A Support Vector Approach to Censored Targets

Pannagadatta K. Shivaswamy Wei Chu Martin Jansche
Center for Computational Learning Systems
Columbia University
New York, NY, 10115
pks2103,chuwei,jansche@cs.columbia.edu

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. Interestingly, this approach provides a general formulation for both standard regression and binary classification tasks. 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. Controlled experiments show the significant performance improvement when majority of the training data is censored. Experimental results on several survival analysis datasets verify that SVCR is very competitive against classical survival analysis models.

Keywords

Support Vector Machines, Survival Analysis, Regression, Censored Data

1. INTRODUCTION

Support Vector Machines (SVM) [17] have achieved enormous success in the last decade. This success is mainly attributed to four factors: 1) Rooted in the statistical learning theory [18], SVMs possess superior generalization capacity; 2) A globally optimal solution is obtainable by solving a convex optimization problem, while the problems of local minima impede other contemporary approaches, such as neural networks; 3) Using the so-called kernel trick, it is convenient to solve non-linear problems in arbitrarily high dimensional feature spaces; 4) Lastly and most importantly, only a part of training samples are involved in solution representation. This sparseness makes training SVMs relatively fast via specialized algorithms, such as the sequential minimal optimization (SMO) algorithms [13, 11]. The speedup facilitates its applications to large-scale learning tasks. SVMs were first proposed for binary classification problems [1], and then successfully extended to other problems like regression [16], ranking [4] etc. However, in many applications, the targets associated with samples of interest cannot be simply indicated by single values, e.g. censored data in survival analysis.

Survival analysis is a well-established field in classical statistics concerned with data of the time to some event. In the standard case, the event is death or failure, but the topic in 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 therapy to death. As another example, an engineering company might be interested in better understanding the failure time of a working component, i.e. the survival time from installation to failure; given a set of attributes of the component, it is typical to investigate how the failure time is related to the attributes. Often, it is of interest to the company to rank all components of the same type according to their susceptibility to failure, so that appropriate precautionary measures can be taken to reduce losses caused by the oncoming failures.

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 the samples. Such non-observed events are quite different from missing data items as well. 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 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* since we only know a lower bound on the failure time. Similarly there can be situations where a failure is discovered but it is not precisely known when the failure occurred. In this case, we only know an upper bound on the failure time. This category of observations are called *left censored* targets. Conventionally survival analysis does not consider left censored data. More interestingly, if there are inspections at fixed intervals of time and a failure is discovered on a particular inspection but the component was known to be in good condition during the previous inspection, there exist both an upper bound and a lower bound on the failure times. Summarizing this discussion, there are four kinds of observations in practice: left censored, right censored, intervals and fixed values. However, intervals are the most general type of observations since if we know that a failure time t satisfies $l \leq t \leq u$ then, by setting $u = +\infty$ we obtain a right censored observation, by setting $l = -\infty$ we obtain a left censored observation and by setting $l = u$, we obtain a fixed value.

To our best knowledge, SVMs have not been attempted on general interval targets. In this paper we develop a support vector regression formulation, which we call SVCR, to learn from censored targets. SVCR inherits the strengths of SVM approaches, such as a *globally optimal solution* by convex quadratic programming, *fast training speed* and *strong generalization capacity*, and also contributes the following additional merits:

- The proposed method predicts survival time rather than only the relative order. Censored data can be translated into ordered pairs and then a ranking function can be trained to order the samples accordingly. However, such a ranking approach is unable to predict the time to the event, it can only order the events.
- The proposed method benefits greatly by learning from (left or right) censored data, where as ranking approaches fail to extract ordered pairs from one-side censored data. It is typical in related applications that the majority of the targets are right censored, since failure/death rate is usually very low, say less than 1% in population. Controlled experiments show the drastic performance difference in presence of plenty of censored observations.

The paper is organized as follows: In Section 2 we formalize the settings of censored targets and discuss the appropriate performance evaluations. In Section 3 we review the well-known approaches that can be adapted to learning from censored data. In Section 4 we propose our formulation for censored data and discuss related issues. We report experimental results with details in Section 5 and conclude in Section 6.

2. CENSORED DATA

In supervised learning, we are given a set of samples and their corresponding targets as training data. Usually the targets can be described by real values, such as the point targets in standard regression and the binary class labels in classification, where as the censored data in survival analysis can be represented by an open-end interval only, i.e. $(l_i, +\infty)$ with $l_i \in \mathbb{R}$. While the targets of fixed values are fairly standard and well studied problems in data mining community, the interval target setting is not a well known one. In this case the target is within an open-end interval (l_i, u_i) with $l_i < u_i$. We now review the different types of targets that define different problems:

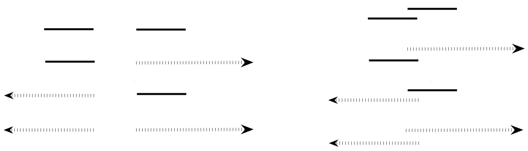

Figure 1: Solid lines indicate finite intervals, dotted lines with arrowheads indicate left or right censored data. *Left*: Pairs of interval targets that can be compared *Right* Pairs of interval targets that can not be compared.

- *Point Targets:* This is the case of standard regression where each sample $\mathbf{x}_i \in \mathbb{R}^m$ has a point target $y_i \in \mathbb{R}$. Thus the tuples $(\mathbf{x}_i, y_i)_{i=1}^n$ denote a typical regression dataset.
- *Binary Class Labels:* The binary class labels are usually denoted by $\{\pm 1\}$. In this case, a dataset is the set of tuples $(\mathbf{x}_i, y_i)_{i=1}^n$ with $\mathbf{x}_i \in \mathbb{R}^m$ and $y_i \in \{\pm 1\}$.
- *Interval Targets:* These are samples for which we have both an upper and a lower bound on the target. The tuple $(\mathbf{x}_i, l_i, u_i)$ with $l_i < u_i, l_i \in \mathbb{R}, \mathbf{x}_i \in \mathbb{R}^m, u_i \in \mathbb{R}$ denotes an interval target.
- *Survival Times:* An uncensored sample in survival analysis is the same as a point target defined above, while a right censored sample 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 samples 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 samples 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 a sample $\mathbf{x}$ by $f(\mathbf{x})$ after learning from the training data. To evaluate the performance on intervals, following definition of average absolute error (AAE) can be used:

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

where $\max(a, b)$ returns a if $a > b$, returns b otherwise. This quantity measures the absolute error outside the target interval. If the predicted target is in the required interval, we do not count that in the absolute error. However, if the predicted target falls outside the actual target interval, we take the absolute value of the difference between the prediction and the closer end point of the interval. The average of these values over an entire dataset gives AAE.

2.2 Swapped Pairs and Rank Score

Receiver Operating Characteristic (ROC) [19] is a popular performance metric to measure the quality of ordering for classification datasets. The area under the ROC curve (AUC) is obtained by noting the area under the curve obtained by plotting the number of true positives against the number of false positives as the predictions are obtained sequentially for the entire dataset. Area under the curve is one for perfect ordering, 0.5 for random guessing and 0 for reverse ordering. One major limitation of this is that it needs class information to give an area under the curve. In the situation that we are considering (censored outputs) there is no class label information. Thus the AUC metric cannot be applied in this scenario, but the idea can be generalized to measure the quality of ordering over censored data.

We introduce a performance metric tailor-made for our scenario, which is closely related to the so-called concordance index [7], a performance measure defined for models of survival analysis. Given a dataset of n samples, there are $\frac{n(n-1)}{2}$ pairs by taking each combination of two samples at a time. We denote this number by *total pairs*. 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*.

Clearly, not all the censored targets can be compared. To illustrate the definition of *comparable pairs*, let us consider a pair of samples: $(\mathbf{x}_i, l_i, u_i)$ 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 $f(\mathbf{x}_i) < f(\mathbf{x}_j)$ when ever $u_i \leq l_j$. 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 overlapped region between the two interval targets. Since the targets could be anywhere in the overlapped region, we cannot identify the order of the two samples precisely; there does not exist a meaningful order in this case (Figure 1). We call such sample pairs *uncomparable 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:

$$\text{RankScore} = \frac{\#\text{comparable} - \#\text{swapped}}{\#\text{comparable}} \tag{2}$$

where #total = #comparable+#uncomparable. If the function f orders every pair of comparable samples in the right order (according to the actual targets) then there are no swapped pairs giving RankScore $= 1$. If it reverses every pair of samples, then all the samples are swapped thereby giving RankScore $= 0$. Finally, if it arbitrarily arranges the samples, nearly half the comparable samples are arranged in the right order and the other half are swapped giving a RankScore$= 0.5$ corresponding to a random guess. The RankScore as in (2) is closely related to Gehan's generalization [5] of the Wilcoxon-Mann-Whitney statistic [12] and thus an AUC-like metric for our scenario of censored data.

3. LITERATURE REVIEW

With the above discussion on censored data, we now discuss some of the existing methods that could (with some modifications) be applied to handle censored data. We briefly describe each method and mention the potential problems faced at the end of each sub-section.

3.1 Regression

Classical regression framework could be applied to handle censored targets in a naive way. Given $(\mathbf{x}_i, y_i)_{i=1}^n$, the problem in regression is to find a function $f : \mathbb{R}^m \to \mathbb{R}$ that closely matches the target y_i for the corresponding $\mathbf{x}_i$. In Support Vector Regression (SVR), the so-called ϵ-insensitve loss, $c(e) = \max(0, |e| - \epsilon)$, where $e = f(\mathbf{x}_i) - y_i$, is minimized along with regularization controlling the capacity [14]. The ϵ-insensitive loss, see the left graph in Figure 2, is zero as long as the absolute difference between the actual and the predicted values is less than ϵ. When this absolute difference exceeds ϵ, there is a cost which grows linearly. When f is linear, $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$; in its linear form, SVR is given by:

$$\min_{\mathbf{w}, b, \xi, \xi^*} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{3a}$$

$$\text{s.t. } y_i - \mathbf{w}^\top \mathbf{x}_i - b \leq \epsilon + \xi_i,\ \xi_i \geq 0 \quad \forall 1 \leq i \leq n; \tag{3b}$$

$$\mathbf{w}^\top \mathbf{x}_i + b - y_i \leq \epsilon + \xi_i^*,\ \xi_i^* \geq 0 \quad \forall 1 \leq i \leq n. \tag{3c}$$

Much of the popularity of SVR comes from the fact that it can be kernelized. By mapping each sample $\mathbf{x}_i$ to a Hilbert space $\mathcal{H}$ using a mapping $\phi : \mathbb{R}^m \to \mathcal{H}$, SVR can regress in the Hilbert space thus giving a complex function in the input space. Further more, these mappings are not done explicitly but implicitly by a kernel function $k : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ such that $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$. The kernelized version of the SVR is given by:

$$\min_{\alpha, \alpha^*} \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(\mathbf{x}_i, \mathbf{x}_j) \tag{4a}$$

$$- \sum_{i=1}^{n}(y_i - \epsilon)\alpha_i + \sum_{i=1}^{n}(y_i + \epsilon)\alpha_i^*$$

$$\text{s.t. } \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0; \quad 0 \leq \alpha_i, \alpha_i^* \leq C\ \forall 1 \leq i \leq n. \tag{4b}$$

Once the above formulation (4) is solved to get the optimal α and α^*, the value of the function at $\mathbf{x}$ is given by: $f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)k(\mathbf{x}_i, \mathbf{x}) + b$.

One naive way to handle censored data with support regression is to consider only those samples for which the targets are known exactly. Given a training dataset $(\mathbf{x}_i, l_i, u_i)_{i=1}^n$, we need to consider the subset of samples from these n samples for which $l_i = u_i$. By using only these samples in (3) (or in the dual (4)), we can learn a function f that predicts the target values. We can as well treat the function thus obtained for ordering the samples.

The main disadvantage of this approach is that, by considering only those samples with point targets, it is totally ignoring the order information available in censored samples. Any pair of samples that are comparable still have some ordering information in them, but this naive approach cannot exploit such information.

3.2 Classification

We present another possible approach to handle censored target in this section making use of classification. Given a dataset $(\mathbf{x}_i, y_i)_{i=1}^n$ with $y_i \in \{\pm 1\}$, the aim here is to find a function $f : \mathbb{R}^m \to \{\pm 1\}$ so that the signs of the actual label and the predicted label match. Support vector machine (SVM) finds a classification rule by maximizing the so called margin [1]. SVM obtains a classification rule $f(\mathbf{x}) = sign(\mathbf{w}^\top \mathbf{x} + b)$ by solving the following quadratic program (QP):

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \tag{5a}$$

$$\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall 1 \leq i \leq n, \tag{5b}$$

where there is a penalty for samples that are either within the margin or on the wrong side of the classification boundary. The parameter C trades off between the margin and the penalty. As in the case of regression, a non-linear decision boundary is found by solving the dual of the problem (5) making use of a kernel function. The kernel version of the SVM formulation is below:

$$\min_{\alpha} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{n} \alpha_i \tag{6a}$$

$$\text{s.t. } \sum_{i=1}^{n} \alpha_i y_i = 0; \quad 0 \leq \alpha_i \leq C \ \ \forall 1 \leq i \leq n. \tag{6b}$$

The resulting non-linear decision boundary in this case is given by: $f(\mathbf{x}) = \sum_{i=1}^{b} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b$ where b can be found as mentioned in [1].

Support Vector Classification problem can be readily modified for ordering the censored data using the constraint classification approach [6], named CC-SVM. Suppose that we have two censored samples $(\mathbf{x}_i, l_i, u_i)$ and $(\mathbf{x}_j, l_j, u_j)$, if the two samples are *comparable*, we can impose constraints in the SVM formulation to maintain the required order. Suppose that $u_i \leq l_j$, then we know that the target value for $\mathbf{x}_j$ is at least as high as that for $\mathbf{x}_i$. We can explicitly require that the this order be maintained by imposing:

$$\mathbf{w}^\top(\mathbf{x}_i - \mathbf{x}_j) + b \leq -1, \qquad \mathbf{w}^\top(\mathbf{x}_j - \mathbf{x}_i) + b \geq 1.$$

Essentially, each pair of *comparable* samples gives rise to two classification samples. Thus the pair of censored samples $(\mathbf{x}_i, l_i, u_i)$ and $(\mathbf{x}_j, l_j, u_j)$ gives rise to the pair of classification samples $(\mathbf{x}_i - \mathbf{x}_j, -1)$ and $(\mathbf{x}_j - \mathbf{x}_i, +1)$ when ever the censored samples are comparable and $u_i \leq l_j$. For a given censored training dataset, for each pair of samples we can use this rule to translate them into classification samples. Thus we now have a classification problem to solve, which can be done with the SVM. The resultant $\mathbf{w}$ (or α in the kernel version) can be used to order the samples according to the targets.

One of the main drawbacks of this approach is that there is a quadratic blow up in the problem size. If there are n samples in the censored training dataset, there are a total of $\frac{n(n-1)}{2}$ pairs. This can be computationally prohibitive for large datasets. Another issue associated with the approach is that while it can order the samples, it can not give a good estimate of the actual output. It only considers the ordering among the samples totally ignoring the actual output values.

3.3 Preference Learning

As discussed in Section 2.2, comparable pairs can be identified from the training data set $(\mathbf{x}_i, l_i, u_i)_{i=1}^n$. Given a comparable pair $\mathbf{x}_i$ and $\mathbf{x}_j$ with $l_j > u_i$, we know that the target value of $\mathbf{x}_i$ is less than that of $\mathbf{x}_j$. In the rank sorted by the target values in a descending order, $\mathbf{x}_i$ is definitely ranked higher than $\mathbf{x}_j$. We can also say $\mathbf{x}_i$ is preferred to $\mathbf{x}_j$ in the criterion of the target value, denoted as $\mathbf{x}_i \succ \mathbf{x}_j$. The techniques proposed for learning pairwise preference relations can be adapted here to train a ranking function.

An ideal ranking function for $\mathbf{x}_i \succ \mathbf{x}_j$ should be consistent with their preference relations, i.e. $f(\mathbf{x}_i) > f(\mathbf{x}_j)$. An appropriate likelihood function was introduced by Chu and Ghahramani [3] to capture the preference relation of $\mathbf{x}_i \succ \mathbf{x}_j$, which is defined as $\Phi\left(\frac{f(\mathbf{x}_i)-f(\mathbf{x}_j)}{\sqrt{2}\sigma}\right)$ where $\sigma > 0$ reflects the noise level in measurement, $\Phi(z) = \int_{-\infty}^{z} \mathcal{N}(\gamma; 0, 1)d\gamma$ and $\mathcal{N}(\cdot; 0, 1)$ denotes a normal distribution with zero mean and unit variance. Coupled with a Gaussian process prior, which is defined as $\mathcal{P}(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}; \mathbf{0}, \mathbf{K})$ where $\boldsymbol{f}$ is a column vector containing function values of the samples in comparable pairs and $\mathbf{K}$ is a Gram matrix whose ij-th element is defined by a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$, the posterior probability of the function values is then proportional to

$$\mathcal{N}(\boldsymbol{f}; \mathbf{0}, \mathbf{K}) \prod_{ij} \Phi\left(\frac{f(\mathbf{x}_i) - f(\mathbf{x}_j)}{\sqrt{2}\sigma}\right),$$

where the index ij runs over all comparable pairs. The maximum a posteriori (MAP) estimate is equivalent to the minimizer of the negative logarithm of the posterior probability, i.e.

$$\frac{1}{2}\boldsymbol{f}^\top \mathbf{K}^{-1}\boldsymbol{f} - \sum_{ij} \ln \Phi\left(\frac{f(\mathbf{x}_i) - f(\mathbf{x}_j)}{\sqrt{2}\sigma}\right).$$

This is a convex programming problem that yields a globally optimal solution. At the MAP estimate denoted as $\boldsymbol{f}_{\text{MAP}}$, the function value at $\mathbf{x}$ is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n} \beta_i k(\mathbf{x}_i, \mathbf{x})$$

where $\beta_i = \sum_{ij} \frac{\partial}{\partial f(\mathbf{x}_i)} \ln \Phi\left(\frac{f(\mathbf{x}_i)-f(\mathbf{x}_j)}{\sqrt{2}\sigma}\right)\Big|_{\boldsymbol{f}=\boldsymbol{f}_{\text{MAP}}}$ and the index ij runs over all comparable pairs. The function values can then be used to determine the preference relations.

An important advantage of preference learning over the constraint classification approach [6] is that the problem size is same as the number of samples rather than quadratic. In applications to censored data, a potential drawback lies in that it is impossible to learn from uncomparable pairs. Any two left censored (or right censored) samples are *uncomparable.* However, it is typical in practical applications that the majority of the samples are left or right censored, as failure/death rate is usually very low, say less than 1% in population.

3.4 Classical Survival Analysis

Classical survival analysis aims to determine a parametric distribution of time to event (failure) T. This distribution of survival time is characterized by the survival function

$$S(t) = \mathcal{P}(T > t),$$

Distribution	$p(t)$	$S(t)$
Weibull	$\gamma\lambda t^{\gamma-1}\exp(-\lambda t^{\gamma})$	$\exp(-\lambda t^{\gamma})$
exponential	$\lambda\exp(-\lambda t)$	$\exp(-\lambda t)$
normal	$\frac{\lambda}{\sqrt{2\pi}}\exp(-\frac{1}{2}(\lambda t)^2)$	$1-\Phi(\lambda t)$
log-normal	$\frac{\lambda}{t\sqrt{2\pi}}\exp(-\frac{1}{2}(\lambda\ln t)^2)$	$1-\Phi(\lambda\ln t)$
log-logistic	$\frac{\gamma\lambda t^{\gamma-1}}{(1+\lambda t^{\gamma})^2}$	$(1+\lambda t^{\gamma})^{-1}$

Table 1: Statistical distributions commonly used in parametric survival analysis. The Gamma distribution is defined as $\Gamma(t)=\int_0^{+\infty} z^{t-1}\exp(-z)dz$ and the cumulative normal distribution is defined as $\Phi(z)=\int_{-\infty}^{z}\mathcal{N}(\gamma;0,1)d\gamma$.

which is the probability that the survival time is greater than t. The probability density function is defined as

$$p(t)=-\frac{dS(t)}{dt}.$$

Another function of interest is the hazard function, given by

$$h(t)=\frac{p(t)}{S(t)},$$

which represents the instantaneous probability of event occurrence at time t. Any parametric distribution over non-negative values of t may be applied as $p(t)$; see Table 1 for a list of distributions commonly used.

Classical regression models are used extensively to study the effect of the attributes on the survival time. These models assume that the overall survival of a population follows one of the parametric distributions shown in Table 1. 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 a vector $\mathbf{x}$ 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. For example, in the Weibull regression model the free parameter λ is equated with $\exp(\mathbf{w}^\top\mathbf{x})$ to ensure that it is positive. This is a familiar technique for Generalized Linear Models. The other distributions in Table 1 can be turned into regression models along similar lines. Cawley et al [2] use a kernelized version of these parametric models.

4. A SUPPORT VECTOR FORMULATION

Our aim in this section is to propose an approach to overcome the potential difficulties of the approaches to learning tasks with censored targets, that were discussed in Section 3, namely:

- Quadratic blow up in the problem size;
- Not utilizing the entire available information;
- No estimate of the actual target.

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). The ideas in ϵ-insensitive loss function can be generalized for this situation. As long as the output $f(x_i)$ is between l_i and u_i, there is no empirical error. 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:

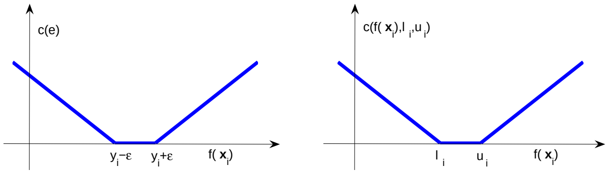

Figure 2: Loss functions: SVR loss (left) and SVCR loss (right).

$$c(f(\mathbf{x}_i), l_i, u_i)=\max(l_i-f(\mathbf{x}_i), f(\mathbf{x}_i)-u_i). \quad (7)$$

Figure 2 shows both the ϵ-insensitive loss of SVR and our modified loss function. The loss we used is exactly the absolute error defined as in (1).

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

$$\begin{aligned} I_u &\overset{\text{def}}{=} \{i|l_i>-\infty, u_i<+\infty\},\\ I_r &\overset{\text{def}}{=} \{i|u_i=+\infty\},\\ I_l &\overset{\text{def}}{=} \{i|l_i=-\infty\}. \end{aligned}$$

Note that there is no overlap between I_r and I_l, since no sample's target takes infinity on both sides.

The set I_u contains the indices of those samples which have both a finite lower bound and a finite upper bound, while I_r and I_l contain the indices of the samples that are right censored and left censored respectively. We further define two index sets

$$L \overset{\text{def}}{=} I_u \cup I_r \quad\text{and}\quad U \overset{\text{def}}{=} I_u \cup I_l, \quad (8)$$

where L contains the indices of those samples 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 dataset:

$$\min_{\mathbf{w},b,\xi,\xi^*} \frac{1}{2}\|\mathbf{w}\|^2 + C\left(\sum_{i\in U}\xi_i + \sum_{i\in L}\xi_i^*\right) \quad (9a)$$

$$\text{s.t. } \mathbf{w}^\top\mathbf{x}_i + b - u_i \le \xi_i^* \quad \forall i\in U; \quad (9b)$$

$$l_i - \mathbf{w}^\top\mathbf{x}_i - b \le \xi_i \quad \forall i \in L; \quad (9c)$$

$$\xi_i \ge 0 \ \ \forall i\in U; \quad \xi_i^*\ge 0 \ \ \forall i \in L. \quad (9d)$$

As can be seen from the formulation (9), it is making use of all the information available in the dataset. It is also possible to kernelize the above formulation following steps similar to that in SVR. By introducing Lagrangian multipliers $\alpha_i^*\ge 0$ for the inequalities in (9b) and $\alpha_i \ge 0$ for the inequalities in (9c), the dual of the above formulation can be shown to be:

$$\min_{\alpha,\alpha^*} \frac{1}{2} \sum_{i,j=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) k(\mathbf{x}_i, \mathbf{x}_j) \qquad \text{(10a)}$$

$$- \sum_{i \in L} l_i \alpha_i + \sum_{i \in U} u_i \alpha_i^*$$

$$\text{s.t.} \sum_{i \in L} \alpha_i - \sum_{i \in U} \alpha_i^* = 0; \qquad \text{(10b)}$$

$$0 \leq \alpha_i, \alpha_i^* \leq C \quad \forall 1 \leq i \leq n. \qquad \text{(10c)}$$

where $\alpha_i = 0 \quad \forall i \in L$ and $\alpha_i^* = 0 \quad \forall i \in U$ are dummy variables. 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.

4.1 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 [13, 11], can be easily adapted for the solution. It is well-known that the complexity of these algorithms is much lower than $\mathcal{O}(n^3)$ due to the sparseness in solution representation, where n denotes the problem size. Empirically we show the algorithm complexity is about $\mathcal{O}(n^{2.1})$. As for linear SVM, the training cost can be further reduced to be linear with n [10, 9, 15].

4.2 Discussion

We note some interesting properties of the formulation (9) in this section. As a general support vector formulation, by setting l_i and u_i appropriately, it can be shown to be equivalent to both SVM formulation (5) and SVR formulation (3).

A classification dataset $(\mathbf{x}_i, y_i)_{i=1}^n$ where $y_i \in \{\pm 1\}$ can be converted to a censored dataset as follows. Following the large margin ideas, a sample $\mathbf{x}_i$ with label $y_i = +1$ can be converted as $(\mathbf{x}_i, +1, +\infty)$, where as $\mathbf{x}_j$ with label $y_j = -1$ is considered as $(\mathbf{x}_j, -\infty, -1)$. It can be observed that $L = \{i | y_i = +1\}$ as per the definition in (8). Thus plugging this sample in the constraint (9c), we get:

$$\mathbf{w}^\top \mathbf{x}_i + b \geq 1 - \xi_i,$$

which confirms with (5b) for a positive sample. Similarly, $U = \{i | y_i = -1\}$ as defined in (8). Plugging this sample in the constraint (9b), we get: $\mathbf{w}^\top \mathbf{x}_i + b + 1 \leq \xi_i$ which can equivalently be written as,

$$-1(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i.$$

Again it is consistent with (5b) for a negative sample. In addition, the objective in the SVCR formulation (9) is the same as that in the SVM formulation (5) thus showing the equivalence. The equivalence can be easily identified from the dual formulations (6) and (10) as well.

The connection to the SVR formulation is much easier to see. Given a regression data set $(\mathbf{x}_i, y_i)_{i=1}^n$ where $y_i \in \mathbb{R}$, each sample $(\mathbf{x}_i, y_i)$ is converted to $(\mathbf{x}_i, y_i - \epsilon, y_i + \epsilon)$ where $\epsilon > 0$ comes from the ϵ-insensitive loss function in SVR. It is straightforward to see that plugging these censored samples in the SVCR constraints (9c) and (9b) gives the SVR formulation (3).

dataset	No. of features	size of train set	size of test set	test pairs (in millions)
Abalone	10	1000	2000	1.99
Bank	8	1000	1788	1.59
California	8	1000	5000	12.49
House	8	1000	5000	12.49

Table 2: Test set size for different datasets.

5. EXPERIMENTS

In this section we describe the experiments that we performed to demonstrate the superiority of the proposed method over the existing methods. We carried out controlled experiments in Section 5.1 on large regression datasets to verify the benefits from censored data. We show that in the presence of a large number of censored samples, performance of our method is much superior compared to the other methods. We also show the superiority of our method over the classical methods on survival datasets in Section 5.2. Finally we report some experimental results on runtime and scalability of our approach in Section 5.3.

5.1 Controlled Experiments

We selected four large regression datasets $(\mathbf{x}_i, y_i)_{i=1}^n$ with $y_i \in \mathbb{R}$.[1] One thousand samples were randomly drawn for training in each dataset and test sets of different sizes as mentioned in Table 2 were randomly drawn too. The competing methods for this set of experiments were: SVCR, SVR, CC-SVM and Gaussian process preference learning (GP-PL). See Section 3 for technical details of these methods. The reason for choosing one thousand training samples was to have a manageable dataset with the CC-SVM approach while still having a reasonably large training dataset. We present two kinds of experiments with these regression datasets below.

5.1.1 All left censored

From the one thousand training samples, we randomly selected η fraction of the samples. Different values of η that we used were $0.5, 0.75, 0.9, 0.95, 0.99$ and 0.995. For these selected samples, we changed the targets so that they became left censored. That is, each sample $(\mathbf{x}_i, y_i)$ was changed so that the new sample was $(\mathbf{x}_i, -\infty, y_i)$. Thus, instead of having a fixed target, these samples were changed so that their targets were at most y_i instead of being fixed at y_i. This gave two kinds of samples in the training dataset: ones that had fixed real target and the others that had censored output. Since it is typical to see failure rates of 5%, 1% or 0.5% fraction of the components in many domains, using such high η is justified.

For SVR, we simply ignored those training samples that had censored targets. Clearly, as the value of η becomes closer to one, the number of samples that are given to SVR reduces accordingly: from 500 samples at $\eta = 0.5$ to 5 samples at

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

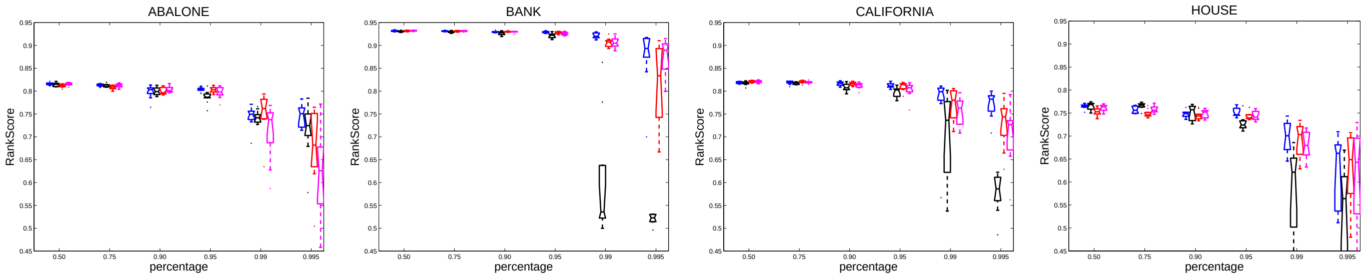

Figure 3: Linear Results: left to right: abalone, bank, California and house datasets. For each percentage (η) the boxplots from left to right: SVCR, SVR, GP-PL and CC-SVM. The notched-boxes have lines at the lower quartile, median, and upper quartile values. The whiskers are lines extending from each 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.

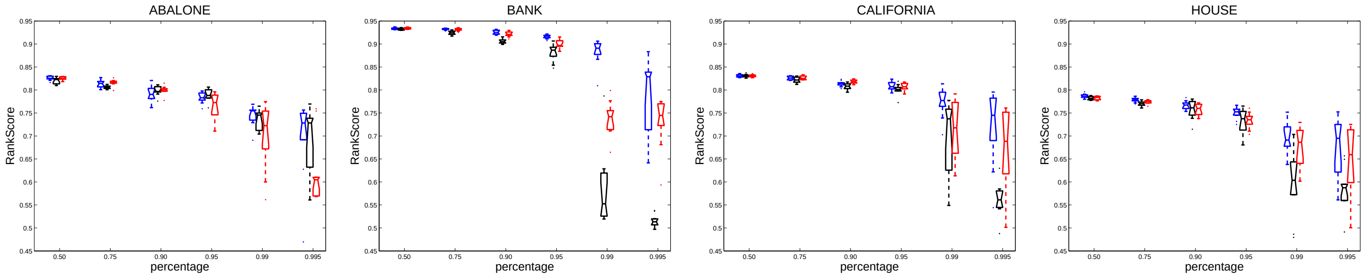

Figure 4: Kernel Results: left to right: abalone, bank, California and house datasets. For each percentage (η) the boxplots from left to right: SVCR, SVR and GP-PL.

$\eta = 0.995$. It is reasonable to expect deterioration in the performance of the SVR as η is increased.

For CC-SVM, we compared each comparable pair and then generated two new samples when ever they were comparable. The generated samples were constrained to preserve the order between them as described in Section 3.2. For linear CC-SVM, we used a fast linear SVM implementation [15]. Due to the quadratic blow up problem mentioned in Section 3.2, solving the kernel CC-SVM became prohibitive, thus we do not have results with kernelized CC-SVM.

Similarly, for GP-PL, we compiled the entire list of comparable samples and gave that to the preference learning algorithm as described in Section 3.3.

For SVCR we used all the one thousand samples with both the censored (η fraction of the training) and uncensored ($1-\eta$ fraction) samples.

For all the methods, parameters were chosen by doing a two-fold cross validation on the training data. Parameters that minimized the average RankScore over the two folds were chosen. Training was then done on the entire training set for each method with chosen parameters. The aim was to see how well the methods could predict on the actual test set that was held out initially from the regression datasets. We did not censor the test dataset since our aim was to see how well the methods could perform with respect to clean test set. For SVCR and SVR we found out both the RankScore and the average absolute error on the entire test data. However, GP-PL and CC-SVM can not produce the actual target output; they can only give an ordering. Thus we also compared RankScores of the different methods. The RankScore was calculated as in (2). Since there were no censored samples in the test set, there were a large number of *comparable* pairs as described in Table 2. Entire experiment was repeated ten times.

Figure 3 has the results for the linear case. Figure 4 shows the results for the polynomial kernel with degree two, i.e. $(1 + \mathbf{x}_i^\top \mathbf{x}_j)^2$. For each dataset, on the x-axis we show the fraction η of the training samples censored. On the y-axis we show the RankScore distribution. The four boxplots indicate the results on SVCR,SVR, GP-PL and CC-SVM respectively. For an exact description of the boxplots, refer to Figure 3. As the fraction η was increased there was a drop in RankScore for all the methods. However, the drop in RankScore for SVCR is much less compared to 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. The RankScore with this η corresponds almost to random guessing (RankScore $= 0.5$). Thus the resulting variance in RankScore for very high η is much smaller. It can be noticed that the gains of SVCR over the other

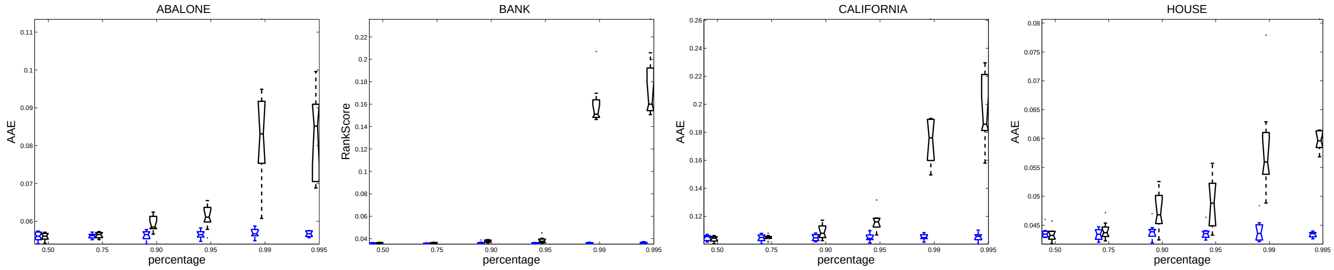

Figure 5: Linear Results: AAE with SVCR and SVR respectively as a function of η with half left censored data. For each percentage (η) the boxplots from left to right: SVCR and SVR.

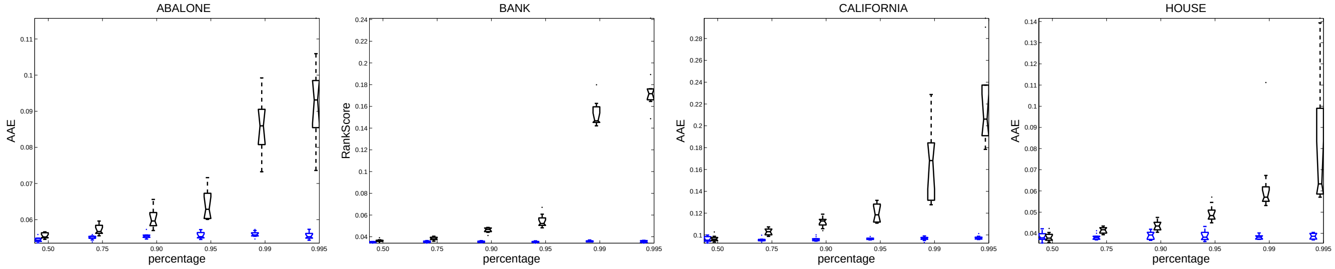

Figure 6: Kernel Results: AAE with SVCR and SVR respectively as a function of η with half left censored data. For each percentage (η) the boxplots from left to right: SVCR and SVR.

methods is more prominent in the kernel version results.

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 that biases the prediction greatly. Thus the predictions given by the SVCR tend to be much less than the actual value for fixed point outputs. However, if we either cross validate on AAE or if there are plenty of both left and right censored examples, we can expect to have a low AAE score as we show in the next section.

5.1.2 *Half left censored*

The experiment setup in this case was very similar to that in Section 5.1.1 but instead of censoring all the η fraction of samples to the left, half of the samples were censored to the left and the other half were censored to the right. Thus half of the samples $(\mathbf{x}_i, y_i)$ from η fraction were converted to $(\mathbf{x}_i, y_i, +\infty)$ and the other half of the samples were converted to $(\mathbf{x}_i, -\infty, y_i)$. Other settings in this experiment were the same as in Section 5.1.1. However, the number of comparable pairs in this setup became more. This is because a left censored and a right censored example might be comparable, but two left censored examples can never be compared. Higher number of comparable pairs meant a better performance by CC-SVM and GP-PL compared to that in Section 5.1.1. Also, in this case, unlike in Section 5.1.1, predictions are not biased towards one side. Figure 5 and Figure 6 show the results. As the η value is increased the AAE of SVR is significantly higher than the AAE of SVCR, as we expect. Tables 3 and 4 give the RankScores in this setup. As can be seen, in the linear case there is no clear winner. In the kernelized version, our method seems to have slight 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.

dataset	# samples	# censored	# features
Lung	228	63	7
Heart	172	97	4
veteran	137	9	10
botdata	309	125	6
nwtco	4028	3457	8

Table 5: Survival Datasets.

5.2 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 5.1 were controlled, in this section we performed the experiments on survival datasets. These datasets are typical datasets that are used in survival analysis literature. We compared our method with several parametric survival distributions (Weibull, exponential, normal, logistic, log-normal and log-logistic) that were introduced in Section 3.4

The five datasets that we used were Lung, Heart, Nwtco, Veteran[2] and botdata [2]. In each of these datasets, missing values, if any, were replaced by the mean of the feature. Table 5 shows the number of features, samples and the num-

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

	Abalone		Bank		California		House	
η	0.99	0.995	0.99	0.995	0.99	0.995	0.99	0.995
SVCR	81.4 ± 0.5	$\mathbf{81.7 \pm 0.2}$	93.1 ± 0.1	93.0 ± 0.1	81.5 ± 0.1	82.2 ± 0.2	$\mathbf{76.0 \pm 0.6}$	$\mathbf{76.5 \pm 0.4}$
SVR	76.3 ± 8.5	78.4 ± 3.4	79.8 ± 20.6	79.6 ± 20.6	73.1 ± 12.4	74.6 ± 11.6	67.3 ± 11.2	68.4 ± 14.1
GP-PL	81.3 ± 0.3	81.3 ± 0.3	$\mathbf{93.2 \pm 0.1}$	$\mathbf{93.2 \pm 0.1}$	$\mathbf{81.7 \pm 0.1}$	$\mathbf{82.3 \pm 0.3}$	74.2 ± 0.4	74.2 ± 0.9
CC-SVM	$\mathbf{81.7 \pm 0.2}$	81.6 ± 0.2	$\mathbf{93.2 \pm 0.1}$	$\mathbf{93.2 \pm 0.1}$	$\mathbf{81.7 \pm 0.1}$	$\mathbf{82.3 \pm 0.3}$	75.4 ± 0.3	75.2 ± 0.7

Table 3: RankScores with linear kernel with half left censored data. RankScores have been multiplied by one hundred.

	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 4: RankScores with polynomial kernel with half left censored data. RankScores have been multiplied by one hundred.

ber of censored samples in each of these datasets. 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 doing a five fold cross validation. The C that gave in lowest AAE over 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 6. It can be seen that SVCR wins almost in 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.

5.3 Runtime and Scalability

The purpose of this experiment is to show that SVCR has favourable runtime compared to CC-SVM and GP-PL. Samples 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. Polynomial kernel with degree two was used in each case. SVMLight [8] was used for training the CC-SVM.

Figure 7 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 are the number of samples *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 as described in Section 3.2. We do not show the runtime of SVR as it is the same as that of SVCR. The slopes of these lines indicate the empirical run time complexity of the three algorithms. SVCR run time $\mathcal{O}(n^{2.1})$ compared favourably to that of preference learning $\mathcal{O}(n^{2.3})$ and CC-SVM $\mathcal{O}(n^{4.1})$.

SVCR could be trained on a 20000 example subset from the California housing in around 170 minutes on average on a Dual Opteron 270 system with 16 GB RAM. GP-PL needed approximately the same time for around 10000 samples. In contrast CC-SVM with pairs formed using 320 samples took around 790 minutes. Thus our method is scalable to large datasets. In addition, specialized algorithms for linear regression can be easily adapted to SVCR making the linear version much faster.

6. CONCLUSION

We studied different approaches that can be applied to an important real world problem. We proposed a new formulation to handle censored data overcoming the problems with the previous approaches. While the generalization is straightforward, SVCR is shown to have interesting connections with SVR and SVM. SVCR is the most general supervised learning problem in that regression and binary classification are special cases of this problem. Experiments showed significant performance gains in the presence of higher levels of censoring. SVCR can compete with the parametric models. SVCR was shown to be scalable and has favourable run time compared to other methods. It is hoped that SVCR will be widely used in survival analysis problems.

Acknowledgements

The authors were partly supported by a research contract from Consolidated Edison. The authors would like to thank

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

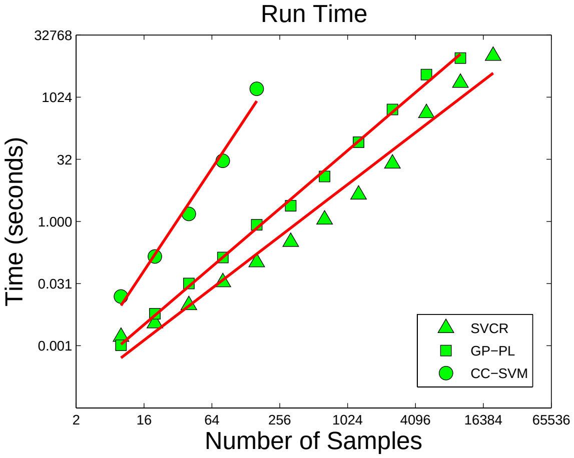

Figure 7: Run times for different approaches. Both x-axis and y-axis are on a log scale. The slopes for SVCR, GP-PL and CC-SVM are 2.0894, 2.3358 and 4.1131 respectively on the shown plot .

David Waltz, Albert Boulanger and Roger Anderson for useful discussions and support.

7. REFERENCES

[1] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[2] 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.

[3] 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.

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

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

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

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

[8] 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.

[9] 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.

[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] 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.

[13] 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.

[14] B. Schölkopf and A. J. Smola. *Learning with Kernels.* MIT Press, Cambridge, Mass., 2002.

[15] V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear SVMs. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 477–484, 2006.

[16] A. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14:199–222, 2004.

[17] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, New York, 1995.

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

[19] 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.