A Bayesian Framework for Active Learning

Richard Fredlund, Richard M. Everson, *Member, IEEE* and
Jonathan E. Fieldsend, *Member, IEEE*

***Abstract*— We describe a Bayesian framework for active learning for non-separable data, which incorporates a *query density* to explicitly model how new data is to be sampled. The model makes no assumption of independence between queried data-points; rather it updates model parameters on the basis of both observations and how those observations were sampled. A 'hypothetical' look-ahead is employed to evaluate expected cost in the next time-step. We show the efficacy of this algorithm on the probabilistic high-low game which is a non-separable generalisation of the separable high-low game introduced by Seung et al.**

Our results indicate that the active Bayes algorithm performs significantly better than passive learning even when the overlap region is wide, covering over 30% of the feature space.

I. Introduction

A central problem of machine learning is the classification of a datum into one of a number of classes. In batch learning a classifier is trained using a training set of examples for which the correct class is also known. The training set is treated as a batch and it is assumed that the order in which these examples is taken is immaterial. In *active* learning, rather than being presented with the data all in one go, the learner is able to select its own training data to learn from. As it can be costly to obtain the class labels, the aim is to learn using as little data as possible. Algorithms and heuristics to select informative data are therefore of interest and a number of approaches have been developed [1], [2], [3], [4], [5], [6], [7]. However, in the main these have assumed that the data are separable. In this paper we develop a framework for active learning with a Bayesian classifier which acknowledges that most real data are not separable. In addition we explicitly model the selection of each successive datum, recognising that the data cannot be treated as independently and identically distributed. This finds a natural expression in a Bayesian setting, permitting the use of Bayesian classifiers which reduce model uncertainty by averaging. We illustrate this framework for active learning with a version of the high-low game, introduced by Seung et al. [1], in which the classes are not separable.

In section II we describe active learning in more detail and give a brief background of previous work and set the context for the Bayesian paradigm. In section III we describe our Bayesian framework for active learning. Section IV illustrates our implementation of this model for two versions of the probabilistic high-low game. Section V shows our results and demonstrates that the active Bayes algorithm performs significantly better than batch learning even when the non-separable overlap region is quite large (over 30% of the feature space). Next we draw conclusions, and finally in the appendix we give an example where a broad query density is better than either sampling at a point or across the whole region.

The authors are with the College of Engineering, Mathematics and Physical Sciences, University of Exeter, EX4 4QF, UK. (email: {rmf206, R.M.Everson, J.E.Fieldsend}@exeter.ac.uk).

II. Background

We focus our attention on binary classification although the framework presented here is straightforwardly extended to other supervised learning problems. In *batch* learning the learner is supplied with a dataset $\mathcal{D}$ comprising N features $x_n \in \mathcal{X}$, $n = 1, \ldots, N$ and the corresponding class C_0 or C_1 to which each feature vector belongs. For simplicity we label the classes 0 and 1 so that the label $y_n \in \{0, 1\}$. The aim is then to assign a new feature x to C_0 or C_1. In a probabilistic setting this is achieved by estimating the predictive probability, $p(y \mid x, \mathcal{D})$, that the class is 1. The predictive probability is found by constructing a model, parametrised by $\theta \in \Theta$, which describes the likelihood of observing the data, $p(\mathcal{D} \mid \theta)$. In the Bayesian paradigm the likelihood is combined with priors on θ to obtain a posterior density:

$$p(\theta \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \theta) p(\theta)}{p(\mathcal{D})}. \tag{1}$$

Class predictions which average over the uncertainty in the parameters can then be found as:

$$p(y \mid x, \mathcal{D}) = \int p(y \mid x, \theta) p(\theta \mid \mathcal{D}) \, d\theta. \tag{2}$$

In active learning the data are observed sequentially and the learner selects — or queries an oracle for — each new datum observed. To clarify the procedure assume that $n-1$ feature vectors and labels have been observed, denoted by $\mathcal{D}_{n-1}$. We divide the observation of a new feature-label pair into two phases, although in practice they are often simultaneous. First, the learner obtains a feature x_n from a particular region of $\mathcal{X}$ determined by the learner and supplied by the *sample oracle*. As we describe below, it is helpful to describe this as a sampling process rather than merely selection. In *pool based* learning the sample oracle supplies x_n from a (finite) pool of data, while in *selective learning* the sample oracle provides a sample from an infinite pool. Secondly, the *label oracle* provides the label y_n of the queried datum x_n.

Under this general paradigm there are three main approaches to selecting the data to be queried.

The first approach is uncertainty based sampling which evaluates the confidence of the classifier on unseen instances and queries the data about which it is most uncertain [2].

Although this idea should favour data close to the estimated decision boundary which are expected to be informative, it also tends to be noise seeking, and may select outliers about which the classifier is uncertain but do not inform overall classification [3].

Secondly, there are version space and query by committee (QBC) approaches which depend upon the notion of a version space [1], [8]. Loosely, version space is the set of classifiers or parametrisations of a classifier family that are consistent with the data thus far observed. New data reduce the volume of version space, because they exclude parametrisations which fail to separate the data. A principle for determining which datum to query next is to choose the x_n which would lead to the greatest expected reduction in version space volume. Since calculating the expected reduction in volume is prohibitively costly because all possible classifiers in version space must be evaluated on each candidate query datum, Seung et al. [1] suggest drawing a sample 'committee' of classifiers from version space and choosing the x_n for which there is maximum disagreement about predicted label amongst the committee; somewhat surprisingly committees of two are quite effective. Seung et al. demonstrate a strong theoretical framework for the separable case, showing that the QBC generalisation error decreases exponentially with the number of data-points queried, as compared to batch learning where the generalisation error decreases with the relatively slow inverse power law.

QBC approaches have been applied to Support Vector Machines (SVMs) with significant success in practical problems [4]. The SVM approach, independently proposed by [5], [6] and [4], who dubbed their version SIMPLE, works well in practice and is considered by Baram et al. [7] to be one of the two best performing active learning algorithms. An SVM works by mapping data into a high-dimensional space, where the two classes are more likely to be separable by a (linear) hyperplane. SIMPLE works by querying the data-point closest to the current decision hyperplane.

The QBC approach is limited by the assumption that the data are separable, which allows portions of version space to be completely eliminated. This restriction is overcome to some extent with SVMs by mapping data into a high dimensional space where it is more likely to be linearly separable. However, QBC and uncertainty based sampling both suffer from the limitation that they optimise on parameter space without directly considering classification cost.

Finally, the third approach called SELF-CONF is to optimise the choice of query in order to maximise its expected utility at the next time step. This was considered by [7] to be one of the two top performing active learning algorithms, along with SIMPLE described above. Roy and McCallum [3] describe a scheme which seeks to directly minimise classification cost in the next time-step by estimating these costs on the basis of its current parameter estimate. They hint that this is 'optimal' for active learning, but the main focus is on finding practical sampling techniques which can be applied to applications such as text classification. However, they use naïve Bayes classifiers and make the assumption that the data selected are independently and identically distributed. This violated independence assumption tends to produce overly sharp posterior distributions, which they overcome by bagging and averaging over the posteriors [3].

Application of these approaches to the classification of non-separable data is not immediate. Uncertainty based sampling and QBC approaches work directly on parameter space without reference to the specific cost function being used, which can lead to learning which eliminates areas of parameter space which do not affect misclassification cost. Although non-separable data means that regions of version space cannot be completely eliminated by new observations, an attractive extension of the idea is to identify version space with the space of model parameters, elements of which are reweighted according to their posterior probability (1). A difficulty which must be tackled however is that the sampled data cannot be considered to be independently and identically distributed (i.i.d.), because successive data are actively selected on the basis of previous observations. This means that, unlike batch learning, the likelihood cannot be factorised as follows:

$$p(x_1, y_1, ..., x_n, y_n \,|\, \theta) = \prod_{i=1}^{n} p(x_i, y_i \,|\, \theta). \tag{3}$$

In the following section we therefore describe a framework for active learning which incorporates a query density to explicitly model the selection procedure and, like SELF-CONF, employs a 'look-ahead' to minimise the expected misclassification cost in the next step.

III. A FRAMEWORK FOR BAYESIAN ACTIVE LEARNING

We assume that prior beliefs about the classifier's parameters are initially encapsulated in the prior distribution $p(\theta)$ and, after n data $\mathcal{D}_n$ have been observed, knowledge about the parameters is summarised in the posterior $p(\theta \,|\, \mathcal{D}_n)$, which in turn serves as the prior for the $(n+1)$-th step.

Rather than selecting a particular datum to query, we model the query as a draw of a sample from the distribution:

$$Q(x; \phi) = \frac{q(x; \phi) p(x)}{\int q(x; \phi) p(x)\, dx} \tag{4}$$

where $p(x)$ is the (unknown) density of features and $q(x; \phi)$ is a *query density* ($q(x; \phi) \geq 0$ for all $x \in \mathcal{X}$ and $\int q(x; \phi)\, dx = 1$) which determines which region of $\mathcal{X}$ is drawn from. The query density is chosen by the learner and depends upon parameters ϕ. To make the nth query the *sample oracle* accepts a parametrisation ϕ_n and returns an x_n sampled according to (4). Choosing q to be constant over $\mathcal{X}$ is equivalent to online learning with examples presented at random, while the algorithm's 'attention' can be narrowly focused on a particular region by choosing q to be narrow, the limiting case being a delta function so that a particular x is selected without reference to the underlying distribution $p(x)$.

Information about θ is gained from the sample x_n, but the primary source of information about class membership is the

Algorithm 1 Bayesian active learning

Require: $p(\theta)$ *Prior on parameters*

```
for n = 1, ..., N
    φ_n = argmin_φ ESTIMATED-RISK(φ)
    Φ_n = {φ_n, φ_{n-1}, ..., φ_1}
    x_n ~ Q(x; φ_n)            Sample oracle: equation (4)
    y_n ~ p(y | x_n)           Label oracle
    D_n = D_{n-1} ∪ {x_n, y_n}
    p(θ | D_n, Φ_n) ∝ p(y_n, x_n | θ, φ_n) p(θ | D_{n-1}, Φ_{n-1})
end for
```

class y_n of x_n, which is returned by the *label oracle* as a sample from $p(y \mid x_n)$. We emphasise that when the problem is not separable repeated interrogation of the label oracle for a single x may yield samples from C_0 and C_1.

Algorithm 1 summarises the Bayesian active learning procedure for a single time-step. Prior beliefs about the model parameters are encapsulated in the prior $p(\theta)$. In pool learning $p(\theta)$ might include information from an unlabeled pool of features. Then at each stage of learning parameters ϕ_n of the query function are determined as those that minimise the expected misclassification cost or risk at the next step; we describe how this is achieved in section III-B. After x_n and y_n have been sampled, the model parameters can be updated using Bayes' rule, which we discuss next.

A. Parameter update

Having obtained a new feature vector x_n from the feature oracle the probability density for the parameters can be updated using Bayes' rule. The new feature is sampled according to (4), but since $p(x)$ itself is unknown, the distribution $Q(x;\phi)$ is modeled by

$$Q(x \mid \theta, \phi) = \frac{q(x;\phi)p(x \mid \theta)}{\int q(x;\phi)p(x \mid \theta)\, dx} \tag{5}$$

where $p(x \mid \theta)$ is the learner's model for $p(x)$ and $Q(x \mid \theta, \phi)$ is thus the learner's model of the sample oracle. This models the distribution from which the observation x_n is drawn, and is used in the Bayesian update of the parameters:

$$p(\theta \mid x_n, \mathcal{D}_{n-1}, \Phi_n) \propto Q(x_n \mid \theta, \phi_n)p(\theta \mid \mathcal{D}_{n-1}, \Phi_{n-1}) \tag{6}$$

where the constant of proportionality is found by integrating the r.h.s. over θ. Note that this posterior density depends upon the queries made through the sequence of query density parameters $\Phi_n = \{\phi_n, \phi_{n-1}, \ldots, \phi_1\}$; we regard these as parameters whose values can be optimised, rather than as random variables like θ.

Additional information is supplied by the label oracle which provides a sample from $p(y \mid x_n)$. This in turn is used to update the posterior as:

$$p(\theta \mid \mathcal{D}_n, \Phi_n) \propto p(y_n \mid x_n, \theta)p(\theta \mid x_n, \mathcal{D}_{n-1}, \Phi_n) \tag{7}$$

where again the constant of proportionality may be found by integration. Clearly (5), (6) and (7) may be combined as:

$$p(\theta \mid \mathcal{D}_n, \Phi_n) \propto p(y_n \mid x_n, \theta)Q(x_n \mid \theta, \phi_n)p(\theta \mid \mathcal{D}_{n-1}, \Phi_{n-1}) \tag{8}$$

in which $p(\theta \mid \mathcal{D}_{n-1}, \Phi_{n-1})$ serves as a prior for the nth observation, and the likelihood of observing $\{x_n, y_n\}$ is assessed by $p(y_n \mid x_n, \theta)Q(x_n \mid \phi_n, \theta)$ which emphasises the dependence of the likelihood on the query sequence.

Note that the whole of parameter space Θ may be regarded as a version space equipped with $p(\theta \mid \mathcal{D}_n, \Phi_n)$ that measures the probability that θ are the correct parameters for the data under this model. Unlike QBC [1] in which, for separable data, the volume of version space is reduced as data are observed, here $\int p(\theta \mid \mathcal{D}_n, \Phi_n)\, d\theta = 1$ but probability mass is redistributed to concentrate on more probable parameter values. We give numerical illustrations of this for the probabilistic high-low game in section IV. If the learner's model includes the data generation process (a 'closed' situation in the terminology of [9]), then in the large n limit we may expect probability mass to be concentrated around the 'correct' parameters, provided that the query sequence Φ_n is not malicious.

After n learning steps, the probability of a feature x belonging to class C_1 is estimated by averaging the predictive likelihood $p(y \mid x, \theta)$ weighted by the learned posterior:

$$p(y \mid x, \mathcal{D}_n, \Phi_n) = \int p(y \mid x, \theta)p(\theta \mid \mathcal{D}_n, \Phi_n)\, d\theta. \tag{9}$$

Note that we assume here that x is drawn from the data distribution $p(x)$, not queried using $q(\cdot)$.

B. Estimated risk

Suppose that $n-1$ features and labels have been queried. To query the nth feature and class the learner must decide on parameters ϕ_n of the query density. In order to do this we 'look-ahead' one step and estimate the expected misclassification cost or risk if $\{x_n, y_n\}$ were selected with ϕ_n and choose ϕ_n to minimise this estimated risk. To emphasise that the feature and its class have not yet been selected we denote hypothetical samples by x'_n and y'_n, and the data including them by $\mathcal{D}'_n = \mathcal{D}_{n-1} \cup \{x'_n, y'_n\}$.

In order to evaluate the risk we assume that losses incurred are known. For simplicity we further assume that the cost of a correct classification is zero, while the cost of incorrectly classifying to the $y = 1$ class or the $y = 0$ class are λ_{01} and λ_{10} respectively. If having observed data $\mathcal{D}'_n$, the learner predicts the probability of x belonging to C_1 according to (9) as $p(y \mid x, \mathcal{D}'_n, \Phi_n)$, then the assignment that minimises the expected misclassification cost [10] is:

$$\mathcal{A}(x \mid \mathcal{D}'_n) = \begin{cases} 1 & \text{if } p(y \mid x, \mathcal{D}'_n, \Phi_n) \geq \lambda_{01}/(\lambda_{01} + \lambda_{10}) \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

The estimated expected misclassification cost or risk for a feature x is therefore:

$$R(x \mid \mathcal{D}'_n, \Phi_n) = \delta_{0,\mathcal{A}(x)} p(y=1 \mid x, \mathcal{D}'_n, \Phi_n)\lambda_{10} + \delta_{1,\mathcal{A}(x)} p(y=0 \mid x, \mathcal{D}'_n, \Phi_n)\lambda_{01} \quad (11)$$

where δ is the Kronecker delta and we have suppressed the explicit dependence of $\mathcal{A}(x)$ on $\mathcal{D}'_n$. Since $R(x \mid \mathcal{D}'_n, \Phi_n)$ depends on the particular x'_n and y'_n returned by the oracles, the estimated cost should be averaged over the query density:

$$R(x \mid \Phi_n) = \iint R(x \mid \mathcal{D}'_n, \Phi_n) p(y'_n \mid x'_n) Q(x'_n \mid \phi_n)\, dx'_n dy'_n. \quad (12)$$

The average misclassification cost anticipated when making a query with query parameters ϕ_n is therefore:

$$R(\Phi_n) = \int R(x \mid \Phi_n) p(x)\, dx \quad (13)$$

$$= \iiint R(x \mid \mathcal{D}'_n, \Phi_n) p(y'_n \mid x'_n) Q(x'_n \mid \phi_n) p(x)\, dx'_n dy'_n dx \quad (14)$$

We emphasise that $R(\Phi_n)$ depends upon only ϕ_n at the nth step because Φ_{n-1} is fixed.

Since $p(x)$ and $p(y \mid x)$ are unknown we estimate $R(\Phi_n)$ using models based on $\mathcal{D}_{n-1}$:

$$\tilde{R}(\Phi_n) = \iiint R(x \mid \mathcal{D}'_n, \Phi_n) p(y'_n, x'_n \mid \mathcal{D}_{n-1}, \Phi_n) p(x \mid \mathcal{D}_{n-1})\, dx'_n dy'_n dx \quad (15)$$

where

$$p(y'_n, x'_n \mid \mathcal{D}_{n-1}, \Phi_n) = \int p(y'_n, x'_n \mid \theta, \phi_n) p(\theta \mid \mathcal{D}_{n-1}, \Phi_{n-1})\, d\theta. \quad (16)$$

The joint density breaks into

$$p(y'_n, x'_n \mid \theta, \phi_n) = p(y'_n \mid x'_n, \theta) Q(x'_n \mid \theta, \phi_n) \quad (17)$$

and

$$p(x \mid \mathcal{D}_{n-1}, \Phi_{n-1}) = \int p(x \mid \theta) p(\theta \mid \mathcal{D}_{n-1}, \Phi_{n-1})\, d\theta. \quad (18)$$

In pool based learning, where the unlabeled features are available at the outset, an estimate for $p(x \mid \mathcal{D}_{n-1})$ to be used in (15) may be made before query based learning begins.

The parameters of the query function ϕ_n are then chosen to minimise the estimated expected risk:

$$\phi_n = \operatorname*{argmin}_{\phi_n} \tilde{R}(\Phi_n) \quad (19)$$

as indicated on line 2 of Algorithm 1.

Implementation of this active learning scheme requires successive calculations of the parameter posterior and the expected risk. Although the expression (8) for the posterior is a straightforward application of Bayes' rule, the presence of the query function renders finding conjugate families unlikely in all but very special circumstances and Monte Carlo sampling or approximate methods are likely to be required in general.

We now demonstrate the framework on the toy problem of the probabilistic high-low game where parameter space is two dimensional.

IV. Probabilistic High-Low Game

Seung et al. [1] demonstrate their version-space model on the toy problem of the high-low game. This is a one dimensional problem containing two uniform classes which meet at some unknown threshold, γ. Here $x \in [0, 1)$ is in class 1 if $x \geq \gamma$ and class 0 otherwise. This version of the high-low game is completely separable and the target given by the *label* oracle is always correct, allowing hypotheses to be kept or ruled out completely on the basis of each new query point.

We consider a straightforward non-separable extension of this model, which we call the probabilistic high-low game. This is similar to the original high-low game with the extension of allowing some overlap between the classes.

Class C_0 is uniform on $[0, \alpha)$ and class C_1 is uniform on $[\beta, 1)$. For simplicity we assume that the relative frequency of the two classes, w_0 and w_1 are known and $P(C_0) = w_0$ and $P(C_1) = w_1$. This model has only two unknown parameters, α and β, and for particular α, β we have:

$$p(x \mid \alpha, \beta) = \frac{w_0}{\alpha}\mathcal{I}_{[0,\alpha)}(x) + \frac{w_1}{1-\beta}\mathcal{I}_{[\beta,1)}(x). \quad (20)$$

This defines $p(x \mid \theta)$ in (5) with the parameters $\theta = \{\alpha, \beta\}$. In order to ensure that the two classes at least meet we add the condition that $\beta < \alpha$. When $\alpha = \beta$ there is no overlap between the classes and this simplifies to the separable high-low game, with $\alpha = \beta = \gamma$. Given this model $p(y = 1 \mid x, \alpha, \beta)$ is also known. An observation may lie in one of three regions: When $x < \beta$ it is definitely in class 0, when $x \geq \alpha$ it definitely lies in class 1. The third region is the overlap region where:

$$p(y=1) = \frac{\alpha w_1}{\alpha w_1 + (1-\beta)w_0} \quad \text{for } \beta < \alpha \quad (21)$$

which depends on the relative frequency of the two classes, w_0 and w_1.

We define the query density $q(x; \phi)$ to be uniform between q_- and q_+, and zero elsewhere. $Q(x \mid \theta, \phi)$ will therefore sample data from $p(x \mid \theta)$, but only in the interval $[q_-, q_+)$.

The algorithm is initialised by a prior $p(\theta) = p(\alpha, \beta)$. We take this to be uniform on the region $\beta \leq \alpha$:

$$p(\alpha, \beta) = \begin{cases} 2 & \text{if} \quad \beta \leq \alpha \\ 0 & \text{otherwise} \end{cases} \quad (22)$$

Having formed the basis of the model by defining $p(x \mid \alpha, \beta)$ and $p(y = 1 \mid x, \alpha, \beta)$ and having defined a prior $p(\alpha, \beta)$, we now reach the major part of the algorithm in which the learner must decide on a suitable ϕ, that is the query interval. Having made $n - 1$ observations, we look ahead to estimate the expected cost of making the nth observation and choose the interval that has the minimum

expected risk (line 2 of Algorithm 1). This is achieved by sampling a hypothetical observation (x', y') from $Q(x'; \phi_n)$, and finding the average expected cost (15) associated with each $q()$ parametrised by $\phi_n \equiv (q_-, q_+)$.

The first step is to sample a hypothetical, α', β' from the 'prior' $p(\alpha, \beta \,|\, \mathcal{D}_{n-1}, \Phi_{n-1})$. Given these we define a *model* of the sample oracle, $Q(x' \,|\, \alpha', \beta', \Phi)$, which allows us to draw a hypothetical x' from the approximation of $p(x)$ by $p(x \,|\, \alpha', \beta')$. Similarly, we find a corresponding hypothetical target y' from $p(y' \,|\, x', \alpha', \beta')$, our *model* of the label oracle.

Having drawn x' and y' we now perform a Bayesian update, for which we will find the corresponding expected cost, and as in (8) we have:

$$p(\alpha, \beta \,|\, x', y', \phi) \propto p(y' \,|\, x', \alpha, \beta) Q(x' \,|\, \alpha', \beta', \phi_n) p(\alpha, \beta) \tag{23}$$

For this hypothetical posterior we now aim to find the corresponding 'optimal classifier'. Following the general case we aim to find the decision boundary and corresponding decision function which minimises the expected cost for a given λ_{01} and λ_{10}. We approximate the model-averaged Bayesian classifier (9) to find $p(y = 1 \,|\, x = \eta) = \iint p(y = 1 \,|\, x = \eta, \alpha, \beta) p(\alpha, \beta \,|\, y', x') d\alpha d\beta$ by a Monte Carlo sample; in the work reported here we use 40 samples of each of (α, β), x' and y'. To minimise the expected misclassification cost, as described in equations (10) and (11), we find the boundary between the decision regions η such that:

$$\begin{aligned} &p(y = 1 \,|\, x = \eta) \\ &\quad \approx \iint p(y = 1 \,|\, x = \eta, \alpha, \beta) p(\alpha, \beta \,|\, y', x') d\alpha d\beta = \lambda, \end{aligned} \tag{24}$$

where $\lambda = \lambda_{01} / (\lambda_{01} + \lambda_{10})$. The corresponding action function $\mathcal{A}(x)$ is given by:

$$\mathcal{A}(x) = \begin{cases} 1 & \text{if } x \geq \eta \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

which classifies $y = 1$ for all $x \geq \eta$ and 0 otherwise. This is the optimum classification for $p(\alpha, \beta \,|\, x', y', \phi_n)$. Finally the average cost $\tilde{R}(\Phi_n)$ is found by averaging $\tilde{R}(x \,|\, \mathcal{D}'_n, \Phi_n)$ with respect to $p(x \,|\, \alpha, \beta, x', y', \phi_n)$.

V. Results for the High-Low Game

First we give an example illustrating a single time-step. In this example, starting with a uniform prior (22) one data point $x_1 = 0.5$ has been queried, and was labeled by the oracle as being in class 1. Figure 1 (top) shows $p(\alpha, \beta \,|\, \mathcal{D}_1, \Phi_1)$, the learner's current beliefs about α, β, having made this observation. As β is the lower boundary for class 1, then $\beta \leq x_1 = 0.5$ and as the figure shows the posterior is zero for all $\beta > x_1$.

Given these current beliefs the algorithm now performs a look-ahead to evaluate the expected cost. This is done by sampling hypothetical 'true' values α', β', from figure 1 (top) and hypothetical observations x', y' in order to evaluate the expected risk $\tilde{R}(q_-, q_+)$ associated with each choice

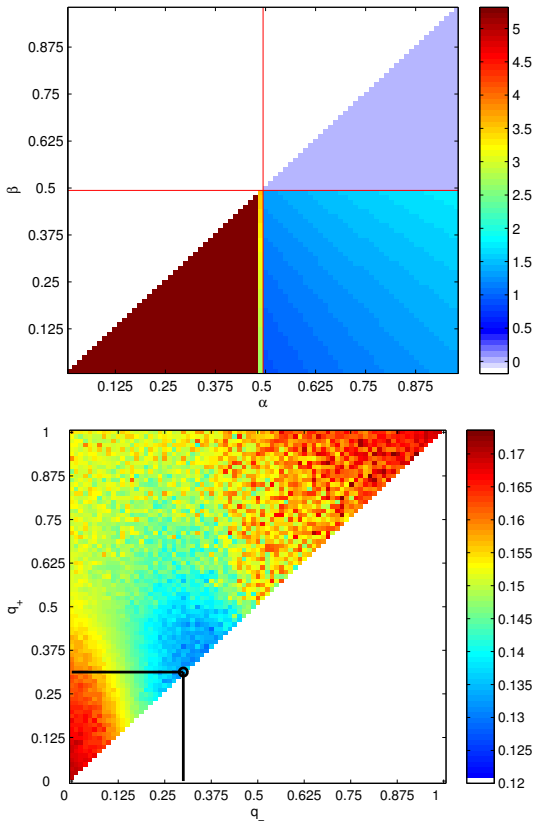

Fig. 1. Top: Posterior density $p(\alpha, \beta \,|\, x_1 = 0.5, y_1 = 1)$ after a single observation $x_1 = 0.5$, indicated by the red cross-hair. Bottom: Expected cost $\tilde{R}(\Phi_1)$ as a function of q_- and q_+, using $p(\alpha, \beta \,|\, x_1 = 0.5, y_1 = 1)$ as the prior. The black circle indicates the minimum.

of sample region $[q_-, q_+)$. These are shown in figure 1 (bottom). As is clear from the figure, the $[q_-, q_+)$ for which the expected cost is minimised lies on the main diagonal, where $q_- = q_+$ at 0.3.

For the probabilistic high-low game, in every example we encountered, the minimum expected cost was found to occur when $q_- = q_+$, implying that the learner should query at a single point rather than probabilistically from a region. In the following we therefore assume that $q_- = q_+$ which allows a very significant narrowing of the space to be searched to find the optimum query. This does however raise the question of whether there are examples of binary classification problems where it is advantageous to select from a region, rather than simply query data at a single point. To answer this question affirmatively we describe in the Appendix a simple example where querying over a region gives some advantage over selection at a point.

To evaluate the performance of the algorithm we measure the 'actual cost' namely the average misclassification cost that would be incurred by a Bayesian classifier encountering examples from $p(x)$. That is, the cost is

$$R_{\text{actual}} = \int R(x \,|\, \mathcal{D}_n, \Phi_n) p(x) \, dx \tag{26}$$

using (11) together with (10) and (9).

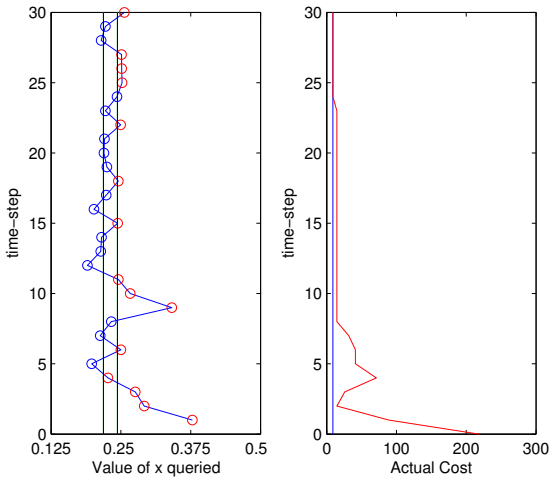

Fig. 2. Left: the x_n queried at each time-step in a single run of the narrow overlap problem. The overlap interval is from $\beta = 0.1875$ to $\alpha = 0.2375$ and is indicated by the vertical lines. The symbol colours indicate the class label returned by the oracle: red for class 1, blue for class 0. Right: the actual cost R_{actual} at each time-step (26). The blue line shows the minimum cost for this problem.

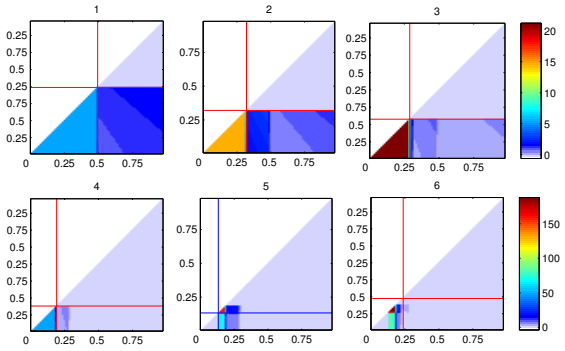

Fig. 3. The first 6 posterior densities associated with the run shown in figure 2. The location of the previous query is shown by the blue and red lines where the class label returned is class 0 and class 1 respectively.

We ran the algorithm on two versions of the probabilistic high-low game, with the two classes spanning the finite region $[0, 1]$. In one version the overlap region was narrow, $\alpha = 0.2375$ and $\beta = 0.1875$, while in the other the overlap region is much wider, $\alpha = 0.6875$ and $\beta = 0.375$. In both cases we choose $w_0 = w_1 = 0.5$ and $\lambda_{01} = \lambda_{10} = 1$. In the overlap region the probability of choosing C_1, given by equation (21), is 0.23 and 0.52 in the narrow and wide overlap cases respectively.

Figure 2 (left) shows the x_n selected in a single run, for the narrow overlap case. Blue and red circles indicate data from class 0 and class 1 respectively, and the vertical lines indicate the true class boundaries $\alpha = 0.2375$ and $\beta = 0.1875$. Figure 2 (right) shows the actual cost associated with this particular run. After 8 data points have been observed, the actual cost has almost reached the minimum for these parameters. Note that after ≈ 18 queries the algorithm tends to query points very close the edges of the overlap region.

Figure 3 shows the sequence of posterior distributions for α, β for the first 6 time-steps. Lines indicate the location of the queries, blue and red indicating class 0 and 1 labels respectively. The algorithm has rapidly narrowed down the probable values of α and β. However, unlike the original separable high-low game in which regions of version space could be completely excluded, in this non-separable problem parameters in the overlap region can only be reweighted as new observations concentrate probability mass.

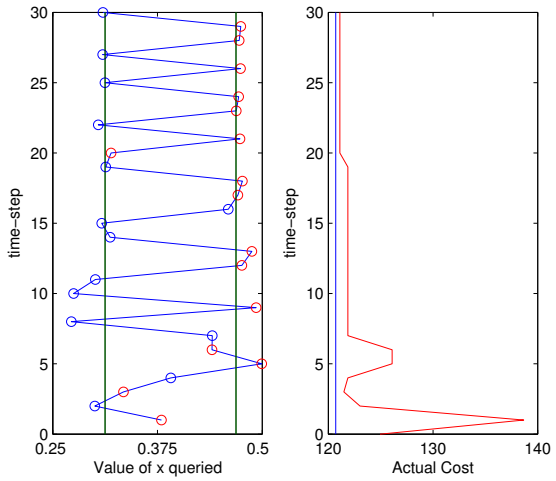

Fig. 4. As figure 2, but for the wide overlap $\alpha = 0.6875$ and $\beta = 0.375$.

Figure 4 shows a single run with a wider overlap region ($\alpha = 0.6875$, $\beta = 0.375$). This is a more difficult problem, and it takes 20 data points for the algorithm to closely approach the minimum actual cost. Again the algorithm rapidly hones in on queries near to the class boundaries.

Figure 5 shows the median cost versus number of queries averaged over 100 runs, for the narrow (top) and wide (bottom) problems. Also shown is the median cost for random selection of queries from $p(x)$ and it is clear that the active algorithm is learning faster than the random sampler.

Note that in both graphs there is an early spike in the actual cost for the active learner. In the wide overlap case, the actual cost after a single observation is greater than the cost using only the uninformative prior. This is because the initial query is always within the overlap region at $x = 0.5$ and the label information, which is inherently misleading here, is over-fitted by the model. Reference to figures 2, 4 and 6 shows that the spike after a few queries is associated with the learner overshooting the overlap region boundaries. However, once the learner has gained information to model where the overlap region lies this is more than compensated for.

Figure 6 shows a scatter plot of the choice of x in each time step over multiple runs in a similar manner to figures 2 and 4. It is interesting to note that over time the algorithm appears to favour points just outside the overlap region, and avoids queries near to the middle of the overlap region. This is particularly noticeable in the wide-overlap problem where after about 15 queries, the algorithm appears to be 'testing' the boundary. Notice also that in the narrow-overlap problem

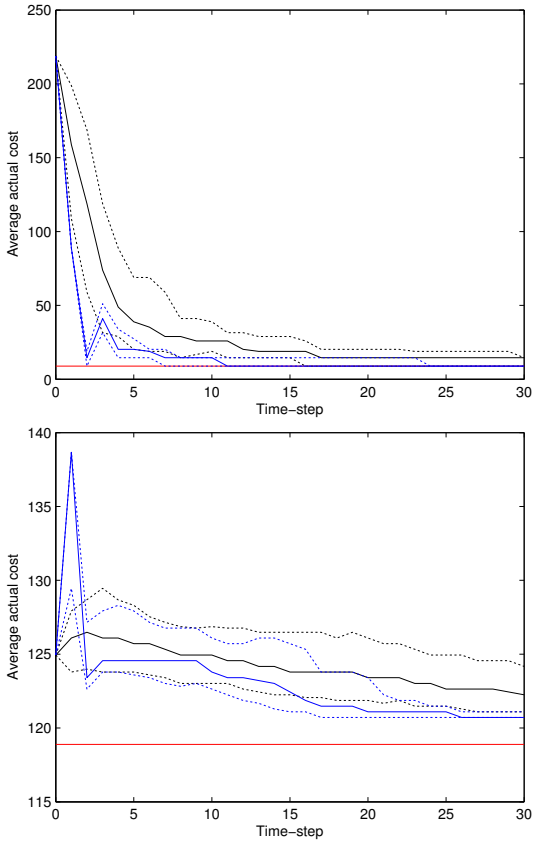

Fig. 5. Median cost after n observations, for random sampling (black), averaged over 200 runs and for active sampling (blue) averaged over 100 runs. The red line is the minimum cost for the problem. Dotted error lines show the 25^{th} and 75^{th} percentiles. Top and bottom panels show results for narrow and wide overlap problems respectively.

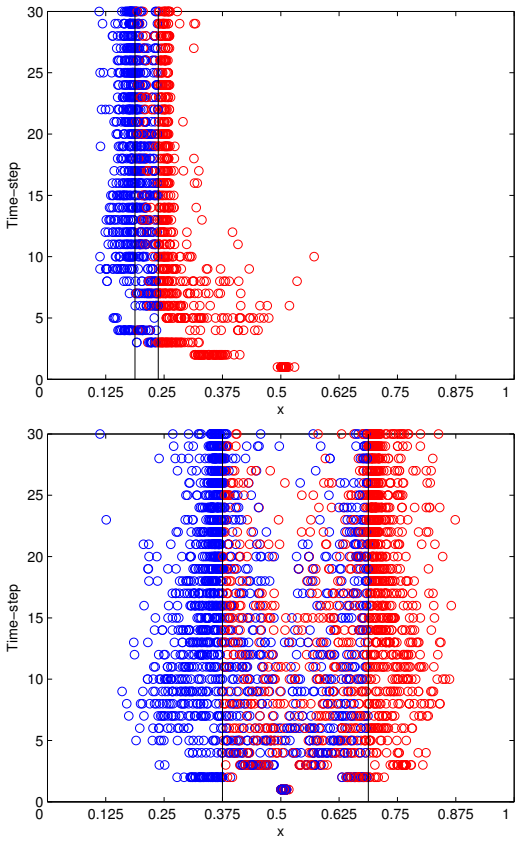

Fig. 6. Query points over multiple runs, for the narrow overlap problem (top) and the wide overlap (bottom). As with figures 2 and 4, the class label returned by the oracle is represented by red circles for class 1 and blue circles for class 0.

the algorithm makes most of its queries between $x = 0.1$ and $x = 0.3$. In other words it fairly rapidly homes in on the vicinity of the overlap region.

Since in the probabilistic high-low game we have only considered point queries ($q_- = q_+$) it is straightforward to plot the estimated cost $\tilde{R}(q_-, q_+)$ for each query as shown for the first 10 queries in Figure 7 for the wide-overlap problem. As can be seen in the first time-step the learner correctly predicts that it should query $x = 0.5$. At subsequent time-steps the graph of $\tilde{R}(q_-, q_+)$ is not very smooth owing to the relatively few Monte Carlo samples used to approximate the integrals. Nonetheless the location of the minimum is quite well defined and after a few time-steps it is clearly more costly to sample in the centre of the overlap region than close to the edges.

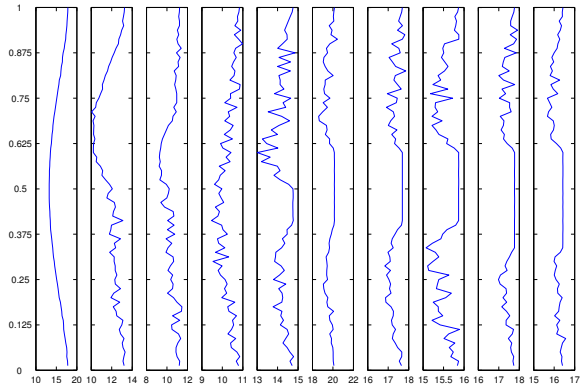

Fig. 7. Predicted cost for 10 time-steps of a single run, with time increasing from left to right. Vertical axes describe the location of the query $q_- = q_+ \in [0, 1]$ and estimated cost $\tilde{R}(q_-, q_+)$ is plotted horizontally. Black horizontal lines indicate the overlap region.

VI. Conclusion

We have presented a framework for probabilistic active learning of non-separable data which is based upon the principle of reducing the estimated misclassification cost of each datum queried. The Bayesian formulation permits averaging over model parameters to reduce parameter uncertainty in predictions. This approach explicitly acknowledges the non-i.i.d. nature of the data in active learning and has the facility to select from arbitrary query density which models the query process. To our knowledge this is the first time active selection from a sampling distribution, rather than specifically at a point has been considered.

The three Gaussians problem, described in the Appendix, shows that, at least in some situations, the optimal way of selecting is not necessarily to select at a single point

or randomly across the whole region, but to select from a distribution. We note that the cost benefit of selecting from an interval is not very large in the three Gaussians problem, which was constructed to test this, and in the high-low game it appears that selecting at a point is optimal anyway. However this may not be the case in all problems and there may be problems where sampling from a specific region offers a considerable advantage. It will be important to characterise those problems in which sampling broadly is better than sampling from a point.

The integrations required to implement this framework do however come with considerable computational costs and is not yet feasible in real world settings. Current work is exploring approximations which will allow this approach to be generally applicable.

APPENDIX

AN EXAMPLE WHERE SAMPLING BROADLY IS OPTIMAL

Because in the probabilistic high-low game simulations, the minimum estimated risk always appeared at $q_- = q_+$ (see Figure 1), which corresponds to selecting x at a point, the natural question arises: is sampling at a single point rather than from a broad query function always optimal? Here we present an example in which the optimal query function is not infinitely narrow but has some finite width. The problem, which we call the 'three Gaussians' problem, is illustrated in Figure 8. It is a binary classification problem, with equal prior probabilities, $p(C_0) = p(C_1) = 1/2$, and $p(x \mid C_1) = (\mathcal{N}(x \mid 3, 1) + \mathcal{N}(x \mid 7, 1))/2$ (red lines in Figure 8). The other class conditional density (green) is also Gaussian: $p(x \mid C_0) = \mathcal{N}(x \mid \mu, 1)$, where μ is the single unknown parameter to be learnt in the problem.

We choose the prior $p(\mu) = \mathcal{N}(\mu \mid 5, 2)$ as indicated by the blue curve. The query density is also chosen to be Gaussian, centred on $x = 5$ and with width σ_q: $q(x) = \mathcal{N}(x \mid 5, \sigma_q^2)$. If $p(\mu \mid x, y)$ is the posterior density of μ having observed a single (x, y) pair, then Figure 9 (top) shows the expected posterior density $p(\mu \mid \Phi = \sigma_q) = \int p(\mu \mid x', y') p(y' \mid x', \mu) Q(x' \mid \sigma_q)\, dx' dy'$ which is the posterior having queried an (x', y') pair averaged over $q(x' \mid \sigma_q)$. We evaluate the look-ahead risk as the entropy of this posterior, so that minimum entropy corresponds to the most compact distribution in parameter space, the probabilistic variant of minimising version space volume. The bottom panel of Figure 9 shows the entropy of these posterior distributions as a function of σ_q. As we can see in this case the minimum cost is given by sampling with $\sigma_q \approx 1.2$.

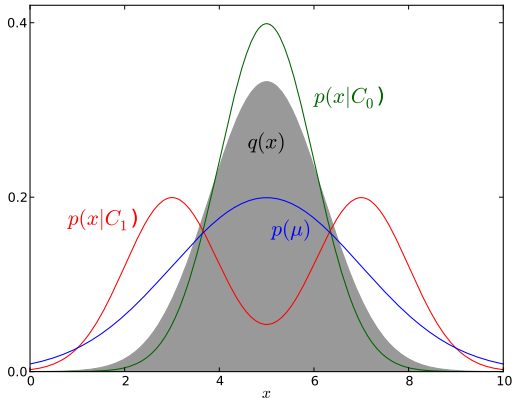

Fig. 8. Illustration of the three Gaussian problem. Class conditional densities $p(x \mid C_i)$ are shown in green and red. All the parameters of $p(x \mid C_1)$ are known, but the mean μ of C_0 class is to be learned; the prior $p(\mu)$ is shown in blue. Data is queried from a query density $q(x) = \mathcal{N}(x \mid 5, \sigma_q)$ whose width σ_q may be varied; the shaded distribution shows a query density of approximately the optimal width.

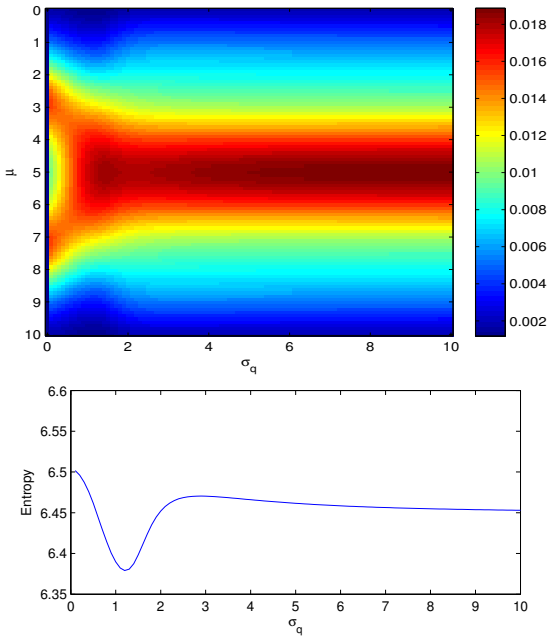

Fig. 9. Top: expected posterior density $p(\mu \mid x, y, \Phi = \sigma_q)$ in the next time-step as a function of σ_q. This is found by averaging across all hypothetical observations (x', y') with x' drawn from $q(x)$. Bottom: entropy of the expected posterior density for each σ_q. This has a minimum when $\sigma_q = 1.2$, which suggests that in the three Gaussian problem $q(x) = \mathcal{N}(x \mid 5, 1.2)$ is the optimum query density for the first observation.

REFERENCES

[1] H. Seung, M. Opper, and H. Sompolinsky, "Query by committee," *Computational Learning Theory*, 1993.

[2] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *17th annual international ACM SIGIR conference*, vol. 29, 1994, pp. 3–12.

[3] N. Roy and A. McCallum, "Toward optimal active learning through sampling estimation of error reduction," in *Proc. 18th International Conf. on Machine Learning*. Morgan Kaufmann, 2001, pp. 441–448.

[4] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Proc. 17th ICML*, 2000.

[5] G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines," in *Proc. 17th International Conf. on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 2000, pp. 839–846.

[6] C. Campbell, N. Cristianini, and A. Smola, "Query learning with large margin classifiers," in *Proc. 17th ICML*, 2000, pp. 111–118.

[7] Y. Baram, R. El-Yaniv, and K. Luz, "Online choice of active learning algorithms," *Journal of Machine Learning Research*, vol. 5, pp. 255–291, 2004.

[8] M.-F. Balcan, A. Beygelzimer, and J. Langford, "Agnostic active learning," *Proc. 23rd ICML*, 2006.

[9] J. Bernardo and A. Smith, *Bayesian Theory*. Wiley, 1994.

[10] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley, 2000.