Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 36 (2014) 470 – 475

Procedia Computer Science

Complex Adaptive Systems, Publication 4
Cihan H. Dagli, Editor in Chief
Conference Organized by Missouri University of Science and Technology
2014- Philadelphia, PA

A Latent Space Support Vector Machine (LSSVM) Model for Cancer Prognosis

William Ford[a*], Walker Land[a]

[a]Binghamton University, 4400 Vestal Parkway East, Binghamton, NY 13902, USA

Abstract

Gene expression microarray analysis is a rapid, low cost method of analyzing gene expression profiles for cancer prognosis/diagnosis. Microarray data generated from oncological studies typically contain thousands of expression values with few cases. Traditional regression and classification methods require first reducing the number of dimensions via statistical or heuristic methods. Partial Least Squares (PLS) is a dimensionality reduction method that builds a least squares regression model in a reduced dimensional space. It is well known that Support Vector Machines (SVM) outperform least squares regression models. In this study, we replace the PLS least squares model with a SVM model in the PLS reduced dimensional space. To verify our method, we build upon our previous work with a publicly available data set from the Gene Expression Omnibus database containing gene expression levels, clinical data, and survival times for patients with non-small cell lung carcinoma. Using 5-fold cross validation, and Receiver Operating Characteristic (ROC) analysis, we show a comparison of classifier performance between the traditional PLS model and the PLS/SVM hybrid. Our results show that replacing least squares regression with SVM, we increase the quality of the model as measured by the area under the ROC curve.
© 2014 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/3.0/).
Peer-review under responsibility of scientific committee of Missouri University of Science and Technology

Keywords: Machine Learning; Partial Least Squares; Support Vector Machine; Microarray; Lung Cancer

1. Introduction

Having a predictive model for cancer patient prognosis can be helpful in directing the course of treatment, whether this is the initial course of treatment or follow-on. Cancer treatments such as radiation and chemotherapy can be harmful to the patient. Therefore, the treatment should only be as aggressive as necessary, but no less aggressive than necessary to insure the highest efficacy and best outcome. If the prognosis of an individual patient is known a priori, the treatment can be tailored such that the patient will have the best chance for survival while suffering as few of the toxic side effects of these treatments as possible.

Gene expression microarray data provides a method of quickly generating a genetic profile of the malignant tissue by quantifying the expression levels of tens of thousands of genes in parallel from a single biological sample. Collecting genetic samples from many patients and then measuring the actual survival time of the patient can provide information

* Corresponding author.
E-mail address: wford2@binghamton.edu

1877-0509 © 2014 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/3.0/).
Peer-review under responsibility of scientific committee of Missouri University of Science and Technology
doi:10.1016/j.procs.2014.09.023

about the link between the expression of certain genes and the rate of disease progression. This information can then be used to build a predictive model capable of providing a prognosis for an individual patient, aiding the physician in tailoring the treatment, thus maximizing the efficacy of the treatment while minimizing the toxic side effects.

In this paper, we build upon previous work [1] and use those prior results as a benchmark to compare against. No other comparable survivability model studies were found using this particular data set. Our prior work showed that partial least squares, in its original formulation, can be used to create a linear prognostic model in a latent space by simultaneously reducing the dimensionality of the original space and building a linear least squares regression model in latent space. Our new approach improves on this by decoupling the latent space creation from the linear regression model. Instead of creating a linear regression model in latent space, we create a support vector machine (SVM) in latent space. Thus, our hypothesis is: *replacing the least squares regression in the PLS model with a support vector machine will increase the model quality as indicated by the area under the receiver operating characteristic (ROC) curve* [7]. We call this model a Latent Space Support Vector Machine (LSSVM).

2. Data Set

To validate the LSSVM as a prognostic tool, we use the same data set we used previously to validate a partial least squares predictive model [1]. The data set consists of 22,283 (Affymetric U133A) gene expression values and clinical data (age, sex, T stage) from 442 patients diagnosed with lung adenocarcinomas [2]. Processing of the raw U133A CEL files are summarized from the primary literature referenced in [1].

Clinical covariates were encoded and appended to the microarray data to complete the data set as follows. The age of the patient in years (at time of diagnosis) was included in the data as is. The gender of the patient was encoded as -1.0 for female and +1.0 for male. Pathologic T-Stage was encoded using 1.0, 2.0, 3.0, and 4.0 to T1, T2, T3, and T4 respectively. The combined microarray probe values, encoded clinical data, and the reduced or processed set thereof are hereinafter referred to as 'features'.

The dependent variable of interest we used was time or death or last contact in months. Since we were interested in classifying a case as 'poor' or 'good' prognosis, the months until death or last contact (after diagnosis) were replaced with a class value of -1.0 (poor) for values less than 60 month and +1.0 (good) for values 60 months or greater.

3. Methods

To construct a prognostic LSSVM classifier from the microarray and clinical data, we constructed a data processing pipeline. Our data processing pipeline consisted of four stages: 1) Data encoding and quality control, 2) Coarse Feature Reduction using standard statistical methods (t-test, variance pruning), 3) Latent Feature Discovery/Optimization of classification model (LSSVM), and finally 4) Validation/Quantification of the quality of the model (ROC AUC metric).

Latent Feature Discovery processing consisted of building a Partial Least Squares model of the remaining probes. However, we deviate from the practice of using the PLS regression model directly. Instead, we decouple the linear least squares regression model portion of PLS from the data transformation. The goal is to discover latent features in the remaining probes and remove as much covariance as possible, then use a support vector machine classification model to perform the final class prediction. It is widely accepted that the SVM structural risk mitigation model provides better generalization performance than the linear least squares empirical risk minimization model.

Once the latent features have been extracted using the PLS transformation matrix, these few remaining features are used to train the SVM model. Based on the success of linear models from our prior study [1], the SVM kernel we used was the linear (dot product) kernel. The optimal SVM parameter, C, was selected using differential evolution (DE) [5]. Although the optimization of a single tunable parameter could have simply been derived using a sensitivity analysis, we chose to include a stochastic optimization method in the processing pipeline for the sake of scalability to larger parameter set optimization when using non-linear kernels.

All data processing algorithms were implemented by us using Python, NumPy, and SciPy, except for ROC analysis (PlotROC) and SVM (LibSVM) which were downloaded from [3].

3.1. Data Pre-processing and Quality Control

After encoding the clinical features and prognosis class (section 2), and including them in the data set, the data were inspected and some problems were discovered. Probe 222086_s_at was missing for cases Moff 1835I, Moff 2362A, and Moff 3009D. These were set to the mean expression value of the remaining cases. All values for probe 207140_at were 'NA', so this probe was removed entirely. An incredibly large expression value (7841550) was found for case (NCI_lung221_U133A) probe (206014_at) and was changed to the mean expression value (5.351) of the remaining cases.

3.2. Coarse Feature Reduction

Coarse feature reduction is necessary to remove as much of the irrelevant data, or noise, from the data as possible. Therefore, we initially reduce the number of features by discarding approximately 99% of the data which is statistically unlikely to contain information useful for classification. Although the Latent Feature Discovery selects signal from noise by discarding features that show low co-variation with the dependent variable, we found in our preliminary experiments that applying PLS to the entire 22,000+ feature data set resulted in a less accurate predictive model.

Coarse feature reduction was achieved by splitting the data into good and poor prognosis groups based on the time of death (60 month threshold). A two sample t-test (unequal variance assumed) was performed for each probe to remove probes with identical means. For each probe, a p-value is computed from a t-statistic and used to reject the null hypothesis that the mean feature values for each class are identical. Features with a p-value less than the cutoff value were selected for further processing in the next pipeline stage - variance pruning.

Variance pruning was used on the remaining features to remove features with low relative variance, and hence little information. From the probes passing the t-test stage, the coefficient of variation was computed. Those features with a coefficient of variation greater than the cutoff value were selected for Latent Feature Discovery (LFD). The final set of features selected for LFD were then normalized to zero mean and unit variance.

3.3. Partial Least Squares

Partial least squares is similar to principal components analysis (PCA) except for the way latent variable (PLS) or principal component (PCA) vectors are computed. PCA uses the direction of maximum variance in input space (X, independent variables, features, etc.) as the principal components, while ignoring the output data (Y, dependent variables, etc.) which is to be modeled. PLS, on the other hand, uses the covariance between X and Y and selects the direction of maximum covariance between X and Y as the latent variable vector. PLS is a constructive process which successively computes new latent variables, projects the data into the subspace orthogonal to the latent variable, repeating as up to d times where d is the dimensionality of the input space (or rank, if the matrix is not full rank), thus 'deflating' one dimension at a time from the input space and 'inflating' one dimension at a time in latent space. Each successive dimension is 'deflated' from the input space, creating a lower dimensional manifold embedded in input space.

In the processing of successively finding latent variables and 'deflating' the input space, a series of orthogonal weight vectors are created and normalized, creating a new orthonormal basis defining the latent space. From this weight matrix, a linear transformation matrix is derived and used to project data from input space to latent space. To derive the PLS weight vector, we used the PLS1 algorithm [8].

Here the superscript *m* denotes the current latent variable being computed, and m+1 denotes the value to be used in the next iteration of the loop. When the loop has completed (e.g. the desired number of latent variables have been computed), the weight matrix *W* is formed by horizontally stacking each basis vector:

$$W = \left[w^1 \middle| w^2 \middle| \ldots \middle| w^n \right] \quad (1)$$

This weight vector w is the new orthonormal basis for the latent space and is used to create a transformation matrix (eq. 2) for projecting data from input space to latent space [4].

$$W^* = W\left(P^T W\right)^{-1} \quad (2)$$

In the traditional PLS process, once a number of latent variables has been selected, and the latent space is iteratively constructed as shown above, a least squares regression model is constructed in latent space. The regression coefficients *in input space* are computed for the model and the model consists of just the regression coefficients which allow the model to be used in input space, eliminating the need to first transform the data. This is where this work and the traditional approach deviate. The transformation matrix (W^*) is computed in (eq. 2) and computation of the regression coefficients are abandoned. In the next stage of the pipeline, W^* is used to project the data into latent space where a SVM model constructed.

3.4. Latent Space Support Vector Machine (LSSVM)

Once a latent space has been constructed and the transformation matrix (W^*) has been obtained, the training data ate transformed to the reduced dimensional latent space. This is achieved by multiplying the original data by the transformation matrix (W^*), thus creating the latent variables (T), from the original data (X):

$$T = XW^* \tag{10}$$

Finally, a Support Vector Machine SVM classifier [6] is constructed in latent space. We call this novel classifier a Latent Space Support Vector Machine. Although in this study we have only created a linear LSSVM model, using the familiar kernel methods, a non-linear LSSVM could also be created.

4. Design of Experiment

Our experimental design consisted of three phases: 1) sensitivity analysis of P, CV, and LV on (traditional PLS) model quality, 2) evolutionary optimization of LSSVM parameters, and 3) sensitivity analysis of the effect of LV on LSSVM quality. All model quality values were derived using a 5-fold cross validation procedure (split the training data into 5 folds; train on 4 of the folds, validate on the remaining fold, repeat for each of the 5 folds). The model quality metric is simply the ROC AUC values averaged over the 5 validation sets.

To estimate the quality of the LSSVM model, we first identified the data processing pipeline parameters that affect the quality of the model (t-test p-value threshold (P), coefficient of variation threshold (CV), the number of latent variables (LV), and the SVM parameters; for the linear kernel is just the SVM regularization parameter (C).

We performed a sensitivity analysis over a wide range of P, CV, and LV. Tor this sensitivity analysis, we used the traditional PLS as a surrogate for LSSVM because: 1) we expect the performance of LSSVM to be highly correlated (albeit improved) with PLS as we vary P, CV, and LV, 2) The evolutionary process can take a long time to complete, and 3) variability due to the selection of the SVM parameter(s) can have on the overall model quality (PLS has no free parameters other than the number of latent variables).

Once optimal P and CV parameters were determined using the ‘surrogate’ mode, these parameters were selected for the LSSVM LV sensitivity analysis. Here we fix P and CV, derive the reduced feature set (latent variables), and perform a sensitivity analysis of the effect the number of latent variables has on the overall LSSVM quality. Using the reduced feature set we performed a 5-fold, differential evolution search for the best LSSVM parameters in the following manner. For each latent variable, for each set of training/validation folds, a population of candidate LSSVM parameters solutions is created. For each candidate solution of LSSVM parameters, the LSSVM model is optimized using the training data and the model is then evaluated (ROC AUC computed) on the validation set for each of the 5 folds. The fitness metric is computed by averaging these 5 AUC values and assigned to the candidate solution. The DE process continues and the best performing model is selected from the final population of candidate solutions. This process is given in Figure 1:

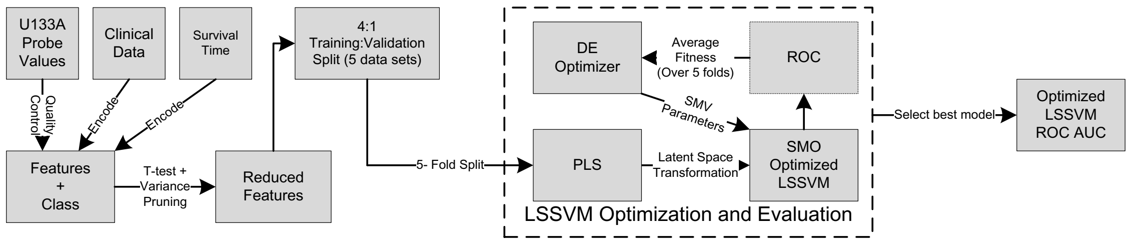

Fig. 1. Data processing pipeline and experimental design.

5. Results and Conclusion

The results of the initial sensitivity analysis were surprising. Whereas our previous work [1] seemed to indicate that using more latent variables generally caused a downward trend in model performance, our expanded sensitivity analysis showed that having more latent variables generally increased the quality of the model, with decreasing returns as the number of latent variables increased. For instance, over a wide range of p values, models with 5 latent variables clearly outperformed models with 1 latent variable (see Figure 2).

Another surprising observation was the effect the coefficient of variation threshold had on the overall quality. In the previous study, we selected a cv of 0.632 by observing the ‘elbow’ in the variance plot, our new sensitivity analysis shows that this value should be closer to 0.2. Using a value of 0.05 for the p threshold was generally a good choice. Figure 3 (a) shows the average validation AUC vs. p cutoff value, and figure 3 (b) shows the average validation AUC vs. coefficient of variation cutoff value for 5 latent variables.

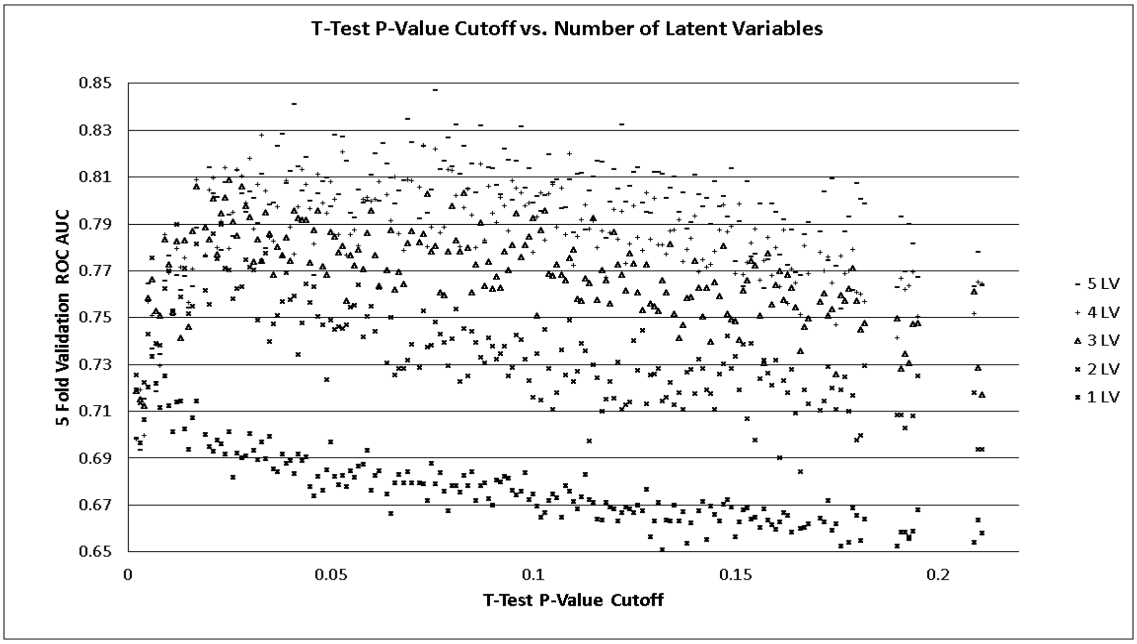

Fig. 2. Effect of number of latent variables on model quality: This figure shows that 5 latent variables outperforms fewer latent variables.

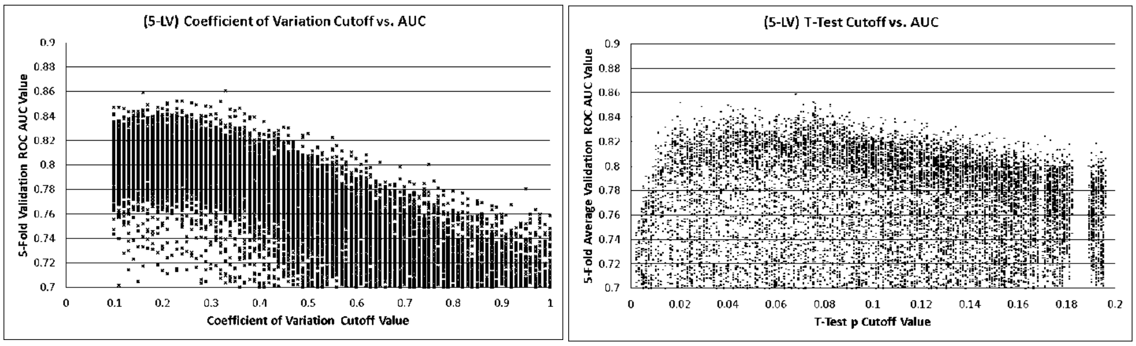

Fig. 3. (a) Effect of t-test p threshold selection on model quality for 5 latent variables: This figure shows that a p value cutoff of 0.02 to 0.1 are reasonable choices; (b) Effect of coefficient of variance threshold selection on model quality for 5 latent variables: This figure shows that values between 0.1 and 0.3 are good choices for the coefficient of variance threshold.

For the final LSSVM analysis, we sought to produce the best possible LSSVM model by optimizing the SVM regularization parameter (C). To find an optimal value, we used DE (population size: 20, 500 generations). At the end of the run, the model with the highest average validation score was selected (shown in figure 5 below). This was repeated for 1 – 5 latent variables. The table below shows the PLS validation scores vs. the LSSVM validation score for a p = 0.05 and cv = 0.2. The table also shows the number of features remaining after coarse feature reduction.

LSSVM (p=0.05, cv=0.2) vs. PLS (p=0.05, cv=0.2)								
LV	**LSSVM AUC**	**PLS AUC**	**AUC % increase**	**p threshold**	**cv threshold**	**num latent**	**#features after t-test reduction**	**#features after cv reduction**
5	0.8622	0.8187	5.3%	0.05	0.2	5	1613	1503
4	0.8427	0.8029	5.0%	0.05	0.2	4	1613	1503
3	0.8185	0.7878	3.9%	0.05	0.2	3	1613	1503
2	0.7782	0.7427	4.8%	0.05	0.2	2	1613	1503
1	0.7049	0.6912	2.0%	0.05	0.2	1	1613	1503

Fig. 4.. Comparison of the PLS model for p=0.5 and cv=0.2 for 1-5 latent variables vs. DE optimized LSSVM for p=0.5 and cv=0.2. The LSSVM model improves performance over traditional PLS by an average of about 4.2%.

Given a fixed set of data produced by an identical feature selection process (i.e. identical values for P, CV, and LV), the LSSVM classifier correctly classifies patient prognosis over PLS by a wide margin – up to a 5.3% improvement. Also note that the best performing model over all analyses (PLS only P, CV, LV sensitivity analysis, and EP derived LSSVM) combined is the 5 Latent Variable LSSVM (LSSVM AUC 0.8622 vs. PLS AUC 0.8607) Since the data are inherently noisy, the variation in PLS performance could also be attributed to how the data were divided during cross validation. Given the 90,000+ PLS models created during this study, it is likely that the top performers simply got a 'lucky split' of the data.

We therefore conclude that our original hypothesis is supported by the data presented herein. We have presented a novel Latent Space Support Vector Machine comprised of a hybrid PLS/SVM model. We have shown that given an identical set of data produced by a coarse feature reduction algorithm, the LSSVM model is capable of predicting the prognosis class (good, poor) of a given patient with and accuracy that is over 5% better than the previously studied PLS model. Furthermore, we have discovered through a widely expanded sensitivity analysis, that our original conclusions in [1] regarding the optimal CV and LV parameters were incorrect; model performance generally increases with the number of latent variables, at least as far as our limited analysis (up to 5 latent variables) has gone.

References

1. Land, W.,et. al., "Kernelized partial least squares for feature reduction and classification of gene microarray data", BMC Systems Biology 2011 5 (Suppl 3):S13.

2. Shedden, et. al., "Gene expression-based survival prediction in lung adenocarcinoma: a multi-site, blinded validation sutdy", Nature Medicine, 2008, 14(8): 822-7.

3. LIBSVM, PlotROC packages, http://www.csie.ntu.edu.tw/~cjlin/

4. Wold, S., et. al., "The PLS method -- partial least squares projections to latent structures -- and its applications in industrial RDP (research, development, and production)", 2004

5. Storn, R.; Price, K. "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces", Journal of Global Optimization 11: 341–359. 1997

6. Cortes C., Vapnik V., "Support-Vector Networks", Machine Learning, 20(3):273-297. 1995.

7. Fawcett, T., "ROC Graphs: Notes and Practical Considerations for Researchers", Pattern Recognition Letters, 27(8):882–891. 2004

8. Bennett, K. P., Embrechts, M. J., "An optimization perspective on kernel partial least squares regression. Advances in Learning Theory: Methods, Models and Applications". NATO Science Series III: Computer & Systems Science, 190, 227–250. 2003.