# Efficient Online Classification Using an Ensemble of Bayesian Linear Logistic Regressors

Narayanan U. Edakunni and Sethu Vijayakumar

School of Informatics, University of Edinburgh
n.u.edakunni@sms.ed.ac.uk, sethu.vijayakumar@ed.ac.uk

**Abstract.** We present a novel ensemble of logistic linear regressors that combines the robustness of online Bayesian learning with the flexibility of ensembles. The ensemble of classifiers are built on top of a Randomly Varying Coefficient model designed for online regression with the fusion of classifiers done at the level of regression before converting it into a class label using a logistic link function. The locally weighted logistic regressor is compared against the state-of-the-art methods to reveal its excellent generalization performance with low time and space complexities.

## 1 Introduction

Research in classification has been dominated by kernel based methods like Support Vector Machine (SVM)[9] and more recently by non-parametric methods like Gaussian Process Classification (GP)[6]. Non-parametric methods like GP derives its success by using a covariance function of the input to model the dependency amongst the responses. The response for a test input is then computed as a linear smooth of all the training responses. This in turn leads to a large overhead in the time and space complexities for training and prediction. The need to store away all of the training points in order to produce a prediction for an unseen input makes the kernel machines ill suited for online learning. On the other hand ensemble learning paradigm is ideal for online learning where the learning model has to adjust its complexity in tune with the training data. When new data is observed from a new region of input space an ensemble learner can add a new model to the region of space and adjust its parameters to model that particular region of space. To implement such an ensemble we need classifiers that have varying responsibilities in different regions of space and are able to learn independent of each other. Conventional ensemble learners combine the classifier predictions either by a majority voting or by linear combination of the votes. This would not work when the ensembles have different levels of confidence over the input space. The combined prediction would be more robust if the predictions are weighted by confidence of individual classifiers before combining them. In this paper we use a linear logistic regression as the base classifier with Bayesian learning for the regression. The combination of the predictions is done at the regression level wherein each learner is endowed with a predictive distribution and the variances of the distribution are used to weigh the predictions of the base learners. The real valued output of the regression is then converted to a class label using a logistic link function. In this paper we combine the robustness and efficiency of a Bayesian learning with the flexibility of ensemble learning to produce an online classifier that is able to adapt its complexity in tune with the observed data. We reuse a Randomly Varying Coefficient model [3] designed for regression to build our ensemble classifier system.

## 2 Randomly Varying Coefficient Model

The Randomly Varying Coefficient model approximates a multivariate non-linear function using a set of local models. Each of the local models has a probabilistic formulation with a parametric model for the *linear fit* and the *extent* of linearity at a particular location in the input space. The strength of RVC arises from the fact that the local models are trained independent of each other unlike [5]. Apart from minimizing the interference between two models, this also allows models to be dynamically allocated and deallocated without the need for relearning. Furthermore, the probabilistic formulation of RVC provides an estimate of the uncertainty in its prediction and allows Bayesian inference rules to be applied in order to learn the parameters.The end result of an RVC is a set of linear regression models distributed in the input space that have different confidence in their predictions in different regions of the input space.

## 3 Local Logistic Regression

In this section, we introduce the probabilistic model for a locally weighted logistic regression based on the probabilistic formulation of RVC called the *logistic RVC (lRVC)*. Here, the probability of a binary class variable $z_i$ is modeled as the output of a logistic link function over a continuous latent variable $y_i$ as :

$$p(z_i = 1|y_i) = 1/(1 + exp(-y_i)), \quad i = 1 \dots N \tag{1}$$

where $i$ is the index over the training data. In turn, the latent variable $y_i$ is modeled as the response of a locally linear regression that follows the RVC formulation. For a locally linear region centered around $\mathbf{x}_c$ a conditional model for the continuous latent variable $y_i$ can be written as:

$$y_i = \boldsymbol{\beta}_i^T \mathbf{x}_i + \epsilon \tag{2}$$

where $\mathbf{x}_i \equiv [(\mathbf{x}'_i - \mathbf{x}_c)^T, 1]^T$ represents the center subtracted, bias augmented $d$ dimensional input vector, $\boldsymbol{\beta}_i \equiv [\beta_i^{(1)} \dots \beta_i^{(d+1)}]^T$ represents the corresponding regression coefficient and $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is the Gaussian mean zero noise with a variance $\sigma^2$. Crucially the latent regression coefficient $\boldsymbol{\beta}_i$ has a Gaussian distribution :

$$\boldsymbol{\beta}_i \sim \mathcal{N}(\hat{\boldsymbol{\beta}}, \mathbf{C}_i) \tag{3}$$

where the magnitude of the covariance $\mathbf{C}_i$ is made proportional to the distance of $\mathbf{x}'_i$ from the center by modeling it as a diagonal covariance with the elements being a quadratic function of the input :

$$\mathbf{C}_i(j, j) = \mathbf{x}_i^T \mathbf{x}_i / h_j^2, \quad j = 1 \dots d \tag{4}$$

This has the effect that for points that lie near the center, the latent regression coefficients $\boldsymbol{\beta}_i$ have similar values, with the distribution of $\boldsymbol{\beta}_i$ being peaked around $\hat{\boldsymbol{\beta}}$. This in turn results in a linear region around the center.

Proceeding along the same lines as Chapter 3 of [6] we now assume a noise-free latent variable $y_i$ by setting $\sigma^2$ to zero. Setting $\sigma^2$ to zero yields the following model for $y_i$ :

$$y_i \sim \mathcal{N}(\boldsymbol{\beta}_i^T \mathbf{x}_i, 0) \tag{5}$$

or equivalently marginalizing $y_i$ :

$$p(z_i = 1|\boldsymbol{\beta}_i) = 1/(1 + exp(-\boldsymbol{\beta}_i^T \mathbf{x}_i)) \tag{6}$$

which corresponds to the classical formulation of a linear logistic regression with regression coefficients $\boldsymbol{\beta}_i$.

We preserve the same probabilistic model as the original RVC for the rest of parameters. This includes a Gamma distribution as a *regularizer* prior over the bandwidth parameters :

$$h_j^2 \sim Gamma(a_j, b_j) \tag{7}$$

and a noninformative Normal prior $\mathcal{N}(\boldsymbol{\mu}, \mathbf{S})$ for the parameter $\hat{\boldsymbol{\beta}}$.

We can now infer the posterior distribution over the parameters of the model using a Variational Bayesian EM similar to [3]. In this procedure, the posterior distribution over the parameters are assumed to be independent and these distributions are iteratively determined one at a time by fixing all other distributions.

For a logistic regression, an additional complication arises in the computation of the posterior distribution over the hidden variables $\boldsymbol{\beta}_i$. The likelihood term $P(z_i|\boldsymbol{\beta}_i)$ is given by a logistic link function whereas the prior over $\boldsymbol{\beta}_i$ is Gaussian and is not conjugate to the likelihood term. We solve this issue by using a Laplacian approximation to approximate the posterior over $\boldsymbol{\beta}_i$ by a Gaussian distribution. The log posterior over the hidden variable $\boldsymbol{\beta}_i$ is given by :

$$\begin{aligned}\mathcal{M} = \ln Q(\boldsymbol{\beta}_i|\mathbf{z}) &= \ln P(z_i|\boldsymbol{\beta}_i) + \left\langle \ln P(\boldsymbol{\beta}_i|\hat{\boldsymbol{\beta}}, \mathbf{C}_i) \right\rangle_{Q(\hat{\boldsymbol{\beta}}),Q(\mathbf{h})} \\ &- \ln \int exp\left( \ln P(z_i|\boldsymbol{\beta}_i) + \left\langle \ln P(\boldsymbol{\beta}_i|\hat{\boldsymbol{\beta}}, \mathbf{C}_i) \right\rangle_{Q(\hat{\boldsymbol{\beta}}),Q(\mathbf{h})} \right)\end{aligned} \tag{8}$$

Laplace approximation of the posterior corresponds to

$$Q(\boldsymbol{\beta}_i|\mathbf{z}) \sim \mathcal{N}(\boldsymbol{\nu}_i, \mathbf{G}_i)$$

where $\boldsymbol{\nu}_i = argmax_{\boldsymbol{\beta}_i} \mathcal{M}$ and $\mathbf{G}_i^{-1} = -\nabla\nabla\mathcal{M}|_{\boldsymbol{\beta}_i = \boldsymbol{\nu}_i}$ is the Hessian of the negative log posterior. The posterior mode $\boldsymbol{\nu}_i$ can be obtained by setting the gradient of $\mathcal{M}$ to zero. However, this procedure does not yield a closed form solution for the posterior mode $\boldsymbol{\nu}_i$. We then have to resort to Newton's update to find the mode iteratively as shown :

$$\boldsymbol{\nu}_i = \boldsymbol{\nu}_i^{old} - (\nabla\nabla\mathcal{M})^{-1}\nabla\mathcal{M} \tag{9}$$

Substituting the forms of $P(\boldsymbol{\beta}_i|\hat{\boldsymbol{\beta}}, \mathbf{C}_i)$ and $P(z_i|\boldsymbol{\beta}_i)$ from eqs. (3) and (6) into eq. (8) and differentiating it with respect to $\boldsymbol{\beta}_i$ we get :

$$\nabla\mathcal{M}\ |_{\boldsymbol{\beta}_i = \boldsymbol{\nu}_i^{old}} = \mathbf{x}_i(z_i - \pi_i) - \langle \mathbf{C}_i \rangle^{-1} (\boldsymbol{\nu}_i^{old} - \tilde{\boldsymbol{\mu}}) \tag{10}$$

$$\nabla\nabla\mathcal{M}\ |_{\boldsymbol{\beta}_i = \boldsymbol{\nu}_i^{old}} = -\mathbf{x}_i \mathbf{x}_i^T \pi_i(1 - \pi_i) - \langle \mathbf{C}_i \rangle^{-1} \tag{11}$$

**Algorithm 1.** Training a local model

```
1: Initialize hyperparameters: $\Theta \equiv \{\boldsymbol{\mu}_0, \mathbf{S}, \mathbf{a}, \mathbf{b}\}$.
2: Input: Batch training data $\mathbf{X}$, $\mathbf{z}$
3: repeat
4:    Initialize $\boldsymbol{\nu}_i^{old} = \tilde{\boldsymbol{\mu}}$, $\pi_i = 1/(1 + exp(-\mathbf{x}_i^T \boldsymbol{\nu}_i^{old}))$ and $w_i = \frac{1}{\pi_i(1-\pi_i)}$.
5:    Estimate posterior hyperparameters $\tilde{\Theta}$ using $\Theta$ and eqs. (13) - (16).
6:    Estimate values of the hyperparameters $\mathbf{a}$ and $\mathbf{b}$ of the regularizer prior using
      eq. (17).
7: until convergence of $\tilde{\Theta}$
```

where $\pi_i = 1/(1 + exp(-\mathbf{x}_i^T \boldsymbol{\nu}_i^{old}))$ and $\langle \mathbf{C}_i \rangle = diag(\mathbf{x}_i^T \mathbf{x}_i / \langle h_j^2 \rangle_{Q(h_j^2)})$. It can be found from eq. (11) that $\mathbf{G}_i^{-1} = -\nabla\nabla\mathcal{M} = \mathbf{x}_i \mathbf{x}_i^T \pi_i(1-\pi_i) + \langle \mathbf{C}_i \rangle^{-1}$ and the estimate for $\boldsymbol{\nu}_i$ can be obtained by substituting eqs. (10) and (11) in eq. (9) yielding :

$$\boldsymbol{\nu}_i = \boldsymbol{\nu}_i^{old} + \mathbf{G}_i(\mathbf{x}_i(z_i - \pi_i) - \langle \mathbf{C}_i \rangle^{-1} (\boldsymbol{\nu}_i^{old} - \tilde{\boldsymbol{\mu}})) \tag{12}$$

which can be simplified using Sherman-Morrison Woodbury theorem to yield :

$$\mathbf{G}_i = \langle \mathbf{C}_i \rangle - \frac{\langle \mathbf{C}_i \rangle \mathbf{x}_i \mathbf{x}_i^T \langle \mathbf{C}_i \rangle}{w_i + \mathbf{x}_i^T \langle \mathbf{C}_i \rangle \mathbf{x}_i} \tag{13}$$

$$\boldsymbol{\nu}_i = \frac{\langle \mathbf{C}_i \rangle \mathbf{x}_i}{(w_i + \mathbf{x}_i^T \langle \mathbf{C}_i \rangle \mathbf{x}_i)}((z_i - \pi_i)w_i + \mathbf{x}_i^T \boldsymbol{\nu}_i^{old} - \mathbf{x}_i^T \tilde{\boldsymbol{\mu}}) + \tilde{\boldsymbol{\mu}} \tag{14}$$

where $w_i = \frac{1}{\pi_i(1-\pi_i)}$.

Posterior over $\hat{\boldsymbol{\beta}}$ based on its likelihood and prior can be derived similar to [3] :

$$Q(\hat{\boldsymbol{\beta}}|\mathbf{z}) \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\mathbf{S}})$$

where

$$\tilde{\mathbf{S}} = (\sum_i \langle \mathbf{C}_i \rangle^{-1} + \mathbf{S}^{-1})^{-1}, \ \tilde{\boldsymbol{\mu}} = \tilde{\mathbf{S}}(\sum_i \langle \mathbf{C}_i \rangle^{-1} \boldsymbol{\nu}_i + \mathbf{S}^{-1}\boldsymbol{\mu}) \tag{15}$$

Similarly, the posterior over $h_j$ is given by :

$$Q(h_j^2|\mathbf{z}) \sim Gamma(\tilde{a}_j, \tilde{b}_j)$$

where

$$\tilde{a}_j = a_j + N/2, \ \tilde{b}_j = b_j + \sum_i \left[ (\boldsymbol{\nu}_{i,j} - \tilde{\boldsymbol{\mu}}_{i,j})^2 + \mathbf{G}_{i,jj} + \tilde{\mathbf{S}}_{jj} \right] / 2\mathbf{x}_i^T \mathbf{x}_i \tag{16}$$

and the optimal values for $a_j$ and $b_j$ are obtained by an update rule given by -

$$a_j = \tilde{a}_j, \quad b_j = \tilde{b}_j \tag{17}$$

Hence, posterior parameters are inferred by using a partial Newton step to infer the posterior of $\boldsymbol{\beta}_i$ followed by EM updates as shown in Algorithm 1.

We observe from the training updates that the base classifier is extremely efficient with a complexity of $O(dMN)$ for training, where $d$ is the number of dimensions of the input space, $M$ the number of local models and $N$ the number of training instances.

## 4 Prediction

Using the learning procedure discussed in the previous section we obtain independently trained local models of the logistic regression. Each of the local models represent a separate classifier with a linear decision boundary. To obtain an aggregate prediction for a particular query input we need to combine these classifiers.

Ensemble learning has been a field of research which has seen considerable amount of research into the ways of combining classifiers [4]. In this paper, we use the same technique as RVC - combining the linear regressors to produce a non-linear regression model followed a logistic transform to obtain a classifier.

Given the ensemble of trained local experts, in order to predict the response $y_q$ for a new query point $\mathbf{x}_q$, we take the normalized product of the *predictive distribution* of each local expert. This is similar to the predictive routine in Bayesian Committee Machines [8]. The predictive distribution of each local expert is given by:

$$y_{q,k} \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}^T \mathbf{x}_{q,k}, {\mathbf{x}_{q,k}}^T (\tilde{\mathbf{S}}_k + \mathbf{C}_{k\mathbf{h}_{mode}})\mathbf{x}_{q,k})$$

where $\mathbf{x}_{q,k}$ refers to the query point with the $k$-th center subtracted and augmented with bias. Blending the prediction of different experts by taking their product and normalizing it results in a Normal distribution given by:

$$y_q \sim \mathcal{N}(\mu, \zeta^2) \quad \text{where} \quad \mu = \frac{\sum_k \alpha_k \tilde{\boldsymbol{\mu}}_k^T \mathbf{x}_{q,k}}{\sum_k \alpha_k}, \;\; \zeta^2 = \frac{1}{\sum_k \alpha_k}. \tag{18}$$

Here, $\mu$ is a sum of the means of each individual expert weighted by the confidence expressed by each expert in its own prediction $\alpha_k$, $\zeta^2$ is the variance and $\alpha_k$ is the precision of each expert:

$$\alpha_k = 1/(\mathbf{x}_{q,k}^T (\tilde{\mathbf{S}}_k + \mathbf{C}_k)\mathbf{x}_{q,k}) \tag{19}$$

The predictive probability for the logistic regression can be obtained by combining the predictive probability of the latent variable $y_q$ with the link function and marginalizing the latent variable to yield :

$$P(z_q = 1|\mathbf{z}) = \int P(z_q = 1|y_q)P(y_q|\mathbf{z})dy_q \tag{20}$$

where $P(z_q = 1|y_q)$ is a logistic function and $P(y_q|\mathbf{z})$ is the predictive distribution given by eq. (18). The integral given in eq. (20) cannot be evaluated analytically and we must rely on numerical methods or sampling to evaluate the integral. In the context of binary classification if we threshold the predictive probability at $\frac{1}{2}$ in order to discriminate between classes, a maximum aposteriori(MAP) prediction would be the same as an averaged prediction as shown in [1] and explained in [6]. Therefore we use MAP predictive estimate for classification. To obtain the MAP prediction we evaluate the integral in eq. (20) by approximating $P(y_q|\mathbf{z})$ by a delta function at its mode. The prediction routine is listed in Algorithm 2.

**Algorithm 2.** Global prediction using local models

```
1: Input: Query point $\mathbf{x}_q$
2: Initialize: $sum_\alpha = 0$, $y_q = 0$
3: for k = 1 to #local models do
4:    $\mathbf{x}_{q,k} = \mathbf{x}_q - \mathbf{x}_{c,k}$
5:    Calculate $\alpha_k$ using eq. (19)
6:    $y_q = y_q + \alpha_k \tilde{\boldsymbol{\mu}}_k^T \mathbf{x}_{q,k}$
7:    $sum_\alpha = sum_\alpha + \alpha_k$
8: end for
9: $y_q = y_q / sum_\alpha$
10: Output : $P(z_q = 1) = 1/(1 + exp(-y_q))$
```

## 5 Online Classification

We can use the same technique as in the original RVC [3] to convert the batch updates derived earlier into online updates. For this we make use of the fact that in a Bayesian inference posterior is given by :

$$posterior_N = \prod_i^N (likelihood_i) \times prior_0$$

where $i$ is an index over the data points. The same can be expressed as a set of online updates :

$$posterior_i \propto likelihood_i \times prior_i \quad ; \quad prior_{i+1} = posterior_i$$

This set of updates implies that at every step of the online update the prior computed over the data seen so far is combined with the likelihood of the current data point to yield the posterior. This new posterior distribution of the parameter is then used as the prior during the next update. Based on this, we can derive the online updates for the logistic RVC that correspond to the batch results derived earlier :

$$\tilde{\mathbf{S}}_i = (\langle \mathbf{C}_i \rangle^{-1} + \mathbf{S}_i^{-1})^{-1} \tag{21}$$

$$\tilde{\boldsymbol{\mu}}_i = \tilde{\mathbf{S}}_i (\langle \mathbf{C}_i \rangle^{-1} \boldsymbol{\nu}_i + \mathbf{S}_i^{-1} \boldsymbol{\mu}_i) \tag{22}$$

$$\tilde{a}_{i,j} = a_{i,j} + 1/2 \tag{23}$$

$$\tilde{b}_{i,j} = b_{i,j} + \left[ (\boldsymbol{\nu}_{i,j} - \tilde{\boldsymbol{\mu}}_{i,j})^2 + \mathbf{G}_{i,jj} + \tilde{\mathbf{S}}_{i,jj} \right] / (2\mathbf{x}_i^T \mathbf{x}_i) \tag{24}$$

where $\boldsymbol{\nu}_i$ and $\mathbf{G}_i$ are given by eq. (13). We repeat the above updates for a single data point $\{\mathbf{x}_i, y_i\}$ till the posteriors $\tilde{\Theta} \equiv \{\tilde{\mathbf{S}}, \tilde{\boldsymbol{\mu}}, \tilde{a}, \tilde{b}\}$ converge. For the $(i+1)$-th point, we then use posterior $\tilde{\Theta}$ of $i$-th step as the prior $\Theta \equiv \{\mathbf{S}, \boldsymbol{\mu}, \mathbf{a}, \mathbf{b}\}$.

When learning from data in an online fashion, we need to dynamically adapt the model complexity of the learning to reflect the complexity of the data being modeled. This can be accomplished by adding and deleting local models depending on whether we need to increase the model complexity or decrease it. We employ a heuristic similar to [3] for addition and deletion of models. Here a new local model is added when the predictive probability of a class falls below a certain threshold. A local model is pruned when there is sufficient overlap between the local regions of two local models.

## 6 Evaluation

Before we proceed to detailed evaluation experiments, we need to specify the evaluation measures that would be used to compare different classifiers. We compare classifiers based on two different measures - *misclassification error* and *target information*. The former is the often used loss function that measures the mean number of misclassifications produced by a classifier on a test set. The *target information* criteria refers to a loss function that takes into account the confidence expressed by the classifier about its prediction. The loss function is given by :

$$\mathcal{I} = \frac{1}{N}\left[\sum_{z_i=1} log_2(P(z_i = 1|x_i^q)) + \sum_{z_i=0} log_2(1 - P(z_i = 1|x_i^q))\right] + 1 \qquad (25)$$

and it measures in bits, the information conveyed by the classifier about the test target. For a baseline classifier that assigns classes at random $\mathcal{I} \to 0$ and for a more confident discrimination of classes $\mathcal{I} \to 1$. It must be noted that this measure has a strong penalty for confident misclassification and can lead to $\mathcal{I} < 0$.

### 6.1 Comparison of Generalization Performance and Efficiency of Learning

In the first evaluation, we compare the generalization performance of lRVC against a Gaussian Process classifier with squared exponential covariance function and a baseline probabilistic linear logistic regressor. The lRVC used in the evaluation used around 20 local models initialised at the cluster centers in the input space. The Gaussian process uses a square exponential kernel and a logistic link function. Hyperparameters of the GP are learnt using a gradient descent. The two classifiers are compared on different benchmark datasets listed in Table 1. The Breast cancer, Heart (Cleveland) and the Ionosphere dataset were obtained from the UCI repository, Pima and synthetic datasets are the same as the ones used in [7][1]. The USPS dataset corresponds to the digit discrimination task listed in [6]. The Catalysis and Gatineau datasets were obtained from the predictive uncertainty challenge [2] were the validation set has been used as test set. The evaluations on the datasets obtained from UCI was carried out on 10 train-test splits of the data and the mean and standard deviations are reported here. For all other datasets a single train-test trial was carried out using the train and test files provided. This makes it possible to compare other classifiers that have previously used the latter datasets. For the Gatineau dataset GP was trained using a subset of 1000 training points due to practical considerations of time and space complexity. The evaluation statistics are listed in Table 2. Also shown in the table is the results for a LIBSVM[2] (with an RBF kernel) evaluation over the same datasets with the parameters being chosen using a 5 fold

[1] The datasets can be obtained from http://www.stats.ox.ac.uk/pub/PRNN/
[2] http://predict.kyb.tuebingen.mpg.de/pages/home.php

**Table 1.** Statistics of the benchmark datasets

| Dataset | #train pts. | #test pts | #dim |
|---|---|---|---|
| Breast cancer | 142 | 427 | 30 |
| Heart(Cleveland) | 149 | 148 | 13 |
| Ionosphere | 175 | 176 | 33 |
| Pima | 200 | 332 | 7 |
| Synthetic | 250 | 1000 | 2 |
| USPS(3-5) | 767 | 773 | 256 |
| Catalysis | 873 | 300 | 617 |
| Gatineau | 3000 | 2176 | 1092 |

**Table 2.** Performance comparison between lRVC and GP in terms of the misclassification error rate and the target information (measured in bits) conveyed by the classifier. The values in parenthesis indicate the standard deviation.

| | lRVC | | GP | | Linear | | SVM |
|---|---|---|---|---|---|---|---|
| | Error | Information | Error | Information | Error | Information | Error |
| Breast | 0.028(0.007) | 0.807(0.010) | 0.026(0.009) | 0.805(0.045) | 0.042(0.014) | 0.797(0.050) | 0.028(0.009) |
| Heart | 0.166(0.017) | 0.432(0.049) | 0.169(0.017) | 0.423(0.039) | 0.173(0.024) | 0.388(0.113) | 0.173(0.022) |
| Ionosphere | 0.152(0.027) | 0.338(0.170) | 0.123(0.025) | 0.535(0.054) | 0.163(0.027) | -2.288(0.963) | 0.078(0.025) |
| Pima | 0.202 | 0.361 | 0.222 | 0.276 | 0.198 | 0.364 | 0.198 |
| Synthetic | 0.100 | 0.649 | 0.093 | 0.658 | 0.114 | 0.611 | 0.100 |
| USPS(3-5) | 0.045 | 0.798 | 0.025 | 0.794 | 0.040 | 0.476 | 0.023 |
| Catalysis | 0.303 | 0.143 | 0.303 | 0.150 | 0.343 | -3.516 | 0.323 |
| Gatineau | 0.090 | 0.570 | 0.090 | 0.588 | 0.154 | -0.484 | 0.090 |

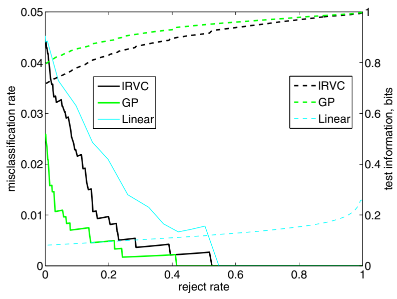
cross validation. The comparison for SVM is restricted to the misclassification error since SVM does not provide a predictive probability. One can see from the results that lRVC is able to match the performance of GP for all the datasets and outperforms the baseline linear classifier especially when the target information is used for the comparison. It must be noted that while lRVC used only a small number of local models for prediction, GP used all of the training set for training and prediction. To emphasize this difference Table 3 shows the time taken by lRVC and GP for training and prediction on a dataset consisting of the USPS digit 3 classified against the rest of the digits. lRVC can be seen to achieve a good generalization performance with a low overhead.

## 6.2 Use of Predictive Confidence Bounds for Rejection

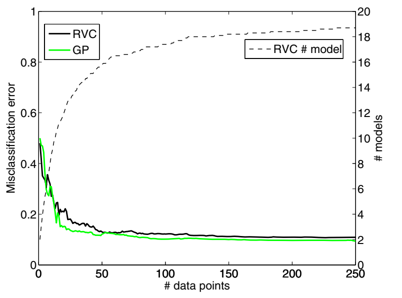
In the next evaluation, we evaluate the confidence bounds learnt by lRVC by plotting the relation between the reject rate, the misclassification error and the target information. In this experiment the first and second moments for the predictive probability were computed by using sampling to evaluate the integral in eq. (20). The test samples which had a variance above a threshold were rejected and the misclassification error was evaluated for the rest of the test data. The dataset used for this purpose was the USPS data. The misclassification error typically decreases as test samples are rejected and an ideal classifier would have a larger reduction in the misclassification error with respect to the rejection rate.

**Table 3.** Comparison between the time taken for training and prediction using a Matlab implementation of lRVC and GP on the USPS dataset

| Method | | 0 | 1 | 2 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| | # training pts. | 1173 | 1028 | 881 | 815 | 767 | 826 | 796 | 783 | 828 |
| | # testing pts. | 1204 | 1065 | 872 | 861 | 773 | 832 | 820 | 749 | 817 |
| GP | Error | 0.011 | 0.002 | 0.018 | 0.005 | 0.026 | 0 | 0.002 | 0.025 | 0.006 |
| | Information | 0.894 | 0.903 | 0.863 | 0.869 | 0.794 | 0.886 | 0.868 | 0.855 | 0.849 |
| | Train time(sec) | 1915.9 | 1475.8 | 1020.8 | 877.0 | 911.4 | 963.7 | 919.9 | 845.8 | 993.0 |
| | Test time(sec) | 47.8 | 37.6 | 25.0 | 22.1 | 20.2 | 23.8 | 23.4 | 20.5 | 24.4 |
| lRVC | Error | 0.009 | 0.005 | 0.022 | 0.005 | 0.045 | 0.003 | 0.009 | 0.032 | 0.012 |
| | Information | 0.944 | 0.960 | 0.886 | 0.966 | 0.798 | 0.972 | 0.955 | 0.883 | 0.946 |
| | Train time(sec) | 582.59 | 509.67 | 440.44 | 402.23 | 382.08 | 400.75 | 370.25 | 363.13 | 383.42 |
| | Test time(sec) | 0.87 | 0.74 | 0.61 | 0.61 | 0.55 | 0.55 | 0.54 | 0.49 | 0.55 |



(a)



(b)

**Fig. 1.** (a) Comparison of error-reject curve for lRVC and GP (b) Online learning dynamics of lRVC compared with GP. The plots are the average performance over 10 trials of different orders of data presentations.

The error and the target information versus the rejection rate for lRVC, GP and the Bayesian linear logistic regressor were evaluated and plotted in fig. (1(a)). It can be seen that lRVC's performance exceeds that of the linear classifier by a large margin and is not significantly different from GP.

## 6.3 Dynamics of Online Learning

In the last evaluation, we use the online updates derived in Section 5 to learn a classifier on the synthetic dataset. The data points are presented to the online learner one at a time and the misclassification error is evaluated over the test data after each training update. The dynamics of the learning process is shown in fig. (1(b)). The learning dynamics is compared with the generalization performance of GP which uses increasing number of training data points and the corresponding test error at each stage is displayed. It can be seen from the figure that online version of lRVC exhibits fast convergence and matches the performance of GP asymptotically.

## 7 Discussion

Local logistic regression is a very competitive method as can be seen from the results of the evaluation. The result makes it more significant when we take it into account that the logistic regression is able to achieve such a good performance using a small number of local models. Moreover the time and space efficiency is linear in terms of the data points and the dimension. In contrast, kernel classification paradigms like GP and SVM have a much higher overhead in training and testing.

The logistic regression formulation in this paper is restricted to binary classification. It can be easily extended to a multi-class classification using a softmax link function instead of a logistic link function. The treatment of the learning remains the same in that case too.

In conclusion, the contribution of this paper has been a probabilistic formulation of a local linear logistic regressor that combines the modeling guarantees of a Bayesian method with the efficiency of an ensemble learner thus making it ideal candidate for online real-time learning.

## References

1. Bishop, C.M.: Neural Networks for Pattern Recognition. Claredon Press (1995)
2. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines (2001), `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
3. Edakunni, N.U., Schaal, S., Vijayakumar, S.: Kernel carpentry for online regression using randomly varying coefficient model. In: International Joint Conference on Artificial Intelligence, pp. 762–767 (2007)
4. Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(3), 226–239 (1998)
5. Nikunj, C.: Oza and Stuart Russell. Online bagging and boosting. In: Artificial Intelligence and Statistics, pp. 105–112 (2001)
6. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)
7. Ripley, B.D.: Pattern Recognition and Neural Networks. Cambridge University Press, Cambridge (1996)
8. Tresp, V.: A Bayesian committee machine. Neural Computation 12(11), 2719–2741 (2000)
9. Vapnik, V.N.: Statistical learning theory. Wiley, New York (1998)