Mapping Preferences into Euclidean Space

Oscar Luaces[a], Jorge Díez[a,*], Thorsten Joachims[b], Antonio Bahamonde[a,b]

[a] *Universidad de Oviedo, Artificial Intelligence Center, Gijón, Asturias, Spain*
[b] *Cornell University, Department of Computer Science, Ithaca, NY, USA*

Abstract

Understanding and modeling human preferences is one of the key problems in applications ranging from marketing to automated recommendation. In this paper, we focus on learning and analyzing the preferences of consumers regarding food products. In particular, we explore machine learning methods that embed consumers and products in an Euclidean space such that their relationship to each other models consumer preferences. In addition to predicting preferences that were not explicitly stated, the Euclidean embedding enables visualization and clustering to understand the overall structure of a population of consumers and their preferences regarding the set of products. Notice that consumers' clusters are market segments, and products clusters can be seen as groups of similar items with respect to consumer tastes. We explore two types of Euclidean embedding of preferences, one based on inner products and one based on distances. Using a real world dataset about consumers of beef meat, we find that both embeddings produce more accurate models than a tensorial approach that uses a SVM to learn preferences. The

[*] Corresponding author: Tel: +34 985 182 588
Email addresses: oluaces@uniovi.es (Oscar Luaces), jdiez@uniovi.es (Jorge Díez), tj@cs.cornell.edu (Thorsten Joachims), abahamonde@uniovi.es (Antonio Bahamonde)

reason is that the number of parameters to learned in embeddings can be considerably lower than in the tensorial approach. Furthermore, we demonstrate that the visualization of the learned embeddings provides interesting insights into the structure of the consumer and product space, and that it provides a method for qualitatively explaining consumer preferences. Additionally, it is important to emphasize that the approach presented here is flexible enough to allow its use with different levels of knowledge about consumers or products; therefore the application field is very wide to grasp an accurate understanding of consumers' preferences.

Keywords:
Preference Learning, matrix factorization, graphical representations,
learning to order, visualization

1. Introduction

In 1927, Thurstone (1927) presented a *law of comparative judgment* to approach to qualitative comparisons from a psychological point of view. According to this law, users *react* differently to each item, and they identify the degree of compatibility with the quality to be compared. The difference of these degrees define the *discriminal process* between pairs of items.

From a Machine Learning perspective, there are two main ways to approach preferences. They can be represented by a real-valued function, which assigns a utility value to the object, or by a preference relation, which compares two different items; see (Hüllermeier and Fürnkranz, 2013). In the first approach, the degrees of compatibility (usually called *utilities*) are considered as the target output that can be learned by means of *ordinal regression or* *classification* methods. This is a suitable approach when it is possible to assume that users assign those utilities depending exclusively on the item being assessed. However, in some cases there is a *batch effect*; that is, the assessment of an item depends on the batch of items included in the same comparison. When this is the case, it is more suitable to use the second approach and consider preferences as a binary relation. The goal here is to learn the *relative ordering* of items given by the user instead of the utility itself. This is the approach used by Herbrich et al. (1999), Joachims (2002), Bahamonde et al. (2007) and Rendle et al. (2009).

In this paper we are concerned with learning preferences expressed by consumers of a kind of products. Consumers typically assign the utilities only as a way to express relative preferences instead of absolute values. Therefore, the datasets that we are going to use are collections of pairwise comparisons, called *preference judgments*, that represent the discriminant process of one consumer between two products. In addition to modeling the products, we also explicitly model individual consumers. To represent the interaction of consumers and products we propose several factorization approaches and a tensorial approach.

A desirable property of factorization approaches is that they entail an embedding of both consumers and products in a common Euclidean space where the utility can be expressed in geometric (or *graphical*) terms. The consequence is that, as a side effect, learning preferences with these approaches provides a setting for visualization of clusters in both consumers and products.

In the following sections we introduce a common framework to learn factorizations and SVM tensorial models. The purpose is to discuss the characteristics of these approaches according only to their mathematical formulations.

In all cases, the objects involved in preferences (consumers and products) can be represented by a combination of a binary identification code or by vectors of feature-values. Notice that, for instance, in food products, the features of consumers or the products are not always available. Moreover, if a food industry is planing to launch a new product there is a reduced set of options that they want to test, and they can be just represented by an identification code. On the other hand, when there is a selected panel of singular consumers, they can be unequivocally identified with a label.

After the formal presentation of the methods, we show the results of an exhaustive experimentation using a real world dataset of consumers of beef meat. First we compare the results of factorization methods with those achieved by SVM. In the datasets used in this paper, factorization methods outperform SVM, probably this is a general fact. One reason is that the number of parameters to be learned is smaller in the factorization approaches. Additionally, the formal models learned in all cases are quite similar and they all capture the possible interactions of both the features of consumers and products.

The contributions of the paper are the following: (i) it presents a common framework for different approaches to learn preferences using matrix factorization, (ii) the paper illustrates the formal presentation with a real world problem of consumers and (food) products, (iii) the last section shows a favorable comparison of factorization and SVM tensorial approaches, (iv) the paper emphasizes the graphical and geometrical possibilities of the factorization methods as a tool to analyze the complex relationships of users and items.

2. Related Work

Learning preferences has been studied with different approaches. A recent Special Issue of the Machine Learning Journal (Hüllermeier and Fürnkranz, 2013) includes some interesting approaches and application fields.

From a conceptual point of view, the aim is to learn an ordering relation from some pairwise comparisons. Thus, the learning task can be read as a binary classification task using SVMs, see for instance (Herbrich et al., 1999; Joachims, 2002).

In this paper we adopt a more general strategy, we explicitly optimize a loss function with regularization. For instance, it is possible to use the logistic loss as in the learner proposed by Rendle et al. (2009). The algorithm presented in that paper was derived from a Bayesian analysis of the ranking problem of a user for a set of items. The algorithm is called *Bayesian Personalized Ranking (BPR)* and was devised as a method to solve the maximum posterior estimation.

We present a general setting that includes, at the same time, a factorization framework and a tensorial approach: a SVM that uses tensor products to model the interactions of consumer and item representations. Tensorial representations were already used, for instance, in (Basilico and Hofmann, 2004; Rendle and Schmidt-Thieme, 2010; Pahikkala et al., 2012). We report a comparison between factorization methods and this SVM approach in the experimental section using a real dataset.

Factorization algorithms were previously used in recommender systems in some of the best ranked systems of the Netflix prize; see for instance (Koren et al., 2009). Many other papers propose matrix factorization for solving specific problems in recommender systems; see for instance (Ocepek et al., 2015).

A software library to do *factorization machines* with a wide variety of options is presented in (Rendle, 2012). Other similar implementations can be found in (Chen et al., 2011; Agarwal and Chen, 2009; Bayer, 2015). Let us remark that the goal of this paper is not to present another implementation. We use a quite straightforward *SGD (Stochastic Gradient Descent)* implementation whose main advantage is that it is the same for 3 different approaches. We want to underscore that the differences arise only from the formulation of the approach, but not from any implementation issue. Additionally, we are interested in discussing the necessity of using the features of the items and consumers involved in the preferences judgments. This is a central point in learning preferences and the approach presented in this paper is quite suitable for this purpose. We may use all the convenient information in a real word application.

Another interesting use of factorizations is presented in (Weston et al., 2010, 2011). The target is information retrieval, and so the aim was to optimize the ranking of labels attached to queries (images or music). In this case the output is an ordered set of labels.

As was said in the introduction, we are concerned with the graphical properties of the model learned from preferences. Both consumers and items are located in an Euclidean space with one specific aim. This is the case, for instance in (Moore et al., 2012; Chen et al., 2012) where the purpose was to build an embedding of songs. The proximity was learned from a collection of *playlists*. Once the map is built, playlists are generated using the relative distances of songs. To model proximity, the authors use Gaussian distributions, and the same tool was used also to add tags to the songs. In this paper we do not assume any distribution of the representation of data in the Euclidean space.

Another paper related to the work reported here is (Xing et al., 2002). Here, the authors learn a metric for Euclidean points that represents *similarities* and *dissimilarities*. The metric is given by a positive semi-definite matrix that is the solution of a convex optimization problem. In our case the factorization eases the learning process since the number of parameters to be estimated may be significantly fewer. Moreover, the preference learning tasks only have dissimilarity examples; there are no similarity cases to guide the induction. On the other hand, in (Xing et al., 2002) there is no reduction of dimensionality neither visualization purposes: the objective is to find clusters. A quite similar approach can be found in (Parameswaran and Weinberger, 2010), in this case to learn a metric for Multi-Task Learning.

To learn metrics is also the aim of Peltonen et al. (2003). The purpose is to reduce the dimensionality from visualization. The source data are collections of labelled data for classification tasks. The proposals are extensions of the so-called Self-Organizing Maps (SOM). However, notice that our purpose is not only to learn a metric, but to learn preferences while represent in a metric space both consumers and products.

Finally, the visualization method presented here is in fact a supervised learning algorithm, like supervised PCA (Koren and Carmel, 2004; Yu et al., 2006; Du et al., 2015) for instance. The difference is that our approach explicitly incorporates the loss function and the definition of similarity that we want to obtain at the end of the process. And, of course, the method presented here is devised for learning preferences.

3. Formal Framework

Let us consider the following dataset

$$D = \{(\boldsymbol{x}_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_n, f(\boldsymbol{x}_n))\}. \tag{1}$$

Here we assume that f is an unknown real function on the space from where inputs $\boldsymbol{x} \in \mathbb{R}^m$ are drawn.

The aim is to find a new function g of input data $\boldsymbol{x}$, that depends also on some parameters θ, such that the variations of f can be predicted by the variations of g. The function g will have an analytical definition that makes it straightforward to compute g on any input. In symbols, the aim of g is to maximize the probability

$$\Pr\big(f(\boldsymbol{x}) > f(\boldsymbol{x'}) \iff g(\boldsymbol{x}, \theta) > g(\boldsymbol{x'}, \theta)\big). \tag{2}$$

In the following, as usual in this context, we will call g a *utility* function.

To learn g we define the following ordering induced by D

$$D_{or} = \{\big(\boldsymbol{x}_i, \boldsymbol{x}_j; [\![f(\boldsymbol{x}_i) > f(\boldsymbol{x}_j)]\!]\big) : i, j = 1, \ldots, n\}. \tag{3}$$

The symbol $[\![p]\!]$ stands for the value 1 when the predicate p is true, and -1 otherwise. In the next subsections we present an approach to learn g from this binary classification task.

Formally, the learning process of the parameters θ of g starts with the dataset D_{or} (Eq. 3). Soon we shall see that we may use only the examples of the *positive* class,

$$D_{or}^{+} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : f(\boldsymbol{x}_i) > f(\boldsymbol{x}_j),\ i, j = 1, \ldots, n\}. \tag{4}$$

Notice that, in fact, we do not need the function f in our approach. In practice, f is hidden and we do not have access to it; otherwise, the dataset (Eq. 1) could be seen as a regression task. Roughly speaking, the dataset D_{or}^{+} is the set of pairs where an explicit ordering has been registered. Each pair is formed by the better and the worse objects. Usually these pairs are called *preference judgments*, see (Joachims, 2002).

We adopted a margin maximization approach, detailed in the next section, to learn from such a dataset in order to include the hypothesis learned by SVMs as in (Herbrich et al., 1999; Joachims, 2002; Bahamonde et al., 2007; Basilico and Hofmann, 2004; del Coz et al., 2005; Díez et al., 2005, 2006). This could also be solved using a probabilistic approach.

4. Maximum Margin Approach

As usual, we assume that all these examples are independently and identically drawn (i.i.d.) from an unknown distribution. Thus, using a *maximum margin* approach, the parameters θ should minimize

$$Loss(\theta, D_{or}^{+}) = \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in D_{or}^{+}} \max\Big(0, 1 - g(\boldsymbol{x}_i, \theta) + g(\boldsymbol{x}_j, \theta)\Big). \tag{5}$$

Following Rendle et al. (2009), margin maximization can be done using an SGD algorithm (Robbins and Monro, 1951) with a regularization term for the parameter θ, $r(\theta)$. Thus, the optimal value, θ^* is given by

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ Loss(\theta) + \nu r(\theta) \tag{6}$$

The idea is to ensure that the difference of the utilities in a preference judgment is at least 1.

$$g(\boldsymbol{x}_i, \theta) - g(\boldsymbol{x}_j, \theta) \geq 1, \quad (\boldsymbol{x}_i, \boldsymbol{x}_j) \in D_{or}^+$$

Of course, this is equivalent to

$$g(\boldsymbol{x}_j, \theta) - g(\boldsymbol{x}_i, \theta) < -1, \quad (\boldsymbol{x}_i, \boldsymbol{x}_j) \in D_{or}^-$$

where D_{or}^- is the subset of D_{or} (Eq. 3) with negative classes. The consequence is that we may get rid of the negative part since it is redundant.

The corresponding optimization with this loss function can be solved with Algorithm 1 that implements this approach using an L_2 regularization term. The updating step due to $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is done by:

$$\theta \quad \leftarrow \quad \theta - \gamma \Big[\frac{\partial (Loss(\theta))_{ij}}{\partial \theta} + \nu \frac{\partial r(\theta)}{\partial \theta} \Big]. \tag{7}$$

That is,

$$\theta \leftarrow \theta + \gamma \Big[\frac{\partial g(\boldsymbol{x}_i, \theta)}{\partial \theta} - \frac{\partial g(\boldsymbol{x}_j, \theta)}{\partial \theta} - \nu \frac{\partial r(\theta)}{\partial \theta} \Big] \tag{8}$$

if

$$1 - g(\boldsymbol{x}_i, \theta) + g(\boldsymbol{x}_j, \theta) > 0.$$

Additionally, to ensure numerical stability, following Weston et al. (2010, 2011), we use a parameter R (a *radius*) such that the *size* of θ is always smaller or equal than R.

Algorithm 1 SGD algorithm to learn a utility function that maximizes the margin as defined in (Eq. 5) using an L_2 regularization

```
**Input:** $D^+_{or}$; {(Eq. 4)}
**Input:** $\gamma > 0$ {*learning rate*}; $\nu > 0$ {*regularization parameter*};
**Input:** $R > 0$ {*radius*};
**assign** random values to the components of $\theta$;
**repeat**
    fetch random $(\boldsymbol{x}_{better}, \boldsymbol{x}_{worse}) \in D^+_{or}$;
    **if** $1 > g(\boldsymbol{x}_{better}, \theta) - g(\boldsymbol{x}_{worse}, \theta)$ **then**
        $\theta \leftarrow \theta + \gamma \left[ \frac{\partial g(\boldsymbol{x}_{better}, \theta)}{\partial \theta} - \frac{\partial g(\boldsymbol{x}_{worse}, \theta)}{\partial \theta} - \nu \frac{\partial r(\theta)}{\partial \theta} \right]$;
        **if** $\|\theta\| > R$ **then**
            $\theta \leftarrow \frac{R}{\|\theta\|} \theta$;
        **end if**
    **end if**
**until** *stop criterion*
```

5. Factorization and Tensorial Approaches

In the last section, inputs were described by a generic vector $\boldsymbol{x}$ and the aim was to emphasize the ordering of these vectors according to f values. Now we are going to get into the structure of inputs as the concatenation of two different vectors, the representation of *consumers* and *items* or *products* (we prefer products to use p instead of i for short in equations). Thus, in the following, we are going to assume that each input data can be split in two parts:

$$\boldsymbol{x} = (\boldsymbol{c}, \boldsymbol{p}).$$

In this section, we introduce three possible definitions of the utility function g. They have in common that rest on the interaction of the vectorial representation of consumers and products.

5.1. Mapping Consumers and Products: Matrix Factorization

In this subsection, we are going to consider an embedding of both consumers and products in a common Euclidean space. Then, the function g (Eq. 2) will be defined in terms of the mappings in the common space.

We assume that *consumers* are described by vectors in a Euclidean space of dimension $|Con|$, while *products* are given by vectors with $|Prod|$ components. We are going to represent them in a common space of dimension k using two linear maps given respectively by matrices $\boldsymbol{W}$ and $\boldsymbol{V}$.

$$\mathbb{R}^{|Con|} \longrightarrow \mathbb{R}^k, \qquad \boldsymbol{c} \rightsquigarrow \boldsymbol{W}\boldsymbol{c}, \tag{9}$$

$$\mathbb{R}^{|Prod|} \longrightarrow \mathbb{R}^k, \qquad \boldsymbol{p} \rightsquigarrow \boldsymbol{V}\boldsymbol{p}. \tag{10}$$

Let us remark that, as usual, we are considering vectors as column matrices.

In this context, the parameter θ to be learned is the set of matrices $\boldsymbol{W}$, $\boldsymbol{V}$. We are trying to solve the optimization problem

$$\boldsymbol{W}^*, \boldsymbol{V}^* = \underset{W,V}{\operatorname{argmin}} \Big(Loss(\boldsymbol{W}, \boldsymbol{V}, D^+_{or}) + \nu r(\boldsymbol{W}) + \nu r(\boldsymbol{V}) \Big). \tag{11}$$

Notice that there are different options to define the interaction of consumers and products. Next, we present two of them.

5.1.1. Inner Products

The first alternative is to formalize the interactions by the following *inner* product of the mappings of consumers and products in $\mathbb{R}^k$.

$$\begin{aligned} g^{in}(\boldsymbol{x}) = g^{in}(\boldsymbol{c}, \boldsymbol{p}) = \langle \boldsymbol{W}\boldsymbol{c}, \boldsymbol{V}\boldsymbol{p} \rangle \\ =(\boldsymbol{W}\boldsymbol{c})^T\boldsymbol{V}\boldsymbol{p} = \boldsymbol{c}^T\boldsymbol{W}^T\boldsymbol{V}\boldsymbol{p} =\sum_{r=1}^{|Con|}\sum_{s=1}^{|Prod|}\left(\boldsymbol{W}^T\boldsymbol{V}\right)_{rs}\left(\boldsymbol{c}\boldsymbol{p}^T\right)_{rs} \\ =(\boldsymbol{V}\boldsymbol{p})^T\boldsymbol{W}\boldsymbol{c} =\sum_{r=1}^{|Con|}\sum_{s=1}^{|Prod|}\left(\boldsymbol{V}^T\boldsymbol{W}\right)_{rs}\left(\boldsymbol{p}\boldsymbol{c}^T\right)_{rs}. \end{aligned} \tag{12}$$

It is interesting to realize that the utility function g^{in} is given by a linear combination of all products formed by one component from the consumer description, and one component of the description of the product.

The type of equation is different if we add one constant component (with value 1 for instance) to the vectorial representation of consumers and products; that is,

$$\boldsymbol{c}^T \leftarrow [\boldsymbol{c}^T\ 1]; \quad \boldsymbol{p}^T \leftarrow [\boldsymbol{p}^T\ 1]. \tag{13}$$

In this case, the utility function can be thought as follows,

$$g^{in}(\boldsymbol{c}, \boldsymbol{p}) =\sum_{r,s}\alpha_{rs}\boldsymbol{c}_r\boldsymbol{p}_s + \sum_r \beta_r\boldsymbol{c}_r + \sum_s \delta_s\boldsymbol{p}_s + \tau, \tag{14}$$

for some real coefficients $\alpha_{r,s}, \beta_r, \delta_s$ and τ. That is, the utility is a polynomial of degree 2 where the monomials of degree 2 are always built by the product of one component of the representation of consumers and other from the products.

If we compare the equations (Eq. 12, 14), we appreciate that the coefficients of the polynomial that defines g^{in} are *factorized* in two matrices, as was mentioned in the introduction.

The partial derivatives needed to implement this approach in Algorithm 1 are the following:

$$\begin{aligned}\frac{\partial g^{in}(\boldsymbol{c},\boldsymbol{p})}{\partial \boldsymbol{W}} &= \frac{\partial (\boldsymbol{V}\boldsymbol{p})^T(\boldsymbol{W}\boldsymbol{c})}{\partial \boldsymbol{W}} = \boldsymbol{V}\boldsymbol{p}\boldsymbol{c}^T,\\ \frac{\partial g^{in}(\boldsymbol{c},\boldsymbol{p})}{\partial \boldsymbol{V}} &= \frac{\partial (\boldsymbol{W}\boldsymbol{c})^T(\boldsymbol{V}\boldsymbol{p})}{\partial \boldsymbol{V}} = \boldsymbol{W}\boldsymbol{c}\boldsymbol{p}^T.\end{aligned}$$

On the other hand, we use the square of the Frobenius norm as the regularization summand.

$$\begin{aligned}r(\boldsymbol{W}) &= \left\|\boldsymbol{W}\right\|_F^2 = Tr(\boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{W}),\\ r(\boldsymbol{V}) &= \left\|\boldsymbol{V}\right\|_F^2 = Tr(\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{V}).\end{aligned}$$

Therefore, the regularization derivatives are

$$\frac{\partial Tr(\boldsymbol{W}^T\boldsymbol{W})}{\partial \boldsymbol{W}} = 2\boldsymbol{W}, \qquad \frac{\partial Tr(\boldsymbol{V}^T\boldsymbol{V})}{\partial \boldsymbol{V}} = 2\boldsymbol{V}. \tag{15}$$

The Frobenius norm of matrices is also used to measure the size of the parameters in Algorithm 1.

5.1.2. Euclidean Closeness

The second option that we explore for defining g is the interaction given by the *closeness*. In symbols, we define

$$\begin{aligned}g^{cl}(\boldsymbol{c},\boldsymbol{p}) &= -\left\|\boldsymbol{W}\boldsymbol{c} - \boldsymbol{V}\boldsymbol{p}\right\|^2\\ &= -\left\|\boldsymbol{W}\boldsymbol{c}\right\|^2 - \left\|\boldsymbol{V}\boldsymbol{p}\right\|^2 + 2\langle \boldsymbol{W}\boldsymbol{c}, \boldsymbol{V}\boldsymbol{p}\rangle\\ &= -\left\|\boldsymbol{W}\boldsymbol{c}\right\|^2 - \left\|\boldsymbol{V}\boldsymbol{p}\right\|^2 + 2g^{in}(\boldsymbol{c},\boldsymbol{p}). \qquad (16)\end{aligned}$$

Notice that, comparing with the utility function defined in (Eq. 12), now we add more summands to the equation. The new utility, g^{cl}, includes the weighted sum of all monomials of degree 2 formed with variables taken from the description of consumers ($\boldsymbol{c}$) or products ($\boldsymbol{p}$). Of course, to guarantee this, we need to add one constant component (with value 1 for instance) to the vectorial representation of consumers and products, see (Eq. 13).

The derivatives needed to implement the learning algorithm are the following:

$$\begin{aligned}
\frac{\partial g^{cl}(\boldsymbol{c},\boldsymbol{p})}{\partial \boldsymbol{W}} &= -\frac{\partial (\boldsymbol{W}\boldsymbol{c})^T(\boldsymbol{W}\boldsymbol{c})}{\partial \boldsymbol{W}} + 2\frac{\partial g^{in}(\boldsymbol{c},\boldsymbol{p})}{\partial \boldsymbol{W}} \\
&= -\boldsymbol{W}(2\boldsymbol{c}\boldsymbol{c}^T) + 2\boldsymbol{V}\boldsymbol{p}\boldsymbol{c}^T, \\
\frac{\partial g^{cl}(\boldsymbol{c},\boldsymbol{p})}{\partial \boldsymbol{V}} &= -\frac{\partial (\boldsymbol{V}\boldsymbol{p})^T(\boldsymbol{V}\boldsymbol{p})}{\partial \boldsymbol{V}} + 2\frac{\partial g^{in}(\boldsymbol{c},\boldsymbol{p})}{\partial \boldsymbol{V}} \\
&= -\boldsymbol{V}(2\boldsymbol{d}\boldsymbol{d}^T) + 2\boldsymbol{W}\boldsymbol{c}\boldsymbol{p}^T.
\end{aligned}$$

We use the same regularization than in the case of the utility defined in terms of the inner product.

The advantage of this definition of g is that the visual semantics is more easy to appreciate. The Euclidean representation of consumers and products are closed or further according with the preferences. The inner product is a simpler equation, but it is harder to visualize.

5.2. Tensor Product

The full description of the utility functions presented in the last subsection (Eq. 14) can be seen as a particular case of a linear function in the tensor product of consumers and products. In symbols,

$$g^{\otimes}(\boldsymbol{c},\boldsymbol{p}) = \langle \boldsymbol{w}, \boldsymbol{c} \otimes \boldsymbol{p} \rangle, \tag{17}$$

where $\boldsymbol{w}$ is a vector in the Euclidean space of dimension $|Con| \times |Prod|$. If we use a pair of indexes to refer to the components of $\boldsymbol{w}$, the previous equation can be written as

$$g^{\otimes}(\boldsymbol{c}, \boldsymbol{p}) = \sum_{r,s} \boldsymbol{w}_{rs} \boldsymbol{c}_r \boldsymbol{p}_s. \tag{18}$$

Once more, this expression includes all the terms of (Eq. 14) provided that a constant component is included in vectors $\boldsymbol{c}$ and $\boldsymbol{p}$.

It is important to emphasize that the number of parameters to be learned in this approach is considerably more than in the factorization cases provided that the value of k (the dimension of the Euclidean space is small). Again, we may learn these parameters, the components of $\boldsymbol{w}$, using the Algorithm 1. For this purpose, we only need to compute the derivative

$$\frac{\partial g^{\otimes}(\boldsymbol{c}, \boldsymbol{p})}{\partial \boldsymbol{w}} = \boldsymbol{c} \otimes \boldsymbol{p}, \tag{19}$$

and the derivative of the regularization summand, that is given by

$$\frac{\partial r(\boldsymbol{w})}{\partial \boldsymbol{w}} = 2\boldsymbol{w}. \tag{20}$$

The Algorithm 1 learns the parameter $\boldsymbol{w}$ of (Eq. 17) using a SGD. The *size* of the parameter is the Euclidean norm of $\boldsymbol{w}$. This approach is then equivalent to a *Support Vector Machine* (SVM) used to learn to rank. In the experimental section we will denote this learning algorithm as SVM$^{\otimes}$.

6. Experimental Results

In this section we report a set of experiments carried out to show the performance of the proposals of this paper. First we present some implementation details of Algorithm 1. Then we introduce the datasets used in the experiments to report the accuracy obtained with each utility function (let us recall that in all cases the learning algorithm is the same). Finally, we show some graphical representations obtained as side effect of the learning process to illustrate the visualization possibilities of the factorization approaches when used to learn consumer preferences.

6.1. Implementation Details

The implementation of the Algorithm 1 was done using *Pegasos* as a model, see (Shalev-Shwartz et al., 2011). Thus, the learning rate follows the equation

$$\gamma = \frac{\gamma_0}{1 + \gamma_s(n-1)}.$$

To avoid many parameters to be adjusted, we fixed $\gamma_0 = 1$, and $\gamma_s = 0.01$. The radius (Section 4) was also fixed: $R = 1$. As usual, n is the ordinal of the iteration.

To update the model learned by the algorithm we used a *mini batch* strategy, averaging the updates every time that 10% of the training examples were processed.

The only adjustable parameter in the algorithm was the *regularization parameter*. We made an internal grid search to determine the best option in the set

$$\nu \in \{10^i : i = -1, \ldots, -10\}$$

using a 2-fold cross validation repeated 3 times on the training set.

Finally, the algorithm stops when the size of the difference of parameter θ in two consecutive iterations is smaller than 10^{-6} or the number of iterations is 5000.

Task	$\lvert D^{+}_{or} \rvert$
Acceptability	3084
Flavor	3080
Tenderness	3313

Table 1: Sizes of the datasets used in the experiments. $|D^{+}_{or}|$ stands for the number of Preference Judgments (Eq. 4). The number of consumers is 392 and there are 307 items

6.2. Datasets

The dataset used in this paper comes from a study carried out to determine the features that entail consumer acceptance of beef meat from seven Spanish breeds (Gil et al., 2001; Sañudo et al., 2004; del Coz et al., 2005; Díez et al., 2005, 2006; Bahamonde et al., 2007). Each piece of meat was described by: the weight of the animal, ageing time, breed, 6 physical features describing its texture and 12 sensory characteristics rated by 11 different experts (132 ratings). The dataset has 307 different items.

In each testing session, 4 or 5 pieces of meat were tested and a group of consumers were asked to rate (on a scale of 1 to 10 points) three different aspects: *tenderness*, *flavor* and overall *acceptance*. The number of consumers involved in this panel was 392. The features of consumers are just sex, age and job.

The preferences expressed by consumers were represented in a dataset of preference judgments like D^{+}_{or} where each input $\boldsymbol{x}$ is the concatenation of the feature description of the item and of the user. We only considered pairs where the preferences of the consumer were strictly different for two items. Thus, in each dataset the number of preference judgments is slightly different.

Table 1 reports the number of preference judgments for each learning task.

The data was preprocessed. The discrete features were binarized in the whole dataset. On the other hand, the continuous features were standardized in each training set; the mean and standard deviation of training data were used to standardize the test set.

6.3. Results and Discussion

To estimate the accuracy of the utility functions learned, we used cross validation in the D_{or}^{+} versions of acceptability, flavor and tenderness.

In addition to feature descriptions of consumers and items, we added a binary *identification* of them. That is to say, each object (consumer or item) includes in its description a vector of dimension the number of objects; in that vector all components are 0 but the one with index the ordinal of the object that has value 1.

To check the role played by these identifiers, we considered two different versions of each dataset: with and without identifiers. In preference learning, sometimes we do not have any feature description of items or consumers, then we can only use such identifiers.

To ease of reading, let us put a simple example. If we have only 3 consumers, their representations can be the following. With id codes the consumers are presented by

$$\begin{aligned}\text{consumer}_1 &= (1, 0, 0, sex_1, age_1, job_1),\\ \text{consumer}_2 &= (0, 1, 0, sex_2, age_2, job_2),\\ \text{consumer}_3 &= (0, 0, 1, sex_3, age_3, job_3).\end{aligned}$$

Dataset	SVM$^{\otimes}$	g^{cl} 2	g^{cl} 10	g^{cl} 100	g^{in} 2	g^{in} 10	g^{in} 100
Acceptability							
no ID	28.2	28.6	26.4	25.4	31.7	26.8	26.5
with ID	21.9	26.9	18.7	16.2	27.5	18.8	16.1
Flavor							
no ID	30.7	35.2	29.9	29.7	36.2	28.5	28.6
with ID	24.6	32.9	21.4	19.9	31.5	20.4	18.1
Tenderness							
no ID	25.5	26.4	24.3	25.0	28.5	23.9	23.9
with ID	21.1	24.3	17.5	15.6	26.4	18.2	16.2

Table 2: Percentages of misclassified preference judgments estimated with 10-fold cross validation using internal grid search for the parameters of the learners. Columns labeled by 2, 10 and 100 report the scores of factorizations obtained with that value of k. The testing was carried out in each fold while training was performed in the remaining 9

Without identification codes, we drop the first three binary components and each consumer will represented only by their sex, age and job. Of course, analogous representations can be used for products.

The first block of experiments used a 10-fold cross validation. Systems were trained using 9 folds and the test was performed on the remaining fold. The scores are reported in Table 2.

We observe that the performance of the SVM that uses the tensor product (SVM$^{\otimes}$) is worse than the performance of the factorization methods (g^{in} and

	g^{cl}			g^{in}		
Dataset	2	10	100	2	10	100
Acceptability						
no ID	39.0	38.3	38.7	37.9	38.0	38.0
with ID	38.7	37.4	37.4	37.7	37.3	36.6
Flavor						
no ID	43.5	43.4	43.0	43.7	43.1	42.5
with ID	43.1	42.6	42.9	43.2	41.3	40.3
Tenderness						
no ID	36.5	35.0	35.2	35.3	34.8	35.2
with ID	35.9	34.7	34.1	35.5	34.6	34.5

Table 3: Percentages of misclassified preference judgments estimated with 10-fold cross validation using internal grid search for the parameters of the learners. Columns labeled by 2, 10 and 100 report the scores of factorizations obtained with that value of k. The training was carried out in each fold while testing was performed in the remaining 9

g^{cl}) that are really quite similar. Additionally, the influence of the dimension of the Euclidean space k (Eq. 9) is dramatic in factorization systems. Greater values of k provide better results. In all cases, the scores of the tensorial version are somewhere in the middle of the factorization scores with $k = 2$ and $k = 10$.

In all cases the use of identifiers improves considerably the scores. In some case the difference is 10 points better with identifiers than without them. The reason is that some items or consumers in test sets were also known in training stage. But this is the case in many sensory data studies. Sometimes the number of options that a food industry is considering for a new product is the whole set of items both in training and in test. On the other hand, if we want to model the assessments of a selected panel of consumers, they must be present in training and in test examples.

In the experiments reported in Table 2, 90% of preference judgments are in the respective training set; therefore, most of the consumers and items in each respective test set appear in the training set too. To check the effect of the appearance of already known objects, and also to check the effect of the number of training examples, we performed two additional experiments. However, in this case we used only factorization systems since the performance of the tensorial systems was very poor. In this way, first we report the experiments carried out with 10-fold cross validation by using each fold as training set and the remaining 9 as test. The results are shown in Table 3.

The results are substantially worse, as the number of training examples is very small. Nevertheless, the impact of the identifiers of consumers and items is beneficial in all cases but one, although the increase in accuracy is smaller than in the experiments reported in Table 2. On the other hand, again we realize that higher values of k give rise to better performance.

Finally, in Table 4 we report an intermediate setting. Now we use only two folds; therefore, half of the items and consumers in the test set already appeared in the training set. As expected, the results are better than those of Table 3, but worse than the scores shown in Table 2. In this case, for $k = 100$, the error is mostly below 25% with identifiers, and around 30% without identifiers. The role of k is again of paramount importance.

	g^{cl}			g^{in}		
Dataset	2	10	100	2	10	100
Acceptability						
no ID	33.0	30.9	31.5	31.2	31.5	31.6
with ID	30.6	26.1	24.7	31.3	28.4	23.6
Flavor						
no ID	37.1	33.3	33.4	37.6	33.9	32.1
with ID	34.9	29.1	28.7	36.3	28.2	24.5
Tenderness						
no ID	29.4	27.8	28.3	28.9	29.1	28.9
with ID	27.1	25.7	24.1	28.3	24.4	23.7

Table 4: Percentages of misclassified preference judgments estimated with 2-fold cross validation using internal grid search for the parameters of the learners. Columns labeled by 2, 10 and 100 report the scores of factorizations obtained with that value of k

6.4. Visualization of Preferences

The graphical possibilities of factorization methods, in addition to good prediction scores, provide also some interesting applications. In particular, visualization is very natural when the Euclidean space has up to 3 dimensions. But another application is clustering in order to find groups of consumers with similar tastes or collections of items with similar appreciations by consumers.

In this subsection we illustrate these applications in sensory data analysis. To create the subsequent visualizations we used all available data with identities, applying Algorithm 1 with g^{cl} and $k = 2$. The idea is to obtain pictures where the proximity of one item and one consumer is the utility that represents the preference.

The resubstitution error in *acceptability* is 15.27%, in *flavor* is 18.47%, and in *tenderness* is 13.91%.

In the graphs, the small dots represent consumers located in $\mathbb{R}^2$ according to their ratings of *acceptability* (Figure 1), *flavor* (Figure 2) and *tenderness* (Figure 3) respectively.

According to the literature about sensory preferences of beef meat, (Gil et al., 2001; Sañudo et al., 2004; del Coz et al., 2005; Díez et al., 2005, 2006; Bahamonde et al., 2007), the most important features that explain the preferences of consumers are *ageing* and *intramuscular fat* (intrafat for short). These are discrete features. Ageing has 3 different values: 1, 7 and 21 days. And intrafat was discretized to obtain 3 options: low, medium and high.

Thus, in the same graph of consumers, we represented the average item with each value of these important features. This is a kind of *tag* in the sense used in (Chen et al., 2012; Moore et al., 2012) of the feature values.

The left part of Figure 1 is a Voronoi diagram of the space where seeds are the centroids of the Euclidean representation of the possible values of ageing. The lowest ageing values, 1 and 7 have centroids very close, and what it is really interesting is the split between the low (1 or 7 days) and high (21 days). In the right hand side of the figure, the centroids of items with medium or high intrafat are near and provide a clear split with consumers that prefer low intrafat values. Notice that the split due to ageing and intrafat are almost

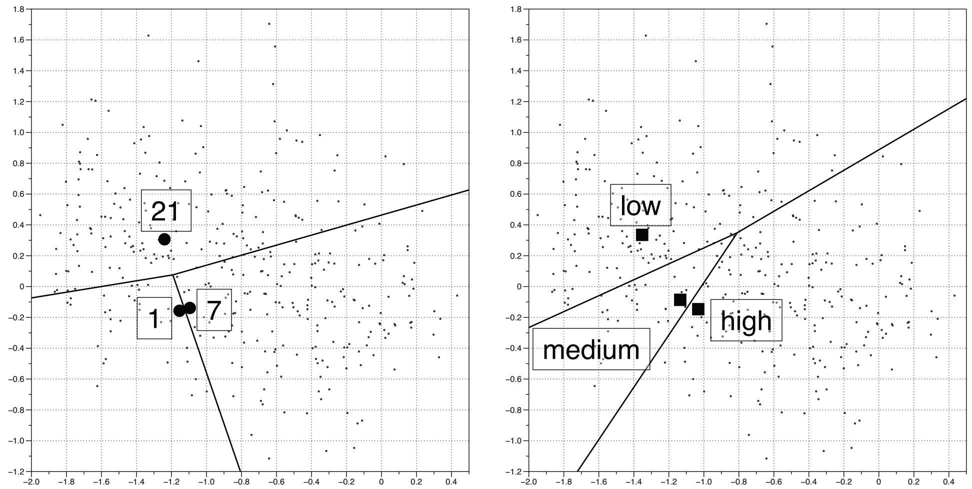

Figure 1: Consumers represented according to their ratings in *acceptability*. Voronoi diagram whose seeds are the centroids of items with different values of ageing (left) and intrafat (right)

the same. That is to say, consumers that like meat with 21 days of ageing also prefer meat with low intrafat values.

In Figure 2 the position of consumers is different with respect to the previous picture, now the feature rated by consumers is flavor. In this case the relevancy of ageing (left hand side of the figure) is clear. Consumers mostly prefer the flavor of meat after 21 days of ageing. Notice that the relative position of the centroids is increasing from left to right. According to intrafat, flavor divides consumers in those that prefer low or medium (their centroids are quite near) and those that prefer the flavor of meat with high intrafat. There are two market segments according to intrafat when the flavor is the target feature.

Finally, Figure 3 depicts consumers located in $\mathbb{R}^2$ according to their rat-

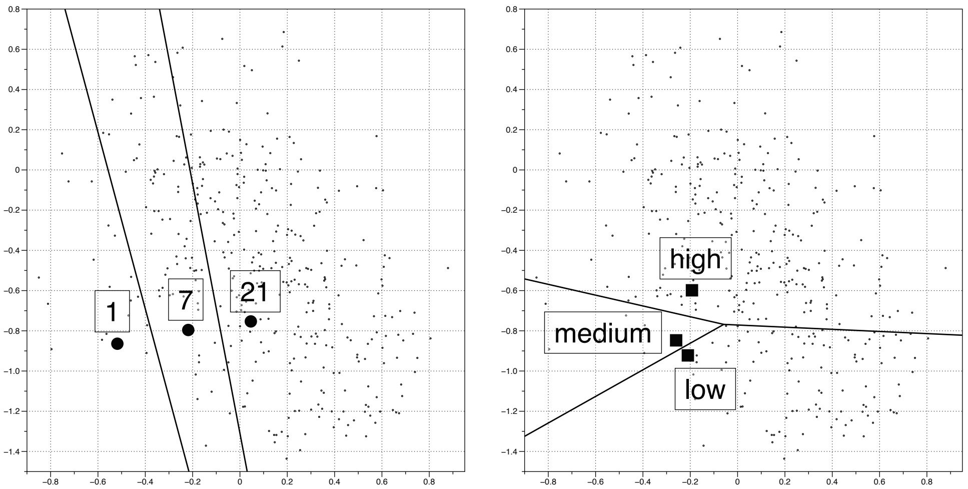

Figure 2: Consumers represented according to their ratings in *flavor*. Voronoi diagram whose seeds are the centroids of items with different values of ageing (left) and intrafat (right)

ings of tenderness. In this case, the centroids are clearly separated. When considering centroids of ageing values we appreciate that 21 is the value more associated with tenderness, this is a well known fact since the ageing is closely related with physical measures of *softness* in meat.

7. Conclusions

We have presented factorization approaches to learning and visualizing preferences of consumers about a kind of products. The models learned are more accurate than existing tensorial approaches that typically use a SVM. The framework presented in this paper includes at the same time factorization and tensorial methods; both cases use the same learning algorithm with a different equation as the goal to optimize. Then, the accuracy of the model

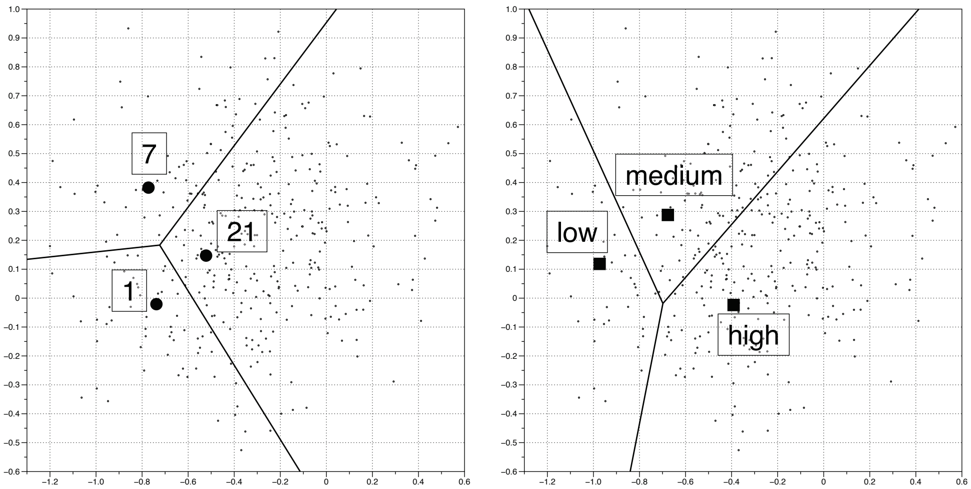

Figure 3: Consumers represented according to their ratings in *tenderness*. Voronoi diagram whose seeds are the centroids of items with different values of ageing (left) and intrafat (right)

can be explained in terms of the number of parameters to learn. Factorization models are obtained with two embeddings and need substantially less parameters than tensorial approaches.

Additionally, embeddings can be seen as Euclidean representations of both consumers and products. The closeness of these representations have a straightforward semantics. Hence, consumers' clusters can be seen as market segments, and products clusters are groups of similar items with respect to consumer tastes.

As in any other knowledge-based system, we observed that the available knowledge about consumers and products is of prime importance. If the identifiers of consumers and products are included, the accuracy of the hypothesis learned is dramatically improved. However, if only identifiers are included, there is a drawback that must be considered: no predictions can be made for new (unknown) consumers and/or products. This is the main limitation of the method, although the method presented here is flexible enough to be able to use the available knowledge.

The overall approach presented in this paper can be extended to other application fields. The requirements include situations where the the interaction of two vectors determines a class or an amount endowed with some kind of ordering. This is the case of *recommender systems* or in general *matrix completions*, well-known applications of embeddings or matrix factorizations. What we emphasize here is the graphical properties of the Euclidean representations. Then, it is possible to learn similarities of objects with respect to their behavior with a class. The applications include *direct marketing* and *fraud detection*.

To check the validity of the proposal we used a set of experiments carried out with real data of sensory analysis of beef meat according to consumer preferences. Factorization methods outperform tensorial SVM. On the other hand, the Euclidean representations obtained in these datasets emphasize the relevance of some well-known traits involved in consumer preferences.

The software used in the experiments can be downloaded from this[1] website.

[1] We will provide a link to download the implementation in the final version of the paper

Acknowledgments

The research reported here is supported in part under grant TIN2011-23558 from the Ministerio de Economía y Competitividad, Spain. The paper was written while A. Bahamonde was visiting Cornell University with grants *Movilidad Campus de Excelencia Internacional* (Universidad de Oviedo) and from *Programa Nacional de Movilidad de Recursos Humanos* (Ministerio de Educación, Cultura y Deporte, Spain).

References

Agarwal, D., Chen, B.-C., 2009. Regression-based latent factor models. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 19–28.

Bahamonde, A., Díez, J., Quevedo, J., Luaces, O., del Coz, J., 2007. How to learn consumer preferences from the analysis of sensory data by means of support vector machines (SVM). Trends in Food Science & Technology 18 (1), 20–28.

Basilico, J., Hofmann, T., 2004. A joint framework for collaborative and content filtering. In: Proceedings of the 27^{th} Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 550–551.

Bayer, I., 2015. fastfm: A library for factorization machines. arXiv preprint arXiv:1505.00641.

Chen, S., Moore, J., Turnbull, D., Joachims, T., 2012. Playlist prediction via metric embedding. In: Proceedings of the 18^{th} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 714–722.

Chen, T., Zheng, Z., Lu, Q., Zhang, W., Yu, Y., 2011. Feature-based matrix factorization. Tech. rep., Apex Data & Knowledge Management Lab, Shanghai Jiao Tong University. arXiv:1109.2271.

del Coz, J. J., Bayón, G. F., Díez, J., Luaces, O., Bahamonde, A., Sañudo, C., 2005. Trait selection for assessing beef meat quality using non-linear SVM. In: Advances in Neural Information Processing Systems 17 (NIPS '04). pp. 321–328.

Díez, J., Del Coz, J., Bahamonde, A., Sañudo, C., Olleta, J., Macie, S., Campo, M., Panea, B., Albertí, P., 2006. Identifying market segments in beef: Breed, slaughter weight and ageing time implications. Meat science 74 (4), 667–675.

Díez, J., del Coz, J., Sañudo, C., Albertí, P., Bahamonde, A., 2005. A kernel based method for discovering market segments in beef meat. Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases: ECML/PKDD 2005, 462–469.

Du, C., Zhe, S., Zhuang, F., Qi, Y., He, Q., Shi, Z., 2015. Bayesian maximum margin principal component analysis. In: Twenty-Ninth AAAI Conference on Artificial Intelligence.

Gil, M., Serra, X., Gispert, M., Angels Oliver, M., Sañudo, C., Panea, B., Olleta, J. L., Campo, M., Oliván, M., Osoro, K., García-Cachan, M., Izquierdo, M., Espejo, M., Martín, M., Piedrafita, J., 2001. The effect of breed-production systems on the myosin heavy chain 1, the biochemical characteristics and the colour variables of longissimus thoracis from seven spanish beef cattle breeds. Meat Science 58 (2), 181–188.

Herbrich, R., Graepel, T., Obermayer, K., 1999. Large margin rank boundaries for ordinal regression. Advances in Neural Information Processing Systems, 115–132.

Hüllermeier, E., Fürnkranz, J., 2013. Editorial: Preference Learning and Ranking. Machine Learning, 1–5.

Joachims, T., 2002. Optimizing search engines using clickthrough data. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 133–142.

Koren, Y., Bell, R., Volinsky, C., aug. 2009. Matrix Factorization Techniques for Recommender Systems. Computer 42 (8), 30 –37.

Koren, Y., Carmel, L., 2004. Robust linear dimensionality reduction. Visualization and Computer Graphics, IEEE Transactions on 10 (4), 459–470.

Moore, J., Chen, S., Joachims, T., Turnbull, D., 2012. Learning to embed songs and tags for playlist prediction. In: Proceedings ISMIR.

Ocepek, U., Rugelj, J., Bosnić, Z., 2015. Improving matrix factorization recommendations for examples in cold start. Expert Systems with Applications.

Pahikkala, T., Airola, A., Stock, M., De Baets, B., Waegeman, W., 2012. Efficient regularized least-squares algorithms for conditional ranking on relational data. Machine Learning, 1–36.

Parameswaran, S., Weinberger, K. Q., 2010. Large margin multi-task metric learning. In: Lafferty, J., Williams, C., Shawe-Taylor, J., Zemel, R., Culotta, A. (Eds.), Advances in Neural Information Processing Systems 23, NIPS. pp. 1867–1875.

Peltonen, J., Klami, A., Kaski, S., 2003. Learning metrics for information visualization. In: Proceedings of the Workshop on Self-Organizing Maps (WSOM'03). pp. 213–218.

Rendle, S., 2012. Factorization Machines with libFM. ACM Transactions on Intelligent Systems and Technology (TIST) 3 (3), 57.

Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L., 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. AUAI Press, pp. 452–461.

Rendle, S., Schmidt-Thieme, L., 2010. Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation. In: Proceedings of the third ACM International Conference on Web Search and Data Mining. ACM, pp. 81–90.

Robbins, H., Monro, S., 1951. A stochastic approximation method. The Annals of Mathematical Statistics, 400–407.

Sañudo, C., Macie, E., Olleta, J., Villarroel, M., Panea, B., Albertı, P., 2004. The effects of slaughter weight, breed type and ageing time on beef meat quality using two different texture devices. Meat Science 66 (4), 925–932.

Shalev-Shwartz, S., Singer, Y., Srebro, N., Cotter, A., 2011. Pegasos: Primal estimated sub-gradient solver for SVM. Mathematical Programming 127 (1), 3–30.

Thurstone, L. L., 1927. A law of comparative judgment. Psychological Review 34 (4), 273.

Weston, J., Bengio, S., Hamel, P., 2011. Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval. Journal of New Music Research 40 (4), 337–348.

Weston, J., Bengio, S., Usunier, N., 2010. Large Scale Image Annotation: Learning to Rank with Joint Word-Image Embeddings. Machine Learning 81 (1), 21–35.

Xing, E. P., Jordan, M. I., Russell, S., Ng, A. Y., 2002. Distance metric learning with application to clustering with side-information. In: Advances in neural information processing systems. pp. 505–512.

Yu, S., Yu, K., Tresp, V., Kriegel, H.-P., Wu, M., 2006. Supervised probabilistic principal component analysis. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 464–473.