Validation Based Sparse Gaussian Processes for Ordinal Regression

P.K. Srijith[1], Shirish Shevade[1], and S. Sundararajan[2]

[1] Computer Science and Automation, Indian Institute of Science, India
{srijith,shirish}@csa.iisc.ernet.in
[2] Yahoo! Labs, Bangalore, India
zensid@yahoo.com

Abstract. This paper proposes a sparse modeling approach to solve ordinal regression problems using Gaussian processes (GP). Designing a sparse GP model is important from training time and inference time viewpoints. We first propose a variant of the Gaussian process ordinal regression (GPOR) approach, leave-one-out GPOR (LOO-GPOR). It performs model selection using the leave-one-out cross-validation (LOO-CV) technique. We then provide an approach to design a sparse model for GPOR. The sparse GPOR model reduces computational time and storage requirements. Further, it provides faster inference. We compare the proposed approaches with the state-of-the-art GPOR approach on some benchmark data sets. Experimental results show that the proposed approaches are competitive.

Keywords: Gaussian processes, ordinal regression, sparse models.

1 Introduction

In ordinal regression problems, examples are labeled from a discrete and ordered set. We consider an ordinal regression problem with r ordered categories denoted by r consecutive integers $Y = \{1, 2, \ldots, r\}$. Given a sample of n labeled independent training examples, $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$, where x_i is an element of a d dimensional input space X ($X \subseteq \mathcal{R}^d$) and $y_i \in Y$, the goal is to learn a decision function $h : X \rightarrow Y$, which generalizes well.

Ordinal regression problems have recently received a lot of interest from the machine learning community. Herbrich et al. [5] proposed a distribution independent learning approach based on a loss function between pairs of examples. Shashua and Levin [8] proposed fixed margin and sums of margin approaches to solve the ordinal regression problem using the support vector machine framework. However the thresholds learnt to perform ordinal regression with this approach need not be ordered. Chu and Keerthi [3] proposed a new formulation which performs implicit ordering of the thresholds for ordinal variables. Recently, Sun et al. [10] extended Kernel discriminant learning for classification to the ordinal regression setting. The sparse Bayesian ordinal regression [1] approach used the proportional odds model and obtained a sparse solution by imposing a zero-mean Gaussian prior distribution over the weight vector.

T. Huang et al. (Eds.): ICONIP 2012, Part II, LNCS 7664, pp. 409–416, 2012.

Gaussian process ordinal regression (GPOR) [2] uses a new non Gaussian likelihood function for modeling the ordinal labels. It performs model selection by maximizing the marginal likelihood. The GPOR approach is among the state-of-the-art algorithms for ordinal regression. However it is not directly applicable to large data sets since it uses all the training set examples to make predictions, thereby resulting in high training time and slow inference.

In this work, we propose a new approach for ordinal regression using Gaussian processes. The proposed approach, leave-one-out GPOR (LOO-GPOR), employs the leave-one-out cross-validation (LOO-CV)[7] technique for model selection. This technique is easier to implement than the Bayesian techniques employed by the GPOR [2] approach. We also propose a sparse GPOR modeling approach, which makes use of a fewer number (user specified) of training set examples. This reduces the training time and storage requirements needed by the full model. It also improves inference speed. On eight real world benchmark datasets the performance of the proposed LOO-GPOR and sparse GPOR models were comparable with that of the GPOR [2] model. Further, the sparse GPOR model achieved the performance using only 20% of training examples as the basis vectors.

The rest of the paper is organized as follows. Section 2 describes the Gaussian process ordinal regression (GPOR)[2] approach. Section 3 discusses the LOO-CV based model selection approach for GPOR (LOO-GPOR). Section 4 describes an approach to develop sparse models for GPOR. Section 5 presents the experimental results of running the proposed approaches on some benchmark data sets.

2 Gaussian Process Ordinal Regression

Gaussian process ordinal regression (GPOR)[2] uses a new likelihood function for the ordinal outputs. Under noisy observations, for an input x and the latent function f, the GPOR likelihood for an ordinal output y is $p(y|f) = \Phi\big(\frac{b_y - f}{\sigma}\big) - \Phi\big(\frac{b_{y-1} - f}{\sigma}\big)$, where σ is the standard deviation of the Gaussian noise and Φ is the Gaussian cumulative distribution function *i.e.* $\Phi(z) = \int_{-\infty}^{z} \mathcal{N}(\delta; 0, 1) d\delta$. The thresholds $b_0, b_1, \ldots, b_r \in \mathcal{R}$ are such that $b_0 \leq b_1 \leq \ldots \leq b_r$. We fix $b_0 = -\infty$ and $b_r = \infty$ so that the likelihood function represents a valid probability distribution over the ordinal outputs. The thresholds $b_1, b_2, \ldots, b_{r-1}$ divide a real line into r contiguous intervals. A real latent function value is mapped to a discrete ordinal output based on the interval it lies. The GPOR approach uses a zero mean Gaussian process prior. However, the GPOR likelihood is not a Gaussian and hence, the posterior $p(\mathbf{f}|\mathcal{D})$ is also not a Gaussian. The GPOR approach works by approximating the posterior as a Gaussian distribution using the Laplace approximation [7] or the expectation propagation (EP) [6] techniques. In GPOR, model selection is performed by maximizing the evidence $p(\mathcal{D}|\theta)$, where θ is the model parameter vector which includes the parameter κ in the covariance function[1], the threshold parameters $b_1, b_2, \ldots, b_{r-1}$ and the noise parameter σ in the

[1] GPOR uses a squared exponential covariance function, $K(x_i, x_j) = \exp(-\frac{\kappa}{2}||x_i - x_j||^2)$.

likelihood function. It uses two approaches for model selection, the maximum a posteriori approach (MAP-GPOR) with Laplace approximation and the expectation propagation approach (EP-GPOR) with variational methods . In GPOR, model selection takes $\mathcal{O}(n^3)$ time since it employs a gradient based optimization method which requires inversion of an $n \times n$ matrix.

3 Leave-One-Out Gaussian Process Ordinal Regression

The leave-one-out Gaussian process ordinal regression (LOO-GPOR) approach is based on the GPOR model but uses the leave-one-out cross-validation (LOO-CV)[7] technique for model selection. In the LOO-GPOR approach, the model parameters are estimated by minimizing the sum of the leave-one-out negative log predictive (NLP)[7] measure over all the training examples. The LOO-GPOR approach uses the GPOR likelihood which results in a non-Gaussian posterior distribution. It obtains a Gaussian approximation of the posterior distribution by using the expectation propagation (EP) [6] technique.

The expectation propagation (EP) [6] technique is used to approximate complex distributions which factorizes into a product of terms. It is used to approximate the non Gaussian posterior $p(f|X,y) = \prod_{i=1}^{n} p(y_i|f(x_i))p(f|X)$ as a Gaussian distribution $q(f) = \prod_{i=1}^{n} \tilde{t}_i(f(x_i)|\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2)\ p(f|X)$, by approximating each non-Gaussian likelihood term $p(y_i|f(x_i))$ as an unnormalized Gaussian $\tilde{t}_i(f(x_i)|\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \tilde{Z}_i \mathcal{N}(f(x_i)|\tilde{\mu}_i, \tilde{\sigma}_i^2)$. The parameters of each likelihood approximation are found iteratively by minimizing the Kullback-Leibler divergence [6] between the posterior using the exact likelihood term and the approximated likelihood term. An iteration i of the EP approach consists of finding the marginal cavity distribution of the i^{th} training example, $q_{-i}(f(x_i))$. It is obtained by leaving out the i^{th} likelihood term and marginalizing over the remaining variables, *i.e.* $q_{-i}(f(x_i)) \propto \int p(f|X) \prod_{j \neq i} \tilde{t_j}(f(x_j)|\tilde{Z_j}, \tilde{\mu_j}, \tilde{\sigma_j^2}) df_j$. It is also obtained by dividing the approximate posterior $q(f)$ by the i^{th} local likelihood approximation $\tilde{t}_i(f(x_i))$. The Gaussian approximation of the posterior has covariance $\mathbf{\Sigma} = (\mathbf{K}^{-1} + \tilde{\mathbf{\Sigma}}^{-1})^{-1}$, where $\mathbf{K}$ is the covariance matrix formed by all training examples and $\tilde{\mathbf{\Sigma}} = diag(\tilde{\sigma}_1^2, \tilde{\sigma}_2^2, \ldots \tilde{\sigma}_n^2)$. The mean of the posterior is $\boldsymbol{\mu} = \mathbf{\Sigma}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\boldsymbol{\mu}}$, where $\tilde{\boldsymbol{\mu}}$ is a vector of $\tilde{\mu}_i$. Then the marginal cavity distribution for the i^{th} training example is Gaussian with mean μ_{-i} and variance σ_{-i}^2, where

$$\sigma_{-i}^2 = \frac{\mathbf{\Sigma}_{ii}}{1 - \tilde{\sigma}_i^{-2}\mathbf{\Sigma}_{ii}}, \quad \mu_{-i} = \boldsymbol{\mu}_i + \sigma_{-i}^2 \tilde{\sigma}_i^{-2}(\boldsymbol{\mu}_i - \tilde{\mu}_i). \tag{1}$$

The negative log predictive (NLP) measure of the i^{th} training example x_i, when learnt using remaining training examples is defined as $-\log p(y_i|\mathbf{X}, \mathbf{y_{-i}}, \theta) = -\log\big(\Phi\big(\frac{b_{y_i} - \mu_{-i}}{\sqrt{\sigma^2 + \sigma_{-i}^2}}\big) - \Phi\big(\frac{b_{y_i - 1} - \mu_{-i}}{\sqrt{\sigma^2 + \sigma_{-i}^2}}\big)\big)$, where $\mathbf{y_{-i}}$ is the output vector of the training examples except i. Here μ_{-i} and σ_{-i}^2 are the leave-one-out predictive mean and variance of the i^{th} training example, when learnt using the remaining training examples. It is same as the mean and variance of the marginal cavity distribution obtained by leaving out the i^{th} likelihood term. Therefore the leave-one-out

predictive mean and variance are given by (1). The LOO-GPOR approach minimizes the sum of the NLP measure over all the training examples, to obtain the optimal model parameters. The optimization problem is defined as follows.

$$(\theta^*) = \arg\min_{\theta} \mathcal{L}(\theta) = \arg\min_{\theta} - \sum_{i=1}^{n} \log p(y_i|\mathbf{X}, \mathbf{y_{-i}}, \theta) =$$

$$\arg\min_{b_1, \Delta_2, \ldots, \Delta_{r-1}, \kappa, \sigma^2} - \sum_{i=1}^{n} \log\left(\Phi\left(\frac{b_{y_i} - \mu_{-i}}{\sqrt{\sigma^2 + \sigma^2_{-i}}} \right) - \Phi\left(\frac{b_{y_i - 1} - \mu_{-i}}{\sqrt{\sigma^2 + \sigma^2_{-i}}} \right) \right)$$

$$\text{s.t.} \quad b_j = b_1 + \sum_{l=2}^{j} \Delta_l \quad \forall j = 2, \ldots, r-1, \quad \Delta_l \geq 0 \quad \forall l = 2, \ldots, r-1. \quad (2)$$

Here the constraint, $b_1 \leq b_2 \leq \ldots \leq b_{r-1}$, is imposed by using positive padding variables Δ_l. Predictions are made by selecting the ordinal category with the highest probability, using the optimal model parameters [2].

The computational time of the LOO-GPOR approach is same as that of the GPOR approach. EP approximation takes $\mathcal{O}(n^3)$ time. Once we have the Gaussian approximation of the posterior, calculation of the NLP measure over all training samples take $\mathcal{O}(n)$ time. The optimization routine takes $\mathcal{O}(n^3)$ time. Hence the total time taken by the LOO-GPOR approach is $\mathcal{O}(n^3)$.

4 Sparse Gaussian Process Ordinal Regression

The ordinal regression models using the Gaussian processes discussed in sections 2 and 3 require $\mathcal{O}(n^3)$ training time and $\mathcal{O}(n^2)$ storage space. It is computationally expensive to use them on large data sets. The sparse modeling approaches use a representative data set (basis vector set) of size s smaller than the training data set to make predictions. It reduces the training time and storage requirements to $\mathcal{O}(ns^2)$ and $\mathcal{O}(ns)$ respectively. We provide an approach to develop a sparse approximation to the GPOR model. The approach is based on the validation based sparse approximation for the Gaussian process classification[9]. The advantage of the proposed sparse model over sparse Bayesian ordinal regression [1] is that the basis vector set size can be fixed by a user defined parameter.

The proposed algorithm uses a two loop approach: 1) basis vector selection, and 2) hyper-parameter adaptation. The basis vector selection loop incrementally selects a set of basis vectors from the training data set. Hyper-parameter adaptation loop performs model selection using the selected basis vectors. The details are presented in Algorithm 1. An advantage of this algorithm is that both the loops make use of the same measure, NLP (negative log predictive), which takes into account moderated probability score, easy to calculate for a GP model.

Let the training data in the basis vector set be denoted by $\mathbf{u}$, the remaining training data be denoted by $\mathbf{u}^c$, the size of $\mathbf{u}$ be denoted by $|\mathbf{u}|$ and the size of $\mathbf{u}^c$ be denoted by $|\mathbf{u}^c|$. Let the maximum size of the basis vector set be a user

Algorithm 1. sparse GPOR algorithm

Input: Basis vector size s
Output: Basis vector u, hyper parameter vector $\theta = (b_1, b_2, \ldots, b_{r-1}, \kappa, \sigma^2)$
Initialize: Hyper parameter vector $\theta = (b_1, b_2, \ldots, b_{r-1}, \kappa, \sigma^2)$.

1. Initialize $\mathbf{A} = \mathbf{K}$, $\mathbf{u} = \Phi$, $\mathbf{u}^c = \{1, 2, \ldots, n\}$, $\hat{\mathbf{f}}_\mathbf{i} = \mathbf{p}_\mathbf{i} = \mathbf{m}_\mathbf{i} = 0 \quad \forall i \in \mathbf{u}^c$.
2. Form a working set $\mathcal{J}$ from $\mathbf{u}^c$.
3. For each $j \in \mathcal{J}$
   (a) Obtain the site parameters m_j and p_j using (5), update $\bar{u}_j = u \cup \{j\}$.
   (b) Update posterior mean $\hat{\mathbf{f}}$ and covariance $\mathbf{A}$.
   (c) Calculate $NLP(\bar{u}_j, \theta) = -\frac{1}{|\bar{u}_j^c|} \sum_{k \in \bar{u}_j^c} \log\Big(\Phi\Big(\frac{b_{y_k} - \hat{\mathbf{f}}_k}{\sqrt{\sigma^2 + \mathbf{A}_{kk}}}\Big) - \Phi\Big(\frac{b_{y_k - 1} - \hat{\mathbf{f}}_k}{\sqrt{\sigma^2 + \mathbf{A}_{kk}}}\Big)\Big)$
4. Choose $i = \arg\min_{j \in \mathcal{J}} NLP(\bar{u}_j, \theta)$
5. Set $u = u \cup \{i\}$, $u^c = u^c - \{i\}$, and update posterior mean $\hat{\mathbf{f}}$ and covariance $\mathbf{A}$ to reflect the inclusion of chosen point in the basis set.
6. If $|u| < s$, go to step 2
7. Estimate the hyper parameters by minimizing $NLP(u, \theta)$.
8. Terminate if stopping criterion is satisfied. Otherwise go to step 1.

supplied parameter s ($s \ll n$). In sparse GPOR, the NLP measure with respect to the set $\mathbf{u}$ is defined as

$$NLP(\mathbf{u}, \theta) = -\frac{1}{|\mathbf{u}^c|} \sum_{i \in \mathbf{u}^c} \log\left(\Phi\left(\frac{b_{y_i} - \mu_i}{\sqrt{\sigma^2 + \sigma_i^2}}\right) - \Phi\left(\frac{b_{y_i - 1} - \mu_i}{\sqrt{\sigma^2 + \sigma_i^2}}\right)\right). \tag{3}$$

Here the NLP measure is calculated on the training examples present in $\mathbf{u}^c$. μ_i and σ_i^2 denote the predictive mean and variance respectively of a data i in $\mathbf{u}^c$. The predictive mean and variance are obtained from the mean and covariance of the posterior distribution for the training latent functions, $p(\mathbf{f}|\mathcal{D}, \theta) \propto \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}) \prod_{i=1}^n p(y_i|f_i)$. The non Gaussian posterior $p(\mathbf{f}|\mathcal{D}, \theta)$ is approximated incrementally as a Gaussian posterior $q(\mathbf{f}|\mathcal{D}, \theta)$ using the assumed density filtering (ADF) [6] technique.

$$p(\mathbf{f}|\mathcal{D}, \theta) \approx q(\mathbf{f}|\mathcal{D}, \theta) \propto \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}) \prod_{i=1}^{n} \exp\Big\{-\frac{p_i}{2}(f_i - m_i)^2\Big\} = \mathcal{N}(\mathbf{f}|\hat{\mathbf{f}}, \mathbf{A}) \tag{4}$$

where $\mathbf{A} = (\mathbf{K^{-1}} + \mathbf{\Pi})^{-1}$, $\hat{\mathbf{f}} = \mathbf{A\Pi m}$, $\mathbf{m} = (m_1, \ldots, m_n)$, and $\mathbf{\Pi} = \text{diag}\ (p_1, \ldots, p_n)$. The variables m_i and p_i are called the site parameters corresponding to the Gaussian approximation of the likelihood $p(y_i|f_i)$ of the i^{th} training example. The site parameters of the training examples not belonging to the basis vector are set to zero. The predictive mean (μ_i) and variance (σ_i^2) of a training example i are given by $\hat{\mathbf{f}}_i$ and $\mathbf{A}_{ii}$ respectively. Upon inclusion of the training example i to the basis vector set we have to update the site parameters corresponding to it as well as posterior mean $\hat{\mathbf{f}}$ and covariance $\mathbf{A}$. Site parameters m_i and p_i are updated as follows [2].

Table 1. Properties of benchmark data sets

Dataset	Attributes	Training Instances	Test Instances
Diabetes	2	30	13
Triazines	60	100	86
Wisconsin	32	130	64
Machine	6	150	59
AutoMPG	7	200	192
Boston	13	300	206
Stocks	9	600	350
Abalone	8	1000	3177

Table 2. Average rank and the Friedman statistic for different approaches

	5 bins		10 bins	
Approach	zero-one	absolute	zero-one	absolute
MAP-GPOR	2.1875	1.875	2.875	2.125
EP-GPOR	2.4375	2	1.875	2
LOO-GPOR	2.25	2.625	2.125	2.625
sparse GPOR	3.125	3.5	3.125	3.25
F_F	0.8735	3.4672	1.8889	1.6823

$$\tilde{z}_{i1} = \frac{b_{y_i} - \mu_i}{\sqrt{\sigma^2 + \sigma_i^2}} \quad ; \quad \tilde{z}_{i2} = \frac{b_{y_i-1} - \mu_i}{\sqrt{\sigma^2 + \sigma_i^2}} \quad ; \quad \mathcal{Z}_i = \Phi(\tilde{z}_{i1}) - \Phi(\tilde{z}_{i2})$$
$$\gamma_i = \frac{\partial \log \mathcal{Z}_i}{\partial \mu_i} = \frac{-1}{\sqrt{\sigma^2 + \sigma_i^2}} \left(\frac{\mathcal{N}(\tilde{z}_{i1}; 0, 1) - \mathcal{N}(\tilde{z}_{i2}; 0, 1)}{\Phi(\tilde{z}_{i1}) - \Phi(\tilde{z}_{i2})} \right)$$
$$\beta_i = \frac{\partial \log \mathcal{Z}_i}{\partial \sigma_i^2} = \frac{-1}{2(\sigma^2 + \sigma_i^2)} \left(\frac{\tilde{z}_{i1}\mathcal{N}(\tilde{z}_{i1}; 0, 1) - \tilde{z}_{i2}\mathcal{N}(\tilde{z}_{i2}; 0, 1)}{\Phi(\tilde{z}_{i1}) - \Phi(\tilde{z}_{i2})} \right)$$
$$\nu_i = \gamma_i^2 - 2\beta_i \quad ; \quad p_i = \frac{\nu_i}{1 - \sigma_i^2 \nu_i} \quad ; \quad m_i = \mu_i + \frac{\gamma_i}{\nu_i}. \tag{5}$$

The posterior covariance $\mathbf{A}$ is updated by maintaining the decomposition mentioned in [9] which takes only $\mathcal{O}(n|\mathbf{u}|)$ time. Such a decomposition avoids expensive matrix inverse operation required on every basis addition. If all the training examples in $\mathbf{u}^c$ are tested for possible inclusion in the basis vector set, computational effort needed to select a single basis vector is $\mathcal{O}(n^2|\mathbf{u}|)$. Instead we select a basis vector from a working set $\mathcal{J}$ of size k, where $\mathcal{J} \subseteq \mathbf{u}^c$. Working set consists of randomly chosen k elements from $\mathbf{u}^c$. Then the computational time to select a single basis vector reduces to $\mathcal{O}(n|\mathbf{u}|k)$. If we fix the maximum size of the basis vector set to s, then the basis vector selection loop takes $\mathcal{O}(ns^2k)$ time. Once the basis vectors are selected hyper-parameters adaptation takes $\mathcal{O}(s^3)$ time. Hence the time complexity of the sparse GPOR approach is $\mathcal{O}(ns^2k)$.

5 Experimental Results

We report the experimental results of our approach on 8 benchmark data sets [2]. Properties of these benchmark data sets are summarized in Table 1. These are metric regression data sets. The continuous target values in these data sets are transformed into ordinal values as discussed in [2]. We conduct experiments on two versions of the data sets, 5 bins and 10 bins. In the former, the number of ordinal categories is 5 while in latter, it is 10. Each data set is randomly partitioned into 20 training and test data set instances.

The optimum model parameter values are obtained by solving the optimization problems (2) and (3). They are solved using a gradient based unconstrained optimization technique. The optimization is run with random as well as fixed [2] initialization of the optimization variables; we report the result for which the objective function value is the least. For the sparse model design experiments, active set size

Table 3. Comparison of results of LOO-GPOR and sparse GPOR with MAP-GPOR and EP-GPOR on benchmark data sets for 5 bins version

	Mean zero-one error(%)				Mean absolute error			
Data	MAP-GPOR	EP-GPOR	LOO-GPOR	sparse GPOR	MAP-GPOR	EP-GPOR	LOO-GPOR	sparse GPOR
Diabetes	54.23±13.78	54.23±13.78	**49.62±13.3**	52.69±12.30	0.66±0.14	0.67±0.14	**0.65±0.18**	0.72±0.23
Triazines	52.91±2.15	**52.62±2.66**	54.36±1.50	56.74±4.09	**0.69±0.02**	0.69±0.03	0.70±0.02	0.75±0.05
Wisconsin	65.00±4.71	65.16±4.65	**64.06±2.53**	64.06±4.67	**1.01±0.09**	1.01±0.09	1.12±0.07	1.10±0.12
Machine	**16.53±3.56**	16.78±3.88	16.61±4.25	16.61±4.06	**0.19±0.04**	0.19±0.04	0.19±0.05	0.19±0.06
AutoMPG	23.78±1.85	**23.75±1.74**	25.78±2.65	25.44±2.13	0.24±0.02	**0.24±0.02**	0.27±0.03	0.26±0.02
Boston	24.88±2.02	**24.49±1.85**	24.85±2.90	26.75±3.04	0.26±0.02	**0.26±0.02**	0.26±0.03	0.29±0.03
Stocks	11.99±2.34	12.00±2.06	**10.60±1.69**	13.69±1.67	0.12±0.02	0.12±0.02	**0.11±0.02**	0.14±0.02
Abalone	**21.50±0.22**	21.56±0.36	21.85±0.29	22.41±0.42	**0.23±0.00**	0.23±0.01	0.24±0.00	0.24±0.01

Table 4. Comparison of results of LOO-GPOR and sparse GPOR with MAP-GPOR and EP-GPOR on benchmark data sets for 10 bins version

	Mean zero-one error(%)				Mean absolute error			
Data	MAP-GPOR	EP-GPOR	LOO-GPOR	sparse GPOR	MAP-GPOR	EP-GPOR	LOO-GPOR	sparse GPOR
Diabetes	83.46±5.73	83.08±5.91	**71.54±10.9**	74.23±14.18	2.14±0.33	2.14±0.33	**1.20±0.31**	1.57±0.92
Triazines	**63.72±4.34**	64.01±3.78	72.21±1.55	71.63±3.17	**1.20±0.07**	1.20±0.07	1.35±0.04	1.38±0.12
Wisconsin	78.52±3.58	78.52±3.51	**75.94±2.45**	76.80±3.73	**2.14±0.18**	2.14±0.18	2.74±0.14	2.58±0.21
Machine	33.81±3.91	**33.73±3.64**	34.24±3.19	33.73±5.84	0.48±0.07	0.47±0.08	0.50±0.07	**0.46±0.07**
AutoMPG	43.96±2.81	**43.88±2.60**	44.30±2.26	46.67±2.26	0.50±0.04	**0.50±0.03**	0.50±0.03	0.58±0.04
Boston	41.53±2.77	41.26±2.86	**41.04±2.25**	42.48±3.29	0.49±0.03	**0.49±0.04**	0.50±0.04	0.55±0.05
Stocks	19.90±1.72	19.44±1.91	**18.74±2.38**	22.46±1.97	0.20±0.02	0.20±0.02	**0.19±0.02**	0.23±0.02
Abalone	42.60±0.91	**42.27±0.46**	42.80±0.49	43.91±0.81	0.51±0.01	**0.51±0.01**	0.53±0.01	0.54±0.01

is taken to be $.2n$. Also we fix the working set size k as $k = min(|\mathbf{u}^c|, 59)$[9]. We use the Gaussian kernel $K(x_i, x_j) = \exp\left(-\frac{\kappa}{2}||x_i - x_j||^2\right)$ in all our experiments.

We compare the generalization performance of the proposed approaches, LOO-GPOR and sparse GPOR, with MAP-GPOR and EP-GPOR results reported in [2]. We use two evaluation metrics to compare the performance, *zero-one error* and *absolute error* [2]. The former gives the fraction of incorrect predictions on test data while the latter gives the average deviation of the predicted test outputs from the actual test outputs. The mean of the zero-one and absolute errors over all the 20 instances are used to compare the performance of the approaches. The mean zero-one errors are reported in percentage and the mean absolute errors are rounded off to 2 decimal places. We prefer methods with low mean zero-one and mean absolute errors. Tables 3 and 4 compare the results of LOO-GPOR and sparse GPOR with MAP-GPOR and EP-GPOR for 5 bins and 10 bins cases respectively.

We observe from Tables 3 and 4 that the results obtained with the proposed approaches are comparable with the existing approaches, MAP-GPOR and EP-GPOR. LOO-GPOR is found to perform better than the existing approaches on three data sets, Diabetes, Wisconsin and Stocks. Sparse GPOR also performs

satisfactory on all the data sets. It performs better than the existing approaches on Diabetes and Wisconsin data sets.

Note that the sparse GPOR approach used only 20% of the total examples ($s = .2n$) as the basis vectors for model design and prediction. This is much less than the number (n) used by each of the other three (full model) approaches. As mentioned earlier, the model selection (Step 7 in Algorithm 1) for a given basis vector set is done using training set examples *not* in the basis vector set. Thus, increase in the number of basis vectors used by the sparse GPOR approach further, will affect model selection and hence, the generalization performance. We observed this in our experiments and therefore restricted s to $.2n$. This number of basis vectors may not be sufficient for some data sets, e.g. Stocks data set. In such cases, it may be a good idea to use full model ordinal regression.

We check whether the performance of the proposed approaches differ significantly from the existing GPOR approaches using the Friedman test [4]. Here we compare 4 approaches on 8 data sets and hence the F distribution has 3 and 21 degrees of freedom. For the level of significance $\alpha = 0.05$, critical F value is 3.07. Table 2 reports the average ranks of the Gaussian process ordinal regression approaches over all the data sets and the Friedman statistic F_F [4] computed over all the approaches. F_F is greater than the critical F value only for the 5 bins case with respect to absolute error. In all other cases it is below the critical F value (due to the ranks being similar) and hence there does not exist any significant differences among the approaches. Thus the proposed sparse GPOR approach is competitive.

References

1. Chang, X., Zheng, Q., Lin, P.: Ordinal Regression with Sparse Bayesian. In: Huang, D.-S., Jo, K.-H., Lee, H.-H., Kang, H.-J., Bevilacqua, V. (eds.) ICIC 2009. LNCS, vol. 5755, pp. 591–599. Springer, Heidelberg (2009)
2. Chu, W., Ghahramani, Z.: Gaussian Processes for Ordinal Regression. J. Mach. Learn. Res. 6, 1019–1041 (2005)
3. Chu, W., Keerthi, S.S.: New Approaches to Support Vector Ordinal Regression. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 145–152. ACM (2005)
4. Demsar, J.: Statistical Comparisons of Classifiers over Multiple Data Sets. J. Mach. Learn. Res. 7, 1–30 (2006)
5. Herbrich, R., Graepel, T., Obermayer, K.: Large Margin Rank Boundaries for Ordinal Regression. In: Advances in Large Margin Classifiers. MIT Press (2000)
6. Minka, T.: A Family of Algorithms for Approximate Bayesian Inference. Ph.D. thesis, Massachusetts Institute of Technology (2001)
7. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press (2005)
8. Shashua, A., Levin, A.: Ranking with Large Margin Principle: Two Approaches. In: Advances in Neural Information Processing Systems 15, pp. 937–944. The MIT Press (2003)
9. Shevade, S., Sundararajan, S.: Validation-Based Sparse Gaussian Process Classifier Design. Neural Computation 21, 2082–2103 (2009)
10. Sun, B.Y., Li, J., Wu, D.D., Zhang, X.M., Li, W.B.: Kernel Discriminant Learning for Ordinal Regression. IEEE Trans. on Knowl. and Data Eng. 22, 906–910 (2010)