REGULARIZED ALGORITHMS FOR RANKING, AND MANIFOLD LEARNING FOR RELATED TASKS

By

Giorgos Zacharia

SB Mathematics (1998)
SB Computer Science, Minor in Economics (1998)
SM Media Arts and Sciences (1999)
Massachusetts Institute of Technology

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2009

Signature of Author ....
Department of Electrical Engineering and Computer Science
January 30, 2009

Certified by..
Tomaso Poggio
Eugene McDermott Professor of Brain and Cognitive Sciences
Thesis Supervisor

Accepted by...
Terry P. Orlando
Chairman, Departmental Committee on Graduate Students

REGULARIZED ALGORITHMS FOR RANKING, AND MANIFOLD LEARNING FOR RELATED TASKS

By

Giorgos Zacharia

Submitted to the Department of Electrical Engineering and Computer Science, on January 30, 2009,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

ABSTRACT

This thesis describes an investigation of regularized algorithms for ranking problems for user preferences and information retrieval problems. We utilize regularized manifold algorithms to appropriately incorporate data from related tasks. This investigation was inspired by personalization challenges in both user preference and information retrieval ranking problems. We formulate the ranking problem of related tasks as a special case of semi-supervised learning. We examine how to incorporate instances from related tasks, with the appropriate penalty in the loss function to optimize performance on the hold out sets. We present a regularized manifold approach that allows us to learn a distance metric for the different instances directly from the data. This approach allows incorporation of information from related task examples, without prior estimation of cross-task coefficient covariances. We also present applications of ranking problems in two text analysis problems: a) Supervise content-word learning, and b) Company Entity matching for record linkage problems.

Thesis Supervisor: Tomaso Poggio
Title: Eugene McDermott Professor of Brain and Cognitive Sciences

AKNOWLEDGEMENTS

I would like to thank my Thesis supervisor, Professor Tommy Poggio, and my Thesis committee members, Professors Patrick Henry Winston, and Tommi Jaakkola. Their guidance and advice were invaluable in bringing this thesis to completion.

I would like to thank Tommy for providing the stimulating environment of the CBCL, and all the CBCL members with which I interacted, or exchanged insightful ideas throughout my sporadic PhD life at MIT. I also have to thank Gadi for the Socratic discussions, and for keeping the espresso machine, and other functions at the CBCL operational.

I would like to acknowledge the support of my Open Ratings, Emporics, and Kayak friends, but this thesis would not be complete without specifically acknowledging the help of Olga Simek, Costas Boussios, Alexandros Kyriakides, and Theodoros Evgeniou.

I dedicate my thesis to my family: my parents Maro and Christakis, my siblings Dr Margarita Zachariou, and soon to be Dr Yiannis Zachariou, and my wife Snejina, and daughter Maria.

TABLE OF CONTENTS

INTRODUCTION

Motivation

The continuous explosive availability of computers and the internet has changed the way humans interact with information. The excess amount of information has made a difficult and time consuming task the human processing and filtering through the available information to find what human users are looking for. When users search for information, they would like to receive the most likely results to satisfy their query first.

Internet search engines alleviate the problem for users when searching for unstructured keyword matches by trying to present the most reputable or through personalization, the most appropriate results for that user first. The motivation for this thesis is to alleviate the information overload problem when the users search for structured, multi-attribute products, documents or other artifacts of information. When the user searches for the ideal multi-attribute product, like a car configuration, a restaurant with ratings, an LCD screen or other electronics, with multiple specifications, and other complex purchasing decisions, where price is not the only driving factor, the users expect to receive the candidate products in the order of their preference. Also, when a user reads through a long article on their smart phone, they prefer to read a summary in the smaller screen, rather than a long multi-page article. When the user expects to read a summarized article, they again expect to receive the important information to them first. There are many similar problems such as these ones, where the information overload problem can be alleviated with intelligent ranking algorithms, so that the human user gets to the right information first.

Ranking as an approach to Information Overload

This thesis investigates primarily practical applications of Preference Modeling. When human users evaluate multi-attribute products, they expect the computer system to present those products in a rank order that is most likely to represent their expected satisfaction from each product configuration. Otherwise the human user will have to browse through multiple configurations of the product, in case there is a better configuration for their taste, several pages later. Many such preference modeling problems can be approached as ranking learning problems, where the objective of the computer system is to learn the representation function of the internal utility function of the users. If the human users have heterogeneous preferences, and therefore different evaluation behaviors, we may build systems that learn personalized ranking functions that try to predict the choices of individuals separately.

Likewise, we can use ranking algorithms as an approach to different Information Retrieval problems. When we build a system for automated summarization of a news article, we expect to rank the information, and concepts of the news article, and select the highest ranked ones to keep in the automatically generated summary. A practical way to build such automated summarization systems is to identify the sentences within the article that have the most important (or highest ranking) content, and construct a summary by concatenating the important sentences, in the order they originally appeared in the full text of the article.

Another information overload problem, where intelligent ranking algorithms can be valuable is the task of record linkage between databases on fuzzy matches of fields. For example, when a corporation tries to standardize the information of customers, or suppliers in their Enterprise Resource Planning (ERP) systems, they need to identify duplicate records, and match the remaining to standardized reference files. With an intelligent ranking algorithm, the human user can accept high confidence matches automatically, and then analyze lower confidence matches faster, by having the candidates presented in order of likelihood of a match.

Ranking algorithms have also been used in other practical domains, where we will not deal with in this thesis. Such domains include Information Extraction problems, such as identification of boundaries in named-entity extraction. (Collins, 2002). Also, multiclass classification problems can be seen as a case of ranking problems, where the label with the highest ranking of confidence is assigned as the likely label for a particular example. These areas of ranking problems can be future directions of research for the ideas presented in this thesis.

Outline of the Thesis

This thesis describes an investigation of regularized algorithms for ranking problems for user preferences and information retrieval problems, and an investigation of regularized manifold algorithms to appropriately incorporate data from related tasks. The second investigation was inspired by personalization challenges in both user preference and information retrieval ranking problems. The thesis consists of four chapters.

The first chapter investigates regularized ranking algorithms for user preference modeling. The algorithms are evaluated on a standard widely available artificial data set framework used by the market research community. The same chapter introduces a simple meta algorithm for combining aggregate information to train better performing individual models.

The second chapter expands on the ideas of the first one by formulating the ranking problem of related tasks as a special version of semi-supervised learning. We investigate how to incorporate instances from related tasks, with the appropriate penalty in the loss function to optimize performance on the hold out sets. We present a regularized manifold approach that allows us to learn a distance metric for the different instances directly from the data. We present experiments on real datasets of user preference modeling problems, and a benchmark dataset for multitask learning algorithms, the Inner London Examination Authority (ILEA).

The third and fourth chapters present two text analysis applications of ranking problems. The third chapter investigates ranking problems for information retrieval. We first introduce a set of statistical and syntactical information features that we will use to learn to identify content words from Yahoo news articles. The results of the binary classification predictions (whether it is a content word, or not) are used for two tasks: Image retrieval, based on the captions associated with the same images in previous articles, and article summarization. We evaluate both tasks on both hold out sets, and with psychophysics experiments. We then repeat the same tasks by reformulating the content word identification as a ranking problem.

The fourth chapter studies the problem of company entity matching. Different corporate and demographic databases identify corporate entities by name and address, and sometimes telephone numbers and other contact information. Since many of these records are created manually, or based on custom reference files, the quality of these databases degrades over time, with duplicated entries, caused by misspellings, abbreviations, and other user generated inconsistencies, and outdated records of companies that have changed locations, renamed themselves, merged, or otherwise ceased operations. Also, when someone tries to enrich such corporate databases with reference files, with additional company data, like industry codes, financials, or other corporate descriptors, the data representation in the two sources make the process impossible to achieve with exact keyword matching, or join operations in relational databases. We present supervised learning algorithms that express the company entity identification problem as a ranking problem. We train our algorithms on a small subset of user generated examples to automate the joining of multi-million record reference files.

Contributions of the Thesis

To summarize, the contributions of the thesis consist of two parts. First we examine ranking problems, as supervised learning problem. We generalize metric based, and choice based ranking problems as a supervised learning binary classification problem of pairwise feature differences. The binary classifier of parwise differences is trained to identify the winning vs. the losing configuration of the two examples compared. We develop preference modeling, supervised content-word extraction, applied to text summarization, and a record linkage application, using this framework of ranking applications.

The second contribution of the thesis is motivated by the problem of learning from related tasks, specifically in the domain of user preference modeling. Such algorithms are useful when we have few examples per tasks, but many relatively similar tasks, which can inform the training of the task specific models. We develop algorithms that utilize regularized manifold learning, to account for the similarity of the foreign task data. We run experiments on real user datasets for preference modeling, and a benchmark dataset for multi-task learning (ILEA), on which our proposed algorithm outperforms the currently reported algorithms in the literature of multi-task learning.

CHAPTER 1: REGULARIZED RANKING FOR USER PREFERENCE MODELING

Introduction

The amount of data capturing preferences of people for particular products, services, and information sources, has been dramatically increasing in recent years largely due for example to electronic commerce. Traditional preference modeling methods such as conjoint analysis (Carroll & Green, 1995), (Green & Srinivasan, 1978), (Green & Srinivasan, 1990)have been used for many preference modeling applications (Wittink & Cattin, 1989) typically with data gathered under controlled conditions such as through questionnaires. However, much of the available information today about choices of people, such as scanner or clickstream data, is not gathered in such a controlled way and therefore is more noisy (Cooley, Srivastava, & Mobasher, 1997), (Kohavi, 2001). It is therefore important to develop new preference modeling methods that are (a) highly accurate, (b) robust to noise, and (c) computationally efficient in order to handle the large amounts of choice data available.

Several statistical approaches to information retrieval, and other ranking problems like preference modeling, assume that there is explicit metric information available (Cui & Curry, 2003), or other side information like transitive rankings (Herbrich, Graepel, & Obermayer, 1999), or frequency of clicks (Joachims, 2002). In choice based data, we only know which combination is the highest ranking, among the available options.

The objective of choice based conjoint analysis, for a market researcher, is to determine the most preferred combination of attributes, typically for new product development (Toubia, Simester, Hauser, & Dahan, 2003)(Toubia, Hauser, & Simester, 2004). In this chapter we present regularized learning methods that can learn such user preferences from choice based data. We compare our SVM based methods with logistic regression (Ben-Akiva & Lerman, 1985) (Louviere, Hensher, & Swait, 2000), Hierarchical Bayes (HB) (DeSarbo & Ansari, 1997)(Allenby, Arora, & Ginter, 1998)(Arora, Allenby, & Ginter, 1998), and the polyhedral estimation methods of (Toubia, Hauser, & Simester, 2004) using simulations as in (Arora & Huber, 2001); (Toubia, Hauser, & Simester, 2004). We show experimentally that the SVM based methods are more robust to noise than both logistic regression and the polyhedral methods, to either significantly outperform or never be worse than both logistic regression and the polyhedral methods, and to estimate nonlinear utility models faster and better than all methods including HB.

Individual users may have different preferences, expressed as different utility functions. The heterogeneity of a population is an informal measure of the variance of the users' utility functions. In this chapter we focus on the problem of learning each individual's preferred combination, which is equivalent to learning personalized ranking models, in an information retrieval task. Therefore, we also extend the SVM for ranking algorithms with a combined classifier approach that handles heterogeneity across many individuals, with promising results compared to HB. We learn each user's utility function as combination of the learned partworth parameters from an individual user's data, and the partworth parameters learned from the aggregate data. Our extension does not involve an intermediate step of clustering (Cohn, Caruana, & McCallum, 2003), or estimating of a distance metric between individuals (Schultz & Joachims, 2003). The combined SVM algorithm tries directly to minimize the number of erroneous choices on a validation set, given the estimated individual and aggregate partworh parameters.

This chapter makes three contributions to the ranking learning problem: We demonstrate the robustness and computational efficiency of regularized methods in general and SVM in particular to ranking problems with noisy data and non-linearities. In addition, we introduce positivity constraints through virtual examples, a prior knowledge commonly available in many ranking estimation problems. Finally, we introduce a combined classifier approach that allows us to exploit information from the aggregate data set. The weighted aggregate information (estimated through cross validation) improves the individual specific models.

Related Work

This chapter is related to preference modeling research in the market research, and the machine learning community, and to information retrieval research. As we will show later in the chapter, the preference modeling problem is equivalent to a ranking learning problem, like many information retrieval problems (Collins, 2002)(Herbrich, Graepel, & Obermayer, 1999)(Joachims, 2002)(Schultz & Joachims, 2003)

(Joachims, 2002) presents a method that reranks the retrieval candidates of search engines, based on observed clickthrough data. Similarly to our approach, he uses SVM to classify vectors of difference, and proves that minimizing the number of classification errors on vectors of differences is equivalent to maximizing the Average Precision (Baeza-Yates & Ribeiro-Net, 1999) in the information retrieval definition. One difference with our approach is that in the clickthrough reranking problem, there is inferred relative ranking information, that can be inferred from the presentation order of the links (links presented before some clicked link, are treated as lower ranked ones). Although the problem investigated by (Joachims, 2002) is more general than the ordinal regression one studied by (Herbrich, Graepel, & Obermayer, 1999), (which requires explicit rankings, with same scale), it is still more restrictive than the one investigated in this chapter: namely winner-loser comparisons, without inferred relative ranking constraints. Also, our methods take advantage of prior knowledge, by incorporating positivity constraints in the training data set. In our experiments, this prior information affects the test set accuracy significantly. The features used by Joachims can all infer the same kind of positivity constraints (since they are all similarity metrics, or other search engines rankings), and we believe they can also improve on the overall performance of the meta-search experiment.

(Collins, 2002) presents a similar approach for reranking algorithms for named-entity extraction, using exponential loss functions on attribute differences. The algorithms presented by Collins use features (which are also transformed to vectors of differences) that include the ranking of a baseline maximum-entropy tagger, and an additional set of global hypotheses. The proposed re-ranking algorithms optimize the weight of the information form the two sources using a validation set. This approach is related to our SVM-Mix approach, where we learn individual specific models by using both the prediction of the aggregate model, and the features specific to the individual. Also, Collins adds two training examples for every comparison (eg "A" is better than "B", and "B" is worse than "A"), something we also utilize, and have observed that the addition of both examples helps improve the accuracy in the binary classification formulation of our ranking problem. The named-entity reranking algorithms also use features that imply positivity constraints (at least the maximum-entropy baseline), which, if added properly in the training set, could again improve performance accuracy.

(Schultz & Joachims, 2003) use SVMs for vectors of differences classification, to learn a distance metric for different classes of documents. Their approach successfully learns weighted Euclidean distances that improve similarity predictions, by just implicit information. The approach is similar to the manifold regularization (Belkin & Niyogi, 2004), semi-supervised clustering (Blum & Mitchell, 1998), and transductive learning (Gammerman, Vapnik, & Vovk, 1998). These approaches are also relevant to the problem of estimating the heterogeneity of the user set, and utilizing training information from the other users, based on a weighted distance metric. In this chapter, we do not attempt to solve the distance metric estimation. Instead, we employ a linear combined classifier approach that gives promising results.

The market research community has traditionally approached utility estimation problems through function estimation. Conjoint analysis is one of the main methods for modeling preferences from data (Carroll & Green, 1995)(Green & Srinivasan, 1978). A number of conjoint analysis methods have been proposed - see for example (Sawtooth Software, 2009). Since the early 1970s conjoint analysis continues to be a very popular approach with hundreds of commercial applications per year (Wittink & Cattin, 1989). In conjoint analysis designing questionnaires is a central issue (Arora & Huber, 2001)(Kuhfeld, Tobias, & Garratt, 1994)(Oppewal, Louviere, & Timmermans, 1994), which, as mentioned above, we do not address here.

Within the Discrete Choice Analysis area users' preferences are modeled as random variables of logit models (Ben-Akiva & Lerman, 1985)(Ben-Akiva, et al., 1997)(McFadden, 1974)(McFadden, 1986). Both conjoint analysis and discrete choice methods have always faced the tradeoff between model (multinomial logit models) complexity and computational ease as well as predictive performance of the estimated model. This trade off is linked to the well known "curse of dimensionality" (Stone, 1985): as the number of dimensions increases an exponential increase in the number of data is needed to maintain reliable model estimation. The SVM-ranking method we present in this chapter can handle this issue, as already shown for other applications (Vapnik, 1998).

A different approach was implemented by (Herbrich, Graepel, & Obermayer, 1999) who instead of trying to apply regression techniques for utility function estimation, they reformulated the problem as an ordinal regression estimation and used SVM to predict transitive ranking boundaries. More recently (Cui & Curry, 2003) used directly SVM for predicting choices of consumers. Our methods are similar with those in (Herbrich, Graepel, & Obermayer, 1999) and (Cui & Curry, 2003): in particular they are almost equivalent to SVM. Unlike (Herbrich, Graepel, & Obermayer, 1999) and (Cui & Curry, 2003), we focus here on choice based conjoint analysis and on the comparison with logistic regression, HB, and polyhedral estimation (Toubia, Hauser, & Simester, 2004).

Finally, recent work by (Toubia, Simester, Hauser, & Dahan, 2003) (Toubia, Hauser, & Simester, 2004) addresses the problem of designing questionnaires and estimating preference models through solving polyhedral optimization problems which are similar to the methods we discuss below. They develop methods for both metric (Toubia, Simester, Hauser, & Dahan, 2003) and choice based (Toubia, Hauser, & Simester, 2004) conjoint analysis. (Toubia, Hauser, & Simester, 2004) focus more on the design of individual-specific questionnaires while we focus on the estimation of a utility function from data. In the experiments below we only use the utility function estimation method of (Toubia, Hauser, & Simester, 2004) and not the questionnaire design method they have developed.

A key difference of our SVM-ranking approach from the method of (Toubia, Hauser, & Simester, 2004) is that in our case we optimize *both* the error on the data *and* the complexity of the solution, simultaneously by solving a standard SVM quadratic optimization problem. (Toubia, Hauser, & Simester, 2004) do not handle this tradeoff between error and complexity through a simultaneous optimization. We conjecture that the difference in performance between our method and the method of (Toubia, Hauser, &

Simester, 2004) shown in the experiments below is due to the difference between the way these two methods handle this trade off.

A Theoretical Framework for Modeling Preferences

Setup and Notation

We consider the standard (i.e. (Louviere, Hensher, & Swait, 2000)) problem of estimating a utility function from a set of examples of past choices all coming from a single individual - so from a single true underlying utility function. We also present an approach of combined classifiers to handle heterogeneity, and better learn personalized utility functions by combining information from both individual and aggregate data.

Formally we have data from n choices where, without loss of generality, the ith choice is among two products (or services, bids, etc) $\{x_i^1, x_i^2\}$. To simplify notation we assume that for each i the first product x_i^1 is the preferred one - we can rename the products otherwise. All products are fully characterized by m-dimensional vectors - where m is the number of attributes describing the products. We represent the jth product for choice i as $x_i^j = \{x_i^j(1), x_i^j(2), \ldots x_i^j(m)\}$. So the ith choice is among a pair of m-dimensional vectors. We are now looking for a utility function that is in agreement with the data, namely a function that assigns higher utility value to the first product - the preferred one - for each pair of choices. This is the standard setup of choice based conjoint analysis . (Louviere, Hensher, & Swait, 2000). Variations of this setup (i.e. cases where we know pairwise relative preferences with intensities) can be modeled in a similar way.

Support Vector Machines for Linear Utility Function Estimation

We first make the standard assumption (Ben-Akiva & Lerman, 1985); (Srinivasan & Shocker, 1973) that the utility function is a linear function of the values (or logarithms of the values, without loss of generality) of the product attributes:

the utility of a product $x = \{x(1), x(2), \ldots, x(m)\}$

is $U(x) = w_1 \cdot x(1) + w_2 \cdot x(2) + \ldots + w_m \cdot x(m)$.

We are looking for a utility function with parameters w_1, $w_2, \ldots w_m$ that agrees with our data, that is we are looking for w_1, $w_2, \ldots w_m$ such that for $\forall i \in \{1, 2, \ldots, n\}$:

$$\begin{aligned} & w_1 \cdot x_i^1(1) + w_2 \cdot x_i^1(2) + \ldots + w_m \cdot x_i^1(m) \geq \\ & w_1 \cdot x_i^2(1) + w_2 \cdot x_i^2(2) + \ldots + w_m \cdot x_i^2(m) \end{aligned} \tag{1}$$

Clearly there may be no w_f that satisfies all n constraints, since in practice the true utility function does not have to be linear and generally there are a lot of inconsistencies in data describing preferences of people. To allow for errors/inconsistencies we use slack variables, a standard approach for optimization methods (Bertsimas & Tsitsikilis, 1997). For each of the n inequality constraints (1) we introduce a positive slack variable ξ_i which effectively measures how much inconsistency/error there is for choice i, like in (Srinivasan & Shocker, 1973) (Joachims, 2002). So we are now looking for a set of parameters w_1, $w_2, \dots w_m$ so that we minimize the error $\sum_{i=1\dots n} \xi_i$ where $\xi_i \geq 0$ and satisfy for $\forall i \in \{1,2,\dots,n\}$:

$$\begin{aligned} & w_1 \cdot x_i^1(1) + w_2 \cdot x_i^1(2) + \dots + w_m \cdot x_i^1(m) \\ & \geq w_1 \cdot x_i^2(1) + w_2 \cdot x_i^2(2) + \dots + w_m \cdot x_i^2(m) - \xi_i \end{aligned} \tag{2}$$

Notice that one may require to minimize the L_0 norm of the slack variables ξ_i so that what is penalized is the number of errors/inconsistencies and not the "amount" of it. In that case the optimization problem becomes an integer programming problem which is hard to solve.

So in this simple model we are looking for a linear utility function that minimizes the amount of error/inconsistencies on the estimation data. This, however, may lead to models that over-fit the current data, are sensitive to noise, and can suffer from the curse of dimensionality - therefore are less accurate and cannot handle well choice data that involve a large number of attributes m and which are noisy (Vapnik, 1998). It is therefore important to augment this model to avoid over-fitting hence improve accuracy performance and handle noise better.

We use a model complexity control that is standard for other data analysis methods, such as for SVM (Vapnik, 1998), (Wahba, 1990), (Girosi, 2003). Intuitively, we require that constraints (1) hold (when they are feasible) with some "confidence margin": we would like to find a function that assigns to the preferred products utility which is larger than that assigned to the non-preferred products by as high an amount as possible.

The SVM-ranking method is to *simultaneously* minimize the error we make on the example data, via minimizing the slack variables ξ_i, *and* maximize the margin with which the solution satisfies the constraints. As in the case of SVM it can be shown (Vapnik, 1998) that this is achieved through the following optimization problem - for simplicity we omit the mathematical derivation and we refer the reader to (Vapnik, 1998) for it:

$$\min_{w_1,\dots,w_m,\xi_i} \sum_{i=1\dots n} \xi_i + \lambda \sum_{f=1\dots m} w_f^2$$

subject to:

$$\begin{aligned} & w_1 \cdot x_i^1(1) + w_2 \cdot x_i^1(2) + \dots + w_m \cdot x_i^1(m) \\ & \geq w_1 \cdot x_i^2(1) + w_2 \cdot x_i^2(2) + \dots + w_m \cdot x_i^2(m) + 1 - \xi_i \end{aligned} \quad (3)$$

for $\forall i \in \{1,2,\dots,n\}$, and

$$\xi_i \geq 0$$

We can rewrite the comparison constraints of (3) in such a way that the utility estimation problem becomes that of classifying vectors in the space of "differences of products". Formally, constraint:

$$w_1 \cdot x_i^1(1) + w_2 \cdot x_i^1(2) + \dots + w_m \cdot x_i^1(m) \geq w_1 \cdot x_i^2(1) + w_2 \cdot x_i^2(2) + \dots + w_m \cdot x_i^2(m) + 1 - \xi_i$$

is the same as:

$$w_1 \cdot \left(x_i^1(1) - x_i^2(1)\right) + w_2 \cdot \left(x_i^1(2) - x_i^2(2)\right) + \ldots + w_m \cdot \left(x_i^1(m) - x_i^2(m)\right) \geq 1 - \xi_i$$

If we label all vectors $\left(x_i^1 - x_i^2\right)$ ("winner - loser") with label $+1$ and all vectors $\left(x_i^2 - x_i^1\right)$ ("loser - winner") with label -1, then searching for a utility function that satisfies the comparison constraints in (3) can be seen as equivalent to searching for a function (hyperplane in the case of linear utility functions) that separates the vectors with the $+1$ labels from the ones with the -1 labels. To see this simply add the constraints:

$$(-1)\left(w_1 \cdot \left(x_i^2(1) - x_i^1(1)\right) + w_2 \cdot \left(x_i^2(2) - x_i^1(2)\right) + \ldots + w_m \cdot \left(x_i^2(m) - x_i^1(m)\right)\right) \geq 1 - \xi_i^{'}$$

and replace ξ_i with $\xi_i + \xi_i^{'}$ in the cost function. The only effect this addition has is equivalent to dividing by half the parameter λ - since the optimal ξ_i is the same as $\xi_i^{'}$ and we count $\xi_i + \xi_i^{'}$ in the cost function. The equivalent SVM classification problem solved then becomes:

$$\min_{w_1,\ldots,w_m,\xi_i\xi_i^{'}} \sum_{i=1\ldots n} \left(\xi_i + \xi_i^{'}\right) + \lambda \sum_{f=1\ldots m} w_f^2$$

subject to:

$$(+1)\left(w_1 \cdot \left(x_i^1(1) - x_i^2(1)\right) + w_2 \cdot \left(x_i^1(2) - x_i^2(2)\right) + \ldots + w_m \cdot \left(x_i^1(m) - x_i^2(m)\right)\right) \geq 1 - \xi_i$$

$$(-1)\left(w_1 \cdot \left(x_i^2(1) - x_i^1(1)\right) + w_2 \cdot \left(x_i^2(2) - x_i^1(2)\right) + \ldots + w_m \cdot \left(x_i^2(m) - x_i^1(m)\right)\right) \geq 1 - \xi_i^{'}$$

for $\forall i \in \{1,\dots,n\}$, and

$$\xi_i \geq 0$$
$$\xi_i' \geq 0$$

which is equivalent to the standard SVM optimization (Vapnik, 1998) (Cortes & Vapnik, 1995) the for the difference vectors with labels ± 1 defined above. We therefore have the following lemma:

Lemma: The solution of formulation (3) is the same as that of a support vector machine linear classification of the vectors of differences of attribute values of the compared products with labels defined as above.

Given this equivalence, the complexity control $\sum_{f=1}^{m} w_f^2$ has also the intuitive interpretation of SVM: in the case that the difference vectors are separable with a hyperplane (which is the case that the feasible space of utility functions satisfying the constraints in (3) with $\xi_i = 0$ is non-empty), the method finds the separating hyperplane that has the largest "margin" from the data (difference vectors) as shown in Figure 1. This is a standard characteristic of SVM and leads to robust to noise solutions with good predictive performance (Vapnik, 1998).

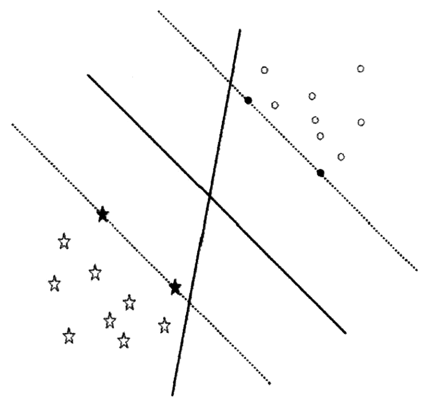


Figure 1 Separating hyperplane and optimal separating hyperplane. Both solid lines separate the two classes, circles and stars, but one leaves the closest points (filled circles and stars) at the maximum distance - within the parallel lines. These filled points are called support vectors and correspond to the "hard choices".

The trade off parameter λ that controls "how much" the constraints need to be satisfied. The equivalency to SVM classification provides some useful characteristics, namely:

- The estimation is done through fast quadratic programming optimization with box constraints, namely constraints that give only upper and lower bounds to the parameters to be estimated;
- The estimated utility function turns out to depend only on certain data - the "hard choices" which are the data touching the margin hyperplanes in Figure 1 that are automatically detected;
- The generalization to highly nonlinear utility functions - that turn out to be linear in parameters (Vapnik, 1998) - is straight forward and computationally efficient (explained below);
- The probabilistic guarantees on the future performance of the estimated model can be given under certain assumptions about the

probability distribution of the data. In particular, it can be shown that the predictive performance of the estimated models - that is, how often the estimated utility assigns higher utility to the correct product for future choices - increases as $\|w\|^2$ (which controls the confidence margin on the estimation data as discussed above) and the error $\sum_i \xi_i$ decrease and as the number of data n increases (Vapnik, 1998) [1].

---

[1] The following theorem is well known for SVM (Vapnik, 1998) (Evgeniou, Pontil, & Poggio, 2000): With probability $1-\eta$, the probability ε that a future point is misclassified by a support vector machine classification solution that makes k misclassifications on n example data and has margin $\|w\|^2$ on this data is bounded by:

$$\varepsilon \le \frac{k}{n} + \Phi\left(\|w\|^2, n, \eta\right)$$

where Φ is decreasing with n and η, and increasing with $\|w\|^2$. One can also replace k with $\sum_i \xi_i$ and use a different Φ. For simplicity we do not give the form of Φ here and refer the reader for example to (Vapnik, 1998) (Evgeniou, Pontil, & Poggio, 2000):. We note that this theorem holds only if the n data (product differences) are i.i.d., which is not necessarily the case for the preference modeling setup. It is an open question how to extend this theorem to the conjoint estimation case. This theorem currently provides only an informal motivation for the proposed approach.

Dual Parameters and Hard Choices

It turns out that like in the case of SVM the utility function estimated through (3) can be written in the form:

$$U(x)=w_i \cdot x = \sum_{i=1}^{n} \alpha_i^* \left(x_i^1 - x_i^2\right) \cdot x \qquad (4)$$

where α_i^* are the dual parameters (Bertsimas & Tsitsikilis, 1997) corresponding to the dual optimization problem of (3). The dual problem is a Quadratic Programming problem with box constraints which has a unique optimal solution and is fast to solve in practice - (Cortes & Vapnik, 1995). It can be shown (Vapnik, 1998) that for the optimal solution (4), and for SVM in general, only a few of the coefficients α_i^* are non-zero. These are the coefficients α_i that correspond to the pairs of products $\left(x_i^1, x_i^2\right)$ hard to choose from. In other words *the utility function model developed from a set of choices is specified only by the "hard" choices, which are automatically found by the SVM-ranking algorithm*. This is in agreement with the intuition that preferences are shaped by the hard choices one has to make. Moreover, although we do not deal with this issue here, intuitively one could also use this characteristic of the SVM-ranking algorithm to design questionnaires in the spirit of (Toubia, Simester, Hauser, & Dahan, 2003). For example there has been work in the area of active learning - see for example (Tong & Koller, 2000) - that can be used for this problem. It is interesting to note that the questionnaire design approach of (Toubia, Simester, Hauser, & Dahan, 2003) is similar in spirit with the active learning methods in the literature. We plan to explore this direction in future work.

Generalization to Nonlinear models

Having shown the equivalence of the utility function estimation formulation, which we outlined above with classification of the differences of attribute vectors using SVM, we can now use standard methods from the literature of SVM to estimate non-linear utility functions by finding non-linear separating surfaces in the space of difference vectors. To this purpose we use the approach of building models within Reproducing Kernel Hilbert Spaces (see for example (Wahba, 1990) (Vapnik, 1998)). These are spaces of functions which can be expressed as linear combinations of complex features (possibly infinite number of them).

Consider for example the simple case where the number of attributes m of the products is 2. When estimating a linear utility function we look for a function

$U(\mathbf{x}) = w_1 \cdot x(1) + w_2 \cdot x(2)$. On the other hand, by estimating a utility function which is a polynomial of degree 2 we can add all attribute interactions. The utility function then is

$$U(\mathbf{x}) = w_1 \cdot x(1)^2 + w_2 \cdot x(2)^2 + w_3 \cdot x(1) \cdot x(2) + w_4 \cdot x(1) + w_5 \cdot x(2)$$

which can be seen again as a linear function if we consider that the products $\mathbf{x}$ are represented using 5 attributes

$\left(x(1)^2, x(2)^2, x(1)\cdot x(2), x(1), x(2)\right)$.

It turns out that one can still estimate the nonlinear functions very efficiently even if the number of primal parameters w_f is very large - for example we include all attribute interactions (Wahba, 1990) (Vapnik, 1998)). This is done by solving the corresponding dual optimization problem, therefore always optimizing for n free parameters α_i *independent of the dimensionality* of the "data" $\mathbf{x}$ or the higher dimensional space created. The dual formulation is always a Quadratic Programming optimization problem with box constraints and number of variables (α_i) equal to n, the number of constraints in (3) (Vapnik, 1998). So the number of variables w_f in the primal formulation (3) is not important (Vapnik, 1998)) - products with a very large number of attributes, as well as highly nonlinear utility functions that, for example, include all (higher) interactions among the product attributes can be computationally efficiently estimated in a robust way.

Notice that for the dual formulation (3), all we need is the $n \times n$ matrix $\mathbf{K}$ of dot products of the data. If we can efficiently compute this matrix of dot products for the new high-dimensional representations (like the 5-dimensional one corresponding to polynomials of degree 2 we showed above), then the number of dimensions of the new space we define does not matter. This can be done through the use of a *kernel function* that defines a dot product in the high dimensional spaces called Reproducing Kernel Hilbert Spaces (RKHS) (Wahba, 1990). We explain the idea with a simple example.

It is simple to check that for the 2 dimensional products we considered above, the dot product between two vectors

$$\left(x(1)^2, x(2)^2, x(1)\cdot x(2), x(1), x(2)\right) \text{and}$$
$$\left(x'(1)^2, x'(2)^2, x'(1)\cdot x'(2), x'(1), x'(2)\right)$$

(the mappings in the 5-dimensional space of the two 2-d (initial) vectors $\mathbf{x}$ and $\mathbf{x}^{'}$) can be written (up to some constant factors) as $\left(\mathbf{x}\cdot\mathbf{x}^{'}+1\right)^2$. Indeed:

$$\begin{aligned}
&(\mathbf{x}\cdot\mathbf{x}'+1)^2 = \left(x(1)x'(1)+x(2)x'(2)+1\right)^2 = \\
&= x(1)^2\cdot x'(1)^2 + x(2)^2\cdot x'(2)^2 + 1 + \\
&+2x(1)\cdot x'(1)\cdot x(2)\cdot x'(2) + 2x(1)x'(1) + 2x(2)x'(2) = \\
&= \left(x(1)^2, x(2)^2, \sqrt{2}x(1)\cdot x(2), \sqrt{2}x(1), \sqrt{2}x(2), 1\right)\cdot \\
&\cdot\left(x'(1)^2, x'(2)^2, \sqrt{2}x'(1)\cdot x'(2), \sqrt{2}x'(1), \sqrt{2}x'(2), 1\right)
\end{aligned}$$

which is the dot product of the five-dimensional vectors (plus constant 1) defined above. The function $\left(\mathbf{x}\cdot\mathbf{x}^{'}+1\right)^2$ is a kernel function.

Using as kernel functions polynomials of degree d, namely $\left(\mathbf{x}\cdot\mathbf{x}^{'}+1\right)^d$, is equivalent to estimating a utility function that is a polynomial of degree d capturing all interactions up to order d among the product attributes. Other kernel functions can be defined (Wahba, 1990) (Vapnik, 1998) - i.e. consider function $e^{\|x-x'\|^2}$ to get highly nonlinear models.

Notice that to solve (3) for the case of utility estimation where the data are "differences of products", we need to compute the dot products of difference vectors. This is a point where the SVM-ranking methods differ from standard SVM. If we represent as $\phi(\mathbf{x})$ and $\phi(\mathbf{y})$ the high dimensional (expanded, non-linear, feature set) maps of initial products $\mathbf{x}$ and $\mathbf{y}$, then we need to compute $(\phi(\mathbf{x})-\phi(\mathbf{y}))\cdot(\phi(\mathbf{x})-\phi(\mathbf{y}))$. This can be written as

$$\begin{aligned}&(\phi(\mathbf{x})-\phi(\mathbf{y}))\cdot(\phi(\mathbf{x})-\phi(\mathbf{y}))=\\&=\phi(\mathbf{x})\cdot\phi(\mathbf{x})+\phi(\mathbf{y})\cdot\phi(\mathbf{y})-2\phi(\mathbf{x})\cdot\phi(\mathbf{y})=\\&=k(\mathbf{x},\mathbf{x})+k(\mathbf{y},\mathbf{y})-2k(\mathbf{x},\mathbf{y}).\end{aligned}$$

where $k(\mathbf{x},\mathbf{y})$ is the kernel evaluated at $\mathbf{x}$ and $\mathbf{y}$, for example $k(\mathbf{x},\mathbf{y})=(\mathbf{x}\cdot\mathbf{y}+1)^d$. So the dot product of the difference of the high dimensional vectors $\phi(\mathbf{x})$ and $\phi(\mathbf{y})$ can still be efficiently computed using only kernels.

Finally, the estimated utility function can be always using the dual parameters α^*, as in the linear case. For example, for a polynomial kernel of degree d (therefore for utility functions that are polynomials of degree d - capturing all interactions among attributes up to degree d), the estimated utility function has the form:

$$U(\mathbf{x})=\sum_{i=1}^{n}\alpha^*\left((\mathbf{x}_i^1\cdot\mathbf{x}+1)^d-(\mathbf{x}_i^2\cdot\mathbf{x}+1)^d\right) \tag{5}$$

where we just replaced dot products with kernel functions. The number of dual parameters α_i^* that are non-zero is still small - corresponding to the constraints that are not satisfied with margin more than 1 by the estimated non-linear utility function. Finally, because of the constraints on the complexity of the utility function through minimizing the norm of $\mathbf{w}$ (primal parameters), it turns out that even for highly non-linear utility functions overfitting can be avoided and the curse of dimensionality is not a problem (Vapnik, 1998). It is the norm $\|\mathbf{w}\|^2$ of the optimal solution that measures the complexity of the solution, and not the number of parameters corresponding to the dimensionality of the created feature space (Vapnik, 1998). For example SVM have been used successfully for many problems where the number of dimensions is in the thousands and the number of data small (Vapnik, 1998).

Estimating the regularization penalty

Parameter λ controls the *tradeoff* between fitting the data ($\sum_i \xi_i$) and the complexity of the model ($\|w\|^2$). There are a number of ways to choose parameter λ (Wahba, 1990) (Vapnik, 1998). For example it can be chosen so that the prediction error in a small validation set is minimized or through cross-validation (also called leave-one-out error) (Wahba, 1990). Briefly, the latter is done as follows.

For a given parameter λ we measure its leave-one-out error as follows: for each of the n choice data, we estimate a utility function using (3) only with the remaining n - 1 data and test if the estimated function correctly chooses the right product for the choice data point not used (left out). We then count for how many of the n choice points there has been an error when they were left out. This is the cross validation (leave-one-out) error for the parameter λ. We then choose the parameter λ with the smallest cross validation error.

We can use cross-validation when we can assume that the future data (choices) come from the same distribution as the data used for estimation (Vapnik, 1998). However, in conjoint analysis this may not be always the case. For example, when the estimation data come from an orthogonal design: the orthogonal design is not a sample from the probability distribution of the future choices. So formally we cannot use cross-validation with an orthogonal design. Therefore, for the experiments we describe below, we use an additional validation set approach, with the assumption that the validation data come from the same probability distribution as the future data.

In the experiments below we tuned λ using cross-validation. We chose, using line search, a λ between 0.001 and 100 (samples every order of magnitude only). Because we have a few data for each individual we use the same λ for all individuals which we chose using the average cross-validation error across all individuals.

Adding Positivity Constraints

The coefficients of a utility function, w_f in problem (3), are sometimes assumed to be positive - if not the values of the corresponding attributes can be often negated and have the corresponding coefficients be positive

(Toubia, Simester, Hauser, & Dahan, 2003). Therefore in practice it is often (but not always) important to add such constraints to the estimation of the utility function. To do so the estimation method, for example in the simple linear case, should be modified by adding to (3) the extra constraints:

$$w_f \geq 0, \forall f = 1,\ldots,m. \tag{6}$$

However such a modification makes the generalization of the method to the nonlinear case using kernels impossible (Vapnik, 1998) It is therefore not possible to add such constraints directly and still be able to estimate efficiently highly non-linear models (Vapnik, 1998).

To avoid this problem we use virtual examples (Niyogi, Poggio, & Girosi, 1998) (Scholkopf, Burges, & Vapnik, 1996). In particular, the positivity of the m parameters w_f is incorporated in the models by adding m (virtual) example difference vectors

$$\{(1, 0,\ldots, 0),(0, 1,\ldots, 0),\ldots,(0, 0,\ldots, 0, 1)\}.$$

These difference vectors correspond to pairs of products that have all attributes the same apart from one: the product with a higher value (by 1) for the one attribute the two products differ is preferred. Formally this modifies problem (3) as follows:

$$\min_{w_1,\ldots,w_m,\xi_i}\left(\sum_{i=1}^{n}\xi_i+\sum_{f=1}^{m}\xi_f\right)+\lambda\sum_{f=1\ldots m}w_f^2 \tag{7}$$

Subject to:

$$w_1 \cdot x_i^1(1) + w_2 \cdot x_i^1(2) + \ldots + w_m \cdot x_i^1(m) \geq$$
$$w_1 \cdot x_i^2(1) + w_2 \cdot x_i^2(2) + \ldots + w_m \cdot x_i^2(m) + 1 - \xi_i$$

for $\forall i \in \{1, \ldots, n\}$,

$$\mathrm{w_f} \geq 1\text{-}\xi_f, \forall \mathrm{f} = 1, \ldots \mathrm{m}$$
$$\xi_f \geq 0$$
$$\xi_i \geq 0$$

Notice that m constraints of the form $w_\mathrm{f} \geq 1\text{-}\xi_f, m$ new slack variables ξ_f, and m constraints $\xi_f \geq 0$ have been added. The slack variables ξ_f push the optimal w_f to be positive. Notice that we can further tune the SVM-ranking method by putting a different weight C on the ξ_f in the cost function so that we can have w_f being more or less pushed towards positivity. For example, if the cost function is

$$\min_{w_1,\ldots,w_m,\xi_i} \sum_{i=1}^{n} \xi_i + C \sum_{f=1}^{m} \xi_f + \lambda \sum_{f=1\ldots m} w_f^2 \tag{8}$$

for a very large C, then all w_f will become positive (if there is a positive feasible solution for w_f). This way one can control the requirement of w_f being positive. *In the experiments we have used the simple method where $C = 1$ so equal weight is put on all slack variables.* Parameter C can in practice also be tuned using cross-validation or a validation set. In the nonlinear case - using kernels - the use of the virtual examples will not force only the linear effects of attribute f to be positive, but the overall effects of this attributes to be positive - for example for a polynomial kernel of degree 2, the virtual examples will force $w_f + w_f^2 \geq 1 - \xi_f$. So in the general nonlinear case the virtual examples as used here will imply that for two products "all else being equal, more of a particular attribute by 1 is better", and this requirement can still be relaxed/controlled by the use of C for the slack variables ξ_f. Using virtual examples to enforce positivity constraints, with non linear utility functions, can be risky when the non linear coefficients are negative. In the experiments presented in this chapter, the linear coefficients are dominating the non linear ones, even when they are negative, so the incorporation of the positivity constraints with virtual examples, still helps improve the algorithm's performance.

The experiments were designed like in (Toubia, Hauser, & Simester, 2004) where the positivity of the underlying utility function is used to capture the assumption that we know for each product attribute which level has the lowest partworth - one can remove that level and assume that all other partworths are positive. If, instead, the products are represented as binary vectors with each attribute corresponding to a number of dimensions equal to the number of levels for that attribute with a 1 at the location of the present level and a 0 elsewhere - often used in practice (Arora & Huber, 2001), (Toubia, Hauser, & Simester, 2004) and also in our experiments - then the virtual examples corresponding to the prior knowledge that the partworth of a level is the smallest one would be, for example, of the form

$$(1,0,0,-1,0,0,0,0,0,0,0,0,0,0,0,0)$$

in the case of 4 attributes with 4 levels each for which we know that for the first attribute the fourth level has the smallest partworth - smaller than the first level in this case. This is the representation we used in the experiments for our method and for logistic regression. Finally we note that one can add other types of prior knowledge to constraint the estimation of the utility function through the use of virtual examples (Scholkopf, Burges, & Vapnik, 1996).

Handling Heterogeneity

The method discussed so far assumes that the data come from a single true underlying utility function and we estimate one utility function. In practice the data may come from many individuals, therefore from different underlying utility functions. A state of the art approach to handling such data is by assuming a priori that all utility functions come from a probability distribution, for example a (unknown) Gaussian, and then estimating all utility functions simultaneously through also estimating the parameters of this distribution, like it is done in the case of Hierarchical Bayes (Lenk, DeSarbo, Green, & Young, 1996) (DeSarbo & Ansari, 1997) (Allenby, Arora, & Ginter, 1998) (Arora, Allenby, & Ginter, 1998)

In Chapter 3, we develop methods along the lines of the ones presented here that can be used to simultaneously estimate many utility functions that are assumed to be related in some way - i.e. all come from the same, unknown, Gaussian distribution. Some possible directions can be, for example, along the lines of boosting (Freund & Schapire, 1996) (Friedman, Trevor Hastie, & Tibshirani, 1998) or learning with heterogeneous kernels

(Bennett & Bredensteiner, 2000). The issue is an open one also in the area of statistical learning theory.

In this chapter we compare our approach with HB even though we estimate one utility function for each individual independently - hence HB has a relative advantage in the experiments since it combines information across all individuals. We extend our SVM-ranking approach along the direction of combining the models estimated for each individual as in (Toubia, Hauser, & Simester, 2004), briefly as follows.

First we estimate one model, for example one linear utility function w_k, for each individual k independently. We then take the mean of the estimated models $w = \frac{1}{N}\sum_k w_k$, where N is the number of individuals. Finally, for each individual we replace w_k with $\gamma_k w_k + (1-\gamma_k)w$. Parameters γ_k are between 0 and 1 and we estimate them by minimizing the mean square error of $\gamma_k w_k + (1-\gamma_k)w$ from the true utility function of each individual k: this gives an upper bound on the performance that can be achieved if we were to estimate γ_k using only the available data as should be done in practice (in practice a validation set can be used to set the parameters γ_k as for the case of parameter λ discussed above). We call this method SVM-Mix to discriminate from the basic SVM-ranking approach in the experiments below. Although this is a very simple and ad hoc approach to handling heterogeneity, the experiments show that the proposed direction is promising. We plan to explore this direction in the future.

Experiments

We run Monte Carlo simulations to study the performance of the methods under varying conditions. Simulations have been often used in the past to study preference modeling methods (i.e. ((Carmone & Jain, 1978)(Toubia, Simester, Hauser, & Dahan, 2003) (Andrews, Ansari, & Currim, 2002)). They are useful, for example, in exploring various domains in order to identify strengths and weaknesses of methods. Below we explore domains that vary according to noise (magnitude) and respondent heterogeneity. We used simulations to compare our methods with logistic regression, the recently proposed polyhedral estimation method of (Toubia, Hauser, & Simester, 2004), and with HB for heterogeneous data, considered a state of the art approach. It is important to note that *in all cases we generated the data in a way that gives an advantage to logistic regression and HB* - that is, the data were generated according to the probability distributions *assumed* by these methods. Moreover, the comparison with HB is not well-defined since our methods are for individual utility estimation while HB uses information across many individuals. SVM-Mix is the only method that can be directly compared with HB.

Design of Simulations

For easy comparison with other work in the literature we followed the basic simulation design used by other researchers in the past. In particular we simply replicated the experimental setup of (Toubia, Hauser, & Simester, 2004), which in turn was based on the simulation studies of (Arora & Huber, 2001). For completeness we briefly describe that setup.

We generated data describing products with 4 attributes, each attribute having 4 levels. Each question consisted of 4 products to choose from. The question design we used was either orthogonal or randomly generated. For the orthogonal design to be well-defined we used 16 questions per individual as in (Toubia, Hauser, & Simester, 2004). The random design is a closer simulation for data that are not from questionnaires, such as more unconstrained consumer choice data.

We simulated 100 individuals. The partworths for each individual were generated randomly from a Gaussian with mean $\left(-\beta,-\frac{1}{3}\beta,\frac{1}{3}\beta,\beta\right)$ for each attribute. Parameter *β* is the magnitude that controls the noise (response accuracy). As in (Toubia, Hauser, & Simester, 2004), we used $\beta=3$ for high magnitude (low noise) and $\beta = 0.5$ for low magnitude (high noise). We modeled heterogeneity among the 100 individuals by varying the variance σ^2 of the Gaussian from which the partworths were generated. The covariance matrix of the Gaussian was a diagonal matrix with all diagonal elements being σ^2. We modeled high heterogeneity using $\sigma^2 = 3\beta$, and low heterogeneity using $\sigma^2 = 0.5\beta$, like in (Toubia, Hauser, & Simester, 2004). As discussed in (Arora & Huber, 2001) and (Toubia, Simester, Hauser, & Dahan, 2003) these parameters are chosen so that the range of average partworths and heterogeneity found in practice is covered.

Notice that for each of the four attributes the mean partworths are the smallest for the first level and the largest for the fourth level - in increasing order. Because the polyhedral estimation method of (Toubia, Hauser, & Simester, 2004) requires that constraints about the relative order of the actual partworths for each level relative to the lowest level are added (in the form of positivity constraints (Toubia, Hauser, & Simester, 2004)), we incorporated this information to all other methods. For our method and for logistic regression this was done using the virtual examples approach discussed earlier in the chapter. For the case of HB this was done by simply constraining the sampling from the posterior during the HB estimation iterations to be such that we only use partworth samples for which the lowest levels are the same ones as the actual lowest levels. Adding constraints to HB can be done in other ways, too, as discussed in (Sawtooth Software, 2009), but none of them is standard. Notice that the relative order may be changing as we sample the partworths for the four levels: we incorporated constraints about the actual lowest levels and not the lowest levels of the mean partworths.

Finally, all experiments were repeated five times - so a total of 500 individual utilities were estimated - and the average performance is reported.

Experimental Results

We compare the methods using the RMSE of the estimated partworths. Both estimated and true partworths were always normalized for comparability. In particular, as in (Toubia, Hauser, & Simester, 2004), each attribute is made such that the sum of the levels is 0, and the utility vector is then normalized such as the sum of the absolute values is 1. We also measured the predictive performance (hit rate) of the estimated models by generating 100 new random questions for each individual and testing how often the estimated utility functions predict the correct winning product. In the table below we report the hit rates below the Root Mean Square Errors (RMSE).

Table 1 Comparison of Methods using RMSE and hit rates in parenthesis. The true utilities are linear and linear utility models are estimated. Bold indicates best or not significantly different than best at p < 0.05 among analytic center, SVM, and logistic regression - the first three columns only. With a * we indicate the best among all columns.

Mag	Het	Design	Analytic	SVM	Logistic	HB	SVM_mix
L	H	**Random**	0.92	**0.69**	0.77	0.60*	0.64
			79.1%	81.8%	81.1%	84.5%	83.1%
L	H	**Orthogonal**	0.75	**0.66**	**0.67**	0.56*	0.61
			81.2%	82.7%	82.7%	85.5%	83.9%
L	L	**Random**	1.15	**0.86**	1.00	0.66*	0.69*
			74.5%	77.4%	75.7%	82.6%	81.7%
L	L	**Orthogonal**	0.89	**0.81**	**0.83**	0.62*	0.67
			76.9%	78.6%	78.3%	83.8%	82.3%
H	H	**Random**	0.67	**0.53**	**0.52**	0.46*	0.48*
			84.0%	85.9%	86.9%	88.2%	87.2%
H	H	**Orthogonal**	0.81	**0.61**	**0.59**	0.49*	0.51*
			80.4%	84.1%	84.9%	87.3%	86.3%
H	L	**Random**	0.65	**0.52**	**0.53**	0.35*	0.37*
			83.3%	86.0%	85.8%	90.3%	89.5%
H	L	**Orthogonal**	0.81	**0.68**	**0.65**	0.34*	0.53
			79.2%	81.3%	82.8%	90.6%	85.8%

Table 1 shows the results. The format of the table is the same as that of (Arora & Huber, 2001) and (Toubia, Hauser, & Simester, 2004). We label our first method that uses individual only data as "SVM-ranking" since it is very similar to SVM classification. The polyhedral method of (Toubia, Hauser, & Simester, 2004) is labeled as "Analytic" - the method is called Analytic Center in (Toubia, Hauser, & Simester, 2004). We also report the results of the SVM-Mix methodology that utilizes aggregate information from across the 100 individuals.

We performed two significance tests: a) one to compare only the analytic center method, logistic regression, and the method proposed here - the three methods that don't combine information across individuals; the best of the first three columns is reported in bold; b) one to find the best among all columns (including HB and SVM-Mix) which we report with a "*". From Table 1 we observe the following:

- SVM significantly outperforms both the analytic center method and logistic regression, the latter for the random designs and when there is noise. It is never worse than logistic regression or the analytic center method.

- Both SVM and SVM-Mix are relatively better for the random design. For example SVM is similar to logistic regression in all orthogonal design cases. We believe this is partly due to the problem with choosing parameter λ for the orthogonal design, as discussed above, and because in general the future data come from a different probability distribution than the estimation data. This limitation also indicates that it may be important to combine the SVM-ranking method with a similar method for designing questionnaires. As shown by (Toubia, Hauser, & Simester, 2004) such an extension to questionnaire design can lead to significant improvements. We leave this as part of future work.

- The SVM-ranking method significantly outperforms both logistic regression and the analytic center method when there is noise for the random design. The performance drop from high magnitude to low magnitude, namely when noise increases, is significantly lower for SVM than for both logistic regression and the analytic center for the random design. It is significantly lower than logistic regression for the orthogonal design but larger than the analytic center method in that case. However the latter is always significantly worse than the SVM-ranking method. The SVM-ranking method is therefore overall more robust to noise than the other methods. We also note that for our method the performance drop from low to high noise is influenced by the relative λs used since different λs are used for the high and low magnitudes (chosen using cross-validation).

- Heterogeneity: the SVM-Mix extension shows promising results. For the random design, HB is better only in the case of low magnitude and high heterogeneity, while in all other cases HB and SVM-Mix perform similarly. This, coupled with the fact that the SVM-ranking method is computationally efficient - while HB is not (Sawtooth Software, 2009) - indicates the potential of the proposed approach.

Estimation of Nonlinear Models

The next set of experiments considers the case where the true underlying utility function of each individual (respondent) is nonlinear. In order to account for the nonlinear effect, we estimate nonlinear models as described earlier in the chapter. A typical nonlinear effect in consumer preferences is simple interactions between two different product attributes (i.e. price and brand). In our case, adding interactions among all product attribute levels (all 16 dimensions) would lead to a large number of parameters to estimate (15*16/2 = 120) which would be computationally intractable for HB. Therefore we only added the interactions between the first two attributes. Since each attribute has 4 levels we added an extra $4 \times 4 = 16$ dimensions capturing all interactions among the 4 levels of the first attribute and the 4 levels of the second one. Thus, the utility function of each individual consisted of the original 16 parameters generated as before, plus 16 new parameters capturing the interactions among the levels of the first two attributes (clearly, without loss of generality, other choices could be made).

These new 16 parameters were generated from a Gaussian with mean 0 and standard deviation σ_{nl}. The size of σ_{nl} controls the size of the interaction parameters of the underlying utility functions. We assume we don't know the sign of the interaction coefficients, other than for the method of (Toubia, Hauser, & Simester, 2004). The method of (Toubia, Hauser, & Simester, 2004) requires prior knowledge of the least desired level of each feature, so we incorporated this information (in the form of positivity constraints for the interaction coefficients) for the analytic center, effectively giving that method an advantage relative to the other ones. In practice we may not know the sign of the interaction coefficients, so we did not add this information to the other three methods.

Our objective is to do experiments with two different levels of nonlinearity: low nonlinearity and high nonlinearity. Our definition of "level of nonlinearity" is given below in the form of a short sequence of computations. For a given σ_{nl}:

- Draw a random population of 1000 utility functions (1000 individuals);
- Generate the set of all 4 x 4 x 4 x 4 = 256 possible 4-attribute products;
- For each individual and each product, compute the absolute values of the nonlinear and linear parts of the utility of the product separately;
- For each individual, add all 256 absolute values of the nonlinear parts and the linear parts separately, and take the ratio between the sum-absolute-nonlinear and the sum-absolute-linear;
- Compute the average of this ratio over the 1000 individuals.

We use the average ratio computed in the last step as a characterization of the relative size of the underlying nonlinear (interaction) effect in the simulated population. In the sequel, we present experiments for the cases where the average-ratio is *25%* (Low nonlinearity) and *75%* (High nonlinearity). In other words, over all possible products the average nonlinear part of the utility is about *25%* (low nonlinearity) or *75%* (high nonlinearity) of the linear part. The values of σ_{nl} that result in the specified levels of nonlinearity are:

- For low magnitude and high heterogeneity: 0.61 and 1.84 (low and high nonlinearity, respectively)
- For high magnitude and low heterogeneity: 1.26 and 3.80 (low and high nonlinearity, respectively)

To estimate the nonlinear utilities using logistic regression and HB we represented the data using 32 dimensional vectors (16 linear plus 16 nonlinear). For the method of (Toubia, Hauser, & Simester, 2004) we followed the suggestion in (Toubia, Simester, Hauser, & Dahan, 2003): we introduced an additional feature with 16 levels corresponding to the 16 nonlinear interaction parameters. Therefore the three methods (other than SVM which as we discussed always estimates the n dual parameters α_i) estimated 32 parameters for each individual. Notice that HB can hardly handle even this low dimensional nonlinear case (Sawtooth Software, 2009), which in contrast is a computationally mild case for the polyhedral method, SVM, and logistic regression. For computational reasons (for HB) and to avoid cluttering we only did experiments in two cases:

1. high magnitude and low heterogeneity - the "easiest" case in practice;

2. low magnitude and high heterogeneity - the "hardest" case in practice, and also the case where our method has the least advantage relative to HB as shown in Table 1.

For computational reasons we also simulated 100 individuals only once (instead of 5 times in the linear experiments case) for these experiments.

In Table 2 we compare only SVM-Mix and HB, since the conclusions about the comparison of the polyhedral method, SVM, and logistic regression are similar as for the linear utility experiments. We show the performances of the logistic, SVM, and polyhedral methods in the Appendix at the end of the chapter. Although the actual utilities are nonlinear, we also estimated linear models to see if it is even worth estimating nonlinear models to begin with. To compare the linear and non-linear models we use hit rates: the percentage of correct prediction of 100 out-of-sample choices. In the Appendix below we report other RMSE errors. In Table 2 we also report the RMSE of the nonlinear parts of the utility functions, which captures the accuracy with which the 16 interaction coefficients are estimated. Therefore in Table 2 we show the hit rates of: linear SVM-Mix with the mixture parameter γ estimated as in the linear experiments, non-linear SVM-Mix where now we estimated two mixture parameters γ_l and γ_{nl} for the linear and non-linear parts of the estimated utility using again the method outlined in the linear experiments, linear HB, and non-linear HB. In parenthesis, for the non-linear models, we report the RMSE of the interaction coefficients (the 16 coefficients for the nonlinear part of the utility function).

Table 2 The true utilities are nonlinear. Hit rates and the RMSE of the estimated interaction coefficients in parenthesis are reported. Bold indicates best or not significantly different than best at $p < 0.05$ across all columns.

Mag	Het	NL	Des	SVM-Mix Lin	SVM-Mix NL	HB Lin	HB NL
	H	L	**Rand**	81.6%	81.1% (**1.43**)	82.7%	81.5% (1.56)
			Orth	81.7%	81.0% (**1.49**)	82.7%	80.7% (1.61)
	H	H	**Rand**	75.3%	78.1% (1.15)	76.2%	**78.6%** (1.33)
			Orth	75.2%	76.6% (1.30)	75.4%	**76.1%** (1.50)
H	L	L	**Rand**	87.9%	87.6% (**1.48**)	88.2%	**88.4%** (1.57)
			Orth	85.5%	84.1% (**1.48**)	89.0%	88.3% (1.61)
H	L	H	**Rand**	78.6%	82.6% (1.14)	79.8%	**83.2%** (1.31)
			Orth	77.5%	78.6% (**1.27**)	79.8%	**81.1%** (1.41)

The results show the following:

- When the non-linearity is low the linear models are generally better than the nonlinear ones.
- When the nonlinearity is high, it is generally better to estimate nonlinear models both for HB and for SVM-Mix.

The best (among linear and nonlinear) HB outperforms the best (among linear and nonlinear) SVM-Mix in the cases it outperformed it in the linear experiments (Table 1). However, the relative differences of the hit rates decrease as the amount of nonlinearity increases. For example in the high nonlinearity case the nonlinear SVM-Mix is similar to HB in 3 out of the 4 cases (Low-High or High-Low for random and orthogonal), while in Table 1 SVM-Mix is similar to HB only in 1 out of the 4 cases. This indicates that the proposed approach has a relative advantage when there are nonlinearities.

When we estimate non-linear models, the RMSE of the nonlinear part of the estimated function is smaller for SVM-Mix than for HB. In other words the SVM-ranking method captures the nonlinear interactions better than HB. In the Appendix below we show that the simple SVM-ranking algorithm (not "Mix") is also on average better than any other method in terms of capturing the nonlinear effects.

Summary and Contributions

This chapter presented two SVM based methods for learning ranking functions from choice based comparisons of the form "the user prefers choice $i = j$ than all other choices $i \in \{1, n\}$ ". In other supervised information retrieval problems, the training data includes explicit or implicit relevance rankings, either by experts, or by empirical observations. In the choice based conjoint problem we only know which the winning combination is. We transformed the problem to a binary classification task of pairwise comparisons, and use an SVM-like regularized loss function to account for noise in the users' choices. We also present an approach of combined classifiers to better learn personalized ranking functions by combining information from both individual and aggregate data. We compared our method with standard algorithms used by the marketing community for utility function estimation: logistic regression, polyhedral estimation, and hierarchical bayes (HB). We evaluated on standard, widely used, simulation data. The experiments show that SVM adapted to ranking problems handle noise and nonlinearities significantly better than the baseline approaches, and significantly faster than HB, the best performing baseline approach.

Preference modeling has been a central problem in the marketing community and is becoming increasingly important in other business areas such as in supply chain and procurement where the procurement processes are automated and data describing past choices are captured. At the same time, the "democratization" of data, in the sense that data is captured everywhere and under any conditions, implies that companies often need to use preference modeling tools that do not assume the data is generated in a controlled environment - i.e. through questionnaires. As the conditions under which preference data are captured vary, and as more and more applications arise, there is an increasing need for new tools and approaches to the problem of preference modeling that are computationally efficient, have high accuracy, and can handle noise and high (multi-attribute products) dimensional data. The work presented here aims at opening a direction of research in the area of preference modeling that can lead to such new approaches and tools. We did not discuss here issues such as how to use the proposed framework for example for designing questionnaires - we believe this is possible, as we briefly discussed and as is indicated by the work of (Toubia, Simester, Hauser, & Dahan, 2003), and we leave this for future research. Instead we focused on laying the foundations for methods and tools to solve a variety of preference modeling problems.

In this chapter we presented a framework for developing computationally efficient preference models that have high accuracy and can handle noisy and large dimensional data. The framework is based on the well-founded field of statistical learning theory (Vapnik, 1998). Highly non-linear conjoint estimation models can be also computationally efficiently estimated. The models estimated depend only on a few data points, the ones that correspond to "hard choices". This can provide useful insights to managers by focusing their attention only to those choices. Moreover, this characteristic can be used to design individual specific questionnaires along the lines of (Toubia, Hauser, & Simester, 2004).

The experiments showed that:

The proposed approach significantly outperforms both the method of (Toubia, Hauser, & Simester, 2004) and standard logistic regression, the latter when there is noise and for the random design. It is never worse than the best among these three methods.

The proposed approach is less sensitive to noise - high response error - than both logistic regression and the method of (Toubia, Hauser, & Simester, 2004). *It is therefore more robust to noise.*

The proposed approach is relatively weaker when data from an orthogonal design are used. This limitation indicates that it may be important to combine the SVM-ranking method with a method similar in spirit for designing questionnaires. As shown by (Toubia, Hauser, &

Simester, 2004), such an extension to questionnaire design can lead to significant improvements. We leave this as part of future work.

The SVM-Mix extension for handling heterogeneity leads to promising results with performance often similar to that of HB;

When the true underlying utility function is nonlinear (for example there are interaction effects between the product attributes) it is better to estimate nonlinear models when the nonlinearity is high. Moreover the SVM-ranking method estimates the interaction coefficients significantly better than all other methods.

Furthermore, the estimation is computationally efficient, so, for example, large datasets for products with large numbers of attributes can also be used - unlike the case of HB.

A number of extensions are possible within this framework for preference modeling. A clear direction for future work is to incorporate to the individual-specific models cross-respondent information in the case of heterogeneity. The experiments show that even a simple SVM-Mix extension for handling heterogeneity is already promising. Another important direction is to develop other preference modeling methods using the principles of Statistical Learning Theory: in particular, there is evidence (see for example (Rifkin, 2002)) that the most important part of the proposed approach is the incorporation of the complexity control in the estimation process. It may be the case that logistic regression with complexity control, for example along the lines of (Zhu & Hastie, 2001), is a more appropriate approach than the one we tested here, since it may better capture the noise model of the data typically assumed in conjoint analysis.

The machinery developed for SVM as well as statistical learning theory can be used for solving in new ways problems in the field of conjoint analysis. For example, one can extend the use of virtual examples we used here for adding positivity constraints on the utility function. Empirical evidence shows that if the original data used to estimate a model are extended to include virtual examples then the performance of the estimated models improves (Scholkopf, Burges, & Vapnik, 1996). Generally virtual examples are data that are either added to the estimation data by the user because of prior knowledge about them, or are generated from the existing data using transformations that the user knows a priori do not alter their key characteristic (i.e. which product is the preferred one) (Scholkopf, Burges, & Vapnik, 1996). Furthermore, models for metric based conjoint analysis (Toubia, Simester, Hauser, & Dahan, 2003) can be also developed within the framework in this chapter, for example in the spirit of SVM regression instead of classification (Vapnik, 1998). Finally, another direction of research is to develop active learning (Tong & Koller, 2000) type methods for the problem of adaptively designing questionnaires, like for example in (Toubia, Simester, Hauser, & Dahan, 2003).

Once the problem of preference modeling is seen within the framework of statistical learning theory and SVM, a number of new methods can be developed for the conjoint analysis field. The work in this chapter does not aim by any means to replace existing methods of preference modeling, but instead to contribute to the field new tools and frameworks that can be complementary to existing ones for solving preference modeling problems. Finally, the experiments presented here are by no means exhaustive: more experiments by other researchers will be needed to establish the relative strengths and weaknesses of the proposed approach, as is always the case with any newly developed method.

Appendix

Hierarchical Bayes[2]

The Hierarchical Bayes algorithm referred to in this chapter, and later in this thesis is a regression approach for learning preference models in two levels:

1)The individual level

2)A higher level, representing the individual preference regression weights as a multivariate normal distribution. (assuming linear representation of utility functions)

The algorithm assumes that the individual linear regression weights are drawn from the multivariate normal distribution:

$$\boldsymbol{w_i} \sim \boldsymbol{N}(\boldsymbol{\alpha}, \boldsymbol{D})$$

Where $\boldsymbol{w_i}$represents the coefficients of the utility function of the ith individual, $\boldsymbol{\alpha}$ is the vector of the means of the regression coefficients, and $\boldsymbol{D}$ is the matrix of variances and covariances of the weights across individuals.

[2] This section is summarized version of the algorithm description in (Sawtooth Software, 2009)

At the individual level the individuals are assumed to have preference behaviors described by the utility function:

$$y_{ij} = x'_{ij} w_i + e_{ij}$$

where e_{ij} is a random error, with mean zero, and variance σ^2.

The algorithm starts a Gibbs sampling iteration with w, α, and the covariances set to zero, and the variances and σ set to one. Then iterates between the following four conditional estimations:

1)**α**, given **D**, and **w**,

2)**D**, given **w**, and **α**

3)**w**, given **α**, **D**, and **σ**

4)**σ**, given **α**, **D**, and **w**

This process is typically repeated for thousands of iterations, until the algorithm achieves convergence.

Nonlinear Experiments: Detailed Results

We show all the results of the nonlinear experiments in Table 3. In each cell we report five performances: a) the RMSE when we estimate a linear model; b) the out of sample hit rate when we estimate a linear model; c) the RMSE of the linear part of the utility when we estimate a nonlinear model; d) the RMSE of the non-linear part when we estimate a nonlinear model; e) the out of sample hit rate of the nonlinear model estimated.

From the results we observe the following:

- **Nonlinear part estimation:** The key result is that SVM-Mix estimates the nonlinear parts of the utility functions better than all other methods, including HB. SVM - without combining information across individuals - is also better than both the logistic regression and HB. It should be noted that all methods have large RMSE as compared to the linear estimations. We attribute this to the fact that each 32-dimensional vector describing a product includes just a single nonzero element out of the total 16 nonlinear elements (since only one of the four levels of the two attributes involved for the nonlinearity is nonzero for each product). In contrast, there are 4 nonzero elements out the 16 linear ones. Effectively there is little information about the nonlinear part of the utility functions.
- **Linear part estimation:** For the linear parts of the estimated utility function the comparison of SVM, logistic, and polyhedral is qualitatively similar as in the linear experiments (Table 1).
- **Linear part estimation comparison with HB:** The difference between HB and "SVM-Mix" for the linear parts of the utility function is relatively smaller than in the linear utility experiments (Table 1): our method is therefore less influenced by nonlinearities than HB.

Table 3 Comparison of methods when nonlinear models are estimated.

Mag	Het	NL	Des	Performance	Analytic	SVM	Logistic	HB	SVMmix
L	H	L	Rand	**RMSE Lin**	0.95	0.70	0.81	0.62	0.67
				Hit Lin	78.1%	80.8%	79.6%	82.7%	81.6%
				RMSE Lin/NL	0.86	0.69	0.77	0.66	0.65
				RMSE NL/NL	1.55	1.52	1.63	1.56	1.43
				Hit NL	77.6%	80.4%	77.8%	81.5%	81.1%
L	H	L	Orth	**RMSE Lin**	0.80	0.69	0.70	0.61	0.65
				Hit Lin	78.7%	80.6%	80.6%	82.7%	81.7%
				RMSE Lin/NL	0.83	0.71	0.72	0.65	0.66
				RMSE NL/NL	1.57	1.61	1.70	1.61	1.49
				Hit NL	78.7%	79.8%	80.2%	80.7%	81.0%
L	H	H	Rand	**RMSE Lin**	1.08	0.85	0.94	0.74	0.79
				Hit Lin	72.8%	75.1%	74.5%	76.2%	75.3%
				RMSE Lin/NL	0.96	0.79	0.81	0.76	0.74
				RMSE NL/NL	1.32	1.20	1.31	1.33	1.15
				Hit NL	77.0%	78.2%	77.1%	78.6%	78.1%
L	H	H	Orth	**RMSE Lin**	0.95	0.82	0.84	0.75	0.76
				Hit Lin	73.4%	74.4%	73.5%	75.4%	75.2%
				RMSE Lin/NL	0.95	0.79	0.78	0.78	0.74
				RMSE NL/NL	1.42	1.36	1.44	1.50	1.30
				Hit NL	75.6%	76.4%	75.9%	76.1%	76.6%
H	L	L	Rand	**RMSE Lin**	0.71	0.55	0.58	0.39	0.39
				Hit Lin	81.8%	84.7%	84.1%	88.2%	87.9%
				RMSE Lin/NL	0.78	0.54	0.59	0.41	0.40
				RMSE NL/NL	1.58	1.54	1.61	1.57	1.48
				Hit NL	80.4%	84.4%	83.7%	88.4%	87.6%
H	L	L	Orth	**RMSE Lin**	0.84	0.67	0.66	0.35	0.51
				Hit Lin	78.9%	81.5%	82.1%	89.0%	85.5%
				RMSE Lin/NL	0.92	0.71	0.70	0.39	0.56
				RMSE NL/NL	1.56	1.55	1.56	1.61	1.48
				Hit NL	76.6%	79.9%	80.7%	88.3%	84.1%
H	L	H	Rand	**RMSE Lin**	0.95	0.74	0.81	0.52	0.43
				Hit Lin	75.1%	77.1%	76.7%	79.8%	78.6%
				RMSE Lin/NL	0.88	0.67	0.67	0.54	0.44
				RMSE NL/NL	1.32	1.17	1.25	1.31	1.14
				Hit NL	78.6%	80.5%	80.8%	83.2%	82.6%
H	L	H	Orth	**RMSE Lin**	0.94	0.83	0.77	0.50	0.56
				Hit Lin	74.0%	75.5%	75.9%	79.8%	77.5%
				RMSE Lin/NL	1.03	0.75	0.75	0.49	0.62
				RMSE NL/NL	1.42	1.33	1.31	1.41	1.27
				Hit NL	74.2%	77.7%	77.5%	81.1%	78.6%

CHAPTER 2: REGULARIZED MANIFOLDS FOR LEARNING FROM RELATED TASKS

Introduction

Recent work in multitask learning (Evgeniou & Pontil, 2004)(Chapelle & Harchaoui, 2005)(Girosi, 2003) has shown that data from related tasks can be used effectively by regularized learning algorithms, with nonlinear loss functions that penalize errors based on aggregate data less than the errors from the task specific data. The same approach can apply to different regularized algorithms, including SVMs, Regularized Least Squares Classification (RLSC) (Rifkin, 2002), or Regularized Logistic Regression (Minka, 2001).

In the first chapter we examined how to use regularized learning algorithms in a hierarchical learning problem, by proposing a simple weighted average algorithm of the task specific SVM ranking algorithm, with an SVM trained on the aggregate dataset. This simple approach allowed us to get performance that was competitive to the Hierarchical Bayes one.

In this chapter we first present how to implement the non-linear loss function approach with RLSC. Then we extend the RLSC algorithm with the graph Laplacian transformation, to show how to pose the same problem as a special case of semi-supervised learning, with regularized manifolds (Belkin & Niyogi, 2004). We also show that, in the linear case, our approach is a generalization of the kernel based methods for multi-task learning, without enforcing constraints on cross-task covariance metrics.

Related Work

Multi-task learning, or transfer learning has received a lot of attention over the last few years in the machine learning community. The work in the transfer learning domain has focused on two directions: a) How to best train the task specific models, with the additional information, if a problem is appropriate for transferring information between tasks, and b) Indentify situations where transferring such information is appropriate.

In this chapter and in this thesis in general, we only deal with the former kind of problem, and we leave the latter to be determined through experiments using cross-validation. The related work to the problem investigated in this chapter assumes that the underlying tasks do share some structure, and they are therefore appropriate for sharing information between tasks (Wilson, Fern, Ray, & Tadepalli, 2007) (Argyriou, Micchelli, Pontil, & Ying, 2007) (Maurer, 2006.). Many approaches rely on hierarchical learning iterations (Sawtooth Software, 2009), or cross-validation iterations (Evgeniou, Toubia, & Pontil, 2007) to assess the task similarity, to optimally transfer information between tasks. Our work is mostly related to (Chen, Song, Wang, & Zhang, 2008) who use regularized Laplacians to learn multiple-label problems, and (Sheldon, 2008) who assesses the task relationship based on a graph structure on the tasks. Like (Evgeniou, Toubia, & Pontil, 2007) we also rely on cross validation to optimize the parameters controlling the transfer of knowledge between the tasks.

Semi-Supervised Learning Intuition

The intuition for this approach is that learning from related data is similar to semi-supervised learning. In both cases, we want to incorporate related data to our problem, but we are not sure about the labels of the instances from the related data, and how much they should contribute to the loss function, compared to the labeled data for the task. Regularized manifolds, and other clustering approaches (Nigam et al 200; Belkin, and Niyogi 2003) try to estimate the labels based on a weighted combination of the labels of the closest labeled instances. The closest instances are chosen based in the transformed space representing the clusters or manifold. In the problem of learning from related tasks, we do not need to estimate the labels of the similar (or close in the distance metric transformation) instances, because they are already labeled. As we have seen in the case of combining preference data, it sometimes helps to incorporate the related information from the instances of other tasks to the training of the task specific models.

The open question is how much penalty we should pay for errors we make on the instances of the related data, compared to the penalty we pay for errors of on the training instances of the same task. In the experiments below we show that the distance of the instances in the manifold space can also be used to appropriately penalize the related data, when training task specific models.

Penalized Regularized Least Squares (PRLSC) for Related Tasks

Consider the standard algorithm for Regularized Least Squares Classification, with l instances of training data per task. We can learn the task specific models by minimizing the following loss function:

$$\min_{f\in H_K} \sum_{i=1}^{l}\left(y_i - f(x_i)\right)^2 + \gamma \|f\|_K^2$$

Now assume that for each task model we have another u instances of related data. We incorporate the related instances in the loss function, but we weigh the penalty we pay for errors in this instances by a factor $0 \le \mu \le 1$. Then our learning with related data loss function becomes:

$$\min_{f\in H_K} \left(\frac{1}{l}\right)\sum_{i=1}^{l}\left(y_i - f(x_i)\right)^2 + \left(\frac{\mu^2}{u}\right)\sum_{i=l+1}^{l+u}\left(y_i - f(x_i)\right)^2 + \gamma \|f\|_K^2$$
$$0 \le \mu \le 1$$

The loss function can be rewritten as follows:

$$\min_{f\in H_K} \frac{1}{l}\sum_{i=1}^{l+u}\left(y_i^{\mu} - f\left(x_i^{\mu}\right)\right)^2 + \gamma \|f\|_K^2$$
$$y_i^{\mu} = y_i, x_i^{\mu} = x_i \quad \text{for } i \le l$$
$$y_i^{\mu} = \left(\frac{l\mu}{u}\right) y_i, x_i^{\mu} = \left(\frac{l\mu}{u}\right) x_i \quad \text{for } l < i \le l+u$$

(9)

And the dual representation of the problem becomes:

$$f^*(x) = \sum_{i=1}^{l} \alpha_i^* K^{\mu}(x, x_i)$$

K^{μ} is the $(l+u)\times(l+u)$ gram matrix $K_{ij}^{\mu} = K(x_i^{\mu}, x_j^{\mu})$

$$Y^{\mu} = \left[y_1 \ldots y_l, \left(\frac{l\mu}{u}\right) y_{l+1} \ldots \left(\frac{l\mu}{u}\right) y_{l+u} \right]$$

Replace $f(x)$, take partial derivatives and solve for α^*

$$\alpha^* = \left(K^{\mu} + \gamma(l+u)I\right)^{-1} Y^{\mu}$$

(10)

Note, that the scaling by the factor μ only works in this derivation, when $f(x)$ is a linear function.

Penalized Laplacian RLSC (PLapRLS), for Related Tasks

Laplacian RLSC algorithms (Belkin & Niyogi, 2004) have been used successfully in other semi-supervised learning settings. The loss function of the Laplacian RLSC penalizes the weighted deviation of the estimated function $f(x)$, for instances i, j that fall close to each other in the geodesic space of a manifold (high weight in the manifold space W_{ij}). The manifold is estimated on both the labeled and the unlabeled data. The additional loss term accounting for the deviation is:

$$\sum_{i,j=1}^{l} V\left(f(x_i) - f(x_j)\right)^2 W_{ij}$$

(11)

In our problem setting, we already have an estimate for the labels, namely the labels from another task in the particular instance. Therefore, we modify the Laplacian RLSC formulation introduced by (Belkin & Niyogi,

2004) to use the actual labels for the additional instances u, and we again penalize the related information with a penalty term μ.

As a reminder a manifold regularization algorithm (Belkin & Niyogi, 2004), for semi-supervised learning with l labeled instances and u unlabeled instances, minimizes the following loss function:

$$\begin{aligned}
f^* &= \operatorname{argmin}_{f \in H_K} \frac{1}{l}\sum_{i=1}^{l} V(x_i, y_i, f) + \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(u+l)^2}\sum_{i,j=1}^{l}\left(f(x_i) - f(x_j)\right)^2 W_{ij} \\
&= \operatorname{argmin}_{f \in H_K} \frac{1}{l}\sum_{i=1}^{l} V(x_i, y_i, f) + \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(u+l)^2} f^T L f \\
&\text{Laplacian } L = D - W
\end{aligned}$$

(12)

In our case we focus in the Laplacian RLSC algorithm, for its algebraic flexibility:

$$\min_{f \in H_K} \frac{1}{l}\sum_{i=1}^{l}\left(y_i - f(x_i)\right)^2 + \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(u+l)^2} f^T L f$$

(13)

When we add additional instances u, with labels from related tasks, our loss function becomes:

$$\min\begin{bmatrix} {}_{f\in H_K}\frac{1}{l}\sum_{i=1}^{l}\left(y_i - f\left(x_i\right)\right)^2 + \\ \frac{\mu^2}{u}\sum_{i=l+1}^{l+u}\left(y_i - f\left(x_i\right)\right)^2 + \gamma_A\|f\|_K^2 + \frac{\gamma_I}{\left(u+l\right)^2} f^T L f \end{bmatrix}$$

(14)

This is equivalent to the following minimization:

$$\min_{f\in H_K}\frac{1}{l}\sum_{i=1}^{l+u}\left(y_i^{\mu} - f\left(x_i^{\mu}\right)\right)^2 + \gamma_A\|f\|_K^2 + \frac{\gamma_I}{\left(u+l\right)^2} f^T L f$$

(15)

$$\begin{aligned} &f^*\left(x\right) = \sum_{i=1}^{l+u}\alpha_i^* K^{\mu}\left(x, x_i\right) \\ &y_i^{\mu} = y_i, x_i^{\mu} = x_i \text{ for } i \le l \\ &y_i^{\mu} = \mu' y_i, x_i^{\mu} = \mu' x_i \text{ for } l < i \le l+u \\ &\mu' = \frac{\mu l}{u} \end{aligned}$$

(16)

K^μ is the $(l+u)\times(l+u)$ gram matrix $K_{ij}^\mu = K(x_i^\mu, x_j^\mu)$

$Y^\mu = \left[y_1 \ldots y_l, \mu y_{l+1} \ldots \mu y_{l+u} \right]$

Replace $f(x)$, take partial derivatives and solve for α^*

$$\alpha^* = \left(K^\mu + \gamma_A lI + \frac{\gamma_I l}{(u+l)^2} LK^\mu \right)^{-1} Y^\mu$$

(17)

Evaluation

User Preference modeling

We evaluate the algorithm in a user preference modeling problem on the publicly available dataset provided by Sawtooth Software. The dataset includes data from 100 individuals, with 10 metric instances of products with five attributes (the users provide metric ratings for each product configuration). We transform the problem to choice based comparisons by creating the vectors of differences of the instances, and classifying each comparison with a "+1" or "-1" for a winning and losing comparisons respectively (see the first chapter for more details).

We experiment with the RLSC algorithm (9),(10) with a subsample of $l = 10$ comparisons per individual, and different amounts of related data $u = \{10, 20, 30, 50, 100\}$ randomly sampled from the other 99 users. We run the experiment with 10-fold out of sample evaluation each time for each one of the 100 individuals. Table 4 shows the average accuracy over 10 experiments per individual, for the 100 individuals. The best performance for each experiment is displayed with bold fonts.

As we can see from the experiments, more foreign examples imply smaller optimal μ (heavier penalty on the data contributed by other individuals). Also we can see that adding more data from other users improves the performance, if the penalty term μ is close to the optimal one, for each number of related data u .

Table 4 Results of RLSC experiments with u instances of related data. And weight μ on loss contributed by the related data. The bold-italic cells, show the best performance.

	u=10	**u=20**	**u=30**	**u=50**	**u=100**
μ=0	18.141 %	18.090 %	18.380 %	18.040%	18.430 %
μ=0.000001	18.268 %	18.117 %	17.847 %	18.152%	18.009 %
μ=0.00001	17.897 %	18.123 %	18.217 %	18.182%	18.164 %
μ=0.0001	17.999 %	18.135 %	18.067 %	18.089 %	18.036 %
μ=0.001	18.182 %	17.835 %	18.092 %	18.140 %	18.135 %
μ=0.01	17.986 %	17.905 %	18.043 %	18.023 %	18.174 %
μ=0.1	17.132 %	16.508 %	16.225 %	15.636 %	***15.242%***
μ=0.2	16.133 %	***15.520 %***	***15.157 %***	***15.323 %***	15.276 %
μ=0.3	***15.998 %***	15.602 %	15.918 %	16.304 %	17.055 %
μ=0.4	16.581 %	16.786 %	17.162 %	17.812 %	19.494 %
μ=0.5	17.455 %	17.810 %	18.676 %	19.838 %	22.090 %
μ=0.6	18.748 %	19.589 %	20.440 %	22.355 %	25.258 %

With this type of approaches (Evgeniou & Pontil, 2004), (Chapelle & Harchaoui, 2005), the instances from other individuals contribute the same amount of information, independently of how similar the other individuals are, with the one for which we are building the model each time.

Table 5 Results of Laplacian RLSC experiments with u instances of related data. And weight μ on loss contributed by the related data. The bold-italic cells, show the best performance.

	u=10	**u=20**	**u=30**	**u=50**	**u=100**
μ=0	17.50%	18.50%	18.38%	18.20%	17.54%
μ=0.000001	17.34 %	19.46 %	17.52 %	18.11 %	20.10 %
μ=0.00001	18.30 %	18.20 %	17.54 %	18.46 %	18.10 %
μ=0.0001	18.56 %	18.76 %	18.02 %	17.73 %	17.90 %
μ=0.001	17.20 %	18.12 %	18.28 %	17.87 %	18.00 %
μ=0.01	16.92 %	17.52 %	17.98 %	17.70 %	18.15 %
μ=0.1	16.86 %	16.68 %	16.04 %	15.58 %	16.30 %
μ=0.2	***14.80 %***	***14.68 %***	***14.86 %***	***14.89 %***	***14.30 %***
μ=0.3	16.22 %	16.76 %	16.74 %	16.57 %	18.60 %
μ=0.4	15.94 %	16.54 %	17.94 %	17.93 %	20.75 %
μ=0.5	17.90 %	16.64 %	18.74 %	19.48 %	20.60 %
μ=0.6	17.74 %	20.20 %	20.60 %	22.38 %	25.35 %

We repeat the same experiments with the Laplacian RLSC approach and show the results in Table 5. We observe the optimal μ gives better performance, than without the Manifold setting. The optimal $\mu = 0.2$ seems to not depend on the amount of additional data, and the addition of more data from other users does not seem to affect the performance significantly (unlike the results in Table 4). The results in Table 5 seem to indicate that the instances from the related data have an impact that depends on the manifold transformation, and the intrinsic penalty term seems to account well for examples that are neighboring on the manifold, and have opposite labels.

Multiple School Exam Score data

One of most commonly datasets used for the evaluation of multi-task learning algorithms is the Inner London Education Authority (ILEA) dataset, which contains exams scores for 15362 students from 139 secondary schools. The input features consist of the following data: year of the exam, gender, VR band, ethnic group, percentage of students eligible for free school meals in the school, percentage of students in VR band one in the school, gender of the school (male, female, mixed), and school denomination. We expand the nominal features as binary features set, and add a bias term, so our final training sets have 27 attributes. Most papers dealing with this dataset, evaluate their training performance on a random split of 75% of each school data, and test on the remaining 25%. However, as it was observed in previous papers(Evgeniou, Micchelli, & Pontil, 2005), the different schools seem to be similar tasks, and in fact, with the previously reported algorithms, the best performance is in fact achieved by pooling all the school data together.

In our experiments we evaluate the contribution of related task data in sparse cases, where the training set is 10%, 25%, 50%, and 75% to also directly compare with (Bakker & Heskes, 2003), and (Evgeniou, Toubia, & Pontil, 2007). Also, for the purposes of direct comparison we report the explained variance on the test set, which is defined as the total variance of the data minus the sum–squared error on the test set, as a percentage of the total data variance (Bakker & Heskes, 2003), (Evgeniou & Pontil, 2004). We ran our experiments for a range of values for the cost on the related task data μ=[0.001 0.01 0.5 0.7 0.8 0.9 1], and a range of γ_l=[0 0.001 0.01

0.1 0.5 1 5 10]. Note that, when γ_I=0 the algorithm is equivalent to the PRLSC case, when μ=1 the algorithm treats the problem as a single task problem, where the data is pooled together from different tasks, and as μ gets closer to 0, it is equivalent to training each school's data separately. For all the experiments we use a fixed $\gamma_A = 1$, and we repeat each experiment 10 times over sub-samples with replacement, based on the 'Split' percentages.

For the case where the data set is split to 75% training points vs. 25% test points, the best performance reported by (Bakker & Heskes, 2003) is 29.5%, and the best performance by (Evgeniou & Pontil, 2004) is 34.37% for $\gamma_A = 1$, while our PLapRLS approach achieves an explained variance of up to 36.88% Also both (Bakker & Heskes, 2003), (Evgeniou & Pontil, 2004) conclude that the dataset seems to behave as single task, and they achieve their best performance by pooling together all the data, as if they are learning a single task. In our PLapRLS experiments, we note that multitask nature of the dataset is more apparent when the number of examples per school is smaller. Our PLapRLS approach can be seen as a generalization of Hierarchical Bayes (Bakker & Heskes, 2003) and other existing multitask kernel approaches (Evgeniou & Pontil, 2004), where the loss function penalizes data from other tasks based on coefficient similarity across tasks, rather than instance similarity (11) (12). We analyze how our approach relates to multitask kernel methods, and HB in the next section.

We show details of our results in Table 6, and illustrate the effect of different settings of the experiment in the graphs in Figure 2, Figure 3, and Figure 4. As we can clearly see from the graphs, the related task data should have less contribution than the task specific data, because the optimal values of μ are usually between 0.5 and 0.9 (Figure 2). Also, the optimal values for the Laplacian regularization γ_I are usually between 0.1, and 0.5, which shows that the graph regularization penalty, while our ambient space regularization parameter is fixed at $\gamma_A = 1$.

The graphs Figure 3 and Figure 4 show clearly that the absolute performance increases as we increase the percentage of the task specific examples, as we move from a 10% split, to a 75% split between training and test points, per school. But in all cases, the addition of some penalized geodesic information, calculated on both task specific, and task related data improves the overall accuracy, independently of the size of the split Figure 3 and Figure 4.

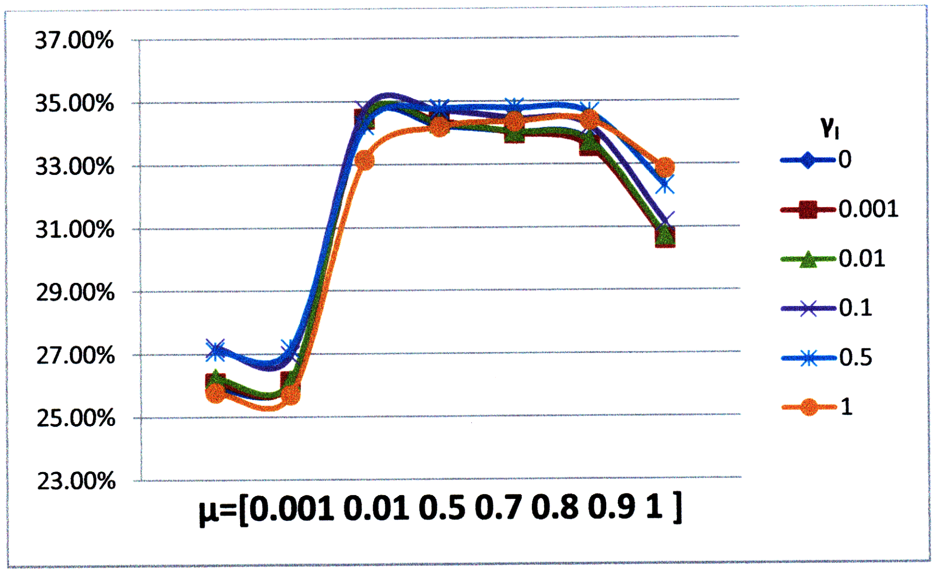


Figure 2 Average Explained variance on the ILEA dataset as function of the related data penalty term μ for different Laplacian regularization penalties γ_I.

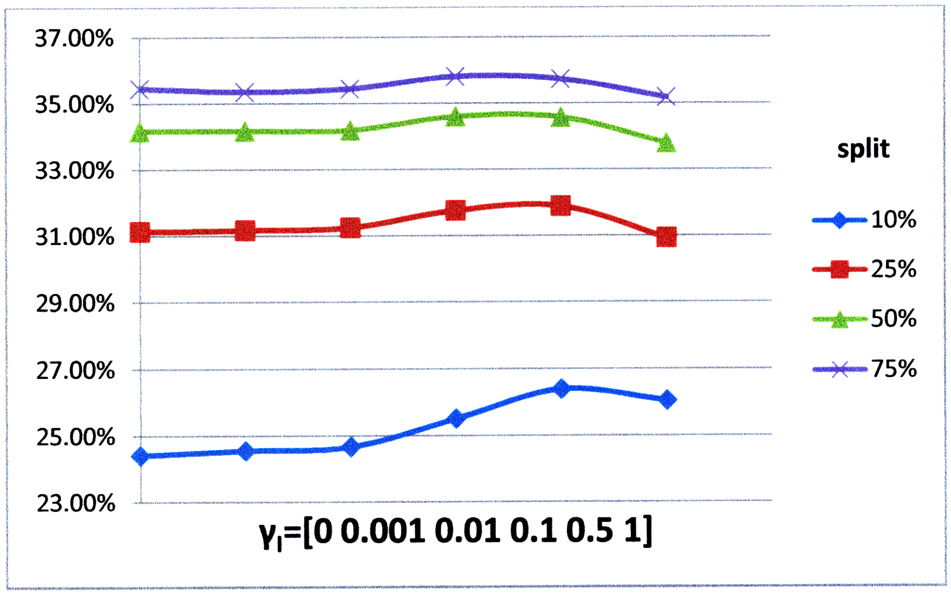


Figure 3 Average Explained variance on the ILEA dataset as function of the Laplacian regularization penalty γ_l for different training/test splits.

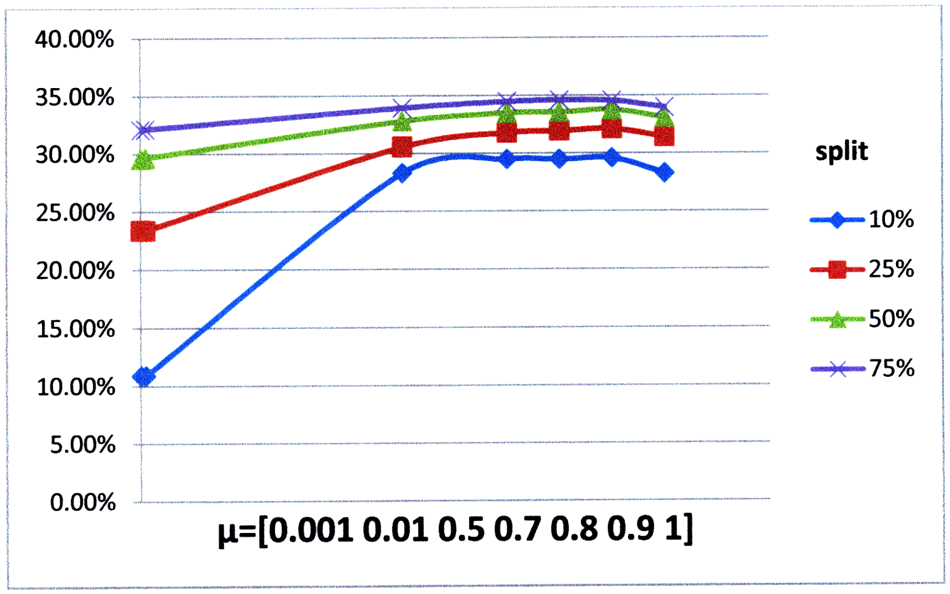


Figure 4 Explained variance on the ILEA dataset as function of the related penalty term μ, for different splits.

Table 6 ILEA experiments. The split column describes the percentage split of training vs. test data. The columns show the results for different penalties μ, and the γ_l row. The bold cells show the best performance for that experiment.

Split	μ	0.001	0.01	0.5	0.7	0.8	0.9	1
	γ_l							
10%	0	11.21%	11.28%	**30.97%**	30.65%	30.29%	29.76%	26.76%
10%	0.001	11.36%	11.42%	**31.21%**	30.86%	30.27%	30.00%	26.76%
10%	0.01	11.56%	11.61%	**31.25%**	30.94%	30.37%	30.13%	26.83%
10%	0.1	12.99%	12.98%	31.50%	**31.54%**	31.13%	31.06%	27.42%
10%	0.5	14.11%	14.24%	31.24%	32.11%	32.18%	**32.26%**	28.63%
10%	1	13.49%	13.21%	30.28%	31.84%	32.05%	**32.06%**	29.47%
10%	5	7.13%	7.41%	22.45%	26.53%	27.15%	27.84%	**30.19%**
10%	10	5.31%	5.10%	17.44%	21.33%	22.25%	23.48%	**29.99%**
25%	0	26.07%	26.33%	**34.23%**	33.97%	33.64%	33.42%	30.22%
25%	0.001	26.40%	26.26%	34.04%	**34.05%**	33.82%	33.27%	30.31%
25%	0.01	26.54%	26.40%	34.08%	**34.10%**	33.87%	33.34%	30.37%
25%	0.1	27.44%	27.19%	**34.45%**	34.36%	34.08%	33.88%	30.90%
25%	0.5	27.07%	26.97%	33.71%	34.45%	**34.54%**	34.35%	32.10%
25%	1	24.76%	24.85%	32.23%	33.75%	33.92%	**34.18%**	32.76%
25%	5	16.09%	16.17%	23.31%	27.26%	28.22%	29.50%	**32.77%**
25%	10	12.86%	12.98%	18.36%	22.16%	23.30%	24.80%	**31.82%**
50%	0	32.14%	32.39%	**35.70%**	35.56%	35.65%	35.44%	32.27%
50%	0.001	32.38%	32.37%	**35.95%**	35.86%	35.42%	35.12%	32.07%
50%	0.01	32.28%	32.32%	**35.83%**	35.61%	35.49%	35.29%	32.49%
50%	0.1	33.04%	33.01%	**36.18%**	36.14%	35.73%	35.49%	32.59%
50%	0.5	32.61%	32.63%	35.62%	35.73%	**35.82%**	35.76%	33.84%
50%	1	31.19%	31.10%	34.41%	35.17%	**35.28%**	35.27%	34.11%
50%	5	23.37%	23.66%	26.53%	29.17%	29.84%	30.94%	**34.24%**
50%	10	19.51%	19.63%	22.06%	24.50%	25.16%	26.66%	**33.00%**
75%	0	34.18%	34.62%	36.55%	**36.64%**	36.51%	36.35%	33.33%
75%	0.001	34.16%	34.43%	36.57%	**36.65%**	36.41%	35.85%	33.37%
75%	0.01	34.56%	34.30%	**36.88%**	36.56%	36.40%	36.15%	33.26%
75%	0.1	35.37%	34.66%	36.80%	36.79%	**36.88%**	36.41%	33.75%
75%	0.5	34.58%	34.90%	36.40%	**36.74%**	36.58%	36.16%	34.74%
75%	1	33.66%	33.65%	35.67%	35.96%	**36.14%**	36.05%	35.09%
75%	5	26.94%	27.34%	28.60%	30.53%	30.99%	31.60%	**34.70%**
75%	10	23.27%	23.17%	24.06%	25.89%	26.80%	27.81%	33.50%

Theoretical similarities with HB, and multitask kernel methods

Existing multitask kernel methods such as the RR-Het algorithm (Evgeniou, Toubia, & Pontil, 2007) (Evgeniou & Pontil, 2004) implement multitask loss functions that penalize the deviation from the average task coefficients. They optimize the task deviation penalty by estimating a covariance matrix D for the linear coefficients of the kernel function, and rely on polynomial expansion to support non-linear multitask scenarios. These approaches estimate the covariance matrix D through cross validation, which is very similar with the approach described in the first chapter of this thesis. RR-Het has the additional regularization benefit of solving all the tasks as a joint optimization (18).

$$min\frac{1}{\gamma}\sum_{i=1}^{I}\sum_{j=1}^{J}\left(x_{ij}w_i - y_{ij}\right)^2 + \gamma\sum_{i=1}^{I}(w_i - w_0)^T D^{-1}(w_i - w_0)$$

Where i,j are different tasks

(18)

Similarly for the Hierarchical Bayes (Bakker & Heskes, 2003) (Evgeniou, Toubia, & Pontil, 2007) (Evgeniou & Pontil, 2004) approach, the conditional posterior distribution of the partworths is:

$$P(\boldsymbol{w}_i|\boldsymbol{w}_0, \sigma, D, data) \propto$$

$$exp\left(-\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{\left(x_{ij}w_i - y_{ij}\right)^2}{2\sigma^2} - \frac{1}{2}\sum_{i=1}^{I}(w_i - w_0)^T D^{-1}(w_i - w_0)\right)$$

Where i,j are different tasks

(19)

We point the reader to (Evgeniou, Toubia, & Pontil, 2007) for the derivations of (18) and (19).

Now recall the PLapRLS minimization from (12)(15):

$$\min_{f\in H_K}\frac{1}{l}\sum_{i=1}^{l+u}\left(y_i^{\mu} - f(x_i^{\mu})\right)^2 + \gamma_A\|f\|_K^2 + \frac{\gamma_I}{(u+l)^2}\sum_{i,j=1}^{u+l}W_{ij}\left(f(x_i) - f(x_j)\right)^2$$

(20)

Where i,j are different examples from the same task. We can transform the loss function of equation (20) to the loss function of equation (18), by assuming linear models, and block matrix version of the Laplacian such that, for each task t:

Use f_t(x)=w_tx, in the loss term,

Use $f_t(\mathrm{x}) - f_0(\mathrm{x}) = (w_t - w_0)\mathrm{x}$ in the regularization term

$W_{ij} = D_t/(x_i - x_j)w_t w_t^T (x_i - x_j)^T$

Therefore, the RR-Het and the Hierarchical Bayes approaches can be seen a special cases of the PLapRLS algorithm, where:

1)$f(\mathrm{x})$ is linear

2)The manifold does not consider the geodesic similarities of each pair of points, but it rather penalizes them based on a covariance like matrix D, of the coefficients w_t of the estimated linear functions $f_t(\mathrm{x}) = \mathrm{x}w_t$, for each task t.

3)RR-Het estimates the matrix D through cross validation, or

Expectation Maximization like iterations (Evgeniou, Toubia, & Pontil, 2007)

Summary and Contributions

We presented PLapRLS, a semi-supervised inspired approach to learning models for related tasks. Our experiments on benchmark datasets show that the value of our methods, as well as other multi-task learning approaches, is more evident, when the task specific datasets have few examples, and the properly penalized related task examples can enrich the training set.

Our approach outperforms all published results on the same dataset, with the same experimental setting. Also, our approach does well without having to learn the cross-task similarity through iterative EM like approaches. Instead, we can rely on the regularized manifold learning setting, but we may need to estimate the intrinsic space regularization parameters (Belkin & Niyogi, 2004), and the related task penalty through cross validation. Our penalized Laplacian approach makes no assumptions about consistent cross task similarities, but it relies on the similarity of training examples in the manifold space, to estimate the relevance of the examples from the related tasks.

CHAPTER 3: SUPERVISED CONTENT WORD EXTRACTION

Introduction

With the plethora of text documents available today in electronic form, we often need to reduce the textual information, for both end user consumption, and for other automated applications. By reducing a text document to a subset of the most informative content words, we can potentially construct machine generated summaries, use the content words for information retrieval tasks, or utilize them as informative features in document classification (Rogati & Yang, 2002). One practical use case for such an application is the delivery of summarized news articles to mobile phones with Personal Digital Assistance (PDA) features. While the user may want to be informed about a specific news item, it is more usable to deliver smaller summary of regular news article, which can be easily read in a small screen, rather than a regular multipage news item.

In this chapter we present a supervised learning approach for the identification of content words from a corpus of news articles and journalist generated summaries. The informal summaries of the news articles, in our corpus, usually appear as captions for images relevant to the news article. Like the rest of this thesis, we approach the supervised content-word learning problem as a choice based ranking problem. We know the chosen content-words for the summary, and the ones not-chosen, but we do not have any information about the partial sort order between the chosen words, or the partial order between the words in the not-chosen set.

We evaluate the quality of the predicted content-words by four different experiments:

a) We evaluate the content words as a feature selection method: we compare how well we can classify the genre of the articles using only content words vs. using the whole text of the original article.

b) Information Retrieval to evaluate the usefulness of the content words as a query method, to retrieve the relevant caption that appeared with a news story. We evaluate the text similarity between the caption database and the content-words vs. the whole text of the original article.

c) We construct summaries based on the identified content-words, and contact psychophysics experiments to evaluate the quality of the constructed summaries vs. summaries constructed by a standard unsupervised summarization technique.

d) Finally we use the content words to query the image database based on the text description of the images, and contact psychophysics experiments where we evaluate how appropriate the selected image is, for an unseen news article. We approach the problem as an information retrieval task (Salton., 1968) where the query is the news story itself, and our results will be a list of candidate pictures or photographs.

Overview

In this chapter we present two content extraction engines: WordEx and SentEx. WordEx attempts to extract the important words from a text document. These are the content-words that summarize the content of the document. SentEx uses WordEx in order to extract important sentences from the text document. We first describe how WordEx chooses the content words. We then describe how SentEx uses the output of WordEx in order to extract sentences from the original text to create cohesive summaries. Finally, we present four applications for WordEx and SentEx (classification, information retrieval, summarization, and text query based image retrieval), evaluating their performance.

WordEx

We use supervised learning to train WordEx to predict content-words on a corpus of news articles. The news stories have three parts: the news story text, an image accompanying the story text, and a caption describing the image. In our corpus, the caption accompanying the image is a summary of the main news story, and most of the content words of the caption, are included in the main news story. WordEx therefore takes a news story text as input and classifies the words in the text into two categories: "content-words" and "other-words". Words classified as "content-words" are ones that are in the story text and also in the caption. Words classified as "other-words" are ones that are in the story text but are not in the caption.

WordEx uses a supervised learning to perform the classification. Our training set consisted of 9394 instances. Each instance is described by a feature vector. The current version of WordEx uses nine features. Each instance is a word, and therefore each of these features is a feature that describes the word. For example, "the number of times the word appears in the story text" is a feature. The feature vector is described in detail in the following section.

The 9394 training instances were obtained from a set of 938 news stories. These news stories were taken randomly from our complete data set described in the next section. Therefore, the stories spanned the whole time period from November 2001 to April 2003. Since part of the objective of this research was to construct automated summarization algorithms, we kept only news articles that had at least 100 words.

From these 938 news stories, with at least one captioned image, we paired the story text with the union of the caption texts appearing with that story. There were 938 story texts, and 1138 caption texts. Therefore, there were more captions than story texts because some news stories had more than one image, and therefore more than one caption. Each word in a story text is an instance that can be used for training. All the story texts have a total of 187882 words. Each word can be classified as a "content-word" or "other-word", depending on whether it also appears in one of the captions associated with the story text. From these 187882 words, 15224 of them were content-words, and 172658 of them were other-words. Based on this definition, our corpus has on average 8% of content-words, and 92% of other-words.

To train WordEx, we did not use all of the 187882 instances. We used a 5% balanced sample. As can be seen in Table 7, only 9394 instances were used, with a roughly equal number of content-words and other-words.

Table 7 The training set used to train WordEx

instances of class content-words	4711
instances of class other-words	4683
Total number of training instances	9394

The feature vector

The story text given to WordEx as input is split up into tokens. Each token is a single word. If a word appears more than once in the text, only a single token is created for that word. A list of 524 stop words is used to remove common words that provide little or no information. For example, words like "and", "if", and "or" are in the list of stop words. No stemming is performed in this version of WordEx. After the stop words are removed, a feature vector is created for each of the remaining words. The following list describes the nine features present in the feature vector of each token.

1. The number of times the word appears in the whole story text.
2. The number of times the word appears in the title of the story text.
3. The number of times the word appears capitalized in the body of the story text.
4. The number of times the word appears capitalized in the title of the story.
5. The part of speech of the first occurrence of the word. (Noun, Verb, etc.) (Brill 1992)
6. The position of the first appearance of the word in the whole story text.

7. The length of the word in characters.
8. The TFIDF score of the word. (Salton 1991).
9. The Information Gain score of the word (Mitchell 1997)

For example, the feature vector created for the word "minister" would be:

3,1,1,1,NNP,2,8,11.669298,0.17626

The first feature has the value of 3, and it indicates that the word "minister" appeared three times in the whole story text (once in the title, and twice in the body of the story). The second feature has the value of 1. This indicates that the word "minister" appears once in the title of the story. The third feature has the value of 1, because the word "minister" appears capitalized once in the body of the story text. The fourth feature also has the value of 1, because the word "minister" appears capitalized once in the title of the story.

The fifth feature indicates the part of speech. The value NNP means that the part-of-speech tagger (Brill, 1992) recognized this word as a proper noun.

The sixth feature has the value of 2. This is the position where the word "minister" appeared for the first time in the whole story text (title included). The word "minister" has eight characters (letters). This is indicated by the value 8 in the vector.

The last two features are the TFIDF and Information Gain scores.

Data

The data we used for our experiments was taken from the Yahoo! News website (Yahoo!). We found the data from this site appropriate because the captions of the images were significantly larger than those of other on-line news sites.

We collected news stories with their images and captions within the time period of 18 months: from November 2001 until April 2003. During this time we collected a total of 35766 news stories. For all of these, we stored the story text and the caption text. In order to conserve disk space, we only stored the images of 3695 of these stories, because the actual images were not needed for the actual construction of the machine learning algorithms. The images were only used for evaluation purposes.

Details

On average, each story text has about 565 words, with stories ranging from as low as 20 words, to as high as 2000 words. Each story has a title. On average, each title has 7 words, with titles ranging from 1 to 16 words.

On average, each caption has about 50 words, with captions ranging from 10 to 140 words. Each news story usually appears on-line with one image. Sometimes, however there can be two images. Each image has its own caption. We say these captions are associated with the news story, because the images they describe appeared on-line as part of the news story.

SentEx

SentEx calculates the average content-word score for each sentence based on the content-words that each sentence contains. Each content-word in the sentence has a confidence score, produced by the base classifier used in WordEx. The formula for the score of each sentence is:

$$S = \frac{1}{n}\sum_{i=1}^{K} C_i$$

S is the score of the sentence. The sentence has a total of n words, and k content-words. Each content-word is numbered from 1 to *K*. C_i is the confidence score of word i. The summary created by SentEx is the collection of the highest-scoring sentences. If SentEx is asked to create a 20% summary for example, it returns 20% of the original sentences, with the highest score.

Evaluation of Applications

We evaluate the quality of the predicted content-words by four different experiments:

1. **Classification:** We test how well we can perform multiclass classification based on the content-words only vs. using the whole text of the original article

2. **Information Retrieval:** We test how well we can retrieve the relevant caption that appeared with a news story, by evaluating similarity between the caption database and the content-words vs. the whole text of the original article

3. **Summarization:** We construct summaries based on the identified content-words, and ask users to evaluate the quality of the constructed summaries vs. summaries constructed by a standard unsupervised summarization technique.

4. **Image Retrieval:** We use the content-word predictions to automatically retrieve an appropriate picture or photograph for a news story, by evaluating the similarity with the caption database, stored with the images.

Table 8 Classification accuracy using the complete text of the articles vs. the content-words only

Size of training set	Percentage Accuracy	
	full story	content-words
10%	67.13	63.25
20%	73.15	69.92
30%	76.2	72.59
40%	75.92	72.89
50%	75.92	72.7
60%	74.94	71.68
70%	73.64	70.36
80%	70.73	67.61
90%	63.75	60.18

Classification

We took a corpus of 294 news stories, each one belonging to one of six categories: Business, Entertainment, Health, Politics, Sports, Technology. We performed classification on these stories using Naive Bayes. Some of the stories were used as the training set and the rest as the test set. We performed 100 iterations of the test and took the average percentage accuracy. Then we extracted the content-words from each news story and performed exactly the same classification, but this time instead of using the full story, we used only the content-words. We used a varying size for the training set and test set. Table 8 shows the results. We can see that the classification results are slightly worse when using only the content-words. However, the difference is statistically insignificant with $p>0.99$

Information Retrieval

The caption of the image can also be thought of as the summary of the news story. We can use the content-words extracted by WordEx to retrieve this caption. Once the content-words are predicted, we use them as the input to a TFIDF Information Retrieval query, which in turn returns a ranked list of candidate captions. Table 9 shows the results for two indexed data sets: a small data set consisting of 938 news stories, and a large data set consisting of 11019 news stories. We compare two different retrieval queries:

1)the full story text (on average, each query contains 560 words)

2)the predicted content-words, which were the output of WordEx (on average, each query contains 62 words)

We observe that in the Information Retrieval task, both the full text, and the content-word approach have worse performance, when used on the larger index. This seems to be a weakness of the TFIDF approach used for the similarity evaluations, but it warrants more investigation. The performance of the two methods is the same with p>0.99 in the small data set experiment. In the larger data set experiment, (although the two approaches have again similar percentages of correctly retrieved captions), the performance of the two methods is statistically significantly different.

Table 9 Number of correct captions retrieved using the full story, the predicted content-words and the actual content-words as the query

	Number of correct captions			
	Small data set		**Large data set**	
	(938 news stories)		(11019 news stories)	
Rank	full story	Predicted content-words	full story	predicted content-words
1st	599	599	3024	2852
2nd	81	83	113	1193
3rd	36	38	58	719
4th	18	11	39	530
5th	15	15	19	368
total	749	746	5897	5662
percentage	80%	80%	53%	51%

Summarization

We used SentEx to create a 20% summary of 99 news articles. These news articles were downloaded from the Yahoo news web site between 11 Nov 2001 and 25 Nov 2001. We evaluated our results by asking humans to evaluate them. We created a web interface where evaluators could rate how good the summary was. The evaluator was presented with a news story and summary, chosen randomly from the 99 news stories. The evaluator could choose from four options: poor, adequate, very good, and cannot decide. The results are presented in Table 10. The table compares our results to those obtained by using OTS, which is described below.

Open Text Summarizer (OTS)

The Open Text Summarizer (OTS 2003) is an open source library that uses a rule-based system with unsupervised scoring techniques to grade sentences for text summarization. The OTS grader consists of three components:

1)A syntax grader

2)A term count grader, and

3)A term frequency grader

Table 10 Psychophysics experiments for the Text Summarization task using SentEx vs. OTS. We note that the two sets of the responses differ significantly (p>0.95) and the SentEx user ratings are better than the OTS ratings.

	OTS		**SentEx**	
Rating	number of ratings	percentage	number of ratings	percentage
very good	27	43%	41	48%
adequate	24	38%	30	35%
poor	12	19%	14	16%
cannot decide	1 (ignored)	(ignored)	3 (ignored)	(ignored)
total	**63**	**100%**	**85**	**100%**

One key feature of the OTS is that it runs a stemmer before it assigns weights to the different words to count derivatives of the same words properly. Before OTS selects the content-words, it removes frequent English words, based on a predefined stop-list. The syntax grader gives higher scores to the first line or the title of the document, if it has one, and the first line of each paragraph. The term count grader, counts the number of times a word appears in the document, and the term frequency grader discounts the weight of words that appear in multiple documents frequently, by calculating the TFIDF score.

Image Retrieval

In the final experiment we use the content-word predictions to automatically find an appropriate picture or photograph for a news story. On-line news sites usually accompany their news stories with an appropriate picture or photograph. To keep the terminology consistent, the name we will use for these pictures or photographs is images. On most news sites, these images also have a caption that describes the image in the context of the news article.

Therefore, we construct a database of past news articles, with the following components per article:

1)The story text - the news story

2)The image - picture, photograph, etc.

3)The caption text - the caption describing the image

In this set of experiments, we train supervised algorithms that learn how to select the appropriate image file for a new article, from a database of previously stored images with their captions.

In this chapter we will only use a text-based approach. We do not perform any image processing. The only data we will use to solve the stated problem is the caption text and the story text. We try to predict if an image is appropriate for a certain story, by using the caption of the image, and not the image itself.

Like in the Information Retrieval experiment described above, we use the text similarity between the predicted content-words, and the available captions in the corpus, to select the closest caption from our corpus. We then select the image that was associated with the stored caption, and attach it to the new article.

Table 11 Psychophysics experiments for the image retrieval experiments

	actual articles		**predicted articles**	
Rating	number of ratings	percentage	number of ratings	percentage
very well	339	45%	311	40%
adequately	239	32%	228	30%
poorly	176	23%	226	30%
cannot decide	46 (ignored)	(ignored)	71 (ignored)	(ignored)
total	**754**	**100 %**	**765**	**100%**

Table 11 shows how human evaluators rated the images that our algorithm selected. We compared this to how they evaluated articles that appeared on the actual news site. An actual article is a story text, together with the actual image that was used on the on-line news site. It is a news story just like it appeared on-line. A predicted article is a news story text, together with an image that was predicted by our algorithm. The algorithm uses WordEx to predict a suitable image for the story text.

We had 65 people taking part in the evaluation. In total, 1519 articles were ranked. We do not consider the "cannot decide" response as an answer. We can see that although the rankings for the predicted articles are slightly worse than those of the actual articles, the two results are statistically similar with p>0.95

Summary and Contributions

We presented a supervised ranking learning approach for content-word extraction. We created statistical and syntactic information features, which we used to train supervised algorithms to predict the content-words that appear in the journalist generated summary of each article. We evaluated the quality of the predicted content-words by running four different experiments: a) as a feature selection method, for classification b) as n Information Retrieval to evaluate the usefulness of the content words as a text query method, c) as an automatic summarization approach, where the content words identified the content sentences, and as d) as a query method for image retrieval.

Our approach can reduce the size of the text to about 10% of the original text, and still maintain the important content of the original text. This has been shown using the four evaluation methods above. When we used WordEx to extract content sentences for summarization, users rate the summarization quality significantly higher than a standard unsupervised summarization method. The same supervised content-word learning approach also allowed us to build a system that retrieves relevant images files for news articles, from a database collected images with textual descriptions. In our psychophysics evaluation, the users rated the retrieved image quality with statistically similar satisfaction ratings ($P>0.95$) as they rated the original samples from the corpus.

CHAPTER 4: COMPANY ENTITY MATCHING

Introduction

In this Chapter, we study the problem of fuzzy company entity matching between disparate databases. The problem arises from manual entry of company name records in different Enterprise Resource Planning systems, like Accounts Payable, Accounts Receivables, Customer Relationship Management, Supply Chain Management and other corporate systems. When a corporation tries to integrate such systems, to build data warehouses or to rationalize their databases, they find out that not only do they have duplicate records of the same corporate entity within the same database, but they also have different fields between databases, different abbreviations, misspellings, and other issues that make integrating these data sources through standard relational database operations impossible. In this chapter we study how to automate the multi-field record linkage problem by treating it as a supervised ranking problem.

The Company Record Matching problem

Most ERP systems, which aggregate corporate entity information, try to capture the basic company identification information that would help a human user uniquely identify a particular company record. With rare exceptions the captured fields include the fields shown in Table 12. More recent files may also include electronic contact information, like contact email address or a company URL. In the files we have processed those cases are still a rare exception.

Table 12 Example business record fields

Field	Example 1	Example 2	Standardized Record
Business Name	Open Ratings	ORI	Open Ratings, Incorporated
Phone Number	781-895-6109	781-895-6109	+1-781-895-6100
Address Field 1	200 West Street	200 West Str	200 West St
Address Field 2	First Floor		
City	Waltham	Waltham	Waltham
State	MA	MA	MA
Postal Code	02451		02451-1121
Country	US		USA

Information service providers like D&B, Experian, InfoUSA, Bureau van Dijk, CreditInfo, and Open Ratings provide such reference files cleansed and from duplicates, and with standardized records indexed by unique identifiers. They also provide data cleansing services with automated algorithms, mostly built on decision trees. The market leader in the business information space is D&B which indexes its records by the Data Universal Numbering System (DUNS), a unique business identifier currently required for doing business with US government, the European Union, and the United Nations. The algorithms presented in this chapter are compared on the proprietary D&B matching engine for benchmarking purposes.

Table 13 Typical matching problems

Field	Record 1	Record 2	Description
Business Name	Open Ratings	ORI	Acronyms
Business Name	Open Ratings	OpenRatings	Merged Words
Business Name	Open Ratings	Operating	Misspellings
Business Name	SideStep	Kayak.com	Different Names; Acquisition
Business Name	General Electric	NBC	Different Name; Subsidiary
Business Name	IBM	Intern. Bus. Mach.	Abbreviation
Business Name	IBM Corp.	I.B.M.	Abbreviation; Missing Words
Phone Number	781-895-6109	781-895-6100	Different phone number. Sometimes both correct, so likely to belong to same block.
City	Boston		Missing City
City	N. York	New York	Abbreviation
City	City of Cambridge	Cambridge, City of	Different word order
City	Boston	Waltham	Different Cities. Mistake or Change of Address.
Postal Code	02139	02142	Different Postal Codes;
Postal Code	02451		Missing Postal Code

The business problem of company name matching problem arises in three frequent use cases:

1)Merging multiple corporate databases to a single warehouse.

2)Data cleansing of a corporate database, by using a reference file.

3)Data enrichment by combining two reference files, that provide different business characteristics. For example, we may want to enrich a reference file with financial data, with socioeconomic characteristics, like 'Minority owned', 'Woman Owned', and 'Small Business Status' data.

The first problem is usually approached by cleansing the individual databases based on a reference file, and indexing the records on the unique identifier provided by the reference file. Therefore all three use cases are essentially an information retrieval problem, where the records of the less standardized file are matched on more a standardized one, and enriched with a unique identifier.

Related Work

The record linkage problem has been investigated for decades (Fellegi & Sunter, 1969) (Newcombe, 1967)in several domains, including company entity matching, name matching for census data, and medical records matching. Most of the literature focuses on rule based methods, statistical, or machine learning methods for training algorithms with little labeled data from a pair of two files (Ipeirotis, Verykios, & Elmagarmid, 2007). The objective is to match the two files, and then merge their fields. Several methods spanning the whole spectrum of machine learning have been tried, including clustering methods with Expectation Maximization (Dempster, Laird, & Rubin, 1977), semi-supervised learning (Winkler W. E., 2002), tree based methods (Cochinwala, Kurien, Lalk, & Shasha, 2001), and SVMs (Bilenko, Mooney, Cohen, Ravikumar, & Fienberg, 2003). Our approach is different from the existing methods in that we want to be able to have one general algorithm, which will work well enough on unseen input files requiring cleansing. Each record of the query file represents a separate multri-attribute query, for which we want to find automatically the most likely correct match. Like the previous chapters of this thesis, our record linkage application is approached as a ranking problem, where the objective is to identify the most likely candidate to be a correct match for the company referred to in the query.

Information Retrieval for Company Record Matching

One approach to the Company Record Matching problem would be to build a text index, and run fuzzy queries for each candidate name. While this approach produces high quality candidates, it is not scalable for large database merging tasks, because the scoring algorithm is not optimized for the particular problem. When we want to merge databases with millions of records, the objective is to select single unique matches with extremely high confidence of correctness. We propose a supervised learning approach where the learning algorithm can distinguish the most likely candidate to be correct.

Our approached is based on creating several distance metric scores that measure the likelihood of two business records being dissimilar, based on typical matching problems, such as the examples shown on Table 13. Then we train a supervised learning algorithm, on the vectors of differences of the labeled examples, to learn a ranking function, similar to the examples described in the previous chapters of this thesis. Finally, we estimate a threshold for separating the correct matches using cross validation, so that we have zero false positive matches on the training set.

Implementation details

Indexing the reference file

The system in implemented by using the Lucene package from the Apache project. Lucene is a customizable text indexing, and search engine, that allows us to implement the distance metrics needed for Company Matcher problem. Before the reference filed is indexed, we preprocess it so that the input name is standardized: we convert common tokens like "Inc." and "Corp." and converted to "Incorporated", "Corporation". We standardize US addresses based on the United States Postal Service guidelines, so for example we convert "Street", and "Str" to "St". Additionally, words are unpluralized, and numbers are replaced with numeric representations. For example "One IKEA Way" is stored as "1 IKEA Way". The address field is parsed into the numeric part, the street name and PO Box number, if they are present. We discard floor, and suite numbers.

Score metrics for candidate matches

Then we create candidates for the records in the input file to be matched using fuzzy search queries. The queries standardize the input record, using the same preprocessing step that we used to index the reference file. The queries are hierarchical, the first query is the most restrictive and the last one is the loosest, until we generate at least 25 candidate matches per input record.

We generate our features by calculating a number of metrics between the input and the candidate index records:

Levenshtein distance: The Levenshtein distance measures the edit distance between two strings. The metric counts the minimum number of additions and deletions required to create the second string from the first one. We calculate the Levenshtein distance for the standardized name. We also calculate the Levenshtein distance for the standardized name with its tokens alphabetically sorted. The distances on the alphabetically sorted tokens help create informative similarity metrics in cases where the tokens are sorted differently in the two sources, or when some token may be missing. We also calculate the Levenshtein distance of the metaphone string of the standardized name, the full address, the alphabetically sorted tokens string of the name, and of the full address (to account for cases where for example the "City" field is found in the "Address Field 1"), the street name, the metaphone of the address, street number, Po Box, city, the metaphone of the city, and the zip code. For all of the above, we also calculate the normalized distance (the Levenshtein distance is divided by the length of the longer of the two records), and the Levenshtein distance between the least frequent tokens in the input and index records.

Cosine similarity of TFIDF weighted vectors representing the tokenized Business Name, the full address, the street name, the address number, PO Box, city, and the zip code.

Jaro-Winkler string similarity (Winkler, 2006) metrics for the full name, the full address, the street name, the city, and for the least frequent tokens. The Jaro-Winkler metric is mostly used for short string similarity comparisons, particularly for record linkage problems in large databases like the census data. The metric is normalized so that a score of zero means the compared strings are totally different, while a score of one means that they are identical. We include more information about the Jaro-Winkler metric in the Appendix II section.

We also track as a nominal attribute the number queries required to generate at least 25 candidates, before terminating the retrieval of candidates. The depth of the queries required to retrieve enough candidates represents the types of fuzzy queries that were required. Additionally, we maintain binary attributes indicating whether the street address for either the input or the index record is blank, and a binary attribute for the presence of the street number.

Supervised ranking learning for matching

Our matching algorithm consists of a two stage prediction process:

1)A binary classifier makes a prediction whether the candidate matches are classified as 'match', or 'nomatch'. If there is a single candidate classified as 'match', and all other candidates are classified as 'nomatch', we assume that we have a single unique match, and link the records as matched. If all the candidates are classified as 'nomatch', we assume the record does not exist in the database corpus used to create the index.

2)If there is more than one candidate in the 'match category', then we need to select a winner as the potential match. We use the same approach described in the first chapter of the thesis (3)(4), where we train a supervised algorithm on the vectors of differences of the different candidate records.

Experiments

We manually created a training set of 20000 positive examples, from two reference files, from two separate information providers of US company data. SA, the first reference file, includes 11 million US businesses, with geocoding data, and PF, the second reference file, contains 13 million businesses with socioeconomic and demographic data fields.

We indexed the PF reference file, and ran queries to generate the score metrics for the 20000 positive examples from the SA file. Then we removed the positive examples from the PF file, and ran the same 20,000 queries to generate the negative examples for the two training files above. We use as our negative examples the top 20,000 records based on the Lucene Similarity metric.

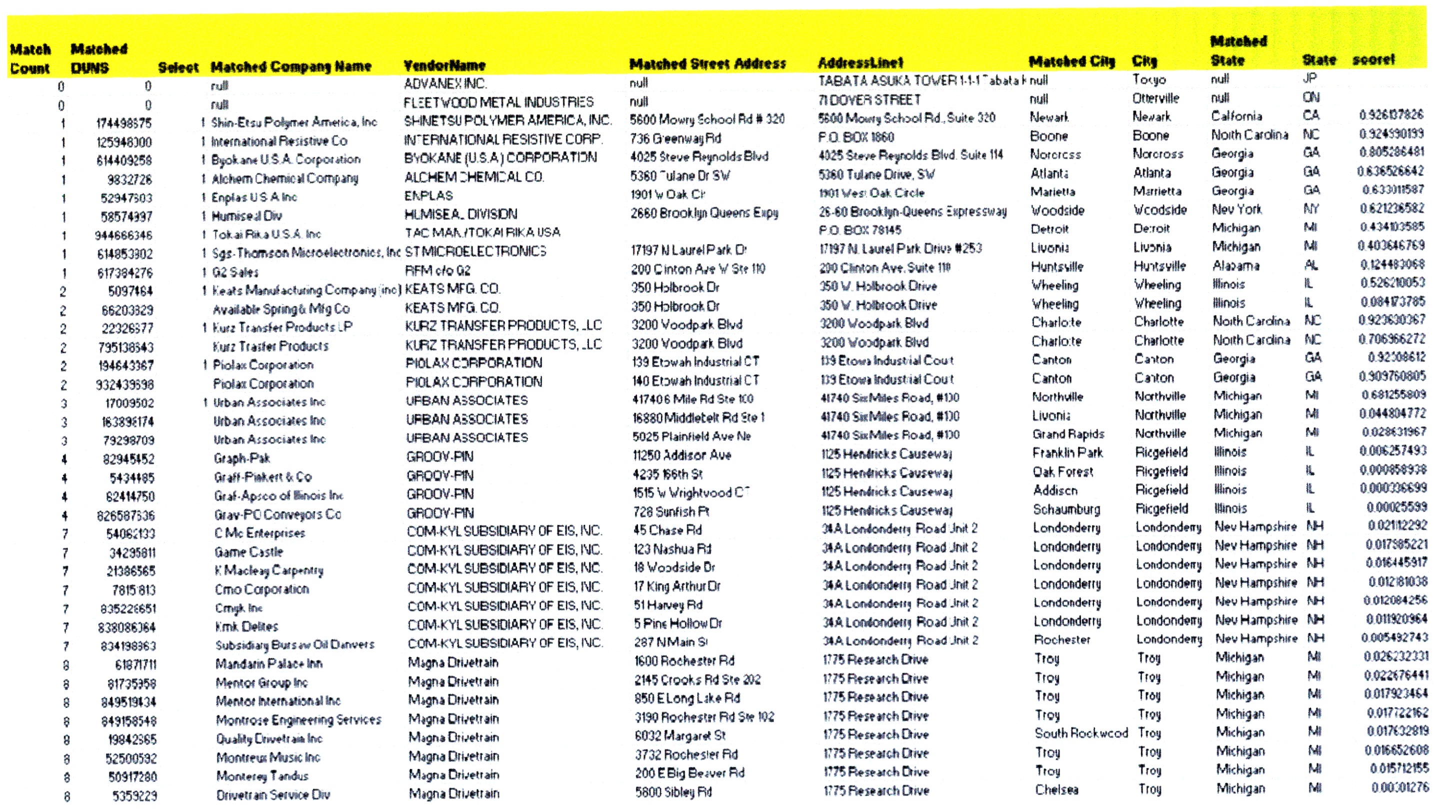
We train one overall model, that includes a binary classifier for match/nomatch, and a ranking model for choosing the best match, when there is more than one potential matching record. We ran files from three different industries for the purposes of data cleansing, and report the results in Table 14

Table 14 Company Matcher performance

	SA file	Defense Industry suppliers	Tools manufacturer suppliers
Correct matches	17608	17930	3119
Mismatches	846	163	198
Correct non Matches	14150	18329	3265
False non matches	159	167	388
Precision	95.42%	99.10%	94.03%
Recall	99.10%	99.08%	88.94%

Summary and Contributions

We presented a supervised ranking learning approach to the record linkage problem of company entity matching. The system produces high enough accuracies to allow automated record linkage, and data cleansing of corporate databases. The approach combined ideas from the first two chapters of this thesis. In a the second chapter we created features that described the potential significance of each keyword based on syntactical, formatting, and statistical, and structural text metrics, and then built supervised learning algorithms that predicted the content words. In this chapter we generated features that described the phonetic, and orthographic characteristics of the different tokens, and used standard phonetic similarity, and string distance metrics to create measures of similarity for the different company entity descriptors. Then, we created supervised ranking learning algorithms, which relied on classifying vectors of differences to rank likely candidates for a record linkage, like the task specific ranking learning approach of the first chapter. Our approach allowed us to link two multimillion record databases, and to automatically standardize supply base data files from different industries.



Match Count	Matched DUNS	Select	Matched Company Name	VendorName	Matched Street Address	AddressLine1	Matched City	City	Matched State	State	score1
0	0		null	ADVANEX INC.	null	TABATA ASUKA TOWER 1-1-1 Tabata k	null	Tokyo	null	JP	
0	0		null	FLEETWOOD METAL INDUSTRIES	null	71 DOVER STREET	null	Otterville	null	ON	
1	174498875	1	Shin-Etsu Polymer America, Inc	SHINETSU POLYMER AMERICA, INC.	5600 Mowry School Rd # 320	5600 Mowry School Rd., Suite 320	Newark	Newark	California	CA	0.926107826
1	125948300	1	International Resistive Co	INTERNATIONAL RESISTIVE CORP.	736 Greenway Rd	P.O. BOX 1860	Boone	Boone	North Carolina	NC	0.924590139
1	614409258	1	Byokane U.S.A. Corporation	BYOKANE (U.S.A.) CORPORATION	4025 Steve Reynolds Blvd	4025 Steve Reynolds Blvd. Suite 114	Norcross	Norcross	Georgia	GA	0.805286481
1	9832726	1	Alchem Chemical Company	ALCHEM CHEMICAL CO.	5360 Tulane Dr SW	5360 Tulane Drive, SW	Atlanta	Atlanta	Georgia	GA	0.636526642
1	52947303	1	Enplas U.S.A Inc	ENPLAS	1901 W Oak Cir	1901 West Oak Circle	Marietta	Marrietta	Georgia	GA	0.633011587
1	58574997	1	Humiseal Div	HUMISEAL DIVISION	2660 Brooklyn Queens Expy	26-60 Brooklyn-Queens Expressway	Woodside	Woodside	New York	NY	0.621236582
1	944666346	1	Tokai Rika U.S.A. Inc	TAC MAN./TOKAI RIKA USA		P.O. BOX 78145	Detroit	Detroit	Michigan	MI	0.434103585
1	614853302	1	Sgs-Thomson Microelectronics, Inc	STMICROELECTRONICS	17197 N Laurel Park Dr	17197 N. Laurel Park Drive #253	Livonia	Livonia	Michigan	MI	0.403646769
1	617384276	1	G2 Sales	RFM c/o G2	200 Clinton Ave W Ste 110	200 Clinton Ave. Suite 110	Huntsville	Huntsville	Alabama	AL	0.124483068
2	5097464	1	Keats Manufacturing Company (inc)	KEATS MFG. CO.	350 Holbrook Dr	350 W. Holbrook Drive	Wheeling	Wheeling	Illinois	IL	0.526210053
2	66203329		Available Spring & Mfg Co	KEATS MFG. CO.	350 Holbrook Dr	350 W. Holbrook Drive	Wheeling	Wheeling	Illinois	IL	0.084173785
2	22326377	1	Kurz Transfer Products LP	KURZ TRANSFER PRODUCTS, LLC	3200 Woodpark Blvd	3200 Woodpark Blvd	Charlotte	Charlotte	North Carolina	NC	0.923630367
2	795138643		Kurz Trasfer Products	KURZ TRANSFER PRODUCTS, LLC	3200 Woodpark Blvd	3200 Woodpark Blvd	Charlotte	Charlotte	North Carolina	NC	0.706366272
2	194643367	1	Piolax Corporation	PIOLAX CORPORATION	139 Etowah Industrial CT	139 Etowa Industrial Court	Canton	Canton	Georgia	GA	0.92308612
2	932439698		Piolax Corporation	PIOLAX CORPORATION	140 Etowah Industrial CT	139 Etowa Industrial Court	Canton	Canton	Georgia	GA	0.909760805
3	17009502	1	Urban Associates Inc	URBAN ASSOCIATES	41740 6 Mile Rd Ste 100	41740 SixMiles Road, #100	Northville	Northville	Michigan	MI	0.681255809
3	163898174		Urban Associates Inc	URBAN ASSOCIATES	16880 Middlebelt Rd Ste 1	41740 SixMiles Road, #100	Livonia	Northville	Michigan	MI	0.044804772
3	79298709		Urban Associates Inc	URBAN ASSOCIATES	5025 Plainfield Ave Ne	41740 SixMiles Road, #100	Grand Rapids	Northville	Michigan	MI	0.028631967
4	82945452		Graph-Pak	GROOV-PIN	11250 Addison Ave	1125 Hendricks Causeway	Franklin Park	Ridgefield	Illinois	IL	0.006257493
4	5434485		Graff-Pinkert & Co	GROOV-PIN	4235 166th St	1125 Hendricks Causeway	Oak Forest	Ridgefield	Illinois	IL	0.000858938
4	62414750		Graf-Apsco of Illinois Inc	GROOV-PIN	1515 W Wrightwood CT	1125 Hendricks Causeway	Addison	Ridgefield	Illinois	IL	0.000336699
4	826587536		Grav-PC Conveyors Co	GROOV-PIN	728 Sunfish Pt	1125 Hendricks Causeway	Schaumburg	Ridgefield	Illinois	IL	0.00025599
7	54062133		C Mc Enterprises	COM-KYL SUBSIDIARY OF EIS, INC.	45 Chase Rd	34A Londonderry Road Unit 2	Londonderry	Londonderry	New Hampshire	NH	0.021112292
7	34295811		Game Castle	COM-KYL SUBSIDIARY OF EIS, INC.	123 Nashua Rd	34A Londonderry Road Unit 2	Londonderry	Londonderry	New Hampshire	NH	0.017985221
7	21386565		K Macleay Carpentry	COM-KYL SUBSIDIARY OF EIS, INC.	18 Woodside Dr	34A Londonderry Road Unit 2	Londonderry	Londonderry	New Hampshire	NH	0.016445917
7	7815813		Cmo Corporation	COM-KYL SUBSIDIARY OF EIS, INC.	17 King Arthur Dr	34A Londonderry Road Unit 2	Londonderry	Londonderry	New Hampshire	NH	0.012181038
7	835226651		Cmyk Inc	COM-KYL SUBSIDIARY OF EIS, INC.	51 Harvey Rd	34A Londonderry Road Unit 2	Londonderry	Londonderry	New Hampshire	NH	0.012084256
7	838086364		Kmk Delites	COM-KYL SUBSIDIARY OF EIS, INC.	5 Pine Hollow Dr	34A Londonderry Road Unit 2	Londonderry	Londonderry	New Hampshire	NH	0.011920964
7	834198363		Subsidiary Burs ... Oil Danvers	COM-KYL SUBSIDIARY OF EIS, INC.	287 N Main St	34A Londonderry Road Unit 2	Rochester	Londonderry	New Hampshire	NH	0.005492743
8	61871711		Mandarin Palace Inn	Magna Drivetrain	1600 Rochester Rd	1775 Research Drive	Troy	Troy	Michigan	MI	0.026232331
8	81735358		Mentor Group Inc	Magna Drivetrain	2145 Crooks Rd Ste 202	1775 Research Drive	Troy	Troy	Michigan	MI	0.022676441
8	849519434		Mentor International Inc	Magna Drivetrain	850 E Long Lake Rd	1775 Research Drive	Troy	Troy	Michigan	MI	0.017923464
8	849158548		Montrose Engineering Services	Magna Drivetrain	3190 Rochester Rd Ste 102	1775 Research Drive	Troy	Troy	Michigan	MI	0.017722162
8	19842365		Quality Drivetrain Inc	Magna Drivetrain	6032 Margaret St	1775 Research Drive	South Rockwood	Troy	Michigan	MI	0.017632819
8	52500592		Montreux Music Inc	Magna Drivetrain	3732 Rochester Rd	1775 Research Drive	Troy	Troy	Michigan	MI	0.016652608
8	50917280		Monterey Tandus	Magna Drivetrain	200 E Big Beaver Rd	1775 Research Drive	Troy	Troy	Michigan	MI	0.015712155
8	5359229		Drivetrain Service Div	Magna Drivetrain	5800 Sibley Rd	1775 Research Drive	Chelsea	Troy	Michigan	MI	0.00301276

Figure 5 Typical examples of single unique matches , and multiple candidates

SUMMARY AND THESIS CONTRIBUTIONS

As the available information in computer systems and the internet has exploded in size, it has become a difficult and time consuming task for humans to process and filter through the available information to find what human users are looking for. When users search for information, they would like to receive the most likely results to satisfy their query first. This thesis focused on ranking algorithms in applications that try to alleviate the information overload problem.

To summarize, the contributions of the thesis consist of two parts. First we examined ranking problems, as supervised learning problem. We generalized metric based, and choice based ranking problems, as well as information retrieval problems, as a supervised learning binary classification problem of pairwise feature differences. The binary classifier of pairwise differences is trained to identify the winning vs. the losing configuration of the two examples compared. We develop preference modeling, supervised content-word extraction applied the approach to text summarization, and a record linkage application, using this framework of ranking applications.

The second contribution of the thesis was motivated by the problem of learning from related tasks, specifically in the domain of user preference modeling. Such algorithms are useful when we have few examples per tasks, but many relatively similar tasks, which can inform the training of the task specific models. We developed algorithms that utilize regularized manifold learning, to account for the similarity of the foreign task data. We run experiments on real user datasets for preference modeling, and a benchmark dataset for multi-task learning (ILEA), on which our proposed algorithm outperforms the currently reported algorithms in the literature of multi-task learning (Evgeniou, Toubia, & Pontil, 2007).

APPENDIX I: RANKING METRICS AND LOSS FUNCTIONS

Throughout this thesis, we have solved different ranking problems by creating loss functions that minimize the number of discordant items. We show in this space that such minimizations also optimize several rank correlation metrics used in statistics, and information retrieval applications.

Kendall's τ coefficient

$$\tau = \frac{P-Q}{P+Q} = \frac{2P}{\binom{n}{2}} - 1 = 1 - \frac{2Q}{\binom{n}{2}}$$

Where:

P is the number of concordant pairs

Q is the number of discordant pairs

Value ranges from -1 for reverse rankings to +1 for same rankings.

0 implies independence

Spearman's rank correlation

One frequently used metric from statistics is the Spearman's rank correlation *ρ*. The computation of *ρ* assumes that we first convert the raw scores of two observations to rankings, and then calculate the rank differences *d*. Then we calculate *ρ* as:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

From the definition of *d*:

$$\sum d_i^2 = \sum (p_i - i)^2 \le \sum p_i^{\,2} + \sum i^2 \le \left(Q + n(n+1)/2\right)^2 + \sum i^2$$

Therefore:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \ge 1 - \frac{6\left[\left(Q + n(n+1)/2\right)^2 + \sum i^2\right]}{n(n^2-1)}$$

Therefore, when we minimize the number of discordant pairs: *Q*, we can get a higher lower bound for *ρ*,

Mean Average Precision[3]

In Information Retrieval the Average Precision of a system measures the relevance of a truncated list of candidate items, and their relative ranking produced by the system.

Therefore the Mean Average Precision is defined as:

[3] For a more detailed derivation of this bound, see (Joachims, 2002)

$$Mean\left(AvgPrec\right) = \frac{1}{n}\sum_{i=1}^{n}\frac{i}{p_i}$$
$where:$
$p_i = rank\ of\ sorted\ retrieved\ item\ i$
$n = number\ of\ ranked\ retrieved\ items$

A permutational computation shows that:

$$\sum_{i=1}^{n} p_i = Q + n\left(n+1\right)/2$$
$Q = number\ of\ discordant\ items$

We can calculate a lower bound on the Mean Average precision by solving the following minimization:

$$\min\frac{1}{n}\sum_{i=1}^{n}\frac{i}{p_i}$$
$subject\ to\ p_i < p_j \in \mathbb{N} \forall i < j$

Use Lagrange multipliers:

$$\min L = \frac{1}{n}\sum_{i=1}^{n}\frac{i}{p_i} + \mu\left[\sum_{i=1}^{n} p_i - Q - n(n+1)/2\right]$$

$$\frac{\partial L}{\partial p_i} = -\frac{i}{n} p_i^{-2} + \mu = 0 \Rightarrow p_i = \sqrt{\frac{i}{n\mu}}$$

$$L = \frac{1}{n}\sum_{i=1}^{n}\frac{i}{\sqrt{\frac{i}{n\mu}}} + \mu\left[\sum_{i=1}^{n}\sqrt{\frac{i}{n\mu}} - Q - n(n+1)/2\right] = 2\sqrt{\frac{\mu}{n}}\sum_{i=1}^{n}\sqrt{i} - \mu\left[Q + n(n+1)/2\right]$$

$$\frac{\partial L}{\partial \mu} = \sqrt{\frac{1}{n\mu}}\sum_{i=1}^{n}\sqrt{i} - \left[Q + n(n+1)/2\right] = 0 \Rightarrow \mu = \frac{1}{n}\left[\sum_{i=1}^{n}\sqrt{i}\Big/\left[Q + n(n+1)/2\right]\right]^2$$

$$\Rightarrow Mean(AvgPrec) \geq \frac{1}{n}\left(\sum_{i=1}^{n}\sqrt{i}\right)^2 \left[Q + n(n+1)/2\right]^{-1}$$

Therefore minimizing *Q* also maximizes the lower bound on the Mean Average Precision.

APPENDIX II: STRING SIMILARITY METRICS

Levenshtein Distance

The Levenshtein Distance measures the edit distance between two strings, by counting the insertions, deletions, and substitutions required to create one string from the second one(Levenshtein, 1966).

Consider the following example from (Levenshtein distance)

1. kitten → sitten (substitution of 's' for 'k')
2. sitten → sittin (substitution of 'i' for 'e')
3. sittin → sitting (insert 'g' at the end).

Therefore the Levenshtein Distance of the strings 'kitten' and 'sitting' is equal to 3.

Jaro-Winkler string similarity

The Jaron-Winkler string distance (Winkler W. E., 1999) is a measure of similarity between two strings, which is best suited for comparisons of short strings, like person names. Given two strings s_1, and s_2, their Jaro-Winkler distance d_w is (Jaro-Winkler):

$$d_w = d_j + lp(1 - d_j)$$

Where:

$d_j = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right)$ the Jaro distance

m is the number of matching characters

t is the number of transpositions

l is the length of common prefix, up to a maximum of four characters and p is a scaling factor controlling the weight of common prefixe, usually set to 0.1

Soundex

The soundex algorithm is a phonetic algorithm for the English language, trying to encode different spellings of the same sounds into the same encoding. Soundex was developed and patented in 1918, and used extensively by the US government (Russell, 1918) as a metric for fuzzy text matching for census data. The soundex representation consists of the first letter of a word, followed by three digits representing the first three consonants of the word. In the example above, 'openratings' has a soundex of O156, while the soundex of 'operating' has a soundex of O163. The consonant mapping table is shown on Table 15

Table 15 Soundex Consonant Mapping

Consonant	**Soundex mapping**
b, f, p, v	1
c, g, j, k, q, s, x, z	2
d, t	3
l	4
m, n	5
r	6

Metaphone

The Metaphone algorithm was developed as an improvement of the soundex algorithm (Philips, 1990). Unlike the soundex encoding, it uses variable length keys, and it uses a larger set of English language rules, to better represent the English pronunciation.

Table 16 Metaphone Implementation from Ruby's Text library

```
[ /([bcdfhjklmnpqrstvwxyz])\1+/,
                    '\1' ], # Remove doubled consonants except g
                                  # [PHP] remove c from regexp.
[ /^ae/,             'E' ],
[ /^[gkp]n/,         'N' ],
[ /^wr/,             'R' ],
[ /^x/,              'S' ],
[ /^wh/,             'W' ],
[ /mb$/,             'M' ],  # [PHP] remove $ from regexp.
[ /(?!^)sch/,       'SK' ],
[ /th/,              '0' ],
[ /t?ch|sh/,         'X' ],
[ /c(?=ia)/,         'X' ],
[ /[st](?=i[ao])/,   'X' ],
[ /s?c(?=[iey])/,    'S' ],
[ /[cq]/,            'K' ],
[ /dg(?=[iey])/,     'J' ],
[ /d/,               'T' ],
[ /g(?=h[^aeiou])/,      ],
[ /gn(ed)?/,         'N' ],
[ /([^g]|^)g(?=[iey])/,
                   '\1J' ],
[ /g+/,              'K' ],
[ /ph/,              'F' ],
[ /([aeiou])h(?=\b|[^aeiou])/,
                    '\1' ],
[ /[wy](?![aeiou])/,     ],
[ /z/,               'S' ],
[ /v/,               'F' ],
[ /(?!^)[aeiou]+/,     ],
```

Lucene Similarity

The Lucene Similarity score is a TFIDF based similarity metric, based on summing the similarity scores between the query terms, and the terms found in the candidate document. For more details, we include the javadoc documentation of the Luceme Similarity Class in **Table 17**

Table 17 Javadoc for Lucene Similarity Class

```
public abstract class Similarity
extends Object
implements Serializable
```

Expert: Scoring API.

Subclasses implement search scoring.

The score of query q for document d correlates to the cosine-distance or dot-product between document and query vectors in a Vector Space Model (VSM) of Information Retrieval. A document whose vector is closer to the query vector in that model is scored higher. The score is computed as follows:

$$\text{score}(q,d) = \text{coord}(q,d) \cdot \text{queryNorm}(q) \cdot \sum_{t \text{ in } q} \left(\text{tf}(t \text{ in } d) \cdot \text{idf}(t)^2 \cdot t.\text{getBoost}() \cdot \text{norm}(t,d) \right)$$

where

1. **tf(t in d)** correlates to the term's *frequency*, defined as the number of times term *t* appears in the currently scored document *d*. Documents that have more occurrences of a given term receive a higher score. The default computation for *tf(t in d)* in DefaultSimilarity is:

$$\text{tf}(t \text{ in } d) = \text{frequency}^{½}$$

2. **idf(t)** stands for Inverse Document Frequency. This value correlates to the inverse of *docFreq* (the number of documents in which the term *t* appears). This means rarer terms give higher contribution to the total score. The default computation for *idf(t)* in DefaultSimilarity is:

$$\text{idf(t)} = 1 + \log\left(\frac{\text{numDocs}}{\text{docFreq+1}}\right)$$

3. **coord(q,d)** is a score factor based on how many of the query terms are found in the specified document. Typically, a document that contains more of the query's terms will receive a higher score than another document with fewer query terms. This is a search time factor computed in coord(q,d) by the Similarity in effect at search time.

4. **queryNorm(q)** is a normalizing factor used to make scores between queries comparable. This factor does not affect document ranking (since all ranked documents are multiplied by the same factor), but rather just attempts to make scores from different queries (or even different indexes) comparable. This is a search time factor computed by the Similarity in effect at search time. The default computation in DefaultSimilarity is:

$$\text{queryNorm(q)} = \text{queryNorm(sumOfSquaredWeights)} = \frac{1}{\text{sumOfSquaredWeights}^{½}}$$

   The sum of squared weights (of the query terms) is computed by the query Weight object. For example, a boolean query computes this value as:

$$\text{sumOfSquaredWeights} = \text{q.getBoost()}^{2} \cdot \sum_{\text{t in q}} (\text{idf(t)} \cdot \text{t.getBoost()})^{2}$$

5. **t.getBoost()** is a search time boost of term *t* in the query *q* as specified in the query text (see query syntax), or as set by application calls to setBoost(). Notice that there is really no direct API for accessing a boost of one term in a multi term query, but rather multi terms are represented in a query as multi TermQuery objects, and so the boost of a term in the query is accessible by calling the sub-query getBoost().

6. **norm(t,d)** encapsulates a few (indexing time) boost and length factors:
   - **Document boost** - set by calling doc.setBoost() before adding the document to the index.
   - **Field boost** - set by calling field.setBoost() before adding the field to a document.
   - **lengthNorm**(field) - computed when the document is added to the index

in accordance with the number of tokens of this field in the document, so that shorter fields contribute more to the score. LengthNorm is computed by the Similarity class in effect at indexing.

When a document is added to the index, all the above factors are multiplied. If the document has multiple fields with the same name, all their boosts are multiplied together:

$$\text{norm}(t,d) = \text{doc.getBoost()} \cdot \text{lengthNorm(field)} \cdot \prod_{\text{field } f \text{ in } d \text{ named as } t} \text{f.getBoost()}$$

However the resulted *norm* value is encoded as a single byte before being stored. At search time, the norm byte value is read from the index directory and decoded back to a float *norm* value. This encoding/decoding, while reducing index size, comes with the price of precision loss - it is not guaranteed that decode(encode(x)) = x. For instance, decode(encode(0.89)) = 0.75. Also notice that search time is too late to modify this *norm* part of scoring, e.g. by using a different Similarity for search.

Information Gain

The information gain metric measures the total entropy for an attribute, if for each of the attribute values a unique classification can be made for the result attribute.

$$InfoGain(Class, Attribute) = H(Class) - H(Class \mid Attribute)$$

The higher the information gain of an attribute, the more valuable for the purposes of feature selection for classification.

TFIDF

The TFIDF(Term Frequency Inverse Document Frequency) metric (Salton, 1991) is a common Information Retrieval weight for measuring how important a token is to a document, given a corpus of documents. The importance increases if the word has a higher frequency in the document, but it decreases if the same word appears in many documents in the corpus.

$$tfidf_{\{i,j\}} = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{|D|}{\left|\{d_j : t_i \in d_j\right|}$$

Where

$n_{i,j}$: the number of occurrences of the considered term in document d_j

$|D|$: total number of documents in the corpus

$\left|\{d_j : t_i \in d_j\right|$: number of documents where the term t_i appears

BIBLIOGRAPHY

Allenby, G. M., & Rossi, P. E. (1999). Marketing Models of Consumer Heterogeneity. *Journal of Econometrics* , 57 - 78.

Allenby, G. M., Arora, N., & Ginter, J. L. (1998). On the Heterogeneity of Demand. *Journal of Marketing Research* , 384-389.

Alon, N., Ben-David, S., Cesa-Bianchi, N., & Haussler, D. (1993). Scale-sensitive dimensions, uniform convergence, and learnability. *Symposium on Foundations of Computer Science.*

Andrews, R., Ansari, A., & Currim, I. (2002). Hierarchical Bayes versus finite mixture conjoint analysis models: a comparison of fit, prediction, and partworth recovery. *Journal of Marketing Research* , 87-98.

Arora, N., & Huber, J. (2001). Improving parameter estimates and model prediction by aggregate customization in choice experiments. *Journal of Consumer Research* .

Arora, N., Allenby, G., & Ginter, J. (1998). A Hierarchical Bayes Model of Primary and Secondary Demand. *Marketing Science* , 29-44.

Baeza-Yates, R., & Ribeiro-Net, B. (1999). *Modern Information Retreival.* Harlow, UK: Addison-Wesley-Longman.

Bakker, B., & Heskes, T. (2003). Task clustering and gating for Bayesian multi–task learning. *Journal of Machine Learning Research* , 83–99.

Baxter, J. (2000). A model for inductive bias learning. *Journal of Artificial Intelligence Research* , 149–198.

Baxter, J. (1997). A Bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning* , 7–39.

Belkin, M., & Niyogi, P. (2004). Semi-supervised Learning on Riemannian Manifolds. *Machine Learning , 56* (Special Issue on Clustering), 209-239.

Ben-Akiva, M., & Boccara, B. (1995). Discrete Choice Models with Latent Choice Sets. *International Journal of Research in Marketing , 12*, 9-24.

Ben-Akiva, M., & Lerman, S. R. (1985). *Discrete Choice Analysis: Theory and Application to Travel Demand.* Cambridge, MA: MIT Press.

Ben-Akiva, M., McFadden, D., Abe, M., Boeckenholt, U., Bolduc, D., Gopinath, D., et al. (1997). Modeling Methods for Discrete Choice Analysis. *Marketing Letters* , 273-286.

Ben-David, S., & Schuller, R. (2003). Exploiting task relatedness for multiple task learning. *Proceedings of Computational Learning Theory (COLT).*

Ben-David, S., Gehrke, J., & Schuller, R. (2002). A theoretical framework for learning from a pool of disparate data sources. *Proceedings of Knowledge Discovery and Datamining (KDD).*

Bennett, K., & Bredensteiner, E. (2000). Duality and Geometry in SVM Classifiers. In P. Langley (Ed.), *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 57-64). San Francisco: Morgan Kaufmann.

Bennett, K., Momma, M., & Embrechts, M. (2002). MARK: A Boosting Algorithm for Heterogeneous Kernel Models. *Proceedings of SIGKDD International Conference on Knowledge Discovery and Data Mining.*

Bertsimas, D., & Tsitsikilis, J. (1997). *Introduction to linear optimization.* Belmont, MA: Athena Scientific.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *In Annual Conference on Computational Learning Theory (COLT-98).*

Bolduc, D., & Ben-Akiva, M. (1991). A Multinomial Probit Formulation for Large Choice Sets. *Proceedings of the sixth IATBR Conference.* Quebec.

Brill, E. (1992). A simple rule-based part of speech tagger. *Proceedings of the Third Annual Conference on Applied Natural Language Processing, ACL.*

Carmone, F., & Jain, A. (1978). Robustness of Conjoint Analysis: Some Monte Carlo Results. *ournal of Marketing Research , 15,* 300-303.

Carroll, D., & Green, P. (1995). Psychometric Methods in Marketing Research: Part I, Conjoint Analysis. *Journal of Marketing Research , 32,* 385-391.

Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D. P., Schapire, R. E., & Warmuth, M. K. (1997). How to use expert advice. *J. ACM , 44* (3), 427–485.

Chapelle, O., & Harchaoui, Z. (2005). A machine learning approach to conjoint analysis. *Neural Information Processing Systems* , 257–264.

Cohn, D., Caruana, R., & McCallum, A. (2003). *Semi-supervised clustering with user feedback.* Cornell University.

Collins, M. (2002). Ranking Algorithms for Named-Entity Extraction: Boosting and the Voted Perceptron. *ACL.*

Cooley, R., Srivastava, J., & Mobasher, B. (1997). Web Mining: Information and Pattern Discovery on the World Wide Web. *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97).*

Cortes, C., & Vapnik, V. (1995). Support Vector Networks. *Machine Learning , 20,* 1-25.

Cui, D., & Curry, D. (2003). Predicting Consumer Choice Using Support Vector Machines with Benchmark Comparisons to Multinomial Logit. *Marketing Science* .

DeSarbo, W., & Ansari, A. (1997). Representing Heterogeneity in Consumer Response Models. *Marketing Letters , 8* (3), 335-348.

Devroye, L., Gyorfi, L., & Lugosi, G. (1996). A Probabilistic Theory of Pattern Recognition. *Applications of mathematics , 31.*

Evgeniou, T., & Pontil, M. (2004). Regularized Multitask Learning. *KDD'04.* Seattle.

Evgeniou, T., Micchelli, C. A., & Pontil, M. (2005). Learning Multiple Tasks with Kernel Methods. *Journal of Machine Learning Research* , 615–637.

Evgeniou, T., Pontil, M., & Poggio, T. (2000). Regularization Networks and Support Vector Machines. *Advances in Computational Mathematics , 13*, 1-50.

Evgeniou, T., Pontil, M., & Poggio, T. (2000). Statistical Learning Theory: A Primer. *International Journal of Computer Vision , 38* (1), 9-13.

Evgeniou, T., Pontil, M., Poggio, T., & Papageorgiou, C. (2003). Image Representations and Feature Selection for Multimedia Database Search. *IEEE Transactions on Knowledge and Data Engineering , 15*, 911-920.

Evgeniou, T., Toubia, O., & Pontil, M. (2007). A Convex Optimization Approach to Modeling Consumer Heterogeneity in Conjoint Estimation. *Marketing Science* , 805-818.

Freund, Y., & Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to Boosting. *Journal of Computer and System Sciences , 55* (1), 119-139.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *International Conference on Machine Learning*, (pp. 148–156).

Friedman, J., Trevor Hastie, & Tibshirani, R. (1998). *Additive logistic regression: a statistical view of boosting.* Department of Statistics, Stanford University.

Gammerman, A., Vapnik, V., & Vovk, V. (1998). Learning by transduction. *Conference on Uncertainty in Artificial Intelligence*, (pp. 148-156).

Girosi, F. (2003). *Demographic Forecasting, PhD Thesis.* Harvard University.

Girosi, F., Jones, M., & Poggio, T. (1995). Regularization theory and neural networks architectures. *Neural Computation , 7*, 219-269.

Green, P., & Srinivasan, V. (1978). Conjoint Analysis in Consumer Research: Issues and Outlook. *Consumer Research* , 103-123.

Green, P., & Srinivasan, V. (1990). Conjoint Analysis in Marketing: New Developments with Implications for Research and Practice. *Journal of Marketing* , 3-19.

Herbrich, R., Graepel, T., & Obermayer, K. (1999). Large Margin Rank Boundaries for Ordinal Regression. *Advances in Large Margin Classifiers* , 29-53.

Huang, J., Kumar, S. R., & Zabih., R. (1998). An automatic hierarchical image classification scheme. *ACM Multimedia* , 219–228.

Imai, T., Schwartz, R., Kubala, F., & Nguyen, L. (1997). Improved topic discrimination of broadcast news using a model of multiple simultaneous topics. *Proc. ICASSP97,* (pp. 727–730).

Jaro, M. A. (1989). Advances in record linking methodology as applied to the 1985 census of Tampa Florida. *Journal of the American Statistical Society* , 414–20.

Jedidi, K., Jagpal, S., & Manchanda, P. (2003). Measuring Heterogeneous Reservation Prices for Product Bundles. *Marketing Science* , 107-130.

Joachims, T. (2002). Optimizing Search Engines Using Clickthrough Data. *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD).* ACM.

Kearns, M., & Schapire, R. (1994). Efficient distribution-free learning of probabilistic concepts. *Journal of Computer and Systems Sciences* , 464-497.

Kohavi, R. (2001). Mining E-Commerce Data: The Good, the Bad, and the Ugl. *Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* (pp. 8-13.).

Kuhfeld, W. F., Tobias, R. D., & Garratt, M. (1994). Efficient Experimental Design with Marketing Research Applications. *Journal of Marketing Research* , 545-557.

Kupiec, J., Pedersen, J. O., & Chen, F. (1995). A trainable document summarizer. *Proceedings of the SIGIR* , 68–73.

Lenk, P. J., DeSarbo, W. S., Green, P. E., & Young, M. R. (1996). Hierarchical Bayes Conjoint Analysis: Recovery of Partworth Heterogeneity from Reduced Experimental Designs. *Marketing Science* , 173-191.

Levenshtein distance. (n.d.). Retrieved 01 31, 2009, from Wikipedia: http://en.wikipedia.org/wiki/Levenshtein_distance

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady* , 707–710.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady 10* , 707–710.

Louviere, J. J., Hensher, D. A., & Swait, J. D. (2000). *Stated Choice Methods: Analysis and Applications.* New York: Cambridge University Press.

Manski, C. F. (1977). The Structure of Random Utility Models. *Theory and Decision* , 229-254.

McCallum, A. (1999). Multi-label text classification with a mixture model trained by EM. *AAAI 99 Workshop on Text Learning* .

McCallum., A. K. (1996, 01 08). *Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering*. Retrieved 2009, from http://www.cs.cmu.edu/~mccallum/bow/

McFadden, D. (1974). Conditional Logit Analysis of Qualitative Choice Behavior. *Frontiers in Econometrics* , 105-142.

McFadden, D. (1986). The Choice Theory Approach to Marketing Research. *Marketing Science* , 275-297.

Minka, T. P. (2001). *Algorithms for maximum-likelihood logistic regression.* CMU Statistics Tech Report 758.

Mitchell, T. (1997). *Machine Learning.* McGraw-Hil.

Niyogi, P., Poggio, T., & Girosi, F. (1998). Incorporating Prior Information in Machine Learning by Creating Virtual Examples. *IEEE Proceedings on Intelligent Signal Processing.*

Oppewal, H., Louviere, J., & Timmermans, H. (1994). Modeling Hierarchical Conjoint Processes with Integrated Choice Experiments. *Journal of Marketing Research* , 92-105.

OTS: Open Text Summarizer. (2003). Retrieved from http://libopts.sourceforge.net

Pauwels, K. H. (2004). How Dynamic Consumer Response, Dynamic Competitor Response and Expanded Company Action Shape Long-term Marketing Effectiveness. *Marketing Science* , 596-610.

Philips, L. (1990). Hanging on the Metaphone. *Computer Language , 7* (12).

Pontil, M., & Verri, A. (1998). Properties of Support Vector Machines. *Neural Computation* , 955-974.

Quinlan, J. R. (1993). *C4.5: programs for machine learning.* New York: Morgan Kaufmann Publishers, Inc.

Rifkin, R. (2002). *Everything Old Is New Again: A Fresh Look at Historical Approaches in Machine Learning, PhD Thesis.* MIT.

Rogati, M., & Yang, Y. (2002). High-Performing Feature Selection for Text Classification. *CIKM'02, ACM,.*

Rui, Y., Huang, T. S., & Chang., S.-F. (1997). Image retrieval: Past, present, and future. *International Symposium on Multimedia Information Processing.*

Russell, R. C. (1918). *Patent No. 1,261,167.* US.

Salton, G. (1991). Developments in automatic text retrieval. *Science* , 974-979.

Salton, G., & McGill., M. J. (1983). *The SMART and SIRE experimental retrieval systems.* New York: McGraw-Hill.

Salton., G. (1968). *Automatic Information Organization and Retrieval.* New York: McGraw-Hill.

Sawtooth Software, I. (2009, 01 08). *HB-Reg: Hierarchical Bayes Regression.* Retrieved from Sawtooth Software, Inc.: http://www.sawtoothsoftware.com/download/info/hbreg.php

Schapire, R. E. (1999). A brief introduction to boosting. *IJCAI,* (pp. 1401–1406).

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning* , 197–227.

Scholkopf, B., Burges, C., & Vapnik, V. (1996). Incorporating Invariances in Support Vector Learning Machines. *Artificial Neural Networks -- ICANN'96.* Berlin: Springer Lecture Notes in Computer Science.

Schultz, M., & Joachims, T. (2003). Learning a Distance Metric from Relative Comparisons. *Proceedings of the Conference on Advance in Neural Information Processing Systems (NIPS).*

Segal, M. N. (1982). Reliability of Conjoint Analysis: Contrasting Data Collection Procedures. *Journal of Marketing Research* , 139-143.

Spearman, C. (1904). The Proof and Measurement of Association between Two Things. *The American Journal of Psychology* (Special Centennial Issue (1987)), 441-471.

Srinivasan, V. (1998). A Strict Paired Comparison Linear Programming Approach to Nonmetric Conjoint Analysis. (J. E. Aronson, & S. Zionts, Eds.) *Operations Research: Methods, Models and Applications* , 97-111.

Srinivasan, V., & Shocker, A. D. (1973). Linear Programming Techniques for Multidimensional Analysis of Preferences. *Psychometrica* , 337-369.

Srinivasan, V., Arun Jain, & Malhotra, N. (1983). Improving the Predictive Power of Conjoint Analysis by Constrained Parameter Estimation. *Journal of Marketing Research* , 433-438.

Stone, C. (1985). Additive Regression and Other Nonparametric Models. *Annals of Statistics* , 689-705.

Tikhonov, A. N., & Arsenin, V. Y. (1977). Solutions of Ill-posed Problems.

Tong, S., & Koller, D. (2000). Support vector machine active learning with applications to text classification. *Proceedings of the Seventeenth International Conference on Machine Learning.*

Toubia, O., Hauser, J. R., & Simester, D. I. (2004). Polyhedral methods for adaptive choice-based conjoint analysis. *Journal of Marketing Research* .

Toubia, O., Simester, D. I., Hauser, J. R., & Dahan, E. (2003). Fast Polyhedral Adaptive Conjoint Estimation. *Marketing Science* , 273-303.

Tversky, A., & Kahneman, D. (1974). Judgment Under Uncertainty: Heuristics and Biases. *Science* , 1124-1131.

Ulrich, K. T., & Eppinger, S. D. (2000). *Product Design and Development.* New York: McGraw-Hill, Inc.

Vapnik, V. (1998). *Statistical Learning Theory.* New York: Wiley.

Vapnik, V., & Chervonenkis, A. (1971). On the Uniform Convergence of Relative Frequences of events to their probabilities. *Th. Prob. and its Applications* , 264-280.

Wahba, G. (1990). Splines Models for Observational Data. *Series in Applied Mathematics, Vol. 59, SIAM.* .

Winkler, W. E. (2006). *Overview of Record Linkage and Current Research Directions.* US Census Bureu, Research Report Series, RRS.

Wittink, D. R., & Cattin, P. (1989). Commercial Use of Conjoint Analysis: An Update. *Journal of Marketing* , 91-96.

Yahoo! (n.d.). *Yahoo! News.* Retrieved from Yahoo!: http://news.yahoo.com

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval* , 69–90.

Zhu, J., & Hastie, T. (2001). Kernel Logistic Regression and the Import Vector Machine. *Proceedings of NIPS2001.* Vancouver.