# A Preference Optimization Based Unifying Framework for Supervised Learning Problems

**Fabio Aiolli and Alessandro Sperduti**

**Abstract** Supervised learning is characterized by a broad spectrum of learning problems, often involving structured types of prediction, including classification, ranking-based predictions (label and instance ranking), and (ordinal) regression in its various forms. All these different learning problems are typically addressed by specific algorithmic solutions.

In this chapter, we propose a general preference learning model (GPLM), which gives an easy way to translate any supervised learning problem and the associated cost functions into sets of preferences to learn from. A large margin principled approach to solve this problem is also proposed.

Examples of how the proposed framework has been effectively used by us to address non-standard real-world applications are reported showing the flexibility and effectiveness of the approach.

## 1 Introduction

Supervised learning is probably the most commonly used learning paradigm and a large spectrum of learning algorithms have been devised for different learning tasks in the last decades. The need for such a large spectrum of learning algorithms is, in part, due to the many real-world learning problems, that are characterized by heterogeneous tasks and problem-specific learning algorithms for their solution. These include classification and regression problems (including multilabel and multiclass classification, and multivariate regression), as well as ranking-based (either label or instance ranking) and ordinal regression problems. Typically, the approach followed to deal with a nonstandard problem is to map it into a series of simpler, well-known problems and then to combine the resulting predictions. Often, however, this type

F. Aiolli (✉) and A. Sperduti
Department of Pure and Applied Mathematics - Padova - Italy, Via Trieste 63, 35131 Padova, Italy
e-mail: aiolli@math.unipd.it, sperduti@math.unipd.it

J. Fürnkranz and E. Hüllermeier (eds.), *Preference Learning*,
DOI 10.1007/978-3-642-14125-6_2, 

of methodology lacks a principled theory supporting it and/or requires too much computational resources to be practical for real-world applications.

In this chapter, we give a survey of a quite general framework, which is able to generalize different types of supervised learning settings into a common preference optimization task. In particular, this is done by considering supervision as a set of order preferences over the predictions of the learner. More generally, we show that supervised learning problems can be characterized by considering two main dimensions, the type of prediction and the type of supervision involved in the problem to be solved. Then, based on this characterization, we are able to map any of these learning problems into a simple preference learning task. From a practical point of view, we show how all these supervised tasks can also be addressed in a simple linear setting, where any problem formulation can be transformed into a binary problem defined on an augmented space, thus allowing the exploitation of very simple optimization procedures available for the binary case. We also stress the flexibility of the preference model, which allows a user to optimize the parameters on the basis of a proper evaluation function. In fact, while in general the goal of a problem in terms of its evaluation function is clear, a crucial issue in the design of a learning algorithm is how to get a theoretical guarantee that the defined learning procedure actually minimizes the target cost function. One advantage of the framework reviewed in this chapter is that it defines a very natural and uniform way to devise and code a cost function into a learning algorithm.

Examples of real-world applications are then discussed. In particular, two recent applications are discussed in more detail. The first application concerns the problem to select the best candidate for a job role. This is an instance ranking problem, where, however, only binary supervision from the past history is available. The second application concerns a patent classification task, where patent applications have to be associated with primary categories as well as secondary categories. This is an example of a label ranking task, which cannot be properly addressed by an ordinal regression approach.

In Sect. 2, we review the general preference learning model (GPLM). Specifically, we show how the preference model generalizes the supervised learning setting by considering supervision as a partial order of (soft) constraints over the learner predictions. In addition, we show (Sect. 2.2) how the suggested generalization can be instantiated to well-known supervised learning problems. In the same section, we also discuss how cost functions for learning problems can be cast by using preferences (Sect. 2.3) and a simple linear model for the learner (Sect. 2.4). Quite general optimization procedures for training models within the proposed framework are also presented (Sect. 2.5). In Sect. 3, different application scenarios are described and discussed. In particular, it is discussed how the GPLM applies to a job candidate selection task and to a patent classification task. In Sect. 4, related works are sketched and a discussion about the proposed approach is given. Finally, in Sect. 5, some future extensions to the preference framework are suggested and final conclusions are drawn.

## 2 GPLM: A General Model for Supervised Learning

### 2.1 The Learning Domain

Let us consider a very general domain with a space of instances $\mathcal{X}$ and a space of class labels $\mathcal{Y}$. For example, this could be the domain of a recommender system where instances might correspond to customers while labels to products, or the domain of an information retrieval system where instances could correspond to documents while labels to queries.

The basic idea underpinning our general preference learning model is that we want to learn the set of parameters of a real valued relevance (or scoring) function defined on instance label pairs

$$f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R},$$

which should approximate the actual *target* function. In a recommender system task, for example, this target function would represent the actual rating (a real value) a customer would give to a given product. Similarly, in the information retrieval example, the target function could represent the log-ratio of the probability of relevance of a document given a query.

We can easily note that, once such a scoring function is computed, a predictor will be able to order instances in $\mathcal{X}$ based on their relevance once any label $y \in \mathcal{Y}$ is selected, and similarly, to order class labels in $\mathcal{Y}$ based on their relevance once any instance $x \in \mathcal{X}$ is selected.

### 2.2 Prediction and Supervision

In supervised learning, supervision is assumed to be provided according to an unknown probability distribution $\mathcal{D}$ over pairs, where the first member is a description of a domain object (instance) and the second member is the corresponding expected prediction (target label). We generalize this setting by considering supervision as (soft) constraints over the learner predictions, that is constraints whose violation entails a cost, or penalty, for the solution. Specifically, we assume a learner makes its predictions on the basis of a set of parameters $\Theta$, characterizing its *hypothesis space*. Each supervision constraint $S$, that cannot be satisfied makes the learner suffer a cost $c(S|\Theta)$. It is easy to notice that this generalizes the above-mentioned case of supervision as instance-label pairs. In fact, this is obtained back when a unitary cost is given to hypotheses generating incorrect labeling.

Now, we are able to show that, by using the setting presented above, it is possible to cast the main types of supervised learning tasks into a taxonomy on the basis of their expected prediction and supervision feedback. To this end, let us first recall the definition of order relations.

### 2.2.1 Definition

A *partial order* is a pair $(\mathcal{P}, \succeq)$ in which $\mathcal{P}$ is a set and $\succeq$ is a reflexive, antisymmetric, and transitive binary relation. A *partial ranking* of length $r$ is a partial order in which the set $\mathcal{P}$ can be partitioned in $r$ sets $\mathcal{P}_1, \ldots, \mathcal{P}_r$ such that $z \in \mathcal{P}_i, z' \in \mathcal{P}_j$, $i < j$, implies $z \succeq z'$ and no further information is conveyed about the ordering within subsets $\mathcal{P}_k$. A *full order* on $\mathcal{P}$ is defined as a partial ranking of length $|\mathcal{P}|$. We denote by $PO(\mathcal{P})$, $PR(\mathcal{P})$, and $FO(\mathcal{P})$ the set of partial orders, partial rankings, and full orders over the set $\mathcal{P}$, respectively.

### 2.2.2 Label Rankings as Qualitative Preferences

A first important family of supervised learning tasks is related to the ordering of the classes on the basis of their relevance for an instance, and thus they are characterized by the fact that predictions should be based on a full order over the labels. This family of problems is referred to as *label rankings*. Supervision is in the form of partial orders over the classes. In our notation, we have supervision $S \in PO(\mathcal{Y})$ and predictions in $FO(\mathcal{Y})$. Different settings can be obtained corresponding to different types of supervision. A few well-known instances are listed in the following:

Category Ranking (CR)

In this setting, the goal is to order categories on the basis of their relevance for an instance. As an example, in a collaborative filtering setting, users could correspond to our instances and the different movies to our classes. Then, one could be interested in the ordering (by relevance) of the set of movies based on user preferences. This is trivially a particular case of label ranking where supervision is given as full orders over $\mathcal{Y}$.

Bipartite Category Ranking (BCR)

In this task, supervision is given as two groups of classes and it is required to predict full orders in which the first group of classes is ranked over the second. As a leading example, in information retrieval, given a document, one might have to rank the available topics with the aim to return the most relevant topics on the top of the list. This is again a specific case of label ranking where supervision is given as partial rankings of length two. This task has been also referred to as category ranking in literature [10]. Here a different terminology is adopted to avoid confusion between these two different tasks.[1]

[1] Note that this task and the two that follow are conceptually different from the task to decide about the membership of an instance. Here, supervision only gives *qualitative* information about the fact that some classes are more relevant than others.

We might also be interested in predictions consisting of the most relevant classes, that is, of a prefix of the full order induced by the relevance function $f(x, y)$. This family of tasks is commonly referred to as *classification* problems. They can, however, be considered as subcases of the BCR ranking task. A few examples of this kind of problems, listed by increasing specificity, is given here:

$Q$-Label Classification (QC)

In this task, the goal is to select the $Q$ most appropriate classes for a given instance, with $Q$ fixed. The supervision here is a partial ranking of length two where a set of exactly $Q$ labels are preferred over the rest.

Single-Label Classification (SC)

In this well-known classification task, the goal is to select exactly one class (the most relevant) for an instance. This is a trivial subcase of QC with $Q = 1$.

### 2.2.3 Instance Rankings as Qualitative Preferences

Another interesting family of tasks is *instance rankings*, where the goal is to order instances on the basis of the relevance of a given class. In our notation, predictions are in $FO(\mathcal{X})$ and supervision is given in the form $S \in PO(\mathcal{X})$.

The duality with respect to label rankings is self-evident. In principle, a corresponding problem setting could be defined for each of the label ranking settings. We can easily see that the well-known *(Bipartite) Instance Ranking* (IR) task, corresponds to BCR and is the one to induce an order such that a given set of instances is top-ranked. A natural application of this kind of prediction is in information retrieval, e.g., when listing the results returned by a search engine. Another interesting application is the one presented in Sect. 3 for job role selections. As in BCR, here supervision consists of partial rankings (this time over the set $\mathcal{X}$) of length two. Another task, which can also be considered in this family, is learning preference relations from a given set of ranked instances. For example, in information retrieval, the task to learn preference relations on the basis of basic preferences given as pairs of documents [19].

The two families of tasks above can be considered *qualitative tasks* since they are concerned with order relations between instance-class pairs. On the other side, *quantitative tasks* are the ones that are more concerned with the absolute values of the relevance of instance-class pairs.

### 2.2.4 Quantitative Predictions

Sometimes there is the necessity to do quantitative predictions about data at hand. For example, in binary classification, one has to decide about the membership of

an instance to a class as opposed to rank instances by relevance. These settings are not directly subsumed by the settings presented above. As we will see, this can be overcome by adding a set of thresholds and doing predictions based on these thresholds.

Multivariate *Ordinal Regression* (MOR)

There are many settings where it is natural to rate instances according to an ordinal scale, including collaborative filtering, where there is the need to predict people ratings on unseen items. Borrowing the movie-related application introduced above, suitable rates for movies could be given as "bad", "fair", "good", and "recommended". With no loss in generality, we can consider the target space as the integer set $\mathcal{Z} = \{0, \ldots, R-1\}$ of $R$ available rates. Following an approach similar to the one in [26], rates are made corresponding to intervals of the real line. Specifically, a set of thresholds $T = \{\tau_0 = -\infty, \tau_1, \ldots, \tau_{R-1}, \tau_R = +\infty\}$ can be defined and the prediction based on the rule

$$\hat{z} = \{\mathrm{i} : f(\mathbf{x}, y) \in (\tau_i, \tau_{i+1})\}.$$

In a typical (*instance-pivoted*) version of the MOR problem, given the target rate $z_y$ w.r.t. the label $y$, a correct prediction will be consistent with the conditions: $f(\mathbf{x}, y) > \tau_i$ when $i \leq z_y$ and $f(\mathbf{x}, y) < \tau_i$ when $i > z_y$. Note that, a different threshold set could also be used for different labels. The well-known *(Univariate) Ordinal Regression*(OR) [20, 31] task is a trivial subcase of MOR when a single class is available. A dual (*label-pivoted*) version of the MOR problem is also possible which can raise when one has to rate classes according to an ordinal scale, and the instance is fixed in this case. An example of this situation is given in Sect. 3.2.

Multilabel Classification (MLC)

In this task, it is required to classify instances with a subset (the cardinality of which is not specified) of the available classes. For us, it is convenient to consider this task as an MOR problem, where only two ranks are available, relevant and irrelevant, and $\mathcal{Z} = \{0, 1\}$. The well-known *Binary Classification* (BC) can be considered a subcase of OR with two ranks $\mathcal{Z} = \{0, 1\}$. Note that this task is considered here conceptually different from SC with two classes. An alternative way to look at the multilabel problem is to add an artificial label, which is always considered less relevant than relevant labels and more relevant than irrelevant labels. In this way, supervision of the same type as for label ranking problems can be given. This approach, named *Calibrated Label Ranking*, has been recently proposed in [17].

Clearly, the taxonomy presented above is not exhaustive but well highlights how many different kinds of structured predictions can be seen as simple constraints over the predictions of a learner. Specifically, they consist of constraints in conjunctive

**Table 1** Supervision of problems in Sect. 2.2. Label and instance rankings (LR and IR, respectively) have a preference for each order relation induced by the supervision $S$. In ordinal regression (MOR), a preference is associated with each threshold and $z \in \mathcal{Z}$ is the rank given by the supervision

| Setting | Supervision P-sets |
|---|---|
| LR | $\{(\mathbf{x}, y_r) \succ (\mathbf{x}, y_s)\}_{(\mathbf{x},y_r)\succeq_S(\mathbf{x},y_s)}$ |
| IR | $\{(\mathbf{x}_i, y) \succ (\mathbf{x}_j, y)\}_{(\mathbf{x}_i,y)\succeq_S(\mathbf{x}_j,y)}$ |
| MOR | $\{(\mathbf{x}, y) \succ \tau_i\}_{i<z} \cup \{\tau_i \succ (\mathbf{x}, y)\}_{i\geq z}$ |

form where each basic preference is defined over the scoring values and/or a set of threshold values. In particular, we can differentiate between two types of order preferences: *qualitative* preferences in the form

$$(\mathbf{x}_i, y_r) \succ (\mathbf{x}_j, y_s)$$

telling that the value of $f(\mathbf{x}_i, y_r)$ should be higher than the value of $f(\mathbf{x}_j, y_s)$, and *quantitative* preferences in the form

$$(\mathbf{x}, y) \succ \tau \text{ or } \tau \succ (\mathbf{x}, y), \tau \in \mathbb{R}$$

relating the value of $f(\mathbf{x}, y)$ to a given threshold $\tau$. In Table 1, a summary of supervision obtained for the most general settings are presented. Particular instantiations to more specific problems are immediate.

## 2.3 Definition of the Preference Problem

We have seen how supervision of typical supervised learning problems can be decomposed in terms of sets of qualitative and/or quantitative preferences over the scoring function of a learner. Here, we show that preferences also give us a flexible way to express cost functions, which can be directly utilized to optimize a learner. Specifically, we consider preference graphs, i.e., directed graphs where nodes take values on the set $\mathcal{H} \equiv (\mathcal{X} \times \mathcal{Y}) \cup \mathcal{T}$ and edges $(h_1, h_2) \in \mathcal{H} \times \mathcal{H}$ represent preferences $h_1 \succ h_2$. We say that a scoring function is consistent with a preference graph whenever it is consistent with all the preferences in the graph. The evaluation of any scoring function can then be performed by checking for how many graphs the scoring function is not consistent with.

Even more general cost functions can be obtained by associating weights (or costs) with the edges of the graphs. In this case, given a preference graph, the cost incurred by a hypothesis is defined as the maximum cost of its unfulfilled preferences (edges). When not explicitly indicated, we assume the weight associated with an edge to be 1. Summarizing, the total cost suffered by a scoring function $f$ for supervision $S$, which is given as a set of preference graphs $G$, is defined as the

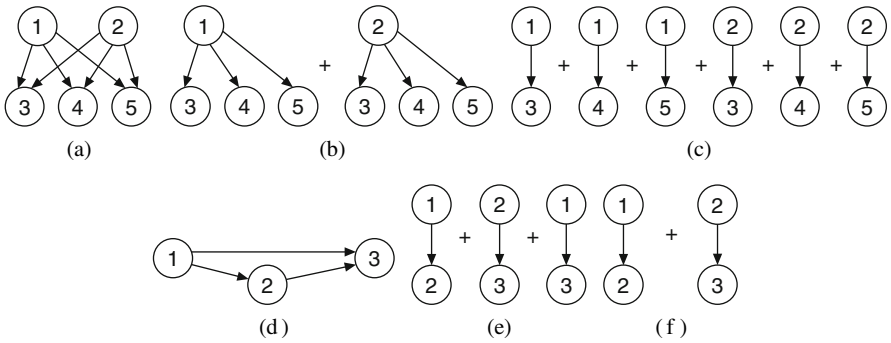


**Fig. 1** Examples of label mappings for 2-label classification (**a–c**) and ranking (**d–f**)

cumulative cost over all the preference graphs. More formally, we have

$$c(G|f) = \sum_{g \in G} c(g|f) \quad \text{and} \quad c(g|f) = \max\{\gamma(\lambda) | \lambda \in E(g) \text{ not fulfilled by } f\},$$

where $\gamma(\lambda)$ represents the weight associated with the preference $\lambda$ and $E(g)$ is the set of edges in $g$.

### Cost Functions for Label Rankings

Many different cost functions which can be, useful for label ranking problems can be easily reproduced in this way. The reader can see [2, 11] for several examples on how to map this kind of supervision into sets of preference graphs. In Fig. 1, there are graphically presented very simple examples of how supervision for a 2-label classification and a ranking problem can be differently mapped into preference graphs thus obtaining different preference optimization problems. Note that only the labels and not the instances, which are fixed in this case, are indicated into the nodes.

The map in Fig. 1a defines a cost function indicating *if* any of the relevant labels are wrongly classified as irrelevant. The map in Fig. 1b would define a different cost function counting *how many* relevant labels are wrongly classified as irrelevant. Finally, the map in Fig. 1c would give the so-called *ranking loss*, i.e., the number of pairs that are not correctly ordered.

Allowing edges with different weights gives further flexibility to the model. A typical example where costs associated with edges can turn out to be useful is in the classification setting, where misclassifications can have different costs. This can be the case in single-label classification when categories are not represented with the same frequencies in the training and the test set. Another interesting case is when there is some structure between the available classes and a different metric for misclassification costs should be introduced. For example, in hierarchical classification, it makes sense to pay costs proportional to the path length in the tree between

the true class and the predicted one. In all these cases, a cost matrix $\Delta$ is used to have a better control over the learning algorithm, where the element $\Delta(y_r, y_s)$ represents the cost of classifying a pattern as $y_r$ when it is actually in $y_s$.

### Cost Functions for Instance Rankings

Concerning the instance ranking setting and BCR-like predictions, a common cost function used in many different domains including information retrieval is the so-called AUC (Area under ROC curve) measure. It can be shown that this is proportional to the number of instance pairs incorrectly ordered and thus it can be trivially represented in our model by using a mapping similar to the one given in Fig. 1c.

### Cost Functions for Prediction of Ratings

A natural definition of a cost function for ordinal regression problems is $c = |\hat{z}(\mathbf{x}) - z(\mathbf{x})|$, where $\hat{z}(\mathbf{x})$ is the rate given as output by the hypothesis and $z(\mathbf{x})$ the correct rate. In this setting, we assume that the evaluation function is somewhat proportional to the distance between ordered rates. Two different maps can be defined to use GPLM for the solution of an ordinal regression problem which are able to mimic this cost function. The easiest way is to consider the number of thresholds that are not correctly ordered w.r.t. $f(\mathbf{x}, y)$. This can be obtained in our framework by mapping this kind of supervision into $R - 1$ graphs where each graph consists of a single preference of type $(\mathbf{x}, y) \succ \tau_r$, whenever $r \leq z(\mathbf{x})$, and $(\mathbf{x}, y) \succ \tau_r$, otherwise. A second way is by using costs associated with different preferences, i.e., the $r$th preference is set to $((\mathbf{x}, y) \succ \tau_r)_{z-i+r}$ whenever $r \leq z$, and $(\tau_r \succ (\mathbf{x}, y))_{r-z}$, otherwise. Note that, with this last, we have a greater flexibility on the definition of the cost function. For example, it can be used when the above assumptions about the distance between different rates are not appropriate for the task at hand.

As an example of application of the second model, consider a $R = 4$ univariate ordinal regression problem. Then, we have three thresholds $T = \{\tau_1, \tau_2, \tau_3\}$ and cost mappings defined as in the following:

$$\begin{aligned}
\mathcal{G}(r = 0) &= \{(\tau_1 \succ (\mathbf{x}, y))_1, (\tau_2 \succ (\mathbf{x}, y))_2, (\tau_3 \succ (\mathbf{x}, y))_3\} \\
\mathcal{G}(r = 1) &= \{((\mathbf{x}, y) \succ \tau_1)_1, (\tau_2 \succ (\mathbf{x}, y))_1, (\tau_3 \succ (\mathbf{x}, y))_2\} \\
\mathcal{G}(r = 2) &= \{((\mathbf{x}, y) \succ \tau_1)_2, ((\mathbf{x}, y) \succ \tau_2)_1, (\tau_3 \succ (\mathbf{x}, y))_1\} \\
\mathcal{G}(r = 3) &= \{((\mathbf{x}, y) \succ \tau_1)_3, ((\mathbf{x}, y) \succ \tau_2)_2, ((\mathbf{x}, y) \succ \tau_3)_1\}
\end{aligned}$$

It is easy to verify that this mapping respects the costs as they could be obtained by the natural cost definition given above. For example, considering the instance $\mathbf{x}$ with target rate 1 being rated 3. Then, it means that the scoring function is such that $f(\mathbf{x}, y) \in (\tau_3, +\infty)$, i.e.,

$$-\infty \leq \tau_1 \leq \tau_2 \leq \tau_3 \leq f(\mathbf{x}, y) \leq +\infty,$$

and hence the cost suffered by the hypothesis is correctly computed by

$$c(r = 1) = \max\{0, +1, +2\} = +2.$$

Trivial extensions of these maps which are suitable for the multivariate ordinal regression problem can also be defined but they are omitted here.

Multilabel Classification

The standard evaluation measure for multilabel classification, the so-called Hamming loss, is defined by the number of incorrect decisions of the classifier. For the instance $\mathbf{x}$, missing one of the target classes $Y(\mathbf{x}) \subseteq \mathcal{Y}$ causes an algorithm to incur in a loss smaller than when missing more target classes. This seems quite natural in many real-world situations. In our setting, this cost function directly derives from the one defined for MOR problems when considering only two rates $\{\text{irrelevant} = 0, \text{relevant} = 1\}$. Finally, cost functions for BC problems can be obtained as a trivial subcase when a single class is available.

## *2.4 A Linear Embedding for Preference Optimization*

In this section, we show that, using a linear form of the scoring function, the preference optimization problems defined by our framework become very simple. Consider a simple form of the relevance function, that is

$$f(\mathbf{x}, y) = w \cdot \phi(\mathbf{x}, y),$$

where $\phi(\mathbf{x}, y) \in \mathbb{R}^d$ is a joint representation of instance-class pairs and $w \in \mathbb{R}^d$ is a weight vector [30]. Note that this form generalizes the more standard form $f(\mathbf{x}, y) = w_y \cdot \phi(\mathbf{x})$, where different weight vectors are associated with different labels. In fact, let $|\mathcal{Y}| = m$, we can write:

$$w = (w_1, \ldots, w_m) \text{ and } \phi(\mathbf{x}, y) = (\underbrace{\mathbf{0}, \ldots, \mathbf{0}}_{y-1}, \phi(\mathbf{x}), \underbrace{\mathbf{0}, \ldots, \mathbf{0}}_{m-y}).$$

With this assumption, it is possible to conveniently reformulate an order constraint as a linear constraint. Let $T = \{\tau_1, \ldots, \tau_{R-1}\}$ be the set of available thresholds, then in the qualitative case, given $\lambda \equiv (\mathbf{x}_i, y_r) \succ (\mathbf{x}_j, y_s)$, we obtain

$$f(\mathbf{x}_i, y_r) > f(\mathbf{x}_j, y_s) \Leftrightarrow \underbrace{(w, \tau_1, \ldots, \tau_{R-1}) \cdot (\phi(\mathbf{x}_i, y_r) - \phi(\mathbf{x}_j, y_s), \underbrace{0, \ldots, 0}_{R-1})}_{\psi(\lambda)} > 0$$

while, in the quantitative case when either $\lambda \equiv (\mathbf{x}, y) \succ \tau_r$ or $\lambda \equiv \tau_r \succ (\mathbf{x}, y)$, and using a suitable $\delta \in \{-1, +1\}$ for shortness, we have

$$\delta(f(\mathbf{x}, y) - \tau_r) > 0 \Leftrightarrow (w, \tau_1, \ldots, \tau_{R-1}) \cdot \underbrace{(\delta\phi(\mathbf{x}, y), \underbrace{0, \ldots, 0}_{r-1}, -\delta, \underbrace{0, \ldots, 0}_{R-r-1})}_{\psi(\lambda)} > 0.$$

In general we can see that supervision constraints of all the above-mentioned problems, can be reduced to sets of linear constraints of the form $\mathbf{w} \cdot \psi(\lambda) > 0$, where $\mathbf{w} = (w, \tau_1, \ldots, \tau_{R-1})$ is the vector of weights augmented with the set of available thresholds, and $\psi(\lambda)$ is a suitable representation of the preference under consideration. The quantity

$$\rho_A(\lambda|\mathbf{w}) = \mathbf{w} \cdot \psi(\lambda)$$

will be also referred to as the margin of the hypothesis w.r.t. the preference. Note that this value is greater than zero when the preference is satisfied and less than zero otherwise. We will say that a preference $\lambda$ is *consistent* with an hypothesis when $\rho_A(\lambda|\mathbf{w}) > 0$. Similarly, for a preference graph $g$, which represents a conjunction of simple preferences, it is required that $\rho_A(\lambda|\mathbf{w}) > 0$ for all $\lambda \in E(g)$. The margin of an hypothesis w.r.t. the whole preference graph $g$ can be consequently defined as the minimum of the margins of preferences contained in g, i.e.,

$$\rho(g|\mathbf{w}) = \min_{\lambda \in E(g)} \rho_A(\lambda|\mathbf{w}).$$

Summarizing, all the problems defined in the taxonomy in Sect. 2.2 can be seen as an homogeneous linear problem in a opportune augmented space. Specifically, any algorithm for linear classification (e.g., perceptron or linear programming) can be used to solve it, provided the problem has a solution.

## 2.5 Learning with Preferences

In earlier sections, we have discussed the structure behind the supervision, how cost functions can be modeled using preference graphs, and how preferences can be linearly embedded by using a linear form for the scoring function. Now, we see how to give learning algorithms that are able to optimize these kind of preference optimization problems.

The goal in a batch learning algorithm is to optimize the parameters $\mathbf{w}$ so as to minimize the expected cost over $\mathcal{D}$, the actual distribution ruling the supervision feedback. More formally, the following has to be minimized

$$R[\mathbf{w}] = \mathbb{E}_{S \sim \mathcal{D}}[c(S|\mathbf{w})].$$

**Table 2** Examples of approximation losses as a function of the margin. $\beta > 0, \theta \in \mathbb{R}$ are intended to be external parameters

| Methods | $l(\rho)$ |
|---|---|
| Perceptron | $\max(0, -\rho)$ |
| $\beta$-margin | $\max(0, \beta - \rho)$ |
| Mod. Least Square | $[1-\rho]_+^2$ |
| Logistic Regression | $\log_2(1 + e^{-\beta\rho})$ |
| Exponential | $e^{-\beta\rho}$ |
| Sigmoidal | $(1 + e^{\beta(\rho-\theta)})^{-1}$ |

Although $\mathcal{D}$ is unknown, we can still try to minimize this function by exploiting the same structure of supervision and as much of the information we can gather from the available training set $\mathcal{S}$.

Specifically, the purpose of a GPLM based algorithm will be to find the hypothesis $\mathbf{w}$ that is able to minimize costs $c(S|\mathbf{w})$. As these are not continuous w.r.t. the parameter vector $\mathbf{w}$, they are approximated by introducing a continuous non-increasing loss function $l : \mathbb{R} \to \mathbb{R}^+$ approximating the indicator function. The (approximate) cost will be then defined by

$$\tilde{c}(S|\mathbf{w}) = \sum_{g \in \mathcal{G}(S)} \max_{\lambda \in g} \gamma(\lambda) l(\rho_A(\lambda|\mathbf{w})).$$

Examples of losses one can use are presented in Table 2.

The general problem can be given as in the following:

- Given a set $\mathcal{V}(\mathcal{S}) = \bigcup_{S \in \mathcal{S}} \mathcal{G}(S)$ of preference graphs
- Find a set of parameters $\mathbf{w}$ in such a way to minimize the functional

$$\mathcal{Q}(\mathbf{w}) = \mathcal{R}(\mathbf{w}) + \mu \mathcal{L}(\mathcal{V}(\mathcal{S})|\mathbf{w}), \tag{1}$$

  where $\mathcal{L}(\mathcal{V}(\mathcal{S})|\mathbf{w}) = \sum_{S \in \mathcal{S}} \tilde{c}(S|\mathbf{w})$ is related to the empirical cost and $\mathcal{R}(\mathbf{w})$ is a regularization term over the set of parameters. Note that for the solution to be admissible when multiple thresholds are used and there are constraints defined over their values (as in the ordinal regression settings), these constraints should be explicitly enforced.

The use of a regularization term in problems of this type has different motivations, including the theory on regularization networks (see e.g., [12]). Moreover, we can see that by choosing a convex loss function and a convex regularization term (let say the quadratic term $\mathcal{R}(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$) it guarantees the convexity of the functional $\mathcal{Q}(\mathbf{w})$ in (1) and then the uniqueness of the solution. Indeed, current kernel-based approaches defined for basic supervised learning tasks can be seen in this form when using the $\beta$-margin with $\beta = 1$. This suggests a new *universal* kernel method, which is able to solve many complex learning tasks [1].

## 3 GPLM Applications

In the following sections, two recent applications of GPLM are presented: for a job candidate selection task [4] and a patent classification task [3]. These real-world applications are discussed in some detail with the aim to give two examples of how a potential user can approach nonstandard supervised learning problems using a GPLM-based strategy.

### *3.1 Job Candidate Selection as a Preferential Task*

In a candidate selection task for filling a job role, one or more candidates have to be selected from a pool of candidates. Without loss of generality, let assume that the $k \geq 1$ most suited candidates for the job are selected. This decision is taken by looking at each candidate profile. Moreover, we may assume that the number $k$ of candidates to select is already known from the beginning. This last point is very important to model the problem. In fact, a candidate will be selected on the basis of which other candidates are in the pool. In other words, no decisions can be taken for a candidate without knowing who else is competing for the same position(s).

Assume the training set consists of past decisions about promotions to a given role. Then, for any of these decisions, we know which candidates were in a selection pool and how many and which candidates were selected for the job. Thus, it seems natural to interpret any past decision as a set of preferences in which the $k$ selected candidates were preferred to the others. More formally, we define $C_t = \{c_1, \ldots, c_{n_t}\}$ to be the set of candidates for the job role (the pool) at time $t$, $S_t = \{s_1^{(t)}, \ldots, s_{k_t}^{(t)}\}$ the set of candidates which got the promotion, and $U_t = \{u_1^{(t)}, \ldots, u_{n_t-k_t}^{(t)}\}$ the set of candidates which were not selected. Thus, there is evidence that $s_i$ was preferred to $u_j$ for each $i \in \{1, \ldots, k_t\}$ and $j \in \{1, \ldots, n_t - k_t\}$. Using our notation, we can write $s_i \succ u_j$. Note that a selection having a pool of cardinality $n_t$ and $k_t$ candidates selected for the job will introduce exactly $k_t \times (n_t - k_t)$ preferences. However, since $k_t \ll n_t$, the order of magnitude is still linear in the number of candidates.

Why not a Simple Binary Task?

One could think of a job role selection as a setting where for each candidate an independent decision is taken. In this case, at any time $t$, we would have exactly $n_t$ independent decisions (e.g., a $+1$ decision, representing that the candidate was selected for the job role, and a $-1$ decision representing that the candidate was not selected for the job role). This could be modeled as a typical binary task where any of the $2^{n_t}$ different outcomes are possible. However, a job role selection is competitive in its nature, i.e., the choice of one candidate instead of another is not

independent on the other's candidates potentials and only a fixed number of candidates can get the promotion. For this reason, the binary task does not seem to be the best choice. This will be confirmed in the experimental section where we have compared the GPLM model against a binary SVM implementation. Finally, it should be noted that the problem tends to be highly unbalanced when considered as a binary problem. In fact, the number of promoted candidates is a very small percentage of the number of candidates, who compete for the promotion. On the other hand, GPLM makes no additional assumption on the sign of the relevance function for different candidates only on the order it induces. This should make the problem easier and more balanced.

### 3.1.1 GPLM with SVM

In Sect. 2.4, we have shown how the preferential problem, i.e., the task to find a linear function, which is consistent with a set of preferences, can be cast as a binary problem. Examples in this case become $\psi(\lambda) = \mathbf{s}_i - \mathbf{u}_j$ for each $\lambda \equiv s_i \succ u_j$. Thus, a standard SVM algorithm applied to this new set of examples can be used to find a solution to the preferential problem.

Specifically, let $\Lambda = \{(S_1, U_1), \ldots, (S_T, U_T)\}$ be the sets involved in past promotions given as a training set for a given role, thus the SVM dual problem will be posed as

$$\begin{aligned} &\arg\max_{\alpha} \sum_t \sum_{s_i \in S_t} \sum_{u_j \in U_t} \alpha_{ij}^{(t)} - \frac{1}{2} \left\| \sum_t \sum_{s_i \in S_t} \sum_{u_j \in U_t} \alpha_{ij}^{(t)} \psi(s_i \succ u_j) \right\|^2 \\ &\text{s.t.} \qquad 0 \le \alpha_{ij}^{(t)} \le \mu, \end{aligned} \tag{2}$$

and the (primal) SVM solution which solves (1) will be in the form

$$\mathbf{w}_{SVM} = \sum_t \sum_{s_i \in S_t} \sum_{u_j \in U_t} \alpha_{ij}^{(t)} \psi(s_i \succ u_j).$$

Note that the kernel computation in this case consists in computing a kernel between preferences (i.e., dot product between their vectorial representations). Nevertheless, this kernel can be easily reduced to a combination of simpler kernels between candidate profiles in the following way:

$$\begin{aligned} \tilde{k}(c_i^1 \succ c_j^1, c_i^2 \succ c_j^2) &= \langle \mathbf{c}_i^1 - \mathbf{c}_j^1, \mathbf{c}_i^2 - \mathbf{c}_j^2 \rangle = \langle \mathbf{c}_i^1, \mathbf{c}_i^2 \rangle - \langle \mathbf{c}_i^1, \mathbf{c}_j^2 \rangle - \langle \mathbf{c}_j^1, \mathbf{c}_i^2 \rangle + \langle \mathbf{c}_j^1, \mathbf{c}_j^2 \rangle \\ &= k(c_i^1, c_i^2) - k(c_i^1, c_j^2) - k(c_j^1, c_i^2) + k(c_j^1, c_j^2), \end{aligned}$$

where $k(c_i, c_j) = \langle \mathbf{c}_i, \mathbf{c}_j \rangle$ is the kernel function associated with the mapping used for the candidate profiles. We have then reduced a preferential task into a binary task

which can be easily solved by a standard SVM by just redefining the kernel function suitably.

Furthermore, using the SVM decision function $f_{\mathrm{SVM}}(\lambda) = \mathrm{sgn}(\langle \mathbf{w}_{\mathrm{SVM}}, \psi(\lambda)\rangle)$ it is possible to determine whether a given order relation is verified between any two candidates. However, to decide which candidates should be selected for a new event $t$, $k_t \times (n_t - k_t)$ calculations of the above-defined function should be computed to obtain the relative order of candidates.

In the following, we show that the selection can actually be computed in linear time. To this end, we can decompose the weight vector computed by the SVM in the following way:

$$\begin{aligned}\mathbf{w}_{\mathrm{SVM}} &= \sum_t \sum_{c_i \in S_t} \sum_{c_j \in U_t} \alpha_{ij}^{(t)} \psi(\mathbf{c}_i \succ \mathbf{c}_j) = \sum_t \sum_{c_i \in S_t} \sum_{c_j \in U_t} \alpha_{ij}^{(t)} (\mathbf{c}_i - \mathbf{c}_j) \\ &= \sum_t \sum_{c_i \in S_t} \sum_{c_j \in U_t} \alpha_{ij}^{(t)} \mathbf{c}_i - \sum_t \sum_{c_i \in S_t} \sum_{c_j \in U_t} \alpha_{ij}^{(t)} \mathbf{c}_j .\end{aligned}$$

This decomposition allows us to decouple, in the computation of the relevance function for a new candidate, the contribution of candidate profiles given in the training set

$$\begin{aligned}f(c) = \langle \mathbf{w}_{\mathrm{SVM}}, \mathbf{c}\rangle &= \sum_t \sum_{c_i \in S_t} \left( \sum_{c_j \in U_t} \alpha_{ij}^{(t)} \right) \langle \mathbf{c}_i, \mathbf{c}\rangle \\ &\quad - \sum_t \sum_{c_j \in U_t} \left( \sum_{c_i \in S_t} \alpha_{ij}^{(t)} \right) \langle \mathbf{c}_j, \mathbf{c}\rangle \\ &= \sum_t \sum_{c_i \in S_t} \left( \sum_{c_j \in U_t} \alpha_{ij}^{(t)} \right) k(c_i, c) - \sum_t \sum_{c_j \in U_t} \left( \sum_{c_i \in S_t} \alpha_{ij}^{(t)} \right) k(c_j, c) \\ &= \sum_t \sum_{c_i \in S_t} \alpha_i^{(t)} k(c_i, c) - \sum_t \sum_{c_i \in S_t} \alpha_j^{(t)} k(c_j, c).\end{aligned}$$

Hence, the relevance function can be directly computed by post-processing the output of the SVM (the $\alpha$ vector) and then building a new model as follows

$$f(c) = \sum_{c_s} \beta_s k(c_s, c),$$

where $\beta_s = \sum_{t:c_s \in S_t} \alpha_s^{(t)} - \sum_{t:c_s \in U_t} \alpha_s^{(t)}$. The new model defined by the $\beta$'s can directly be used by an SVM, and it returns the correct relevance for any candidate.

### 3.1.2 Experimental Setting

Our data were collected from the Human Resources data warehouse of a bank. Specifically, we have considered all the events related to the promotion of an employee to the job role of director of a branch office (target job role). The data used ranges from January 2002 to November 2007. Each event involves from a minimum of 1 promotion up to a maximum of 7 simultaneous promotions. Since for each event a short list of candidates was not available, we were forced to consider as candidates competing for the promotion(s) all the employees which at the time of the event were potentially eligible for promotion to the target job role. Because of that, each event $t$ typically involves $k_t$ "positive" examples, i.e., the employees that were promoted, and $n_t \gg k_t$ "negative" examples, i.e., eligible employees that were not promoted. As already stated, $k_t$ ranges from 1 to 7, while $n_t$ ranges (approximately) from 3,700 to 4,200, for a total of 199 events, 267 positive examples, and 809,982 negative examples.[2] Each candidate is represented, at the time of the event, through a profile involving 102 features. Of these features, 29 involve personal data, such as age, sex, title of study, zone of residence, etc., while the remaining 73 features codify information about the status of service, such as current office, salary, hours of work per day, annual assessment, skills self-assessment, etc. The features, and the way they are numerically coded, were chosen in such a way that it is impossible to recognize the identity of an employee from a profile. Moreover, we were careful in preserving, for each numerical feature, its inherent metric if present, e.g., the ZIP codes where redefined so that the geographic degree of proximity of two areas is preserved in the numerical proximity of the new codes associated with these two areas.

### 3.1.3 Results

To test whether learning preferences was better than using a binary classifier where binary supervision is used for training and the score of the resulting classifier used to rank the instances belonging to the same event, we have performed a set of experiments on a representative subset of the whole dataset. The binary classifier was an SVM with gaussian kernel and the values to use for the hyperparameters were decided through a validation set. The gaussian kernel was used also for learning preferences. The results showed that it is better to learn preferences as the SVM obtained a total accuracy of 61.88% versus an accuracy of 76.20% obtained for the approach based on learning preferences. The accuracy measures how many ranking relations are correctly predicted. The cost mapping we used for the GPLM is the one described in Sect. 3.1 that is each training selection was mapped into the set of

[2] Note that the same employee can play the role of negative example in several events. Moreover, it might also be a positive example.

preferences obtained between any "selected" profile and any "not selected" profile. The SVMlight [22] implementation has been used for all the experiments.

### 3.2 Three-Layered Patent Classification as a Preferential Task

In many applicative contexts in which textual documents are labeled with thematic categories, a distinction is made between the primary and the secondary categories that are attached to a given document. The primary categories represent the topics that are central to the document, while the secondary categories represent topics that the document somehow touches upon, albeit peripherally. For instance, when a patent application is submitted to the European Patent Office (EPO), a primary category from the International Patent Classification (IPC) scheme[3] is attached to the application, and that category determines the expert examiner who will be in charge of evaluating the application. Secondary categories are instead attached for the only purpose of identifying related prior art, since the appointed expert examiner will need to determine the novelty of the proposed invention against existing patents classified under either the primary or any of the secondary categories. For the purposes of EPO, failing to recognize the true primary category of a document is thus a more serious mistake than failing to recognize a true secondary category.

We now propose GPLM models for the principled solution of the three-layered classification task. Let $d$ denote a document having the set $P(d) = \{c_p\}$ (a singleton) as the set of its primary categories, $S(d) = \{c_{s_1}, \ldots, c_{s_k}\}$ as the (possibly empty) set of its secondary categories, and $N(d) = \{c_{n_1}, \ldots, c_{n_l}\}$ as the set of its noncategories, such that $C = P(d) \cup S(d) \cup N(d)$.

GPLM: Ordinal Regression for Three-Layered Classification

One could be tempted to interpret the three-layered classification problem as a label-pivoted (multivariate) ordinal regression (MOR) problem, i.e., the problem to give a rank from the ordered set {primary, secondary, noncategory} to each category for a given instance. In the following, we first give a GPLM mapping already presented in [2] which can be demonstrated to be equivalent to the ordinal regression method in [7]. Then, we discuss why, in our opinion, this setting does not exactly match the three-layered classification in the patent classification application. Our experiments, which will be summarized in the following, will support this claim.

For ordinal regression, a GPLM model is built by considering two thresholds (see Fig. 2), e.g., $\tau_p$ and $\tau_s$. For each training document, the relevance function of a primary category should be above the threshold $\tau_p$, while the relevance function for any other category (either secondary or non-category) should be below the threshold $\tau_p$.

[3] http://www.wipo.int/classifications/en/

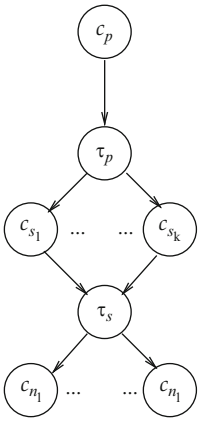


**Fig. 2** GPLM mapping for ordinal-regression supervision

On the other hand, the relevance function of any secondary category should be above the threshold $\tau_s$, while any noncategory should be below the threshold $\tau_s$. Summarizing, the preference graph for a given training document will be as in Fig. 2. As a simple example, consider the set of categories $C = \{c_1, c_2, c_3, c_4, c_5\}$ and a training document $d$ such that $P(d) = \{c_1\}$, $S(d) = \{c_2, c_3\}$, and $N(d) = \{c_4, c_5\}$. The set of preferences we generate is

$$\Lambda = \{(c_1 \succ_d \tau_p), (\tau_p \succ_d c_2), (\tau_p \succ_d c_3), (c_2 \succ_d \tau_s), (c_3 \succ_d \tau_s), (\tau_s \succ_d c_4), (\tau_s \succ_d c_5)\}$$

Finally, three-layered classification will be performed by selecting the category reaching the highest relevance score as primary category, and among the others, all the categories reaching a relevance score above the threshold $\tau_s$, as secondary categories.

At this point, we can discuss a little more about the OR-based preference model. In particular, in (multivariate) ordinal regression, it is assumed that, for each document, the rate given to a category is independent from the rate given to other categories. This assumption would be reasonable when discriminating between relevant categories (primary and secondaries) and noncategories, since this is not a "competitive" decision, but is far less reasonable when one has to choose exactly one (the most relevant) among relevant categories as the primary category for a document, since in this case we actually have a "competitive" decision. Thus, in this last case, the choice of the primary category is strongly dependent on which are the relevant categories. This difference recalls the difference between single-label classification (which is competitive) and multilabel classification (which is not competitive) in multiclass classification tasks. In other words, requiring the relevance score for the primary category to be higher than a given threshold seems

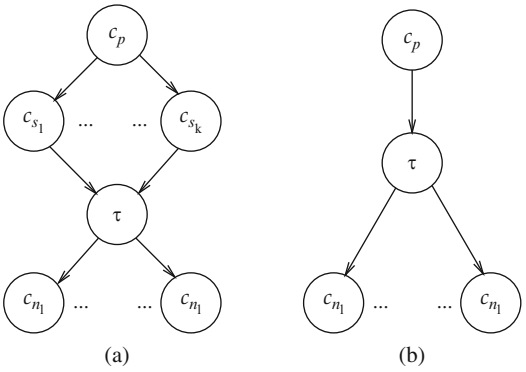


**Fig. 3** GPLM mapping for supervision with (**a**) nonempty secondary category set and (**b**) empty secondary category set

an unnecessary constraint which eventually could lead to a deteriorate overall performance.

GPLM: Ad-Hoc Mapping for Three-Layered Classification

A variant of the ordinal regression scheme, which seems more suitable for the task of three-layered classification, can be built as follows. Let us interpret the primary category as the most relevant among the relevant categories. This constraint is introduced by the insertion of a set of qualitative preferences between the primary and all the secondary categories. Moreover, given the multilabel nature of the problem to discern the secondary categories with respect to the remaining categories, a single threshold $\tau$ on the relevance scores has to be added between the secondary categories and the noncategories. The categories reaching a relevance score above the threshold (apart from the one recognized as the primary category) will be predicted as secondary categories. See Fig. 3a for a graphical representation of this kind of preference model. Note that whenever $S(d) = \emptyset$, this means that the relevance values for categories in $C \setminus P(d)$ are all below the threshold. To cope with this situation, the qualitative preferences can be collapsed into a direct quantitative preference between the primary category and the threshold. See Fig. 3b for a graphical description of this kind of preference. As a simple example, consider the set of categories $C = \{c_1, c_2, c_3, c_4, c_5\}$ and a training document $d$ such that $P(d) = \{c_1\}$, $S(d) = \{c_2, c_3\}$, and $N(d) = \{c_4, c_5\}$. The set of preferences we generate is

$$\Lambda = \{(c_1 \succ_d c_2), (c_1 \succ_d c_3), (c_2 \succ_d \tau), (c_3 \succ_d \tau), (\tau \succ_d c_4), (\tau \succ_d c_5)\}.$$

Similarly, if $d$ is instead such that $P(d) = \{c_1\}$, $S(d) = \emptyset$, $N(d) = \{c_2, c_3, c_4, c_5\}$, this will generate the set of preferences

$$\Lambda = \{(c_1 \succ_d \tau), (\tau \succ_d c_2), (\tau \succ_d c_3), (\tau \succ_d c_4), (\tau \succ_d c_5)\}$$

### 3.2.1 Experimental Setting

We have evaluated our method on the WIPO-alpha Intellectual Property Organization (WIPO) in 2003. The dataset consists of 75,250 patents classified according to version 8 of the International Patent Classification scheme (IPC). Each document $d$ has one primary category (known as the *main IPC symbol* of $d$), and a variable (possibly null) number of secondary categories (the *secondary IPC symbols* of $d$). To avoid problems due to excessive sparsity, and consistently with previous literature [13], we only consider categories at the subclass level of the IPC scheme; each of the 630 IPC subclasses is thus viewed as containing the union of the documents contained in its subordinate groups.

WIPO-alpha comes partitioned into a training set *Tr* of 46,324 documents and a test set *Te* of 28,926 documents. In our experiments, we used the entire WIPO-alpha set of 75,250 documents. Each document includes a title, a list of inventors, a list of applicant companies or individuals, an abstract, a claims section, and a long description. As in [13], we have only used the title, the abstract, and the first 300 words of the "long description". Pre-processing has been obtained by performing stop word removal, punctuation removal, down-casing, number removal, and Porter stemming. Vectorial representations have been generated for each document by the well-known "ltc" variant of cosine-normalized *tfidf* weighting. We refer the reader to [3] for a complete description of the experimental setting and the dataset.

Two additional baseline methods have been defined. In the first baseline (dubbed "Baseline1"), a binary classifier is built for each $c \in C$ (by using as positive examples of category $c_i$ all the documents that have $c_i$ either as a primary or as a secondary category) and use the real-valued scores returned by each classifier for $d$: the category for which the largest score has been obtained are selected as the primary category, while the set of secondary categories are identified by optimizing a threshold for each individual category and selecting the categories whose associated classifier has returned a score above its associated threshold. We have indeed implemented this approach (by using standard binary SVMs). A slightly stronger approach (dubbed "Baseline2") consists in performing two different classification tasks, a first one (by means of an SVM-DDAG [28] single-label classifier $h_P$) aimed at identifying the primary category of $d$, and a second one (by means of a multilabel classifier $h_S$ consisting of $m$ SVM-based binary classifiers $h_S^i$, one for each category $c_i \in \{c_1, \ldots, c_m\}$) aimed at identifying, among the remaining categories, the secondary categories of $d$. The $h_P$ classifier is trained by using, as positive examples of each $c_i$, only the training documents that have $c_i$ as primary category. Each of the $h_S^i$ is instead trained by using as positive examples only the training documents that have $c_i$ as secondary category, and as negative examples only the training documents that have $c_i$ as noncategory (those that have $c_i$ as primary category are discarded).

### 3.2.2 Results

The results obtained for the different classifiers are summarized in Table 3. Ad-hoc evaluation measures have been used. In particular, the $F_1$ measure is computed

**Table 3** Micro-averaged $F_1^3$ values obtained by the classifiers

| | $F_1^{PS}$ | $F_1^{SN}$ | $F_1^{PN}$ | $F_1^3$ |
|---|---|---|---|---|
| Baseline1 | 0.851 | 0.180 | 0.482 | 0.499 |
| Baseline2 | **0.886** | 0.200 | 0.464 | 0.504 |
| Ordinal regression | 0.7847 | 0.1774 | **0.5343** | 0.5077 |
| GPLM Adatron | 0.8433 | **0.2138** | 0.5129 | **0.5206** |

for each pair of layers and then combined to form a single measure $F_1^3$. The first two rows report the performances of the two baseline classifiers. It can be observed that they have almost identical $F_1^3$ and are not so good in telling apart secondary categories from noncategories ($F_1^{SN}$). The third row reports the performance of the ordinal regression classifier, which turns out to have the best separation between primary and noncategories ($F_1^{PN}$) but a quite low performance on separating primary and secondary categories ($F_1^{PS}$). These results seem coherent with the analysis we have given in Sect. 3.2 as the separation between primary categories and noncategories is overconstrained by the ordinal regression model. The overall performance ($F_1^3$) slightly improves over the baseline classifiers. The fourth row reports the performance of the GPLM using an own implementation of the Kernel–Adatron [15] as optimizer. With respect to the baselines and the ordinal regression classifier, there is a clear improvement on $F_1^{SN}$, while $F_1^{PS}$ decreases. Overall, however, there is a significant improvement in $F_1^3$.

## 4 Related Work and Discussion

Some other efforts have been made to generalize label ranking tasks. The first work on this we are aware of is [18] where the authors show how different label ranking problems can be cast into a linear problem which is solvable by a perceptron in an augmented feature space. In [21], a variant is presented in which the ranking is performed based on a voting strategy on classifiers discriminating between label pairs. In [11], the authors propose a setting in which a label ranking problem is map into a set of preference graphs and a convex optimization problem is defined to solve it. Our preference model proposed in [5] generalizes on these two previous approaches, by proposing a more flexible way to model cost functions for the same problems, and giving a kernel-based large margin solution for these kind of tasks. More recently, in [30], a large margin method to solve single-label problems with structured (e.g., hierarchical) output has been proposed. This last approach is not, however, directly applicable to solve general label ranking tasks as it requires the solving of an optimization problem with a different constraint for each possible (label) ranking and also the decoding step can show exponential complexity for general cost functions. In [24] it has been shown that this technique is, however, feasible when applied to certain cost functions that are relevant for information retrieval ranking tasks.

The general task of instance ranking is gaining a large popularity especially in the information retrieval community where the typical need is to rank documents based on their relevance for a query. In this context, this task is commonly referred to as *learning to rank*. The approaches to this general task can be divided into three categories: *point-wise*, *pair-wise*, and *list-wise*. This taxonomy is similar to the one presented in this chapter. In the point-wise approach, see for example [9, 16, 25, 27, 29], the input space are single documents and the output space are real values or ordinal categories. This kind of settings are a subset of the tasks that have been referred to as quantitative tasks in this chapter, namely the class of instance-pivoted Multivariate Ordinal Regression. In the pair-wise approach, see for example [6, 8, 14, 20, 23], the input space are document pairs (basically preferences) and the output space are documents ordered by relevance. This kind of settings are those tasks which have been here referred to as qualitative tasks, and Instance Rankings in particular. Finally, in the list-wise approach, see for example [32], the input space is the whole document set and typically a direct optimization of the evaluation function is required. This last approach is very challenging as these evaluation functions are often noncontinuous and nondifferentiable.

One clear advantage of the approach presented in this chapter, with respect to all the ones sketched in this section, is its ability to treat uniformly the label and the instance ranking settings as well as the regression setting, exploiting a preference-centric point of view. Reducing all of these problems to preference optimization implies that any optimization technique for preference learning can be used to solve them all.

## 5 Conclusion and Future Extensions

We have discussed a general preference model for supervised learning and applications to complex prediction problems, as job selection and patent classification. The first application is an instance-ranking problem while the second application is a label-ranking problem where categories have to be associated with patents according to a three-layered structure (primary, secondary, non-category).

An interesting aspect of the proposed preference model is that it allows to codify cost functions as preferences and naturally plug them into the same training algorithm. In this view, the role of the cost functions resembles the role of kernels in kernel-machines. Moreover, the proposed method gives a tool for comparing different algorithms and cost functions on a same learning problem.

In the future, it would be interesting to explore extensions to the model, including: (a) Considering models with disjunctive preferences as it would increase the flexibility of the model. (b) Studying new fast (approximate) algorithms when the number of examples/preferences are simply too large to be coped with by standard learning algorithms. (c) Extending the concept of preferences to preferences to a given degree, i.e., when a preference constraint have to be fulfilled with a given margin.

## References


1. F. Aiolli, Large margin multiclass learning: models and algorithms. Ph.D. thesis, Department of Computer Science, University of Pisa, 2004. http://www.di.unipi.it/phd/tesi/tesi_2004/PhDthesisAiolli.ps.gz
2. F. Aiolli, A preference model for structured supervised learning tasks, in *Proceedings of the IEEE International Conference on Data Mining (ICDM)* (2005), pp. 557–560
3. F. Aiolli, R. Cardin, F. Sebastiani, A. Sperduti, Preferential text classification: Learning algorithms and evaluation measures. Inf. Retr. **12**(5), 559–580 (2009)
4. F. Aiolli, M. De Filippo, A. Sperduti, Application of the preference learning model to a human resources selection task, in *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM)* (Amsterdam, NL, 2009), pp. 203–210
5. F. Aiolli, A. Sperduti, Learning preferences for multiclass problems, in *Advances in Neural Information Processing Systems* (MIT, Cambridge, MA, 2005) pp. 17–24
6. C.J.C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G.N. Hullender, Learning to rank using gradient descent, in *Proceedings of the International Conference on Machine Learning (ICML)* (2005), pp. 89–96
7. W. Chu, S. Sathiya Keerthi, Support vector ordinal regression. Neural Comput. **19**(3), 792–815 (2007)
8. W.W. Cohen, R.E. Schapire, Y. Singer, Learning to order things. J. Artif. Intell. Res. **10** 243–270 (1999)
9. K. Crammer, Y. Singer, Pranking with ranking, in *Advances in Neural Information Processing Systems (NIPS)* (2002), pp. 641–647
10. K. Crammer, Y. Singer, A family of additive online algorithms for category ranking. J. Mach. Learn. Res. **3**, 1025–1058 (2003)
11. O. Dekel, C.D. Manning, Y. Singer, Log-linear models for label ranking, in *Advances in Neural Information Processing Systems* (2003)
12. T. Evgeniou, M. Pontil, T. Poggio, Regularization networks and support vector machines. Adv. Comput. Math. **13**, 1–50 (2000)
13. C.J. Fall, A. Törcsvári, K. Benzineb, G. Karetka, Automated categorization in the International Patent Classification. SIGIR Forum **37**(1), 10–25 (2003)
14. Y. Freund, R.D. Iyer, R.E. Schapire, Y. Singer, An efficient boosting algorithm for combining preferences. J. Mach. Learn. Res. **4**, 933–969 (2003)
15. T.T. Friess, N. Cristianini, C. Campbell, The kernel adatron algorithm: a fast and simple learning procedure for support vector machines, in *Proceedings of International Conference of Machine Learning (ICML)* (1998), pp. 188–196
16. T.T. Friess, N. Cristianini, C. Campbell, Subset ranking using regression, in *Proceedings of the International Conference on Learning Theory (COLT)* (Springer Berlin/Heidelberg, 2006), pp. 605–619
17. J. Fürnkranz, E. Hüllermeier, E. Mencía, K. Brinker, Multilabel classification via calibrated label ranking. Mach. Learn. **73**(2), 133–153 (2008)
18. S. Har-Peled, D. Roth, D. Zimak, Constraint classification for multiclass classification and ranking, in *Advances in Neural Information Processing Systems* (2002), pp. 785–792
19. R. Herbrich, T. Graepel, P. Bollmann-Sdorra, K. Ober-mayer, Learning a preference relation for information retrieval, in *Proceedings of the AAAI Workshop Text Categorization and Machine Learning* (1998)
20. R. Herbrich, T. Graepel, K. Obermayer, Large margin rank boundaries for ordinal regression, in *Advances in Large Margin Classifiers* (MIT, 2000), pp. 115–132
21. E. Hüllermeier, J. Fürnkranz, W. Cheng, K. Brinker, Label ranking by learning pairwise preferences. *Artif. Intell.* **172**(16–17), 1897–1916 (2008)
22. T. Joachims, Making large-scale svm learning practical, in *Advances in Kernel Methods - Support Vector Learning* ed. by B. Schölkopf, C. Burges, A. Smola (MIT, 1999)
23. T. Joachims, Optimizing search engines using clickthrough data, in *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)* (2002) pp. 133–142

24. Q. Le, A. Smola, Direct optimization of ranking measures. Technical report, NICTA, Canberra, Australia, 2007
25. P. Li, C. Burges, Q. Wu, Mcrank: Learning to rank using multiple classification and gradient boosting, in *Advances in Neural Information Processing Systems (NIPS)* (MIT, 2008), pp. 897–904
26. P. McCullagh, J.A. Nelder, *Generalized Linear Models* (Chapman & Hall, 1983)
27. R. Nallapati, Discriminative models for information retrieval, in *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR)* (ACM, 2004), pp. 64–71
28. J.C. Platt, N. Cristianini, J. Shawe-Taylor, Large margin DAGs for multiclass classification, in *Advances in Neural Information Processing Systems (NIPS)* (1999), pp. 547–533
29. A. Shashua, A. Levin, Ranking with large margin principle: Two approaches, in *Advances in Neural Information Processing Systems (NIPS)* (2002), pp. 937–944
30. I. Tsochantaridis, T. Hofmann, T. Joachims, Y. Altun, Support vector machine learning for interdependent and structured output spaces, in *Proceedings of the International Conference on Machine learning (ICML)* (2004), pp. 1453–1484
31. H. Wu, H. Lu, S. Ma, A practical svm-based algorithm for ordinal regression in image retrieval, in *Proceedings of the ACM international conference on Multimedia* (2003), pp. 612–621
32. F. Xia, T. Liu, J. Wang, W. Zhang, H. Li, Listwise approach to learning to rank: theory and algorithm, in *Proceedings of the International Conference on Machine Learning (ICML)* (2008), pp. 1192–1199

http://www.springer.com/978-3-642-14124-9

Preference Learning
Fürnkranz, J.; Hüllermeier, E. (Eds.)
2011, IX, 466 p., Hardcover
ISBN: 978-3-642-14124-9