CS-E4710 Machine Learning: Supervised Methods

Lecture 12: Predicting multiple and structured labels

Juho Rousu

December 1, 2020

Department of Computer Science
Aalto University

Label ranking

Label ranking

- Training output are given as lists of pairwise preferences $A \succ B$ between labels: defines a partial order "label A is preferable to label B"
- Model ranks all labels: outputs a total order, that is, all possible labels given in sequential order
- Loss function is between two rankings: loss in incurred if the prediction has $B \succ A$ and the ground truth has $A \succ B$

Training

X1	X2	X3	X4	Preferences
0.34	0	10	174	$A \succ B$, $B \succ C$, $C \succ D$
1.45	0	32	277	$B \succ C$
1.22	1	46	421	$B \succ D$, $A \succ D$, $C \succ D$, $A \succ C$
0.74	1	25	165	$C \succ A$, $C \succ D$, $A \succ B$
0.95	1	72	273	$B \succ D$, $A \succ D$
1.04	0	33	158	$D \succ A$, $A \succ B$, $C \succ B$, $A \succ C$

Prediction $B \succ D \succ C \succ A$

0.92	1	81	382	4	1	3	2

Ground truth

LOSS

0.92	1	81	382	2	1	3	4

Label ranking: definitions

- X is the input space, $\Sigma = \{1, \ldots, K\}$ set of labels
- $\mathcal{Y} = \{Y | Y \subset \Sigma \times \Sigma\}$ is the output space of all possible sets of pairwise preferences $y_k \succ y_l$ over K labels
- $S = \{(\mathbf{x}_i, Y_i)\}_{i=1}^m, (\mathbf{x}_i, Y_i) \in X \times \mathcal{Y}$ is a set of training examples
- Each $Y_i \in \mathcal{Y}$ is a set of pairwise preferences
- $(p \succ q) \in Y_i$ denotes label p is preferable to label q given input x_i

From multiclass classification to label ranking

- A multiclass predictor based on linear classification is relatively straightforward to convert to a label ranking model
- For each label p, we have a model $\mathbf{w}_p^T \mathbf{x}$ that assigns a compatibility score between the inputs $\mathbf{x}$ and the label p
- In multiclass classification, we only needed to make the correct class y_i the top-ranked one $\mathbf{w}_{y_i}^T \mathbf{x}_i \geq \mathbf{w}_p^T \mathbf{x}_i$ for all $p \neq y_i$
- In label ranking need to order all labels instead of just ranking the correct class to the top:

$$\mathbf{w}_p{}^T \mathbf{x} \geq \mathbf{w}_q{}^T \mathbf{x} \text{ if } (p \succ q) \in Y$$

Label ranker as a classifier

- The constraint
$$\mathbf{w}_q^T \mathbf{x}_i \geq \mathbf{w}_p^T \mathbf{x}_i$$
corresponds to a hyperplane classifier
$$y_{pqi} \mathbf{w}_{pq}^T \mathbf{x}_i \geq 0$$
where $\mathbf{w}_{pq} = \mathbf{w}_p - \mathbf{w}_q$ and
$$y_{pqi} = \begin{cases} +1 & \text{if } p \succ q \in Y_i \\ -1 & \text{if } q \succ p \in Y_i \\ 0 & \textit{otherwise} \end{cases}$$
- It is unlikely that the data will be linearly separable for all hyperplanes
- Minimization of the number of misclassified data is NP-hard
- We will again use the Hinge loss as the surrogate loss function

Hinge loss for label ranking

- The loss for one example $(\mathbf{x}, Y)$ is an average of the Hinge losses over the set of label preferences Y:

$$\frac{1}{|Y|} \sum_{p \succ q \in Y} \max(0, 1 - (\mathbf{w}_p{}^T \mathbf{x} - \mathbf{w}_q{}^T \mathbf{x}))$$

- Maximizes the average functional margin over pairs of label preferences
- Minimizes an convex upper bound on the number of labels that are in inverted order (Kendall's distance of ranked sequence of labels)

Label ranking SVM[1]

- Label ranking SVM is given by

$$\min_{\mathbf{w}_k, k=1\dots,K} \frac{\lambda}{2} \sum_{k=1}^{K} \left\| \mathbf{w}_k \right\|^2 + \sum_{i=1}^{m} \frac{1}{|Y_i|} \sum_{\{p \succ q\} \in Y_i} \xi_{pqi}$$

$$\text{s.t. } {\mathbf{w}_p}^T \mathbf{x}_i - {\mathbf{w}_q}^T \mathbf{x}_i > 1 - \xi_{pqi}$$

$$\text{for all } \{p \succ q\} \in Y_i, i = 1, \dots, m$$

$$\xi_{pqi} \geq 0$$

- Objective:
  - Regularizes the sum of norms of all label classifiers - indirectly maximizes the margins
  - Slack ξ_{pqi} corresponds to the upper bound on the Hinge loss for $\mathbf{x}_i$ and label pairs $(p, q) \in Y$

[1] Gärtner & Vembu, 2009

Multilabel classification

Multilabel classification

- In multilabel classification, a subset of the labels $y_k, k = 1, \ldots, K$ is associated with each input
- Loss functions are defined on vectors of labels

Training

X1	X2	X3	X4	A	B	C	D
0.34	0	10	174	0	**1**	**1**	0
1.45	0	32	277	0	**1**	0	**1**
1.22	1	46	421	0	0	0	1
0.74	1	25	165	0	**1**	**1**	**1**
0.95	1	72	273	**1**	0	**1**	0
1.04	0	33	158	**1**	**1**	**1**	0

Binary preferences on a fixed set of items: liked or disliked

Prediction

0.92	1	81	382	**0**	**1**	**0**	**1**

LOSS

Ground truth

0.92	1	81	382	**1**	**1**	**0**	**1**

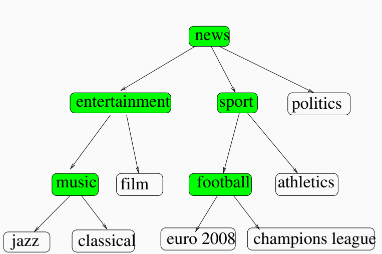
Loss function compares two multilabels

Multilabel classification

- Inputs are vectors $\mathbf{x} \in \mathbb{R}^d$ (possibly obtained through some preprocessing)
- Outputs are binary vectors $\mathbf{y} = (y_1, \ldots, y_K) \in \{-1, +1\}^K = \mathcal{Y}$
- Loss function compares two binary vectors $\mathbf{y}$ and $\mathbf{y}'$
    - Zero-one loss: $L_{0/1}(\mathbf{y}, \mathbf{y}') = \begin{cases} 1 & \mathbf{y} \neq \mathbf{y}' \\ 0 & \mathbf{y} = \mathbf{y}' \end{cases}$
    - Hamming loss: $L_{Hamming}(\mathbf{y}, \mathbf{y}') = \sum_{k=1}^{K} \mathbf{1}\{y_k \neq y_k'\}$
    - Structural losses: based on the dependency structures of the labels y_k (e.g. hierarchical)

Running example: Hierarchical Multilabel Classification

Goal: Given document $\mathbf{x}$, and hierachy $T = (V, E)$, predict multilabel $\mathbf{y} \in \{+1, -1\}^k$ where the positive labels $\mathbf{y}_i$ take the form of set of partial paths from root to an internal node in T



Binary relevance model for multilabel classification

Binary relevance (BR) models are a simple multilabel prediction approach relying on binary classification:

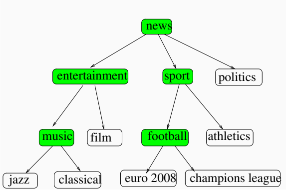
- Assume that the individual labels $y_k, 1 \leq k \leq K$ are independent (probably violated in practise!)
- Build a binary classifier $h_k(\mathbf{x}) \in \{-1, +1\}$ for each individual label y_k
- predicted multilabel is the vector $(h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_K(\mathbf{x}))$

Binary relevance model for multilabel classification

- Binary relevance models are often competitive in practice
- However, they ignore dependencies between the labels
- Thus the predicted vector $(h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_K(\mathbf{x}))$ may contain combinations of labels that are rarely or never seen in test data (e.g. some label y_k may be 1 only if another label y_j has value 1 as well)
- Another problem is that multilabel data is often biased towards the negative class:
  - Only few variables per example have value 1
  - Only a small fraction of examples has value 1 for a given variable
- The binary classifiers may be negatively biased as a consequence (have high False Negative rate)

Binary relevance model in Hierarchical Multilabel Classification

- BR model would predict each node of the hierarchy (topic) independently
- A very small fraction of documents belong to each specific topic: leaf nodes are dominated by negative examples, BR model might be biased towards the negative class
- Independent prediction may cause a child node to be predicted positive even if the parent is negative - this goes against of how we think of hierarchical taxonomies



Multilabel classification without BR decomposition

- Ideally, we would like to learn a model that directly predicts the multilabel vector $h : X \mapsto \{-1, +1\}^K$
- We start by defining a linear model mapping an input vector $\mathbf{x} \in \mathbb{R}^d$ to an output $\mathbf{y} \in \mathbb{R}^K$ by

$$\mathbf{W}^T\mathbf{x} = \mathbf{y}$$

  where $\mathbf{W} \in \mathbb{R}^{d \times K}$ is a **matrix** of weights, with weight vectors $\mathbf{w}_k$ as columns $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_k] = [w_{jk}]_{j=1,k=1}^{d,K}$
- We can think of each column defining a linear model $\mathbf{w}_k^T\mathbf{x}$ predicting the label y_k
- A weight w_{jk} is interpreted as the importance of input variable x_j to predict the label y_k

Multilabel classification BR decomposition

- We represent the compatibility of the pair $(\mathbf{x}, \mathbf{y})$ by the sum of margins of the column based models:
  $f(\mathbf{x}, \mathbf{y}) = \sum_{k=1}^{K} y_k \mathbf{w}_k^T \mathbf{x} = \mathbf{y}^T \mathbf{W}^T \mathbf{x}$
- Equivalently, we can write the same as a Frobenius inner product $\langle A, B \rangle_F = \sum_{i,j} a_{ij} b_{ij}$ between two matrices $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$
- We get $\mathbf{y}^T \mathbf{W}^T \mathbf{x} = \sum_{j=1}^{d} \sum_{k=1}^{K} w_{jk}(x_j y_k) = \langle \mathbf{W}, \mathbf{x}\mathbf{y}^T \rangle_F$
- The matrix $\mathbf{x}\mathbf{y}^T$ gives a joint representation for the input and output:

$$\mathbf{x}\mathbf{y}^T = \begin{bmatrix} x_1 y_1 & \dots & x_1 y_K \\ x_2 y_1 & \dots & x_2 y_K \\ \vdots & \ddots & \vdots \\ x_d y_1 & \dots & x_d y_K \end{bmatrix} = [y_1\mathbf{x}, y_2\mathbf{x}, \dots, y_K\mathbf{x}]$$

- An entry $x_j y_k$ models the dependency between the j'th input variable and the k'th label

Joint feature map

- We can flatten the two matrices into vectors by concatenating their columns into a long vector

$$\mathbf{w} = \mathbf{vec}(\mathbf{W}) = (\mathbf{w}_1^T, \mathbf{w}_2^T, \ldots, \mathbf{w}_K)^T$$

  and

$$\phi(\mathbf{x}, \mathbf{y}) = \mathbf{vec}(\mathbf{x}\mathbf{y}^T) = (y_1\mathbf{x}^T, y_2\mathbf{x}^T, \ldots, y_K\mathbf{x}^T)^T$$

- $\phi(\mathbf{x}, \mathbf{y})$ is an example of a **joint feature map** for the pair $(\mathbf{x}, \mathbf{y})$
- The compatibility score for the pair $(\mathbf{x}, \mathbf{y})$ can be now written as

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = \langle W, \mathbf{x}\mathbf{y}^T \rangle = \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y})$$

- The prediction of our model for input $\mathbf{x}$ will be

$$\mathbf{y}^* = \mathrm{argmax}_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y})$$

Learning objective

- Our goal will be to learn $\mathbf{w}$ so that the correct pairs $(\mathbf{x}_i, \mathbf{y}_i)$ are ranked above all the incorrect pairs $(\mathbf{x}_i, \mathbf{y}), \mathbf{y} \neq \mathbf{y}_i$
- We can express our goal as the constraint:

$$\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}) \geq 0, \text{ for all } \mathbf{y} \neq \mathbf{y}_i$$

- Or alternatively:

$$\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) \geq \max_{\mathbf{y} \neq \mathbf{y}_i} \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y})$$

- It is not likely that the constraint can be satisfied for all pairs $(\mathbf{x}_i, \mathbf{y}_i)$, i.e. the correct pairs $(\mathbf{x}_i, \mathbf{y}_i)$ may not be linearly separable for the incorrect pairs $(\mathbf{x}_i, \mathbf{y}), \mathbf{y} \neq \mathbf{y}_i$
- Minimization of the number of incorrect pairs ranked above the correct pairs is also computationally hard

Learning objective

- Hence we will use a soft margin formulation, corresponding to constraints

$$\mathbf{w}^T\phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T\phi(\mathbf{x}_i, \mathbf{y}) \geq 1 - \xi_i, \text{ for all } \mathbf{y} \neq \mathbf{y}_i$$

  which call for establishing a functional margin of at least $1 - \xi_i$ between the correct pair and all incorrect pairs
- The constraints correspond to a multilabel Hinge loss:

$$L_{MLHinge}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{w}) = \max_{\mathbf{y} \neq \mathbf{y}_i}(0, 1 - (\mathbf{w}^T\phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T\phi(\mathbf{x}_i, \mathbf{y}))$$

- The loss measures the amount of slack needed by the highest scoring incorrect multilabel to have a functional margin at least 1 compared to the correct multilabel

Multilabel SVM

- Adding regularization for the weight vector **w** we obtain an optimization problem for multilabel SVM:

$$\min_{\mathbf{w},\xi\geq 0} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{i=1}^{m}\xi_i$$
$$\text{s.t. } \mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}_i) - \mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}) \geq 1-\xi_i, \forall i, \mathbf{y}\in\mathcal{Y}-\{\mathbf{y}_i\}$$

,

- Alternatively, we can rewrite the optimization problem in terms of the multilabel Hinge loss:

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{i=1}^{m}L_{MLHinge}(\mathbf{x}_i,\mathbf{y}_i,\mathbf{w})$$

,

- Let us derive a stochastic gradient algorithm for this probem

Stochastic gradient optimization for multilabel SVM

- We rewrite the objective as a average over training points:

$$\min_{\mathbf{w}} \frac{1}{m}\sum_{i=1}^{m}\left(\frac{\lambda}{2}\|\mathbf{w}\|^2 + \max_{\mathbf{y}\neq\mathbf{y}_i}(0, 1-(\mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}_i) - \mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}))\right)$$

- The multilabel Hinge loss for a single training example is piecewise differentiable, with the gradient (formally subgradient):

$$\begin{aligned}\partial L_{MLHinge}(\mathbf{x}_i,\mathbf{y}_i,\mathbf{w}) &= \partial\left(\max_{\mathbf{y}\neq\mathbf{y}_i}(0, \left(1-(\mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}_i) - \mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}))\right))\right)\\ &= \phi(\mathbf{x}_i,\bar{\mathbf{y}}) - \phi(\mathbf{x}_i,\mathbf{y}_i)\end{aligned}$$

where $\bar{\mathbf{y}} = \mathrm{argmax}_{\mathbf{y}\neq\mathbf{y}_i}\left(1-(\mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}_i) - \mathbf{w}^T\phi(\mathbf{x}_i,\mathbf{y}))\right)$ is the incorrect multilabel with the smallest margin, that is, the highest scoring incorrect multilabel

Stochastic gradient optimization for multilabel SVM

Initialize $\mathbf{w} = 0$;

repeat

Draw a random training example $(\mathbf{x}_i, \mathbf{y}_i)$

Find the multilabel with the highest loss:

$\bar{\mathbf{y}} = \mathrm{argmax}_{\mathbf{y} \neq \mathbf{y}_i}(1 - (\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}))$

Update if Hinge loss is positive:

if $\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \phi(\mathbf{x}_i, \bar{\mathbf{y}}) < 1$ **then**

Choose a stepsize η

Update the weights towards the negative gradient:

$\mathbf{w} = \mathbf{w} - \eta(\lambda \mathbf{w} + \phi(\mathbf{x}_i, \bar{\mathbf{y}}) - \phi(\mathbf{x}_i, \mathbf{y}_i))$

end if

until Stopping criterion is satisfied

Tackling large multilabel spaces

- The bottleneck of the above stochastic gradient algorithm is finding the multilabel with the highest Hinge loss

$$\bar{\mathbf{y}} = \mathrm{argmax}_{\mathbf{y} \neq \mathbf{y}_i}(1 - (\mathbf{w}^T\phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T\phi(\mathbf{x}_i, \mathbf{y}))$$

- This is due to the large number of terms the maximization is computed over
- With K different labels $y_k \in \{-1, +1\}$, we have 2^K different binary multilabels $\mathbf{y}$, leading to maximization over $2^K - 1$ terms
- We need efficient methods to tackle the large multilabel space

Tackling large multilabel spaces

- In general, finding the multilabel with the highest Hinge loss is computationally hard

$$\bar{\mathbf{y}} = \mathrm{argmax}_{\mathbf{y} \neq \mathbf{y}_i} (1 - (\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}))$$

- Efficient (polynomial-time) algorithms exist for special structures, for example
  - Label sequence learning: dynamic programming algorithms similar to Hidden Markov Model inference algorithms
  - Hierarchical classification: dynamic programming over the tree
- Typically efficient algorithms rely one decomposing the compatibility score into a sum over the parts (substructures) of the output structure

$$\mathrm{argmax}_{\mathbf{y} \in \mathcal{Y}} \sum_{j=1}^{d_y} \mathbf{w}_j^T \phi_j(\mathbf{x}, \mathbf{y})$$

  and making use of the dependency structures between parts to avoid exhaustive enumeration of $\mathcal{Y}$

Tackling large multilabel spaces

- In many cases, no pre-defined structure is available
- In these cases, the training data can be used to give a approximate solution: we solve instead

$$\bar{\mathbf{y}}(\mathbf{x}) = \mathrm{argmax}_{\mathbf{y} \in \mathcal{Y}_S - \{\mathbf{y}_i\}} (1 - (\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}))$$

  where $\mathcal{Y}_S = \{\mathbf{y} | (\mathbf{x}_i, \mathbf{y}) \in S\}\} \subset \mathcal{Y}$ contains the multilabels seen in the training data
- This is in general relatively fast and effective, but requires that the training data covers enough of the relevant output space

Joint features

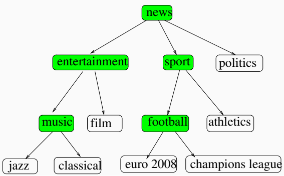
- We assumed so far that the joint feature map is $\phi(\mathbf{x}, \mathbf{y}) = \mathbf{vec}(\mathbf{x}\mathbf{y}^T)$
- However, in general we can first map the inputs and outputs to new spaces using any suitable basis functions, and then compute the joint feature map

  $$\phi(\mathbf{x}, \mathbf{y}) = \mathbf{vec}(\phi_x(\mathbf{x})\phi_y(\mathbf{y})^T) = \phi_x(\mathbf{x}) \otimes \phi_y(\mathbf{x}),$$

  where $\otimes$ denotes the tensor product
- One joint feature for each input-output feature pair $\phi_{x,k}(\mathbf{x})\phi_{y,\ell}(\mathbf{y})$: we can track co-occurring input-output features
- Makes no no prior assumption of which input-output feature pairs might be relevant

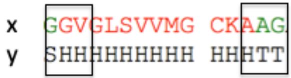
Joint feature map: hierarchical document classification

- $\phi_x(\mathbf{x})$ is the bag of words (word frequencies) of the document
- $\phi_y(\mathbf{y})$ is the vector of edge-label indicators: $\psi_{e,u}(\mathbf{y}) = 1$ if adjacent pair of nodes $e = (i, j)$ is labeled $u \in \{(-1,-1)(-1,+1)(+1,-1),(+1,+1)\}$
- $\phi(\mathbf{x}, \mathbf{y}) \in F_{xy}$ contains counts of a word co-occurring with an adjacent label pair in example $(\mathbf{x}, \mathbf{y})$
- Weights $\mathbf{w}$ are learned to pick up importance input features (words) predictive of an adjacent pair of labels



Joint feature maps: label sequence learning

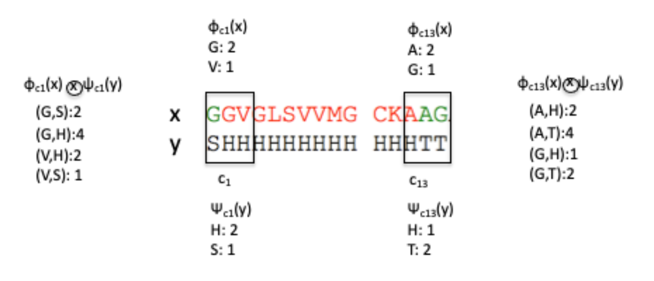
- Assume the task is to predict a label for every symbol in a sequence (e.g. annotating biosequences)
- Usually locality matters: nearby input positions have larger influence to the output than far away ones
- The joint feature map $\phi(\mathbf{x}, \mathbf{y}) = \mathbf{vec}(\phi_x(\mathbf{x})\phi_y(\mathbf{y})^T)$ does not allow directly to represent this
    - It contains every pair of input-output features, irrespective of how far in sequence they are



Joint feature maps: aligned input and output

- We can define a sliding window spanning a few adjacent positions $c = i_{start} \ldots i_{end}$
- Compute a joint feature map over the window $\phi_c(\mathbf{x}, \mathbf{y}) = \phi_{c1}(\mathbf{x}) \otimes \psi_{c1}(\mathbf{y})$
- The joint feature map is computed as the sum of window-specific joint feature maps:

$$\phi(\mathbf{x}, \mathbf{y}) = \sum_{c \in C} \phi_c(\mathbf{x}_c, \mathbf{y}_c),$$



Loss functions for structures

Loss functions for structures

- The multilabel Hinge loss

$$L_{MLHinge}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{w}) = \max_{\mathbf{y} \neq \mathbf{y}_i}(0, 1 - (\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}))$$

  is an convex upper bound for the Zero-one loss:

$$L_{0/1}(\mathbf{y}, \mathbf{y}') = \begin{cases} 1 & \mathbf{y} \neq \mathbf{y}' \\ 0 & \mathbf{y} = \mathbf{y}' \end{cases}$$

- It treats all incorrect multilabels the same
- However, multilabels with only a few incorrect labels might be preferable c than those with many errors
- The most common loss that can represent this kind of preference is the Hamming loss:

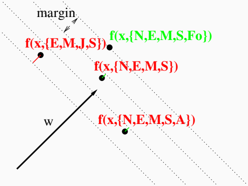
$$L_{Hamming}(\mathbf{y}, \mathbf{y}') = \sum_{k=1}^{K} \mathbf{1}\{y_k \neq y'_k\}$$

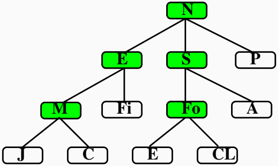
Example: Hierarchical classification

- We can use the Hamming loss within the multilabel Hinge loss by replacing the functional margin 1 with the Hamming loss

$$L_{MLHinge}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{w}) = \max_{\mathbf{y} \neq \mathbf{y}_i}(0, {\color{red} L_{Hamming}(\mathbf{y}, \mathbf{y}')} - (\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}))$$

- The more incorrect the output $\mathbf{y}$, the larger the required margin
- In the joint feature space, the constraints induce a set of hyperplanes, corresponding to different levels of Hamming loss
- The point $\phi(\mathbf{x}_i, \mathbf{y})$ will be constrained to lie in the correct side of the hyperplane that has distance $L_{Hamming}$ from $\phi(\mathbf{x}_i, \mathbf{y}_i)$






BBC News | ENTERTAINMENT | Football pundit accuses Posh

Football pundit accuses Posh

David and Victoria Beckham are permanently in the public eye

BBC football pundit Mark Lawrenson has accused David Beckham and his pop star wife Victoria of 'courting publicity'.

Lawrenson, an analyst on BBC1's Football Focus, spoke out during a discussion about Beckham's sending off in Thursday's World Club Championship match.

Generalizations

The above described methods generalize to other settings (details out of scope of this course):

- Instead of Hamming loss, we can use application dependent loss functions that contain prior information of the severity of errors
- The models can be kernelized for applications where high-dimensional input and output spaces are needed
- The outputs are not restricted to be multilabels but can be general object
- The over all algorithm stays the same
- The representations of inputs and outputs as well as the procedures for finding the outputs with the highest loss typically needs to changed

Summary

- Label ranking can be used for tasks where several labels may be relevant but their preferences differ
- Label ranking can be formulated as a regularized loss minimization problem and solved by stochastic gradient approaches
- Multilabel classification is used for applications where a particular subset is relevant for an input
- Dependency structures between the labels and inputs can be modeled through joint feature maps
- Hamming loss can be used to measure the distance between two label vectors

End of the course

- Last assignment deadline: Tomorrow 2.12.2020 23:59
- Course exam (online in Mycourses): Friday 18.12. at 13:00-16:00. It will be a mixture of essay style and multiple choice questions.
- Answer the anonymous course feedback survey: It will open on December 11 and close December 31. One extra point will be awarded for everybody who answers.