

# Regularization for Multi-Output Learning

Lorenzo Rosasco

9.520

**Goal** In many practical problems, it is convenient to model the object of interest as a function with multiple outputs.

In machine learning, this problem typically goes under the name of multi-task or multi-output learning. We present some concepts and algorithms to solve this kind of problems.

- Examples and Set-up
- Tikhonov regularization for multiple output learning
- Regularizers and Kernels
- Vector Fields
- Multiclass
- Conclusions

# Costumers Modeling

## Costumers Modeling

the goal is to model buying preferences of several people based on previous purchases.

## borrowing strength

People with similar tastes will tend to buy similar items and their buying history is related.

The idea is then to predict the consumer preferences for all individuals **simultaneously** by solving a multi-output learning problem.

Each consumer is modelled as a task and its previous preferences are the corresponding training set.

# Multi-task Learning

We are given  $T$  scalar tasks.

For each task  $j = 1, \dots, T$ , we are given a set of examples

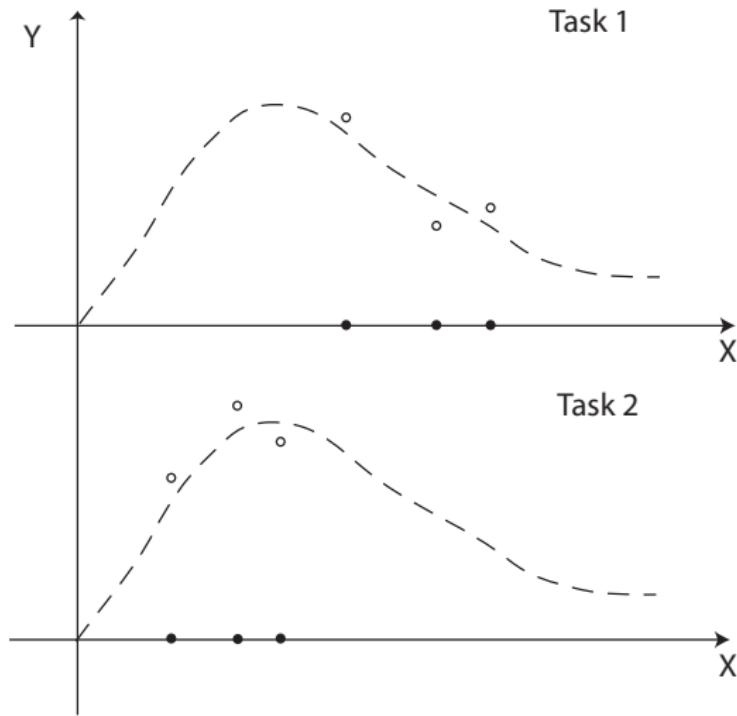
$$S_j = (x_i^j, y_i^j)_{i=1}^{n_j}$$

sampled i.i.d. according to a distribution  $P_j$ .

The goal is to find

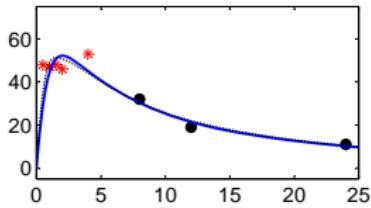
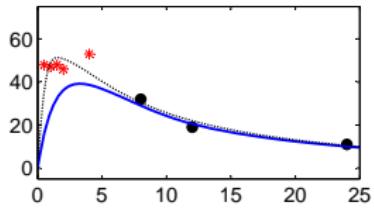
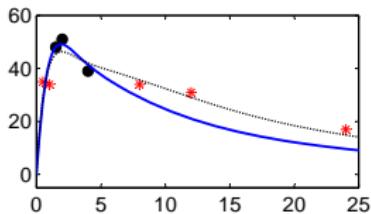
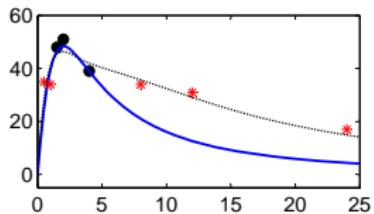
$$f^j(x) \sim y \quad j = 1, \dots, T.$$

# Multi-task Learning



# Pharmacological Data

Blood concentration of a medicine across different times. Each task is a patient.



Single-task

Multi-task

Red dots are test and black dots are training points.

( pics from Pillonetto et al. 08)

# Names and Applications

Related problems:

- conjoint analysis
- transfer learning
- collaborative filtering
- co-kriging

Examples of applications:

- geophysics
- music recommendation (Dinuzzo 08)
- pharmacological data (Pillonetto et al. 08)
- binding data (Jacob et al. 08)
- movies recommendation (Abernethy et al. 08)
- HIV Therapy Screening (Bickel et al. 08)

The framework is very general.

- The input spaces can be different.
- The output space can be different.
- The hypotheses spaces can be different

# How Can We Design an Algorithm?

In all the above problems one can think of improving performances, by exploiting relation among the different outputs.

A possible way to do this is penalized empirical risk minimization

$$\min_{f^1, \dots, f^T} ERR[f^1, \dots, f^T] + \lambda PEN(f^1, \dots, f^T)$$

Typically

- The error term is the sum of the empirical risks.
- The penalty term enforces similarity among the tasks.

We are going to choose the square loss to measure errors.

$$ERR[f^1, \dots, f^T] = \sum_{j=1}^T \frac{1}{n_j} \sum_{i=1}^{n_j} (y_i^j - f^j(x_i^j))^2$$

## MTL

Let  $f^j : X \rightarrow \mathbb{R}$ ,  $j = 1, \dots, T$  then

$$ERR[f^1, \dots, f^T] = \sum_{j=1}^T I_{S_j}[f^j]$$

with

$$I_S[f] = \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

# Building Regularizers

We assume that input, output and hypotheses spaces are the same, i.e.

$$X_j = X,$$

$$Y_j = Y,$$

and

$$\mathcal{H}_j = \mathcal{H},$$

for all  $j = 1, \dots, T$ .

We also assume  $\mathcal{H}$  to be a RKHS with kernel  $K$ .

# Regularizers: Mixed Effect

For each component/task the solution is the same function plus a component/task specific component.

$$PEN(f^1, \dots, f^T) = \lambda \sum_{j=1}^T \|f^j\|_K^2 + \gamma \sum_{j=1}^T \|f^j - \sum_{s=1}^T f^s\|_K^2$$

# Regularizers: Graph Regularization

We can define a regularizer that, in addition to a standard regularization on the single components, forces stronger or weaker similarity through a  $T \times T$  positive weight matrix  $M$ :

$$PEN(f^1, \dots, f^T) = \gamma \sum_{\ell, q=1}^T \|f^\ell - f^q\|_K^2 M_{\ell q} + \lambda \sum_{\ell=1}^T \|f^\ell\|_K^2 M_{\ell\ell}$$

# Regularizers: cluster

The components/tasks are partitioned into  $c$  clusters:  
components in the same cluster should be similar.

Let

- $m_r, r = 1, \dots, c$ , be the cardinality of each cluster,
- $I(r), r = 1, \dots, c$ , be the index set of the components that belong to cluster  $c$ .

$$PEN(f^1, \dots, f^T) = \gamma \sum_{r=1}^c \sum_{l \in I(r)} \|f^l - \bar{f}_r\|_K^2 + \lambda \sum_{r=1}^c m_r \|\bar{f}^r\|_K^2$$

where  $\bar{f}^r, r = 1, \dots, c$ , is the mean in cluster  $c$ .

# How can we find a the solution?

We have to solve

$$\min_{f^1, \dots, f^T} \left\{ \frac{1}{n} \sum_{j=1}^T \sum_{i=1}^n (y_i^j - f^j(x_i))^2 + \lambda \sum_{j=1}^T \|f^j\|_K^2 + \gamma \sum_{j=1}^T \|f^j - \sum_{s=1}^T f^s\|_K^2 \right\}$$

(we considered the first regularizer as an example).

The theory of RKHS gives us a way to do this using what we already know from the scalar case.

# Tikhonov Regularization

We now show that for all the above penalties we can define a suitable RKHS with kernel  $Q$  (and re-index the sums in the error term), so that

$$\min_{f^1, \dots, f^T} \left\{ \sum_{j=1}^T \frac{1}{n_j} \sum_{i=1}^{n_j} (y_i^j - f^j(x_i))^2 + \lambda PEN(f^1, \dots, f^T) \right\}$$

can be written as

$$\min_{f \in \mathcal{H}} \left\{ \frac{1}{n_T} \sum_{i=1}^{n_T} (y_i - f(x_i, t_i))^2 + \lambda \|f\|_Q^2 \right\}$$

# Kernels at Rescue

Consider a (joint) kernel  $Q : (X, \Pi) \times (X, \Pi) \rightarrow \mathbb{R}$ , where  $\Pi = 1, \dots, T$  is the index set of the output components.  
A function in the space is

$$f(x, t) = \sum_i Q((x, t), (x_i, t_i)) c_i,$$

with norm

$$\|f\|_Q^2 = \sum_{i,j} Q((x_j, t_j), (x_i, t_i)) c_i c_j.$$

# A Useful Class of Kernels

Let  $A$  be a  $T \times T$  positive definite matrix and  $K$  a scalar kernel. Consider a kernel  $Q : (X, \Pi) \times (X, \Pi) \rightarrow \mathbb{R}$ , defined by

$$Q((x, t), (x', t')) = K(x, x') A_{t, t'}.$$

Then the norm of a function is

$$\|f\|_Q^2 = \sum_{i,j} K(x_i, x_j) A_{t_i t_j} c_i c_j.$$

# Regularizers and Kernels

If we fix  $t$  then  $f^t(x) = f(t, x)$  is one of the tasks. The norm  $\|\cdot\|_Q$  can be related to the scalar products among the tasks.

$$\|f\|_Q^2 = \sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$$

This implies that :

- A regularizer of the form  $\sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$  defines a kernel  $Q$ .
- The norm induced by a kernel  $Q$  of the form  $K(x, x')A$  can be seen as a regularizer.

The matrix  $A$  encodes relations among outputs.

# Regularizers and Kernels

If we fix  $t$  then  $f^t(x) = f(t, x)$  is one of the tasks. The norm  $\|\cdot\|_Q$  can be related to the scalar products among the tasks.

$$\|f\|_Q^2 = \sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$$

This implies that :

- A regularizer of the form  $\sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$  defines a kernel  $Q$ .
- The norm induced by a kernel  $Q$  of the form  $K(x, x')A$  can be seen as a regularizer.

The matrix  $A$  encodes relations among outputs.

# Regularizers and Kernels

If we fix  $t$  then  $f^t(x) = f(t, x)$  is one of the tasks. The norm  $\|\cdot\|_Q$  can be related to the scalar products among the tasks.

$$\|f\|_Q^2 = \sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$$

This implies that :

- A regularizer of the form  $\sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$  defines a kernel  $Q$ .
- The norm induced by a kernel  $Q$  of the form  $K(x, x')A$  can be seen as a regularizer.

The matrix  $A$  encodes relations among outputs.

# Regularizers and Kernels

If we fix  $t$  then  $f^t(x) = f(t, x)$  is one of the tasks. The norm  $\|\cdot\|_Q$  can be related to the scalar products among the tasks.

$$\|f\|_Q^2 = \sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$$

This implies that :

- A regularizer of the form  $\sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$  defines a kernel  $Q$ .
- The norm induced by a kernel  $Q$  of the form  $K(x, x')A$  can be seen as a regularizer.

The matrix  $A$  encodes relations among outputs.

# Regularizers and Kernels

We sketch the proof of

$$\|f\|_Q^2 = \sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$$

Recall that

$$\|f\|_Q^2 = \sum_{ij} K(x_i, x_j) A_{t_i t_j} c_i c_j$$

and note that if  $f^t(x) = \sum_i K(x, x_i) A_{t, t_i} c_i$ , then

$$\langle f^s, f^t \rangle_K = \sum_{i,j} K(x_i, x_j) A_{s, t_i} A_{t, t_j} c_i c_j.$$

We need to multiply by  $A_{s,t}^{-1}$  (or rather  $A_{s,t}^\dagger$ ) the last equality.

# Regularizers and Kernels

We sketch the proof of

$$\|f\|_Q^2 = \sum_{s,t} A_{s,t}^\dagger \langle f^s, f^t \rangle_K$$

Recall that

$$\|f\|_Q^2 = \sum_{ij} K(x_i, x_j) A_{t_i t_j} c_i c_j$$

and note that if  $f^t(x) = \sum_i K(x, x_i) A_{t, t_i} c_i$ , then

$$\langle f^s, f^t \rangle_K = \sum_{i,j} K(x_i, x_j) A_{s, t_i} A_{t, t_j} c_i c_j.$$

We need to multiply by  $A_{s,t}^{-1}$  (or rather  $A_{s,t}^\dagger$ ) the last equality.

# Examples I

Let  $\mathbf{1}$  be the  $T \times T$  matrix whose entries are all equal to 1 and  $\mathbf{I}$  the  $d$ -dimensional identity matrix.

The kernel

$$Q((x, t)(x', t')) = K(x, x')(\omega \mathbf{1} + (1 - \omega) \mathbf{I})_{t, t'}$$

induces a penalty:

$$A_\omega \left( B_\omega \sum_{\ell=1}^T \|f^\ell\|_K^2 + \omega T \sum_{\ell=1}^T \|f^\ell - \frac{1}{T} \sum_{q=1}^T f^q\|_K^2 \right)$$

where  $A_\omega = \frac{1}{2(1-\omega)(1-\omega+\omega T)}$  and  $B_\omega = (2 - 2\omega + \omega T)$ .

## Examples II

The penalty

$$\frac{1}{2} \sum_{\ell,q=1}^T \|f^\ell - f^q\|_K^2 M_{\ell q} + \sum_{\ell=1}^T \|f^\ell\|_K^2 M_{\ell\ell}$$

can be rewritten as:

$$\sum_{\ell,q=1}^T \langle f^\ell, f^q \rangle_K L_{\ell q}$$

where  $L = D - M$ , with  $D_{\ell q} = \delta_{\ell q} (\sum_{h=1}^T M_{\ell h} + M_{\ell q})$ .

The kernel is  $Q((x, t)(x', t')) = K(x, x') L_{t,t'}^\dagger$ .

## Examples III

The penalty

$$\epsilon_1 \sum_{c=1}^r \sum_{l \in I(c)} \|f^l - \bar{f}_c\|_K^2 + \epsilon_2 \sum_{c=1}^r m_c \|\bar{f}^c\|_K^2$$

induces a kernel  $Q((x, t)(x', t')) = K(x, x')G_{t,t'}^\dagger$  with

$$G_{lq} = \epsilon_1 \delta_{lq} + (\epsilon_2 - \epsilon_1) M_{lq}.$$

The  $T \times T$  matrix  $M$  is such that  $M_{lq} = \frac{1}{m_c}$  if components  $l$  and  $q$  belong to the same cluster  $c$ , and  $m_c$  is its cardinality ( $M_{lq} = 0$  otherwise).

# Tikhonov Regularization

Given the above penalties and re-indexing the sums in the error term

$$\min_{f^1, \dots, f^T} \left\{ \sum_{j=1}^T \frac{1}{n_j} \sum_{i=1}^{n_j} (y_i^j - f^j(x_i))^2 + \lambda PEN(f^1, \dots, f^T) \right\}$$

can be written as

$$\min_{f \in \mathcal{H}} \left\{ \frac{1}{n_T} \sum_{i=1}^{n_T} (y_i - f(x_i, t_i))^2 + \lambda \|f\|_Q^2 \right\}$$

where  $\mathcal{H}$  is the RKHS with kernel  $Q$  and we consider a training set  $(x_1, y_1, t_1), \dots, (x_{n_T}, y_{n_T}, t_{n_T})$  with  $n_T = \sum_{j=1}^T n_j$ .

# Representer Theorem

A representer theorem can be proved using the same technique of the standard case

$$f(x, t) = f^t(x) = \sum_{i=1}^n Q((x, t), (x_i, t_i)) c_i,$$

where the coefficients are given by

$$(\mathbf{Q} + \lambda I)\mathbf{C} = \mathbf{Y}.$$

where  $\mathbf{C} = (c_1, \dots, c_n)^T$ ,  $\mathbf{Q}_{ij} = Q((x_i, t_i), (x_j, t_j))$  and  $\mathbf{Y} = (y_1, \dots, y_n)^T$ .

Note that we can write the empirical risk as,

$$\frac{1}{n_T} \|\mathbf{Y} - \mathbf{QC}\|_{n_T}^2$$

The minimization with gradient descent show that the coefficients can be found by setting  $\mathbf{C}^0 = 0$  and considering for  $i = 1, \dots, t-1$  the following iteration

$$\mathbf{C}^i = \mathbf{C}^{i-1} + \eta(\mathbf{Y} - \mathbf{QC}^{i-1}),$$

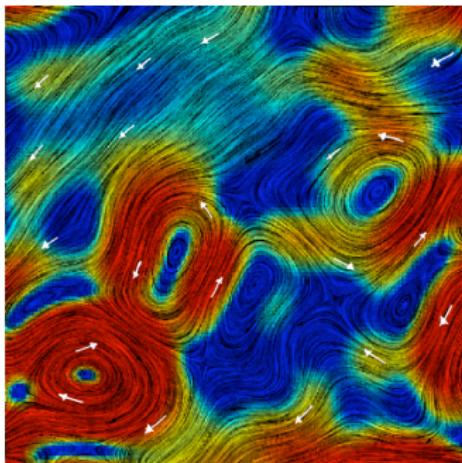
where  $\eta$  the step size.

Regularization can be achieved by early stopping.

- The effect of MTL is especially evident when few examples are available for each task.
- The complexity of Tikhonov regularization can be reduced when some (all) input points are the same (Dinuzzo et al. 09, Baldassarre et al. 09).
- The design of efficient kernel is a considerably more difficult problem than in the scalar case.

# Learning Vector Fields: Example

We sample the velocity fields of an incompressible fluid and want to recover the whole velocity field.



To each point in the space we associate a velocity vector.

(figures from Macêdo and Castro 08)

# Learning Vector fields

It is the most natural extension of the scalar setting.

We are given a training set of points

$S = \{(x_1, y_1), \dots, (x_n, y_n)\}$ , where

- $x_1, \dots, x_n \in \mathbb{R}^p$
- $y_1, \dots, y_n \in \mathbb{R}^T$

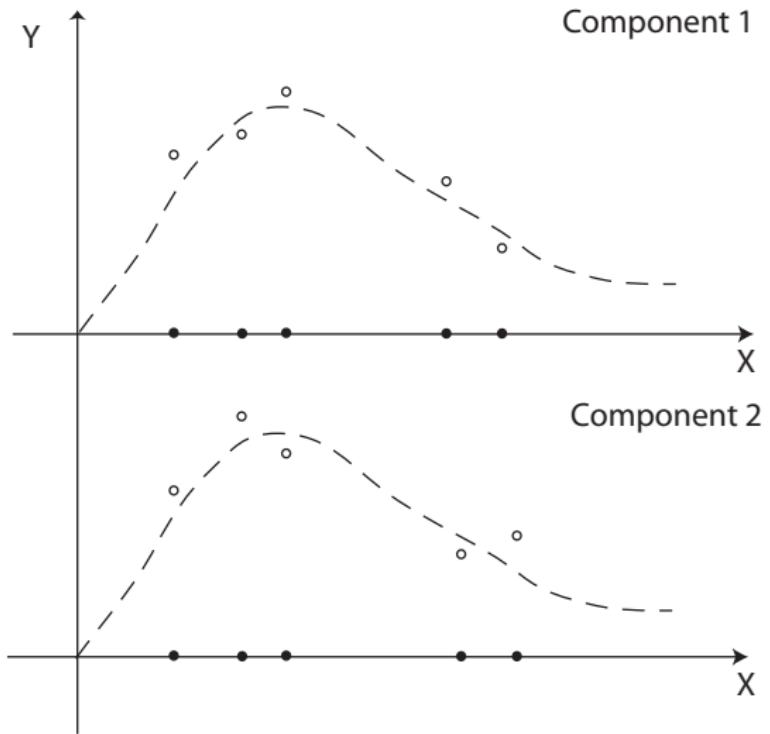
As usual the point are assumed to be sampled (i.i.d.) according to some probability distribution  $P$ .

The goal is to find

$$f(x) \sim y,$$

where  $y$  is a vector.

# Vector fields Learning



# Error Term for Vector fields

Note that

$$ERR[f^1, \dots, f^T] = \frac{1}{n} \sum_{j=1}^T \sum_{i=1}^n (y_i^j - f^j(x_i^j))^2$$

can be written as

VFL

$$ERR[f] = \frac{1}{n} \sum_{i=1}^n \|y_i - f(x_i)\|_T^2, \quad \|y - f(x)\|_T^2 = \sum_{j=1}^T (y^j - f^j(x))^2$$

with  $f : X \rightarrow \mathbb{R}^T$  and  $f = f^1, \dots, f^T$ .

# Error Term for Vector fields

Note that

$$ERR[f^1, \dots, f^T] = \frac{1}{n} \sum_{j=1}^T \sum_{i=1}^n (y_i^j - f^j(x_i^j))^2$$

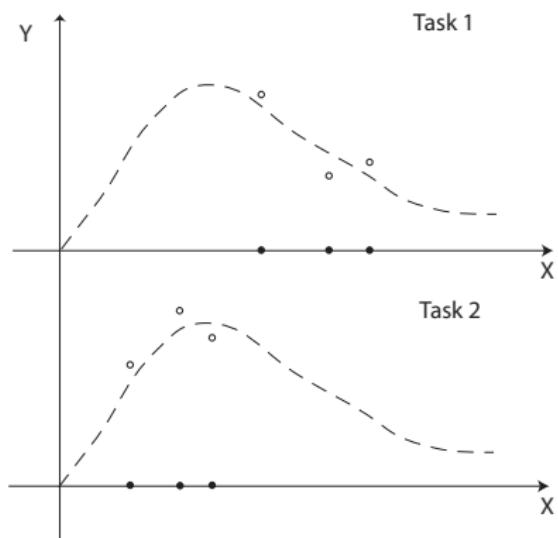
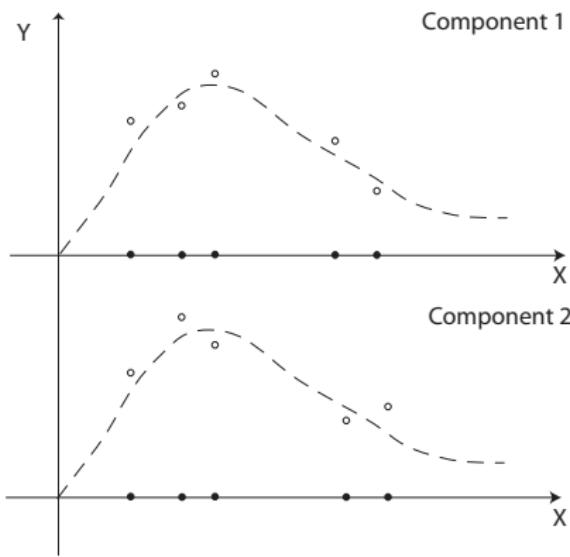
can be written as

VFL

$$ERR[f] = \frac{1}{n} \sum_{i=1}^n \|y_i - f(x_i)\|_T^2, \quad \|y - f(x)\|_T^2 = \sum_{j=1}^T (y^j - f^j(x))^2$$

with  $f : X \rightarrow \mathbb{R}^T$  and  $f = f^1, \dots, f^T$ .

# Vector fields vs Multi-task Learning



The two problems are clearly related.

- Tasks can be seen as components of a vector fields and viceversa
- In multitask we might sample each task in a different way, so that when we consider the tasks together we are essentially augmenting the number of sample available for each individual task.

## Multiclass

In multi-category classification each input can be assigned to one of  $T$  classes. We can think of encoding each class with a vector, for example: class one can be  $(1, 0 \dots, 0)$ , class 2  $(0, 1 \dots, 0)$  etc.

## Multilabel

Images contain at most  $T$  objects each input image is associate to a vector

$$(1, 0, 1 \dots, 0)$$

where 1/0 indicate presence/absence of the an object.

## Multiclass

In multi-category classification each input can be assigned to one of  $T$  classes. We can think of encoding each class with a vector, for example: class one can be  $(1, 0 \dots, 0)$ , class 2  $(0, 1 \dots, 0)$  etc.

## Multilabel

Images contain at most  $T$  objects each input image is associate to a vector

$$(1, 0, 1 \dots, 0)$$

where 1/0 indicate presence/absence of the an object.

# One Versus All

Consider the coding where class 1 is  $(1, -1, \dots, -1)$ , class 2 is  $(-1, 1, \dots, -1)$  ...

One can easily check that the problem

$$\min_{f_1, \dots, f_T} \left\{ \frac{1}{n} \sum_{j=1}^T \sum_{i=1}^n (y_i^j - f^j(x_i))^2 + \lambda \sum_{j=1}^T \|f^j\|_K^2 \right\}$$

is exactly the one versus all scheme with regularized least squares.

# One Versus All

Consider the coding where class 1 is  $(1, -1, \dots, -1)$ , class 2 is  $(-1, 1, \dots, -1)$  ...

One can easily check that the problem

$$\min_{f_1, \dots, f_T} \left\{ \frac{1}{n} \sum_{j=1}^T \sum_{i=1}^n (y_i^j - f^j(x_i))^2 + \lambda \sum_{j=1}^T \|f^j\|_K^2 \right\}$$

is exactly the one versus all scheme with regularized least squares.

Kernel Methods and regularization can be used in many situations when the object of interest is a multi output function.

Kernel/Regularizer choice is crucial

- Sparsity
- Manifold
- ????

# Sparsity Across Tasks

Assume that each task is of the form

$$f^t(x) = \sum_{j=1}^p \phi_j(x) c_j^t$$

where  $\phi_1, \dots, \phi_p$  are the same features for all tasks.

A penalization can be written as

$$\sum_j \|\mathbf{c}_j\|_\tau$$

where  $\mathbf{c}_j = (c_j^1, \dots, c_j^T)$  are the coefficients corresponding to the  $j$ -th feature across the various tasks.

# Sparsity Across Tasks

Assume that each task is of the form

$$f^t(x) = \sum_{j=1}^p \phi_j(x) c_j^t$$

where  $\phi_1, \dots, \phi_p$  are the same features for all tasks.

A penalization can be written as

$$\sum_j \|\mathbf{c}_j\|_T$$

where  $\mathbf{c}_j = (c_j^1, \dots, c_j^T)$  are the coefficients corresponding to the  $j$ -th feature across the various tasks.