

A co-training algorithm for multi-view data with applications in data fusion

Mark Culp^{1*}, George Michailidis²

¹Department of Statistics, West Virginia University, Morgantown WV

²Department of Statistics, University of Michigan, Ann Arbor MI

February 14, 2009

Abstract

In several scientific applications, data are generated from two or more diverse sources (views) with the goal of predicting an outcome of interest. Often it is the case that the outcome is not associated with any single view. However, the synergy of all measurements from each view may yield a more predictive classifier. For example, consider a drug discovery application in which individual molecules are described partially by several assay screens based on diverse profiles and partially by their chemical structural fingerprint. A common classification problem is to determine whether the molecule is associated with a disease. In this article, a co-training algorithm is developed to utilize data from diverse sources to predict the common class variable. Novel enhancements for variable importance, robustness to a mislabeled class variable, and a technique to handle unbalanced classes are applied to the motivating data set, highlighting that the approach attains strong performance and provides useful diagnostics for data analytic purposes. In addition, comparisons to a framework with data fusion using PLS are also assessed on real data. An **R** package for performing the proposed approach is provided as supplementary material.

Keywords: multi-view learning, co-training, data fusion, partial least squares, random forests

1 Introduction

The development of a drug from a candidate molecule in a chemical library involves a series of steps and significant costs to assess its effectiveness in treating a disease. It is not uncommon that a drug

*Corresponding author. Email: mculp@stat.wvu.edu, phone: 304 293-3607, fax: 304 293-2272.

shown to be effective in treating a disease, yields a side-effect that is detected years later, which in turn renders it unusable for its original purpose. It is well-established that decisions made early in the drug discovery process may identify these effective compounds with side-effects, so that measures can be taken to reduce the side-effect, or to remove the compound from further consideration so as to incur less overall cost. Hence, analysis of measurements obtained from direct tests on the compound may provide insight about potential side-effects. Measurements about a molecule's chemical structure provide information about its topology necessary for interacting with living organisms (e.g. solubility, permeability, etc.) [1]. Alternatively, measurements obtained from biological screens can consist of assay profiles taken on diverse targets, including receptors, enzymes, and ADME screens taken from specific organisms [2]. Analysis of data consisting of multiple measurement sources (biology and chemistry) requires the combination of novel classification approaches in conjunction with effective data analysis tools to yield informative results that can aid in the drug discovery process.

The above problem is an example of *multi-view learning*, a setting in which the available features are obtained from diverse sources and can be partitioned into disjoint sets (henceforth referred to as views), while the class variable (response) is available (observed) for only a subset m out of the n total observations. The goal is to incorporate all the information available from the views and from all the n observations to accurately predict the missing $n - m$ responses. The rationale behind the term multi-view is that each group of measurements provides its own perspective with respect to the response, and performance improvements lie in the synergy between views. This problem has a strong similarity to data fusion ones, which involve incorporating several disparate groups of measurements (i.e. views) into a common framework for modeling data [3]. From this perspective, multi-view learning can be regarded as an extended case of data fusion. Other applications of multi-view learning occur in the fields of genetics, where the goal is to incorporate microarray data (view 1) and clinical data (view 2) on patients to predict disease [4], chemogenomics, and proteomics [5, 6], combination of spectral values and orthogonal design matrices [7], text and web analysis [8] and computational linguistics [9].

The three general approaches considered in this work for developing a classifier that can predict with multi-view data are: (i) no view distinction, (ii) data fusion with dimensionality reduction, and (iii) data fusion with co-training. The no view distinction approach combines all the views into a single view, but performs rather poorly in the presence of highly heterogenous views, as is the case for the drug discovery application. In the case of heterogenous measurement views, one fairly straightforward approach to combine view data is to first apply a dimensionality reduction technique on each view individually. Commonly used techniques in the chemometrics literature include principal components analysis (PCA), partial least squares (PLS) and variants of the multidimensional scaling such as the Isomap [10, 11]. A

simple approach to combine the reduced dimension features is to fit an additive model with an appropriate link function depending on the type of the response variable (e.g. numerical, binary, count, etc.). On the other hand, the co-training approaches generate a multi-view algorithm from a self-training procedure [8, 9, 12]. The multi-view step typically involves training a classifier directly on each view (within view step), and then the algorithm provides a mechanism to combine predictions across views into one overall estimate (between view step). The available algorithms differ by the classifiers incorporated at the within view step and between view step [13].

In this work, we propose a co-training procedure that iteratively trains a base classifier within each view and then combines the classification results to obtain a single estimate for each observation (refer to Section 2.2). The proposed algorithm is a novel form of co-training, which is more suitable for problems involving both classification and data analysis. In particular, we show that in addition to achieving good performance, the algorithm proves robust to mislabeled responses, can accommodate highly unbalanced classes for the response variable, exhibits global convergence properties under certain conditions and provides a mechanism for assessing importance of each variable. The usefulness of the proposed approach is illustrated on the motivating drug discovery data, where the results regarding overall performance of the classifier, variable importance and robustness to mislabeled responses provide insights on the relationships between compounds and biological and chemical views. Finally, the proposed approach is shown to exhibit significant advantages over a data fusion technique using PLS.

2 Methods

Data fusion methods incorporate data from multiple views with different measurement types into a common framework for classification purposes (or regression). The naive data fusion approach is to combine the information into one large heterogenous data matrix and then fit a classifier directly to it, ignoring view distinction (referred to as no-view-distinction approach). In this section, we consider two alternative data fusion approaches, one based on dimensionality reduction using PLS and the other based on a co-training algorithm with random forests being the underlying classifier.

Formally, the setting is as follows: each observation (compound in the motivating application) is associated with a set of continuous features (view 1) and a set of binary ones (view 2). Specifically, let X_1, X_2, \dots, X_{p_1} denote n -dimensional *continuous* data vectors, i.e. $X_i \in \mathbb{R}^n$ and Z_1, Z_2, \dots, Z_{p_2} n -dimensional *binary* ones; i.e. $Z_i \in \{0, 1\}^n$. Further, define the $n \times p_1$ matrix \mathbf{X} for the continuous data and the $n \times p_2$ matrix \mathbf{Z} for the binary data. The observations in \mathbf{X}, \mathbf{Z} are p_1 and p_2 -dimensional

vectors, denoted by x_i, z_i , respectively. Finally, the response (class) variable Y is assumed to be a binary vector of length n . Class labels are observed only for $|L|$ observations, while they are missing for the remaining $|U| = n - |L|$ ones. Hence, the response Y and the continuous data \mathbf{X} and binary data \mathbf{Z} emit the following partitions:

$$Y = \begin{pmatrix} Y_L \\ Y_U \end{pmatrix}, \mathbf{X} = \begin{pmatrix} \mathbf{X}_L \\ \mathbf{X}_U \end{pmatrix}, \mathbf{Z} = \begin{pmatrix} \mathbf{Z}_L \\ \mathbf{Z}_U \end{pmatrix}.$$

The observations corresponding to L and U are referred to as labeled and unlabeled, respectively.

The primary goal is to develop a classifier ψ trained from data $(Y_L, \mathbf{X}, \mathbf{Z})$ to predict Y . Specifically, the notation $\psi_{\mathbf{X}_L}(x)$ indicates that the classifier was generated from the \mathbf{X}_L data and the class variable Y_L . Further, the classifier can be used to predict any observation x , where $\hat{P}(Y(x) = 1 | X_L)$ and $\hat{P}(Y(x) = 0 | X_L)$ are the corresponding probability class estimates obtained from the classifier. When x_i is an observation from the $L \cup U$ set, we write $\hat{P}(Y_i = 1 | X_L)$ in place of $\hat{P}(Y(x_i) = 1 | X_L)$ to simplify notation. Similar probability class estimates are defined for classifiers generated from \mathbf{Z}_L , or in the case of the proposed co-training \mathbf{X}, \mathbf{Z} .

2.1 Data Fusion with partial least squares

The Partial Least Squares regression is a powerful analysis method developed in chemometrics that can relate a descriptor matrix to a response variable Y_L [14]. It performs dimensionality reduction by favoring dimensions in the direction of high correlation with the response and high variance in the feature data. It can assess variable importance, can implement kernel weights to improve performance, and tends to show robustness to a partially mislabeled class vector [15, 10]. Next, we study the data fusion capabilities of PLS with both no view distinction and fused-PLS. In the case of applying PLS directly to the combined data matrix $(\mathbf{X}_L, \mathbf{Z}_L)$, PLS tends to favor the information available in the continuous variables over that of binary ones in its construction, because of its inherent property of favoring variables with high variation. As a result fitting PLS with no view distinction will not fully utilize the binary \mathbf{Z} information (refer to [7] for discussion on a similar problem).

The dimensionality reduction component of PLS provides an elegant way of incorporating the $(\mathbf{X}_L, \mathbf{Z}_L)$ in a data fusion problem [3, 7]. We refer to this approach as *fused-PLS*. The idea is to apply PLS to the \mathbf{X}_L data directly to obtain reduced data features $\tilde{X}_1, \dots, \tilde{X}_{k_1}$ with $k_1 \leq p_1$. A similar set of features is obtained for Z with dimension $k_2 \leq p_2$. At this time the reduced features are now on the same scale and can be fused together into one matrix $(\tilde{\mathbf{X}}, \tilde{\mathbf{Z}})$. Then one can employ any classification or regression

technique directly on the new fused matrix [16]. In this setting, we have found that generalized additive models work particularly well whenever (k_1, k_2) are sufficiently less than (p_1, p_2) :

$$g(y) = \alpha + \sum_{\ell} f_{\ell}(\tilde{x}_{\ell}) + \sum_{\ell} h_{\ell}(\tilde{z}_{\ell}),$$

where α is the intercept, $g(\cdot)$ is the link function (logistic with a binary response), and $\{f_{\ell}, h_{\ell}\}$ are the functions fit to the reduced features. The additive model allows the procedure to adjust the contribution of each reduced feature relative to all other reduced features with respect to the class variable. In particular, a dimension in \tilde{X} that has little to no value will have $\hat{f}_{\ell} \approx 0$. This is akin to approaches in functional data analysis that first employ a dimensionality reduction technique to obtain a reduced set of features that are in turn used in an additive model [17].

2.2 Co-training with ensemble tree based classifiers

In this subsection we motivate the co-training approach for performing data fusion with ensemble tree based classifiers. We start by providing the necessary details for this type of classifier.

Tree type classifiers partition the feature space into regions, where the majority of the observations come from one of the classes and classify observations accordingly (i.e. to the majority class). A greedy sequential algorithm is used to construct the tree, whose optimal size is selected through cross-validation (refer to [18] for additional details). Extensive empirical work shows that the data driven mechanism of tree construction leads to high variance and moderate bias, which implies that small changes in the data significantly alter the construction of the tree and its resulting classification accuracy.

To overcome this shortcoming, the following modification has proved very successful in practice. Instead of constructing a single tree, one builds a group of them (forest) by applying a random selection mechanism to the data (both to the observations and the features). Specifically, one randomly selects observations with replacement to create many alternative data sets. Further, at each step of tree construction, only a small randomly selected subset of the variables is used to construct the partition. Finally, each observation is predicted by each tree in the ensemble (forest) and its final predicted label corresponds to the majority class. It should be noted that these random features provide speed, robustness and improved accuracy [19]. The averaging of many tree classifiers in random forests allows the procedure to reduce variance of the existing ensemble relative to any single tree in it, a fact following directly from the strong law of large numbers. Further, it has been shown empirically, that constructing trees of maximum sizes leads to reductions in bias as well. It can be seen that generating an ensemble from several large trees

achieves both simultaneous bias and variance reduction relative to a single tree. The reliance of the ensemble on large trees results in producing correlated trees, although they were constructed independently of each other. In fact, there are several examples where the tree-to-tree correlation increases as the size of the ensemble increases, while the classification error remains nearly constant, an important phenomenon pointing to a learning effect as the ensemble grows (discussed extensively in [19]).

As with PLS, the random forests (RF) procedure also allows for the assessment of variable importance. The corresponding mechanism works as follows: a number of observations were left out when constructing each individual tree in the ensemble, the so-called out-of-bag sample in the literature. Then, the constructed tree is used to predict the out-of-bag observations. Further, by permuting the values of each variable one at a time in the out-of-bag sample, new predictions can be obtained and the difference from the original predictions calculated. Averaging these differences over all trees in the ensemble provides a score for the importance of each variable, since the larger this value, the larger the contribution of this variable relative to the others in the data. Negative values and zero values indicate that the variable is not useful for classification purposes.

Ensemble tree based methods lead to highly accurate classifiers, while providing information about variable importance and robustness to mislabeled responses. A shortcoming of such approaches with heterogeneous data is that continuous variables are most often used in tree construction at the expense of binary features. This is illustrated in Figure 1, where for a continuous variable $n_j - 1$ possible splits can be chosen for defining a new smaller data partition from a larger one (n_j is the number of observations in the larger partition), as opposed to a single split for a binary variable independent of n_j . The tree is designed to greedily choose one possible split point from the entire heterogeneous data matrix for each split, which inherently favors the continuous data. The dominance of continuous variables tends to be amplified in ensemble methods due to their construction.

2.2.1 A co-training algorithm

Next, we discuss a novel co-training approach that can handle multi-view data. Algorithmically, the proposed co-training procedure is quite similar to other co-training procedures available in the literature, such as co-SVM [13]. However, a number of novel enhancements have been incorporated that makes the proposed algorithm more suitable for multi-view classification problems. Specifically, (i) we employ a simplified base classification framework which results in global convergence under certain conditions (a property not typical for co-training, refer to Section 3.2.1), (ii) make crucial adjustments for handling unbalanced class labels, (iii) illustrate effective robustness to a partially permuted class variable, and (iv)

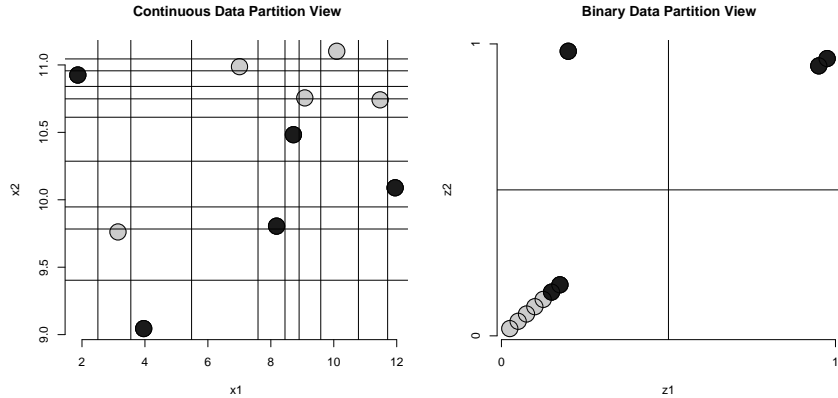


Figure 1: Plot of data consisting of a continuous view of measurements (X) and a binary view of measurements (Z). (left) Illustration that classification tree algorithms provide several opportunities for splitting between two continuous variables. (right) Analogous split result for two binary variables (observations are jittered).

perform variable importance in the multi-view setting.

Next we present the Fitting-the-Fits based co-training (co-FTF) algorithm. The first step consists of training an initial *within view* classifier for both the X measurements (continuous data) and Z measurements (binary data), respectively. For the initialization step, we are only working with the labeled data, i.e. the matrices \mathbf{X}_L , \mathbf{Z}_L and class variable Y_L . The within view classifier can be any that can assign probability class estimates (PCE) to unlabeled observations¹ [16]. The within view classifiers used to predict either an observation x in X or z in Z are denoted by:

$$\{\psi_{\mathbf{X}_L}(x_i), \psi_{\mathbf{Z}_L}(z_i)\}.$$

Recall, that L in the above notation indicates that the classifier was trained or generated using \mathbf{X}_L data with response Y_L or \mathbf{Z}_L data with response Y_L .

Next comes the *between view* step which has two components. First, we identify the view, X or Z , that gives the most confident classification for each observation i , i.e.

$$v_i = \arg \max\{\psi_{\mathbf{X}_L}(x_i), \psi_{\mathbf{Z}_L}(z_i)\},$$

a procedure repeated for all observations in $L \cup U$. Subsequently, a new classifier $\psi_{\mathbf{X},\mathbf{Z}}$ is defined that

¹It is important to note that, theoretically PLS could be applied as a within view classifier; however, our initial experience indicates poor performance (refer to [12] with a PLS-based self-training example).

Algorithm 1 Fitting The Fits Based Co-training (co-FTF_{c*})

- 1: Set $F = L$ and initialize training class variable as $Y_F^{(0)} = Y_L$.
- 2: **for** $k \in \{1, \dots, k^*\}$ **do**
- 3: $Y_L^{(k-1)} = Y_L$
- 4: Within View:
 Fit classifier $\psi_{\mathbf{X}_F}(x_i), \psi_{\mathbf{Z}_F}(z_i)$ to response $Y_F^{(k-1)}$ and data $\mathbf{X}_F, \mathbf{Z}_F$.
- 5: **Set** $F = L \cup U$.
- 6: Between View:
 Apply the classifier on the \mathbf{X} and \mathbf{Z} data to obtain PCEs for each observation in the data. **Compute** $n \times 1$ view vector v such that

$$v_i = \arg \max_{v=\mathbf{X}, \mathbf{Z}} \begin{cases} \hat{P}(Y_i = 1 | v), \hat{P}(Y_i = 0 | v) & : \text{missing}(c^*) \\ \hat{P}(Y_i = 0 | v) & : c^* = 0 \\ \hat{P}(Y_i = 1 | v) & : c^* = 1 \end{cases}.$$

- 7: **Define** classifier $\psi_{\mathbf{X}, \mathbf{Z}}(x_i, z_i)$ in (1) and use it to obtain PCE $\hat{P}^{(k)}(Y_i = 1 | \mathbf{X}, \mathbf{Z})$ for each $i \in L \cup U$.
- 8: **end for**
- 9: **Output** response estimate

$$\hat{Y} = [\hat{Y}'_L, \hat{Y}'_U]' \text{ such that } \hat{Y}_i = \mathbf{1}_{\{\hat{P}(Y_i=1|x_i) \geq 0.5\}}.$$

classifies each observation in accordance with the most confident classification determined by v_i ,

$$\psi_{\mathbf{X}, \mathbf{Z}}(x_i, z_i) = \begin{cases} \psi_{\mathbf{X}_L}(x_i) : v_i \text{ corresponds to source X} \\ \psi_{\mathbf{Z}_L}(z_i) : v_i \text{ corresponds to source Z} \end{cases}. \quad (1)$$

This classifier is applied to all the observations in $L \cup U$ to obtain PCE vectors $\hat{P}^{(0)}(Y = 0 | \mathbf{X}, \mathbf{Z})$ and $\hat{P}^{(0)}(Y = 1 | \mathbf{X}, \mathbf{Z})$. The within view step and between view step results in a fused classifier generated from the X and Z measurements.

The procedure now enters the iterative phase where it enhances classification by incorporating observations for which the response is missing. In this setting, we utilize the fact that the response has a missing component. A *training class variable* of the following form is defined

$$Y^{(1)} = \begin{pmatrix} Y_L \\ \hat{P}^{(0)}(Y_U = 1 | \mathbf{X}, \mathbf{Z}) \end{pmatrix},$$

where the U indicates that the PCEs for the unlabeled data are used for Y_U since Y_U is missing. The procedure then performs the within view step again to generate $\{\psi_{\mathbf{X}}(x_i), \psi_{\mathbf{Z}}(x_i)\}$ and this time the

classifiers were generated with class variable $Y^{(1)}$ and the full data matrices \mathbf{X}, \mathbf{Z} . The between view step is then executed to define a new classifier $\psi_{\mathbf{X}, \mathbf{Z}}$ of the form (1) with \mathbf{X} in place of \mathbf{X}_L and \mathbf{Z} in place of \mathbf{Z}_L . New PCE vectors are then obtained by applying the classifier to the \mathbf{X}, \mathbf{Z} data, denoted by $\hat{P}^{(1)}(Y_i = 0 \mid \mathbf{X}, \mathbf{Z})$, and $\hat{P}^{(1)}(Y_i = 1 \mid \mathbf{X}, \mathbf{Z})$ for each $i \in L \cup U$. This phase is repeated for $k = 2, \dots, k^*$ times using training class variables $Y^{(2)}, \dots, Y^{(k)}$. Upon exiting the algorithm (after k^* iterations), the response \hat{Y} is constructed from the PCEs, which provides an estimated class for each observation in the data. In addition, the final classifier $\phi_{\mathbf{X}, \mathbf{Z}}(x_i, z_i)$ can be used to directly predict new observations unavailable at the time of training. This algorithm is referred to as co-FTF.

The co-FTF algorithm can be adapted to better account for situations in which the classes are heavily unbalanced, i.e. the proportion of classes in Y_L heavily favor one class over the other. In our experience, classifiers tend to predict observations as the dominant class with high confidence in these situations. For example, suppose that class zero is heavily favored over class one. For convenience we write the PCEs of the within-view classifier at step k of co-FTF for observation (x_i, z_i) as $\hat{P}(Y_i = 0 \mid \mathbf{X}), \hat{P}(Y_i = 1 \mid \mathbf{X}), \hat{P}(Y_i = 0 \mid \mathbf{Z}), \hat{P}(Y_i = 1 \mid \mathbf{Z})$. Suppose that the following situation holds for these PCEs:

$$\begin{aligned} \hat{P}(Y_i = 0 \mid \mathbf{X}) &> \hat{P}(Y_i = 1 \mid \mathbf{X}) \\ \hat{P}(Y_i = 0 \mid \mathbf{Z}) &< \hat{P}(Y_i = 1 \mid \mathbf{Z}) \\ \hat{P}(Y_i = 0 \mid \mathbf{X}) &> \hat{P}(Y_i = 1 \mid \mathbf{Z}) \end{aligned}$$

In this situation, the fused classifier from co-FTF would yield, $\psi_{\mathbf{X}, \mathbf{Z}}(x_i, z_i) = \psi_{\mathbf{X}}(x_i)$ and ignore the classification to class one by the binary Z information. However, it is often more desirable to retain classification of class one over that of the highly favored classification to class zero. To correct for this, we introduce a new parameter in co-FTF algorithm, which allows the user to specify class c^* ($c^* = 1$ in the example), so that the source with the maximum PCE for class c^* is selected. Therefore, the co-FTF algorithm can be set to favor class zero ($c^* = 0$), to favor class one ($c^* = 1$), and to favor neither class ($c^* = NA$). The compact co-FTF procedure with input parameter k^* (number of iterations) and parameter c^* is presented in Algorithm 1.

As with RF and PLS, the proposed co-FTF can also produce variable importance results whenever RF are used as the within view classifiers. The variable importance results are obtained from the original within view random forests. To incorporate this variable importance measure with co-FTF first take each observation in $L \cup U$, let i_k be the view indicator such that $i_k = 1$ if the max PCE of observation k is from view one, and similarly for $i_k = 2$. Then define the variable importance measure for variable j in

view v as:

$$\text{Variable Score}_j = \sum_{k=1}^{LUU} S_{kj}(X_v) \mathbf{1}_{\{i_k=v\}}$$

where $S_{kj}(X_v)$ is the variable score for observation k . To obtain a variable ranking from the co-FTF algorithm, we concatenate the variable scores and sort them with high positive values being most important.

Notice that one useful aspect of co-FTF is that the algorithm distinguishes between the different views of the data when training the classifier. As a result, the within view classifier can be tailored to the structure of the specific measurement type in the view. The user is free to select the within view classifier that works best for the data within that view. In Section 3.2.1 we discuss computational issues and parameter estimation involving co-FTF.

3 Results

In this section, we illustrate the proposed methodology on the drug discovery problem discussed in the introduction section. The data set contains 438 compounds that are known to be effective in treating a particular disease. In addition, it is also known for each compound whether it exhibits a certain adverse effect (AE). For the data at hand, the AE corresponds to a disease of a major organ (details are confidential); 92 compounds had an AE, while the remaining 346 did not. It should be noted that presence or absence of this particular AE has significant implications for the drug’s success in treating the original disease. New compounds are being studied with the goal of reducing the AE, while maintaining their effectiveness against the original disease. However, direct detection of the AE for these new compounds can take large amounts of time and money, while classification of AE status using standard biological and chemical measurements on the compound is significantly faster and less costly. The trade-off is that classification results exhibited a certain degree of uncertainty; hence, high quality classification techniques are very desirable for the problem at hand.

The biological measurements contain information about the necessary dose to achieve 50% inhibition against a receptor, e.g. liver cells, kidney cells, proteins, etc. There are 191 receptors denoted by X_1, \dots, X_{191} [20, 2]. The chemical measurements provide an indicator (153 of them in total) of a specific substructure in the compound [1]. Therefore, the available data can be represented by a 438×344 matrix, whose columns can be partitioned into two subsets; the first containing the 191 continuous bio-

Table 1: Average percent decrease (s.d.) in testing accuracy over 50 samples due to permutation of the class variable (higher value indicates a larger deterioration relative to the case of no permutation).

	%-Labeled	Biology	Chemistry
Random Forest	20	-0.06 (0.11)	2.36 (0.26)
	80	0.04 (0.23)	0.81 (0.59)
Partial Least Squares	20	0.70 (0.27)	0.62 (0.26)
	80	0.00 (0.00)	0.14 (0.30)

logical measurements (denoted by X) and the second the 153 binary chemical measurements (denoted by Z).² The goal is to construct a classifier that accurately predicts whether a compound exhibits an AE.

We first classify the compounds by using PLS and RF on the biology and chemistry data separately. The goal is to assess the performance, variable importance, and robustness to a mislabeled class variable for each classifier on these data. This will also help us determine which view is expected to contribute the most to the problem at hand. Next, a combined analysis of the (X_L, Z_L) data using both PLS and RF was performed. Finally, the data fusion version of PLS and the co-FTF algorithm discussed in the methods section was used. Comparative results for assessing performance and robustness are shown for all methods. However, variable importance is not currently available for PLS based data fusion, but is available for the remaining methods. Computational issues of the proposed co-FTF approach are also discussed.

3.1 Separate analysis of the biology and chemistry views

We start by using PLS and RF on the two views separately. Both methods are applied using the default settings available in the respective **R** packages [14, 21]. The decision rule for both techniques is determined by $I\{\hat{P} > 0.5\}$, where \hat{P} is the estimated PCE vector. For RF, the classification accuracy using the full biology data is 0.81 for 500 trees and using the full chemistry data is 0.83 for 500 trees. In the case of PLS, the number of dimensions, d , was estimated as 2 via 10-fold cross-validation for both data sets with the classification accuracy being 0.82 for the biology data and 0.84 for the chemistry view. Classifying all observations as class zero resulted in the prior of 0.79 indicating that both classifiers were better than guessing for each data set.

It is of interest to assess these techniques on independent testing data sets. To this end, we employ

²The data are available in blinded format in the cftf **R** package as supplemental material to this document.

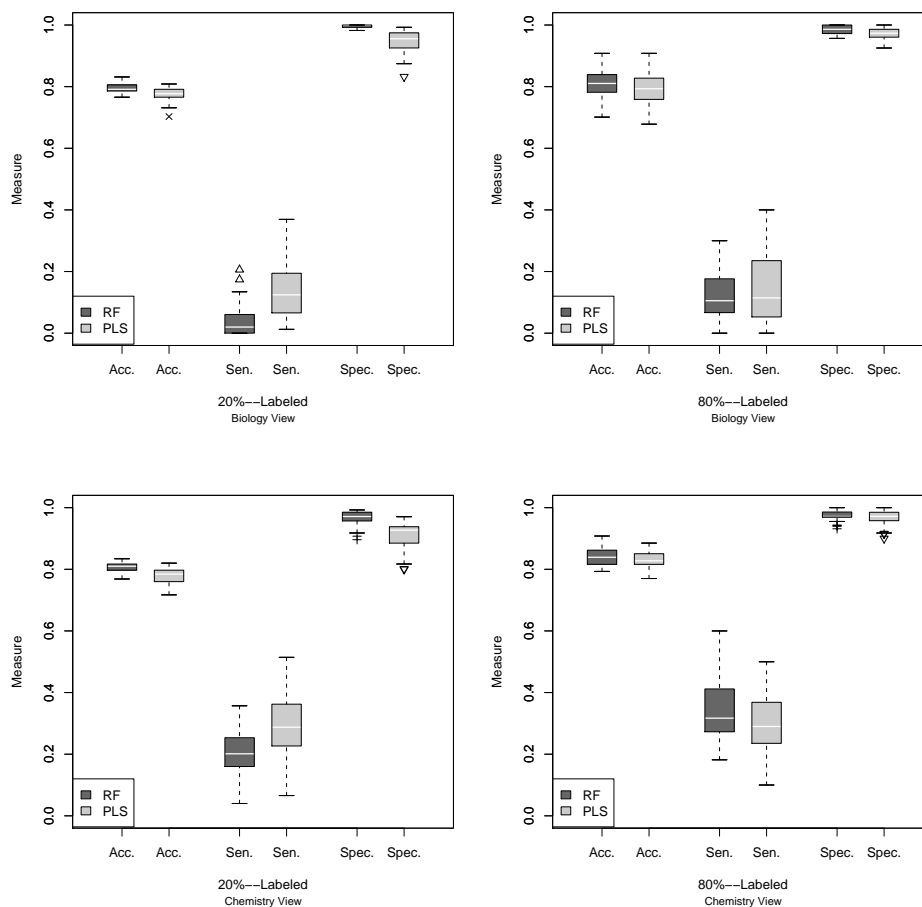


Figure 2: Accuracy, sensitivity, and specificity results for RF and PLS on the biology (top row) and chemistry (bottom row) data sets. The first column provides the results for 20% used in training with 80% used for testing and vice-versa for the second column.

a standard approach, where the data are partitioned into a labeled (training) set that treats $P\%$ of the observations as labeled data and the remaining $(100 - P)\%$ as unlabeled (test set); the percentage was set to $P = \{20, 80\}$. The accuracy, sensitivity and specificity measures were averaged over 50 randomly selected labeled/unlabeled data sets for each P . The results based on the testing data are shown in Figure 2. The accuracy results show little difference between the RF and PLS classifiers. For the biology data, notice that PLS results in a higher sensitivity value than RF, especially in the 20%-labeled case. This suggests that PLS is classifying more compounds as being associated with AE. The same is true for the PLS on the chemistry data in the 20%-labeled case. Also, sensitivity values are higher in general for both classifiers when using the chemistry data instead of the biology data.

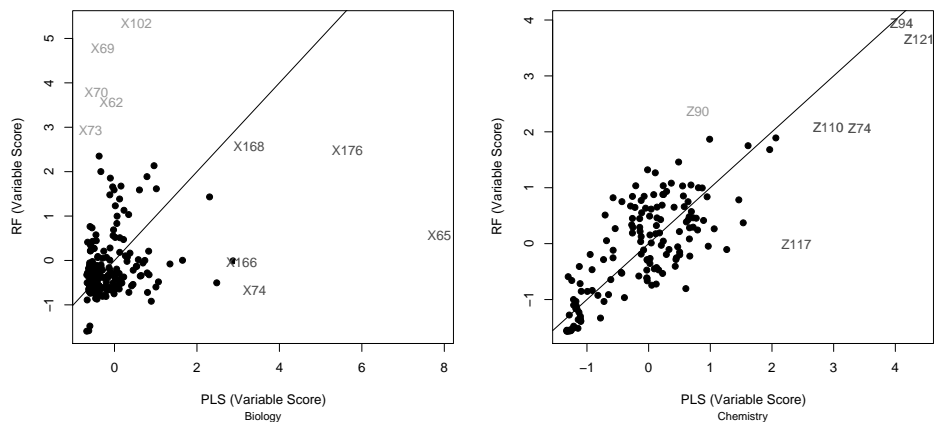


Figure 3: The normalized variable importance scores plotted with RF against PLS for the biology (left) and chemistry (right) data sets. The top five variables from each procedure are marked on the plot.

The AE class variable indicates a side-effect associated with a drug, which may be difficult to ascertain exactly; e.g. a side-effect related to the function of the gall bladder is hard to detect. As a result, we are most interested in classifiers that are robust to a somewhat mislabeled class vector, and the next experiment addresses this key issue with each data set separately. To this effect, we permuted the classes of the AE variable for 5% of the observations in the labeled data. The permutation was performed using stratified sampling so that the proportion of AE cases and non-AE cases flipped would remain constant in each partition (i.e. about three of every four compounds are from the non-AE group). The percent decrease in testing accuracy from the base case of no class permutation to the 5%-permuted case are reported in Table 1 for both 20%-labeled and 80%-labeled configurations. From this table, none of the values of percent decrease in testing accuracy reach an alarming magnitude, which indicates that both classifiers show robustness to noise. It is important to note that for some classifiers (not shown) we observed values as high as 20-30% decrease, which is highly undesirable (refer to [19] for more on this).

Next we consider the variable importance results for both RF and PLS on the biology and chemistry data separately. The interest is in the relationship between the variable scores from these different classifiers. As a rule of thumb we postulate that if two separate classifiers agree on the top variables then it provides evidence of a true underlying importance. On the other hand, if there is disagreement, it is unclear on what to conclude about the true importance of the variables. For example, disagreement can be explained by variable correlation, or simply by different features in the data that stand out separately for each classifier. In Figure 3, we provide a scatterplot of the normalized variable importance scores (mean zero, unit variance) from each procedure for the biology data (left panel) and chemistry data (right

panel). Clearly, the closer the values are to the line $y = x$ the more similar the scores. The correlation between the RF and PLS score vectors for the biology data is 0.23 and for the chemistry data is 0.80. From these results, PLS and RF agree quite strongly with respect to important variables in the chemistry view and suggest different variables as important for the biology view. Pairwise correlations calculated between the top biology variables reveal that the largest absolute correlation between variables chosen by RF and chosen by PLS is 0.31 corresponding to the pair X_{62} and X_{176} . This result strongly indicates that the two techniques regard different sets of variables as important for classification purposes.

The evidence from the separate analysis of each data set indicates that both RF and PLS perform similarly on these data, are robust to noise in the class variable, and provide useful variable importance diagnostics. Next we investigate combining these data sets with the intent of improving the above results.

3.2 Combined analysis of the biology and chemistry views

The naive approach for combining the views is to create one large data matrix (X_L, Z_L) and then fit PLS and RF directly to it. We refer to these as the no-view-distinction classification approaches, denoted by PLS(NVD) and RF(NVD), respectively. In the first experiment, Figure 4 gives the accuracy and sensitivity values for these approaches using $P = \{20, 80\}$. The accuracy for PLS(NVD) was slightly lower than that of RF(NVD) in both the 20%-labeled and 80%-labeled configurations. The sensitivity for PLS(NVD) was somewhat higher than that of RF(NVD) in the 20%-labeled configuration and slightly lower in the 80%-labeled configuration. A more informative comparison can be made with the results obtained from the separate analyses. The median sensitivity from both Figure 2 and Figure 4 are recorded in the following table for convenience:

Table 2: Median sensitivity from the results in Figure 2 and Figure 4.

	PLS (X)	RF(X)	PLS(Z)	RF(Z)	PLS(NVD)	RF(NVD)	Fused-PLS	co-FTF
20%	0.124	0.020	0.288	0.201	0.118	0.041	0.248	0.271
80%	0.114	0.105	0.290	0.317	0.118	0.127	0.218	0.429

It can be seen that neither PLS(NVD) nor RF(NVD) are appropriately combining the two views, since the sensitivity values for PLS(NVD) and RF(NVD) are very similar to those for PLS(X) and RF(X), respectively. So the \mathbf{X} data (biology view) are overwhelming the \mathbf{Z} data (chemistry view) in the NVD approaches.

To address this, PLS naturally extends to data fusion problems by first fitting PLS separately on

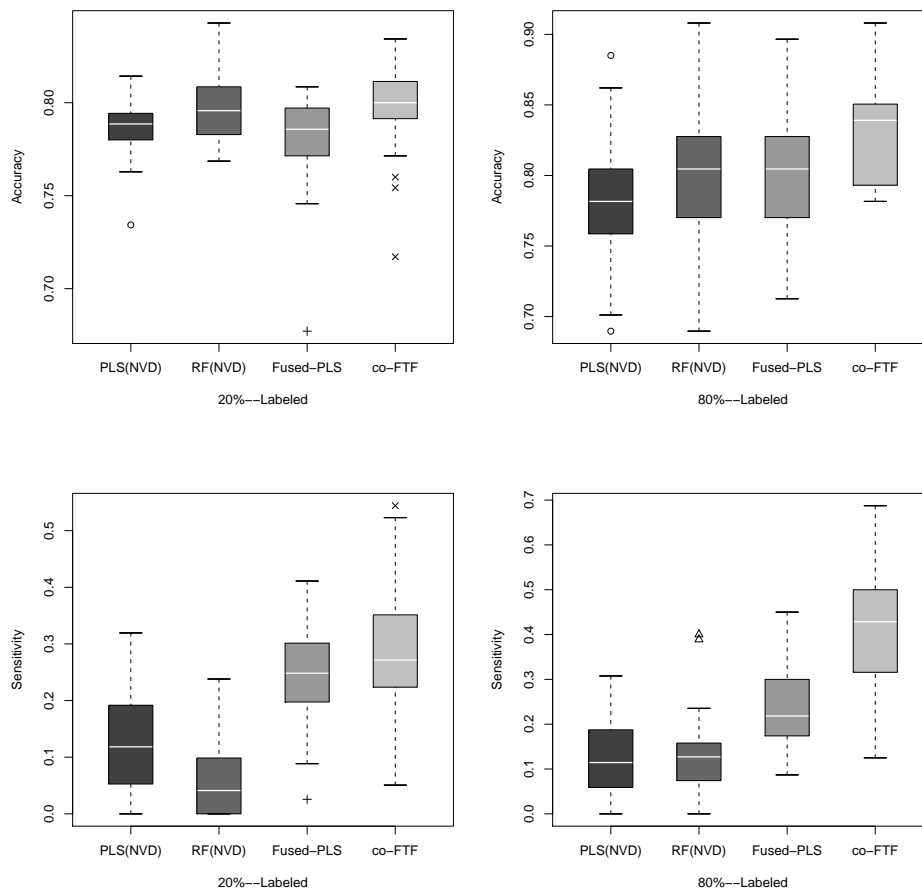


Figure 4: Accuracy and sensitivity results for RF(NVD), PLS(NVD), Fused-PLS and co-FTF on the drug discovery data. The first column provides the results for 20% used in training with 80% used as testing and vice-versa for the second column.

the biology and chemistry data and then fitting a logistic additive model to the combined reduced data dimensions as discussed in the methodology section. Both the internal PLS and logistic model parameters can be estimated by cross-validation.³ The co-FTF algorithm proposed in this work is another example of data fusion in which classifiers are iteratively fitted within views and between views. The k^* parameter that controls the number of iterations was set to 5, while the c^* that indicates that a prediction of a compound for the AE is favored at the between view step is set to 1 (refer to Section 3.2.1).

The accuracy and sensitivity results for fused-PLS and co-FTF are shown in Figure 4 and the median sensitivities are summarized in Table 2. The fused-PLS sensitivity result shows significant improvement

³To speed-up computation of PLS we noticed that $d = 2$ dimensions was consistently chosen by 10-fold cross-validation in each example and that setting was used directly.

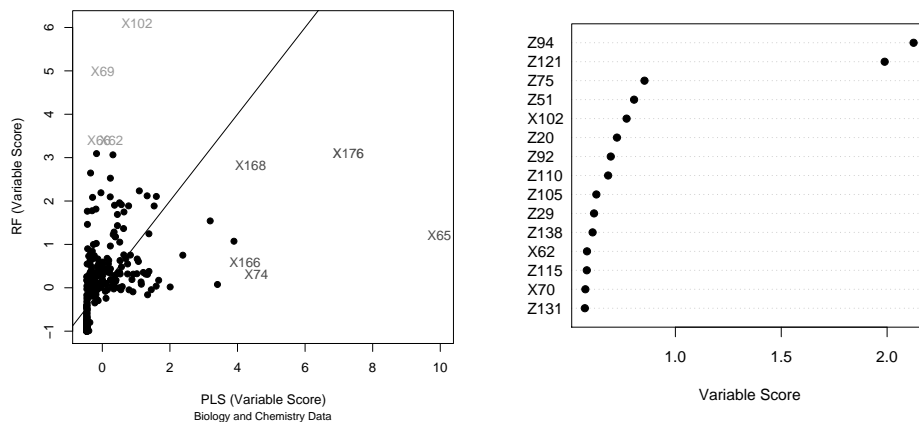


Figure 5: (left) The normalized variable importance scores plotted with RF(NVD) against PLS(NVD) for the drug discovery data. (right) The top fifteen variables selected by the co-FTF procedure.

over the PLS(NVD) approach, but does not exhibit any significant improvement over the PLS approach utilizing the chemistry view. On the other hand, the co-FTF procedure showed improvement over all other RF techniques in the 20%-labeled and 80%-labeled configurations. However, it is noteworthy that the median sensitivity for PLS using the chemistry view and with $P = 20$ has a similar performance to co-FTF, which makes this result difficult to interpret. In the 80%-labeled configuration the median for co-FTF outperforms all other medians by a factor of 10%; in addition, the median accuracy outperforms all other ones by a factor of about 2% for the 80%-labeled configuration. A performance improvement of this magnitude on the class variable as difficult to classify as AE status illustrates that co-FTF can utilize the information available in the biology and chemistry views appropriately, to obtain improvements with a large enough labeled set.

Next, we examine the robustness of the data fusion approaches to mislabeled responses. As before, we permute 5% of the labeled class values in both the 20%-labeled and 80%-labeled configurations and compute the percent decrease in accuracy. The results are given in Table 3. It can be seen that both fused-PLS and co-FTF preserve the robustness qualities noted earlier.

Table 3: Average percent decrease (s.d.) in testing accuracy over 50 samples due to permutation of the class variable.

%-Labeled	Fused-PLS	co-FTF
20	2.00 (0.36)	1.75(0.34)
80	1.56 (0.39)	2.22(0.41)

Table 4: The c^* Parameter.

%-Labeled	Measure	$c^* = 1$	$c^* = 0$	$c^* = NA$
20	Acc.	0.799 (0.003)	0.787 (0.001)	0.795 (0.002)
80	Acc.	0.829 (0.005)	0.797 (0.007)	0.820 (0.006)
20	Sen.	0.282 (0.015)	0.000 (0.000)	0.051 (0.008)
80	Sen.	0.424 (0.017)	0.034 (0.006)	0.186 (0.011)

Next, we consider variable importance for RF(NVD), PLS(NVD) and co-FTF. To the best of our knowledge it is currently unknown how to assess variable importance with fused-PLS. The details for computing the variable importance scores for each technique are given in the methods section. We first consider the scatterplot of the normalized variable scores from RF(NVD) and PLS(NVD) shown in Figure 5. The correlation between the two variable importance score vectors was 0.41, and the top pairwise variable correlation magnitude remains 0.31 for the pair X_{62} and X_{176} . Notice that none of the chemistry variables are among the top important variables for PLS(NVD) and RF(NVD). From this result, it seems that the no-view-distinction approach is not utilizing the chemistry information appropriately due to the dominance of the continuous biology data, as discussed above. The variable importance results for co-FTF are shown in the right panel of Figure 5. The top variables correspond to several ones seen in the separate analyses. The results indicate a successful integration of information from both views, with no particular preferences shown towards continuous variables.

3.2.1 Computational issues

Next, we address computational issues associated with the co-FTF procedure, including considerations for parameters (c^*, k^*) and execution time. The *cFTF* **R** package is available as supplementary material to this document and the data can be accessed by executing *data(pharm)* at the **R** prompt after loading the library. The package utilizes both support vector machines (SVM) and random forest (RF) as the underlying classifiers (for more on SVM and its applications in chemometrics refer to [18]).

The parameter c^* allows the procedure to favor a particular class at the between view step. The parameter c^* can take values in the set $\{0, 1, NA\}$, where zero indicates that the no-AE class is favored, one indicates that the AE class is favored and NA indicates that neither class is favored. Table 4 provides the average accuracy and sensitivity over 50 samples for c^* with $k^* = 5$ and $P = \{20, 80\}$. The accuracy and sensitivity results in the table show that the algorithm using $c^* = 1$ outperforms the other parameter settings. Interestingly, the $c^* = NA$ case is fairly close to the fused-PLS results above, providing no real

Table 5: Convergence Results for SVM and RF.

k	SVM				RF			
	Acc.	Sen.	Crit.	Time (Min)	Acc.	Sen.	Crit.	Time (Min)
1	0.831	0.500	0.289710	0.031	0.845	0.778	0.000744927	0.670
2	0.831	0.500	0.000351	0.062	0.849	0.720	0.000279943	1.277
3	0.836	0.556	0.000065	0.092	0.845	0.692	0.000232385	1.888
4	0.836	0.556	0.000036	0.120	0.845	0.667	0.000224955	2.481
5	0.836	0.556	0.000013	0.149	0.849	0.677	0.000192325	3.072
10	0.836	0.556	0.000000	0.295	0.822	0.568	0.000169396	5.984
15	0.836	0.556	0.000000	0.443	0.836	0.618	0.000159174	8.868
20	0.836	0.556	0.000000	0.588	0.817	0.553	0.000173309	12.075
25	0.836	0.556	0.000000	0.733	0.813	0.538	0.000154642	15.044

advantage over fitting the RF directly to the chemistry data. The latter result indicates that setting $c^* = 1$ allows co-FTF to more aggressively combine the biology and chemistry information, which can result in improved performance.

The next experiment is designed to address choice of the k^* parameter, with respect to execution time and convergence. The time the algorithm runs depends significantly on the execution time for the base classifier. It is worth noting that after the initialization step this time depends on $n = |L \cup U|$, but does not depend on $|L|$ and $|U|$ individually. For the drug discovery data, we employ a typical execution of the co-FTF algorithm with both RF and SVM for 25 iterations with $c^* = 1$ and with 50%-labeled data. The SVM results are provided as an example in which co-FTF converges globally, which is in sharp contrast with executing co-FTF with RF as the base classifier. The iteration number, accuracy, sensitivity, convergence criteria ($\|\hat{P}^{(k)} - \hat{P}^{(k-1)}\|$) and time from the function call are each recorded in Table 5. Notice that time increases linearly as the number of iterations increases for both classifiers. Further, co-FTF using SVM as the base classifier is much faster than co-FTF with RF. The co-FTF algorithm achieves global convergence for the SVM, that is k^* is chosen so that $\|\hat{P}^{(k)} - \hat{P}^{(k-1)}\| < \delta$ for some tolerance $\delta > 0$ (formal conditions for convergence of co-FTF with SVMs are under consideration). For RF, the co-FTF procedure does not achieve global convergence. Estimation of the parameter k^* is currently unknown for co-training in general, but in our experience choosing this parameter between 2 and 5 iterations typically works well. However, caution must be exercised when choosing k^* , since performance almost always deteriorates after a certain number of iterations for RF as observed in the above table.

4 Conclusions

The main contribution of this work is the proposed co-FTF framework which features (i) a data fusion component to combine multiple views, (ii) incorporation of unlabeled data in classification, (iii) flexibility for employing arbitrary classifiers within each view, (iv) considerations for handling an unbalanced class variable, and (v) the ability to assess variable importance. The comparison between co-FTF and data fusion with PLS illuminates future directions that can improve each approach considerably, such as variable importance for fused-PLS, computational enhancements to speed up the procedures, observation diagnostics and dimensionality reduction in the multi-view setting, and better usage of unlabeled data in classification. Both procedures were assessed on real data motivated from drug discovery problems, but each one is applicable to any multi-view data set with minor modifications.

References

- [1] A. Leach and V. Gillet. *An Introduction to Chemoinformatics*. Kluwer Academic Publishers, London, 2003.
- [2] R. Lundblad. *Chemical Reagents for Protein Modification*. CRC Press Inc., 2004.
- [3] H. Mitchell. *Multi-sensor Data Fusion An Introduction*. Springer-Verlag, Berlin, 2007.
- [4] J. Jansen, R. Bro, H. Hoefsloot, F. Van den Berg, J. Westerhuis, and A. Smilde. Parafasca: Asca combined with parafac for the analysis of metabolic fingerprinting data. *J. Chemometrics*, 22(2):114–121, 2008.
- [5] Y. Yamanishi, J. Vert, and M. Kanehisa. Protein network inference from multiple genomic data: A supervised approach. *Bioinformatics*, 20(1):363–370, 2004.
- [6] M. Daszykowski, W. Wu, A. Nicholls, R. Ball, T. Czekaj, and B. Walczak. Identifying potential biomarkers in lc-ms data. *J. Chemometrics*, 21(7):292–302, 2007.
- [7] K. Jorgensen, V. Segtnan, and T. Naes. A comparison of methods for analysing regression models with both spectral and designed variables. *J. Chemometrics*, 18:451–464, 2004.
- [8] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. *Computational Learning Theory*, pages 92–100, 1998.

- [9] S. Abney. Understanding the yarowsky algorithm. *Computational Linguistics*, 30(3):365–395, 2004.
- [10] S. De Jong and C. Ter Braak. Comments on the pls kernel algorithm. *J. Chemometrics*, 8:169–174, 1994.
- [11] E. Levina, A. Wagaman, A. Callender, G. Mandair, and M. Morris. Estimating the number of pure chemical components in a mixture by maximum likelihood. *J. Chemometrics*, 21:24–34, 2007.
- [12] M. Culp and G. Michailidis. An iterative algorithm for extending learners to a semi-supervised setting. *J. Comp. and Graph. Statis.*, 17(3):545–571, 2008.
- [13] U. Brefeld and T. Scheffer. Co-em support vector learning. *International Conference on Machine Learning*, 21:16, 2004.
- [14] B. Mevik and R. Wehrens. The pls package: Principal component and partial least squares regression in r. *J. Statist. Soft.*, 18(2), 2007.
- [15] R. Leardi. Application of genetic algorithm-pls for feature selection in spectral data sets. *J. Chemometrics*, 14(5):643–655, 2000.
- [16] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning (Data Mining, Inference and Prediction)*. Springer Verlag, 2001.
- [17] J. Faraway. Modeling continuous shape change for facial animation. *Statis. and Comp.*, 14:357–363, 2004.
- [18] S. Caetano, B. Ustun, S. Hennessy, J. Smeyers-Verbeke, W. Melssen, G. Downey, L. Buydens, and Y. Heyden. Geographical classification of olive oils by the application of cart and svm to their ft-ir. *J. Chemometrics*, 21:324–334, 2007.
- [19] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
- [20] W. Hunter. Rational drug design: a multidisciplinary approach. *Mol. Med. Today*, 1(1):31–34, 1995.
- [21] A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.