om toernooiwinst.

Door hun goede prestaties van de voorbije jaren zitten de Rode Duivels in divisie A van de gloednieuwe

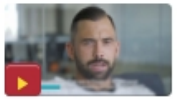
RODE DUIVELS

Martinez: "Moeten alle Rode Duivels op zelfde tactische pagina krijgen"

Martinez: "Zou vergissing zijn om Radja te selecteren"

Vertonghen: "We weten dat er een kans is om te winnen en daar gaan we vol voor"

Batshuayi: "Ik wil topschutter van het WK worden"



Defour stopt bij de Duivels: "Wil carrière zo lang mogelijk uitoefenen"

< >

Rode Duivels

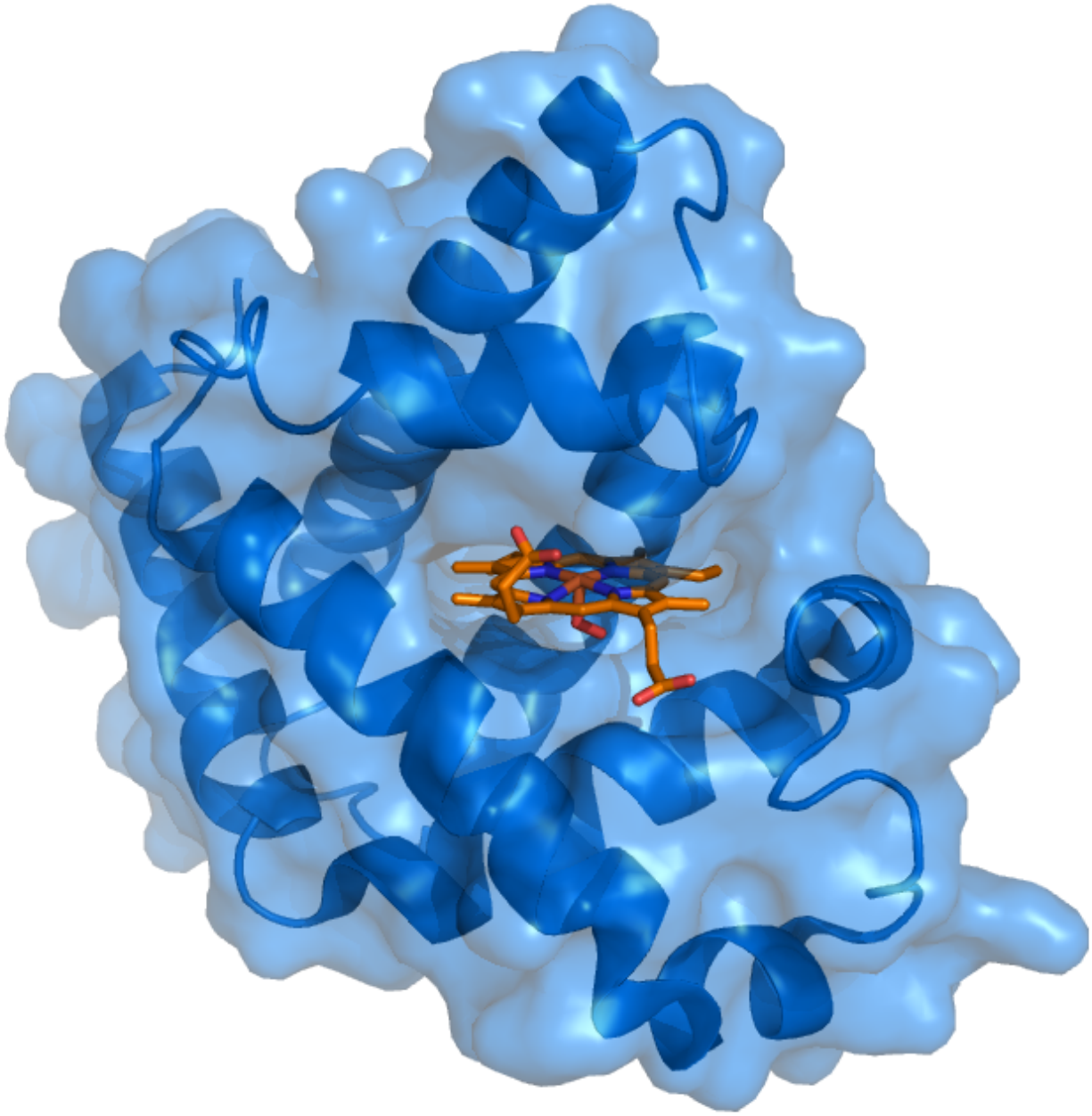
Een Twitter-lijst door @sporza

Rode Duivels

Kevin De Bruyne @DeBruyneKev

Multi-label classification: the example of document categorization

		Tennis	Football	Biking	Movies	TV	Belgium
01101	Text1	0	1	0	0	1	1
00111	Text2	1	0	0	0	0	1
01110	Text3	0	0	0	1	1	0
10001	Text4	0	0	1	0	1	0
01011	Text5	1	0	0	1	0	0
11110	Text6	?	?	?	?	?	?

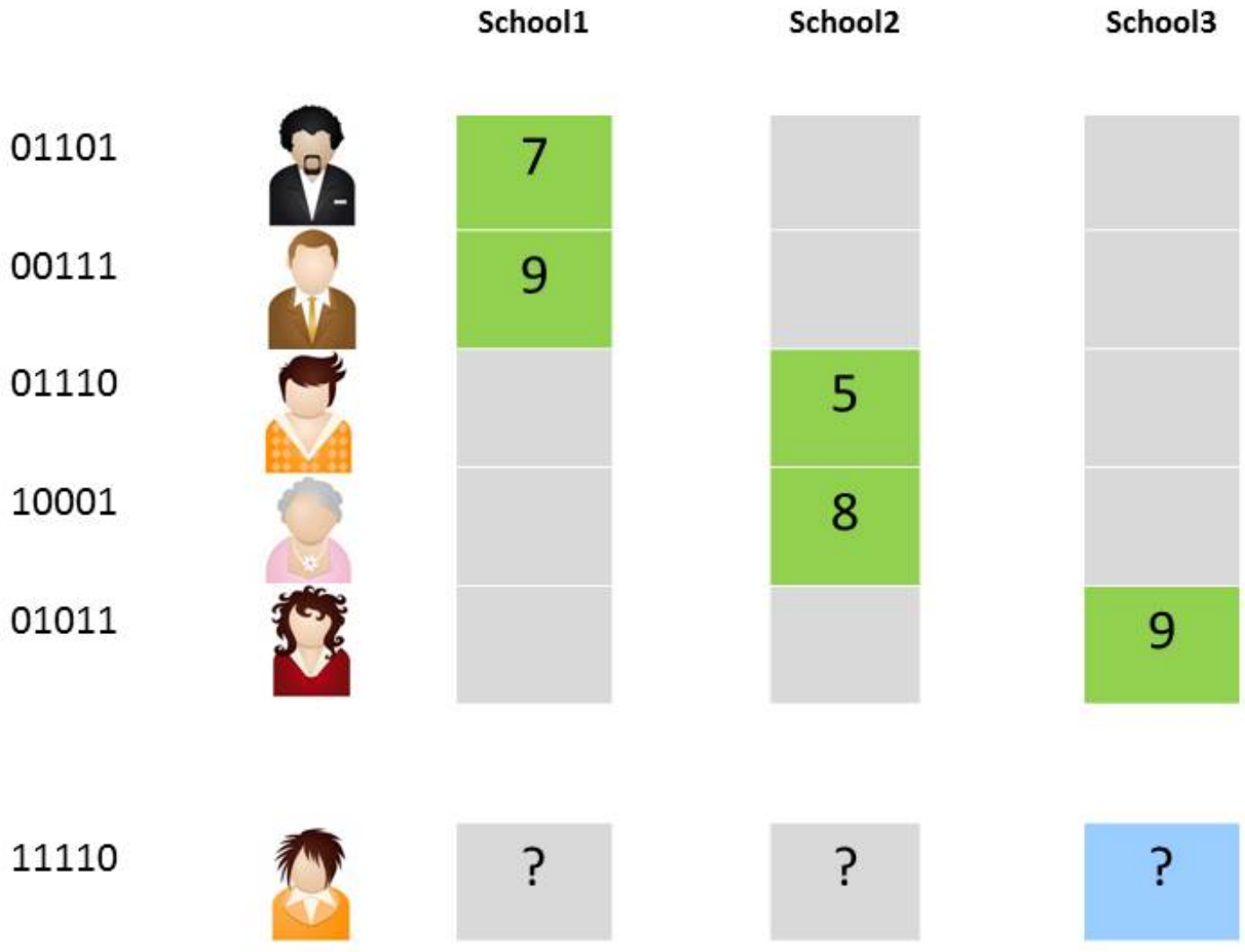


Multivariate regression:
the example of protein-ligand interaction prediction

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101 	1,3	0,2	1,4	1,7	3,5	1,3
00111 	2	1,7	1,5	7,5	8,2	7,6
01110 	0,2	0	0,3	0,4	1,2	2,2
10001 	3,1	1,1	1,3	1,1	1,7	5,2
01011 	4,7	2,1	2,5	1,5	2,3	8,5
11110 	?	?	?	?	?	?


SUNNYBANK
MIAMI
SEMPER FLOREAT COSTA
KEDRON
TRINITY BAY
ENDEAVOUR
YERONGA
SCIENTIA AC HUMANITATE
CORINDA
ROCHEDALE
SHS
RECTE CRESCAM
The Cathedral School
TOWNSVILLE
BRISBANE STATE HIGH SCHOOL
SCIENTIA est POTESTAS
FERNY GROVE STATE HIGH SCHOOL
ALWAYS AIM HIGH
ELANORA STATE HIGH SCHOOL
URANGAN S.H.S.
WORKING TOGETHER
CENTENARY HEIGHTS
CLEVELAND
HIGH
SEMPER DIGNE
SUNSHINE BEACH HIGH
QUALITY IN ALL WE DO
MERRIMAC STATE HIGH SCHOOL
PRIDE IN EXCELLENCE
MAROOCHYDORE
HIGH SCHOOL
NITI ET SERVIRE
STATE HIGH SCHOOL
CAIRNS
COMPASSION
FAITH
?

Multi-task learning:
the example of predicting student marks



There are a lot of multi-target prediction problems around...

Upcoming article

Multi-target prediction:
A unifying view on problems and methods

Willem Waegeman · Krzysztof Dembczyński · Eyke Hüllermeier

Received: date / Accepted: date

Abstract Multi-target prediction (MTP) is concerned with the simultaneous prediction of multiple target variables of diverse type. Due to its enormous application potential, it has developed into an active and rapidly expanding research field that combines several subfields of machine learning, including multivariate regression, multi-label classification, multi-task learning, dyadic prediction, zero-shot learning, network inference, and matrix completion. In this paper, we present a unifying view on MTP problems and methods. First, we formally discuss commonalities and differences between existing MTP problems. To this end, we introduce a general framework that covers the above subfields as special cases. As a second contribution, we provide a structured overview of MTP methods. This is accomplished by identifying a number of key properties, which distinguish such methods and determine their suitability for different types of problems. Finally, we also discuss a few challenges for future research.

https://arxiv.org/abs/1809.02352

Overview of this talk

General framework

Definition (Multi-target prediction)

A multi-target prediction setting is characterized by instances $\boldsymbol{x} \in \mathcal{X}$ and targets $\boldsymbol{t} \in \mathcal{T}$ with the following properties:

P1. A training dataset $\mathcal{D}$ consists of triplets $(\boldsymbol{x}_i, \boldsymbol{t}_j, y_{ij})$, where $y_{ij} \in \mathcal{Y}$ denotes a score that characterizes the relationship between the instance $\boldsymbol{x}_i$ and the target $\boldsymbol{t}_j$.

P2. In total, n different instances and m different targets are observed during training, with n and m finite numbers. Thus, the scores y_{ij} of the training data can be arranged in an $n \times m$ matrix Y, which is in general incomplete, i.e., Y has missing values.

P3. The score set $\mathcal{Y}$ is one-dimensional. It consists of nominal, ordinal or real values.

P4. The goal consists of predicting scores for any instance-target couple $(\boldsymbol{x}, \boldsymbol{t}) \in \mathcal{X} \times \mathcal{T}$.

Conventional MTP settings

- Side information for targets is normally not available.
- **Multivariate regression** (e.g., predicting whether a protein will bind to a set of experimentally developed small molecules).
- **Multi-label classification** (e.g., assigning appropriate category tags to documents).
- **Multi-task learning** (e.g., predicting student marks in the final exam for a typical high-school course).

Conventional MTP settings

Definition (Multivariate regression)

A multivariate regression problem is a specific instantiation of the general framework, which exhibits the following additional properties:

P5. The cardinality of $\mathcal{T}$ is m. This implies that all targets are observed during training.

P6. No side information is available for targets. Without loss of generality, we can hence assign the numbers 1 to m as identifiers to targets, such that the target space is $\mathcal{T} = \{1, ..., m\}$.

P7. The score matrix Y has no missing values.

P8. The score set is $\mathcal{Y} = \mathbb{R}$.

Multivariate regression

		Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101		1,3	0,2	1,4	1,7	3,5	1,3
00111		2	1,7	1,5	7,5	8,2	7,6
01110		0,2	0	0,3	0,4	1,2	2,2
10001		3,1	1,1	1,3	1,1	1,7	5,2
01011		4,7	2,1	2,5	1,5	2,3	8,5
11110		?	?	?	?	?	?

Conventional MTP settings

Definition (Multi-task learning)

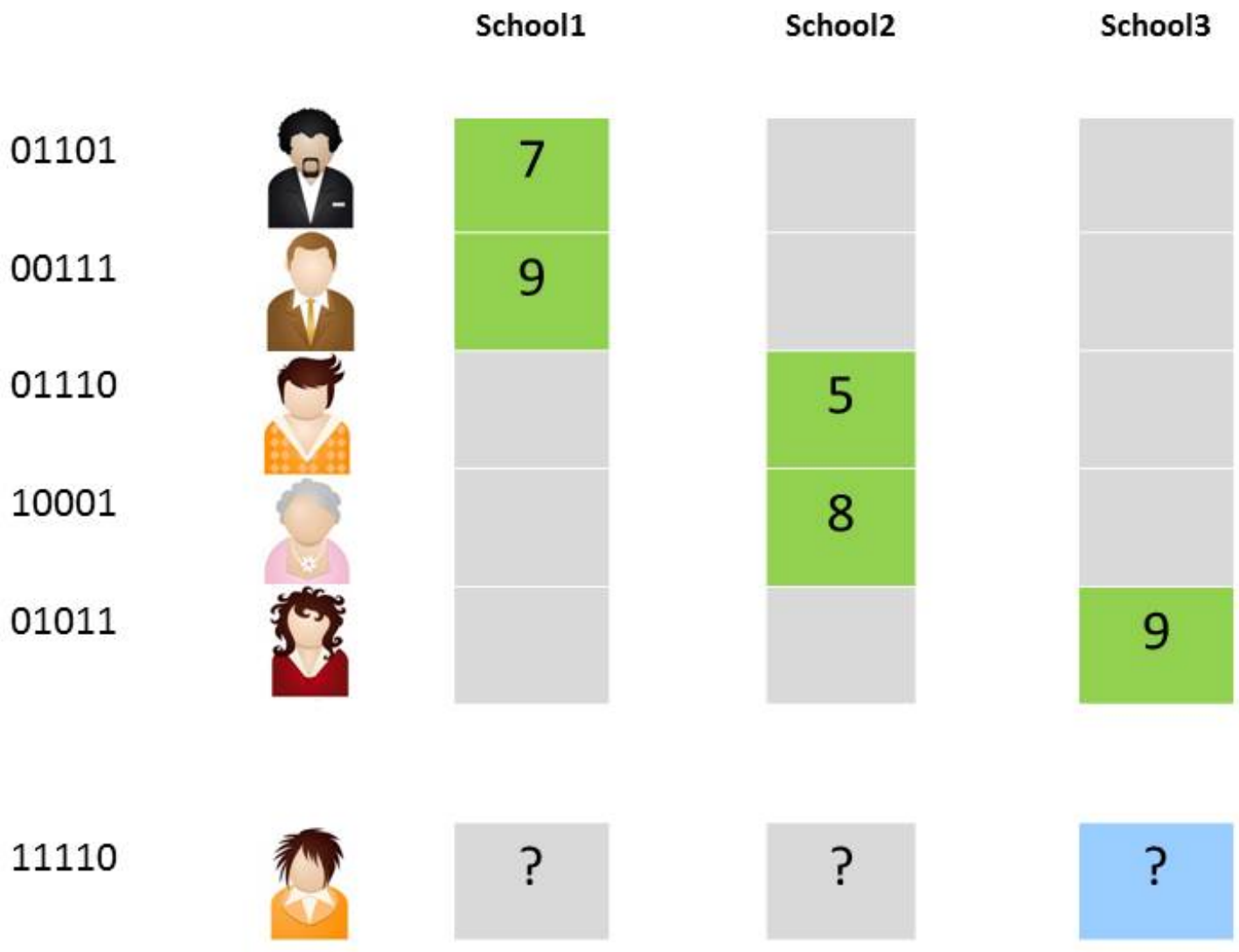
A multi-task learning problem is a specific instantiation of the general framework, which exhibits the following additional properties:

P5. The cardinality of $\mathcal{T}$ is m; this implies that all targets are observed during training.

P6. No side information is available for targets. Again, the target space can hence be taken as $\mathcal{T} = \{1, ..., m\}$.

P8a. The score set is homogenous across columns of Y, e.g., $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \mathbb{R}$.

Multi-task learning



Conventional MTP settings

Definition (Multi-label classification)

A multi-label classification problem is a specific instantiation of the general framework, which exhibits the following additional properties:

P5. The cardinality of $\mathcal{T}$ is m; this implies that all targets are observed during training.

P6. No side information is available for targets. Again, without loss of generality, we can hence identify targets with natural numbers, such that the target space is $\mathcal{T} = \{1, ..., m\}$.

P7. The score matrix Y has no missing values.

P8b. The score set is $\mathcal{Y} = \{0, 1\}$.

Multi-label classification

		Tennis	Football	Biking	Movies	TV	Belgium
01101	Text1	0	1	0	0	1	1
00111	Text2	1	0	0	0	0	1
01110	Text3	0	0	0	1	1	0
10001	Text4	0	0	1	0	1	0
01011	Text5	1	0	0	1	0	0
11110	Text6	?	?	?	?	?	?

Conventional MTP settings

Definition (Label ranking)

A multi-label classification problem is a specific instantiation of the general framework, which exhibits the following additional properties:

P5. The cardinality of $\mathcal{T}$ is m; this implies that all targets are observed during training.

P6. No side information is available for targets. Again, without loss of generality, we can hence identify targets with natural numbers, such that the target space is $\mathcal{T} = \{1, ..., m\}$.

P7. The score matrix Y has no missing values.

P8c. The score set is $\mathcal{Y} = \{1, \ldots, m\}$, and the scores (interpreted as ranks) are such that $y_{ij} \neq y_{ik}$ for all $1 \leq j, k \neq m$.

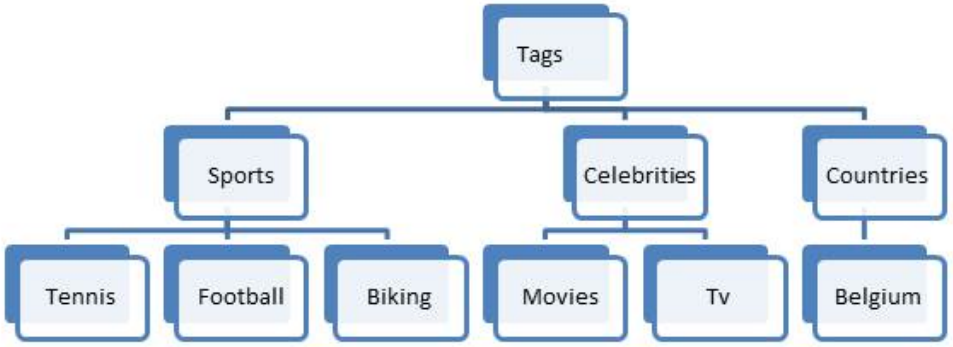
Conventional MTP settings

- In **label ranking**[1], each instance is associated with a ranking (total order) of the targets.

		Tennis	Football	Biking	Skating	Running	Walking
01101	User 1	2	1	4	3	5	6
00111	User 2	1	4	3	5	6	2
01110	User 3	4	5	1	2	3	6
10001	User 4	4	3	2	6	1	5
01011	User t5	1	3	5	2	6	4
11110	User 6	?	?	?	?	?	?

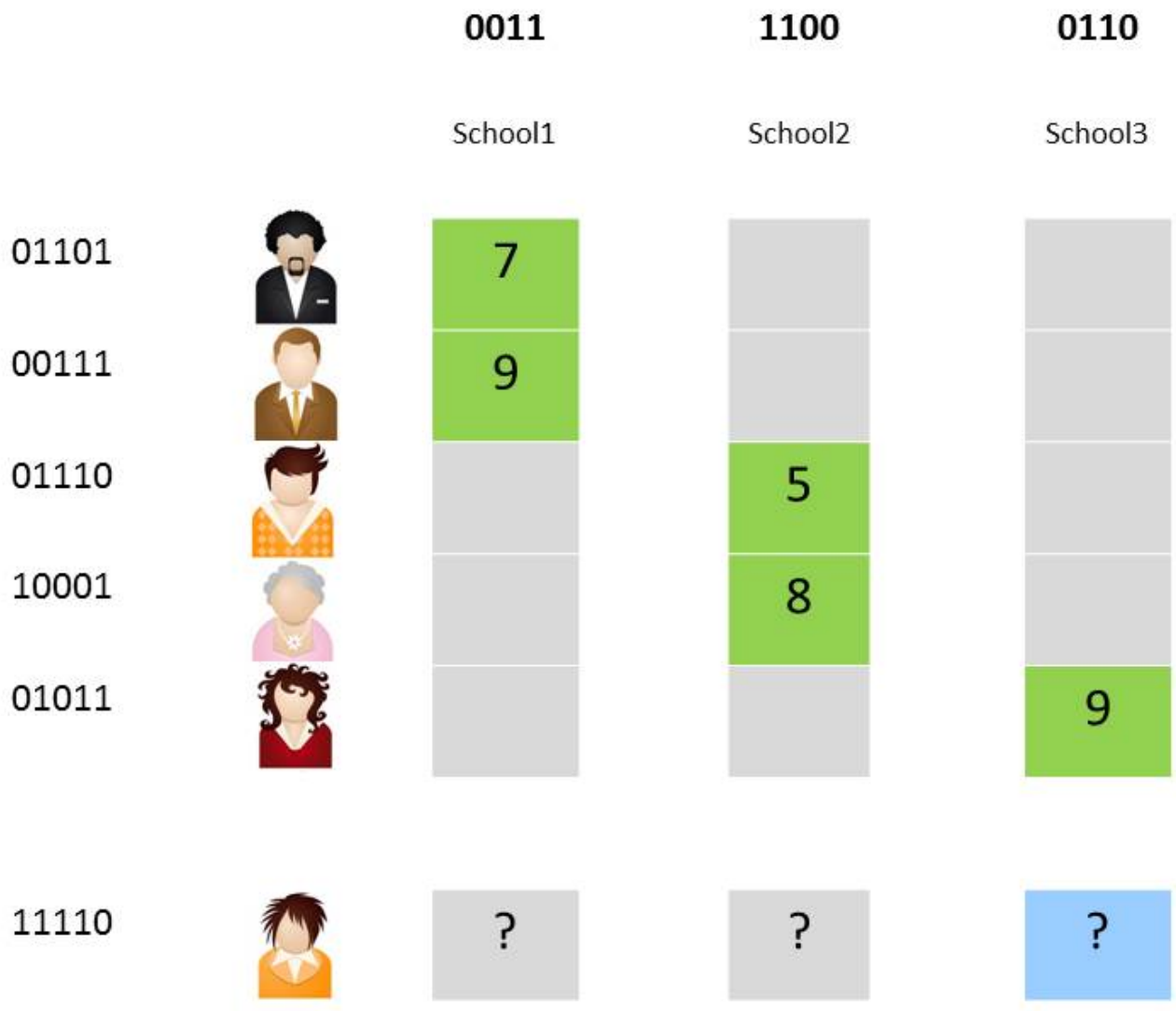
[1] E.H., J. Fürnkranz, W. Cheng, K. Brinker. Label Ranking by Learning Pairwise Preferences, Artificial Intelligence, 172, 2008.

Let's assume a document hierarchy:
How would you call this machine learning problem?

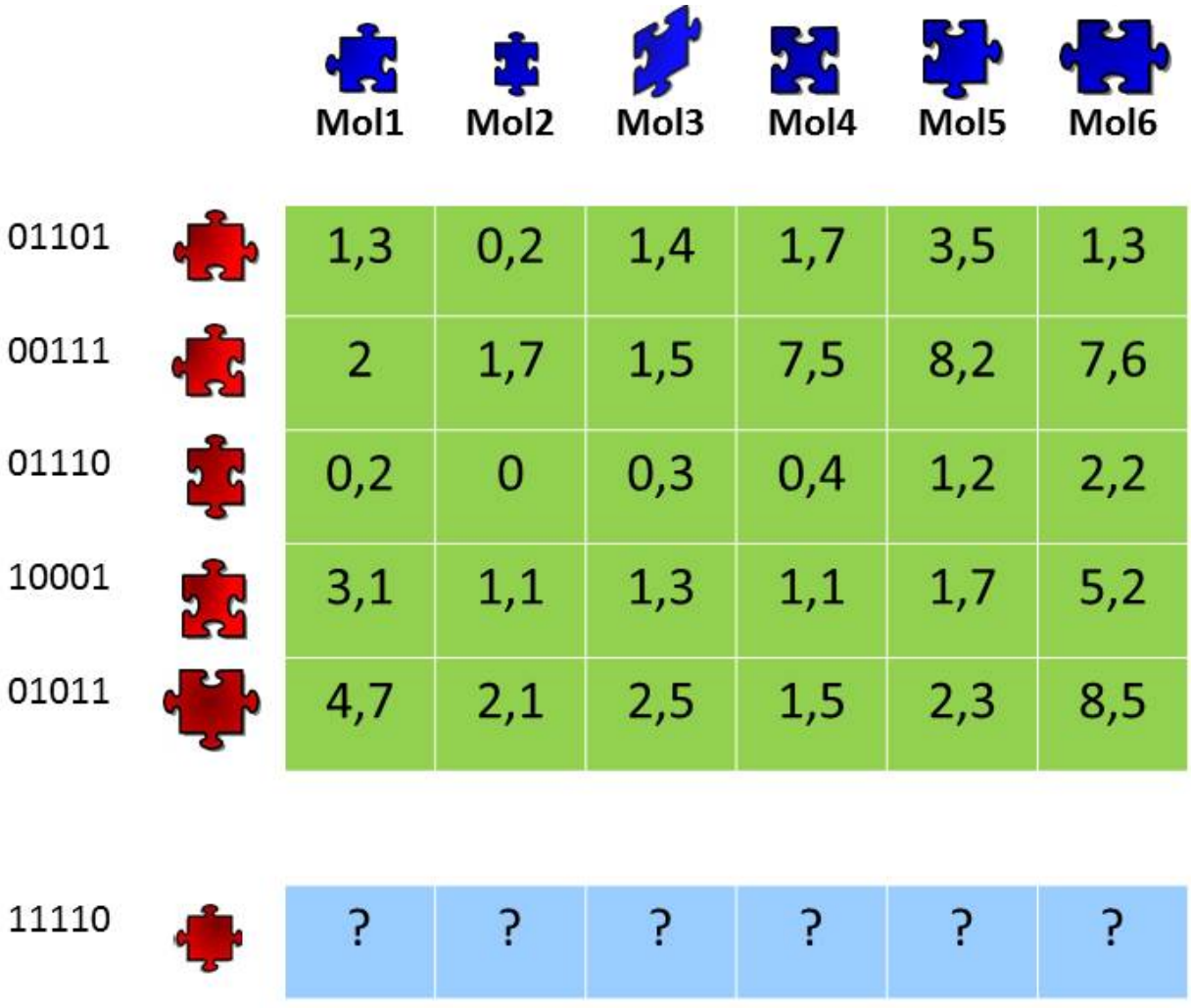


01101	**Text1**	0	0	0	0	0	1
00111	**Text2**	0	0	1	0	1	1
01110	**Text3**	0	0	0	1	1	0
10001	**Text4**	0	0	1	0	1	0
01011	**Text5**	1	0	0	1	0	0
11110	**Text6**	?	?	?	?	?	?

Let's assume a target representation:
How would you call this machine learning problem?



Let's assume a target representation:
How would you call this machine learning problem?



Learning with side information on targets

- Additional side information about the target space is available.
- Examples:
  - Representation for the target molecules in drug design application (**structured representation**).
  - Taxonomie on document categories (**hierarchy**).
  - Information about schools and courses (geographical location, qualifications of the teachers, reputation of the school, etc.) in student mark forecasting application (**feature representation**).
- Such problems are often referred to as dyadic prediction, link prediction, or network inference settings.

Learning with side information on targets

- Generally speaking, such settings cover problems that obey the four properties listed in the MTP definition.
- Labels y_{ij} can be arranged in a matrix Y, which is often sparse.
- Thus, one may argue that **dyadic prediction** is nothing else than **multi-task learning with task features**.
- However, MTP terminology is rarely used in the dyadic prediction literature.

Inductive versus transductive learning problems

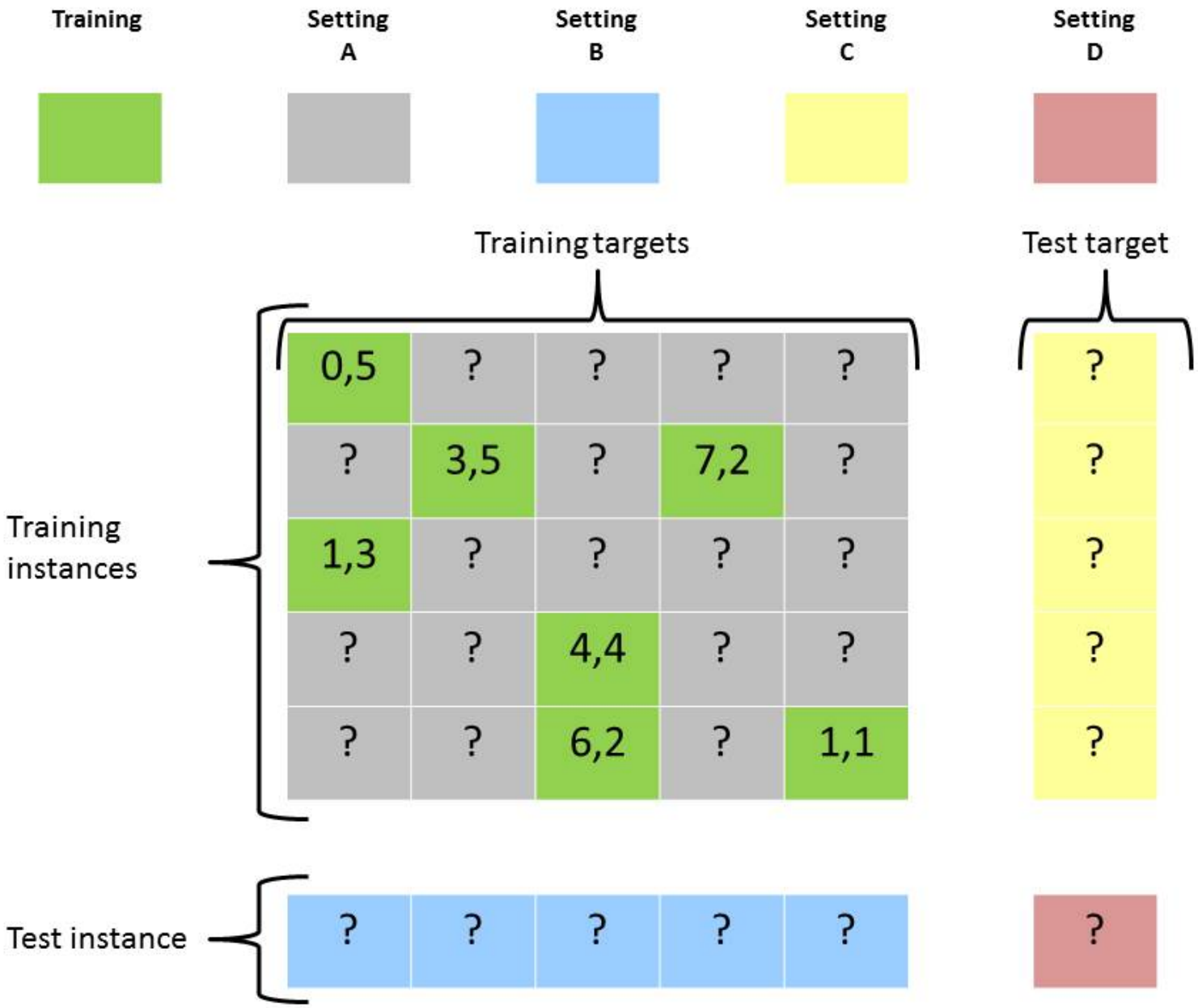
- In the previous problems,
  - predictions need be be generated for novel instances,
  - whereas the set of targets is known beforehand and observed during the training phase.
- These problems are **inductive** w.r.t. instances and **transductive** w.r.t. targets.
- **Side information** is of crucial importance for generalizing to novel targets that are unobserved during the training phase.

Inductive versus transductive learning problems

$g(.,.)$: target similarity

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6	Mol7
01101 	1,3	0,2	1,4	1,7	3,5	1,3	?
00111 	2	1,7	1,5	7,5	8,2	7,6	?
01110 	0,2	0	0,3	0,4	1,2	2,2	?
10001 	3,1	1,1	1,3	1,1	1,7	5,2	?
01011 	4,7	2,1	2,5	1,5	2,3	8,5	?
11110 	?	?	?	?	?	?	?

Important subdivision of different learning settings



Inductive versus transductive learning problems

Definition (Zero-shot learning)

A zero-shot learning problem is a specific instantiation of the general framework with the following additional property:

P5*. $m < m^* = |\mathcal{T}|$. Some targets are hence not observed during training, but may nevertheless appear at prediction time.

Inductive versus transductive learning problems

Definition (Zero-shot learning)

A zero-shot learning problem is a specific instantiation of the general framework with the following additional property:

P5*. $m < m^* = |\mathcal{T}|$. Some targets are hence not observed during training, but may nevertheless appear at prediction time.

- By substituting P5 with P5*, one now tackles problems that are inductive instead of transductive w.r.t. targets.
- The same subdivision can be made for instances.
- In total, the four different settings referred to as A, B, C, D can be distinguished (in the presence of side information).
- Theoretically, settings B and C are identical/symmetric, though there are practical differences/asymmetries.

Inductive versus transductive learning problems

Definition (Matrix completion)

A matrix completion problem is a specific instantiation of the general framework with the following additional properties:

P5. The cardinality of $\mathcal{T}$ is m. This implies that all targets are observed during training.

P6. No side information is available for targets. Without loss of generality, we can hence assign identifiers to targets from the set $\{1, ..., m\}$ such that the target space is $\mathcal{T} = \{1, ..., m\}$.

P9. The cardinality of $\mathcal{X}$ is n. This implies that all instances are observed during training.

P10. No side information is available for instances. Without loss of generality, we can hence assign identifiers to instances from the set $\{1, ..., n\}$, such that the instance space is $\mathcal{X} = \{1, ..., n\}$.

What we don't cover under MTP

- Our formal framework is rather generic.

What we don't cover under MTP

- Our formal framework is rather generic.
- In principle, every prediction problem with (original) output space $\mathcal{Y}$ could be seen as a special case by taking $\mathcal{T} = \mathcal{Y}$ and $\{0, 1\}$ as a score set (or through any other binary reduction).

What we don't cover under MTP

- Our formal framework is rather generic.
- In principle, every prediction problem with (original) output space $\mathcal{Y}$ could be seen as a special case by taking $\mathcal{T} = \mathcal{Y}$ and $\{0, 1\}$ as a score set (or through any other binary reduction).
- Each candidate output is treated as a target, the task is to predict whether or not the sought output corresponds to that candidate.

What we don't cover under MTP

- Our formal framework is rather generic.
- In principle, every prediction problem with (original) output space $\mathcal{Y}$ could be seen as a special case by taking $\mathcal{T} = \mathcal{Y}$ and $\{0, 1\}$ as a score set (or through any other binary reduction).
- Each candidate output is treated as a target, the task is to predict whether or not the sought output corresponds to that candidate.
- Consequently, a consistent prediction has to obey strong (deterministic) dependencies between the targets (a single 1, rest 0).

What we don't cover under MTP

- Our formal framework is rather generic.
- In principle, every prediction problem with (original) output space $\mathcal{Y}$ could be seen as a special case by taking $\mathcal{T} = \mathcal{Y}$ and $\{0, 1\}$ as a score set (or through any other binary reduction).
- Each candidate output is treated as a target, the task is to predict whether or not the sought output corresponds to that candidate.
- Consequently, a consistent prediction has to obey strong (deterministic) dependencies between the targets (a single 1, rest 0).
- Includes multi-class classification and structured output prediction (SOP) as special cases.

What we don't cover under MTP

- Our formal framework is rather generic.
- In principle, every prediction problem with (original) output space $\mathcal{Y}$ could be seen as a special case by taking $\mathcal{T} = \mathcal{Y}$ and $\{0, 1\}$ as a score set (or through any other binary reduction).
- Each candidate output is treated as a target, the task is to predict whether or not the sought output corresponds to that candidate.
- Consequently, a consistent prediction has to obey strong (deterministic) dependencies between the targets (a single 1, rest 0).
- Includes multi-class classification and structured output prediction (SOP) as special cases.

What we don't cover under MTP

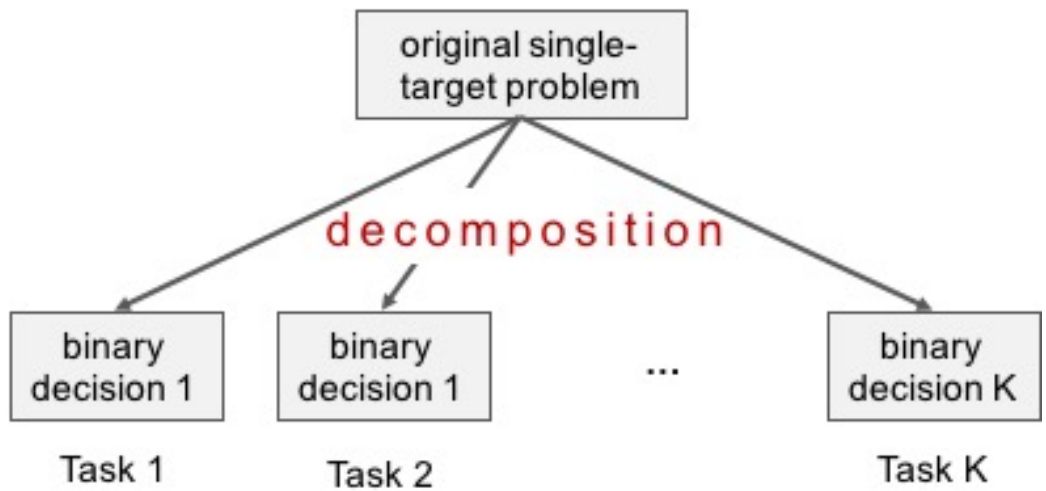
- Multi-class classification as a special case of MTP (?)

		Class
01101	Inst. 1	5
00111	Inst. 2	1
01110	Inst. 3	4
10001	Inst. 4	3
01011	Inst. 5	2
11110	Inst. 6	?

		Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
01101	Inst. 1	0	0	0	0	1	0
00111	Inst. 2	1	0	0	0	0	0
01110	Inst. 3	0	0	0	1	0	0
10001	Inst. 4	0	0	1	0	0	0
01011	Inst. 5	0	1	0	0	0	0
11110	Inst. 6	?	?	?	?	?	?

What we don't cover under MTP

- **Conceptually**, viewing each candidate prediction as a separate target appears artificial. Actually, one is still interested in a single prediction, not multiple ones.
- The multi-target problem was only produced through **decomposition** of a singe-target problem into multiple binary tasks/decisions.
- To comply with the corresponding **consistency constraints**, a kind of post-processing (like the decoding step in ECOC) is normally required (separation over tasks is not even possible in principal).



What we don't cover under MTP

- **Algorithmically**, the multi-target perspective is not typical of SOP. Instead, such methods are specifically tailored for output spaces that are often huge but equipped with a strong structure.
- Also excluded are prediction problems where the ground truth cannot be represented in a matrix format with optional side information, such as problems involving multi-instance learning representations or dyadic feature representations.

Overview of this talk

Multi-target prediction

- For a feature vector $\boldsymbol{x}$, predict a vector of responses $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$ using a function/hypothesis $\boldsymbol{h}$:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_p) \quad \xrightarrow{\boldsymbol{h}(\boldsymbol{x})} \quad \hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m)$$

Multi-target prediction

- For a feature vector $\boldsymbol{x}$, predict a vector of responses $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$ using a function/hypothesis $\boldsymbol{h}$:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_p) \quad \xrightarrow{\boldsymbol{h}(\boldsymbol{x})} \quad \hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m)$$

- Compared to single-target prediction, a multitude of **multivariate loss functions**

$$\ell : \mathcal{Y}^m \times \mathcal{Y}^m \to \mathbb{R}$$

  are conceivable.

Multi-target prediction

- For a feature vector $\boldsymbol{x}$, predict a vector of responses $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$ using a function/hypothesis $\boldsymbol{h}$:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_p) \quad \xrightarrow{\boldsymbol{h}(\boldsymbol{x})} \quad \hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m)$$

- Compared to single-target prediction, a multitude of **multivariate loss functions**

$$\ell : \mathcal{Y}^m \times \mathcal{Y}^m \to \mathbb{R}$$

  are conceivable.
- **Problem:** Given a target loss ℓ, find a (Bayes) predictor $\boldsymbol{h}$ that minimizes expected loss with regard to ℓ.

Multi-target prediction

- For a feature vector $\boldsymbol{x}$, predict a vector of responses $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$ using a function/hypothesis $\boldsymbol{h}$:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_p) \quad \xrightarrow{\boldsymbol{h}(\boldsymbol{x})} \quad \hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m)$$

- Compared to single-target prediction, a multitude of **multivariate loss functions**

$$\ell : \mathcal{Y}^m \times \mathcal{Y}^m \to \mathbb{R}$$

  are conceivable.
- **Problem:** Given a target loss ℓ, find a (Bayes) predictor $\boldsymbol{h}$ that minimizes expected loss with regard to ℓ.
- **Key question:** Can we achieve this goal through simple reduction, i.e., by training one model for each target independently? Or can we do better with more sophisticated methods?

The individual target view

- How can we improve the predictive accuracy of a single label by exploiting information about other labels?
- Goal: predict a value of y_i using $\boldsymbol{x}$ and any available information on other targets y_j.
- The problem is usually defined through univariate losses $\ell_i(y_i, \hat{y}_i)$.
- Domain of y_i is either continuous or nominal.
- Independent models vs. regularized (shrunken) models.
- James-Stein paradox (to be discussed later).

The joint target view

- The problem is defined through multivariate losses $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})$.
- Is reduction to single-target prediction (decomposition over targets) still possible, and even if so, can we improve over such strategies by using more expressive models?
- Important: **Structure of loss** $\ell(\cdot)$, possible **dependencies between targets**, multivariate distribution of $\boldsymbol{y}$.

Multivariate loss functions

- **Decomposable** over examples: A loss L is decomposable over examples if it can be written in the form

$$L = \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, \boldsymbol{h}(\boldsymbol{x}_i)) \,,$$

  i.e., as a sum of losses over all (test) examples.

Multivariate loss functions

- **Decomposable** over examples: A loss L is decomposable over examples if it can be written in the form

$$L = \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, \boldsymbol{h}(\boldsymbol{x}_i)) \,,$$

  i.e., as a sum of losses over all (test) examples.

- **Decomposable** over targets: A multivariate loss ℓ is decomposable over targets if it can be written as

$$\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \sum_{i=1}^{m} \ell_i(y_i, h_i(\boldsymbol{x}))$$

  with suitable single-target losses ℓ_i.

Macro- and micro-averaging

- Macro-averaging

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

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

$$L = \frac{1}{4}\left(L_1 + L_2 + L_3 + L_4\right)$$

Macro- and micro-averaging

- Macro-averaging

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

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

$$L = \frac{1}{4}\left(L_1 + L_2 + L_3 + L_4\right)$$

Macro- and micro-averaging

- Macro-averaging

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

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

$$L = \frac{1}{4}\left(L_1 + L_2 + L_3 + L_4\right)$$

Macro- and micro-averaging

- Macro-averaging

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

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

$$L = \frac{1}{4}\left(L_1 + L_2 + L_3 + L_4\right)$$

Macro- and micro-averaging

- Macro-averaging

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

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

$$L = \frac{1}{4}\left(L_1 + L_2 + L_3 + L_4\right)$$

Macro- and micro-averaging

- Micro-averaging

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

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

$$L = \sum_{i,j} \ell(y_{ij}, \hat{y}_{ij})$$

Macro- and micro-averaging

- Micro-averaging

True labels

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

Predicted labels

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

$$L = \sum_{i,j} \ell(y_{ij}, \hat{y}_{ij})$$

Same weight of every prediction vs. same weight of every target.

Instance-wise losses

- Averaging over instances

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

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

$$L = \frac{1}{6}\Big(\ell(\boldsymbol{y}_1, \hat{\boldsymbol{y}}_1) + \ell(\boldsymbol{y}_2, \hat{\boldsymbol{y}}_2) + \ell(\boldsymbol{y}_3, \hat{\boldsymbol{y}}_3) + \ell(\boldsymbol{y}_4, \hat{\boldsymbol{y}}_4) + \ell(\boldsymbol{y}_5, \hat{\boldsymbol{y}}_5) + \ell(\boldsymbol{y}_6, \hat{\boldsymbol{y}}_6)\Big)$$

Instance-wise losses

- Averaging over instances

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

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

$$L = \frac{1}{6}\Big(\ell(\boldsymbol{y}_1, \hat{\boldsymbol{y}}_1) + \ell(\boldsymbol{y}_2, \hat{\boldsymbol{y}}_2) + \ell(\boldsymbol{y}_3, \hat{\boldsymbol{y}}_3) + \ell(\boldsymbol{y}_4, \hat{\boldsymbol{y}}_4) + \ell(\boldsymbol{y}_5, \hat{\boldsymbol{y}}_5) + \ell(\boldsymbol{y}_6, \hat{\boldsymbol{y}}_6)\Big)$$

Instance-wise losses

- Averaging over instances

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

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

$$L = \frac{1}{6}\Big(\ell(\boldsymbol{y}_1, \hat{\boldsymbol{y}}_1) + \ell(\boldsymbol{y}_2, \hat{\boldsymbol{y}}_2) + \ell(\boldsymbol{y}_3, \hat{\boldsymbol{y}}_3) + \ell(\boldsymbol{y}_4, \hat{\boldsymbol{y}}_4) + \ell(\boldsymbol{y}_5, \hat{\boldsymbol{y}}_5) + \ell(\boldsymbol{y}_6, \hat{\boldsymbol{y}}_6)\Big)$$

Instance-wise losses

- Averaging over instances

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

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

$$L = \frac{1}{6}\Big(\ell(\boldsymbol{y}_1, \hat{\boldsymbol{y}}_1) + \ell(\boldsymbol{y}_2, \hat{\boldsymbol{y}}_2) + \ell(\boldsymbol{y}_3, \hat{\boldsymbol{y}}_3) + \ell(\boldsymbol{y}_4, \hat{\boldsymbol{y}}_4) + \ell(\boldsymbol{y}_5, \hat{\boldsymbol{y}}_5) + \ell(\boldsymbol{y}_6, \hat{\boldsymbol{y}}_6)\Big)$$

Instance-wise losses

- Averaging over instances

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

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

$$L = \frac{1}{6}\Big(\ell(\boldsymbol{y}_1, \hat{\boldsymbol{y}}_1) + \ell(\boldsymbol{y}_2, \hat{\boldsymbol{y}}_2) + \ell(\boldsymbol{y}_3, \hat{\boldsymbol{y}}_3) + \ell(\boldsymbol{y}_4, \hat{\boldsymbol{y}}_4) + \ell(\boldsymbol{y}_5, \hat{\boldsymbol{y}}_5) + \ell(\boldsymbol{y}_6, \hat{\boldsymbol{y}}_6)\Big)$$

Instance-wise losses

- Averaging over instances

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

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

$$L = \frac{1}{6}\Big(\ell(\boldsymbol{y}_1, \hat{\boldsymbol{y}}_1) + \ell(\boldsymbol{y}_2, \hat{\boldsymbol{y}}_2) + \ell(\boldsymbol{y}_3, \hat{\boldsymbol{y}}_3) + \ell(\boldsymbol{y}_4, \hat{\boldsymbol{y}}_4) + \ell(\boldsymbol{y}_5, \hat{\boldsymbol{y}}_5) + \ell(\boldsymbol{y}_6, \hat{\boldsymbol{y}}_6)\Big)$$

Instance-wise losses

- Averaging over instances

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

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

$$L = \frac{1}{6}\Big(\ell(\boldsymbol{y}_1, \hat{\boldsymbol{y}}_1) + \ell(\boldsymbol{y}_2, \hat{\boldsymbol{y}}_2) + \ell(\boldsymbol{y}_3, \hat{\boldsymbol{y}}_3) + \ell(\boldsymbol{y}_4, \hat{\boldsymbol{y}}_4) + \ell(\boldsymbol{y}_5, \hat{\boldsymbol{y}}_5) + \ell(\boldsymbol{y}_6, \hat{\boldsymbol{y}}_6)\Big)$$

Examples of MTP loss functions

- **F-measure**:

$$F(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{2 \sum_{i=1}^{K} y_i \hat{y}_i}{\sum_{i=1}^{K} y_i + \sum_{i=1}^{K} \hat{y}_i},$$

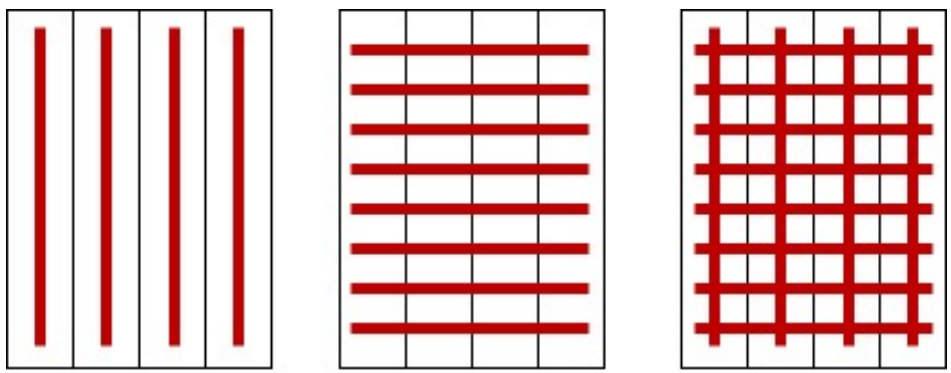
where $\mathbf{Y} = (y_1, \ldots, y_K) \in \{0, 1\}^K$ and $\hat{\mathbf{Y}} = (\hat{y}_1, \ldots, \hat{y}_K) \in \{0, 1\}^K$.
Can be used in macro- and micro-averaging, but also instance-wise.

Examples of MTP loss functions

- **F-measure**:

$$F(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{2\sum_{i=1}^{K} y_i \hat{y}_i}{\sum_{i=1}^{K} y_i + \sum_{i=1}^{K} \hat{y}_i},$$

where $\mathbf{Y} = (y_1, \ldots, y_K) \in \{0,1\}^K$ and $\hat{\mathbf{Y}} = (\hat{y}_1, \ldots, \hat{y}_K) \in \{0,1\}^K$.
Can be used in macro- and micro-averaging, but also instance-wise.



Examples of MTP loss functions

- **F-measure**:

$$F(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{2\sum_{i=1}^{K} y_i \hat{y}_i}{\sum_{i=1}^{K} y_i + \sum_{i=1}^{K} \hat{y}_i},$$

  where $\mathbf{Y} = (y_1, \ldots, y_K) \in \{0,1\}^K$ and $\hat{\mathbf{Y}} = (\hat{y}_1, \ldots, \hat{y}_K) \in \{0,1\}^K$. Can be used in macro- and micro-averaging, but also instance-wise.

- The **Hamming loss** averages over mistakes on individual labels:

$$\ell_H(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq y_i]\!]$$

Examples of MTP loss functions

- **F-measure**:

$$F(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{2 \sum_{i=1}^{K} y_i \hat{y}_i}{\sum_{i=1}^{K} y_i + \sum_{i=1}^{K} \hat{y}_i},$$

  where $\mathbf{Y} = (y_1, \ldots, y_K) \in \{0, 1\}^K$ and $\hat{\mathbf{Y}} = (\hat{y}_1, \ldots, \hat{y}_K) \in \{0, 1\}^K$. Can be used in macro- and micro-averaging, but also instance-wise.

- The **Hamming loss** averages over mistakes on individual labels:

$$\ell_H(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq y_i]\!]$$

- The **subset 0/1 loss** simply checks for entire correctness:

$$\ell_{0/1}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = [\![\boldsymbol{y} \neq \hat{\boldsymbol{y}}]\!] = \max_i [\![y_i \neq y_i]\!]$$

Examples of MTP loss functions

- The **rank loss** compares binary targets with a **predicted ranking**:

$$\ell_r(\boldsymbol{y}, \sigma) = \sum_{(i,j):y_i > y_j} \left([\![\sigma(i) > \sigma(j)]\!] + \frac{1}{2}[\![\sigma(i) = \sigma(j)]\!] \right),$$

  where σ is a permutation/ranking of the targets typically induced by a scoring function $\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^m$ s.t. $\sigma(i) \leq \sigma(j) \Leftrightarrow f_i(\boldsymbol{x}) \geq f_j(\boldsymbol{x})$.

Examples of MTP loss functions

- The **rank loss** compares binary targets with a **predicted ranking**:

$$\ell_r(\boldsymbol{y}, \sigma) = \sum_{(i,j):y_i>y_j} \left([\![\sigma(i) > \sigma(j)]\!] + \frac{1}{2}[\![\sigma(i) = \sigma(j)]\!] \right) ,$$

  where σ is a permutation/ranking of the targets typically induced by a scoring function $\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^m$ s.t. $\sigma(i) \leq \sigma(j) \Leftrightarrow f_i(\boldsymbol{x}) \geq f_j(\boldsymbol{x})$.

- **Precision at position k** compares a binary targets with **predicted top-ranking**:

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

  where $\hat{Y}_k$ is a set of k labels predicted by the learner (perhaps by thresholding a ranking/scoring function).

Examples of MTP loss functions

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

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

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

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

Examples of MTP loss functions

- **Squared error loss** (typically used in multivariate regression):

$$\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \,,$$

  where $\boldsymbol{y}, \hat{\boldsymbol{y}} \in \mathbb{R}^m$.

Two simple yet extreme multi-label losses

- The **Hamming loss** averages over mistakes on individual labels:

$$\ell_H(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq y_i]\!]$$

- The **subset 0/1 loss** simply checks for entire correctness:

$$\ell_{0/1}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = [\![\boldsymbol{y} \neq \hat{\boldsymbol{y}}]\!] = \max_i [\![y_i \neq y_i]\!]$$

Hamming vs. subset 0/1 loss

- What is the risk-minimizing (Bayes) prediction for the Hamming loss $1/m \sum_{i=1}^{m} [\![y_i \neq y_i]\!]$ and the subset 0/1 loss $[\![\boldsymbol{y} \neq \hat{\boldsymbol{y}}]\!]$, respectively, given the following conditional distribution $P(Y_1, Y_2 \,|\, \boldsymbol{x})$?

y_1	y_2	$P(y_1, y_2 \,\|\, \boldsymbol{x})$
0	0	0.3
0	1	0.3
1	0	0.0
1	1	0.4

Hamming vs. subset 0/1 loss

- What is the risk-minimizing (Bayes) prediction for the Hamming loss $1/m \sum_{i=1}^{m} [\![y_i \neq y_i]\!]$ and the subset 0/1 loss $[\![\boldsymbol{y} \neq \hat{\boldsymbol{y}}]\!]$, respectively, given the following conditional distribution $P(Y_1, Y_2 \,|\, \boldsymbol{x})$?

y_1	y_2	$P(y_1, y_2 \,\vert\, \boldsymbol{x})$
0	0	0.3
0	1	0.3
1	0	0.0
1	1	0.4

$$
\begin{aligned}
\mathbb{E}_{\mathbf{Y}} \ell_{0/1}(\boldsymbol{y}, \hat{\boldsymbol{y}}) &= 0.3\, \ell_{0/1}((0,0), \hat{\boldsymbol{y}}) + 0.3\, \ell_{0/1}((0,1), \hat{\boldsymbol{y}}) + \\
&\quad 0.0\, \ell_{0/1}((1,0), \hat{\boldsymbol{y}}) + 0.4\, \ell_{0/1}((1,1), \hat{\boldsymbol{y}}) \\
&= 1 - P(\hat{\boldsymbol{y}} \,|\, \boldsymbol{x})
\end{aligned}
$$

Hamming vs. subset 0/1 loss

- What is the risk-minimizing (Bayes) prediction for the Hamming loss $1/m \sum_{i=1}^{m} [\![y_i \neq y_i]\!]$ and the subset 0/1 loss $[\![\boldsymbol{y} \neq \hat{\boldsymbol{y}}]\!]$, respectively, given the following conditional distribution $P(Y_1, Y_2 \,|\, \boldsymbol{x})$?

y_1	y_2	$P(y_1, y_2)$
0	0	0.3
0	1	0.3
1	0	0.0
1	1	0.4

$$
\begin{aligned}
\mathbb{E}_{\mathbf{Y}} \ell_H(\boldsymbol{y}, \hat{\boldsymbol{y}}) &= 0.3\, \ell_H((0,0), \hat{\boldsymbol{y}}) + 0.3\, \ell_H((0,1), \hat{\boldsymbol{y}}) + 0.4\, \ell_H((1,1), \hat{\boldsymbol{y}}) \\
&= \frac{1}{2}\big(0.3([\![0 \neq y_1]\!] + [\![0 \neq y_2]\!]) + 0.3([\![0 \neq y_1]\!] + [\![1 \neq y_2]\!]) + \\
&\qquad 0.4([\![1 \neq y_1]\!] + [\![1 \neq y_2]\!])\big) \\
&= \frac{1}{2}\big(0.6[\![0 \neq y_1]\!] + 0.4[\![1 \neq y_1]\!] + 0.3[\![0 \neq y_2]\!] + 0.7[\![0 \neq y_2]\!]\big) \\
&= 1/2\, \mathbb{E}_{Y_1} \ell_H(y_1, \hat{y}_1) + 1/2\, \mathbb{E}_{Y_2} \ell_H(y_2, \hat{y}_2)
\end{aligned}
$$

Hamming vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is the **marginal mode**:

$$h_i^*(\boldsymbol{x}) = \arg \max_{y_i \in \{0,1\}} P(y_i \,|\, \boldsymbol{x}) \,, \quad i = 1, \ldots, m,$$

  while for the subset 0/1 loss it is the **joint mode**:

$$\mathbf{h}^*(\boldsymbol{x}) = \arg \max_{\boldsymbol{y} \in \mathcal{Y}} P(\boldsymbol{y} \,|\, \boldsymbol{x}) \,.$$

- Marginal mode vs. joint mode.

$\boldsymbol{y}$	$P(\boldsymbol{y})$
0 0 0 0	0.30
0 1 1 1	0.17
1 0 1 1	0.18
1 1 0 1	0.17
1 1 1 0	0.18

Marginal mode: 1 1 1 1
Joint mode: 0 0 0 0

Hamming vs. subset 0/1 loss

- **Proposition**[2]: The following upper bound holds for $m > 3$:

$$\mathbb{E}_{\mathbf{Y}}\, \ell_H(\mathbf{Y}, \mathbf{h}^*_{0/1}(\boldsymbol{x})) - \mathbb{E}_{\mathbf{Y}}\, \ell_H(\mathbf{Y}, \mathbf{h}^*_H(\boldsymbol{x})) < \frac{m-2}{m+2}$$

  Moreover, this bound is tight, i.e.

$$\sup_P \left(\mathbb{E}_{\mathbf{Y}}\, \ell_H(\mathbf{Y}, \mathbf{h}^*_{0/1}(\boldsymbol{x})) - \mathbb{E}_{\mathbf{Y}}\, \ell_H(\mathbf{Y}, \mathbf{h}^*_H(\boldsymbol{x}))\right) = \frac{m-2}{m+2}\,,$$

  where the supremum is taken over all probability distributions on $\mathcal{Y}$.

[2] K.D., W.W., W. Cheng, E.H. On Label Dependence and Loss Minimization in Multi-Label Classification. Machine Learning, 88, 2012.

Hamming vs. subset 0/1 loss

Under specific conditions, the risk minimizers for ℓ_H and $\ell_{0/1}$ are provably **equivalent**, i.e.,

$$\boldsymbol{h}^*_H(\boldsymbol{x}) = \boldsymbol{h}^*_{0/1}(\boldsymbol{x})\,,$$

for example, when

- the probability of the joint mode satisfies

$$P\big(\boldsymbol{h}^*_{0/1}(\boldsymbol{x}) \,|\, \boldsymbol{x}\big) > 0.5\,,$$

- or the targets $Y_1, \ldots, Y_m$ are **conditionally independent**.

Target dependence

We distinguish between conditional and unconditional (in)dependence of targets[3].

- **Unconditional/marginal dependence**:

$$P(\boldsymbol{Y}) \neq \prod_{i=1}^{m} P(Y_i)$$

Often due to model similarities, i.e., $f_i(\boldsymbol{x}) = g_i(\boldsymbol{x}) + \epsilon_i$ for $i = 1, \ldots, m$, with similarities in the structural parts $g_i(\cdot)$, which implies **correlation** between targets.

[3] K.D., W.W., W. Cheng, E.H. On Label Dependence and Loss Minimization in Multi-Label Classification. Machine Learning, 88, 2012.

Target dependence

We distinguish between conditional and unconditional (in)dependence of targets[3].

- **Unconditional/marginal dependence**:

$$P(\boldsymbol{Y}) \neq \prod_{i=1}^{m} P(Y_i)$$

  Often due to model similarities, i.e., $f_i(\boldsymbol{x}) = g_i(\boldsymbol{x}) + \epsilon_i$ for $i = 1, \ldots, m$, with similarities in the structural parts $g_i(\cdot)$, which implies **correlation** between targets.

- **Conditional dependence**:

$$P(\boldsymbol{Y} \mid \boldsymbol{x}) \neq \prod_{i=1}^{m} P(Y_i \mid \boldsymbol{x})$$

[3] K.D., W.W., W. Cheng, E.H. On Label Dependence and Loss Minimization in Multi-Label Classification. Machine Learning, 88, 2012.

Target dependence

- marginal (in)dependence $\not\Leftrightarrow$ conditional (in)dependence
- Example:

x_1	y_1	y_2	P	x_1	y_1	y_2	P
0	0	0	0.25	1	0	0	0
0	0	1	0	1	0	1	0.25
0	1	0	0	1	1	0	0.25
0	1	1	0.25	1	1	1	0

- Strong conditional dependence, for example $P(Y_1 = 0|x_1 = 1)P(Y_2 = 0|x_1 = 1) = 0.5 \times 0.5 = 0.25 \neq 0.$
- Yet, labels are marginally independent: Joint probability is the product of the marginals $P(y_1) = P(y_2) = 0.5$.

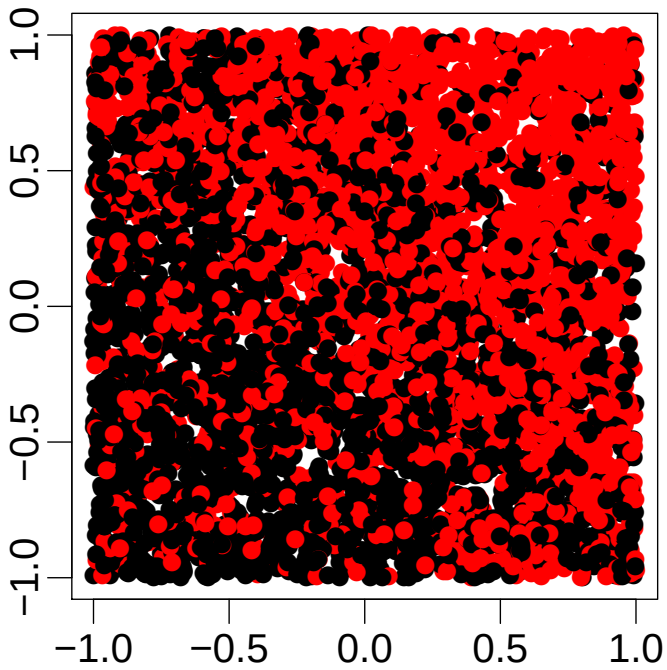
Synthetic data

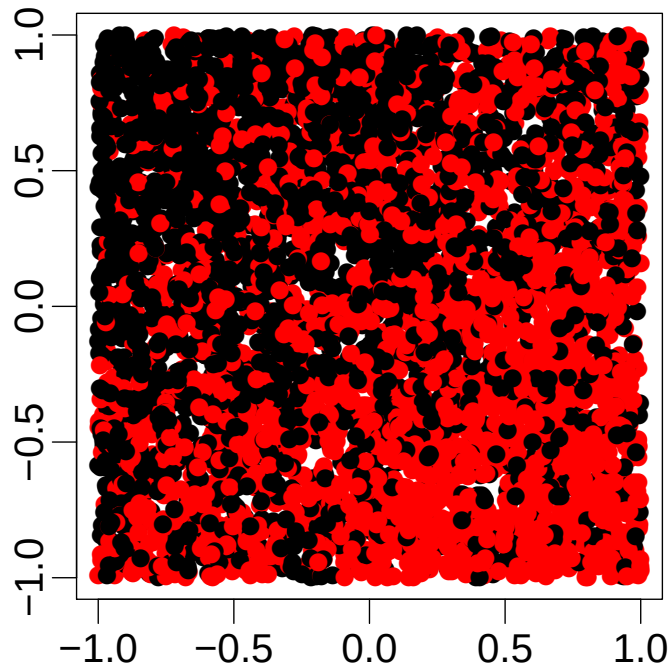
- Two conditionally independent models:

$$f_1(\boldsymbol{x}) = \frac{1}{2}x_1 + \frac{1}{2}x_2, \quad f_2(\boldsymbol{x}) = \frac{1}{2}x_1 - \frac{1}{2}x_2$$

- Logistic model to assign labels:

$$P(y_i = 1) = \frac{1}{1 + \exp(-2f_i)}$$





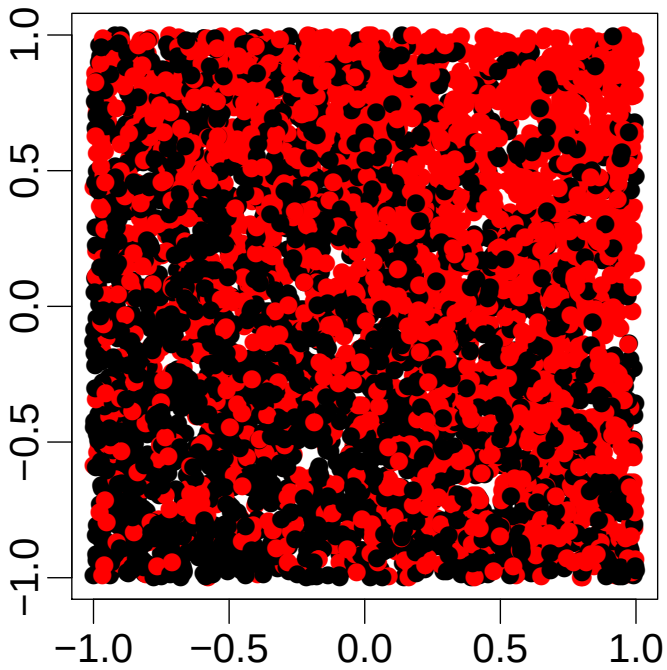
Synthetic data

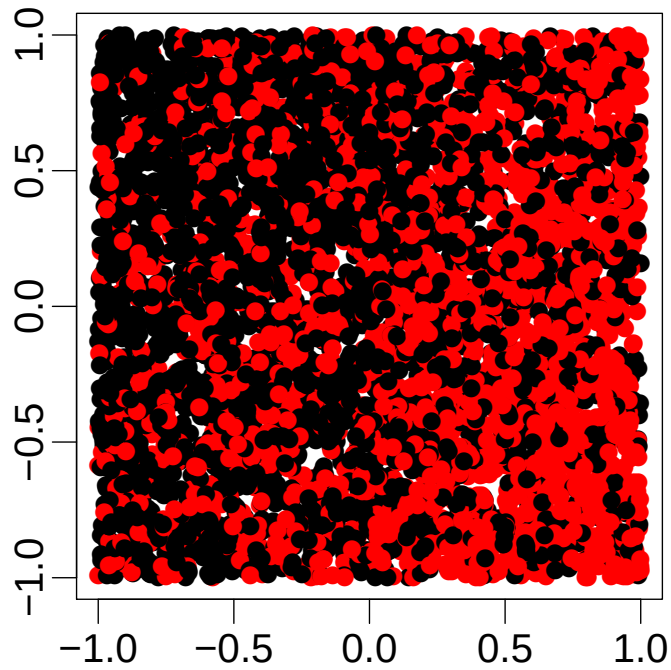
- Two conditionally dependent models:

$$f_1(\boldsymbol{x}) = \frac{1}{2}x_1 + \frac{1}{2}x_2 \quad f_2(y_1, \boldsymbol{x}) = y_1 + \frac{1}{2}x_1 - \frac{1}{2}x_2 - \frac{2}{3}$$

- Logistic model to assign labels:

$$P(y_i = 1) = \frac{1}{1 + \exp(-2f_i)}$$





Hamming vs. subset 0/1 loss

- Binary relevance (BR): Train two binary classifiers for targets y_1 and y_2 independently.
- Label powerset (LP): Train a 4-class classifier on meta-classes $c_1 = (0, 0)$, $c_2 = (0, 1)$, $c_3 = (1, 0)$, $c_4 = (1, 1)$.

Conditional independence		
classifier	Hamming loss	subset 0/1 loss
BR LR	0.4232	0.6723
LP LR	0.4232	0.6725

Conditional dependence		
classifier	Hamming loss	subset 0/1 loss
BR LR	0.3470	0.5499
LP LR	0.3610	0.5146

Rank loss

- The rank loss compares binary targets with a predicted ranking:

$$\ell_r(\boldsymbol{y}, \boldsymbol{f}(\boldsymbol{x})) = \sum_{(i,j):y_i>y_j} \left([\![f_i(\boldsymbol{x}) < f_j(\boldsymbol{x})]\!] + \frac{1}{2} [\![f_i(\boldsymbol{x}) = f_j(\boldsymbol{x})]\!] \right)$$

- To minimize this loss, it is enough to sort the targets by their probability of relevance.
- **Theorem**[4]: A ranking function that sorts the labels according to their probability of relevance, i.e., using the scoring function $\boldsymbol{f}(\cdot)$ with

$$f_i(\boldsymbol{x}) \;=\; P(Y_i = 1 \,|\, \boldsymbol{x})\,,$$

minimizes the expected rank loss.

[4] K.D., W.W., W. Cheng, E.H. On Label Dependence and Loss Minimization in Multi-Label Classification. Machine Learning, 88, 2012.

Loss functions and target dependence

- Optimal (pointwise) prediction requires information about $P(\boldsymbol{y} \mid \boldsymbol{x})$.

Loss functions and target dependence

- Optimal (pointwise) prediction requires information about $P(\boldsymbol{y} \mid \boldsymbol{x})$.
- **Independence** simplifies learning a lot, since learning marginals is much easier than learning a joint distribution ($\rightarrow$ graphical models).

Loss functions and target dependence

- Optimal (pointwise) prediction requires information about $P(\boldsymbol{y} \,|\, \boldsymbol{x})$.
- **Independence** simplifies learning a lot, since learning marginals is much easier than learning a joint distribution ($\rightarrow$ graphical models).
- **Structure of the loss function** has an important influence, too, due to the "interaction" between ℓ and P:

$$\mathbb{E}_{\mathbf{Y}}\, \ell(\mathbf{Y}, \hat{\boldsymbol{y}}) = \sum_{\boldsymbol{y}} \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) P(\boldsymbol{y} \,|\, \boldsymbol{x})\,.$$

In some cases, such as F-measure optimization, knowledge of properties of P instead of complete distribution is therefore enough.

Loss functions and target dependence

- Optimal (pointwise) prediction requires information about $P(\boldsymbol{y} \,|\, \boldsymbol{x})$.
- **Independence** simplifies learning a lot, since learning marginals is much easier than learning a joint distribution ($\rightarrow$ graphical models).
- **Structure of the loss function** has an important influence, too, due to the "interaction" between ℓ and P:

$$\mathbb{E}_{\mathbf{Y}} \, \ell(\mathbf{Y}, \hat{\boldsymbol{y}}) = \sum_{\boldsymbol{y}} \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) P(\boldsymbol{y} \,|\, \boldsymbol{x}) \,.$$

  In some cases, such as F-measure optimization, knowledge of properties of P instead of complete distribution is therefore enough.
- **Conditional independence** of P and **decomposability** of ℓ are sufficient conditions for Bayes-optimality of target-wise Bayes predictor.

Overview of this talk

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B
Similarity-enforcing methods	B
Relation-exploiting methods	B and D
Relation-constructing methods	B
Representation-exploiting methods	B and D
Representation-constructing methods	A and B

A baseline method:
learning a model for each target independently

		Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101		1,3	0,2	1,4	1,7	3,5	1,3
00111		2	1,7	1,5	7,5	8,2	7,6
01110		0,2	0	0,3	0,4	1,2	2,2
10001		3,1	1,1	1,3	1,1	1,7	5,2
01011		4,7	2,1	2,5	1,5	2,3	8,5
11110		?	?	?	?	?	?

A baseline method:
learning a model for each target independently

		Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101		1,3	0,2	1,4	1,7	3,5	1,3
00111		2	1,7	1,5	7,5	8,2	7,6
01110		0,2	0	0,3	0,4	1,2	2,2
10001		3,1	1,1	1,3	1,1	1,7	5,2
01011		4,7	2,1	2,5	1,5	2,3	8,5
11110		?	?	?	?	?	?

A baseline: Independent Models

		Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101		1,3	0,2	1,4	1,7	3,5	1,3
00111		2	1,7	1,5	7,5	8,2	7,6
01110		0,2	0	0,3	0,4	1,2	2,2
10001		3,1	1,1	1,3	1,1	1,7	5,2
01011		4,7	2,1	2,5	1,5	2,3	8,5
11110		?	?	?	?	?	?

A baseline: Independent Models

Linear basis function model for i-th target:

$$f_i(\boldsymbol{x}) = \boldsymbol{a}_i^\intercal \phi(\boldsymbol{x})\,,$$

Solving as a joint optimization problem:

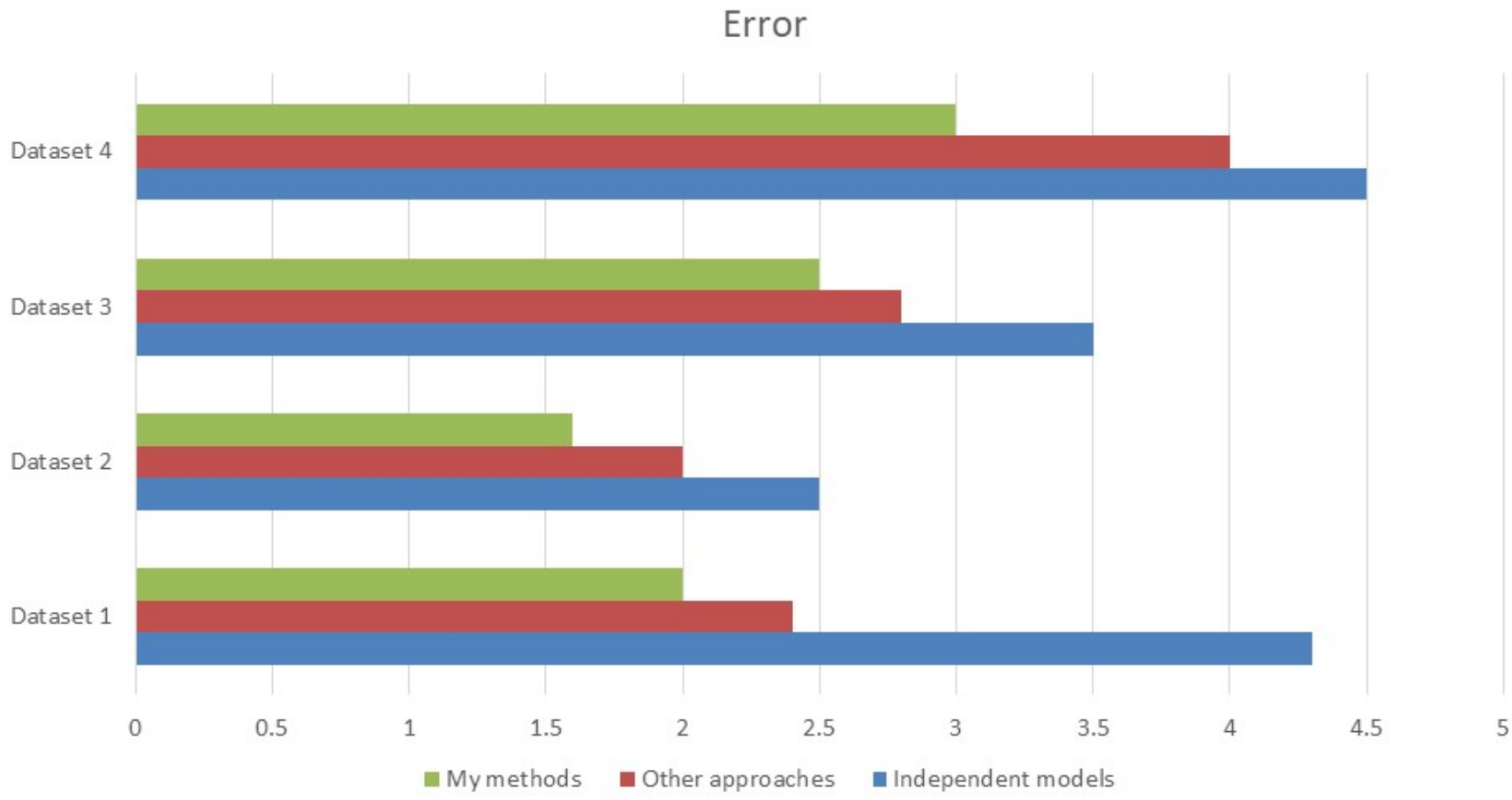
$$\min_A ||Y - XA||_F^2 + \sum_{i=1}^{m} \lambda_i \, ||\boldsymbol{a}_i||^2\,,$$

$$Y : (n \times m) \qquad X : (n \times p) \qquad A : (p \times m)$$

With the following notations:

$$X = \begin{bmatrix} \phi(\boldsymbol{x}_1)^T \\ \vdots \\ \phi(\boldsymbol{x}_n)^T \end{bmatrix} \qquad A = [\boldsymbol{a}_1 \quad \cdots \quad \boldsymbol{a}_m]\,.$$

The results section of a typical MTP paper...



Independent models a.k.a. binary relevance, models that do not exploit target dependencies, one-versus-all, etc.

Learning a model for each target independently is still state-of-the-art in extreme multi-label classification[5]:

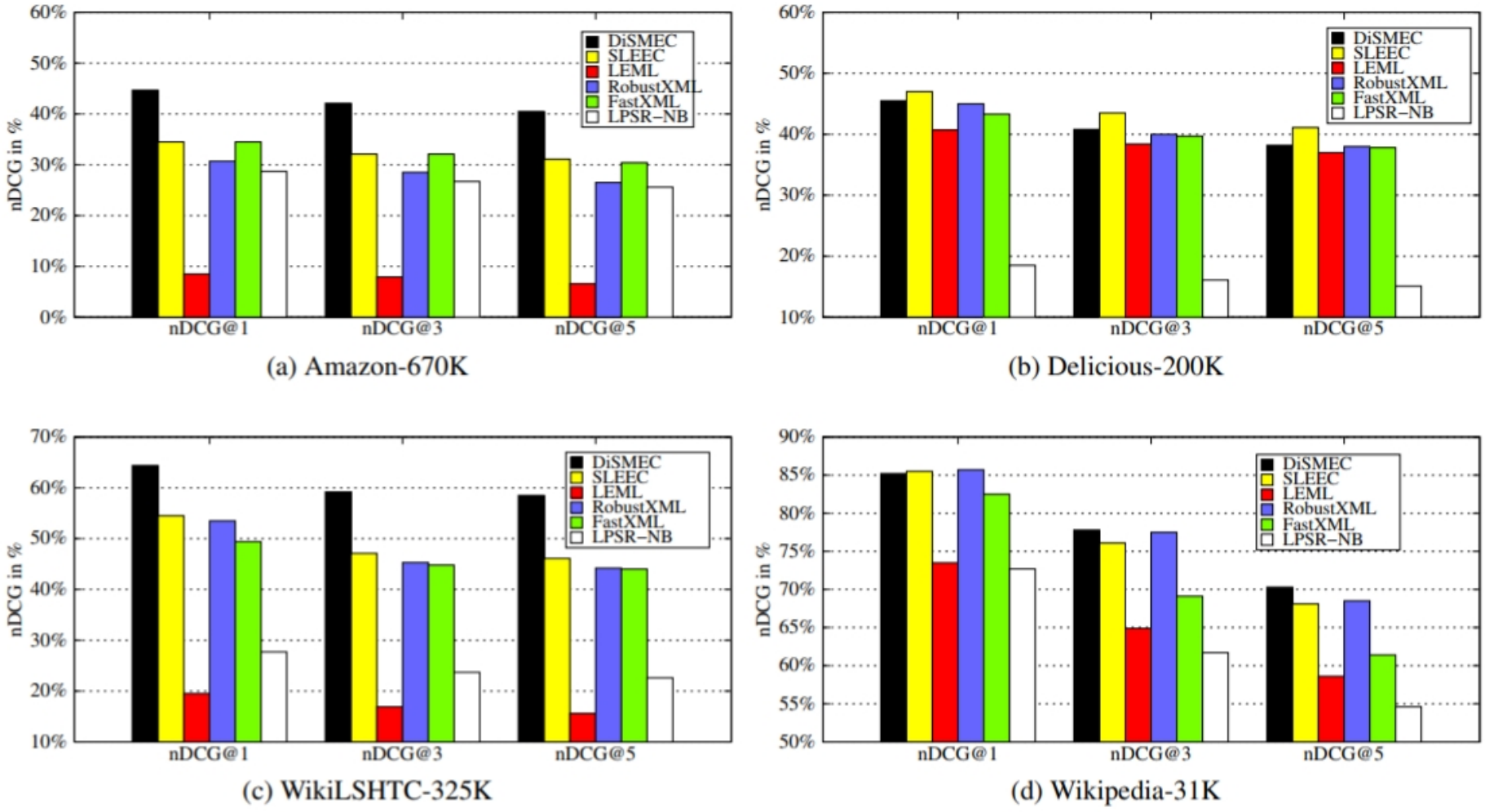


Figure 3: nDCG@k for k=1, 3 and 5

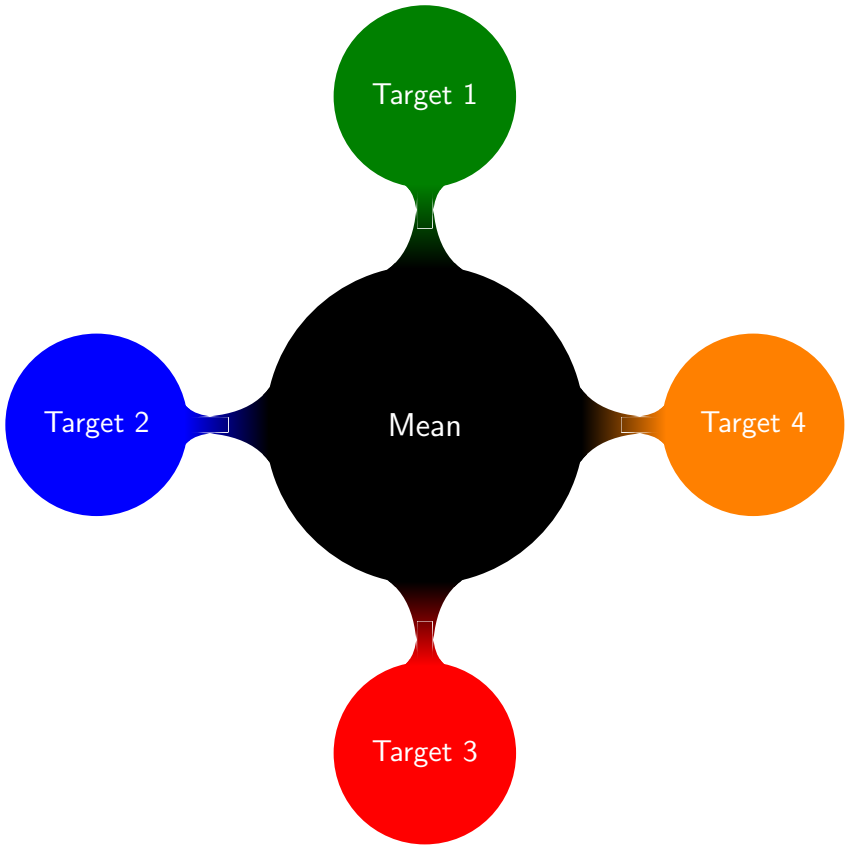
[5] Babbar and Schölkopf, DISMEC: Distributed Sparse Machines for Extreme Multi-label classification, WSDM 2017

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B
Similarity-enforcing methods	B
Relation-exploiting methods	B and D
Relation-constructing methods	B
Representation-exploiting methods	B and D
Representation-constructing methods	A and B

Mean-regularized multi-task learning[6]

- **Simple assumption**: models for different targets are related to each other.
- **Simple solution**: the parameters of these models should have similar values.
- **Approach**: bias the parameter vectors towards their mean vector.



$$\min_A ||Y - XA||_F^2 + \lambda \sum_{i=1}^{m} ||\boldsymbol{a}_i - \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{a}_j||^2 ,$$

[6] Evgeniou and Pontil, Regularized multi–task learning, KDD 2004.

Joint feature selection

- Enforce that the same features are selected for different targets[7]:

$$\min_{A} ||Y - XA||_F^2 + \lambda \sum_{j=1}^{p} ||\boldsymbol{a}_j||^2$$

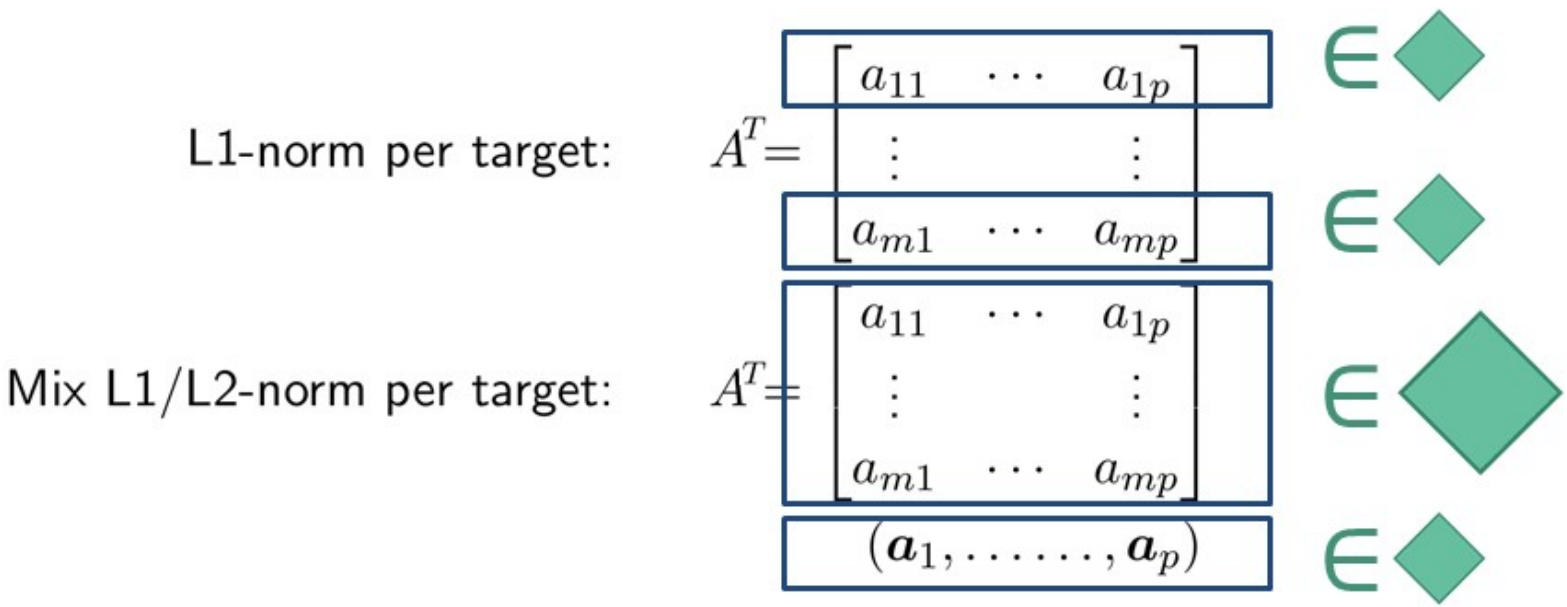
[7] Obozinski et al. Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing 2010

Joint feature selection

- Enforce that the same features are selected for different targets[7]:

$$\min_A ||Y - XA||_F^2 + \lambda \sum_{j=1}^{p} ||\boldsymbol{a}_j||^2$$

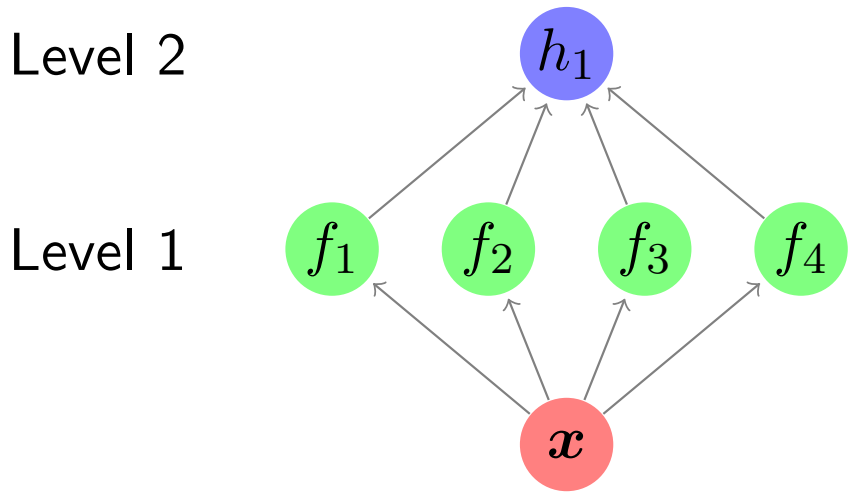
- The vectors $\boldsymbol{a}_j$ now represent the columns of matrix A^T:



[7] Obozinski et al. Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing 2010

Stacking (Stacked generalization)

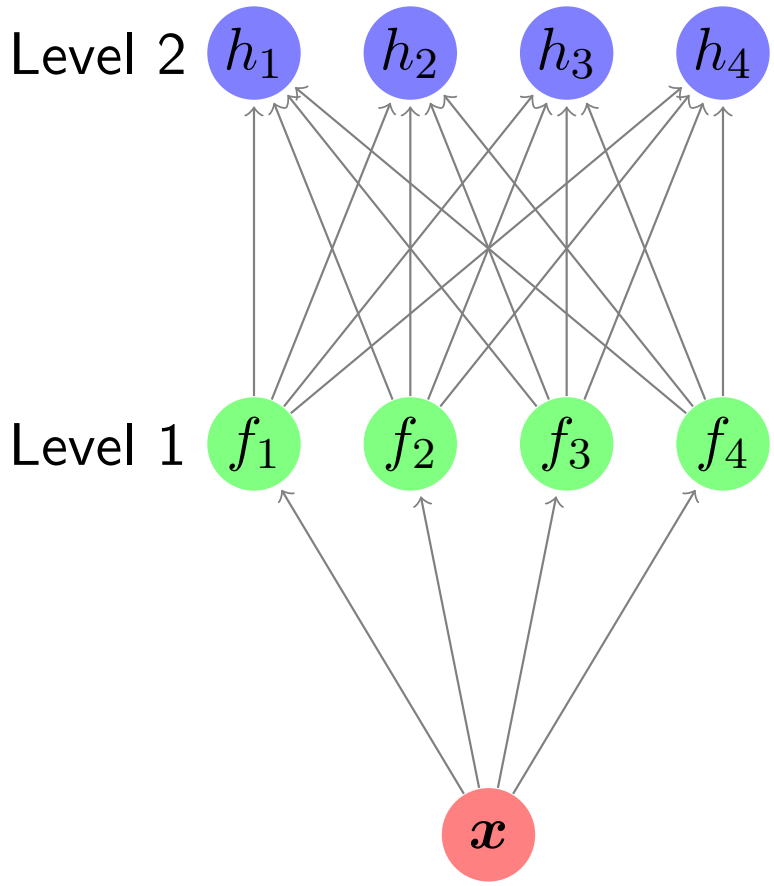
- Originally introduced as a general ensemble learning or blending technique.[8]
- Level 1 classifiers: apply a series of ML methods on the same dataset (or, one ML method on bootstrap samples of the dataset)
- Level 2 classifier: apply an ML method to a new dataset consisting of the predictions obtaining at Level 1



[8] Wolpert, Stacked generalization. Neural Networks 1992

Stacking applied to multi-target prediction[9]

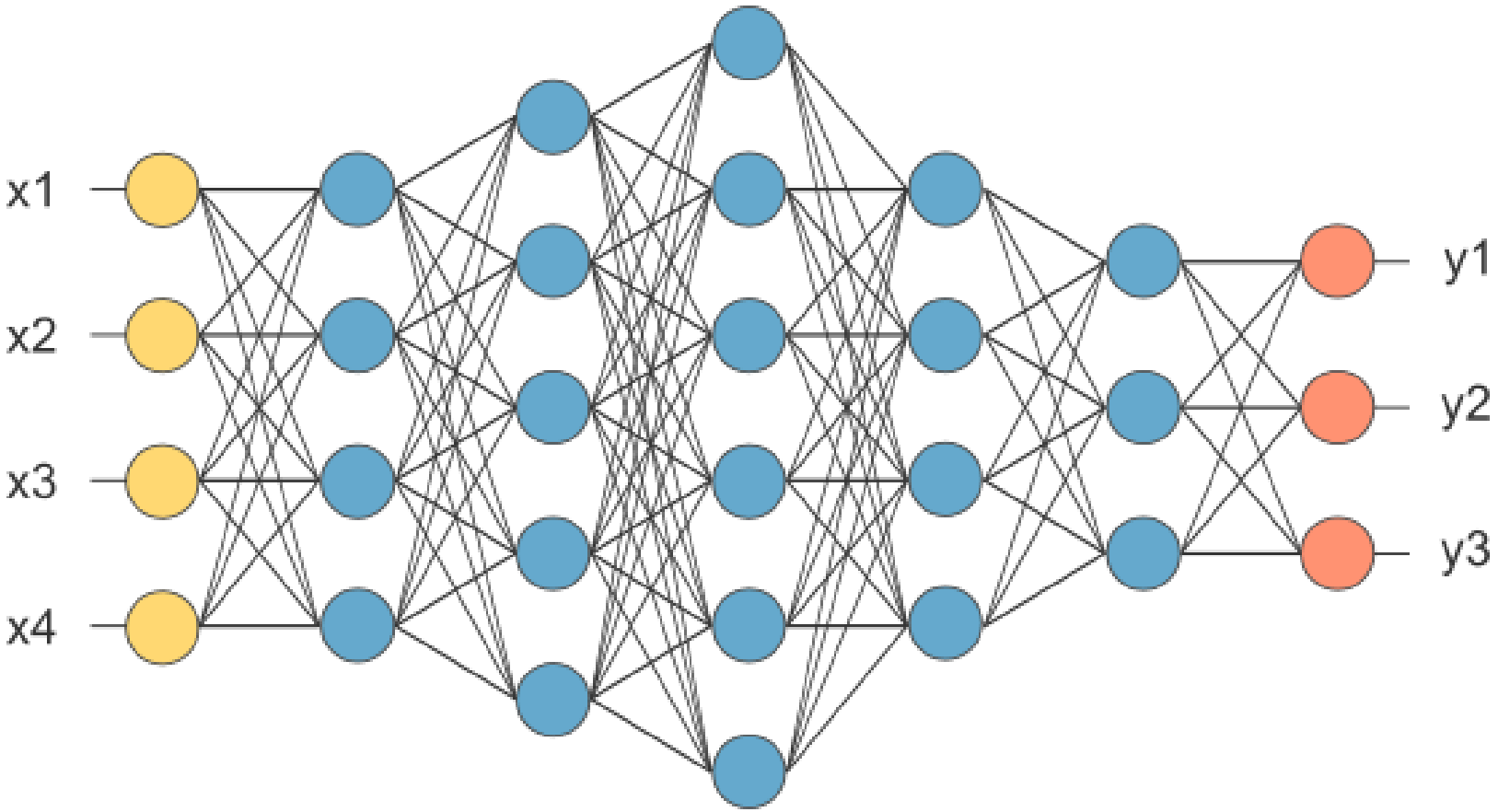
- Level 1 classifiers: learn a model for every target independently
- Level 2 classifier: learn again a model for every target independently, using the predictions of the first step as features



[9] Cheng and Hüllermeier, Combining Instance-based learning and Logistic Regreession for Multi-Label classification, Machine Learning, 2009

Enforcing similarity in (Deep) Neural Networks

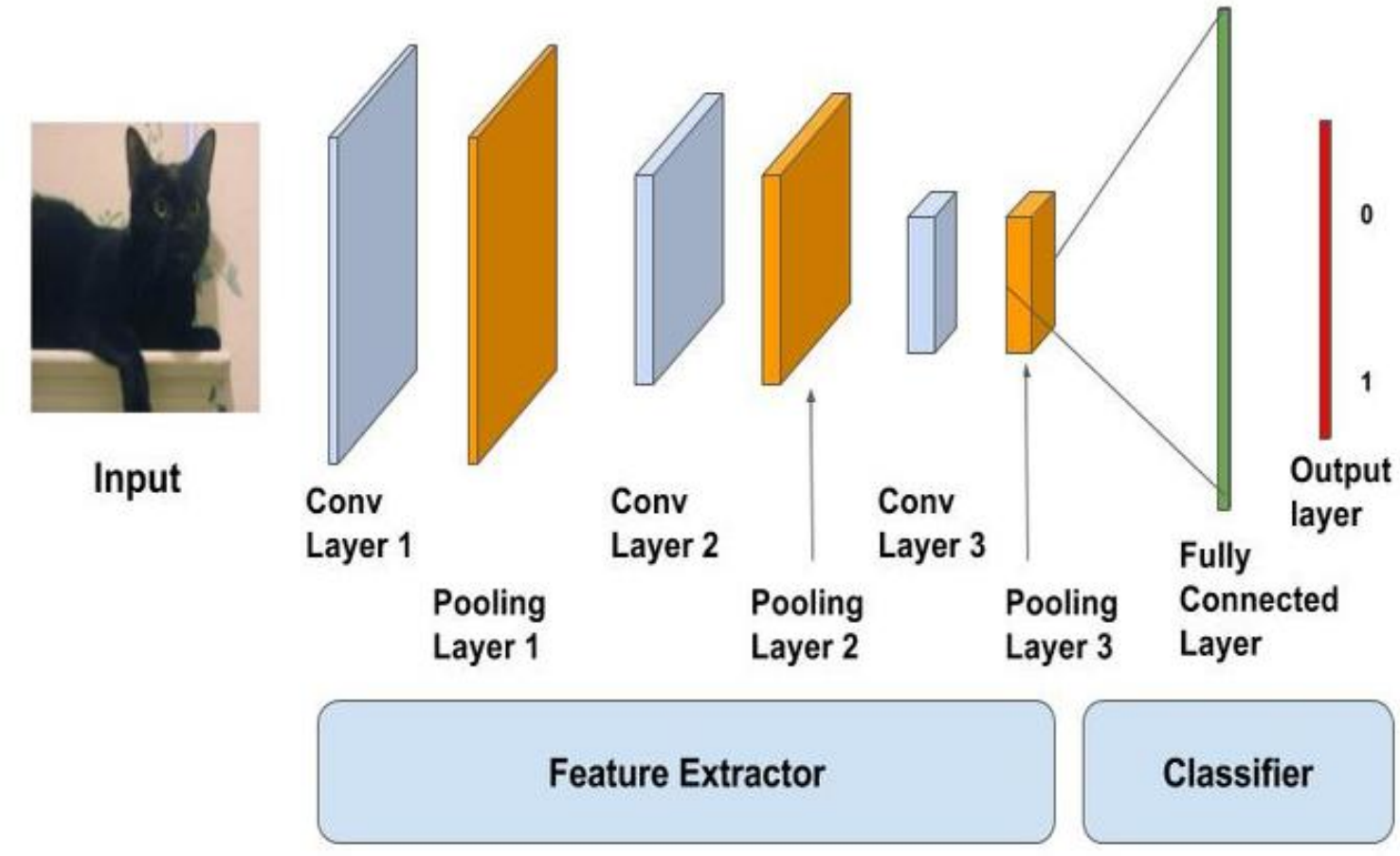
Commonly-used architecture: weight sharing among targets[10]



[10] Caruana, Multitask learning: A knowledge-based source of inductive bias. Machine Learning 1997

Re-using Pretrained Models in (Deep) Neural Networks

Commonly-used training method: first train on targets that have a lot of observations, only train some parameters for targets that have few observations [11]



[11] Keras Tutorial: Transfer Learning using pre-trained models

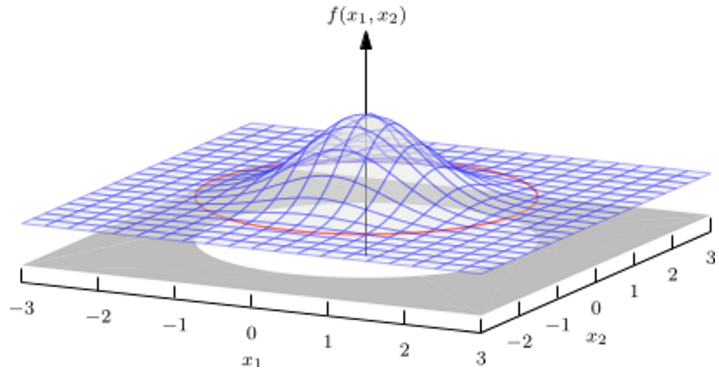
Question

In which situations are similarity-enforcing models capable of outperforming independent models w.r.t. predictive performance?

- Always
- When p is sufficiently large
- When m is sufficiently large
- When the targets are sufficiently correlated

An intuitive explanation: James-Stein estimation

- Consider a sample of a multivariate normal distribution $\boldsymbol{y} \sim N(\boldsymbol{\theta}, \sigma^2 \mathbf{I})$.

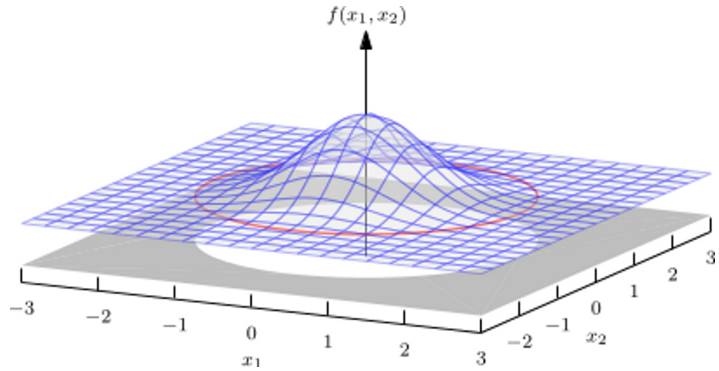


- What is the best estimator of the mean vector $\boldsymbol{\theta}$?
- Evaluation w.r.t. MSE: $\mathbb{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^2]$

[12] W. James and C. Stein. Estimation with quadratic loss. In Proc. Fourth Berkeley Symp. Math. Statist. Prob. 1, pages 361-379, 1961

An intuitive explanation: James-Stein estimation

- Consider a sample of a multivariate normal distribution $\boldsymbol{y} \sim N(\boldsymbol{\theta}, \sigma^2 \mathbf{I})$.

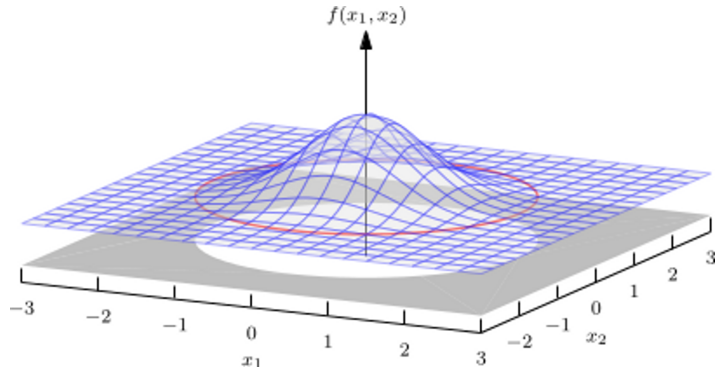


- What is the best estimator of the mean vector $\boldsymbol{\theta}$?
- Evaluation w.r.t. MSE: $\mathbb{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^2]$
- Single-observation maximum likelihood estimator: $\hat{\boldsymbol{\theta}}^{\mathrm{ML}} = \boldsymbol{y}$

[12] W. James and C. Stein. Estimation with quadratic loss. In Proc. Fourth Berkeley Symp. Math. Statist. Prob. 1, pages 361-379, 1961

An intuitive explanation: James-Stein estimation

- Consider a sample of a multivariate normal distribution $\boldsymbol{y} \sim N(\boldsymbol{\theta}, \sigma^2 \mathbf{I})$.

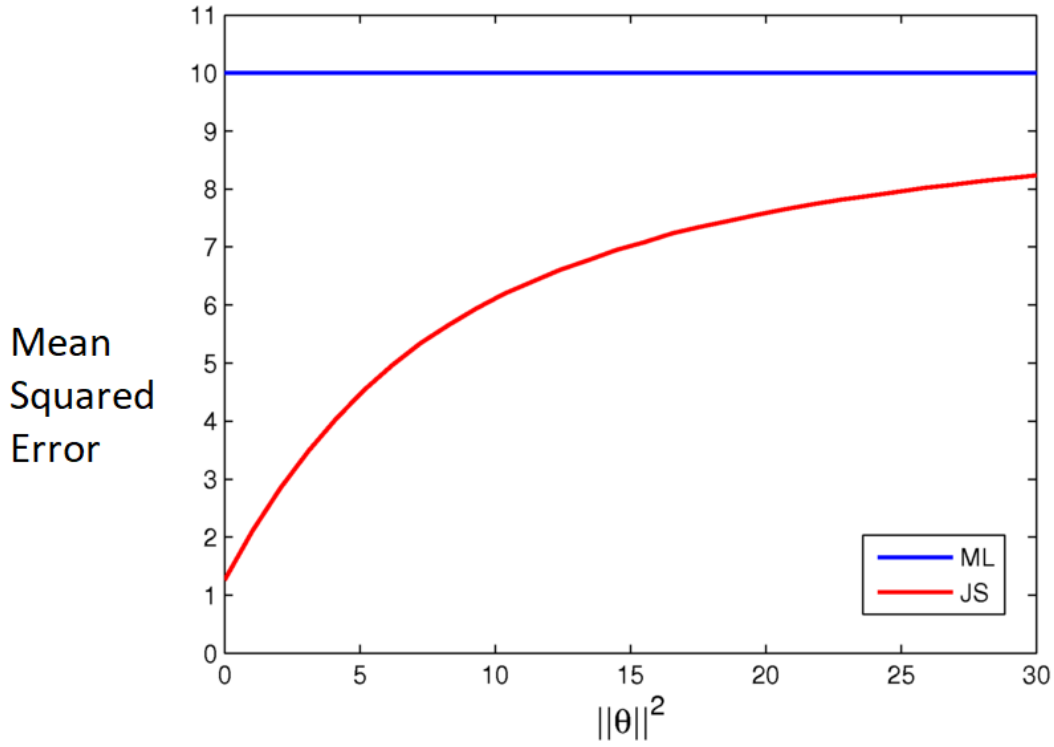


- What is the best estimator of the mean vector $\boldsymbol{\theta}$?
- Evaluation w.r.t. MSE: $\mathbb{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^2]$
- Single-observation maximum likelihood estimator: $\hat{\boldsymbol{\theta}}^{\mathrm{ML}} = \boldsymbol{y}$
- James-Stein estimator[12]:

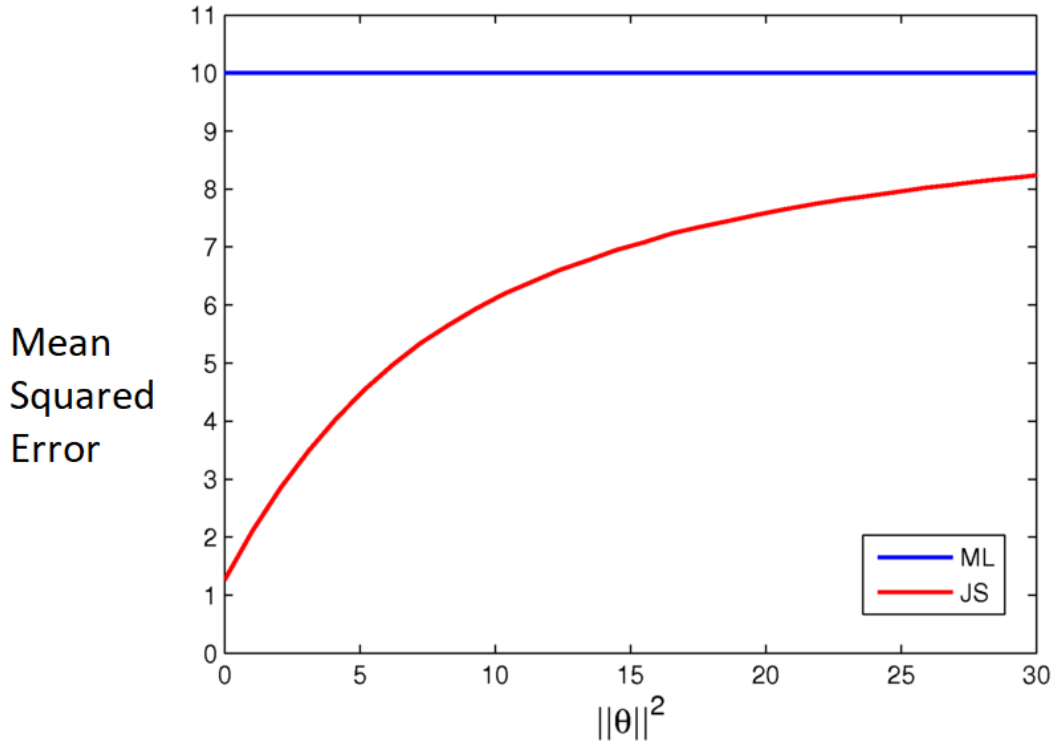
$$\hat{\theta}^{\mathrm{JS}} = \left(1 - \frac{(m-2)\sigma^2}{\|\boldsymbol{y}\|^2}\right) \boldsymbol{y}$$

[12] W. James and C. Stein. Estimation with quadratic loss. In Proc. Fourth Berkeley Symp. Math. Statist. Prob. 1, pages 361-379, 1961

- Works best when the norm of the mean vector is close to zero:



- Works best when the norm of the mean vector is close to zero:



- Regularization towards other directions is also possible:

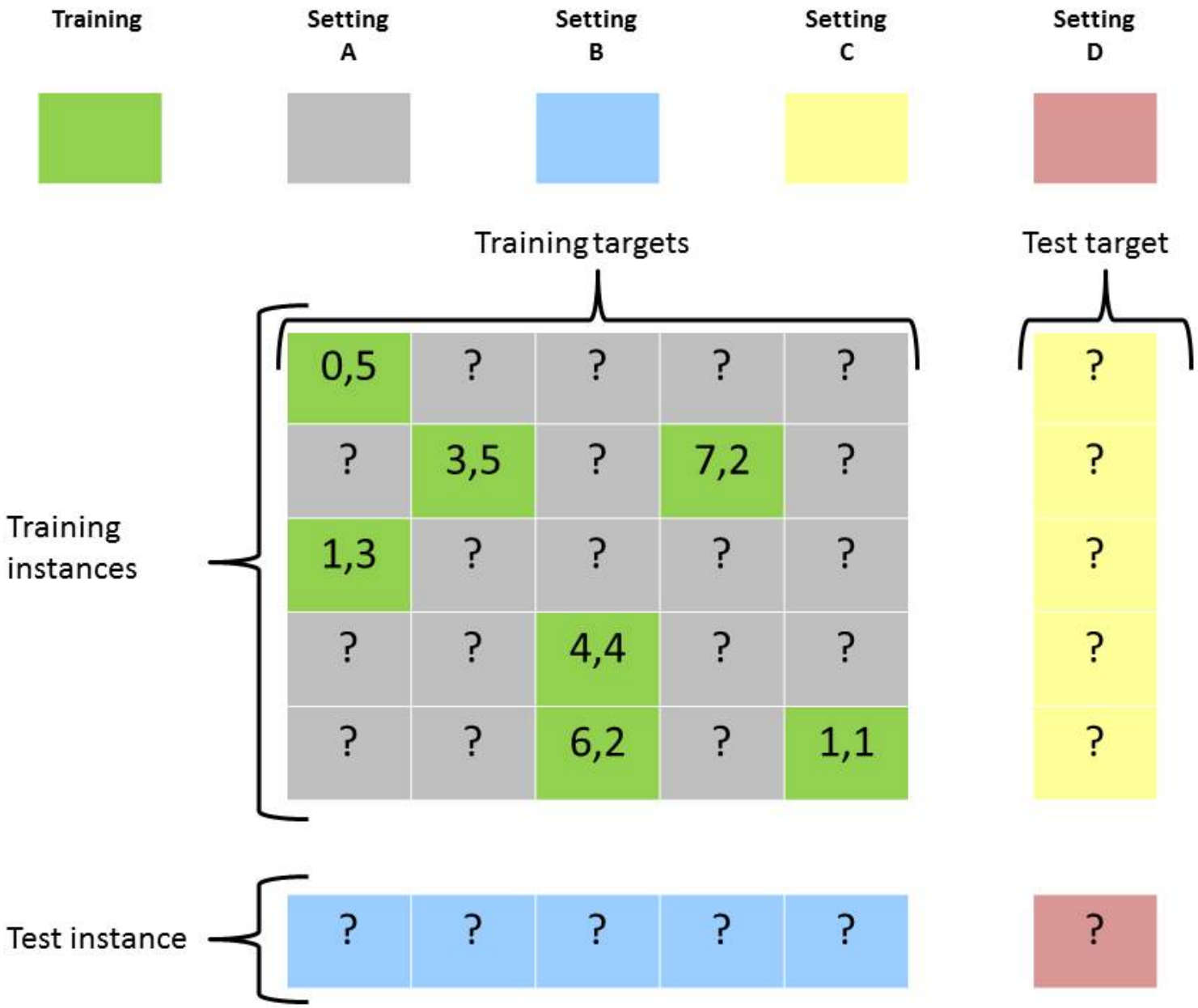
$$\hat{\theta}^{\mathrm{JS}+} = \left(1 - \frac{(m-2)\sigma^2}{\|\boldsymbol{y} - \boldsymbol{v}\|^2}\right)(\boldsymbol{y} - \boldsymbol{v}) + \boldsymbol{v}$$

- Only outperforms the maximum likelihood estimator w.r.t. the sum of squared errors over all components, **and only when** $m \geq 3$

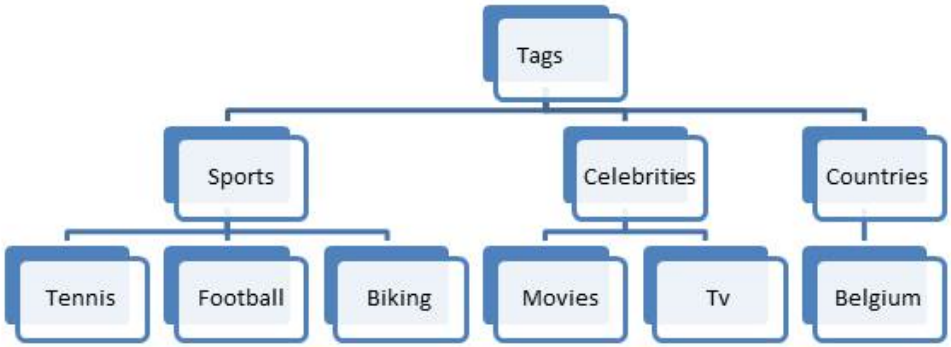
A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B
Similarity-enforcing methods	B
Relation-exploiting methods	B and D
Relation-constructing methods	B
Representation-exploiting methods	B and D
Representation-constructing methods	A and B

Different learning settings revisited



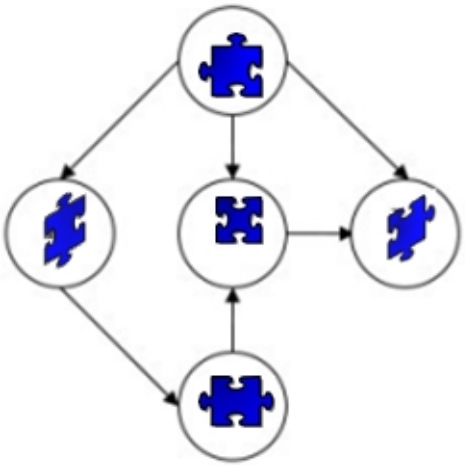
An example from the introduction revisited



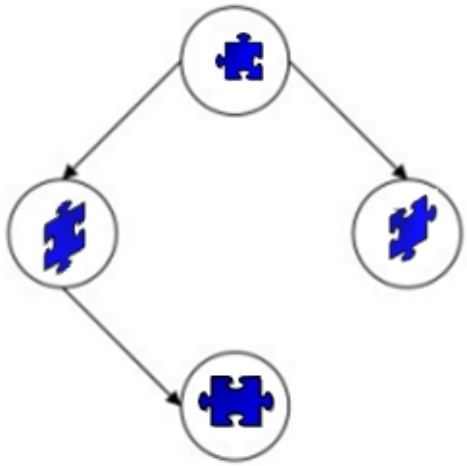
		Tennis	Football	Biking	Movies	Tv	Belgium
01101	**Text1**	0	0	0	0	0	1
00111	**Text2**	0	0	1	0	1	1
01110	**Text3**	0	0	0	1	1	0
10001	**Text4**	0	0	1	0	1	0
01011	**Text5**	1	0	0	1	0	0
11110	**Text6**	?	?	?	?	?	?

Exploiting relations in regularization terms

Graph



Tree



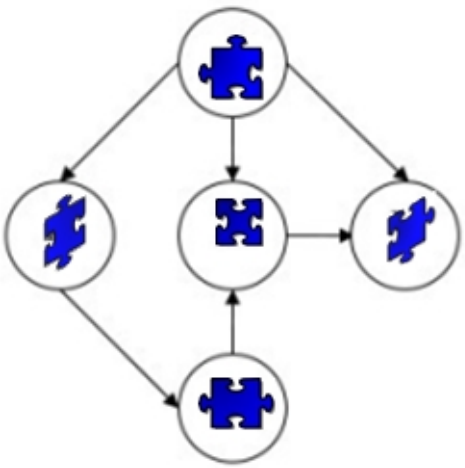
Similarity

	1	0.26	0.26	0.04
	0.26	1	0.7	0.57
	0.26	0.7	1	0.44
	0.04	0.57	0.44	1

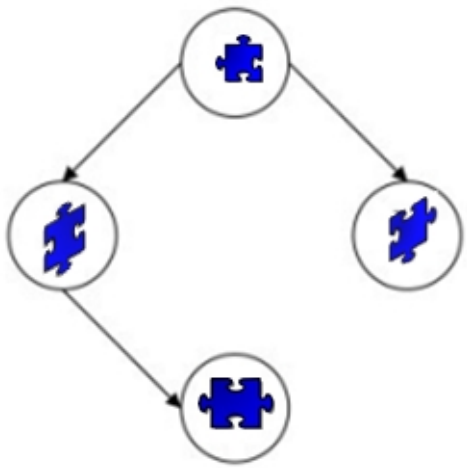
[13] Gopal and Yang, Recursive regularization for large-scale classification with hierarchical and graphical dependencies, KDD 2013

Exploiting relations in regularization terms

Graph



Tree



Similarity

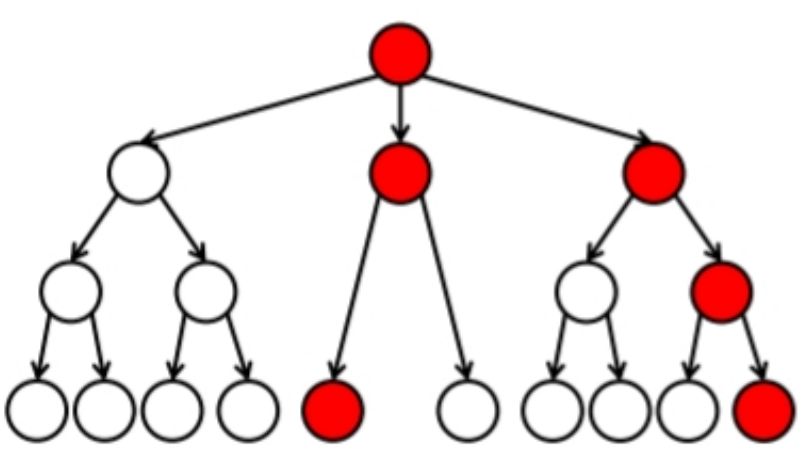
	1	0.26	0.26	0.04
	0.26	1	0.7	0.57
	0.26	0.7	1	0.44
	0.04	0.57	0.44	1

Graph-based regularization is an approach that can be applied to the three types of relations[13]:

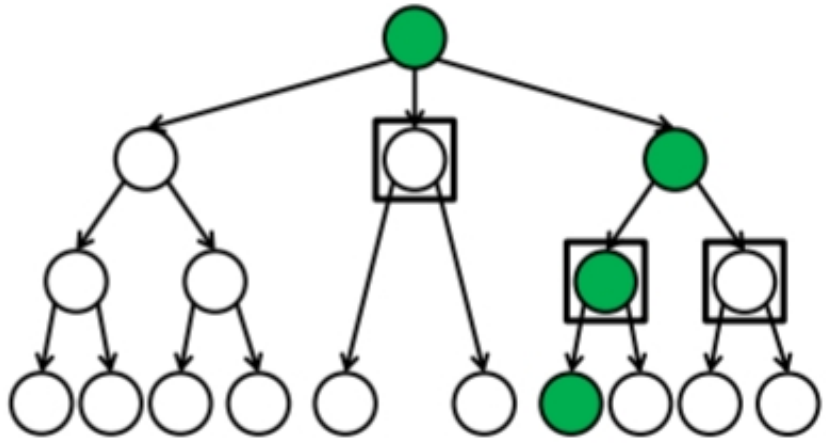
$$\min_A ||Y - XA||_F^2 + \lambda \sum_{i=1}^{m} \sum_{j \in \mathcal{N}(i)} ||\boldsymbol{a}_i - \boldsymbol{a}_j||^2$$

[13] Gopal and Yang, Recursive regularization for large-scale classification with hierarchical and graphical dependencies, KDD 2013

Hierarchical multi-label classfication



(a) Ground truth.



(b) Prediction A.

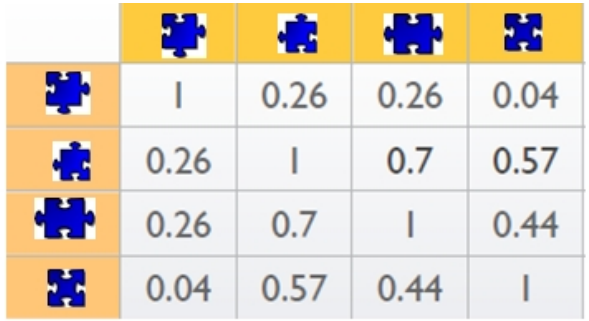
In addition to performance gains in general, hierarchies can also be used to define specific loss functions, such as the H-loss[14]:

$$\ell_{Hier}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{j:y_j \neq \hat{y}_j} c_j \left[\!\left[anc(y_j) = anc(\hat{y}_j) \right]\!\right]$$

c_i depends on the depth of node i

[14] Bi and Kwok, Bayes-optimal hierarchical multi-label classification, IEEE Transactions on Knowledge and Data Engineering, 2014

Exploiting similarity measures among targets



	1	0.26	0.26	0.04
	0.26	1	0.7	0.57
	0.26	0.7	1	0.44
	0.04	0.57	0.44	1

Can be done within the framework of vector-valued kernel functions[15]:

$$f(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{w}^T \Psi(\boldsymbol{x}, \boldsymbol{t}) = \sum_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}) \in \mathcal{D}} \alpha_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})} \Gamma((\boldsymbol{x}, \boldsymbol{t}), (\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}))$$

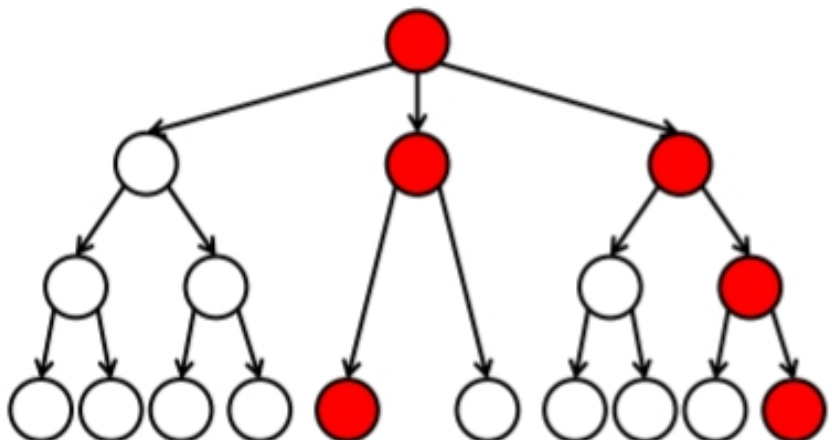
Model the joint kernel as a product of an instance kernel $k(\cdot, \cdot)$ and a target kernel $g(\cdot, \cdot)$:

$$\Gamma((\boldsymbol{x}, \boldsymbol{t}), (\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})) = k(\boldsymbol{x}, \bar{\boldsymbol{x}}) \cdot g(\boldsymbol{t}, \bar{\boldsymbol{t}})$$

[15] Alvarez et al., Kernels for vector-valued functions: a review, Foundation and Trends in Machine Learning, 2012

Converting graphs to similarities or target representations

- **Similarities:** use graph structure to express target similarities
  e.g. the shortest-path kernel between two nodes
- **Representations:** often characteristics of a specific vertex or edge
  e.g. the number of positive labels that are siblings of a vertex[16]

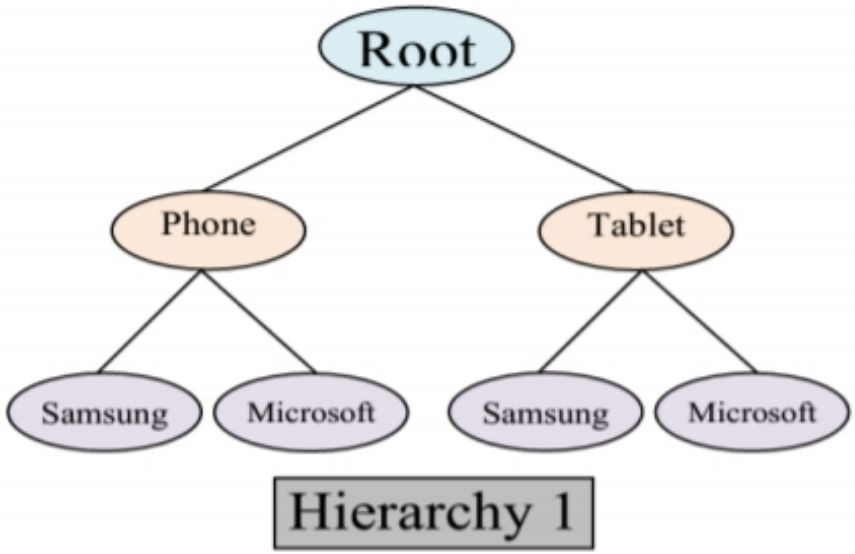


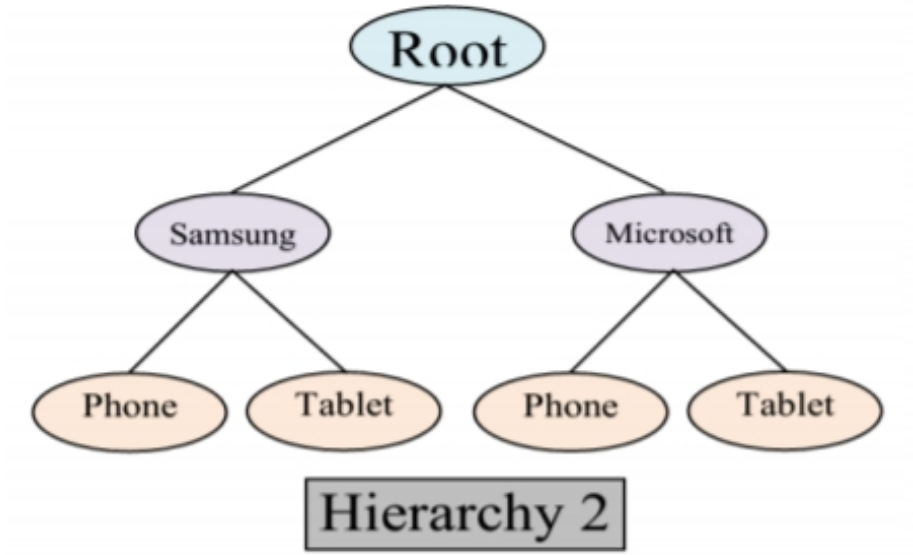
[16] Rousu et al., Kernel-based learning of hierarchical multilabel classification models, JMLR 2006

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B
Similarity-enforcing methods	B
Relation-exploiting methods	B and D
Relation-constructing methods	B
Representation-exploiting methods	B and D
Representation-constructing methods	A and B

Constructing target hierarchies

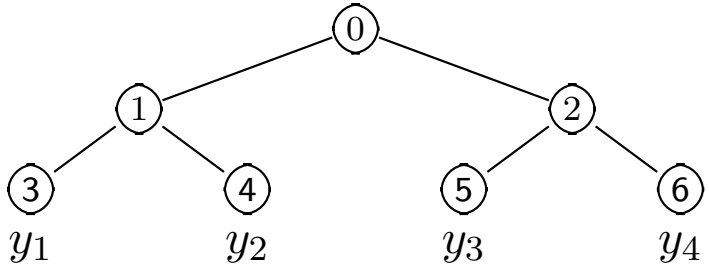




- It might be difficult for a human expert to define a hierarchy[17]
- Perhaps one can try to learn the hierarchy from data?
- Algorithms: level flattening, node removal, hierarchy modification, hierarchy generation, etc.

[17] Rangwala and Naik, Tutorial on Large-Scale Hierarchical Classification, KDD 2017.

Label trees ($\neq$ decision trees)



- Organize classifiers in a tree structure (one leaf $\Leftrightarrow$ one label)
- Mainly used in multi-class and multi-label classification
- Goal is fast prediction: almost logarithmic in the number of labels
- Algorithms: Label embedding trees[18], Nested dichotomies[19], Conditional probability trees[20], Hierarchical softmax[21], FastText[22], Probabilistic classifier chains[23]

---

[18] Bengio et al., Label embedding trees for large multi-class tasks, NIPS 2010

[19] Frank and Kramer, Ensembles of nested dichotomies for multi-class problems, ICML 2004

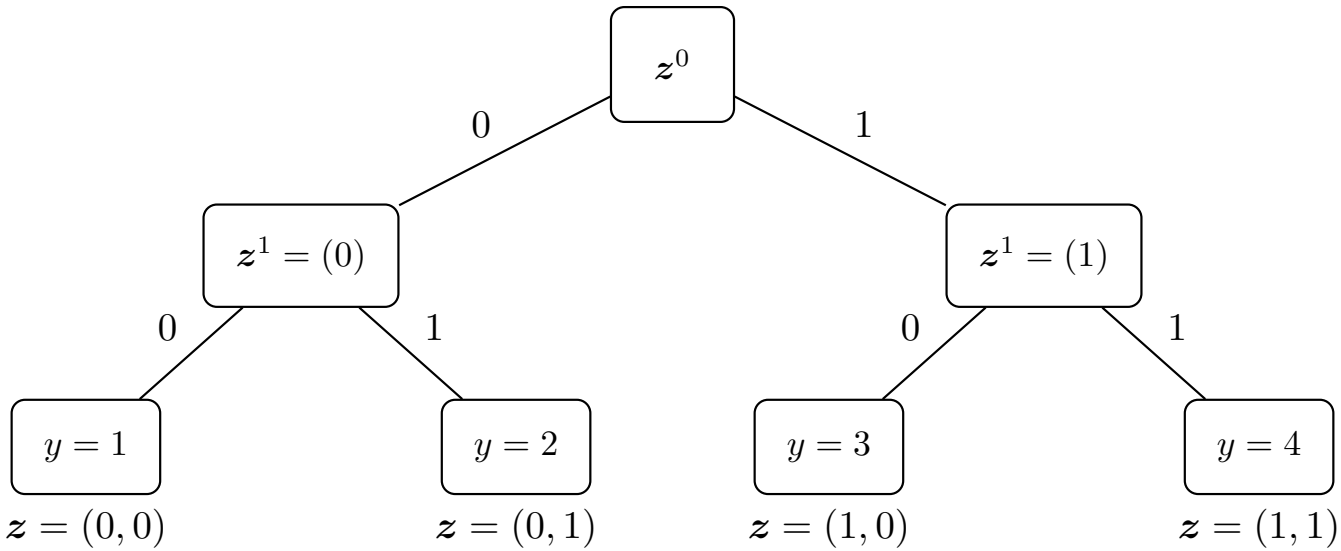
[20] Beygelzimer et al., Conditional probability tree estimation analysis and algorithms. UAI 2009

[21] Morin and Bengio, Hierarchical probabilistic neural network language model, AISTATS 2005

[22] Joulin et al., Bag of tricks for efficient text classifcation. CoRR, abs/1607.01759, 2016

[23] Dembczynski et al., Bayes optimal multilabel classification via probabilistic classifier chains, ICML 2010

Hierarchical softmax / Probabilistic classifier trees



- Encode the targets by a **prefix code** ($\Rightarrow$ tree structure)[24]
- Multi-class classification: each label y **coded** by $\boldsymbol{z} = (z_1, \ldots, z_l) \in \mathcal{C}$
- Multi-label classification: a label vector $\boldsymbol{y} = (y_1, \ldots, y_m)$ is a prefix code.

[24] Dembczynski et al., Consistency of probabilistic classifier trees. ECMLPKDD 2016

Probabilistic classifier chains

- Estimate the joint conditional distribution $P(\boldsymbol{Y} \,|\, \boldsymbol{x})$.
- For optimizing the subset 0/1 loss:

$$\ell_{0/1}(\boldsymbol{y}, \hat{y}) = [\![\boldsymbol{y} \neq \hat{y}]\!]$$

- Repeatedly apply the **product rule of probability**:

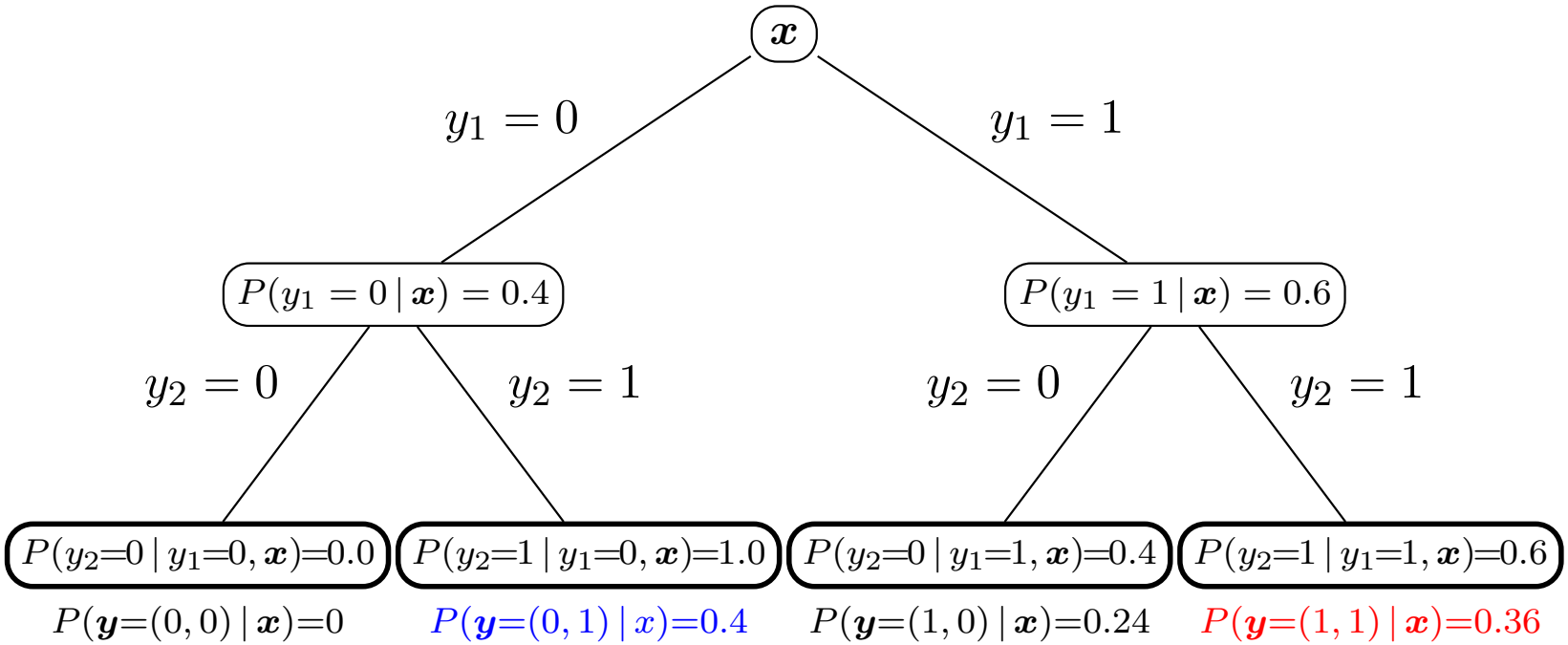
$$P(\boldsymbol{Y} = \boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{i=1}^{m} P(Y_i = y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1}) \,.$$

- Learning relies on constructing probabilistic classifiers for estimating

$$P(Y_i = y_i | \boldsymbol{x}, y_1, \ldots, y_{i-1}) \,,$$

  independently for each $i = 1, \ldots, m$.

- Inference relies on exploiting a probability tree:



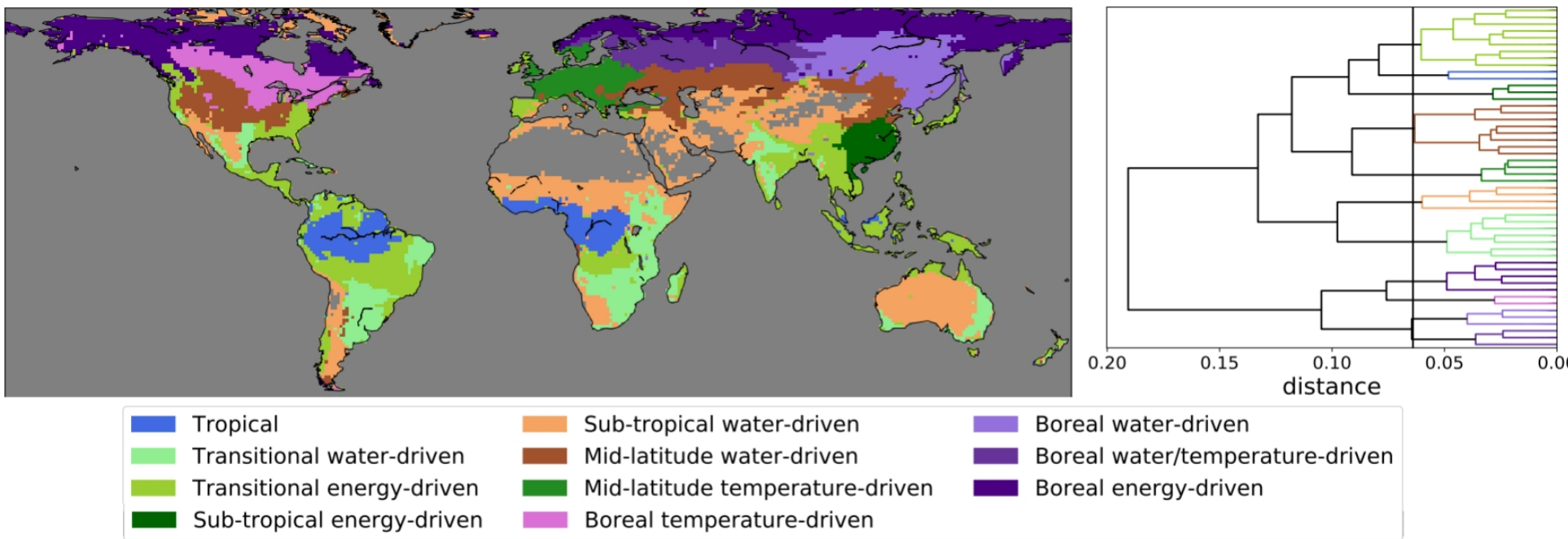
- For subset 0/1 loss one needs to find $\boldsymbol{h}(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} P(\boldsymbol{y} \mid \boldsymbol{x})$.
- Greedy and approximate search techniques with guarantees exist.[25]
- Other losses: compute the prediction on a sample from $P(\boldsymbol{Y} \mid \boldsymbol{x})$.[26]

[25] Kumar et al., Beam search algorithms for multilabel learning, Machine Learning 2013

[26] 23 Dembczynski et al., An analysis of chaining in multi-label classification, ECAI 2012

Constructing hierarchies to obtain additional insight

- Application in climate science
- Result of learning 20000 tasks simultaneously with a multi-task learning method
- Followed by hierarchical clustering of the learned weight vectors[27]:



[27] Papagiannopoulou et al. Globral hydro-climatic biomes identified with multi-task learning, Geoscientific Model Development Discussions 2018

Constructing target similarities by output kernel learning

- Consider models $\mathbf{f} : \mathcal{X} \to \mathbb{R}^{\mathbf{m}}$
- Training dataset $\{\mathbf{x_i}, \mathbf{y_i}\}_{\mathbf{i=1}}^{\mathbf{n}}$
- Learnable gram matrix G for output kernel $g(\boldsymbol{t}, \boldsymbol{t}')$
- Learn output kernel and model parameters jointly[28]:

$$\min_{G \in \mathbb{R}^{m \times m}} \left[\min_{\mathbf{f} \in \mathcal{F}} \sum_{i=1}^{n} \frac{||\mathbf{f}(\mathbf{x_i}, \cdot) - \mathbf{y_i}||_{\mathbf{2}}^{\mathbf{2}}}{2\lambda} + \frac{||\mathbf{f}||_{\mathcal{F}}^{\mathbf{2}}}{2} + \frac{||G||_F^2}{2} \right]$$

(a) USPS digits (b) Caltech 101 (c) Caltech 256

[28] Dinuzzo et al., Learning Output Kernels with Block Coordinate Descent, ICML 2011

Constructing decision rules among targets

Table 2 Extended multi-label WEATHER dataset

Outlook	Temperature	Humidity	Windy	Play	Icecream	Tea	Lemonade	Dontplay
Rainy	65	70	Yes	0	0	1	0	1
Rainy	71	91	Yes	0	0	1	0	1
Sunny	85	85	No	0	1	0	1	1
Sunny	80	90	Yes	0	1	0	1	1
Sunny	72	95	No	0	1	0	1	1
Sunny	69	70	No	1	0	0	1	0
Sunny	75	70	Yes	1	0	0	1	0
Overcast	83	86	No	1	0	0	1	0
Overcast	64	65	Yes	1	0	0	1	0
Overcast	72	90	Yes	1	0	0	1	0
Overcast	81	75	No	1	0	0	1	0
Rainy	70	96	No	1	0	0	1	0
Rainy	68	80	No	1	0	0	1	0
Rainy	75	80	No	1	0	0	1	0

[29] Loza-Mencia and Janssen, Learning rules for multi-label classification: a stacking and separate-and-conquer approach

Constructing decision rules among targets

Table 2 Extended multi-label WEATHER dataset

Outlook	Temperature	Humidity	Windy	Play	Icecream	Tea	Lemonade	Dontplay
Rainy	65	70	Yes	0	0	1	0	1
Rainy	71	91	Yes	0	0	1	0	1
Sunny	85	85	No	0	1	0	1	1
Sunny	80	90	Yes	0	1	0	1	1
Sunny	72	95	No	0	1	0	1	1
Sunny	69	70	No	1	0	0	1	0
Sunny	75	70	Yes	1	0	0	1	0
Overcast	83	86	No	1	0	0	1	0
Overcast	64	65	Yes	1	0	0	1	0
Overcast	72	90	Yes	1	0	0	1	0
Overcast	81	75	No	1	0	0	1	0
Rainy	70	96	No	1	0	0	1	0
Rainy	68	80	No	1	0	0	1	0
Rainy	75	80	No	1	0	0	1	0

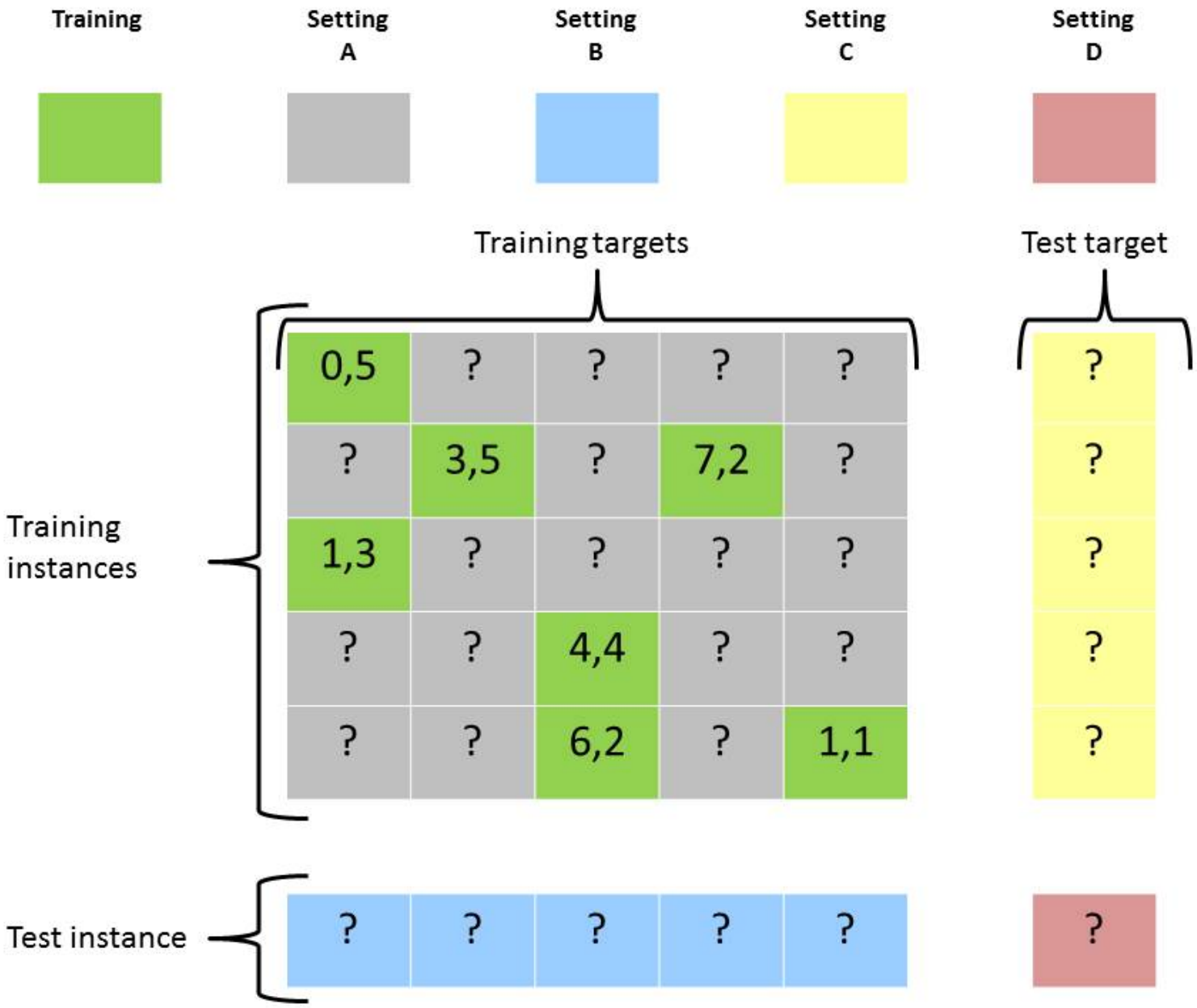
Potential inferred rule[29]: **TEA $\rightarrow$ NOT LEMONADE**

[29] Loza-Mencia and Janssen, Learning rules for multi-label classification: a stacking and separate-and-conquer approach

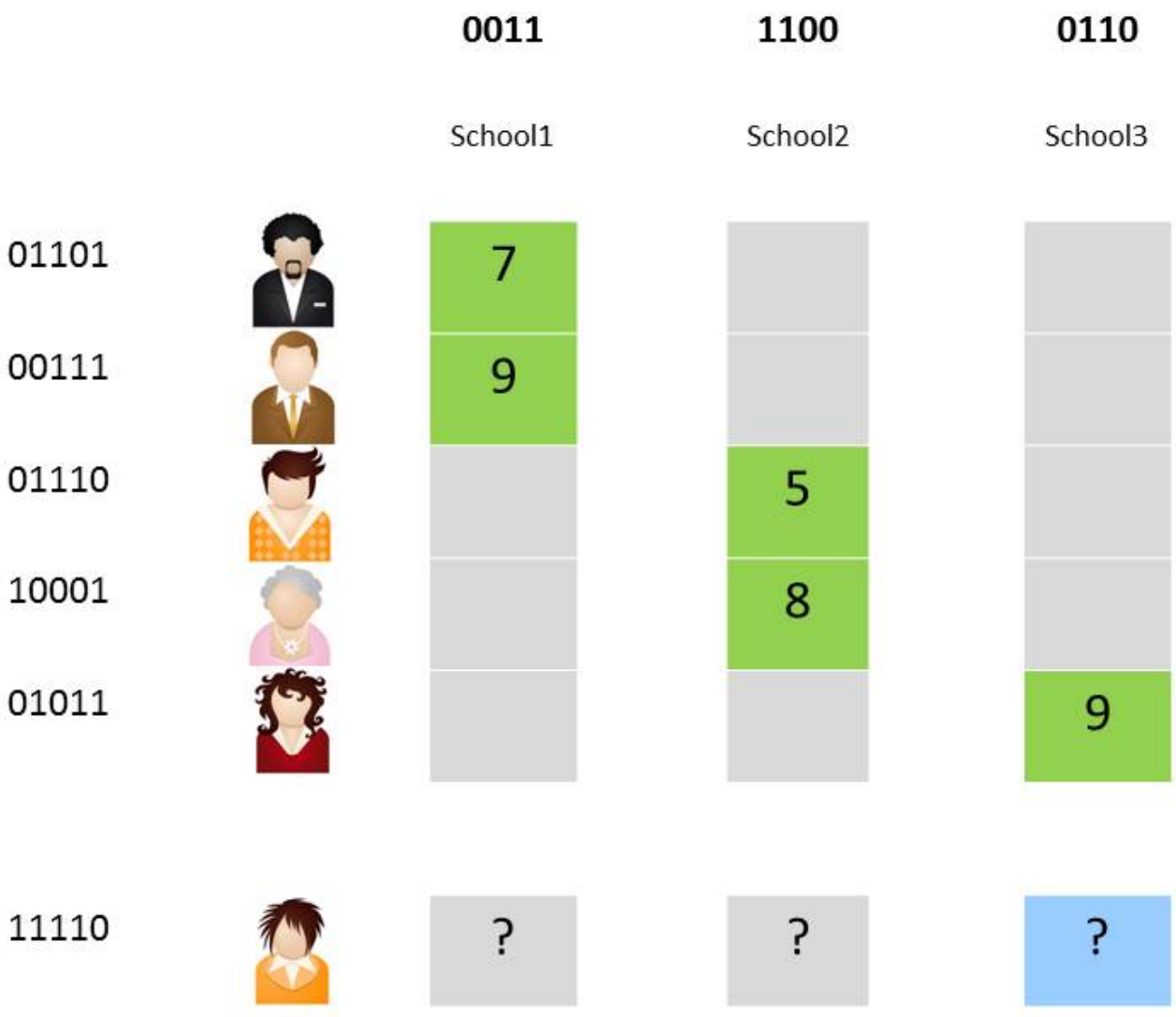
A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B
Similarity-enforcing methods	B
Relation-exploiting methods	B and D
Relation-constructing methods	B
Representation-exploiting methods	B and D
Representation-constructing methods	A and B

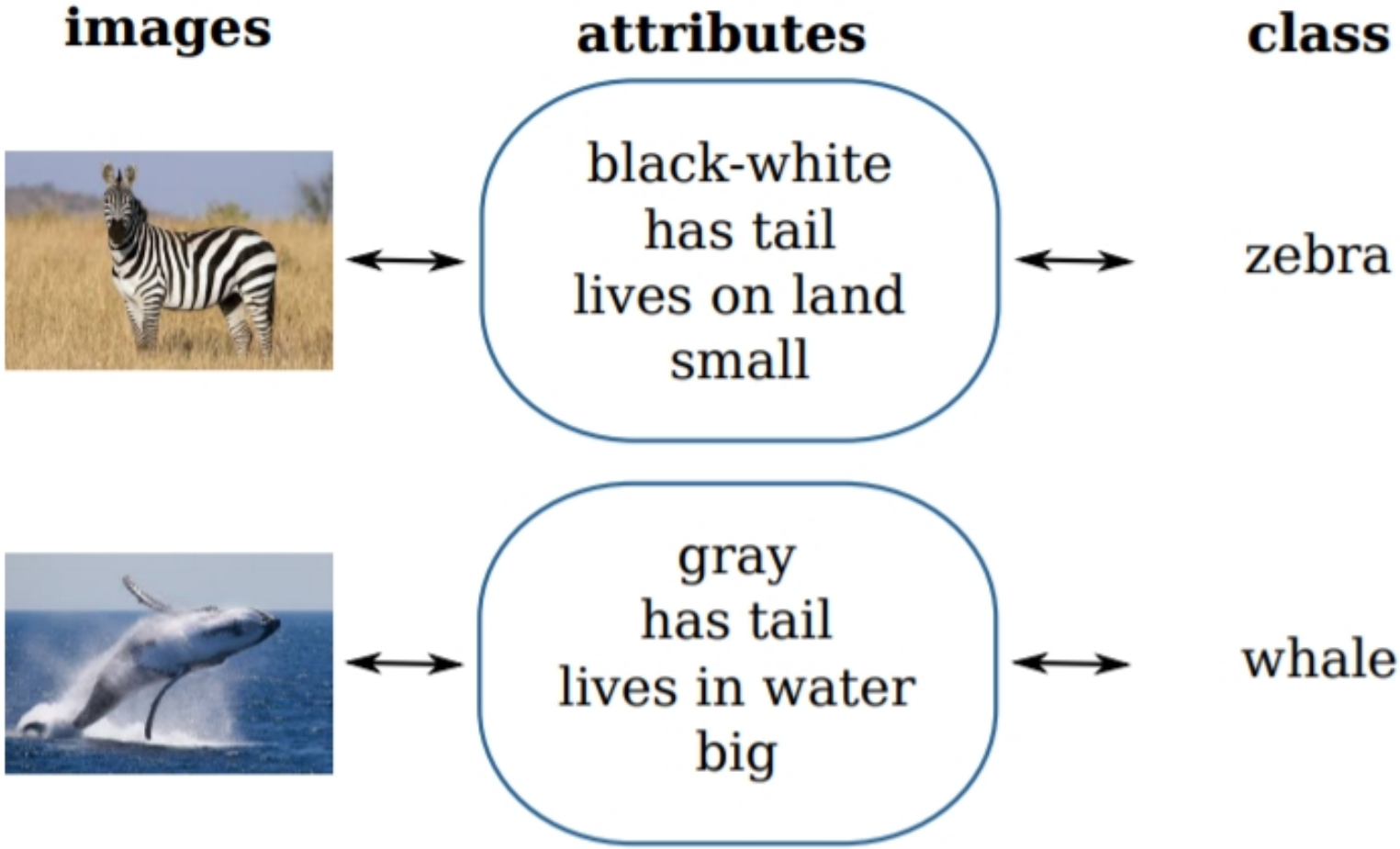
Different learning settings revisited



An example revisited



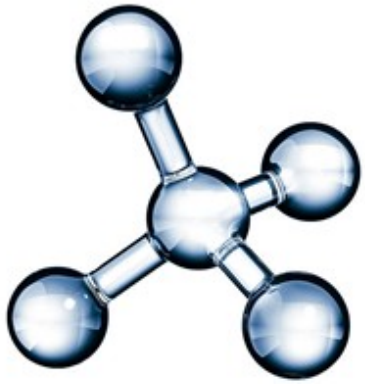
A target representation in computer vision

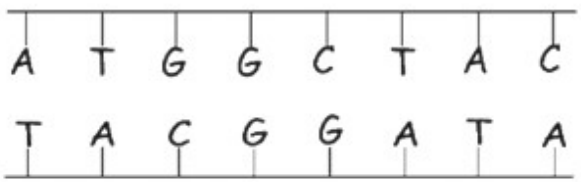


Target representations are the key element of zero-shot learning methods[30]

[30] Examples taken from the CVPR 2016 Tutorial on Zero-shot learning for Computer Vision

Target representations can take many forms

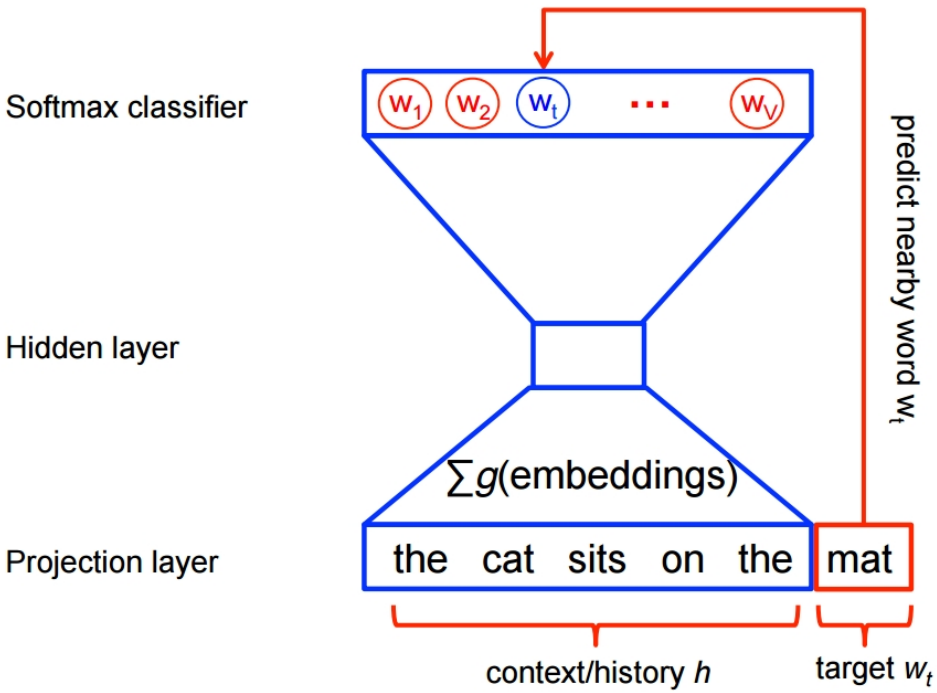


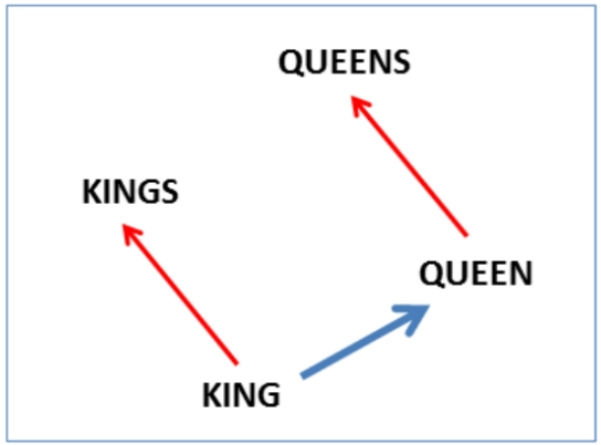


Learning target embeddings from text: Word2Vec

Predict the probability of the next word w_t given the previous words h[31]:

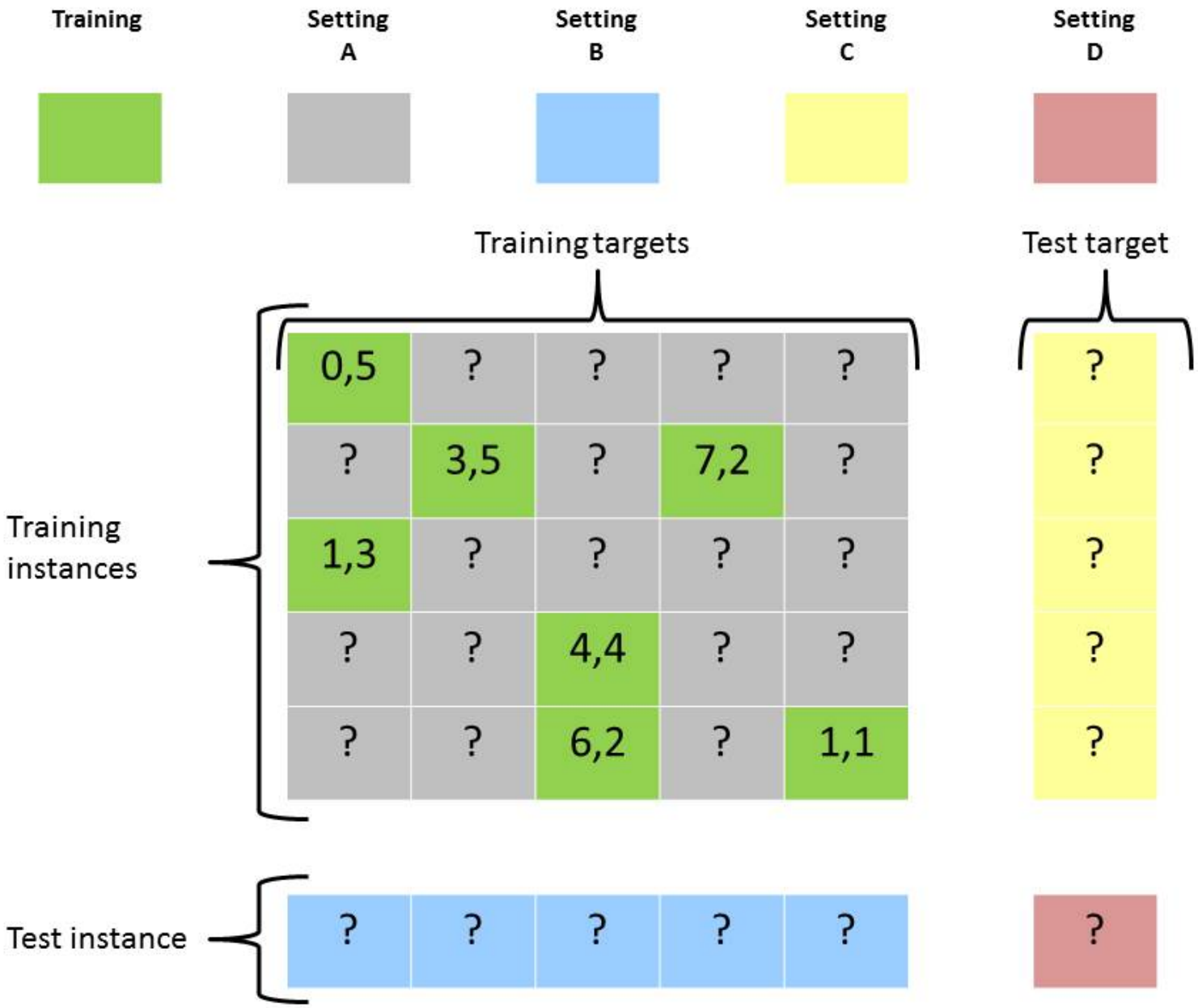
$$P(w_t \mid h) = \frac{\exp(f(w_t, h))}{\sum_{\text{allwords}} \exp(f(w_t, h))}$$





[31] Mikolov et al., Efficient Estimation of Word Representations in Vector Space, Arxiv 2013

Different learning settings revisited



Kronecker kernel ridge regression

Pairwise model representation in the primal:

$$f(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{w}^T \left(\phi(\boldsymbol{x}) \otimes \psi(\boldsymbol{t})\right)$$

Kronecker product pairwise kernel in the dual[32]:

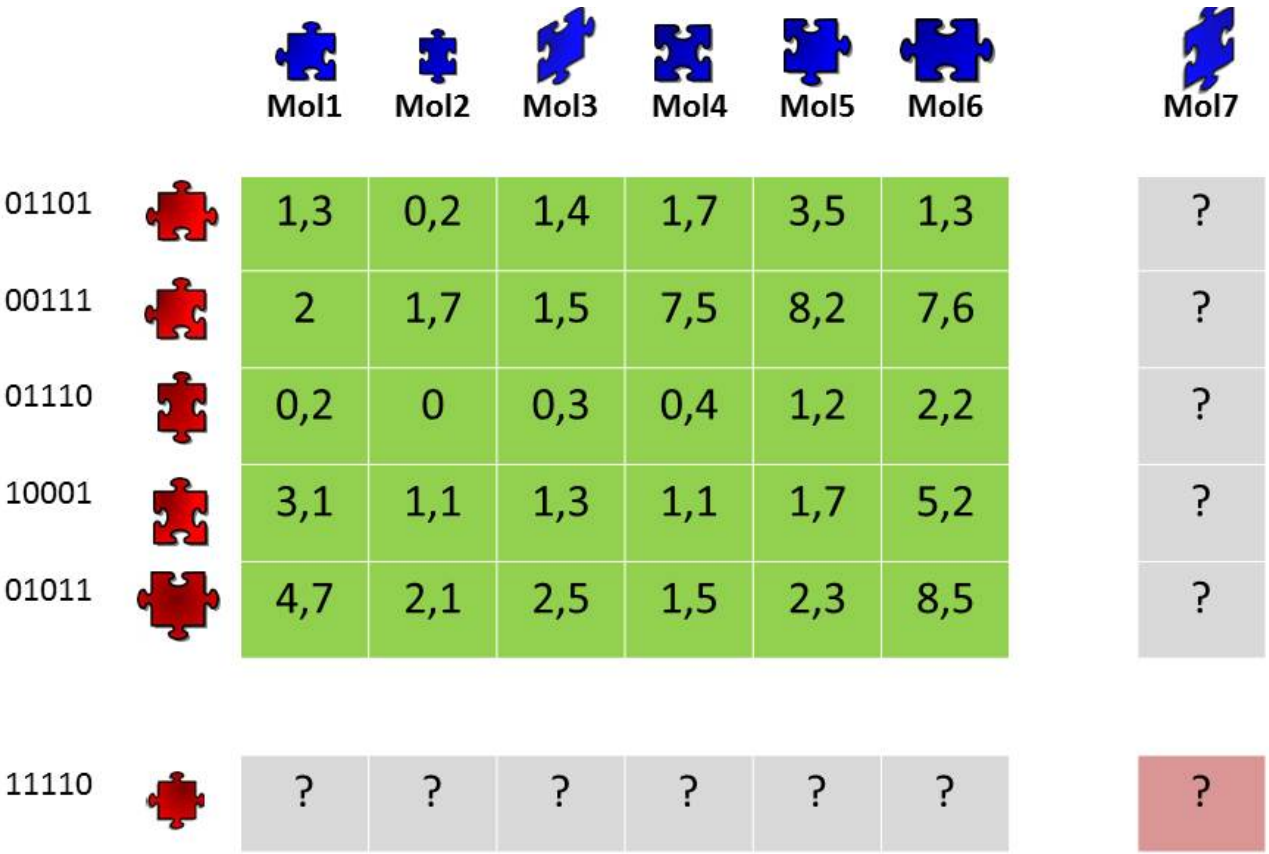
$$f(\boldsymbol{x}, \boldsymbol{t}) = \sum_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}) \in \mathcal{D}} \alpha_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})} k(\boldsymbol{x}, \bar{\boldsymbol{x}}) \cdot g(\boldsymbol{t}, \bar{\boldsymbol{t}}) = \sum_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}) \in \mathcal{D}} \alpha_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})} \Gamma((\boldsymbol{x}, \boldsymbol{t}), (\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}))$$

Least-squares minimization with $\mathbf{z} = \mathrm{vec}(Y)$:

$$\min_{\boldsymbol{\alpha}} ||\boldsymbol{\Gamma}\boldsymbol{\alpha} - \mathbf{z}||_2^2 + \lambda \boldsymbol{\alpha}^\intercal \boldsymbol{\Gamma} \boldsymbol{\alpha}$$

[32] Stock et al., A comparative study of pairwise learning methods based on kernel ridge regression, Neural Computation 2018

Two-step zero-shot learning[33] [34]



[33] Pahikkala et al. A two-step approach for solving full and almost full cold-start problems in dyadic prediction, ECML/PKDD 2014.

[34] Romero-Paredes and Torr, An embarrassingly simple approach to zero-shot learning, ICML 2015.

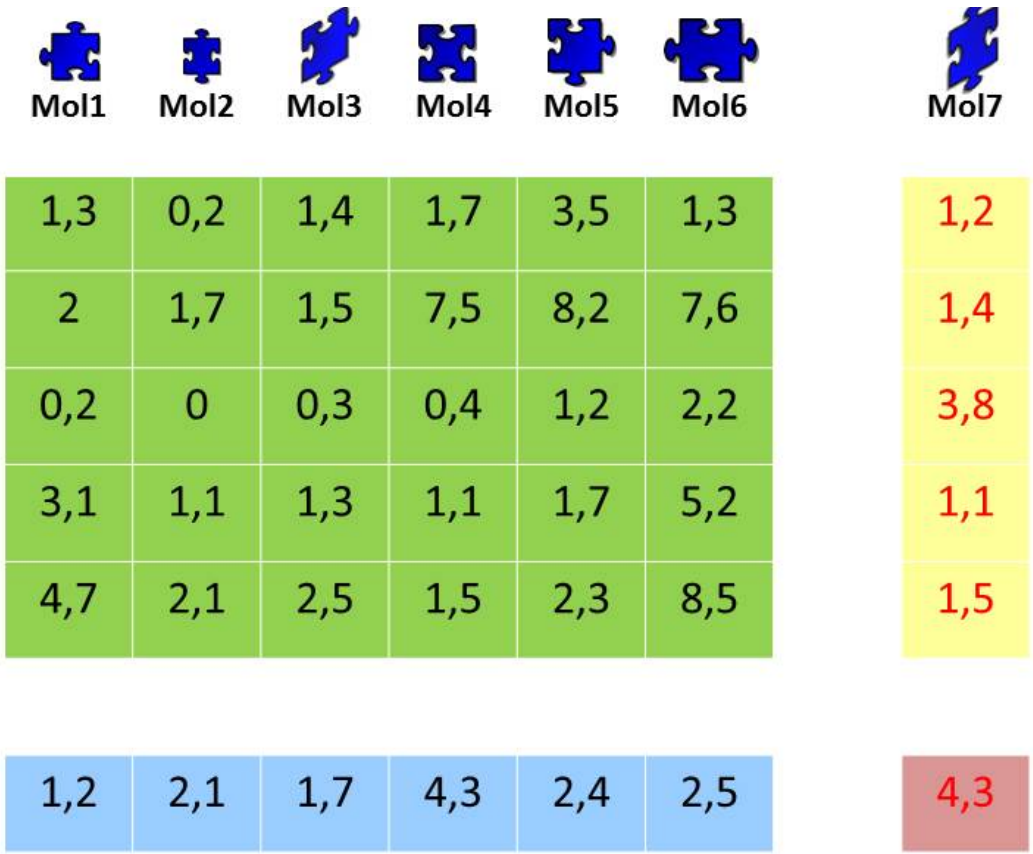
Two-step zero-shot learning[35] [36]

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101 	1,3	0,2	1,4	1,7	3,5	1,3
00111 	2	1,7	1,5	7,5	8,2	7,6
01110 	0,2	0	0,3	0,4	1,2	2,2
10001 	3,1	1,1	1,3	1,1	1,7	5,2
01011 	4,7	2,1	2,5	1,5	2,3	8,5
11110 	1,2	2,1	1,7	4,3	2,4	2,5

[35] Pahikkala et al. A two-step approach for solving full and almost full cold-start problems in dyadic prediction, ECML/PKDD 2014.

[36] Romero-Paredes and Torr, An embarrassingly simple approach to zero-shot learning, ICML 2015.

Two-step zero-shot learning[37] [38]



[37] Pahikkala et al. A two-step approach for solving full and almost full cold-start problems in dyadic prediction, ECML/PKDD 2014.

[38] Romero-Paredes and Torr, An embarrassingly simple approach to zero-shot learning, ICML 2015.

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\mathsf{T}$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_m))^\mathsf{T}$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\intercal$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_m))^\intercal$$

- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\intercal A^{IT} = \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_1 \mathbf{I}\right)^{-1} Y$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\intercal$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_m))^\intercal$$

- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\intercal A^{IT} = \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_1 \mathbf{I}\right)^{-1} Y$$

- Step 2: generalizing to new targets

$$f^{\mathsf{TS}}(\mathbf{x}, \mathbf{t}) \quad = \quad \mathbf{g}(\mathbf{t})^\intercal \left(\mathbf{G} + \lambda_2 \mathbf{I}\right)^{-1} \mathbf{f}_T(\mathbf{x})^\intercal$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\mathsf{T}$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_m))^\mathsf{T}$$

- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\mathsf{T} A^{IT} = \mathbf{k}(\mathbf{x})^\mathsf{T} (\mathbf{K} + \lambda_1 \mathbf{I})^{-1} Y$$

- Step 2: generalizing to new targets

$$\begin{aligned} f^{\mathsf{TS}}(\mathbf{x}, \mathbf{t}) &= \mathbf{g}(\mathbf{t})^\mathsf{T} (\mathbf{G} + \lambda_2 \mathbf{I})^{-1} \mathbf{f}_T(\mathbf{x})^\mathsf{T} \\ &= \mathbf{k}(\mathbf{x})^\mathsf{T} (\mathbf{K} + \lambda_1 \mathbf{I})^{-1} Y (\mathbf{G} + \lambda_2 \mathbf{I})^{-1} \mathbf{g}(\mathbf{t}) \end{aligned}$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\mathsf{T}$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_m))^\mathsf{T}$$

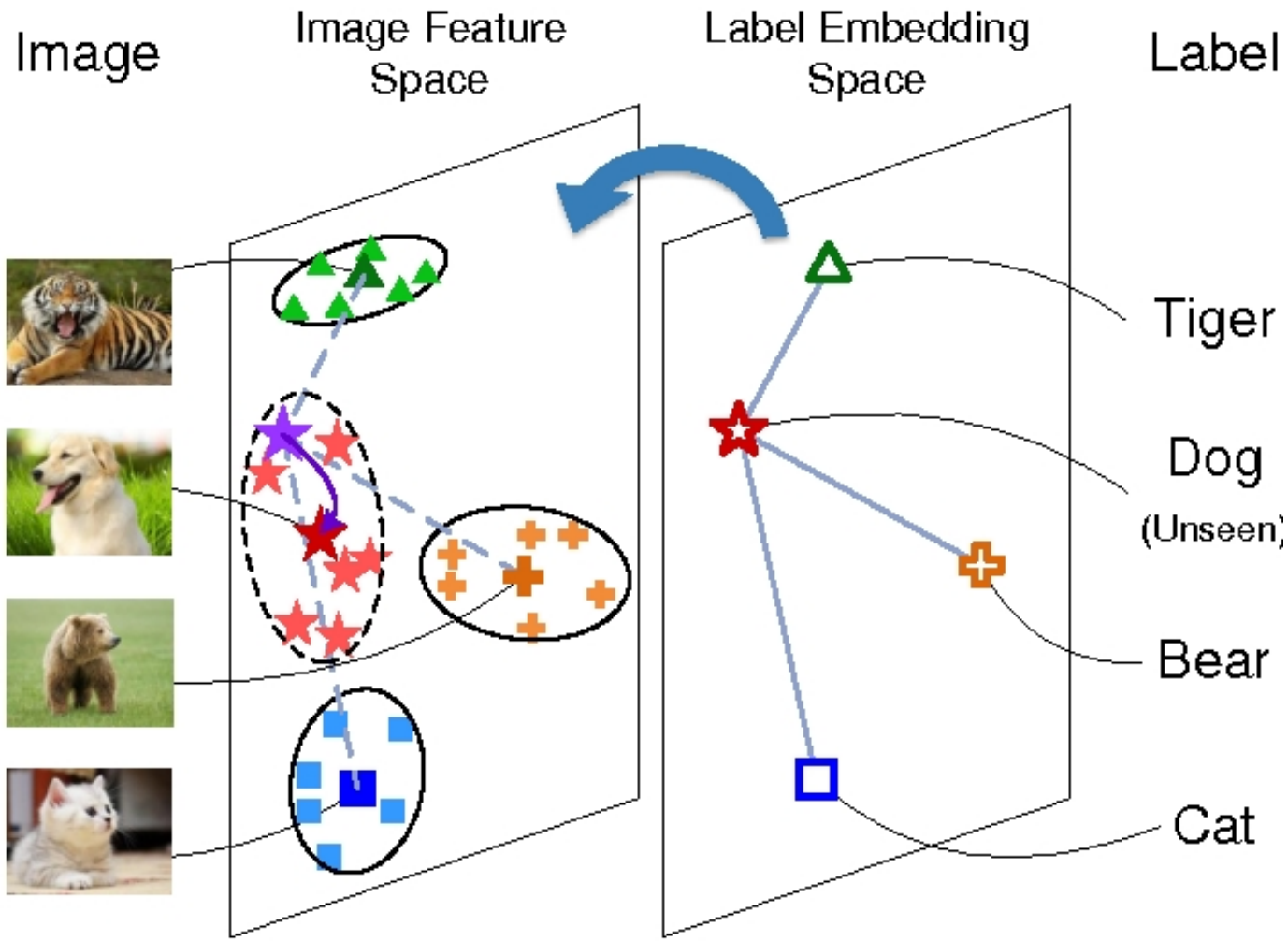
- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\mathsf{T} A^{IT} = \mathbf{k}(\mathbf{x})^\mathsf{T} \left(\mathbf{K} + \lambda_1 \mathbf{I}\right)^{-1} Y$$

- Step 2: generalizing to new targets

$$\begin{aligned} f^{\mathsf{TS}}(\mathbf{x}, \mathbf{t}) &= \mathbf{g}(\mathbf{t})^\mathsf{T} \left(\mathbf{G} + \lambda_2 \mathbf{I}\right)^{-1} \mathbf{f}_T(\mathbf{x})^\mathsf{T} \\ &= \mathbf{k}(\mathbf{x})^\mathsf{T} \left(\mathbf{K} + \lambda_1 \mathbf{I}\right)^{-1} Y \left(\mathbf{G} + \lambda_2 \mathbf{I}\right)^{-1} \mathbf{g}(\mathbf{t}) \\ &= \mathbf{k}(\mathbf{x})^\mathsf{T} A^{\mathsf{TS}} \mathbf{g}(\mathbf{t}) \\ &= \boldsymbol{w}^T \left(\phi(\boldsymbol{x}) \otimes \psi(\boldsymbol{t})\right) \end{aligned}$$

Zero-shot learning in computer vision



Zero-shot learning in computer vision

Pairwise model representation as before:

$$f(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{w}^T \left(\phi(\boldsymbol{x}) \otimes \psi(\boldsymbol{t})\right)$$

Inference in a structured prediction fashion:

$$\hat{c}(\boldsymbol{x}) = \arg\max_{\boldsymbol{t} \in \mathcal{T}} f(\boldsymbol{x}, \boldsymbol{t})$$

Different optimization problems:

- Multi-class objective[39]
- Ranking objective[40]
- Regression objective[41]
- Canonical correlation analysis

Different model formulations:

- Linear embeddings
- Nonlinear embeddings

---

[39] Akata et al., Evaluation of Output Embeddings for Fine-Grained Image Classification, CVPR2015

[40] Frome et al., Devise: A deep visual-semantic embedding model, NIPS 2013

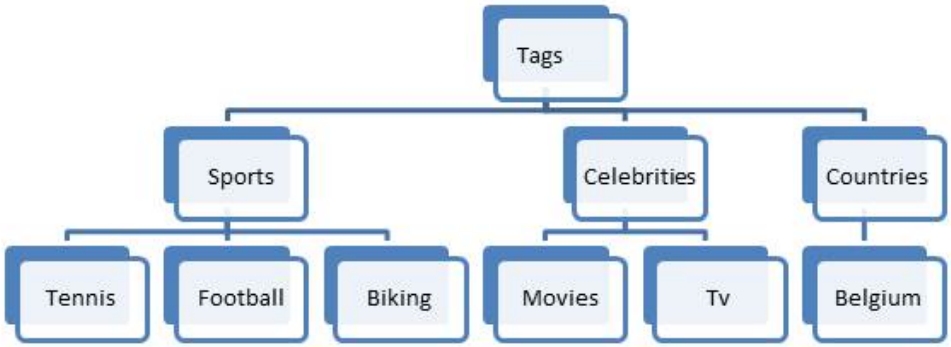
[41] Socher et al., g. Zero-shot learning through cross-modal transfer, NIPS 2013

Question

In which situation(s) is it useful to exploit target relations and representations?

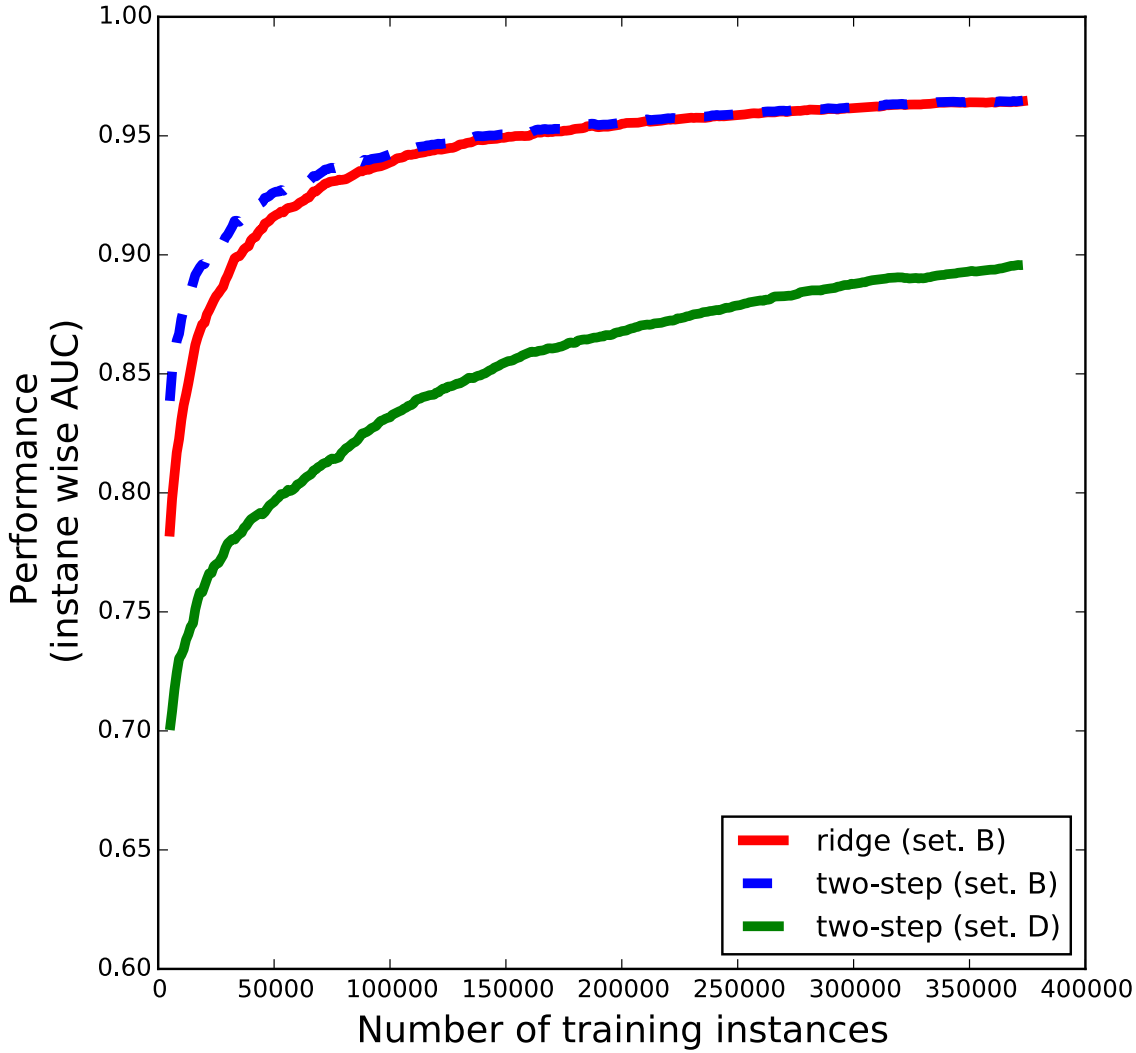
- In Setting B, when n is sufficiently large
- In Setting B, when n is sufficiently small
- In Setting D, when n is sufficiently large
- In Setting D, when n is sufficiently small

A case study on the Wikipedia dataset



01101	**Text1**	0	0	0	0	0	1
00111	**Text2**	0	0	1	0	1	1
01110	**Text3**	0	0	0	1	1	0
10001	**Text4**	0	0	1	0	1	0
01011	**Text5**	1	0	0	1	0	0
11110	**Text6**	?	?	?	?	?	?

The answer



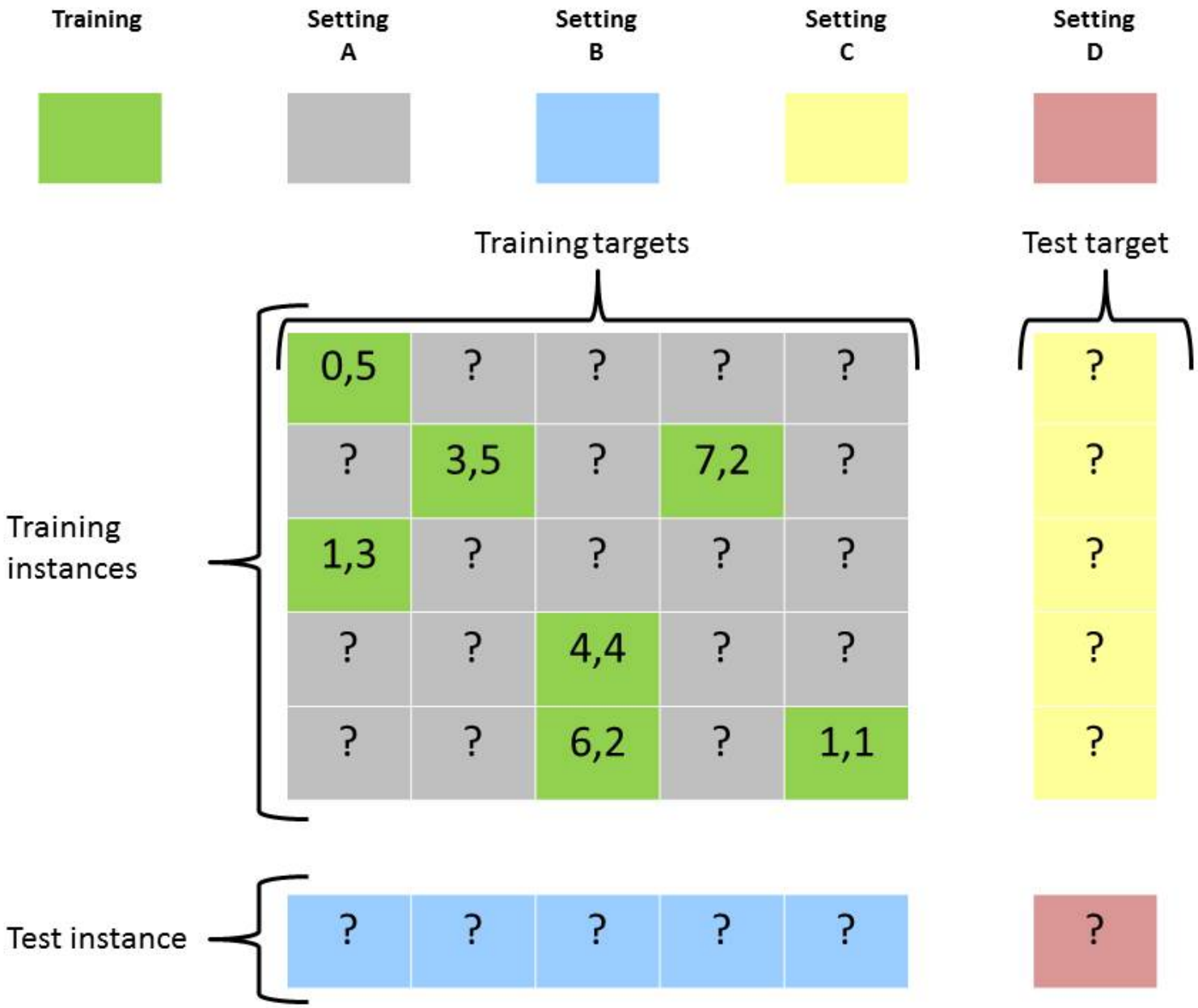
$12,000$ labels: from $5,000$ to $350,000$ instances[42]

[42] M. Stock, Exact and efficient algorithms for pairwise learning, PhD thesis, 2017

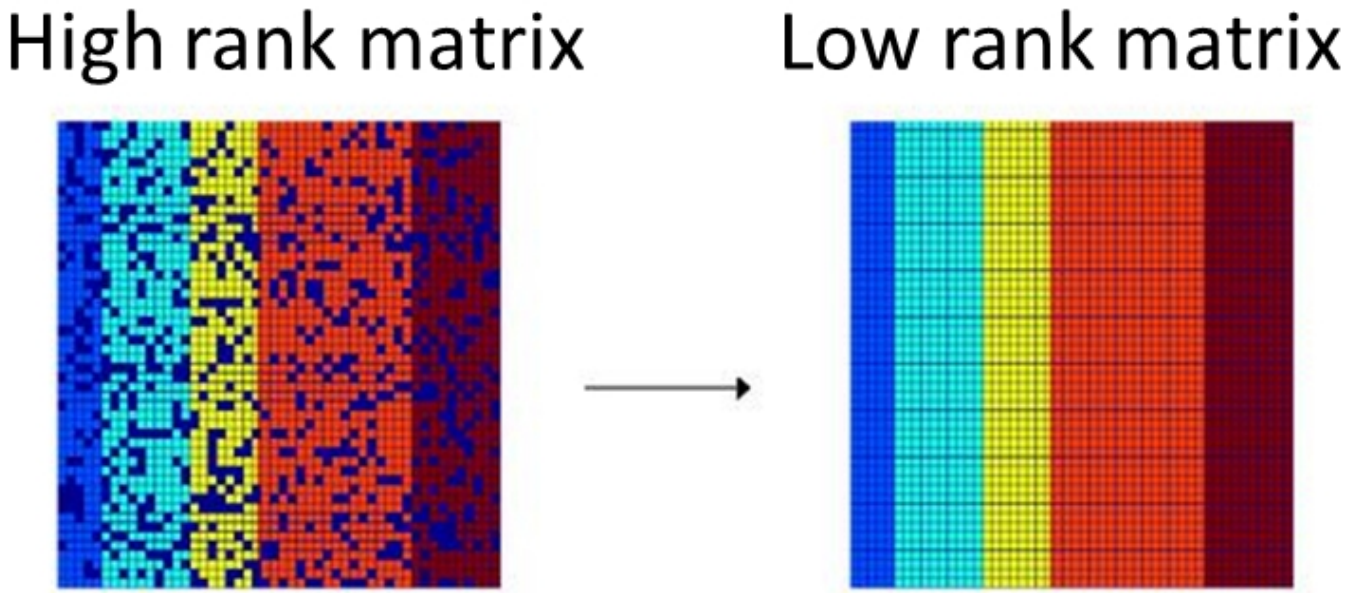
A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B
Similarity-enforcing methods	B
Relation-exploiting methods	B and D
Relation-constructing methods	B
Representation-exploiting methods	B and D
Representation-constructing methods	A and B

Different learning settings revisited



Low-rank approximation in Settings B and C



Typically perform a low-rank approximation of the parameter matrix[43]:

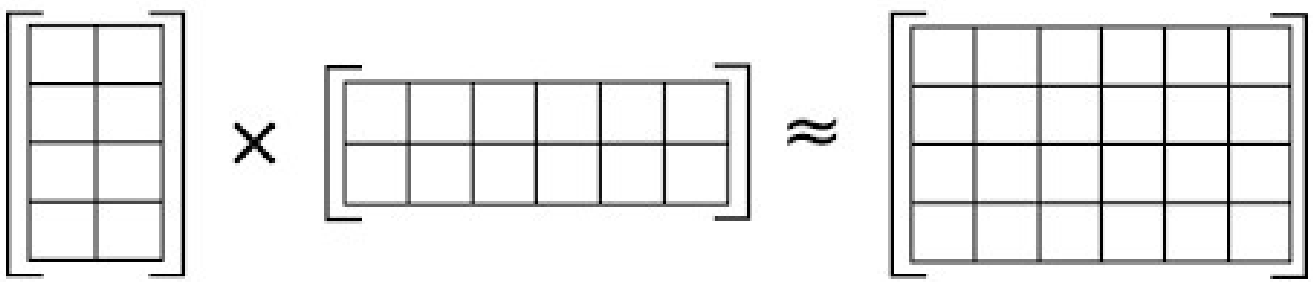
$$\min_A ||Y - XA||_F^2 + \lambda \operatorname{rank}(A)$$

[43] Chen et al., A convex formulation for learning shared structures from multiple tasks, ICML 2009.

Low-rank approximation in Settings B and C

- A: parameter matrix of dimensionality $p \times m$
- p: the number of features
- m: the number of targets
- Assume a low-rank structure of A:

$$U \quad \times \quad V \quad = \quad A$$



- We can write $A = VU$ and $A\boldsymbol{x} = VU\boldsymbol{x}$
- V is a $p \times \hat{m}$ matrix
- U i an $\hat{m} \times m$ matrix
- $\hat{m}$ is the rank of A

Low-rank approximation in Settings B and C

Overview of methods

- Popular for multi-output regression, multi-task learning and multi-label classification
- Linear as well as nonlinear methods
- Algorithms:
  - Principal component analysis[44], Canonical correlation analysis[45], Partial least squares
  - Singular value decomposition[46], Alternating structure optimization[47]
  - Compressed sensing[48], Output codes[49], Landmark labels[50], Bloom filters[51], Auto-encoders[52]

---

[44] Weston et al., Kernel dependency estimation, NIPS 2002

[45] Multi-label prediction via sparse infinite CCA, NIPS 2009

[46] Tai and Lin, Multilabel classification with principal label space transformation, Neural Computation 2012

[47] Zhou et al., Clustered Multi-Task Learning Via Alternating Structure Optimization, NIPS 2011

[48] Hsu et al., Multi-label prediction via compressed sensing. NIPS 2009

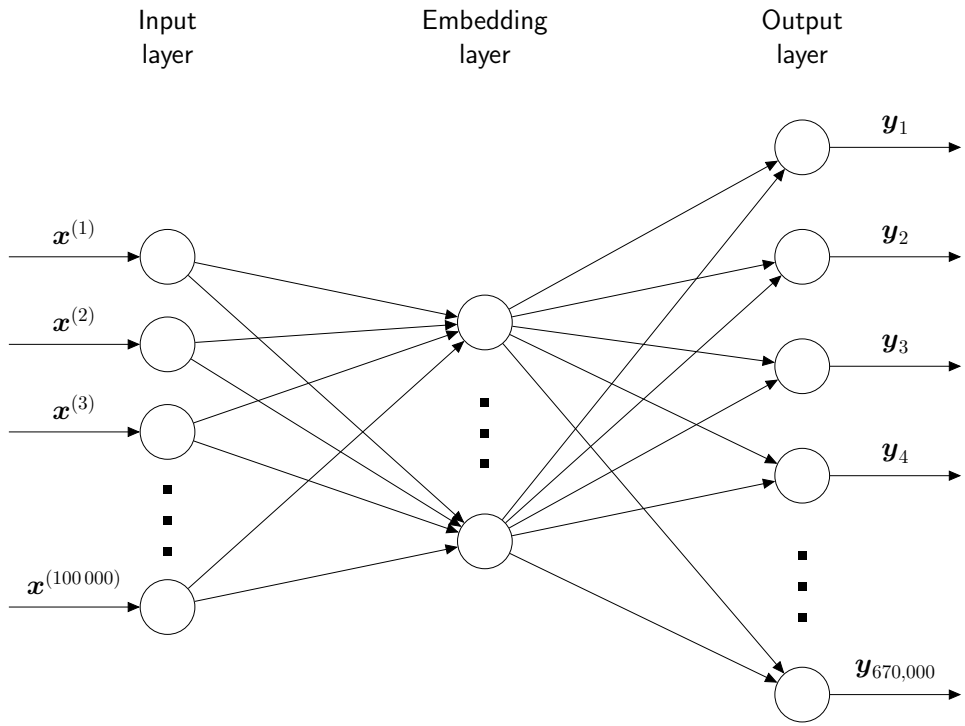
[49] Zhang and Schneider, Multi-label Output Codes using Canonical Correlation Analysis, UAI 2011

[50] Balasubramanian and Lebanon, The landmark selection method for multiple output prediction, ICML 2012

[51] Cissé et al., Robust bloom filters for large multilabel classification tasks, NIPS 2013

[52] Wicker et al., A nonlinear label compression and transformation method for multi-label classification using autoencoders, PAKDD 2016

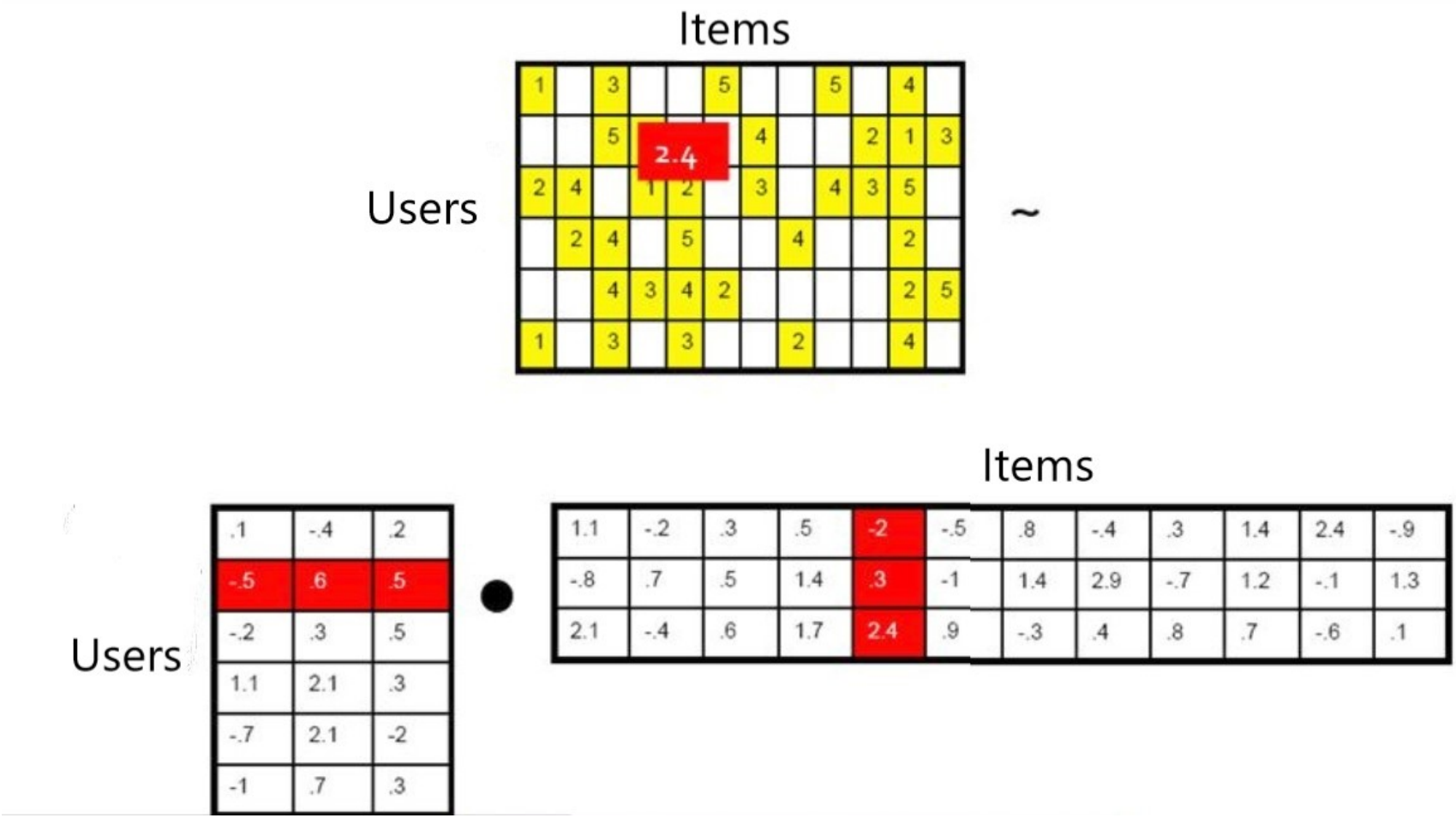
Target embeddings in neural networks



- Mapping input to output via embedding layer
- Nonlinear alternative to $A\boldsymbol{x} = VU\boldsymbol{x}$

Low-rank approximation in Setting A

Factorize the matrix Y instead of the parameter matrix A:



$$Y \quad = \quad U \quad \times \quad V$$

Low-rank approximation in Setting A

Overview of algorithms

- Nuclear norm minimizationan[53]
- Gaussian processes[54]
- Probabilistic methods[55]
- Spectral regularization[56]
- Non-negative matrix factorization[57]
- Alternating least-squares minimization[58]

[53] Candes and Recht, Exact low-rank matrix completion via convex optimization. Foundations of Computational Mathematics 2008

[54] Lawrence and Urtasun, Non-linear matrix factorization with Gaussian processes, ICML 2009

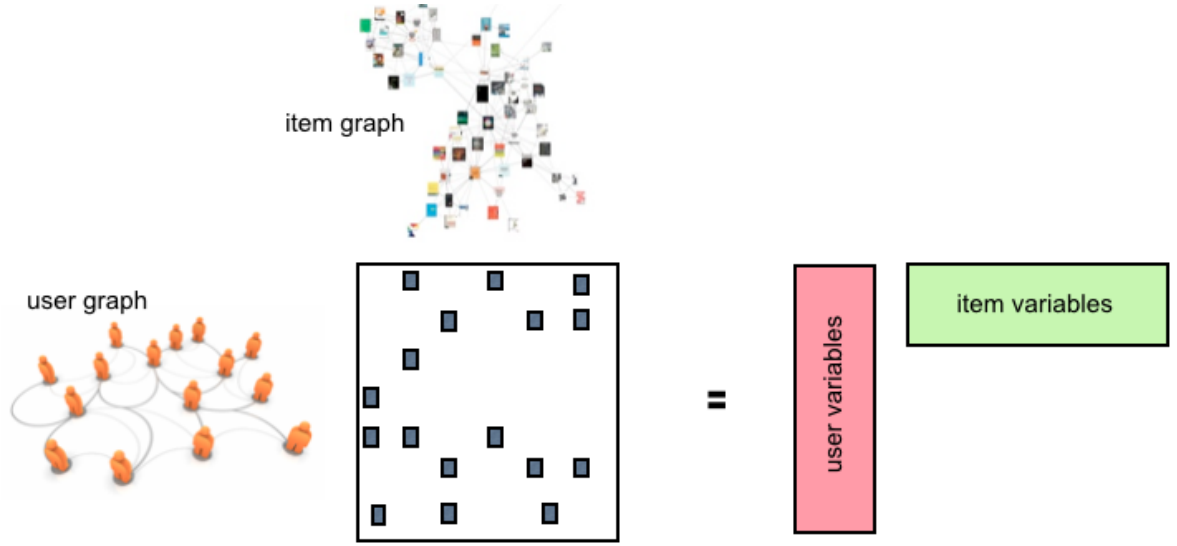
[55] Shan and Banerjee, Generalized probabilistic matrix factorizations for collaborative filtering, ICDM 2010

[56] Mazumder et al., Spectral regularization algorithms for learning large incomplete matrices., JMLR 2010

[57] Gaujoux and Seoighe, A flexible R package for nonnegative matrix factorization. BMC bioinformatics 2010

[58] Jain et al., Low-rank matrix completion using alternating minimization, ACM Symposium on Theory of Computing 2013

Matrix factorization with side information for Setting A



- Construct **implicit** features $(\boldsymbol{x}^I, \boldsymbol{t}^I)$ for users and items with matrix factorization methods
- Exploit **explicit** features $(\boldsymbol{x}^E, \boldsymbol{t}^E)$ (a.k.a. side information)
- Concatenate:

$$\boldsymbol{x}^C = (\boldsymbol{x}^I, \boldsymbol{x}^E), \qquad \boldsymbol{t}^C = (\boldsymbol{t}^I, \boldsymbol{t}^E)$$

- Apply methods that we have seen before[59][60]:

$$f(\boldsymbol{x}^C, \boldsymbol{t}^C) = \boldsymbol{w}^T \big(\phi(\boldsymbol{x}^C) \otimes \psi(\boldsymbol{t}^C)\big)$$

[59] Menon and Elkan, A log-linear model with latent features for dyadic prediction, ICDM 2010

[60] Volkovs and Zemel, Collaborative filtering with 17 parameters, NIPS 2012

When is it useful to construct target representations?

Does not work well in extreme multi-label classification[61]:

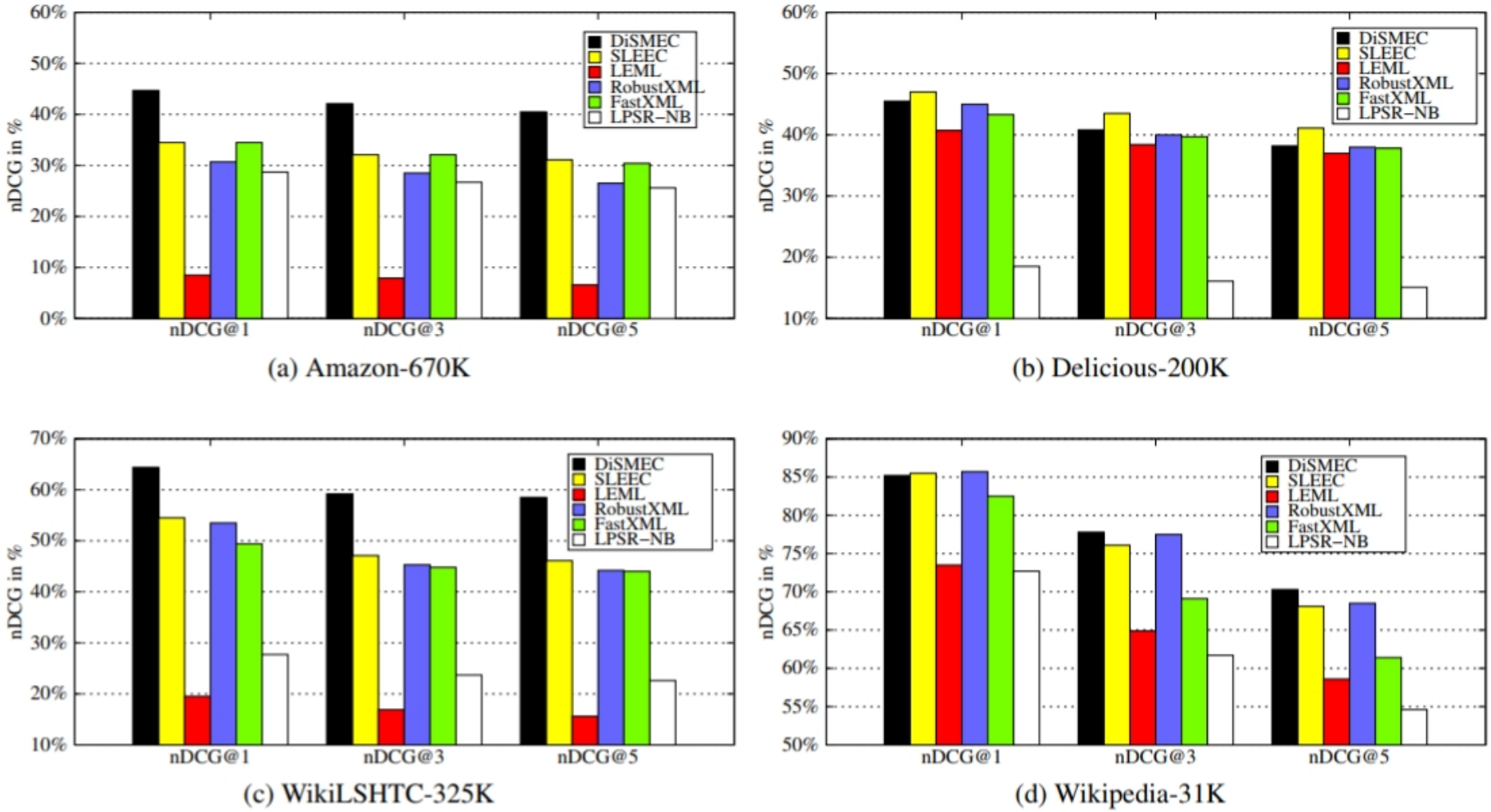


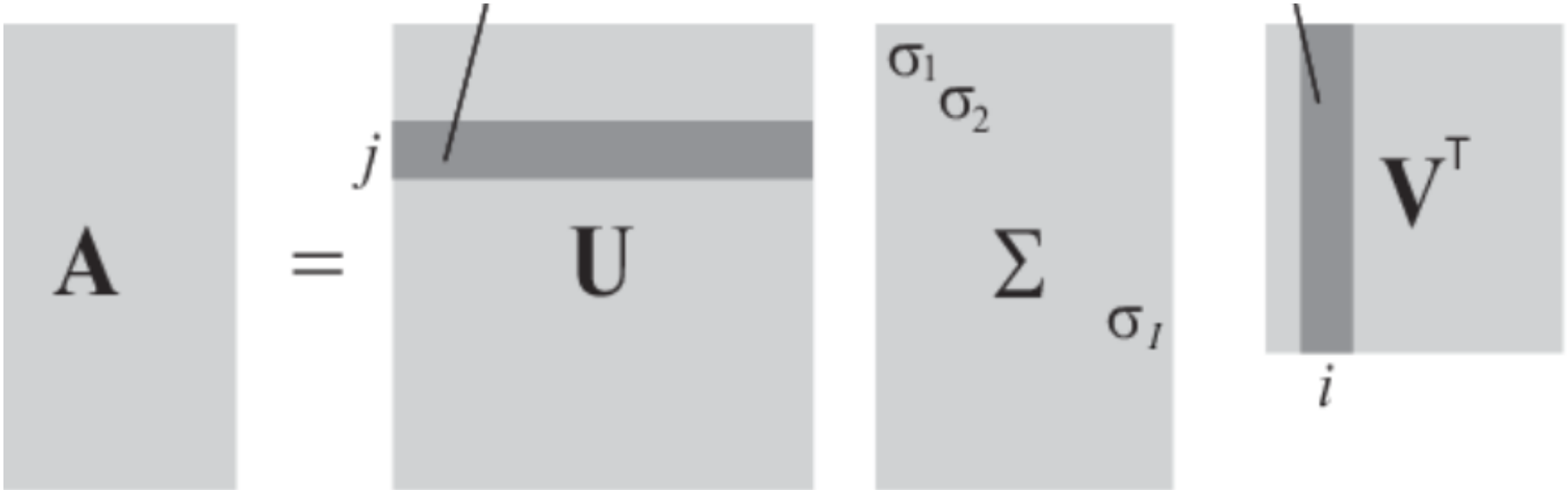
Figure 3: nDCG@k for k=1, 3 and 5

[61] Babbar and Schölkopf, DISMEC: Distributed Sparse Machines for Extreme Multi-label classification, WSDM 2017

When is it useful to construct target representations?

SVD interpretation

Representation of instance (or feature) Representation of target



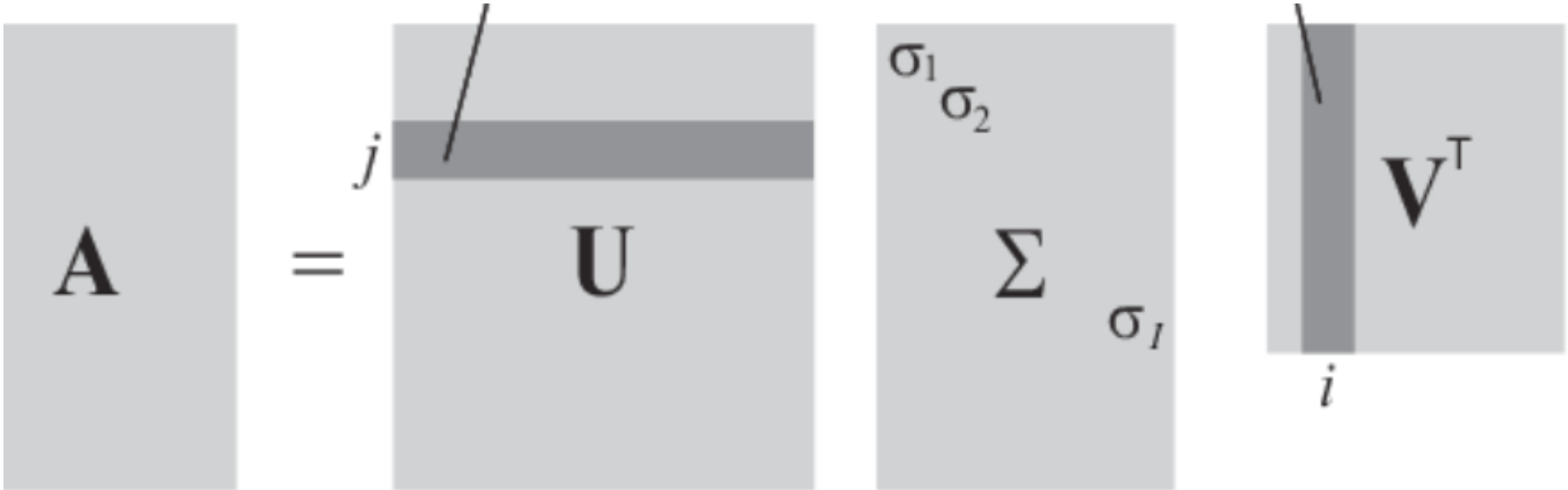
$p \times m$ $p \times p$ $p \times m$ $m \times m$

- $\sigma_1, \sigma_2, ...$: singular values of A
- Rank of A = number of non-zero **singular values**

When is it useful to construct target representations?

SVD interpretation

Representation of instance (or feature) Representation of target



$p \times m$ $p \times p$ $p \times m$ $m \times m$

- $\sigma_1, \sigma_2, ...$: singular values of A
- Rank of A = number of non-zero **singular values**
- **High rank** when a lot of singular values differ from zero
- **Low rank** when a lot of singular values are zero
- Singular values **give insight** in what can be gained

Conclusions

- Multi-target prediction is an active field of research that connects different types of machine learning problems
- In the corresponding subfields of machine learning, problems have typically been solved in isolation, without establishing connections between methods
- When analyzing MTP methods, it is important to understand several concepts, such as the influence of loss functions, and the availability and absence of side knowledge

Upcoming paper:
Waegeman et al.
Multi-Target Prediction:
A Unifying View on Problems and Methods
(available in a few days on Arxiv)

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20
- **Masera and Blanzieri**, AWX: An integrated approach to hierarchical multi-label classification, Tuesday 14h40

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20
- **Masera and Blanzieri**, AWX: An integrated approach to hierarchical multi-label classification, Tuesday 14h40
- **Decubber et al.**, Deep F-measure maximization in multi-label classification: a comparative study, Tuesday 14h20

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20
- **Masera and Blanzieri**, AWX: An integrated approach to hierarchical multi-label classification, Tuesday 14h40
- **Decubber et al.**, Deep F-measure maximization in multi-label classification: a comparative study, Tuesday 14h20
- **Park and Read**, A blended metric for multi-label optimization and evaluation, Thursday 11h40

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20
- **Masera and Blanzieri**, AWX: An integrated approach to hierarchical multi-label classification, Tuesday 14h40
- **Decubber et al.**, Deep F-measure maximization in multi-label classification: a comparative study, Tuesday 14h20
- **Park and Read**, A blended metric for multi-label optimization and evaluation, Thursday 11h40
- **Rafailidis and Crestani**, Deep collaborative filtering with multifaceted contextual information in location-based social networks, Thursday 14h20

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20
- **Masera and Blanzieri**, AWX: An integrated approach to hierarchical multi-label classification, Tuesday 14h40
- **Decubber et al.**, Deep F-measure maximization in multi-label classification: a comparative study, Tuesday 14h20
- **Park and Read**, A blended metric for multi-label optimization and evaluation, Thursday 11h40
- **Rafailidis and Crestani**, Deep collaborative filtering with multifaceted contextual information in location-based social networks, Thursday 14h20
- **Du et al.**, POLAR: Attention-based CNN for one-shot personalized article recommendation, Thursday 14h40

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20
- **Masera and Blanzieri**, AWX: An integrated approach to hierarchical multi-label classification, Tuesday 14h40
- **Decubber et al.**, Deep F-measure maximization in multi-label classification: a comparative study, Tuesday 14h20
- **Park and Read**, A blended metric for multi-label optimization and evaluation, Thursday 11h40
- **Rafailidis and Crestani**, Deep collaborative filtering with multifaceted contextual information in location-based social networks, Thursday 14h20
- **Du et al.**, POLAR: Attention-based CNN for one-shot personalized article recommendation, Thursday 14h40
- **Lan et al.**, Personalized thread recommendation of MOOC discussion forums, Thursday 14h20

Multi-target prediction papers at ECML/PKDD 2018

- **Djerrab et al.**, Output Fisher embedding regression, Tuesday, 15h20
- **Pikalos et al.**, Global multi-output decision trees for interaction prediction, Thursday 11h20
- **Masera and Blanzieri**, AWX: An integrated approach to hierarchical multi-label classification, Tuesday 14h40
- **Decubber et al.**, Deep F-measure maximization in multi-label classification: a comparative study, Tuesday 14h20
- **Park and Read**, A blended metric for multi-label optimization and evaluation, Thursday 11h40
- **Rafailidis and Crestani**, Deep collaborative filtering with multifaceted contextual information in location-based social networks, Thursday 14h20
- **Du et al.**, POLAR: Attention-based CNN for one-shot personalized article recommendation, Thursday 14h40
- **Lan et al.**, Personalized thread recommendation of MOOC discussion forums, Thursday 14h20
- **Marecek et al.**, Matrix completion under interval uncertainty, Thursday 16h30

Questions? Remarks?