Progress in implementing the RankSVM paradigm

Ravi Kalia
Department of Statistics
University of Oxford
Oxford, 0X1 3TG
kalia@maths.ox.ac.uk

October 9, 2009

Abstract

This paper outlines work to date relating to the implementation and efficacy of the RankSVM algorithm in the R programming language. Model performance at the training and test stage is presented on artificial data. The procedure is compared with Proportional Odds Logistic Regression. Results to date are promising; however some known issues are currently being subjected to further careful analysis.

Keywords: empirical findings, statistical software, ordinal classification, ranking, machine learning, support vector machines.

1 Introduction

The ordinal classification task is that of assigning patterns to one of several classifications which possess a preference relation. That is, for a pattern $\mathbf{x} \in \mathbf{X}$ there is an association to an integer $y \in \{ 1, \ldots, K\}$ corresponding to a classification $c_y \in \mathbf{C} = \{c_1, \ldots, c_K\}$ where there exists a preference (reflexive, transitive and complete) relation $\leq$ on $\mathbf{C}$.

A mapping, $\mathbf{R} : \mathbf{X} \to \mathbf{C}$, which partitions the pattern space into ranks is sought using supervised learning. We concentrate on the *best*, in the maximum-margin sense, deterministic mapping constructed from the RankSVM approach. A review of existing approaches to ranking patterns using supervised learning can be found in Kalia (2006).

Briefly, RankSVM is a non-linear ordinal classification procedure that simultaneously fits a number of SVM classifiers in a constant Reproducing Kernel Hilbert Space (RKHS) by minimizing an ϵ-insensitive loss function through a quadratic approximation and iteratively reweighted least squares. A conservative allocation scheme is used to classify test examples,

since simultaneous use of classifiers does not automatically identify a class. There are a number of tuning and hyper-parameters that are determined by heuristics and validation rules.

In the next section we outline the formulation of RankSVM model and the parameter estimation process. From here, we consider an example dataset, and compare performance of RankSVM to a well-known statistical approach - Proportional Odds Logisitc Regression (POLR) (See Kalia(2006)). We conclude with our findings and a roadmap on completing a robust implementation.

1.1 The RankSVM model

The RankSVM model is designed to non-linearly deal with the lack of a natural metric among ordinal classes. Therefore, its precept is modelling ordered classes as multivariate data. To do so we make use of thermometer coding. Consider a dataset in which record i has pattern $\mathbf{x}_i$ and ordinal response y_i for $i = 1, \ldots, n$. y_i can be transformed into a row vector $\mathbf{z}_i$ where the j^{th} component (column) is:

$$z\left(y_i, j\right) = I_{\{y_i \leq j\}} - I_{\{y_i > j\}} \tag{1}$$

The multivariate generalisation (in the Rank SVM model) of constructing a sequence of hyperplanes, $\mathbf{H}\left(\cdot\right) = \left(H^1\left(\cdot\right), \ldots, H^K\left(\cdot\right)\right)$, that separate ordered classes is:

For all $\mathbf{x}(r) \in \mathbf{X}$ associated with $c_r, (y = r)$

$$H^r\left(x\right) = \mathbf{w}^r \Phi\left(\mathbf{x}\left(r\right)\right) + b^r = 0 \tag{2}$$

such that

$$z(s, r) H^r\left(x\left(r\right)\right) \geq 1 \tag{3}$$

for $r = 1, \ldots, K$ and $s = 1, \ldots, K$

Geometrically, we find a set of non-linear functions (equivalent to defining a Reproducing Kernel Hilbert Space - RKHS) such that in the space, linear separation of examples with different classifications is possible. More than that, we ask that the space be such that rotations and scalings have been performed to align the classifications in distance from the origin according to their rank. We are constructing an *ordered* RKHS. Put another way the boundaries are to be aligned so that an observation $x(r)$ is a member of a classification r if it falls above (positive exceedence of linear sum in RKHS) the boundaries corresponding to being less than or equal to rank r; for all higher ranks, the observation should fall below the corresponding boundaries.

Implicit in our formulation is the notion of irregular distances between classes, however these can be controlled by the non-linear functions in the RKHS. Therefore, rather than impose a vector of distances and then optimize, we allow the best distances to be set in the RKHS automatically from the tuning process - this is outlined next.

1.2 Parameter estimation

Given a sample of pairs $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ we train the model and estimate parameters to best allow for separation of classified examples. The process follows from above with calculus relating to Lagrange multipliers, Wolfe duals, Kuhn-Tucker equations and maximising margins - see Kalia (2007). In order to allow for misclassified examples in the training set, we have a penalty function that smoothes the parameters, favouring those that control the capacity (in the sense of Vapnik (1998)) of the model.

Parameter estimation relies on finding those parameters which achieve the following optimization criteria:

Minimize

$$\frac{1}{2}\sum_{r=1}^{K}||\mathbf{w}^r||^2 + C\sum_{i=1}^{n}\theta(u_i) \tag{4}$$

by varying $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_n)$ and $\mathbf{b} = (b_1, \ldots, b_n)$ subject to:

$$z\left(y_i, r\right) H^r\left(x\left(r\right)\right) \geq 1 \tag{5}$$

for $r = 1, \ldots, K$ and $i = 1, \ldots, n$, where

$$\theta_\epsilon(t) = \left(Max\left(0, t - \epsilon\right)\right)^2, \epsilon > 0, \tag{6}$$

C is a cost parameter relating to smoothness and

$$u_i = ||\mathbf{z}_i - sign\left(\mathbf{W}\Phi(\mathbf{x}_i) + \mathbf{b}\right)|| \tag{7}$$

Our rationale is that the minimisers of the above functional correspond to the *best* set of hyperplanes, from the space of all feasible hyperplanes which meet model criteria. The estimates are best in the sense that they maximise the distance between the hyperplanes while adhering to the separation of data by classifications; the maximum-margin principle.

Operationally, optimization is achieved via a backtracking algorithm where a quadratic approximation to the solution using iteratively re-weighted least squares will converge to the optimum because of the convex nature of the function. The final expression for

hyperplane specification is given by expanding the estimated parameters on their support vectors, since we do not estimate the non-linear functions in the RKHS directly. As a consequence, the final algorithm estimates a different but related set of parameters - $n(k+1)$ of them - many of which are zero (allowing for machine precision) corresponding to the internal points of the data cloud for each class label, relative to its convex hull. Details may be found in Kalia (2007).

There are however a number of tuning parameters relating to the optimization task and selection of RKHS.

1.3 Parameters

Model tuning parameters are determined by managing a trade-off between validation and training set error rates over a grid search on candidate values.

Choosing the tuning parameters search region, for the Rank SVM model is based on conventional heuristics.

- Model Parameter: Cost, C: The weight given to the sum of the penalties from misclassification. SVM studies have chosen values ranging from 1 to 10000 in increments of powers of 10.
- Model Parameter: Insensitivity, ϵ: The level of misclassification that the penalty function is insensitive to. That is, the amount of misclassification measure which the penalty function ignores. The maximum value, beyond which there is no incremental influence, is $2\sqrt{k}$.
- Algorithm Convergence, γ : Parameter which determines when to stop the optimization process and accept current solution; contingent on little , γ, change in objective function. Convergence criteria are usually set to 10^{-3}.
- Algorithm Dispersion, σ: Quantity that uniquely identifies RKHS for the Gaussian kernel used. Practitioner reviews suggests a default value that is the number of observations, once the covariates have been standardised.

Tuning parameters have great influence in determining the efficacy of the model. The search is however computationally expensive; this area needs further careful treatment.

2 Artificial data

In this section we describe an artificial dataset which will be used to assess the performance of the Rank SVM model against POLR with linear and cross term inputs.

A bi-linear function, given by Herbrich et al. (1999), which ranks patterns $\mathbf{x} = (x_1, x_2)$ to a class y is as follows:

$$y = i \iff 10\,(x_1 - 0.5)\,(x_2 - 0.5) + \epsilon \in (\theta(r_{i-1}), \theta(r_i)) \tag{8}$$

where $\epsilon \sim N(0, 0.125)$ and $\theta \in (-\infty, -1, -0.1, 0.25, 1, +\infty)$

We simulate $n = 1200$ patterns in the unit square, anchored at the origin, contained in the positive quadrant according to a uniform distribution.

Figure 1 shows the level sets corresponding to the rank boundaries θ in the feature space and those patterns $\mathbf{x}$ that have changed rank when perturbed by the error ϵ.

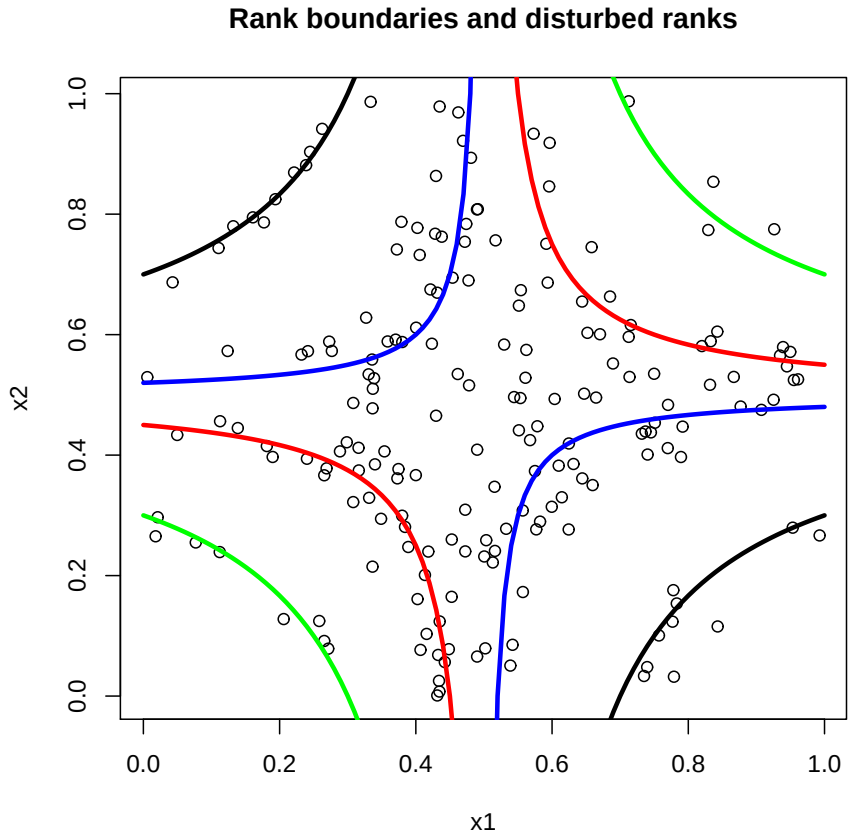

Figure 1: Level sets and perturbed ranks in feature space.

The algorithms need to learn this non-linear mapping in the presence of noise.

Since the feature space is 2 dimensional, we can use colour on bivariate plots to examine the result of training the model and classifier performance on test data.

3 Training and validation

Assessing performance of the two classifiers is done via a three-way data split. That is, partitioning the data into training, validation and test sets (one-third each). At the model selection stage, tuning parameters are chosen by comparing error rates on the training set to the validation set. The test set is used in the final performance comparison - making it difficult to hill-climb the test set. (The training, validation and test sets are stratified partitions of the observations, so that the fitted model is tested to the same criteria that it is trained with.)

	Rank				
DataSet	1	2	3	4	5
1	47	118	96	82	55
2	49	119	97	83	55
3	48	118	97	83	53

Performance comparisons on the test set will be made via analysis of confusion matrices (cross tables of actual to predicted performance), bivariate plots of patterns with classification labels and error rates.

First however, we turn to model selection for both classifiers.

3.1 Training Classifiers

Model training for the POLR classifier needs only training data. There are no tuning parameters. Therefore the validation set is not used in its parameter estimation. The RankSVM model, however, needs the validation set to determine tuning parameters.

The final performance is assessed via model predictions on test data.

3.1.1 POLR parameter estimation

The POLR model, outlined in Kalia (2006), estimates coefficients and cut-offs via a maximum likelihood approach, the likelihood being maximised using Iteratively Re-Weighted Least Squares (IRWLS).

We fit two POLR models, one with just the two covariates and the other with the two covariates and cross-terms (product of two covariates) as a third covariate.

Summary statistics for the two fits are below.

```
Call:
polr(formula = as.factor(observed_ranks[training_index]) ~ (x1)[training_index] +
    (x2)[training_index], Hess = TRUE)

Coefficients:
                          Value Std. Error   t value
(x1)[training_index] -0.7197500  0.3621471 -1.987452
(x2)[training_index] -0.3927259  0.3415586 -1.149805

Intercepts:
    Value   Std. Error t value
1|2 -2.7195  0.2968    -9.1619
2|3 -0.9308  0.2617    -3.5560
3|4  0.1541  0.2578     0.5979
4|5  1.3510  0.2694     5.0146

Residual Deviance: 1218.375
AIC: 1230.375
```

The coefficients for the two covariates are only marginally significant. The ordinal regression on linear covariates does not pick up the relationship with ranks. However the constant intercept terms do pick up that there are differences, reflecting frequencies in class probabilities but ignoring patterns.

```
Call:
polr(formula = as.factor(observed_ranks[training_index]) ~ x1[training_index] +
    x2[training_index] + cross_term, Hess = TRUE)

Coefficients:
                       Value Std. Error   t value
x1[training_index] -70.73094   6.303308 -11.22124
x2[training_index] -71.36642   6.392500 -11.16409
cross_term         143.32767  12.732273  11.25704

Intercepts:
    Value    Std. Error t value
1|2 -49.8654   4.4641   -11.1704
2|3 -37.1570   3.2948   -11.2774
3|4 -31.5118   2.8916   -10.8977
```

```
4|5 -20.7908   1.9750   -10.5268

Residual Deviance: 258.0593
AIC: 272.0593
```

The cross term model has all coefficients and intercepts being significant. It also has lower deviance and AIC scores, which confirms that of the two models the cross term ordinal POLR regression provides a better fit.

3.1.2 SVM parameter estimation

As mentioned in earlier sections, RankSVM parameters are estimated through an iterative scheme once the tuning parameters have been chosen. Model estimation is a two stage process based on identifying the best tuning parameters and training the model parameters. This is achieved by a trade-off in minimising validation (tuning parameters) and training (model parameters) error rates.

Our search is done on a grid over the tuning parameters for Cost, Insensitivity and Dispersion. We also search over the convergence criteria, since this is important in the computational time needed to optimize, and effectiveness of, the model.

Better than random results have been found using adhoc parameters. However, a detailed search of a sensible search space is still being conducted on a dedicated server. The results of the search can be used to analyse sensitivities and and provide suggestions for contracting the search space. Another process for estimation currently being considered is a Nelder-Mead search to estimate the tuning parameters. Insight into how sensitive the training and validation error will be to parameters changes is a useful by-product of the grid search.

Once a model has been selected, the next stage is to assess the performance on test data.

4 Model performance on test data

The validation set performance of the two models can be seen using confusion matrices and error rates, once the model has been chosen. However, a fair analysis can only be performed on the test data set, which is shown below using confusion matrices, error rates and bivariate plots in the covariate space which show ranks as coloured labels.

What we hope to see, once the results are available, is that RankSVM outperforms POLR for our synthetic dataset.

5 Findings

Partial performance results from our experiments confirm that there is value viewing ordinal data through a multivariate SVM perspective. However, there are also a number of issues that need to be addressed:

- The optimization and tuning task is computationally expensive. This slows the prediction process. However we can re-write the R code to make smarter use of the programming features of the language, and also think of specialist algorithms to improve performance.
- Results from RankSVM are good given that the search finds suitable parameters. A grid in a correct search region is an area that could do with further work.
- Given our experience with grid searches, we believe that there are non-unique tuning parameters for each test and training error rate combination, this is because up to a scaling some parameters are equivalent. It would be useful to find a restricted region where an injective mapping from parameters to error rates holds. Along with its mathematical elegance, the computational time could be substantially reduced.
- In our analysis the pattern can be treated as a continuous variable, regardless of its true underlying nature. It is worth considering if special metrics can be used for binary, multinomial and ordinal covariates. This could lead to better results.
- We had assumed in the construction of the RankSVM algorithm that only the support vectors would contribute non-zero weights. In the SVM case, Platt's SMO Algorithm is effective at determining which data points are support vectors. For the RankSVM model we assumed that zero, or close to zero, weights would determine unused data points (vectors). However, computationally, it is not clear vector weights (corresponding to interior data points) should be left out from the final model. One simple solution is to try all subsets of data points and select that which minimise the error. Another would be to throw out those observations, whose weights fail to meet metric criteria. This can lead to faster implementation of predictions.

References

[1] Herbrich, R., T. Graepelnd and K. Obermayer (1999). Large margin rank boundaries for ordinal regression. In Advances in Large Margin Classifiers, pp. 599–607. MIT Press, Cambridge, MA.

[2] Kalia, R. (2006). Modern pattern recognition methods for ordered classes. Technical Report,Department of Statistics, University of Oxford.

[3] Kalia, R. (2007). A Kernel approach to ordinal classification Transfer Report, Department of Statistics, University of Oxford.

[4] Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola (Eds.), Advances in kernel methods: Support vector machines. Cambridge, MA: MIT Press.

[5] Vapnik, V. (1998). Statistical Learning Theory. Wiley, New York, NY.