Actively querying superset labels using indecision: the k-nn case

Sebastien Destercke, Vu-Linh Nguyen, Mylene Masson

Université de Technologie de Compiègne

Learning with superset labels

Features			Labels
$[0.1, 1.5]$	$\ldots$	0.6	a
0.3	$\ldots$	0.2	$\{a, b\}$
0.3	$\ldots$	$[0.2, 0.5]$	$\{a, b, c\}$

- Partial data can induce uncertainty in learning process.
- May happen in a number of situations :
  - Expert labelling,
  - Using easily accessible information to get label set (e.g., actor list to do facial recognition of TV series pictures),
  - Data collection with sensors of various qualities,
  - Data descriptions using different levels of details (coarsening)

Learning from partial data

Two view points

- adapting classical approaches to learn one optimal model. For instance, defining specific loss functions [T. Cour *et al*, 2011.]

  → Necessary assumptions on the missingness process
- learning (IP) models to get set of optimal models from all completions of partial data. For example, the paper of [E. Hullermeier, 2014].

  → No or few assumptions on the missingness process

This work

◇ We adopt the second view regarding data completions

◇ We wonder about which data to query to make better predictions

Some formalisation

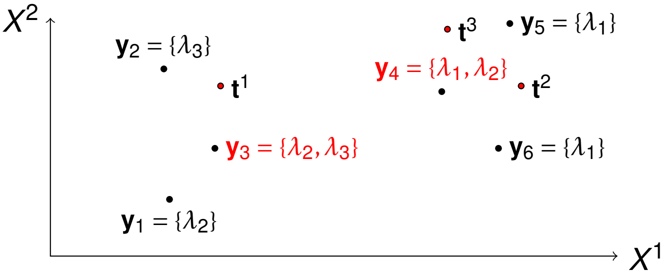
- A training set $D = \{x_i, \mathbf{y}_i\}$ with
  - $x_i \in R^d$ are precise values
  - $\mathbf{y}_i \subseteq \mathscr{Y} = \{\lambda_1, \ldots, \lambda_M\}$ are partial, a.k.a. superset labels
- An evaluation set $T = \{t_i\}$ of input instances, $t_i \in R^d$
- Possibly a decision function $h : D \rightarrow \mathscr{Y}$ providing a precise prediction

Which labels $\mathbf{y}_i$ should we query to improve our model accuracy/decisiveness ?

K-nn classifier for partially labelled data

A simple (maximax) way to take decision despite partial labels [E. Hullermeier & J. Beringer, 2006]

$$h(\mathbf{t}) = \arg\max_{\lambda \in \Omega} \sum_{\mathbf{y}_k \in \mathbf{N_t}} w_k \mathbb{1}_{\lambda \in \mathbf{y}_k}. \quad (1)$$

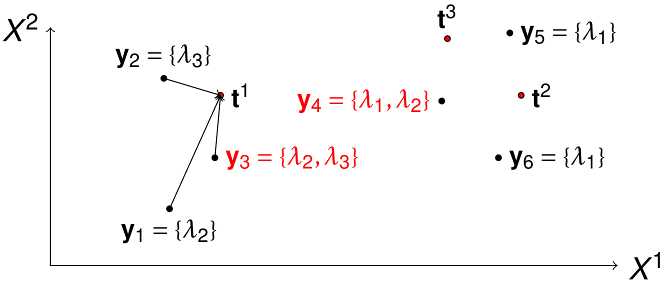


◊ Which partial label is the more informative ?

First idea : the more decision it is involved in, the more informative it is

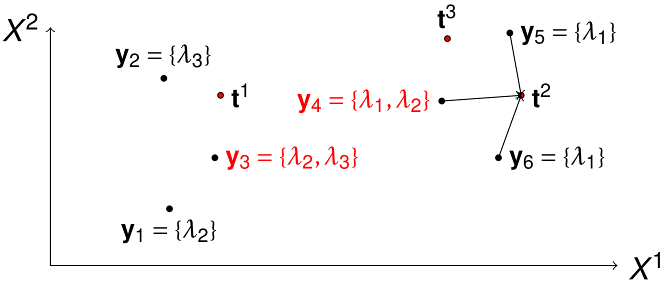
1. define a given number K of neighbours
2. for each x_i with partial label, count the number of items in T of which it is a neighbour
3. query the item $\mathbf{y}_i$ involved in most decisions

First idea : the more decision it is involved in, the more informative it is



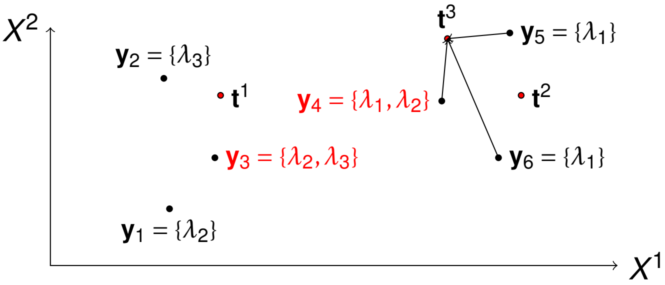
- $\mathbf{y}_3$ involves in decision of $\{\mathbf{t}^1\}$.

First idea : the more decision it is involved in, the more informative it is



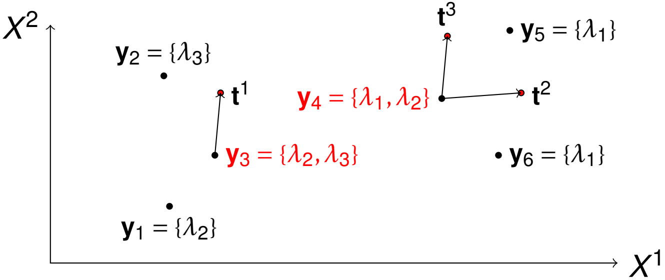
- $\mathbf{y}_3$ involves in decision of $\{\mathbf{t}^1\}$.
- $\mathbf{y}_4$ involves in decisions of $\{\mathbf{t}^2\}$.

First idea : the more decision it is involved in, the more informative it is



- $\mathbf{y}_3$ involves in decision of $\{\mathbf{t}^1\}$.
- $\mathbf{y}_4$ involves in decisions of $\{\mathbf{t}^2, \mathbf{t}^3\}$.

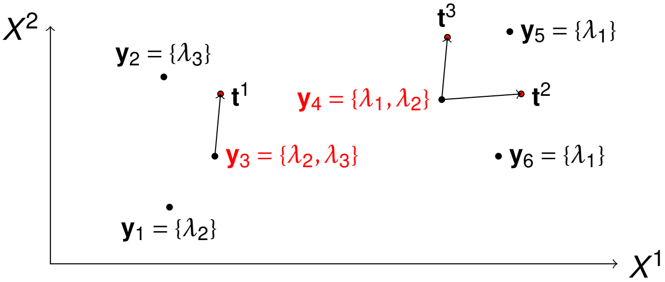
First idea : the more decision it is involved in, the more informative it is



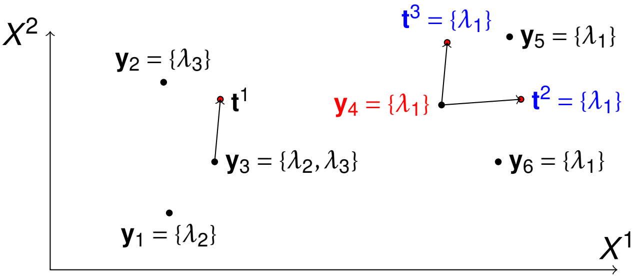
- $\mathbf{y}_3$ involves in decision of $\{\mathbf{t}^1\}$.
- $\mathbf{y}_4$ involves in decisions of $\{\mathbf{t}^2, \mathbf{t}^3\}$.

▷ This strategy chooses $\mathbf{y}_4$ as the optimal query.

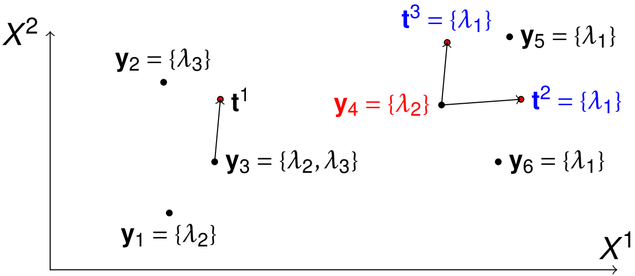
First idea : The more decision it is involved, the more informative it is



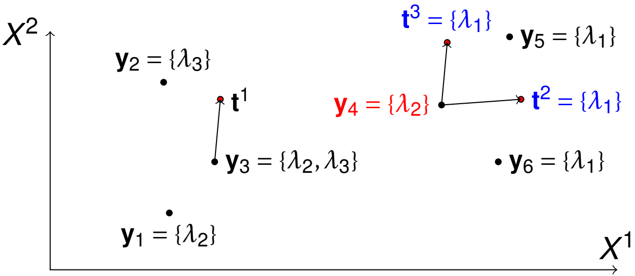
- ◇ It is simple, but is it a good idea ?
  - ▷ What do we gain if we query $\mathbf{y}_4$?



- If $\mathbf{y}_4 = \lambda_1$, then decisions are λ_1.



- If $\mathbf{y}_4 = \lambda_1$, then decisions are λ_1.
- If $\mathbf{y}_4 = \lambda_2$, then decisions are λ_1.



- If $\mathbf{y}_4 = \lambda_1$, then decisions are λ_1.
- If $\mathbf{y}_4 = \lambda_2$, then decisions are λ_1.

  ▷ Querying $\mathbf{y}_4$ does not change predictions.

Second idea : The more uncertainty the label introduces, the more informative it is

- Assume $\mathbf{y}_1, \ldots, \mathbf{y}_K$ are neighbours
- Set $\mathbf{L_t} = \{(l_1^t, \ldots, l_K^t) | l_k^t \in \mathbf{y}_k^t\}$ is the selection of partial labels
- Set of possible predicted labels

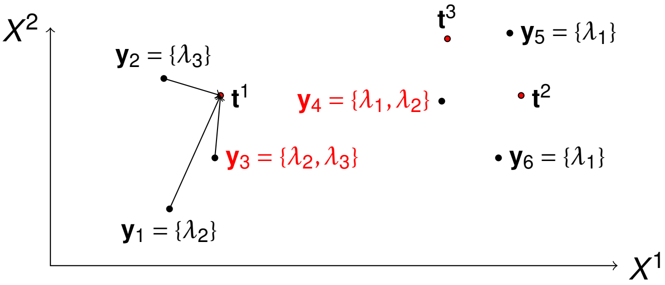
$$\mathbf{PL_t} = \{\lambda \in \Omega | \exists \mathbf{l}^t \in \mathbf{L_t} \text{ s.t } \lambda \in \widehat{\lambda}_{\mathbf{l}^t}\}$$

- Set of necessary predicted labels

$$\mathbf{NL_t} = \{\lambda \in \Omega | \forall \mathbf{l}^t \in \mathbf{L_t}, \lambda \in \widehat{\lambda}_{\mathbf{l}^t}\}$$

An instance t is said to be ambiguous if $\mathbf{PL_t} \neq \mathbf{NL_t}$

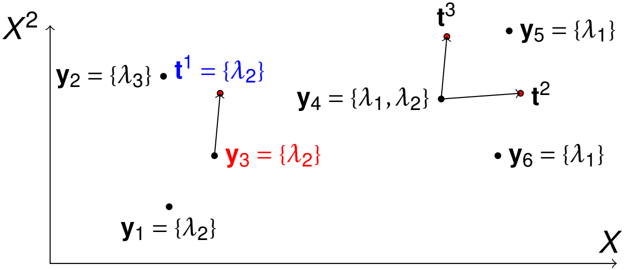
Second idea : The more uncertainty the label introduces, the more informative it is



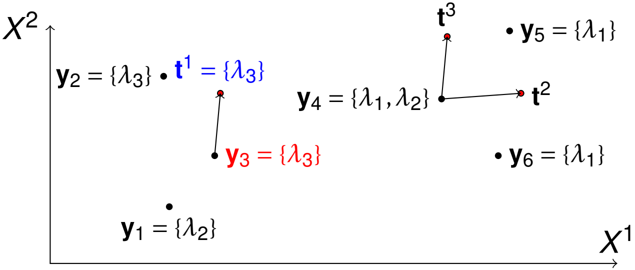
◇ Assuming a weighted 3-nn, $\mathbf{PL}_{\mathbf{t}^1} = \{\lambda_2, \lambda_3\}$, $\mathbf{NL}_{\mathbf{t}^1} = \{\}$

Second idea : The more uncertainty the label introduces, the more informative it is

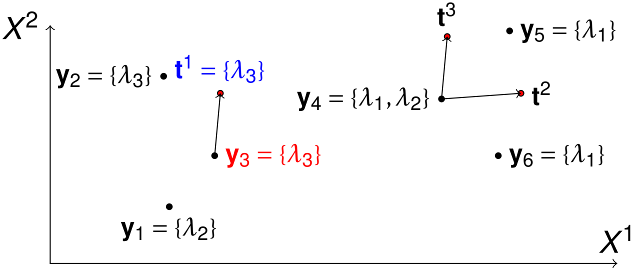
- A data x_i can reduce ambiguity on t if after knowing $\mathbf{y}_i$, $\mathbf{NL_t}$ or $\mathbf{PL_t}$ can potentially change
- For a given K and x_i, count the number of items in $\mathbf{T}$ for which it can reduce ambiguity
- Query the $\mathbf{y}_i$ with greatest potential ambiguity reductions



- If $\mathbf{y}_3 = \lambda_2$, then decision is λ_2.



- If $\mathbf{y}_3 = \lambda_2$, then decision is λ_2.
- If $\mathbf{y}_3 = \lambda_3$, then decision is λ_3.



- If $\mathbf{y}_3 = \lambda_2$, then decision is λ_2.
- If $\mathbf{y}_3 = \lambda_3$, then decision is λ_3.

$\diamondsuit$ Querying $\mathbf{y}_3$ can reduce ambiguity, while $\mathbf{y}_4$ does not.

$\triangleright$ This strategy choose $\mathbf{y}_3$ as the optimal query.

Computational considerations

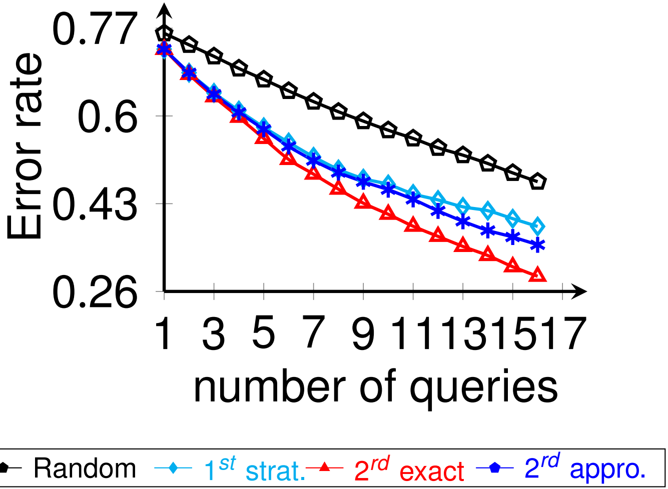
Link with computational social choice

- Computing **PL**, **NL** : equivalent to possible/necessary winner in plurality voting with partial voter preferences
- Querying a label $\mathbf{y}_i$: eliciting precise preferences of a voter

Computational issues

- Computing **NL** : can be done in linear time
- Computing **PL** in unweighted case : can be done in cubic time (reducible to maximum flow problem)
- Computing **PL** in weighted case : seems NP-hard (reducible to 3-dimensional matching), but dynamical programming maybe possible, otherwise approximate

Experimental results illustrated on one data set



Other related questions

- Each query provides information about the ”imprecisiation“ process → can we use it to improve our results
- In practice, optimal model may be identifiable from partial information → under which conditions ?
- Can the idea be efficiently extended to other simple settings (learning NCC, Classification trees) or more complex ones (learning preferences, dynamical models)