A Support Vector Approach to Censored Targets

Pannagadatta K. Shivaswamy
CCLS, Columbia University
New York, NY 10115
pks2103@cs.columbia.edu

Wei Chu
CCLS, Columbia University
New York, NY 10115
chuwei@cs.columbia.edu

Martin Jansche
Google, Inc.
New York, NY 10011
jansche@acm.org

Abstract

Censored targets, such as the time to events in survival analysis, can generally be represented by intervals on the real line. In this paper, we propose a novel support vector technique (named SVCR) for regression on censored targets. SVCR inherits the strengths of support vector methods, such as a globally optimal solution by convex programming, fast training speed and strong generalization capacity. In contrast to ranking approaches to survival analysis, our approach is able not only to achieve superior ordering performance, but also to predict the survival time very well. Experiments show a significant performance improvement when the majority of the training data is censored. Experimental results on several survival analysis datasets demonstrate that SVCR is very competitive against classical survival analysis models.

1 Introduction

Support Vector Machines (SVM) [16, 9] have achieved enormous success in the last decade. This success is mainly attributed to four factors: superior generalization capacity, globally optimal solution from a convex optimization problem, ability to handle non-linear problems using the so-called "kernel trick", and the sparseness of the solution which makes it possible to have specialized fast algorithms such as the sequential minimal optimization [14, 11]. SVMs were first proposed for binary classification problems and then subsequently extended to other problems like regression, clustering [12], ranking [3], etc. This paper extends SVM for the case of censored targets in survival analysis.

Survival analysis is a well-established field in statistics concerned time-to-event data. In the standard case, the event is death or failure, but the topic is much broader. It is applied not only in clinical research, but also in reliability engineering and financial insurance, etc. Classical examples of survival time measurements may include the time a kidney graft remains functional, the time a patient with colorectal cancer survives once the tumor has been removed by surgery, and so forth. All these times, named *survival time*, are triggered by an initial event followed by a subsequent event, such as from a kidney graft to graft failure, or from a surgical procedure to death.

There is one major difference between survival data and other types of numerical data: the time to the event occurring is not necessarily observed in all cases. Such non-observed events are quite different from missing data items. Suppose that some components are studied over a fixed period of observation – some of them fail but most of them do not fail in the observation period. For those components that do fail, their failure times (target values) are known precisely. For those components that do not fail, we can only say that their survival times are longer than the observation period. Such a target value is referred to as *right censored*, meaning we only know a lower bound on the failure time. Similarly there can be situations where only an upper bound on the failure time is known resulting in *left censored* observations. More generally, there could be observations for which both an upper and a lower bound are known. Such intervals are the most general type of observations since the other types of observations – fixed target, left censored, right censored – are special cases of it.

To the best of our knowledge, SVMs have not formulated for general interval targets. In this paper we develop a formulation, called SVCR, to learn from censored targets. SVCR inherits the strengths of SVM approaches such as a *globally optimal solution*, *fast training speed*, and *strong generalization capacity*. An extended version of this paper [15] has more details and more extensive experiments.

2 Censored Data

In supervised learning, we are given a set of labeled instances (observations) as training data, where an instance consists of a data vector (explanatory variables, attributes) plus a target (response variable). Depending on the type of target we obtain different problems. **Point Targets:** This is the case of standard regression where each vector $\mathbf{x}_i \in \mathbb{R}^m$

1550-4786/07 $25.00 © 2007 IEEE
DOI 10.1109/ICDM.2007.93

has a point target $y_i \in \mathbb{R}$. **Binary Class Labels:** The binary class labels are usually denoted by $y_i \in \{\pm 1\}$ while the attributes are still as in regression, that is, $\mathbf{x}_i \in \mathbb{R}^m$. **Interval Targets:** These are instances for which we have both an upper and a lower bound on the target. The tuple $(\mathbf{x}_i, l_i, u_i)$ with $\mathbf{x}_i \in \mathbb{R}^m, l_i \in \mathbb{R}, u_i \in \mathbb{R}, l_i < u_i$ denotes an interval target. **Survival Times:** An uncensored observation in survival analysis is the same as a point target defined above, while a right censored observation is written as $(\mathbf{x}_i, l_i, +\infty)$ whose survival time is greater than $l_i \in \mathbb{R}$. Finally, although not typical, for the sake of completeness, left censored observations are written as $(\mathbf{x}_i, -\infty, u_i)$ whose target is at most $u_i \in \mathbb{R}$.

The definition of interval targets provides a general description of the above observations. Suppose there is a dataset $(\mathbf{x}_i, l_i, u_i)_{i=1}^n$ of n observations with interval targets where $l_i < u_i$. The aim is to learn a function $f : \mathbb{R}^m \to \mathbb{R}$ so that the function values approximate the target values. In the following sections, we discuss performance measures for learning from such a dataset.

2.1 Average Absolute Error

Ideally, the regression function $f : \mathbb{R}^m \to \mathbb{R}$ should give the best guess on the target value of an instance $\mathbf{x}$ by $f(\mathbf{x})$ after learning from the training data. To evaluate the performance on intervals, the following definition of average absolute error (AAE) - which measures the absolute error outside the target interval - can be used:

$$\text{AAE} = \frac{1}{n}\sum_{i=1}^{n} \max\left(0, l_i - f(\mathbf{x}_i)\right) + \max\left(0, f(\mathbf{x}_i) - u_i\right). \quad (1)$$

2.2 Swapped Pairs and Rank Score

The Receiver Operating Characteristic (ROC) [17] is a popular performance metric to measure the quality of ordering for classification tasks. In this section an ROC-like metric is introduced for the censored data which is also closely related to the so called concordance index [6], a performance measure defined for models of survival analysis.

Given a dataset of n instances, there are $\binom{n}{2} = n(n-1)/2$ distinct pairs of instances. If we have a perfect ordering function f, then it would predict $f(\mathbf{x}_i) < f(\mathbf{x}_j)$ whenever $u_i \leq l_j$. However, in practice, a function learned from limited data does make mistakes. If the actual censored targets satisfy $u_i \leq l_j$ but $f(\mathbf{x}_j) < f(\mathbf{x}_i)$ then we call the pair (i, j) a *swapped pair*.

Not all the censored targets can be compared. To illustrate the definition of *comparable pairs*, let us consider a pair of instances, $(\mathbf{x}_i, l_i, u_i)$ and $(\mathbf{x}_j, l_j, u_j)$. The pair can be compared when $u_i \leq l_j$ (or $u_j \leq l_i$). To preserve the order of $\mathbf{x}_i$ and $\mathbf{x}_j$, the optimal function f should satisfy $f(\mathbf{x}_i) < f(\mathbf{x}_j)$ whenever $u_i \leq l_j$. Similarly if $u_j \leq l_i$ then the desired function must satisfy $f(\mathbf{x}_j) < f(\mathbf{x}_i)$. If neither of the two conditions (that is, $u_i \leq l_j$ or $u_j \leq l_i$) is satisfied, there exists an overlapping region between the two interval targets. We call such pairs *incomparable pairs* since there is no meaningful order of the targets for such pairs. We quantify the quality of an ordering function f by calculating the fraction of comparable pairs of samples that are correctly ordered by the function f, thus:

$$\text{RankScore} = \frac{\#\text{comparable} - \#\text{swapped}}{\#\text{comparable}}. \quad (2)$$

If the function f orders every pair of comparable samples in the right order (according to the actual targets) then RankScore $= 1$. If it reverses every pair of samples, then RankScore $= 0$. Finally, for random ordering RankScore would be around 0.5. The RankScore as in (2) is also closely related to Gehan's generalization [4] of the Wilcoxon-Mann-Whitney statistic [13] and thus an AUC-like metric for our scenario of censored data.

3 A Support Vector Formulation

Consider a censored dataset $(\mathbf{x}_i, l_i, u_i)_{i=1}^n$ as defined in Section 2. In this setting, we need the predicted value for $\mathbf{x}_i$ to be within the interval (l_i, u_i). As long as the output $f(x_i)$ is between l_i and u_i, there is no penalty. We penalize if the output is more than u_i or if it is less than l_i. Thus, the loss function for this case becomes:

$$c(f(\mathbf{x}_i), l_i, u_i) = \max(0, l_i - f(\mathbf{x}_i), f(\mathbf{x}_i) - u_i). \quad (3)$$

The loss is exactly the absolute error defined in (1).

Note that when $l_i = -\infty$ or $u_i = +\infty$, this loss function becomes one sided. Let us partition the index set $\{1, 2, \ldots n\}$ into three disjoint sets as follows:

$$I_{\text{u}} \stackrel{\text{def}}{=} \{i | l_i > -\infty, u_i < +\infty\},$$
$$I_{\text{r}} \stackrel{\text{def}}{=} \{i | u_i = +\infty\},$$
$$I_{\text{l}} \stackrel{\text{def}}{=} \{i | l_i = -\infty\}.$$

Note that there is no overlap between I_{r} and I_{l}, since no target takes infinity on both sides.

The set I_{u} contains the indices of those instances which have both a finite lower bound and a finite upper bound, while I_{r} and I_{l} contain the indices of the instances that are right censored and left censored, respectively. We further define two index sets

$$L \stackrel{\text{def}}{=} I_{\text{u}} \cup I_{\text{r}} \text{ and } U \stackrel{\text{def}}{=} I_{\text{u}} \cup I_{\text{l}}, \quad (4)$$

where L contains the indices of those instances whose targets have a finite lower bound while U contains the indices of those having a finite upper bound.

We now propose the following formulation for the censored regression task:

$$\min_{\mathbf{w},b,\xi,\xi^*} \frac{1}{2}\|\mathbf{w}\|^2 + C\left(\sum_{i\in U}\xi_i + \sum_{i\in L}\xi_i^*\right) \tag{5a}$$

$$\text{s.t. } \mathbf{w}^\top \mathbf{x}_i + b - u_i \le \xi_i^* \quad \forall i \in U; \tag{5b}$$

$$l_i - \mathbf{w}^\top \mathbf{x}_i - b \le \xi_i \quad \forall i \in L; \tag{5c}$$

$$\xi_i \ge 0 \ \ \forall i \in U; \quad \xi_i^* \ge 0 \ \ \forall i \in L. \tag{5d}$$

As can be seen from the formulation (5), it is making use of all the information available in the dataset. By introducing Lagrangian multipliers $\alpha_i^* \ge 0$ for the inequalities in (5b) and $\alpha_i \ge 0$ for the inequalities in (5c), the dual of the above formulation can be shown to be:

$$\min_{\alpha,\alpha^*} \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(\mathbf{x}_i,\mathbf{x}_j) \tag{6}$$
$$-\sum_{i\in L} l_i\alpha_i + \sum_{i\in U} u_i\alpha_i^*$$
$$\text{s.t. } \sum_{i\in L}\alpha_i - \sum_{i\in U}\alpha_i^* = 0; 0 \le \alpha_i, \alpha_i^* \le C \ \forall 1 \le i \le n.$$

where $\alpha_i = 0 \ \ \forall i \in L$, $\alpha_i^* = 0 \ \ \forall i \in U$ are dummy variables and $k(\mathbf{x}_i,\mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$. The dot product $\mathbf{x}_i^\top \mathbf{x}_j$ can be replaced by a kernel function to obtain a non-linear function mapping. At the optimal solution, with the dummy variables, the function value of $\mathbf{x}$ is represented by $f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)k(\mathbf{x}_i,\mathbf{x}) + b$. Note that usually a small fraction of $\{\alpha_i - \alpha_i^*\}$ is non-zero.

Algorithm Complexity The support vector formulation leads to a standard quadratic programming problem. The problem size is equal to $|U| + |L|$. Fast and scalable algorithms of convex programming, such as SMO [14, 11], can be easily adapted for the solution. Empirically we show the algorithm complexity is about $\mathcal{O}(n^{2.1})$. As is the case for linear SVM, the training cost can be further reduced to be linear in n [10, 8].

Connections to other Support Vector methods The standard Support Vector Regression (SVR) and Classification (SVC) formulations are just special cases of (5). It is easy to see that, given a regression dataset $(\mathbf{x}_i, y_i)_{i=1}^n$, by converting each sample to $(\mathbf{x}_i, y_i - \varepsilon, y_i + \varepsilon)$ where $\varepsilon > 0$ comes from the ε-insensitive loss, (5) reduces to SVR. Similarly, in case of classification, a sample $(\mathbf{x}_i, +1)$ is converted to $(\mathbf{x}_i, 1, \infty)$ and a negative sample is converted into $(\mathbf{x}_i, -\infty, -1)$; it is easy to see that these samples, when plugged into (5), give rise to SVC constraints.

4 Experiments

In this section we present the experiments that were performed to validate the proposed method. We briefly describe the competing methods below; further details and results can be found in [15].

- Support Vector Regression (SVR): This classical method is used as a baseline. Since SVR can handle only uncensored point targets, all the censored data were ignored; only those observations for which the failure times were known exactly were used.
- Constraint Classification approach (CC-SVM) [5]: In this approach, two binary classification constraints are added for each comparable pair to maintain the order; the resulting classifier is then used as a ranking function. Note that there is a quadratic blow-up in the number of constraints added. Also, this approach can only rank the instances but cannot predict the failure times.
- Gaussian Process Preference Learning (GP-PL) [2] defines an appropriate likelihood function on the target values and puts a Gaussian process prior. The resulting convex optimization can be efficiently solved. Like CC-SVM, this method also can give only a ranking but cannot predict survival times.
- Classical Parametric Models (CPM): These models assume that the overall survival of a population follows one of a family of parametric distributions, such as the Weibull, exponential, normal etc. These unconditional models can be turned into regression models (conditional models) by replacing one of the free parameters with a (suitably transformed) linear predictor. The linear predictor is simply the inner product of a column vector $\mathbf{w}$ of unknown regression coefficients and the vector $\mathbf{x}_i$ of observed attributes for the item of interest. The linear predictor is usually transformed to satisfy constraints on the parameter of the unconditional survival distribution.

4.1 Simulated Censoring

We selected four large regression datasets $(\mathbf{x}_i, y_i)_{i=1}^n$ with $y_i \in \mathbb{R}$.[1] For each dataset, one thousand observations were drawn uniformly at random for training and the remaining observations were used as the test set.

4.1.1 All Left Censored

From the one thousand training instances, we randomly selected a fraction η of instances. Different values of η that

[1] These regression datasets are available at `http://www.liacc.up.pt/~ltorgo/Regression/DataSets.html`.

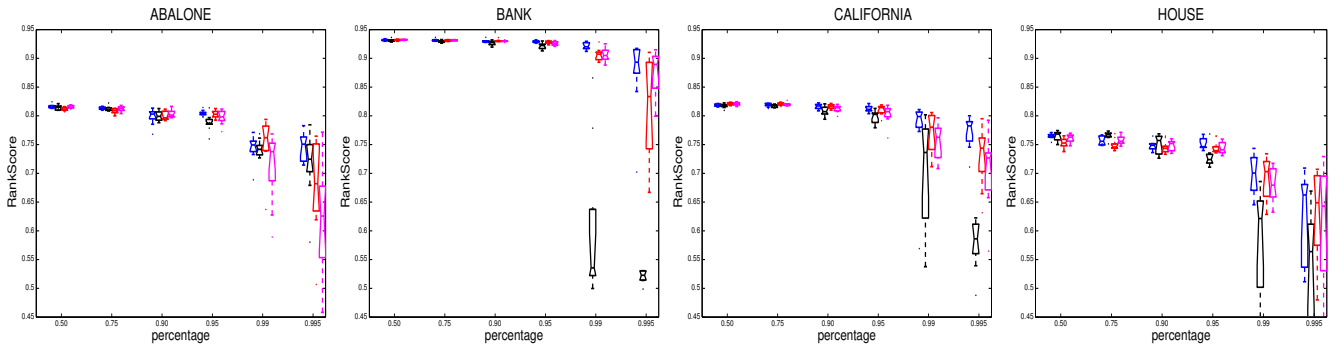

Figure 1. Linear Results: left to right: abalone, bank, California and house datasets. For each percentage (η) the boxplots from left to right: SVCR, SVR, GP-PL and CC-SVM. The notched-boxes have lines at the lower quartile, median, and upper quartile values. The whiskers are lines extending from each end of the box to the most extreme data value within $1.5\cdot$IQR(Interquartile Range) of the box. Outliers are data with values beyond the ends of the whiskers, which are displayed by dots.

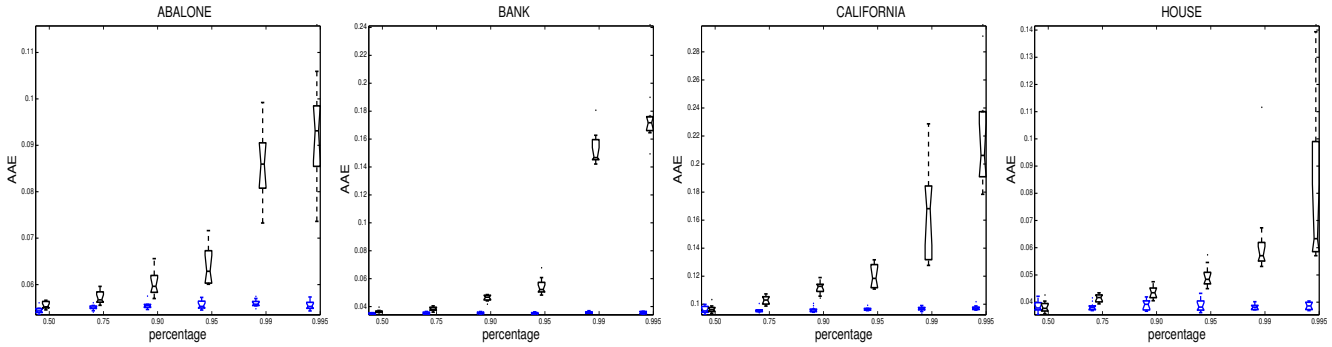

Figure 2. Kernel Results: AAE with SVCR and SVR respectively as a function of η with half left censored data. For each percentage (η) the boxplots from left to right: SVCR and SVR.

we used were $0.5, 0.75, 0.9, 0.95, 0.99$ and 0.995. For these selected instances, we changed the targets to be left censored. That is, each instance $(\mathbf{x}_i, y_i)$ was changed so that the new instance was $(\mathbf{x}_i, -\infty, y_i)$. Thus, instead of having a fixed target, these instances were changed so that their targets were at most y_i instead of being fixed at y_i.

For all the methods, parameters were chosen by two-fold cross-validation on the training data. Parameters that minimized the average RankScore over the two folds were selected. For SVCR and SVR we obtained both the RankScore and the average absolute error on the entire test data. However, GP-PL and CC-SVM cannot produce the actual target output; they can only give an ordering. Thus we also compared RankScores of the different methods. The entire experiment was repeated ten times.

Figure 1 shows the results with a linear kernel. As the fraction η increases, RankScore decreases for all the methods. However, the drop in RankScore for SVCR is much less than for the other methods. The boxplots for SVR look much shorter for $\eta = 0.995$ than $\eta = 0.95$ in some of the plots. This is because, at $\eta = 0.995$, there is very little information available for the SVR, and the RankScore corresponds almost to random guessing (RankScore = 0.5). Thus the resulting variance in RankScore for very high η is much smaller. We note that it is unreasonable to expect SVCR to do very well in terms of average absolute error (AAE) in this case. This is because, for all the censored samples, the output of SVCR is required to be at most y_i, which biases the prediction greatly. Thus the predictions given by the SVCR tend to be much less than the actual value for fixed targets.

4.1.2 Half Left Censored

The experiment setup in this case was very similar to that in Section 4.1.1, but instead of censoring all the η fraction of instances to the left, half of them were censored to the left and the other half were censored to the right. Thus half of the instances $(\mathbf{x}_i, y_i)$ were converted to $(\mathbf{x}_i, y_i, +\infty)$ and the

	Abalone		Bank		California		House	
η	0.99	0.995	0.99	0.995	0.99	0.995	0.99	0.995
SVCR	$\mathbf{82.0 \pm 0.5}$	82.0 ± 0.5	$\mathbf{93.2 \pm 0.1}$	93.2 ± 0.1	$\mathbf{83.0 \pm 0.2}$	$\mathbf{82.9 \pm 0.1}$	$\mathbf{78.5 \pm 0.3}$	$\mathbf{78.0 \pm 0.3}$
SVR	78.8 ± 3.6	73.7 ± 13.1	79.7 ± 18.3	79.3 ± 18.7	74.7 ± 11.2	75.3 ± 11.6	67.7 ± 12.2	71.3 ± 6.1
GP-PL	81.9 ± 0.4	$\mathbf{82.1 \pm 0.4}$	93.2 ± 0.2	$\mathbf{93.3 \pm 0.1}$	82.7 ± 0.3	82.5 ± 0.27	78.0 ± 0.3	77.7 ± 0.2

Table 1. RankScores with polynomial kernel with half left censored data. RankScores have been multiplied by one hundred.

other half were converted to $(\mathbf{x}_i, -\infty, y_i)$. The other settings in this experiment were the same as in Section 4.1.1. However, the number of comparable pairs in this setup became larger. This is because a left censored and a right censored observation might be comparable, but two left censored observations can never be compared. Due to lack of space we only present results with a polynomial kernel of degree two in this section. Since running CC-SVM in this case was computationally prohibitive, we do not have results for that method. Also, in this case, unlike in Section 4.1.1, predictions are not biased towards one side. Figure 2 shows the AAE for SVR and SVCR (GP-PL is left out as it only gives a ranking). As the value of η increases, the AAE of SVR is significantly higher than the AAE of SVCR, as one would expect. Table 1 gives the RankScores in this setup. One can see that our method has an advantage over the other methods. We conclude this section noting that SVCR has higher performance potential (over other methods) when there are significantly many one-sided censored examples (which would mean less information for the other methods).

4.2 SVCR Versus Classical Models

In this section we study how the SVCR method compares with the classical parametric models (CPM). While the experiments in Section 4.1 used simulated censoring, in this section we performed the experiments on survival datasets. These datasets are typical datasets that are used in the survival analysis literature. We compared our method with several parametric survival distributions: the Weibull, exponential, normal, logistic, log-normal, and log-logistic models.

The five datasets that we used were Lung, Heart, Nwtco, Veteran[2] and botdata [1]. In each of these datasets, missing values, if any, were replaced by the mean of the attribute. Each of these datasets was divided into two folds of equal size. Two training runs were then performed. The first run used the first fold as training data and the second fold as unseen test data. Training consisted of model selection and parameter estimation. Model selection was performed by exhaustively trying out all CPM models for different parametric survival distributions (one of Weibull, exponential, normal, logistic, log-normal, or log-logistic). The model with the lowest AAE was selected, as determined by five-fold cross-validation on the training data. The winning model was then retrained on the entire training data, and the fitted model was used to predict survival times for the unseen test data. The whole process was repeated a second time with the role of training and test data reversed. Similarly SVCR was trained using one fold and was tested on the other fold. The parameter C of SVCR was chosen by five-fold cross-validation. The value of C that resulted in the lowest AAE by cross-validation was then chosen. SVCR was then trained on the entire fold and was tested on the unseen test data.

Results are shown in Table 2. It can be seen that SVCR wins in almost every case in terms of AAE. In terms of RankScore, the results are not in favor of any one method. We attribute this to the fact that cross-validation was done on AAE in the first place for both the methods. We believe that better RankScores can be achieved if the cross-validation is done using RankScore as the criterion or if the objective of SVCR is modified to minimize the RankScore directly.

4.3 Runtime and Scalability

Instances were chosen randomly from the California housing regression dataset. They were used for SVCR without any modification. Appropriate datasets were generated by comparing the target values for GP-PL and CC-SVM. For each sample size, the algorithms were run on five different randomly chosen training sets of that size. The user times were noted for each run and the numbers were averaged to get a final run time for each sample size. A polynomial kernel with degree two was used in each case. SVM-Light [7] was used for training the CC-SVM.

Figure 3 shows the run times plotted on a log-log plot. The three lines shown in the plot were obtained by linear regression on the points plotted for the respective methods. It is quite evident that SVCR has an advantage over other methods. The number of samples shown on the x-axis is the size of the training set *before* modifying them for CC-SVM. Thus the higher slope of the CC-SVM curve is attributed to the blow-up in the problem size. We do not show the runtime of SVR as it is the same as that of SVCR. The slopes

[2]These four datasets can be found in the R package "Survival".

		lung		heart		veteran		botdata		nwtco	
		CPM	SVCR	CPM	SVCR	CPM	SVCR	CPM	SVCR	CPM	SVCR
AAE	Fold1	**154.4**	155.5	**128.2**	135.8	74.9	**69.8**	22.0	**13.3**	709.5	**441.4**
	Fold2	148.3	**142.2**	277.6	**174.4**	**92.4**	92.9	23.3	**20.5**	831.9	**510.1**
	Avg	151.3	**148.8**	202.9	**155.1**	83.6	**81.3**	22.6	**16.9**	770.7	**475.8**
RankScore	Fold1	**0.633**	0.595	**0.652**	0.623	**0.673**	0.668	0.790	**0.888**	**0.717**	0.671
	Fold2	0.615	**0.635**	0.540	**0.576**	**0.705**	0.704	**0.769**	0.766	**0.680**	0.598
	Avg	**0.624**	0.615	0.598	**0.599**	**0.689**	0.686	0.780	**0.827**	**0.698**	0.635

Table 2. AAE and RankScore on survival datasets for classical methods and SVCR. RankScores have been multiplied by one hundred.

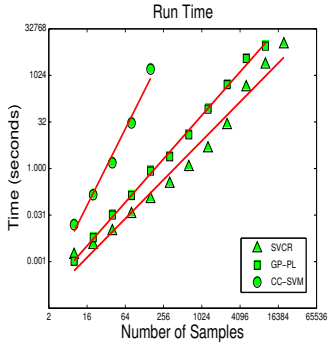

Figure 3. Run times for different approaches. Both x-axis and y-axis are on a log scale. The slopes for SVCR, GP-PL and CC-SVM are 2.0894, 2.3358 and 4.1131 respectively on the shown plot.

of these lines indicate an empirical run time complexity of the three algorithms. SVCR run time $\mathcal{O}(n^{2.1})$ compared favorably to that of preference learning $\mathcal{O}(n^{2.3})$ and CC-SVM $\mathcal{O}(n^{4.1})$.

5 Conclusion

We studied different approaches that can be applied to the survival time prediction/ranking; We proposed a new formulation to handle censored data overcoming some of the problems with the previous approaches. Experiments showed significant performance gains in the presence of higher levels of censoring. SVCR also competes with the parametric models. SVCR was shown to be scalable and has favorable run time compared to other methods. We recommend that SVCR be widely used in survival analysis.

References

[1] G. C. Cawley, N. L. C. Talbot, G. J. Janacek, and M. W. Peck. Sparse Bayesian kernel survival analysis for modeling the growth domain of microbial pathogens. *IEEE Transactions of Neural Networks*, 17(2):471–481, 2006.

[2] W. Chu and Z. Ghahramani. Preference learning with gaussian processes. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 137–144, 2005.

[3] W. Chu and S. S. Keerthi. Support vector ordinal regression. *Neural Computation*, 19:792–815, 2007.

[4] A. E. Gehan. A generalized Wilcoxon test for comparing arbitrarily singly-censored samples. *Biometrika*, 52:203–223, 1965.

[5] S. Har-Peled, D. Roth, and D. Zimak. Constraint classification for multiclass classification and ranking. In *NIPS*, pages 785–792. MIT Press, 2003.

[6] F. E. Harrell Jr. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer Series in Statistics. Springer, 2001.

[7] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 169–185. MIT Press, 1998.

[8] T. Joachims. Training linear svms in linear time. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226, 2006.

[9] V. Kecman. *Learning and Soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models*. MIT Press, Cambridge, MA, USA, 2001.

[10] S. S. Keerthi and D. DeCoste. A modified finite newton method for fast solution of large scale linear svms. *Journal of Machine Learning Research*, 6:341–361, Mar 2005.

[11] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to SMO algorithm for SVM regression. *IEEE Transactions on Neural Networks*, 11(5):1188–1193, 2000.

[12] D. Lee. An improved cluster labeling method for support vector clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(3):461–464, 2005. Member-Jaewook Lee.

[13] H. B. Mann and D. R. Whitney. On a test of whether one of 2 random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18:50–60, 1947.

[14] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, Cambridge, MA, USA, 1999.

[15] P. K. Shivaswamy, W. Chu, and M. Jansche. A support vector approach to censored targets. Technical Report CCLS-07-03, CCLS, Columbia University, 2007.

[16] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[17] M. H. Zweig and G. Campbell. Receiver-operating characteristic (roc) plots: a fundamental evaluation tool in clinical medicine. *Clinical chemistry*, 39(4):561–577, 1993.