SCOAL: A Framework for Simultaneous Co-Clustering and Learning from Complex Data

MEGHANA DEODHAR and JOYDEEP GHOSH
University of Texas at Austin

For difficult classification or regression problems, practitioners often segment the data into relatively homogeneous groups and then build a predictive model for each group. This two-step procedure usually results in simpler, more interpretable and actionable models without any loss in accuracy. In this work, we consider problems such as predicting customer behavior across products, where the independent variables can be naturally partitioned into two sets, that is, the data is dyadic in nature. A pivoting operation now results in the dependent variable showing up as entries in a "customer by product" data matrix. We present the Simultaneous CO-clustering And Learning (SCOAL) framework, based on the key idea of interleaving co-clustering and construction of prediction models to iteratively improve both cluster assignment and fit of the models. This algorithm provably converges to a local minimum of a suitable cost function. The framework not only generalizes co-clustering and collaborative filtering to model-based co-clustering, but can also be viewed as simultaneous co-segmentation and classification or regression, which is typically better than independently clustering the data first and then building models. Moreover, it applies to a wide range of bi-modal or multimodal data, and can be easily specialized to address classification and regression problems. We demonstrate the effectiveness of our approach on both these problems through experimentation on a variety of datasets.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining*

General Terms: Algorithms

Additional Key Words and Phrases: Predictive modeling, co-clustering, classification, regression, dyadic data, multimodal data

ACM Reference Format:

Deodhar, M. and Ghosh, J. 2010. SCOAL: A framework for simultaneous co-clustering and learning from complex data. ACM Trans. Knowl. Discov. Data. 4, 3, Article 11 (September 2010), 31 pages.
DOI = 10.1145/1839490.1839492 http://doi.acm.org/10.1145/1839490.1839492

This research was supported by NSF grants IIS-0325116, IIS-0307792 and IIS-0713142.

Authors' address: M. Deodhar and J. Ghosh, Department of Electrical and Computer Engineering, The University of Texas at Austin, 1 University Station, Austin, TX 78712; email: Meghana@mail.utexas.edu; ghosh@ece.utexas.edu.

DOI 10.1145/1839490.1839492 http://doi.acm.org/10.1145/1839490.1839492

1. INTRODUCTION

While it is common practice to develop a single learned model (e.g., a classification or regression model, or a single ensemble of multiple base-learners) to characterize a given dataset, for many problems it is practically advantageous to partition the population into multiple, relatively homogeneous segments and then develop separate models for each segment [Baumann and Germond 1993; Djukanovic et al. 1993; Oh and Han 2001; Lokmic and Smith 2000]. For example, an e-tailer attracts different types of browsers, from casual shoppers to bulk purchasers, and may want to model their purchasing inclinations separately. Similarly while forecasting electric load usage, it is advisable to build separate predictive models for weekdays, weekends and holidays. Advantages of such divide-and-conquer approaches include not only improved accuracy and reliability in general, but also improved interpretability as well, since the component models are often far simpler [Sharkey 1996]. Typically the partitioning is done a priori based on domain knowledge or a separate segmentation routine [Baumann and Germond 1993; Djukanovic et al. 1993], before the predictive models are learnt.

This article is concerned with situations where the independent variables can be naturally partitioned into two (or more) groups that are associated with their corresponding modes. We then simultaneously cluster along each mode, as well as fit a learned model to each co-cluster. The approach can *alternatively be viewed as a model-based generalization of biclustering or co-clustering*, which is a technique that simultaneously clusters along multiple axes and has been successfully applied in several domains like text clustering and microarray data analysis [Cho et al. 2004; Cheng and Church 2000; Dhillon et al. 2003]. Co-clustering is traditionally applied to a matrix of data values, where the rows are data points and the columns are features, for example, in microarray data the rows are genes and columns are experiments, in recommender systems the rows are customers and the columns are products. Co-clustering exploits the duality between the two axes to improve on single-sided clustering. In our extension, along with the data matrix, a set of variables is associated with the rows, another with the columns, and a third set with the combination of rows and columns. Any of these sets of variables can be empty. For each co-cluster, a model is learned to predict a matrix cell value given the corresponding row and column attributes. Generalization refers to predicting the missing values in the data matrix, as well as values when new rows/columns are added.

Example. To concretize the preceding discussion, consider the problem of predicting customer purchase decisions (recommending products to customers). The dataset in this case is a matrix of customers by products, where the cell values are class labels representing whether a customer buys a certain product or not. This matrix will have missing values where the corresponding customer-product choice is unknown. Customers and products are described by attributes (also referred to as covariates [Agarwal and Merugu 2007]) that represent the independent variables. The covariates include customer demographics and product attributes such as the price, market share, quality, etc. Additionally, features associated with a customer-product pair, for example, whether the customer received a special discount on the product, could also be available. The problem is to predict the choices for the missing customer-product combinations, as well as behavior of new customers, or choices made for new products. Note that collaborative filtering approaches for this problem will make use only of the matrix entries and ignore customer/product attributes [Herlocker et al. 1999; George and Merugu 2005]. On the other extreme, a typical classification model will form a map between the features associated with a given customer-product pair and the corresponding matrix entry, but will not consider nearby customers or products in this process. For the classifier, the dependent variable values are nothing but the matrix entries, and the independent variables are grouped into variables associated with the rows, columns or both. For a diverse population of customers and a wide range of different products, it is unlikely that all the customer-product preferences can be well explained by a single model. Instead, it may be more feasible to learn models that closely represent the preferences of only a subset of the customers for a subset of the products.

A co-clustering approach will simultaneously cluster the customers and products based on the matrix entries. It will then use the entries of the corresponding co-cluster to predict a missing value. If done properly, this gives better results than standard recommender systems [George and Merugu 2005], however this approach still ignores the customer and product attributes, that is, the prediction is solely based on the value of the dependent variable in a suitably identified neighborhood. Our approach exploits both neighborhood information as well as the available customer/product attributes. The idea is to co-cluster the entire data matrix into blocks of customers and products such that each block can be well characterized by a single predictive model. Note that the similarity between two data entries is now determined not by the similarity between the values themselves, but rather between the corresponding predictive models. Moreover, our model based co-clustering-cum-learning algorithm achieves this by interleaving clustering and construction of classification models to iteratively improve both cluster assignment and fit of the models. This simultaneous approach is better than independently clustering the data first and then building classification models. We also exhibit a cost function that is steadily decreased in both steps of the iterative process, till one reaches a local minimum, thereby guaranteeing convergence.

The SCOAL approach is not restricted to classification, but forms a very versatile framework for prediction problems in general. Depending on the application, SCOAL can construct local models that are generalized linear models (GLMs) or even other predictive models like neural networks (MLP, RBF, etc.). In order to present a concrete description of SCOAL, we focus on two special cases (i) classification using logistic regression, which is discussed in Section 3 and (ii) linear regression models, discussed in Section 4. Further, in Section 5 we show how these 2 instances are special cases of a GLM based soft generative model. We then illustrate how the SCOAL framework can be used with a number of regularization techniques that provide robustness and improve accuracy (Section 7). In Section 8, we develop a simple but effective model selection approach to determine an appropriate number of local models to cover the problem space. To highlight the effectiveness of our approach, we consider diverse applications involving the following classification and regression problems (i) predicting course choices made by graduate students in Section 9.2, (ii) predicting the number of items of a given product purchased by a customer, using a formidable, real dataset in Section 10.1, and (iii) predicting unknown user-movie ratings using the MovieLens dataset in Section 10.2.

In the rest of the articles, we refer to the data points as customers and axes as products, based on our motivating applications. Our approach is however not restricted to a customer-product matrix, and is applicable to any bi-modal dataset. It can also be readily extended to multi-modal data, for example, a 3-D tensor (data cube) with sets of variables associated with one or more of the axes.

Notation. Lower case letters represent scalars, for example, a, z, lowercase, boldface letters represent vectors, for example, $\mathbf{b}, \mathbf{c}, \boldsymbol{\beta}$, uppercase letters like Z, W represent matrices, calligraphic uppercase letters like $\mathcal{Z}$ represent tensors. Individual elements of a matrix, for example, Z are represented as z_{ij}, where i and j are the row and column indices respectively.

2. RELATED WORK

In this section, we present three components of related work, beginning with several "divide and conquer" approaches that have been used for solving complex classification and regression problems. This is followed by a brief introduction to co-clustering and its applicability to prediction problems. Finally, we discuss approaches based on simultaneous clustering and modeling.

2.1 Multiple Localized Models

There are several examples of the use of localized prediction models in load forecasting systems, where clustering is used to distinguish smaller, homogeneous groups of data and a prediction model is then fitted for each cluster in a two step sequential process [Baumann and Germond 1993; Djukanovic et al. 1993]. Clustering based prediction models have also been widely used in economics [Oh and Han 2001; Lokmic and Smith 2000]. Sfetsos and Siriopoulos [2004] propose an iterative algorithm to cluster time series data such that each cluster consists of data points with similar linear models. This is followed by a heuristic to identify a single cluster to be used for future predictions. This clustering algorithm is one sided and linear model based, a special case of our "two-sided" co-clustering with associated generalized linear models.

Another body of work that is related to input space partitioning and modeling is that comprised of decision tree based approaches. The well known CART system [Breiman et al. 1984] approximates a nonlinear function by local, piecewise constants. The M5' approach proposed by Wang and Witten [1997] builds on CART and the M5 method developed by Quinlan [1992] to predict a continuous response variable value by inducing model trees. The basic idea is to build a tree using a splitting criterion that minimizes the variation in the response values going down each branch. A linear predictive model is learnt at each tree node using only the attributes in the corresponding sub-tree. The tree is then pruned as long as the estimated error reduces. A smoothing procedure is used to compute the predicted value, in which the values output by linear models along the path from a leaf node to the root are suitably combined.

The mixture-of-experts framework [Jordan et al. 1991; Ramamurti and Ghosh 1998] simultaneously partitions the input space while learning models for each partition. The partitioning is soft however, that is, multiple models are involved in varying amounts for producing any particular input-output map, which makes the system less interpretable and actionable as compared to our proposed approach. Our partitioning is based on co-clustering the data into a grid of rectangular blocks and is hence more structured. Moreover our approach is able to smooth over the joint input-output space which is not achieved by a mixture-of-experts.

In the bioinformatics domain, clustering of genes is often used as a preprocessing step for the classification of experiments (samples) in microarray data analysis [Liu et al. 2005; Jornsten and Yu 2003]. A cluster is represented by the mean of the expression profiles across all its member genes, which acts as a dimensionality reduction step for the classification process. This helps to reduce gene redundancies and constructs parsimonious and more interpretable classification models. A simultaneous clustering and classification algorithm is proposed by Zhang et al. [2005], which uses a voting based classifier ensemble to improve a clustering solution. The labels assigned by an initial clustering are used to train a set of diverse classifiers. Data points that lie on cluster boundaries are relabeled by combining the classifier predictions using a majority vote. This process is iterated to refine the clustering solution.

2.2 Co-Clustering

Co-clustering (also known as biclustering) has been used in several diverse data mining applications like clustering microarray data [Cheng and Church 2000; Cho et al. 2004], text mining [Dhillon et al. 2003] and marketing applications [Wedel and Steenkamp 1991]. Most clustering or co-clustering approaches cannot handle missing data and assume a full data matrix. However the formulation by Banerjee et al. readily handles missing data [Banerjee et al. 2007], and has been shown to perform significantly better than traditional collaborative filtering techniques in a recommender system setting [George and Merugu 2005], where the data is a matrix of customer-movie ratings. The known ratings are used to simultaneously cluster customers and movies and compute summary statistics for the co-clusters, which are then used to predict unknown ratings, using an instance of the Bregman co-clustering algorithm [Banerjee et al. 2007].

2.3 Simultaneous Clustering and Prediction

The idea of simultaneous clustering and regression was introduced in the marketing literature by Wedel and Steenkamp [1981], who proposed a generalized fuzzy clusterwise regression technique to find both customer segments and market structure. Each cluster includes fractional membership from all customers and products and is hence a fuzzy co-cluster. Each cluster has a regression model that predicts the preferences as a linear combination of the product attributes. The cluster memberships and models are estimated so as to reduce the total squared error between the actual preferences and the predicted preferences. However, the hard version of this method corresponds to diagonal co-clustering [Madeira and Oliveira 2004], where only a subset of products is associated with each customer group. In contrast, this paper is concerned with partitional co-clustering, which covers the entire customer-product matrix.

The Predictive Discrete Latent Factor model approach (PDLF) [Agarwal and Merugu 2007], addresses the problem of predicting dyadic response variables (e.g., ratings in a customer-product matrix), when covariate information (e.g., customer and product attribute information) is available. The PDLF model simultaneously incorporates the effect of the covariates, through a *single* global model as well as any local structure that may be present in the data, through a block (co-cluster) specific constant. Scalable, generalized EM based algorithms are formulated to estimate the parameters of hard or soft versions of the proposed model. While this approach is motivated by a problem setting very similar to ours, the PDLF model is complementary to our proposed SCOAL approach, which constructs an independent prediction model in each co-cluster. Recent work by Agarwal and Chen [2009] generalizes the PDLF approach to a regression based latent factor modeling technique. An important feature of this technique is that it effectively utilizes the covariates to deal with the cold start problem, that is, making predictions for new "customers" and "products".

3. SIMULTANEOUS CO-CLUSTERING AND CLASSIFICATION

3.1 Problem Definition

We now describe the problem formulation for the classification setting. Let m be the total number of customers and n the total number of products. The data can be represented as an $m \times n$ matrix Z of customers and products, with cells z_{ij} representing the corresponding class labels, for example, whether customer i buys product j or not. Throughout the following discussion, we assume that we are dealing with a 2 class problem and $z_{ij} \in \{-1, +1\}$, however, the algorithm can easily be generalized to deal with multiclass settings. The problem formulation and solution for regression models is given in Section 4.

Every customer and product pair is described by a vector of covariates $\mathbf{x_{ij}}$. The covariate vector is typically composed of the customer attributes, $\mathbf{c_i}$, the product attributes, $\mathbf{p_j}$ as well as annotations associated with the customer-product pair, $\mathbf{a_{ij}}$. Note that up to two of these three vectors ($\mathbf{c_i}$,$\mathbf{p_j}$,$\mathbf{a_{ij}}$) could be empty. It is assumed that each class label z_{ij} is primarily determined by the attributes of the corresponding customer-product pair via a logistic regression model.[1] Thus, the log odds is modeled as a linear combination of the customer and product attributes given by

$$ln\frac{P(z_{ij}=1|\mathbf{x_{ij}})}{1-P(z_{ij}=1|\mathbf{x_{ij}})} = f(\mathbf{x_{ij}}),$$

where $\mathbf{x_{ij}}^T = [1, \mathbf{c_i}^T, \mathbf{p_j}^T, \mathbf{a_{ij}}^T]$ is a vector consisting of the customer-product covariates, and $f(\mathbf{x_{ij}}) = \boldsymbol{\beta}^T\mathbf{x_{ij}}$ is a linear model with parameters $\boldsymbol{\beta}^T = [\beta_0, \boldsymbol{\beta_c}^T, \boldsymbol{\beta_p}^T, \boldsymbol{\beta_a}^T]$. The similarity of the cell values is now defined based on the similarity of their underlying logistic regression models. The aim is to simultaneously cluster the rows (customers) and columns (products) into a grid of k row clusters and l column clusters,[2] such that the class labels within each co-cluster are predicted by a single, common classification model. The co-cluster assignments along with the classification models for the co-clusters can be used to predict the class labels for missing customer-product combinations.

Formally, let ρ be a mapping from the m rows to the k row clusters and γ be a mapping from the n columns to the l column clusters. A weight w_{ij} is associated with each cell z_{ij}. The weights of the known (training) matrix cell values are set to 1. The missing cell values, that are to be predicted are given a weight of 0. In general, the weight is not restricted to 0 or 1 and can take other values. This formulation allows the prediction framework to deal with data uncertainties, where less certain values can be given comparatively lower but non-negative weights. We now want to find a co-clustering defined by (ρ, γ) and associated set of classification models $\{\boldsymbol{\beta}^{gh}\}$ that minimize the following objective function:

$$\sum_{g=1}^{k}\sum_{h=1}^{l}\sum_{u:\rho(u)=g}\sum_{v:\gamma(v)=h} w_{uv}ln(1+exp(-z_{uv}\boldsymbol{\beta^{gh}}^T\mathbf{x_{uv}})), \quad (1)$$

where z_{uv} is the original value (class label) in row u, column v of the matrix, with associated weight w_{uv}. Here, $\boldsymbol{\beta^{gh}}$ denotes the vector of coefficients of the model associated with the co-cluster that the cell value z_{uv} is assigned to. Since the weights for the missing z_{uv} values are set to 0, the objective function essentially ignores them and is simply the log loss summed only over all the known elements of matrix Z. Minimizing this objective function is equivalent to maximizing the log-likelihood of the data.

3.2 Algorithm for Logistic Regression Based Classification

A co-clustering (ρ, γ), that reduces the cost function (1) can be obtained by a simple iterative algorithm. Since the objective function is the log loss summed over all the elements of the matrix, it can be expressed as a sum of row or column losses. If row u is assigned to row cluster g (i.e., $\rho(u) = g$), the row

[1] The focus of this article is on simultaneous co-clustering and classification, rather than obtaining the best possible classifier; therefore, we have chosen a standard, fairly flexible classifier rather than experiment with a multitude of classifier options.

[2] This form of co-clustering is often called partitional co-clustering [Madeira and Oliveira 2004]

error is

$$E_u(g) = \sum_{h=1}^{l} \sum_{v:\gamma(v)=h} w_{uv} ln(1 + exp(-z_{uv} \boldsymbol{\beta}^{\boldsymbol{gh}^T} \mathbf{x_{uv}})).$$

Since any missing values in the row u will have a weight 0, the error $E_u(g)$ is effectively computed only over the known values in row u. For a given column clustering and model parameter sets $\{\boldsymbol{\beta}^{\boldsymbol{gh}}\}$, the best choice of the row cluster assignment for row u is the g that minimizes this error, that is,

$$\rho^{new}(u) = \arg{}_g\min\ E_u(g).$$

Each row is hence assigned to the row cluster that minimizes the row error. A similar approach is used to (re)-assign columns to column clusters. Such row and column cluster updates hence decrease the objective function and improve the clustering solution. Note that updating column cluster assignments could cause the best row assignments to change and vice-versa. Thus optionally, the row and column cluster reassignment steps can be repeated several times and in arbitrary order until both row and column cluster memberships converge.

Given the current row and column cluster assignments, the co-cluster models need to be updated, that is, the co-efficient vector $\boldsymbol{\beta}$ has to be updated for each co-cluster. To update the model for a row cluster g of size r and column cluster h of size c, train a logistic regression model with the $r \times c$ values within the co-cluster, weighted by their corresponding weight values. The missing values present in the co-cluster have weights of 0 and are essentially ignored. The logistic regression model is hence trained using only the known training samples $(\mathbf{x_{uv}}, z_{uv})$. In the more general case of arbitrary valued weights, this step involves updating a weighted logistic regression model [Lee and Liu 2003] rather than a simple logistic regression model. The output will be an updated vector $\boldsymbol{\beta}^{\boldsymbol{gh}}$ of coefficients that minimizes the model log loss given by

$$L = \sum_{u=1}^{r} \sum_{v=1}^{c} w_{uv} ln(1 + exp(-z_{uv} \boldsymbol{\beta}^{\boldsymbol{gh}^T} \mathbf{x_{uv}})).$$

The model update step is hence guaranteed to decrease the objective function (1).

The resulting algorithm is a simple iterative algorithm described in Figure 1. Step 1 minimizes the objective function due to the property of logistic regression, Steps 2(a) and 2(b) directly minimize the objective function. The objective function hence decreases at every iteration. Since this function is bounded from below by zero, the algorithm is guaranteed to converge to a local minimum.

Predicting missing class labels. After the co-cluster assignments and the co-clusterwise classification models are obtained by the algorithm, the missing class labels can be predicted easily. Let z_{uv} be a missing cell value that has been assigned to row cluster g and column cluster h. $\mathbf{x_{uv}}$ is the vector of attributes of row u and column v and $\boldsymbol{\beta}^{\boldsymbol{gh}}$ represents the model parameters of the logistic regression model of the assigned co-cluster. The logistic regression

```
Algorithm: SCOAL
Input: $Z_{m \times n}$, $W_{m \times n}$, covariates
Output: Co-clustering $(\rho, \gamma)$ and co-cluster models $\boldsymbol{\beta}$s
Begin with a random co-clustering $(\rho, \gamma)$
Repeat

    Step 1
    Update co-cluster models
    for $g = 1$ to $k$ do
        for $h = 1$ to $l$ do
            Train a logistic regression model with training samples $(\mathbf{x_{uv}}, z_{uv})$ in
            co-cluster $(g, h)$, with weights $w_{uv}$, to obtain an updated $\boldsymbol{\beta^{gh}}$.
        end for
    end for

    Step 2(a)
    Update $\rho$ - assign each row to the row cluster that minimizes the row error
    for $u = 1$ to $m$ do
        $\rho(u) = \arg_g \min \ \sum_{h=1}^{l} \sum_{v:\gamma(v)=h} w_{uv} ln(1 + exp(-z_{uv} \boldsymbol{\beta^{gh}}^T \mathbf{x_{uv}}))$
    end for

    Step 2(b)
    Update $\gamma$ - assign each col. to the col. cluster that minimizes the col. error
    for $v = 1$ to $n$ do
        $\gamma(v) = \arg_h \min \ \sum_{g=1}^{k} \sum_{u:\rho(u)=g} w_{uv} ln(1 + exp(-z_{uv} \boldsymbol{\beta^{gh}}^T \mathbf{x_{uv}}))$
    end for
    Optional: repeat steps 2(a) and 2(b) until convergence

Until convergence
Return $(\rho, \gamma)$ and $\boldsymbol{\beta}$s
```

Fig. 1. Pseudocode for simultaneous co-clustering and classification.

model is used to obtain the probability of z_{uv} of belonging to the positive class as follows

$$P(z_{uv} = 1) = \frac{1}{1 + e^{-\boldsymbol{\beta^{gh}}^T \mathbf{x_{uv}}}}$$

A suitable threshold t is used to convert the probabilities into class labels, that is, $z_{uv} = 1$ if $P(z_{uv} = 1) > t, z_{uv} = -1$ otherwise.

4. SIMULTANEOUS CO-CLUSTERING AND REGRESSION

In the regression setting, Z is an $m \times n$ matrix of "customers" and "products", with cells representing the corresponding customer-product preference values, ratings or choice probabilities. Here we assume the generative model to be a linear model, where the preference value $z_{ij} \in \mathbb{R}$ is modeled as a linear combination of the corresponding covariates. The preference value is estimated as $\hat{z}_{ij} = \beta_0 + \boldsymbol{\beta_c}^T \mathbf{c_i} + \boldsymbol{\beta_p}^T \mathbf{p_j} + \boldsymbol{\beta_a}^T \mathbf{a_{ij}}$. Similar to the problem definition in Section 3.1, the aim is to simultaneously cluster the customers and products into a grid of k row clusters and l column clusters, such that preference values within each co-cluster have similar linear models and can be represented by a single common model. We want to find a co-clustering defined by (ρ, γ) and the associated $k \times l$ regression models that minimize the following objective function

$$\sum_{g=1}^{k}\sum_{h=1}^{l}\sum_{u:\rho(u)=g}\sum_{v:\gamma(v)=h} w_{uv}(z_{uv} - \hat{z}_{uv})^2, \tag{2}$$

where

$$\hat{z}_{uv} = \boldsymbol{\beta}^{\boldsymbol{gh}^T}\mathbf{x_{uv}}. \tag{3}$$

The only difference here as compared to the classification case is the loss function, which is now squared loss rather than log loss. A co-clustering (ρ, γ), that minimizes the objective function can be obtained by an algorithm similar to the one described in Section 3. The cluster reassignment steps assign each row or column to the row or column cluster that minimizes the row or column error.

The model for row cluster g of size r and column cluster h of size c is updated by finding the $\boldsymbol{\beta}$ that minimizes

$$\sum_{u=1}^{r}\sum_{v=1}^{c} w_{uv}(z_{uv} - \boldsymbol{\beta}^{\boldsymbol{gh}^T}\mathbf{x_{uv}})^2.$$

In case of 0/1 weights, this is equivalent to ignoring missing values and updating the $\boldsymbol{\beta}$ for each co-cluster by least squares regression using only the non-missing (training) values within the co-cluster. In case of a general set of weights, the $\boldsymbol{\beta}$ is a solution to a weighted least squares problem. Close to convergence, since the co-cluster memberships change only by a few matrix entries in each iteration, it is more efficient to incrementally update the regression models rather than retraining from scratch. This can be achieved by updating the QR factorization of the coefficient matrix to reflect the addition and deletion of data points [Gill et al. 1981]. Least squares regression finds a solution for $\boldsymbol{\beta}$ that minimizes the sum of the squared errors between the original values and the predicted values. The model update step is hence guaranteed to decrease the objective function.

After the algorithm converges, the co-cluster assignments and co-clusterwise regression models can be used to predict unknown customer-product preference values. A missing value z_{uv} is predicted as $\hat{z}_{uv} = \boldsymbol{\beta}^{\boldsymbol{gh}^T}\mathbf{x_{uv}}$.

5. GENERATIVE MODEL FOR SOFT SCOAL

Logistic and linear regression models with which we developed the SCOAL algorithm in Sections 3 and 4 are special cases of a large class of models known as generalized linear models (GLMs) [McCullagh and Nelder 1983]. We begin by stating the relevant properties of GLMs. If the response variable y follows a GLM, then a function g of the mean response is modeled as an unknown linear function $\boldsymbol{\beta}^T\mathbf{x}$ of the independent variables $\mathbf{x}$. g is referred to as the link function. For example, in linear least-squares regression the link function is the identity, while in logistic regression, g is the logit function $(g(y) =$

$log(\frac{y}{1-y})$). Another important characteristic of GLMs is that the distribution of the response variable conditioned on $\mathbf{x}$ belongs to a member of the exponential family. A single-parameter exponential family [Banerjee et al. 2005] is a set of distributions whose probability density function can be expressed as

$$f(x;\theta) = h(x)\exp(\theta t(x) - \psi(\theta)),$$

where θ is known as the natural parameter, $\psi(\theta)$ is the cumulant generating function and $h(x)$ and $t(x)$ are functions of x. The Gaussian, Poisson and Bernoulli distributions are examples of exponential families. Under linear least-squares regression and logistic regression models, the response variable belongs to the Gaussian and Bernoulli exponential families respectively.

The formulation of the SCOAL meta-algorithm with GLMs for co-cluster models is based on an intuitive generative model, consisting of a mixture of $k \times l$ exponential family distributions, corresponding to the $k \times l$ co-clusters. Each matrix entry z_{ij}, given the covariates $\mathbf{x_{ij}}$ is assumed to be generated from this mixture model as follows

$$P(z_{ij}|\mathbf{x_{ij}}) = \sum_{I=1}^{k}\sum_{J=1}^{l}\alpha_{IJ} f_{\psi}\left(z_{ij};\boldsymbol{\beta^{I,J}}^{T}\mathbf{x_{ij}}\right),[i]_1^m, [j]_1^n, \tag{4}$$

where α_{IJ} denotes the mixture component priors and f_ψ is an exponential family distribution with cumulant $\psi(.)$. The natural parameter $\theta_{i,j,I,J}$ of the exponential family is the linear function $\boldsymbol{\beta^{I,J}}^{T}\mathbf{x_{ij}}$, where $\boldsymbol{\beta^{I,J}}$ denotes the regression coefficients of component (I, J). Response values z_{ij} hence belong fractionally to the components (I, J), with the mean response for each component modeled as a linear function of the corresponding covariates. Note that the co-cluster assignments here are *soft*, in contrast with the hard assignments in the formulation in Section 3, where each row/column belonged to exactly 1 row/column cluster.

By assuming that the matrix elements are generated independent and identically distributed with weights w_{ij}, the incomplete data log-likelihood is given by

$$L(\boldsymbol{\beta}, \boldsymbol{\alpha}|Z) = \sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij} log P(z_{ij}), \tag{5}$$

where $\boldsymbol{\beta}, \boldsymbol{\alpha}$ collectively denote all the model coefficients and priors. It is however not possible to directly maximize this data log-likelihood and we hence associate latent variables $\rho(i)$ and $\gamma(j)$, denoting row and column cluster membership, with each element z_{ij}. $\rho(i)$ and $\gamma(j)$ take values from 1 to k and 1 to l, respectively. We first construct the free energy function [Neal and Hinton 1998] as a sum of the expected complete data log-likelihood and the entropy of the latent variables with respect to a distribution $\tilde{p}(\rho(i), \gamma(j))$, that is,

$$F(\tilde{p}, \boldsymbol{\beta}, \boldsymbol{\alpha}) = \sum_{ij} w_{ij} E_{\tilde{p}_{ij}}[log P(z_{ij}, \rho(i), \gamma(j))] + \sum_{ij} w_{ij} H(\tilde{p}_{ij}), \text{ where} \tag{6}$$

$$E_{\tilde{p}_{ij}}[log P(z_{ij}, \rho(i), \gamma(j))] = \sum_{IJ} \tilde{p}_{ij}(I, J) log(\alpha_{IJ} f_{\psi}(z_{ij}|\theta_{i,j,I,J})),$$

```
Algorithm: Soft SCOAL
Input: $Z_{m\times n}$, $W_{m\times n}$, covariates, $k, l$, exponential family with cumulant $\psi$
Output: Sets of regression coefficients $\boldsymbol{\beta}^{I,J}$, priors $\boldsymbol{\alpha}$ and co-cluster assignments $\tilde{p}$

Begin with arbitrarily initialized assignments $\tilde{p}$
Repeat
        E-step
        Update Row Cluster Assignments:
        $\tilde{p}_i(I) = c_i(\prod_{j,J}(\alpha_{IJ} f_\psi(z_{ij}; \boldsymbol{\beta^{I,J}}^T \mathbf{x_{ij}}))^{w_{ij}\tilde{p}_j(J)})^{\frac{1}{w_i}}, \quad \forall [i]_1^m, [I]_1^k,$
        where $c_i$ is a normalizing factor s.t. $\sum_{I=1}^k \tilde{p}_i(I) = 1$ and $w_i = \sum_j w_{ij}$.

        Update Column Cluster Assignments:
        $\tilde{p}_j(J) = c_j(\prod_{i,I}(\alpha_{IJ} f_\psi(z_{ij}; \boldsymbol{\beta^{I,J}}^T \mathbf{x_{ij}}))^{w_{ij}\tilde{p}_i(I)})^{\frac{1}{w_j}}, \quad \forall [j]_1^n, [J]_1^l,$
        where $c_j$ is a normalizing factor s.t. $\sum_{J=1}^l \tilde{p}_j(J) = 1$ and $w_j = \sum_i w_{ij}$.

        M-step
        Update Priors:
        $\alpha_{IJ} = \frac{\sum_{ij} w_{ij}\tilde{p}_i(I)\tilde{p}_j(J)}{\sum_{ij} w_{ij}}, \quad \forall [I]_1^k, [J]_1^l,$

        Update Regression Coefficients:
        $\boldsymbol{\beta}^{I,J} = argmax_{\boldsymbol{\beta}} \sum_{ij} w_{ij} \sum_{IJ} \tilde{p}_i(I)\tilde{p}_j(J)(z_{ij}\boldsymbol{\beta^{I,J}}^T \mathbf{x_{ij}} - \psi(\boldsymbol{\beta^{I,J}}^T \mathbf{x_{ij}})), \quad \forall [I]_1^k, [J]_1^l.$
Until convergence
Return $(\tilde{p}, \boldsymbol{\beta}, \boldsymbol{\alpha})$
```

Fig. 2. EM algorithm for estimation of the soft SCOAL model.

$$H(\tilde{p}_{ij}) = -\sum_{IJ} \tilde{p}_{ij}(I, J) log(\tilde{p}_{ij}(I, J)).$$

It can be shown that maximizing $F(\tilde{p}, \boldsymbol{\beta}, \boldsymbol{\alpha})$ with respect to $\tilde{p}$, $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ is equivalent to maximizing the data log-likelihood given by Eq. (5) [Neal and Hinton 1998]. $F(\tilde{p}, \boldsymbol{\beta}, \boldsymbol{\alpha})$ can be maximized by the classical EM procedure, which alternates between maximizing F with respect to $\tilde{p}$ for fixed $\boldsymbol{\beta}, \boldsymbol{\alpha}$ (E-step) and maximizing F with respect to $\boldsymbol{\beta}, \boldsymbol{\alpha}$ for fixed $\tilde{p}$ (M-step) and is guaranteed to converge to a local maximum of F.

However, the standard EM algorithm used to estimate the model is not scalable and can be very slow in practice for large m and n. To address this issue, one can impose structural constraints on the generative mixture model. SCOAL does so by restricting $\tilde{p}_{ij}$ to the form $\tilde{p}_{ij}(\rho(i), \gamma(j)) = \tilde{p}_i(\rho(i))\tilde{p}_j(\gamma(j))$, that is, $\rho(i)$ and $\gamma(j)$ are independent a-posteriori. Hence, we assume that every row i of the data matrix Z belongs to row cluster $I = 1$ to k with probability $\tilde{p}_i(I)$ and similarly every column j belongs to column cluster $J = 1$ to l with probability $\tilde{p}_j(J)$. By decoupling the row and column cluster assignments, SCOAL estimates the model very efficiently by an EM-based algorithm. The steps of the algorithm are illustrated in Figure 2. The exact form of the regression coefficients update step can be derived based on the assumed exponential family distribution. For the special case of a Gaussian distribution, the coefficients are computed by a weighted least squares regression. Each step monotonically increases the free energy function, finally converging to a locally optimal solution. Note that this algorithm is a generalization of the instances of the *hard* SCOAL algorithm described in Sections 3 and 4, for soft cluster assignments. Specifically, SCOAL with linear least-squares regression models corresponds to a mixture of Gaussians in Eq. (4), which results in the squared error cost function in Eq. (2). This model is also related to the soft PDLF model [Agarwal and Merugu 2007], where the mean of the z_{ij}s in each co-cluster is modeled as a sum $\boldsymbol{\beta}^T \mathbf{x_{ij}} + \delta_{I,J}$, where $\boldsymbol{\beta}$ is a global regression model and $\delta_{I,J}$ is a co-cluster specific offset.

6. EXTENSIONS

Extension to Other Regression and Classification Models. The algorithm presented in Sections 3 and 4 is not restricted to linear or logistic regression models, but actually suggests a meta-algorithm that can be extended to other predictive models by modifying (2) and/or (3). For example, in the regression setting, the data can be modeled by a collection of neural network models (MLP, RBF, etc.) or regression models with the L_1 norm regularization (Lasso [Hastie et al. 2001]). For classification problems one can use decision trees or Naive Bayes classifiers. The model update and the cluster reassignment steps will parallel those in Figure 1 to now reduce the loss function corresponding to the assumed predictive model.

Extension to Tensor Data. In the previous analysis, we assumed that the data was dyadic or bi-modal. In this section, we discuss its extension to the more general tensor setting. Let $\mathcal{Z}$ represent a tensor dataset with N modes. Let $\mathcal{X}$ denote the set of covariates associated with the elements of $\mathcal{Z}$. If there is heterogeneity along all the N modes, the aim is to assign entities along each of the N modes to clusters, such that all the response variables within each co-cluster can be represented by a common prediction model. Note that each co-cluster will now be an N-mode subtensor.

The problem formulation and objective function for the N-mode case are a natural extension of the 2-mode case. In case of a real-valued $\mathcal{Z}$, assuming linear regression models, the objective function is the squared error summed over all the elements of $\mathcal{Z}$. Since the objective function is still additive over the individual tensor elements, the iterative algorithm in Figure 1 can be readily extended to achieve a locally optimal solution. The co-cluster model update step now uses all the data entries within the subtensor corresponding to each co-cluster as training examples for the prediction models. The cluster update steps cycle through the modes, treating them like rows/columns in the bi-modal case. As in the case of bi-modal data, the algorithm is guaranteed to converge to a locally optimal solution.

7. REGULARIZATION

The SCOAL framework involves fitting $k \times l$ independent models to the data, one in each co-cluster. If p is the number of covariates, the total number of parameters to be learnt in case of linear models is $k \times l \times (1 + p)$. The overall model will have too many parameters for large values of k and l and may overfit in cases where training data is limited. In this section, we explore a number of regularization approaches that perform smoothing/shrinkage across the parameters to alleviate the overfitting problem.

7.1 Shrinkage Methods

One possible alternative to obtaining a generalizable set of $k \times l$ models is to update the model loss function to include a regularization term. Such approaches shrink the coefficients of each co-cluster model by imposing a penalty on their size. For instance, in a regression setting, assuming linear models, shrinkage can be achieved by ridge regression or lasso. Such a modification does not however change the SCOAL loss function (2) and the meta-algorithm for simultaneous co-clustering and regression can still be used to achieve a local minimum of the loss function. The only change is that the model update step will now involve learning ridge regression or lasso models rather than least squares models.

7.1.1 *Ridge Regression.* Ridge regression adds a penalty term equal to the sum of the squared model coefficients to the total squared residue to be minimized [Hastie et al. 2001]. The model coefficients hence minimize the following loss function,

$$\boldsymbol{\beta}^{\text{ridge}} = \arg_{\boldsymbol{\beta}} \min \left\{ \sum_{i=1}^{N} (y_i - \boldsymbol{\beta}^T \mathbf{x_i})^2 + \lambda \sum_{j=1}^{p} \boldsymbol{\beta}_j^2 \right\},$$

where y_i and $\mathbf{x_i}$ are the response variable and the vector of independent variables respectively. $\lambda \geq 0$ is a complexity parameter that controls the amount of regularization. The effect of increasing the value of λ is to reduce the magnitude of the regression coefficients.

The ridge regression solution is given by:

$$\boldsymbol{\beta}^{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T \mathbf{y} ,$$

where X is the design matrix and $\mathbf{y}$ is the vector of responses. Another advantage of using ridge regression is that the additional of λ to the diagonal of $X^T X$ makes it non-singluar, which is useful if $X^T X$ is not full rank.

7.1.2 *Lasso.* Like ridge regression, lasso adds a penalty term to the sum squared residue [Hastie et al. 2001]. The difference is that the penalty term in this case is the L_1 norm penalty, as compared to the L_2 norm ridge penalty. The model coefficients hence minimize the following,

$$\boldsymbol{\beta}^{\text{lasso}} = \arg_{\boldsymbol{\beta}} \min \left\{ \sum_{i=1}^{N} (y_i - \boldsymbol{\beta}^T \mathbf{x_i})^2 + \lambda \sum_{j=1}^{p} |\boldsymbol{\beta}_j| \right\}.$$

As the complexity parameter $\lambda \geq 0$ is increased, the model coefficients are driven to zero, leading to parsimonious models. This formulation makes the coefficients non-linear in the y_is, which can be computed by a quadratic programming approach.[3] Another important advantage of lasso is that it improves model interpretability by identifying the subset of the predictors that are the most influential.

In general, regularization can be achieved by Bridge regression [Friedman 2008], which defines a family of regression loss function penalized with the p-norm penalty. Note that ridge regression and lasso are special cases of bridge regression with the L_2 and L_1 norm penalties, respectively.

7.2 Reduced Parameter Approach

The simultaneous co-clustering and prediction approach with $k \times l$ independent models and a single prediction model for all the data are two extremes of the modeling spectrum in terms of the number of model parameters to tune. We now propose an intermediate approach, which constructs $k \times l$ models but with smoothing or regularization achieved by sharing parameters across certain sets of models. Here we assume that the covariates $\mathbf{x_{ij}}$ include only customer attributes $\mathbf{c_i}$ and product attributes $\mathbf{p_j}$. The co-cluster models are constructed in such a way that the customer coefficients of all models for the same row cluster and the product coefficients of all models for the same column cluster are constrained to be identical. This reduced setting has $(1 + |C|) \times k + (1 + |P|) \times l$ parameters, where $|C|$ and $|P|$ are the number of customer and product attributes respectively, which will be considerably lower than the total number of parameters for an independent set of $k \times l$ models. We now describe how the model parameters can be obtained for the regression problem.

If z_{uv} is the original value in row u, column v of the matrix Z that is assigned to row cluster g and column cluster h, the predicted value $\hat{z}_{uv}$ is now given by

$$\hat{z}_{uv} = \beta^{g}_{c0} + \boldsymbol{\beta}^{\boldsymbol{g}^T}_{\boldsymbol{c}} \mathbf{c_u} + \beta^{h}_{p0} + \boldsymbol{\beta}^{\boldsymbol{h}^T}_{\boldsymbol{p}} \mathbf{p_v},$$

where β^{g}_{c0} and β^{h}_{p0} are the customer and product intercepts and $\boldsymbol{\beta}^{\boldsymbol{g}}_{\boldsymbol{c}}$ and $\boldsymbol{\beta}^{\boldsymbol{h}}_{\boldsymbol{p}}$ are the customer and product coefficient vectors for the row cluster g and column cluster h respectively.

We still want to find a co-clustering defined by (ρ, γ) and associated regression models that minimize the objective function (2). The row and column cluster assignment steps remain the same. The update models step now involves solving a constrained optimization problem. Instead of updating $k \times l$ linear models independently, this step now updates the k row cluster models, such that the product coefficients are fixed and the customer coefficients are updated, and then the l column cluster models, in which the customer coefficients are fixed and the product coefficients are updated.

To update the model for row cluster g with r rows and n columns solve

$$min\, \mathbf{w}^T \left(\left(\mathbf{y} - X_c\left[\beta^{g}_{c0}, \boldsymbol{\beta}^{\boldsymbol{g}^T}_{\boldsymbol{c}}\right]^T\right) \otimes \left(\mathbf{y} - X_c\left[\beta^{g}_{c0}, \boldsymbol{\beta}^{\boldsymbol{g}^T}_{\boldsymbol{c}}\right]^T\right)\right),$$

where $\otimes$ represents elementwise multiplication. X_c is an $(r * n) \times (1 + |C|)$ matrix of the customer attributes, with the first column set to 1 for the intercept. The

[3] The following matlab code `http://www.cs.ubc.ca/~schmidtm/Software/lasso.html` was used for experimentation.

response variable $\mathbf{y}$ is a vector with $r * n$ elements, given by

$$\mathbf{y} = \mathbf{z} - X_p\left[\beta_{p0}^h, \boldsymbol{\beta}_{\boldsymbol{p}}^{\boldsymbol{h}^T}\right]^T,$$

where $\mathbf{z}$ is a vector of all the $r * n$ preference values in the row cluster g with associated weights $\mathbf{w}$, $[\beta_{p0}^h, \boldsymbol{\beta}_{\boldsymbol{p}}^{\boldsymbol{h}^T}]$ is a vector of the product coefficients of the corresponding column clusters and X_p is a matrix of size $(r * n) \times (1 + |P|)$ representing the product attributes corresponding to the preference values. The column cluster models are updated similarly. This update ensures that all the customer coefficients of models within the same row cluster and the product coefficients of models within the same column cluster are updated simultaneously and are identical.

8. MODEL SELECTION

The SCOAL meta-algorithm described in Section 3 requires the number of row clusters k and the number of column clusters l as an input. If the input k and l values are larger than the "true" k and l, then trying to learn the large number of parameters would tend to cause overfitting and lead to poor generalization on test data. In this section, we deal with the possibility of overfitting by choosing the k and l values in a systematic way.

Since the SCOAL algorithm directly tries to minimize a prediction loss function, it can be extended to use a cross validation procedure to select k and l that give the lowest validation set error. This is done by splitting the training dataset to obtain a validation set, on which the parameters k and l are tuned. We propose an efficient, top-down "bisecting" greedy algorithm that begins with $k = 1$ and $l = 1$ and then iteratively tries to increase the number of row and column clusters as long as the validation set error reduces. The detailed steps of the algorithm referred to as M-SCOAL are as follows. (ρ, γ) denotes the row and column cluster assignments and M the set of co-cluster models.

Algorithm M-SCOAL

(1) Run SCOAL with $k = 1$ and $l = 1$ and initialize (ρ, γ) and M.

(2) Try to split a row cluster. Select the row cluster with maximum average error on the validation set. Split the cluster into two by assigning half the rows with the largest row errors to a new row cluster and update ρ accordingly. Use this co-clustering (ρ, γ), with k increased by 1 to initialize a new run of SCOAL and compute the validation set error. If the error is lower than the error before splitting the row cluster, accept the split; otherwise revert to the previous co-clustering and set of models.

(3) Try to split a column cluster. Select the column cluster with maximum average error on the validation set. Split the column cluster into two and update γ. Use this co-clustering (ρ, γ), with l increased by 1 to initialize a new run of SCOAL. Follow a procedure similar to that in Step (2) to decide whether to accept the split.

(4) If neither a row or column cluster split reduces the error, then k and l are the same as at the end of the previous iteration. Terminate and use the current (ρ, γ) to initialize a final run of SCOAL; otherwise, loop back to Step (2).

Table I. Synthetic Datasets

	m,n	$\|C\|$,$\|P\|$	k,l	Noise σ^2
Dataset 1	100, 80	3,4	3,2	5
Dataset 2	100, 80	3,4	3,2	15
Dataset 3	500, 300	3,4	4,3	5
Dataset 4	900, 500	5,5	8,6	5

A key advantage of the model selection procedure is that it initializes each run of SCOAL with the previous co-clustering, which is likely to alleviate the local minima problem caused by poor initialization and lead to a better final solution. Although the model selection procedure involves several runs of SCOAL, the initialization from the previous run causes each one to converge much faster as compared to random initialization. Hence, M-SCOAL is not much slower than SCOAL. We found this to be well-supported by empirical evidence.

9. EXPERIMENTAL EVALUATION OF CLASSIFICATION RESULTS

9.1 Synthetic Datasets

The algorithm described in Section 3 was first evaluated on a number of synthetic datasets. We used synthetic data for experimentation as an initial sanity check before working with real data. These experiments also indicate the amount of improvement localized classification models provide when the model assumptions match the generative model for the data. Another motivation for using synthetic data was to evaluate the SCOAL model selection procedure (M-SCOAL) in a controlled setting, where the true k and l values are known. Each of the four synthetic datasets (Table I) was created by generating $k \times l$ blocks of data, each corresponding to a true cluster of class labels generated by a logistic regression model. Each block has a different logistic regression generative model, with its coefficient vector ($\boldsymbol{\beta}$) set randomly. The blocks are then appended together in the form of a grid to form a data matrix, whose rows and columns are then randomly shuffled. To assign a class label to a cell z_{ij} within each block, we begin by taking a linear combination of randomly generated customer and product attributes with the co-efficient vector $y_{ij} = \boldsymbol{\beta}^{\boldsymbol{T}} \mathbf{x_{ij}}$. We then add random Gaussian noise with variance σ^2 to all the y values. We obtain the probability of a cell belonging to the positive class as $P(z_{ij} = 1) = \frac{1}{1+e^{-y_{ij}}}$. A threshold of 0.5 is used to convert the probabilities into class labels.

Table I describes the synthetic datasets that were used for experimentation. Dataset 1 and 2 are very similar, dataset 2 has more noise and hence a weaker relationship with the underlying generative model as compared to dataset 1. Dataset 3 and 4 are larger, also with a substantial amount of noise. The datasets along with details of their generative models can be accessed at `http://www.ece.utexas.edu/~deodhar/modelCCData`.

9.1.1 *Results on Synthetic Data.* We evaluate the ability of SCOAL to classify unknown matrix values in the synthetic datasets. The data is split as 90% training and 10% test (missing) and the technique in Section 3 is used to predict the class label of the missing values, given the true k and l values as input. M-SCOAL, on the other hand selects the most suitable number of row and column clusters. The predicted labels are compared to the true labels and the classification quality is evaluated using precision, recall, F-measure and classification error. We compare the SCOAL approaches to the partitional co-clustering algorithm, Bregman co-clustering [Banerjee et al. 2007], which uses only the matrix Z without any attribute information. The Bregman co-clustering algorithm is very flexible and can work with several distance measures and co-cluster definitions. The special case of Bregman co-clustering that we compare with uses squared Euclidean distance as the distance measure and tries to find uniform co-clusters that minimize the distance of the data points within the co-cluster to the co-cluster mean,[4] since this case best matches the data generation process.

In order to apply Bregman co-clustering (CC) to this problem, in matrix Z, we encode positive class labels by the value 1 and negative by 0. The co-clustering algorithm approximates the cell values within each co-cluster by the co-cluster mean μ_{gh}. If a missing cell z_{ij} is assigned to row cluster g and column cluster h, with co-cluster mean μ_{gh}, we assign a class label to z_{ij} using the rule $z_{ij} = 1$ if $\mu_{gh} >$ threshold , $z_{ij} = -1$ otherwise. If the threshold is selected to be 0.5 this rule can be interpreted as assigning a missing cell the majority class label within its co-cluster.

In order to validate our claim that simultaneous co-clustering and modeling does better than a sequential, two-step approach of first partitioning the data and then learning models, we compare with Co-cluster Models, which partitions the data *a priori* by applying Bregman co-clustering to matrix Z, and then learns logistic regression models in each co-cluster. We also compare our approach with a single logistic regression classification model (Global Model), which is SCOAL with $k = 1$ and $l = 1$.

Table II displays the precision, recall, F-measure and classification error for SCOAL, M-SCOAL, co-clustering (CC), Co-cluster models and a single logistic regression model (Global Model). The results are averaged over 5 random 90-10% splits of the data. The values in parentheses are the standard errors. The threshold is set to 0.5 for all these experiments since the same threshold was used to obtain class labels from probability values while generating the synthetic data. One can observe that on all the synthetic datasets SCOAL and M-SCOAL do significantly better than CC, Co-cluster models and Global Model in terms of both the F-measure and the classification error. The performance of M-SCOAL is comparable to that of SCOAL with the true k and l. In some cases, M-SCOAL actually does better than SCOAL, which is explained by the initialization procedure of M-SCOAL that reduces the susceptibility of SCOAL to poor local minima. Table III shows the k and l values selected by M-SCOAL

[4] This corresponds to scheme 2 of the Bregman co-clustering algorithm [Banerjee et al. 2007] with squared Euclidean distance.

Table II. Comparison of Classification Performance on 4 Synthetic Datasets

Algorithm	Precision	Recall	F-Measure	Classification Error
Dataset 1				
Global Model	0.894 (0.004)	0.952 (0.004)	0.922 (0.004)	0.131 (0.006)
CC	0.881 (0.007)	0.948 (0.003)	0.913 (0.003)	0.149 (0.006)
Co-cluster Models	0.91 (0.006)	0.95 (0.005)	0.929 (0.003)	0.117 (0.005)
SCOAL	**0.967 (0.003)**	**0.979 (0.002)**	**0.973 (0.002)**	**0.044 (0.002)**
M-SCOAL	0.961 (0.008)	0.969 (0.005)	0.965 (0.006)	0.058 (0.009)
Dataset 2				
Global Model	0.883 (0.008)	0.927 (0.009)	0.904 (0.007)	0.151 (0.01)
CC	**0.910 (0.002)**	0.880 (0.004)	0.895 (0.003)	0.159 (0.004)
Co-cluster Models	0.883 (0.003)	0.929 (0.007)	0.906 (0.004)	0.149 (0.005)
SCOAL	0.908 (0.006)	**0.937 (0.005)**	**0.922 (0.005)**	**0.124 (0.007)**
M-SCOAL	0.906 (0.007)	0.934 (0.004)	0.920 (0.005)	0.127 (0.007)
Dataset 3				
Global Model	0.926 (0.001)	0.972 (0)	0.948 (0)	0.092 (0.001)
CC	0.935 (0.004)	0.956 (0.004)	0.945 (0.001)	0.096 (0.001)
Co-cluster Models	0.928 (0.002)	0.969 (0.001)	0.948 (0.001)	0.091 (0.002)
SCOAL	0.968 (0.002)	0.979 (0)	0.974 (0.001)	0.046 (0)
M-SCOAL	**0.974 (0.001)**	**0.981 (0.001)**	**0.978 (0.001)**	**0.038 (0.001)**
Dataset 4				
Global Model	0.928 (0)	0.971 (0)	0.949 (0)	0.09 (0)
CC	0.931 (0.003)	0.968 (0.002)	0.949 (0.001)	0.09 (0.001)
Co-cluster Models	0.936 (0)	0.969 (0)	0.952 (0)	0.083 (0)
SCOAL	**0.979 (0)**	**0.984 (0)**	**0.981 (0)**	**0.032 (0)**
M-SCOAL	0.979 (0)	0.984 (0)	0.981 (0)	0.032 (0.005)

Table III. k and l Values Selected by M-SCOAL on the Synthetic Datasets

	Data 1	Data 2	Data 3	Data 4
Run	$k = 3, l = 2$	$k = 3, l = 2$	$k = 4, l = 3$	$k = 8, l = 6$
1	3,2	2,3	4,3	8,6
2	2,4	3,2	4,3	8,6
3	3,2	3,2	4,3	8,6
4	3,2	3,2	4,3	8,7
5	2,2	3,2	4,3	8,6

on the synthetic datasets, over 5 trials, highlighting that the selected values are consistently very close to the true k and l values.

On these datasets we also evaluate the ability of SCOAL to reconstruct the original data matrix. We find that on all the datasets this approach is consistently able to recover a close approximation of the original data matrix. Additionally, the cluster assignments made by the algorithm closely match the true underlying cluster labels.

9.2 Recommender System Application

This classification problem deals with predicting the course choices made by masters students at a large Midwestern University. The objective is to use the information of previous known course choices of students to predict unknown choices. These predictions can be used to recommend the right courses for students to take in the future. The data includes a matrix of 326 students vs. 32 courses with class labels = 1 if the student took the course and −1 otherwise. Each student has attributes including the student's career aspiration and undergraduate degree. The course attributes include the department offering the course, the course evaluation score and a binary variable indicating whether the course is quantitative. The dataset is skewed with unequal priors of the positive and negative classes. Around 25% of the student course choices are positive and the rest negative.

9.2.1 *Classifying Missing Cell Values.* In order to test the classification capability of SCOAL on this problem, the data is split as 90% training and 10% test and the class labels in the test set, that is, the unknown student-course choices, are predicted using the co-cluster models. Results are obtained by averaging over 10 random 90-10% data splits. We compare the SCOAL (with logistic regression models) results with CC, Co-cluster Models and Global Model. The number of row and column clusters for SCOAL, CC and Co-cluster Models are set to 2. We additionally compare with the SCOAL model selection procedure described in Section 8 (M-SCOAL). For assigning class labels to the missing matrix entries we use a threshold that is varied from 0.1 to 0.9 to get a range of precision-recall tradeoffs.

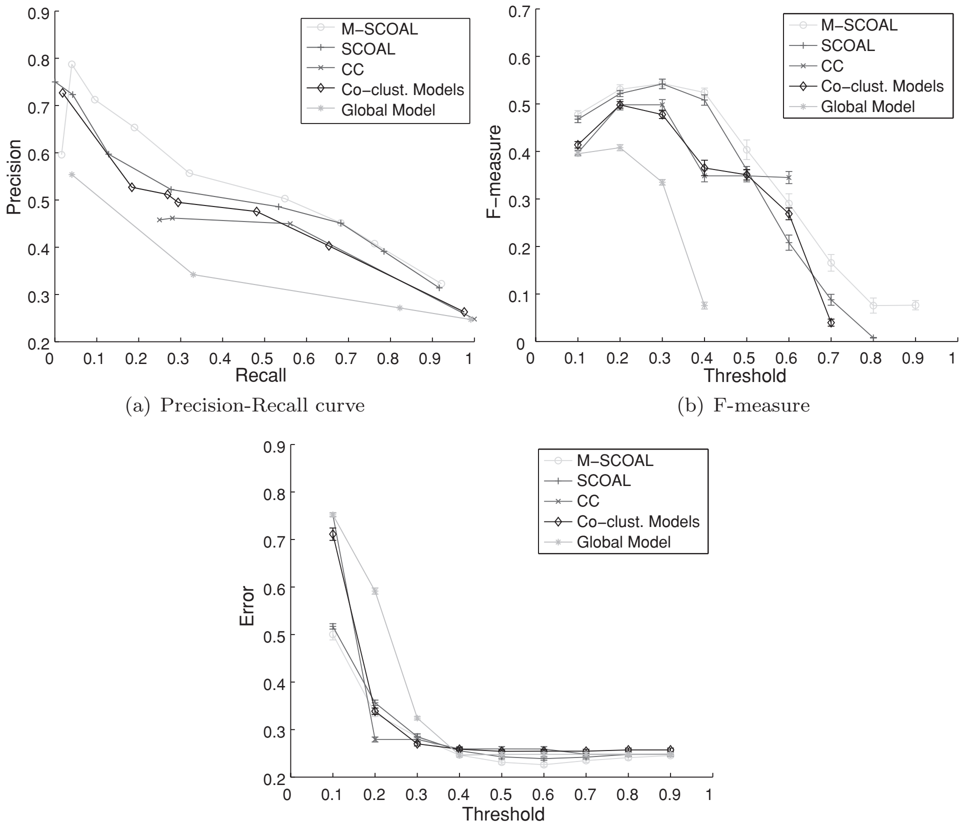
Figures 3(a), 3(b), and 3(c) display the precision-recall curves, the F-measure and the classification error of the algorithms at different values of the threshold. Beyond a certain threshold CC, Co-cluster Models and Global Model classify all the data points as belonging to the negative class, causing the F-measure to be undefined. Such points are excluded from the Precision-Recall curve and the F-measure plot. Both M-SCOAL and SCOAL do significantly better than CC, Co-cluster Models and Global Model in terms of precision and recall as can be seen in the Precision-Recall curves and hence their F-measure is consistently better than the other approaches over different values of the threshold. The classification error of the SCOAL approaches is also lower than the other approaches. At threshold value 0.2 however, CC has a lower error as it has a much smaller number of false positives.

10. EXPERIMENTAL EVALUATION OF REGRESSION RESULTS

10.1 Real Marketing Dataset

We applied SCOAL to a challenging marketing application. Given purchase information for a set of customers and products along with customer and product attributes, we simultaneously clustered customers and products and used the co-clustering solution to predict unknown customer-product purchase information. The obtained co-clusters also provide information about customer segments in the market and equivalent product groups, achieving simultaneous market segmentation and structure, an important problem in marketing research [Grover and Srinivasan 1987]. The dataset that we used is the publicly available ERIM dataset[5] consisting of household panel data collected by

[5] URL: http://www.gsb.uchicago.edu/kilts/research/db/erim/.



(a) Precision-Recall curve

(b) F-measure

(c) Classification error

Fig. 3. Evaluation of classification of missing student-course choices in the recommender system application. Note that CC has previously been shown to perform better than classic collaborative filtering approaches [George and Merugu 2005].

A.C. Nielsen, which is well known in the marketing community and has been used by several researchers [Kim and Rossi 1994; Kim and Sullivan 1998; Seetharaman et al. 1999]. This dataset has purchase information for six product categories over a period of 3 years from 1985-1988 for households in Sioux Falls, South Dakota. The dataset includes household demographics and product characteristics.

The data preprocessing steps we took are similar to the data selection procedure used by Seetharaman et al. [1999]. We have six product categories (ketchup, tuna, sugar, tissue, margarine and peanut butter) with a total of 121 products. Brands with very low market share in each product category are omitted. We select households that made at least two purchases in each product category, resulting in a set of 1714 households. We select six household attributes - income, number of residents, male head employed, female head employed, total visits and total expense and three product attributes - market share, price, number of times the product was advertised. The data can be represented by a data matrix of households and brands where the cell values are the number of units of a brand purchased by a household, aggregated over the time the household was tracked. The number of units purchased can be used as an indicator of brand preference.

10.1.1 *Dataset Properties.* The data matrix is extremely sparse, with 74.86% of the values being 0. The distribution of the number of units purchased is very skewed. 99.12% of the values are below 20, while the remaining values are very large and range upto around 200. These few, large values that form the tail in the histogram of the matrix entries, can be considered as outliers with respect to the rest of the values.

10.1.2 *Standardization of the Data.* The dimensions (items) in this application are products from 6 different product categories. The product attributes such as price and extent of advertising could vary widely from one category to another. When we construct a linear model for a co-cluster we weigh the attributes of all the products in the co-cluster by the same set of coefficients. However, the products in the co-cluster could be from different categories with very different ranges of attribute values. We hence need to standardize the product attributes to make them comparable across categories. We transform each product attribute value a to $a^{'} = \frac{a-\mu_c}{\sigma_c}$, where μ_c and σ_c are the mean and standard deviation within the corresponding product category c. This problem does not arise in case of the customer attributes since they are relatively comparable. The matrix cell values, which are the number of units purchased could also be very different across categories and have to be standardized. The cell values z_{ij} within each sub-matrix of all the products belonging to a specific product category c are transformed to $z_{ij}^{'} = \frac{z_{ij}-\mu_{z_c}}{\sigma_{z_c}}$ where μ_{z_c} and σ_{z_c} are the mean and standard deviation of the all the values in the sub-matrix. Since the standardization of the data is a linear transformation, the dataset properties described in Section 10.1.1 continue to hold.

10.1.3 *Data Reconstruction.* Table IV displays the mean squared error of the approximated data matrix obtained by the different algorithms on the entire standardized dataset, averaged over 10 runs. Global Model is a single linear regression model, while CC is Bregman Co-clustering [Banerjee et al. 2007], which does not use the attribute information. Co-cluster Models is a two step modeling procedure of first co-clustering and then learning linear regression models in each co-cluster. “Row Clustering” is SCOAL with the number of column clusters set to 1 and “Column Clustering” is SCOAL with the number of row clusters set to 1. Reduced Model is the SCOAL approach with the reduced set of parameters as described in Section 7.2. Note that SCOAL obtains the best reconstruction of the original matrix as compared to the other approaches in terms of MSE (mean squared error). Table IV also shows the average R^2 of the linear models constructed within each co-cluster. The R^2 values are actually quite low, indicating that a strong linear relationship does not really exist in the data, which is to the disadvantage of the simultaneous co-clustering and regression algorithm.

Table IV. Reconstruction Error on Entire ERIM Dataset

Algorithm	MSE	Avg. R^2
Global Model (k=1, l=1)	0.930 (0)	0.0696
CC (k=10, l=4)	0.842 (0.003)	–
Co-cluster Models (k=10, l=4)	0.835 (0.002)	0.0122
Row Clustering (k=10, l=1)	0.887 (0)	0.0896
Column Clustering (k=1, l=4)	0.88 (0.001)	0.0714
SCOAL (k=10, l=4)	**0.794 (0.005)**	0.1308
Reduced Model (k=10, l=4)	0.876 (0)	0.0426

Table V. Comparison of Prediction Error on ERIM Dataset

Algorithm	Train Err.	Test Err.	Test Err. Orig.	Avg. R^2
Global Model (k=1, l=1)	0.931 (0.003)	0.928 (0.023)	16.776 (0.584)	0.067
Cluster Models (k=4)	0.927 (0.003)	0.917 (0.026)	16.854 (0.51)	0.067
CC (k=4, l=4)	0.853 (0.003)	0.905 (0.022)	15.501 (0.511)	–
SCOAL (k=4, l=4)	0.823 (0.002)	0.905 (0.021)	15.688 (0.519)	0.113
Reduced Model (k=4, l=4)	0.878 (0.002)	0.887 (0.023)	15.339 (0.553)	0.056
M-SCOAL	0.874 (0.004)	**0.873 (0.019)**	**15.032 (0.416)**	0.071

10.1.4 *Predicting Unknown Data Values.* The prediction error of the different approaches on this problem is computed by averaging over 10 random 90-10% training and test data splits. Here we additionally compare with a two-step sequential approach (Cluster Models) that first clusters the customers based on their attributes and then fits regression models in each customer cluster. Table V shows the training error (Train Err.), the test set error (Test Err.) on the standardized data and the test set error on the original, unstandardized dataset obtained by back transforming the standardized data (Test Err. Orig.). A more comprehensive comparison of the SCOAL approaches with alternative modeling techniques is presented in Section 10.3.

As mentioned in Section 10.1.1 the data is skewed with a few very large outliers. In the presence of outliers the clusterwise models overfit the training data and do not generalize well to the test data. SCOAL is the most susceptible to overfitting since it is the most complex and involves the most number of parameters. SCOAL does better than Global Model and Cluster Models but slightly worse than CC. Reduced Model does better than SCOAL in this scenario since it has fewer parameters and a simpler overall model, which generalizes better. Reduced Model does slightly better than CC as well, indicating that using the attribute information in the prediction process helps. However, this improvement is small, which can be explained by the fact that the data does not show a very strong linear relation.

Linear "least squares" regression is very sensitive to outliers and a few outliers could skew the model results. Hence, on this dataset, for a fair comparison of linear model based techniques with respect to prediction of missing values, we need some way of dealing with outliers. It is unreasonable for a model based on linear regression to capture both small as well as extremely large values simultaneously and a more suitable approach would be to separate out these two very different sets of values and model them independently. A threshold of 20 number of units purchased was used to separate the bulk of the matrix

Table VI. Prediction Error on ERIM Dataset (Low-Valued Entries)

Algorithm	Train Err.	Test Err.	Test Err. Orig.	Avg. R^2
Global Model (k=1, l=1)	0.913 (0.001)	0.920 (0.01)	4.24 (0.06)	0.093
Cluster Models (k=4)	0.91 (0.001)	0.917 (0.01)	4.228 (0.059)	0.084
CC (k=4, l=4)	0.833 (0.001)	0.890 (0.009)	4.002 (0.056)	–
SCOAL (k=4, l=4)	0.804 (0.001)	0.883 (0.007)	3.965 (0.044)	0.143
Reduced Model (k=4, l=4)	0.849 (0.001)	0.872 (0.009)	3.893 (0.052)	0.08
M-SCOAL	0.848 (0.004)	**0.856 (0.007)**	**3.832 (0.035)**	0.081

entries (99.12%) from the tail of high values. This can easily be handled by the co-clustering algorithm by setting the weight of the outlier points to 0. We now focus on the prediction problem and the model for the bulk of the entries, the results of which are illustrated in Table VI.

Results on the Low-Valued Matrix Entries. Table VI shows the training mean squared error and the test set error for the different approaches on this problem. One can see that M-SCOAL outperforms all the other approaches. The number of customer and product clusters identified by M-SCOAL range from $1-2$ and $4-6$ respectively. All the SCOAL approaches do significantly better than Global Model and Cluster Models on the test set. SCOAL and Reduced Model also do slightly better than CC. Avg. R^2 is the average R^2 of the regression models. One can observe that while the average R^2 for SCOAL is the best among the different approaches, it is still relatively small. Moreover, Reduced Models and M-SCOAL have a lower R^2 than Global Model, but also a lower test MSE. The improvement in the prediction error is hence primarily due to representing the response values in each co-cluster by a different mean as compared to one global mean. Learning multiple linear models does not help as much due to the weak inherent linear relationship in the data.

Our complete model for the prediction problem consists of the model constructed for the bulk of the matrix entries as described above and a linear model for the outliers. A classifier is trained to appropriately select one of the two models to predict each unknown matrix value. An alternative way of dealing with outliers is to reduce the influence of the extremely high values on the constructed models, allowing them to generalize better. This is achieved by giving high-valued matrix entries a very small fixed weight, enabling the linear models to focus on the bulk of the values and rather than fit a few high values. Through experiments conducted for both these cases [Deodhar and Ghosh 2007], we observe that here as well, the SCOAL-based approaches do better than the other techniques.

10.2 MovieLens Dataset

Another application on which we evaluated the SCOAL framework is that of predicting user-movie ratings in a recommender system setting. We applied SCOAL to the MovieLens dataset, which consists of 100,000 ratings (1-5) from 943 users on 1682 movies made available by the GroupLens Research Project at the University of Minnesota.[6] We used a set of 23 attributes including user

[6] The data can be downloaded from `http://www.grouplens.org/system/files/ml-data.tar__0.gz`.

Table VII. Comparison of Prediction Error on the MovieLens Dataset

Algorithm	Training Err.	Test Err.	Avg. R^2
Global Model (k=1, l=1)	1.192 (0)	1.192 (0.004)	0.059
Cluster Models (k=4)	1.185 (0.001)	1.189 (0.003)	0.065
CC (k=4, l=4)	0.881 (0.001)	0.942 (0.004)	–
Reduced Model (k=4, l=4)	0.887 (0.002)	**0.924 (0.004)**	0.16
SCOAL (k=4, l=4)	0.876 (0.001)	0.943 (0.004)	0.257
M-SCOAL	0.883 (0.002)	0.937 (0.003)	0.256

demographics like age, gender, employment status and movie date and genre. The dataset is extremely sparse, with the proportion of known user-movie ratings as small as 6.3% of the size of the data matrix.

Table VII displays the training and test set mean squared error of different approaches on this problem, computed by averaging over 10 random 80-20% training and test data splits. As in the case of the ERIM dataset, the SCOAL based approaches do well, with Reduced Model doing significantly better than the competing techniques. Reduced Model and M-SCOAL improve upon SCOAL as well, indicating the importance of reducing the number of parameters to be tuned. The k and l selected by M-SCOAL range from $4-5$ and $3-5$ respectively. The average R^2 of the SCOAL approaches is significantly larger than that of the Global Model. Hence, the improvement in the prediction error of SCOAL is due to both, a collection of better fitting linear models as well as a co-cluster specific mean of the response variables, which is in contrast to the situation on the ERIM dataset (Table VI).

Figure 4 provides interesting insights into the nature of the structure in the data learnt by a single global model versus the collection of local models constructed by SCOAL. Both, the global model and SCOAL ($k = 4, l = 4$) are initially trained on a small subset (10%) of the training data. The number of training points input to the two predictive techniques is then progressively increased along the x-axis in 80 batches each of 1000 randomly selected points. SCOAL uses the previously learnt co-clustering to initialize each new run with an increased training dataset size. Mean squared error on a held out test set is plotted on the y-axis. One can notice that the global model captures the linear structure in the data fairly quickly and attains a reasonable accuracy on the test set with relatively few training examples. In contrast, the performance of SCOAL with a very small amount of training data is not as good, since the available data is not sufficient to fit the 16 local models. However, as more training data is provided, the global model saturates in performance, while SCOAL continues to fit the data better by capturing more complex local structures.

10.3 Comparison with Alternate Approaches

In this section, we present a comparative study of SCOAL evaluated against a number of alternate predictive modeling approaches, described as follows.

(1) *Two-Way Cluster Models*. The customers are clustered into k row clusters based on the customer attributes using a 1-sided clustering algorithm like

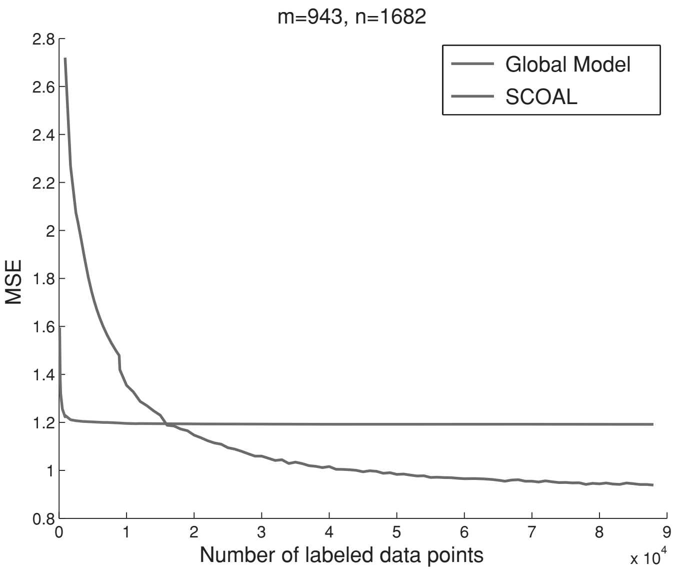


Fig. 4. Error on a held out test set of models learnt at increasing amounts of training data on the MovieLens dataset.

k-means. The products are independently clustered into l column clusters based on the product attributes. The 1-sided row and column cluster assignments induce a grid of $k \times l$ co-clusters on the data matrix. A linear regression model that relates the independent variables to the target values is then learnt in each co-cluster.

(2) *Co-Cluster Models*. The partitional Bregman co-clustering algorithm [Banerjee et al. 2007] is applied to the matrix of data values to obtain $k \times l$ co-clusters. Linear regression models are then trained in each co-cluster. This approach is included to indicate how a simultaneous co-clustering and learning strategy compares with a two-step sequential co-clustering and learning procedure.

(3) M5'. This is the model tree algorithm proposed by Wang and Witten [1997], discussed in Section 2. This is again a sequential approach, in which the input space is first partitioned using a decision tree, followed by the construction of predictive models for each partition.

(4) *Neural Network*. A multilayer perceptron (MLP) model is evaluated on the datasets to give some idea of how a single non-linear model would perform in contrast to modeling the data using a collection of linear models.

Table VIII compares and contrasts the test set mean squared error of the different approaches on the ERIM and MovieLens datasets. The SCOAL and reduced parameter approaches with ridge regression and M-SCOAL are included in this table. Two-way cluster models and co-cluster models use $k = 4, l = 4$. The parameter in M5' that specifies the minimum number of instances required to split a node is set to 100. The MLP has 2 hidden layers, each with

Table VIII. Mean Squared Error of Different Modeling Techniques on the ERIM and MovieLens Datasets, Averaged Over 10 90-10% Train-Test Data Splits

Algorithm	ERIM	ERIM (Low Valued Entries)	MovieLens
SCOAL	15.807 (0.49)	3.965 (0.044)	0.943 (0.004)
Reduced Model	**14.021 (0.443)**	3.902 (0.04)	**0.93 (0.004)**
M-SCOAL	15.032 (0.416)	**3.832 (0.035)**	0.935 (0.005)
Two-way cluster models	17.062 (1.038)	4.091 (0.0267)	1.168 (0.005)
Co-cluster models	16.049 (0.588)	3.967 (0.034)	0.931 (0.003)
M5'	15.457 (0.816)	3.877 (0.037)	1.116 (0.006)
MLP	17.169 (0.938)	4.395 (0.193)	1.462 (0.127)

Table IX. Comparison of Regularized Linear Regression Models on the ERIM and MovieLens Datasets

Regularization	Global Model	SCOAL	Reduced Model
ERIM			
No regularization	16.776 (0.584)	15.688 (0.519)	15.339 (0.553)
Ridge	17.013 (0.528)	15.807 (0.49)	**14.021 (0.443)**
Lasso	16.776 (0.584)	15.051 (0.315)	15.154 (0.475)
ERIM (low values entries)			
No regularization	4.24 (0.06)	3.965 (0.044)	3.893 (0.052)
Ridge	4.24 (0.06)	3.949 (0.033)	3.902 (0.04)
Lasso	4.24 (0.06)	3.957 (0.031)	**3.866 (0.044)**
MovieLens			
No regularization	1.192 (0.004)	0.943 (0.004)	**0.924 (0.004)**
Ridge	1.192 (0.004)	0.946 (0.004)	0.93 (0.004)
Lasso	1.192 (0.004)	0.938 (0.006)	0.94 (0.018)

10 hidden units and has a learning rate of 0.3. The SCOAL approaches perform consistently well, giving better results than even M5' and the MLP model. Two-way cluster models and co-cluster models have higher error than SCOAL, validating the benefits of a simultaneous partitioning and modeling strategy as compared to a priori segmentation and modeling. Nonlinear models like the MLP possibly need more extensive tuning to give better performance.

10.4 Evaluation of Regularized Models

We now evaluate the effects of the regularization methods discussed in Section 7.1 on the prediction error of SCOAL. Table IX compares the mean squared error of SCOAL without regularization with SCOAL based on ridge regression and lasso models on the ERIM and the MovieLens datasets. Each value in the table is an average obtained over 10 random 90-10 % train-test splits of the data. The value of the regularization parameter λ in each case is selected by cross-validation. One can see that on the ERIM dataset, which is known to have a very skewed distribution of response values and the presence of outliers, ridge regression and lasso boost the performance of SCOAL.

10.5 Nonlinear Co-Cluster Models

As described in Section 6, the SCOAL algorithm is not restricted to logistic or linear regression models but can use other predictive models in each co-cluster by suitably modifying the objective function (2) and/or (3). In this section, we

Table X. Mean Squared Error of SCOAL Techniques, Using MLPs as Predictive Models in Each Co-Cluster vs. a Single Global MLP, on the ERIM and MovieLens Datasets

Algorithm	ERIM	ERIM (Low Valued Entries)	MovieLens
Global	15.406 (0.67)	4.0183 (0.0533)	1.138 (0.004)
SCOAL	17.575 (0.433)	4.076 (0.037)	0.967 (0.002)
M-SCOAL	**15.204 (0.434)**	**3.917 (0.033)**	**0.953 (0.005)**

evaluate simultaneous co-clustering and regression with a non-linear model (Multi-Layer Perceptron) learnt in each co-cluster. Table X compares the test MSE of a single MLP modeling all the data (Global) with the SCOAL and M-SCOAL approaches, averaged over 10 random 90-10 % train-test data splits. The MLP used is a feed-forward backpropagation network, with the hidden layer using the log-sigmoid transfer function and the output layer using the linear transfer function. The MLP model complexity for the global model and the SCOAL models is tuned through cross-validation. The learning rate for all models is set to 0.05. SCOAL is run with four row and four column clusters respectively, while M-SCOAL determines the most suitable number of row and column clusters. The trend observed here is similar to that in the comparisons with linear models illustrated in Sections 10.1 and 10.2. M-SCOAL consistently outperforms the global model, while SCOAL seems to overfit slightly on the ERIM dataset due to the presence of outliers. Comparing these results with the ones in Table VIII, one can observe that on all datasets, SCOAL with linear models does better than SCOAL with MLP models. This is probably because a collection of simple linear models is sufficient to capture the heterogeneity and structure in these datasets. Multiple non-linear models, with a larger number of parameters require more tuning and may be an overkill.

11. SUMMARY

Based on the results in Sections 9 and 10 we observe that the simultaneous co-clustering and prediction approach holds promise for both classification and regression problems, at least for the datasets examined so far. One should be able to further improve the results by using alternative prediction models within each co-cluster that more closely conform to the data characteristics. This is particularly true for the ERIM dataset which is known to have significant outliers, but we still used linear least squares regression as it is widely adopted and understood. Note that overfitting of the models to outliers is addressed to some extent by the regularization techniques. Further improvements can be achieved by using a more robust error function rather than squared error [Huber 1981]. Moreover, non-linear models would help because the limitations of linear models for this dataset is quite evident by the low R^2 values obtained by several researchers who previously applied such models to ERIM. We would like to point out that the SCOAL framework shows the most value on large datasets that are heterogeneous enough to require a collection of localized models for adequate representation and where enough data is available to tune the parameters of each of the models.

The results in Section 10.3 illustrate that SCOAL does very well in terms of prediction accuracy as compared to a variety of competing techniques. Apart from accuracy, SCOAL additionally provides the following features that distinguish it from related predictive modeling techniques.

(1) SCOAL captures the inherent structure in the data by a collection of simple (possibly linear) local models. The models indicate the degree to which predictors influence the response in different regions of the input space and are hence easily actionable. This ease of interpretability of the models is valuable to applications in domains such as marketing.
(2) The SCOAL algorithm is simple and computationally efficient. For instance, in case of least squares linear regression models, each iteration is linear in the size of the data.
(3) The independence of the operations involved in the SCOAL meta-algorithm allows it to be easily parallelized and adapted to multi-core architectures. This enables the SCOAL framework to scale to very large real life datasets.
(4) The generic SCOAL framework can be suitably tailored to the application by the right choice of predictive models, cost function and regularization mechanism.
(5) The fit of the predictive models in their local regions provides an idea of the goodness of representation of different regions of the input space. This intuition is useful in devising a mechanism that estimates the reliability/certainty of predicted response values [Deodhar and Ghosh 2009].

12. CONCLUDING REMARKS

Simultaneous co-clustering and modeling is a promising framework that generalizes co-clustering, collaborative filtering and traditional segment-wise modeling. Note that this approach is not limited to the algorithms presented in Sections 3 and 4, but forms a broad and versatile framework for solving difficult classification and regression problems in general.

While this article concentrated on marketing and recommender system data for illustration, there are many other domains characterized by data matrices supplemented by annotated row and column entities. For example, this approach can be used to analyze microarray data with gene and experiment annotations, social network settings with sets of attributes attached to persons (rows) and relationships (columns), and clustering of web documents annotated by link as well as semantic information. It will be worthwhile to investigate specific instances of this framework (with specific choices of the co-clustering model used as well as of the classifier/predictor used) in terms of their suitability for different problems within this wide range of application domains.

ACKNOWLEDGMENTS

We would like to thank Prof. McAlister and Andrea Godfrey of the McCombs School of Business at the University of Texas at Austin for the student course choices dataset and insightful discussions.

REFERENCES

AGARWAL, D. AND CHEN, B. 2009. Regression-based latent factor models. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09)*. 19–28.

AGARWAL, D. AND MERUGU, S. 2007. Predictive discrete latent factor models for large scale dyadic data. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'07)*. 26–35.

BANERJEE, A., DHILLON, I., GHOSH, J., MERUGU, S., AND MODHA, D. 2007. A generalized maximum entropy approach to Bregman co-clustering and matrix approximation. *J. Mach. Learn. Resear. 8*, 1919–1986.

BANERJEE, A., MERUGU, S., DHILLON, I., AND GHOSH, J. 2005. Clustering with Bregman divergences. *J. Mach. Learn. Resear. 6*, 1705–1749.

BAUMANN, T. AND GERMOND, A. 1993. Application of the kohonen network to short-term load forecasting. In *Proceedings of the International Conference on Neural Networks to Power System (ANNPS)*. 407–412.

BREIMAN, L., FRIEDMAN, J. H., OLSHEN, R. A., AND STONE, C. J. 1984. *Classification and Regression Trees*. Wadsworth, Belmont, CA.

CHENG, Y. AND CHURCH, G. M. 2000. Biclustering of expression data. In *Proceedings of the International Conference on Intelligent Systems for Molecular Biology (ICMB)*. 93–103.

CHO, H., DHILLON, I. S., GUAN, Y., AND SRA, S. 2004. Minimum sum squared residue co-clustering of gene expression data. In *Proceedings of the SIAM Conference on Data Mining (SDM)*.

DEODHAR, M. AND GHOSH, J. 2007. A framework for simultaneous co-clustering and learning from complex data. Department of Electrical and Computer Engineering University of Texas at Austin, IDEAL-2007-08, `http://www.lans.ece.utexas.edu/papers/techreports/deodhar07Coclust.pdf`.

DEODHAR, M. AND GHOSH, J. 2009. Mining for the most certain predictions from dyadic data. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'09)*. 249–258.

DHILLON, I., MALLELA, S., AND MODHA, D. 2003. Information-theoretic co-clustering. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'03)*. 89–98.

DJUKANOVIC, M., BABIC, B., SOBAJIC, D., AND PAO, Y. 1993. Unsupervised/supervised learning concept for 24-house load forecasting. *IEE Proc.-Generation, Transmiss. Distrib. 140*, 311–318.

FRIEDMAN, J. 2008. Fast sparse regression and classification. Tech. rep., Stanford University.

GEORGE, T. AND MERUGU, S. 2005. A scalable collaborative filtering framework based on co-clustering. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'05)*. 625–628.

GILL, P. E., MURRAY, W., AND WRIGHT, M. H. 1981. *Practical Optimization*. Academic Press, Harcourt Brace and Company, London.

GROVER, R. AND SRINIVASAN, V. 1987. A simultaneous approach to market segmentation and market structuring. *J. Market. Res.*, 139–153.

HASTIE, T., TIBSHIRANI, R., AND FRIEDMAN, J. 2001. *The Elements of Statistical Learning*. Springer, New York.

HERLOCKER, J., KONSTAN, J., BORCHERS, A., AND RIEDL, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 230–237.

HUBER, P. J. 1981. *Robust Statistics*. Wiley, New York.

JORDAN, M., JACOBS, R., NOWLAN, S. J., AND HINTON, G. E. 1991. Adaptive mixtures of local experts. *Neural Computat. 3*, 79–87.

JORNSTEN, R. AND YU, B. 2003. Simultaneous gene clustering and subset selection for sample classification via MDL. *BMC Bioinformat. 19,* 9, 1100–1109.

KIM, B. AND ROSSI, P. 1994. Purchase frequency, sample selection, and price sensitivity: The heavy-user bias. *Market. Lett.*, 57–67.

KIM, B. AND SULLIVAN, M. 1998. The effect of parent brand experience on line extension trial and repeat purchase. *Market. Lett.*, 181–193.

Lee, W. and Liu, B. 2003. Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*.

Liu, X., Krishnan, A., and Mondry, A. 2005. An entropy-based gene selection method for cancer classification using microarray data. *BMC Bioinformat. 6*, 76.

Lokmic, L. and Smith, K. A. 2000. Cash flow forecasting using supervised and unsupervised neural networks. *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000)* , 6343.

Madeira, S. C. and Oliveira, A. L. 2004. Biclustering algorithms for biological data analysis: A survey. *IEEE Trans. Comput. Biol. Bioinf 1*, 1, 24–45.

McCullagh, P. and Nelder, J. A. 1983. *Generalized Linear Models*. Chapman and Hall, London.

Neal, R. M. and Hinton, G. E. 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, MIT Press, 355–368.

Oh, K. and Han, I. 2001. An intelligent clustering forecasting system based on change-point detection and artificial neural networks: Application to financial economics. In *Proceedings of the Itawaii International Conference on Systems Science (HICSS-34)*. 3011.

Quinlan, J. R. 1992. Learning with continuous classes. In *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*. World Scientific, 343–348.

Ramamurti, V., and Ghosh, J. 1998. On the use of localized gating in mixtures of experts networks. (*invited paper*) In *Proceedings of the SPIE Conference on Applications and Science of Computational Intelligence*. 24–35.

Seetharaman, P., Ainslie, A., and Chintagunta, P. 1999. Investigating household state dependence effects across categories. *J. Market. Res.*, 488–500.

Sfetsos, A. and Siriopoulos, C. 2004. Time series forecasting with a hybrid clustering scheme and pattern recognition. *Systems, Man and Cybernetics, Part A, IEEE 34*, 399–405.

Sharkey, A. 1996. On combining artificial neural networks. *Conn. Sci. 8*, 3/4, 299–314.

Wang, Y. and Witten, I. H. 1997. Inducing model trees for continuous classes. In *Proceedings of the 9th European Conference on Machine Learning*. 128–137.

Wedel, M. and Steenkamp, J. 1991. A clusterwise regression method for simultaneous fuzzy market structuring and benefit segmentation. *J. Market. Res.*, 385–396.

Zhang, S., Neagu, D., and Balescu, C. 2005. Refinement of clustering solutions using a multi-label voting algorithm for neuro-fuzzy ensembles. In *Proceedings of the International on Natural Computation (ICNC)*. 1300–1303.

Received January 2009; revised September 2009; accepted November 2009