Unbiased Online Active Learning in Data Streams

Wei Chu*
Microsoft
Redmond, WA, USA
chu.wei@microsoft.com

Martin Zinkevich
Yahoo! Labs
Sunnyvale, CA, USA
maz@yahoo-inc.com

Lihong Li
Yahoo! Labs
Sunnyvale, CA, USA
lihong@yahoo-inc.com

Achint Thomas
Yahoo! Labs
Sunnyvale, CA, USA
aothomas@yahoo-inc.com

Belle Tseng
Yahoo! Labs
Sunnyvale, CA, USA
belle@yahoo-inc.com

ABSTRACT

Unlabeled samples can be intelligently selected for labeling to minimize classification error. In many real-world applications, a large number of unlabeled samples arrive in a streaming manner, making it impossible to maintain all the data in a candidate pool. In this work, we focus on binary classification problems and study selective labeling in data streams where a decision is required on each sample sequentially. We consider the unbiasedness property in the sampling process, and design optimal instrumental distributions to minimize the variance in the stochastic process. Meanwhile, Bayesian linear classifiers with weighted maximum likelihood are optimized online to estimate parameters. In empirical evaluation, we collect a data stream of user-generated comments on a commercial news portal in 30 consecutive days, and carry out offline evaluation to compare various sampling strategies, including unbiased active learning, biased variants, and random sampling. Experimental results verify the usefulness of online active learning, especially in the non-stationary situation with concept drift.

Categories and Subject Descriptors

G.3 [**Probabilities and Statistics**]: Probabilistic Algorithms (including Monte Carlo); I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*

General Terms

Algorithms, Experimentation, Performance

*The work was done when WC was with Yahoo! Labs.

KDD'11, August 21–24, 2011, San Diego, California, USA.

Keywords

Active Learning, Adaptive Importance Sampling, Unbiasedness, Bayesian Online Learning, Data Streaming

1. INTRODUCTION

Active learning holds the promise of reducing the amount of labeled data in supervised learning required to reach a certain level of performance such as classification accuracy. Many of the successful applications are for the pool-based scenario [20], where a *static* collection of unlabeled data are given, from which a subset are chosen for labeling. In other situations where unlabeled examples arrive sequentially, *stream-based* active learning [5, 7] is more appropriate. Since the size of the data stream is often large (and even unbounded) in real-world applications, it is impractical to store all unlabeled data and then run active learning. Rather, an online algorithm has to decide, for the current unlabeled datum, whether to query its label or not. It is typically not allowed to query labels for data *in the past.*

Generally, the subset of data chosen for labeling are *biased* in the sense that they do *not* faithfully represent the original distribution of data. Therefore, optimizing a classifier based on this biased labeled set is problematic, especially in problems where the positive and negative classes are not separable. In the literature, importance sampling (IS) has been applied to reweight labeled data to remove the bias (e.g., [2, 4]). As with most importance-sampling-based methods, controlling variance is crucial for producing reliable classifiers in active learning.

As a motivating application, consider the detection of abusive user-generated content (UGC) on the Web. Common forms of UGC abuse include *commercial spam*, *offensive language*, *adult content*, etc. UGC abuse detection is challenging for many reasons. First, at Yahoo!, UGC is extremely diverse and Yahoo! receives millions of items a day, rendering it impractical for human arbitrators to examine every posted item. Because abusers can probe the detection system to determine what items are posted, they can easily bypass static, hand-coded detection rules. Machine-learned, automated detection is thus necessary. Second, similar to email spam detection, patterns of abuse evolve over time in a possibly adversarial manner. Therefore, the traditional train-once-and-deploy mode almost surely fails, as is confirmed by our analysis in Section 6. A natural solution to

tackle these challenges is to periodically retrain an abuse classifier by including new, publicly visible UGC as training data, whose labels are provided by paid human editors. Active learning can be used to reduce the editor's labeling efforts as new unlabeled data arrive.

This work has two contributions: we apply an online active learning paradigm [2] in a domain which is not IID, and develop a new algorithm using the importance weighting principle. This is the first paper that applies online active learning to a truly dynamic problem. Whereas [2] used a sequence of data drawn uniformly at random from the MNIST data set, we try to discriminate between commercial spam and good comments in user-generated content. We begin by establishing that the dataset has concept drift. Then, we show how online active learning is affected by this drift by comparing its performance on the data in its original order versus running the algorithm on a shuffled variant of the data. What this paper shows is that online active learning is more powerful in real dynamic problems than in environments where data arrives IID. This analysis is done using the probit model [3] to classify examples. We come up with an online update variant based on [15] that can handle weighted examples through an approximation technique. We also study a time-decay variant of the probit model.

2. PROBLEM SETTINGS

Let us denote by $\mathcal{X}$ the feature space and by $\mathcal{Y}$ the label space of input samples. An unknown distribution over $\mathcal{X} \times \mathcal{Y}$ is denoted by $p(\mathbf{x}, y)$, where $\mathbf{x}$ denotes a column vector of features and $y \in \{+1, -1\}$ in binary classification problems. Let $p(y|\mathbf{x}; \theta)$ be a predictive model parameterized by θ.

To optimize θ, we need a criterion to evaluate the goodness of a value of θ. Usually, the criterion is defined as

$$R(\theta) = \int \int G(\mathbf{x}, y; \theta) p(\mathbf{x}, y) dy d\mathbf{x}$$

where $G(\mathbf{x}, y; \theta)$ is a measurement function over samples. For instance, in the empirical risk minimization (ERM) framework, $G(\mathbf{x}, y; \theta)$ is usually a loss function, and the integral above is approximated by an empirical risk

$$\hat{R}_n = \frac{1}{n} \sum_{i=1}^{n} G(\mathbf{x}_i, y_i; \theta), \tag{1}$$

using n samples drawn i.i.d. from $p(\mathbf{x}, y)$.

In the setting of online active learning (a.k.a. selective labeling and selective sampling), we observe unlabeled samples $\mathbf{x}$ from $p(\cdot)$ sequentially, whereas an agent reveals y upon our request on $\mathbf{x}$. We deliberately select a subset of unlabeled samples for labeling, thus the resulting set of labeled samples might be distributed differently from $p(\mathbf{x})$.

When the labeled samples are drawn from an instrumental distribution $q(\mathbf{x}, y)$, according to the importance sampling principle, the empirical risk can be evaluated with reweighted labeled samples as follows,

$$\hat{R}_{n,q} = \frac{1}{\sum_{i=1}^{n} \frac{p(\mathbf{x}_i, y_i)}{q(\mathbf{x}_i, y_i)}} \sum_{i=1}^{n} \frac{p(\mathbf{x}_i, y_i)}{q(\mathbf{x}_i, y_i)} G(\mathbf{x}_i, y_i; \theta). \tag{2}$$

Since the instrumental distribution $q(\mathbf{x}, y)$ can be factorized by $p(y|\mathbf{x})q(\mathbf{x})$ in this case, the objective is then simplified to

$$\hat{R}_{n,q} = \frac{1}{\mathcal{B}} \sum_{i=1}^{n} \beta_i G(\mathbf{x}_i, y_i; \theta), \tag{3}$$

where $\beta_i = \frac{p(\mathbf{x}_i)}{q(\mathbf{x}_i)}$ and $\mathcal{B} = \sum_{i=1}^{n} \beta_i$. Note that $\beta_i \propto \frac{1}{q(\mathbf{x}_i)}$ if $\mathbf{x}_i$ is i.i.d. from $p(\mathbf{x})$.

2.1 Sampling Distribution

There is a strong connection between online active learning and adaptive importance sampling (AIS) [17]. A popular approach in AIS finds an optimal q to minimize risk variance: $q^* = \arg\min_q E_{\mathbf{x} \sim q} \left[\left(\hat{R}_{n,q} - E_{\mathbf{x} \sim q} \left[\hat{R}_{n,q} \right] \right)^2 \right]$, where the samples are drawn from $q(\mathbf{x})$. The following propositions ensure the soundness of (3).

PROPOSITION 1. *(Unbiasedness) If $\beta_i = \frac{p(\mathbf{x}_i)}{q(\mathbf{x}_i)}$, then we have:*
(1) $E_{\mathbf{x} \sim q}[\beta_i] = 1$;
(2) $E_{\mathbf{x} \sim q}[\mathcal{B}] = n$;
and
(3) $E_{\mathbf{x} \sim q}[\beta_i G(\mathbf{x}_i, y_i; \theta)] = R(\theta)$.

PROPOSITION 2. *(Asymptotic Variance) For $\hat{R}_{n,q}$ in (3), we have $\sqrt{n} \left(\hat{R}_{n,q} - R(\theta) \right) \to N(0, \sigma_q^2)$*
where $\sigma_q^2 = \int \left(\frac{p(\mathbf{x})}{q(\mathbf{x})} \right)^2 (G(\mathbf{x}, y; \theta) - R(\theta))^2 q(\mathbf{x}) d\mathbf{x}$.

In brief, Proposition 1 is a direct consequence of the definition of β_i. Proposition 2 follows from known results for the "Delta method" [17, 19, 22]. AIS minimizes the variance estimate σ_q^2 above using $q^*(\mathbf{x}) \propto p(\mathbf{x}) |G(\mathbf{x}, y; \theta) - R(\theta)|$. However, we do not know y in online active learning when making label request decisions. $G(\mathbf{x}, y; \theta)$ could be estimated by expected estimate, i.e., $\hat{G}(\mathbf{x}; \theta) = \sum_{y'} p(y'|\mathbf{x}; \theta) G(\mathbf{x}, y'; \theta)$. $R(\theta)$ could also be estimated by the current $\hat{R}_{n,q}$. In practice, if the expected risk is close to 0, we may simply choose $q^*(\mathbf{x}) \propto p(\mathbf{x}) |\hat{G}(\mathbf{x}; \theta)|$ with the estimate above.

As a special case, the measurement function $G(\mathbf{x}, y; \theta)$ in (1) can be chosen as $-\log p(y|\mathbf{x}; \theta)$, and the risk minimizer is equivalent to the Maximum Likelihood estimate. For the importance-weighted version (3), the minimizer is then equivalent to the Maximum Weighted Likelihood estimate.

3. ONLINE BAYESIAN PROBIT

In the ERM framework, stochastic gradient descent [24] is commonly applied to updating the model parameters θ. Given a weighted labeled sample $(\mathbf{x}_i, y_i, \beta_i)$ at time t, the update rule is $\theta_{t+1} = \theta_t + \eta \beta_i \frac{\partial G(\mathbf{x}_i, y_i; \theta)}{\partial \theta}$, where θ_t and θ_{t+1} denote the current and new parameters, respectively, and η the step size. Note that the selection probability β_i becomes a scaling factor of the step size. It is nontrivial to identify an appropriate step size and iteration number in stochastic gradient descent. Sometimes all labeled samples need be stored for periodic batch learning.

In this work, we resort to online Bayesian learning to maintain the posterior distribution of the weight vector in linear classifiers. For simplicity, we focus on linear models for problems involving binary classification. Denote by $\mathbf{w}$ the weight vector of the linear model at time t, and the function value is given by $\mathbf{x}^\top \mathbf{w}$. The distribution of $\mathbf{w}$ is modeled as a multi-variate Gaussian distribution with mean μ_t and covariance Σ_t:

$$p(\mathbf{w}_t) = N(\mathbf{w}; \mu_t, \Sigma_t). \tag{4}$$

Often, $\mu_0 = \mathbf{0}$ and $\Sigma_0 = \mathbf{I}$ are used for initialization. Suppose a labeled sample $(\mathbf{x}_i, y_i)$ is available for training at time t. The likelihood function is the *probit* function defined as:

$$\mathcal{P}(y_i|\mathbf{x}_i, \mathbf{w}) = \Phi(y_i \mathbf{x}_i^\top \mathbf{w}), \tag{5}$$

where $\Phi(z) = \int_{-\infty}^{z} N(v; 0, 1)\, dv$ is the cumulative distribution function of the standard Gaussian distribution. By Bayes' theorem, the posterior distribution of $\mathbf{w}$ is proportional to the product of the likelihood in (5) and the prior distribution in (4):

$$p(\mathbf{w}|\mathbf{x}_i, y_i) \propto \mathcal{P}(y_i|\mathbf{x}_i, \mathbf{w}) N(\mathbf{w}; \mu_t, \Sigma_t). \tag{6}$$

Unfortunately, the posterior is non-Gaussian. In practice, the first two moments of $\mathbf{w}$ are often used to construct a Gaussian approximation. Here, our approximation is based on a variational approach known as Adaptive Density Filter (ADF) [12, 15]. Given a posterior distribution p, ADF finds a Gaussian approximation that matches the first two moments of p. Specifically, let $N(\mathbf{w}; \mu_{t+1}, \Sigma_{t+1})$ be the target Gaussian, whose parameters $\{\mu_{t+1}, \Sigma_{t+1}\}$ are chosen to minimize the Kullback-Leibler divergence:

$$\min \mathbf{KL}\left(\Phi(y_i \mathbf{x}_i^\top \mathbf{w}) N(\mathbf{w}; \mu_t, \Sigma_t) \| N(\mathbf{w}; \mu_{t+1}, \Sigma_{t+1})\right).$$

This optimization problem can be solved analytically by moment matching up to the second order, yielding:

$$\mu_{t+1} = \mu_t + \alpha\,(\Sigma_t \mathbf{x}_i) \tag{7}$$

$$\Sigma_{t+1} = \Sigma_t - \delta\,(\Sigma_t \mathbf{x}_i)(\Sigma_t \mathbf{x}_i)^\top \tag{8}$$

where

$$\alpha = \frac{y_i}{\sqrt{\mathbf{x}_i^\top \Sigma_t\, \mathbf{x}_i + 1}} \frac{N(z)}{\Phi(z)}$$

$$\delta = \frac{1}{\sqrt{\mathbf{x}_i^\top \Sigma_t\, \mathbf{x}_i + 1}} \frac{N(z)}{\Phi(z)} \left(\frac{N(z)}{\Phi(z)} + z\right)$$

with $z = \frac{y_i \mathbf{x}_i^\top \mu_t}{\sqrt{\mathbf{x}_i^\top \Sigma_t \mathbf{x}_i + 1}}$. If the dimension of $\mathbf{x}_i$ is high, the covariance matrix Σ_t can be restricted to be *diagonal*. This restriction corresponds to the idea of mean-field approximation; see [9] for a successful application of this method in a search engine setting. Then, the parameter update above takes $\mathcal{O}(d)$ time on average, where d is the average number of non-zero features. Hereafter, we focus on diagonal covariance matrices only.

In online active learning, unlabeled samples come in sequentially. For an unlabeled sample, we may use the current classifier's predictive distribution on the label to decide whether to request its label. Given parameters $\theta = \{\mu_t, \Sigma_t\}$ of the classifier at time t, the predictive distribution for sample $\mathbf{x}$ is given by: $\mathcal{P}(y|\mathbf{x}; \theta) = \int \Phi(y \mathbf{x}^\top \mathbf{w}) N(\mathbf{w}; \mu_t, \Sigma_t)\, d\mathbf{w}$, where $\Phi(y \mathbf{x}^\top \mathbf{w})$ is the probit likelihood and $N(\mathbf{w}; \mu_t, \Sigma_t)$ is the approximate posterior distribution of $\mathbf{w}$ at time t. The integral can be exactly computed by

$$\mathcal{P}(y = +1|\mathbf{x}; \theta) = \Phi\left(\frac{\mathbf{x}^\top \mu_t}{\sqrt{\mathbf{x}^\top \Sigma_t\, \mathbf{x} + 1}}\right). \tag{9}$$

In the ERM framework, class label prediction is based on function value only (e.g., $\mathbf{x}^\top \mathbf{w}$ or $\Phi(\mathbf{x}^\top \mathbf{w})$). To visualize the conceptual difference between the predictive probability in (9) and the function-value-only prediction, we present a synthetic case in two-dimensional feature space in Figure 1.

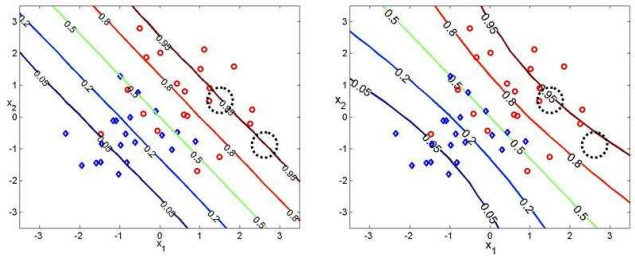

Figure 1: An illustration on predictive probabilities (right) in contrast with function values (left).

The red circles and blue diamonds represent the labeled samples of two classes used for training. In the right graph, the contour curves are indexed with predictive probabilities of the class of red circles. In the left graph, the contour curves are indexed with confidence level of belonging to the class of red circles based on function values only. The two dotted circles in black indicate two regions of interest where unlabeled samples come from. One region is about the center of red circles with predictive probability 0.95, and another region is a bit far away from training samples. In function-value-only prediction, the latter region is also predicted with probability 0.95 because of the hyper-plane determined by $\mathbf{w}$, whereas the predictive probability in (9) lowers the confidence on the latter region. The advantage comes from the quadratic term $\mathbf{x}^\top \Sigma_t\, \mathbf{x}$ in the denominator, which tends to be large on regions far from training samples. In some high-dimensional feature spaces, the difference on predictive uncertainty could be more significant. Even when the covariance matrix Σ_t is constrained to be diagonal, different attributes could have different uncertainties.

To summarize, three critical advantages are identified for online Bayesian learning:

- The step size per example can be computed analytically using ADF, without the need for a line search or learning rate tuning;
- Consistency in predictive class probabilities, which can be utilized by Bayesian decision theory to find optimal cutoffs for various utility functions;
- Model uncertainty is explicitly considered, which makes active learning more sensible on novel samples, especially in high-dimensional spaces.

3.1 Weighted Likelihood

Online active learning induces a stochastic process that selects unlabeled samples with certain probability for labeling. The selected samples come from an instrumental distribution $q(\cdot)$ rather than $p(\cdot)$, the distribution of interest. We use the weighted likelihood bootstrap sampling procedure [16] to select the samples to be sent for labeling. Suppose we obtain a labeled sample $(\mathbf{x}_i, y_i, \beta_i)$ through online active learning, where β_i denotes the sampling probability of $\mathbf{x}_i$. The weighted likelihood function is then defined by

$$\mathcal{P}(y_i|\mathbf{x}_i, \mathbf{w}, \beta_i) = \mathcal{P}^{\beta_i}(y_i|\mathbf{x}_i, \mathbf{w}) \tag{10}$$

where $\beta_i = \frac{1}{q(\mathbf{x}_i)}$ is the importance weight. Asymptotic properties of maximum weighted likelihood estimators have been studied [22]. In this section, we give approximate up-

date rules coupled with weighted likelihood to update the model in an online fashion. The variational approximation as in (7) and (8) finds an optimal Gaussian approximation. Now we attempt to find a Gaussian approximation, parameterized by $\check{\mu}_{t+1}$ and $\check{\Sigma}_{t+1}$, which is adjusted to the weighted likelihood as in (10). Based on the approximation $N(\mathbf{w}; \mu_{t+1}, \Sigma_{t+1})$ and the weight β_i in weighted likelihood [9], we have

$$\check{\Sigma}_{t+1}^{-1} - \Sigma_t^{-1} = \beta_i(\Sigma_{t+1}^{-1} - \Sigma_t^{-1})$$

$$\check{\Sigma}_{t+1}^{-1}\check{\mu}_{t+1} - \Sigma_t^{-1}\mu_t = \beta_i(\Sigma_{t+1}^{-1}\mu_{t+1} - \Sigma_t^{-1}\mu_t)$$

Reorganizing terms, we reach the following update rules:

$$\check{\Sigma}_{t+1} = \Sigma_{t+1}(\beta_i\Sigma_t + (1-\beta_i)\Sigma_{t+1})^{-1}\Sigma_t \quad (11)$$

$$\check{\mu}_{t+1} = \check{\Sigma}_{t+1}(\beta_i\Sigma_{t+1}^{-1}\mu_{t+1} + (1-\beta_i)\Sigma_t^{-1}\mu_t) \quad (12)$$

These rules are in the same spirit of the scaling effect on step size in stochastic gradient descent in the ERM framework.

3.2 Dynamics

So far the model assumes a stationary data distribution $p(\mathbf{x}, y)$ and then the variance on weights would converge towards zero as labeled samples increase. Gradually, the model would stop learning. We are handling time series patterns in data streams, so that the model we have investigated should be capable of adapting to changes in *non-stationary* situations. If we can access all historical labeled samples, we can introduce a decay factor to decrease the influence on the posterior from outdated samples. This decay can be achieved by down-weighting the likelihood of such out-of-date samples. However, in online active learning, we cannot store all training samples for revisiting their weighted likelihood. Fortunately we can introduce a memory loss factor on the current model, the prior distribution of $\mathbf{w}$ at time t. Then the joint likelihood becomes

$$\mathcal{P}^{\beta_i}(y_i|\mathbf{x}_i, \mathbf{w})N^{\gamma}(\mathbf{w}; \mu_t, \Sigma_t)$$

where γ is the loss factor and $0 \ll \gamma \leq 1$. Dynamics corrections can be derived analogously as in (11) and (12) by replacing the 1's by γ.

3.3 Sampling Distribution

According to Proposition 2, the instrumental distribution $q(\cdot)$ should be proportional to the entropy estimate. Since the current model may deviate from the truth, we introduce a tolerance parameter ϵ to reflect misclassification rate, where $0 \leq \epsilon \ll 1$, so that the predictive probability becomes

$$\check{p}(y|\mathbf{x}; \theta) = \epsilon + (1 - 2\epsilon)p(y|\mathbf{x}; \theta),$$

where the predictive probability $p(y|\mathbf{x}; \theta)$ is defined as in (9). The instrumental distribution we applied in this work is defined as follows:

$$q(\mathbf{x}) = \sum_y -\check{p}(y|\mathbf{x}; \theta)\log_2(\check{p}(y|\mathbf{x}; \theta)). \quad (13)$$

The right panel of Figure 2 plots example curves of the labeling probability again the function value $\mathbf{x}^\top\mathbf{w}$ for a fixed $\epsilon = 0.1$ and different values of $\sigma^2 = \mathbf{x}^\top\Sigma\mathbf{x}$. The left panel plots the labeling probability given by [4], which is based on function values only: $q(\mathbf{x}) \propto \frac{b}{b+|\mathbf{x}^\top\mathbf{w}|}$. Both instrumental distributions in Figure 2 are conceptually similar while the key difference lies in our method's capability of modifying labeling probability based on prediction uncertainty.

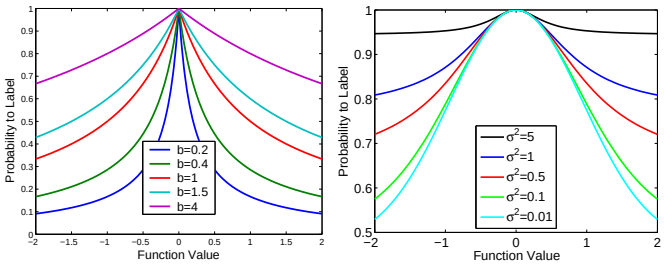

Figure 2: Label selection probabilities based on function value (left) and entropy (right).

Algorithm 1 Online Active Learning with Bayesian Probit

Input: data stream $\{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$, classifier $\theta = \{\mu, \Sigma\}$
$t \leftarrow 0$, $\mu \leftarrow \mathbf{0}$, $\Sigma \leftarrow \mathbf{I}$
repeat
  $t \leftarrow t + 1$, $H_t \leftarrow p(+1|\mathbf{x}_t; \theta)$,
  sample $\tau \sim \mathcal{U}_{(0,1)}$
  if $\tau < q(\mathbf{x}_t)$ **then**
    acquire label y_t
    update model $\{\mu, \Sigma\}$ by $(\mathbf{x}_t, y_t, \beta_t)$
  end if
until the end of data stream
Output: $\{\mu, \Sigma\}$ and predictive probabilities $\{H_1, H_2, \ldots\}$

Algorithm 1 summarizes our algorithm for online active learning based on the techniques discussed in this section.

4. DISCUSSION

When we make IID assumptions on non-IID data, there can be undesirable consequences. In this section, we discuss two simple examples of what can go wrong. In particular, we ask:

1. Is there an example of a problem where shuffling the data would not improve performance, but training only on early data would?
2. Is there an example of a problem where shuffling the data would improve performance?

The first example is straightforward. To connect it to the spam problem, we call it the novel spam attack problem.

EXAMPLE 1. ***The Novel Spam Attack Problem:*** *Over N days, there are N distinct messages in the dataset: no distinct message shares any features with another distinct message. The first distinct message occurs on the first day k times, the second distinct message occurs on the second day k times, etc. The label of each distinct message is random.*

Note that once a message has been seen and labeled, it will never be misclassified again. Thus, an algorithm that memorizes examples will make $N/2$ mistakes. If the data is shuffled, this will not improve the result, because there is no way to generalize between distinct messages. On the other hand, an algorithm that trains only on the first day will get the first day right, but will at best in expectation make $(N-1)k/2$ mistakes on the rest of the data.

A second example shows where shuffling does have an impact. It focuses on a known spammer approach, where they create e-mails that would be misclassified by the current

classifier, but keep the problem separable. It is related to a classic problem in online learning, that the VC dimension can be low while the mistake bounds are infinite. Consider learning a threshold on the $[0, 1]$ interval.

EXAMPLE 2. ***Spammer Splitting the Difference:*** *Define $\mathcal{X} = [0, 1]$. Let a be the highest value of a negative example seen so far (or 0 if no negative example has seen so far), and b the lowest value of a positive example seen so far (or 1 if no positive example has been seen so far). The next instance is $\frac{a+b}{2}$, and the probability that it is positive is $\frac{1}{2}$.*

This is an example when, given the examples in order, there is almost nothing to be done. Even though all positive examples have higher values than the negative ones, the positive examples monotonically decrease in the instance space, and the negative examples monotonically increase. But when the examples are shuffled, it becomes much easier, for this monotonicity can be leveraged. If, when you see an example x, you have seen another example x' with a positive label where $x > x'$, then x is positive. Similarly, if you have seen another example x'' with a positive label where $x < x''$, then x is negative. Define p to be the number of positive examples, and $\{x_1 \dots x_p\}$ to be the positive examples in order of arrival. So, if we consider shuffling the positive indices, we only get a positive examples wrong if it is the highest number seen so far. From folklore, if you shuffle the numbers from 1 to p, the number of times you see a number higher than any you have seen so far is $\Theta(\log p)$ times. Thus, if there are m examples, then one will make $\Theta(m)$ mistakes if the data is unshuffled, but if the data is shuffled, then one will make $\Theta(\log m)$ mistakes.

Thus, the ability for spammers to push closer and closer to the boundary over time can be incredibly powerful. However, note that if shuffling improves performance directly, this implies a connection between spam attacks. If spam attacks were completely novel, then shuffling would not improve performance.

5. RELATED WORK

Given unlabeled examples, two types of active learning scenarios exist [20]. In the *pool-based* scenario (e.g., [13]), a *static* set of unlabeled examples are given, from which a subset is chosen for labeling. The setting considered in the present paper falls into the other scenario known as *stream-based* [5, 7], in which the learner has to decide, for each example in the data stream, whether to query a label (and then updates the classifier if a label is obtained). For applications like UGC abuse detection where examples arrive sequentially, the stream-based model is more natural.

Many heuristics exist for scoring the *informativeness* of unlabeled examples in active learning, including those based on uncertainty sampling [13], the principle of maximal disagreement [8, 21], model variance minimization [6, 23], and version/model space pruning [1, 5, 14], etc. Most relevant to this work is probably the heuristic based on estimated error reduction [10, 18], where an unlabeled example is scored based on the estimated reduction in the classification error if that example is labeled and included in the training set. However, most of these selection criteria are deterministic. Hence, the labeled training data can represent a distribution that is very different from the original data distribution, which introduces bias in the learning process. In contrast, our active learning scheme is probabilistic, so that importance weighting is easily incorporated to remove bias in the labeled data. This property is crucial as it ensures that the learned classifier is optimized against the right distribution.

The idea of using importance weighting for de-biasing has been used in prior work. [4] combines probabilistic selective sampling in Perceptron-like algorithms. However, these algorithms do not explicitly control variance, so the mistake bounds become larger as variance increases. The IWAL algorithm [2], on the other hand, guarantees bounded variance by a careful specification of the labeling probabilities, but calculating the probabilities is expensive except for special cases. Our algorithm takes a further step and directly relates the labeling probabilities to the variance of the risk of the final classifier, and the probabilities are easier to compute.

Regarding metrics in online learning, it is worth noting that [11] has shown that the AUC measure has a fundamental limitation since it implicitly uses different misclassification cost distributions for different classifiers. To overcome the incoherence in terms of misclassification costs, a valid alternative to the AUC was proposed.

6. EXPERIMENTAL RESULTS

This section reports experimental results on commercial spam moderation in UGC to verify the usefulness of the proposed algorithm for online active learning. We implemented online probit regression model with diagonal covariance matrix as in [9] as the classifier. For comparison purposes, three unbiased active learning strategies were implemented to couple with the Bayesian classifier:

- Entropy-based criteria: The sampling probability is a function of entropy estimate as in (13), see the right graph in Figure 2;
- Function-value-based criteria: The sampling probability is defined as $\frac{b}{b+|\mathbf{x}^\top \mathbf{w}|}$ proposed by [4], see the left graph in Figure 2;
- Random: As a baseline approach, a fraction of samples are selected randomly to labeling.

We have also implemented their corresponding biased variants, i.e. selected samples are equally weighted rather than $\frac{1}{q(\mathbf{x})}$, to show the advantages of ensuring unbiasedness.

6.1 Settings

We give a brief introduction to data collection, and then highlight temporal concept drift in the data stream.

6.1.1 UGC Spam Detection

On a commercial news portal, users are allowed to post comments to discuss news articles. Around the end of 2010, we selected a subset of UGC to human editors for commercial spam detection. The data collection lasted 30 days. The selection rule was kept unchanged in the 30 days. Human editors manually examined the comment content and gave a label such as clean, commercial spam or other types of abuse. Based on the editorial labels, we collected clean UGC and commercial spam UGC into our data set. Our goal here is to build an online classifier to distinguish commercial spam from clean UGC. The UGC samples are indexed by submission timestamp as in time series. We extracted features from UGC text content only, including text analysis results, named entities and TFIDF. There are about 0.26 million UGC samples and 0.27 million features in the data set. The

feature vector of a UGC sample is usually sparse, containing about 90 non-zero elements on average. The daily commercial spam rate is about 1% $\sim$ 5%. It should be noted that the spam rate in our data set does not represent the real-world spam rate in the commercial news portal at all, though daily spam patterns have been well preserved in the sample selection procedure.

6.1.2 Evaluation Methodology

The samples from the first day were used to train a binary classifier as an initial model, and the samples from the remaining days were used for evaluation. The UGC data set contains evaluation data from 29 days. The samples in the remaining days were sorted by the time that the comment was posted. The online Bayesian classifiers in all candidate models were initialized by the first-day model in the same way. The simulation was then carried out as follows: (a) The classifier takes the next sample in the data stream and makes a prediction on its label, named as a one-step-ahead prediction; (b) the active learning algorithm decided whether to acquire the editorial label or not; (c) if label acquisition was requested, the editorial label was revealed without any delay and a model update was executed by the online classifier using the labeled sample as in Section 3.1; (d) steps (a), (b) and (c) were repeated until the end of the data stream.

To evaluate generalization performance, we took the results of one-step-ahead predictions in the simulation, i.e. the output H in Algorithm 1, and computed the following metrics:

- Negative logarithm of predictive probability (NLP) is defined as $\frac{1}{N}\sum_{i=1}^{N} -\log p(y_i|\mathbf{x}_i)$ where $p(y_i|\mathbf{x}_i)$ is the predictive probability given by the classifier as in (9). NLP is 0 for a perfect classifier;
- Area under ROC (AUC) is a popular performance metric to measure the quality of predictive ordering on classification datasets. In our experiments, predictive ordering was determined by sorting the predictive probabilities $p(y_i = +1|\mathbf{x}_i)$ in descending order. AUC is 1 for perfect ordering and 0.5 for random guessing;
- Weighted loss (WL) is defined as $c_n * FN + c_p * FP$ where c_n and c_p are pre-fixed cost scalar, FN is the number of false negative predictions, and FP is the number of false positive predictions. Based on Bayesian decision theory, the optimal threshold is $c_p/(c_p + c_n)$ rather than 0.5 in the cost-sensitive setting, i.e. the classifier yields positive label if $p(y_i = +1|\mathbf{x}_i) \geq c_p/(c_p + c_n)$, otherwise negative label.[1] In the following case study, we set $c_n = 9$ and $c_p = 1$. This tradeoff makes sense in user generated data, because if a comment is blocked, then a real person becomes immediately aware that his or her comment was blocked and can try again, and if spam gets through, then many readers are inconvenienced.

To veil business-sensitive information, we only report *relative gains* over the baseline performance of the first-day initial model. For NLP and WL, the relative gain is defined as 1-Score/Baseline, while for AUC it is Score/Baseline-1. In all cases, a larger metric implies better performance.

[1]We assume our classifier's predictive probability is correct. The loss to give away positive label (but the true label is negative) is $c_p(1 - p(y_i = +1|\mathbf{x}_i))$, whereas the loss of negative label is $c_n p(y_i = +1|\mathbf{x}_i)$. The losses incurred by the two opposite actions are balanced only when $p(y_i = +1|\mathbf{x}_i) = c_p/(c_p + c_n)$.

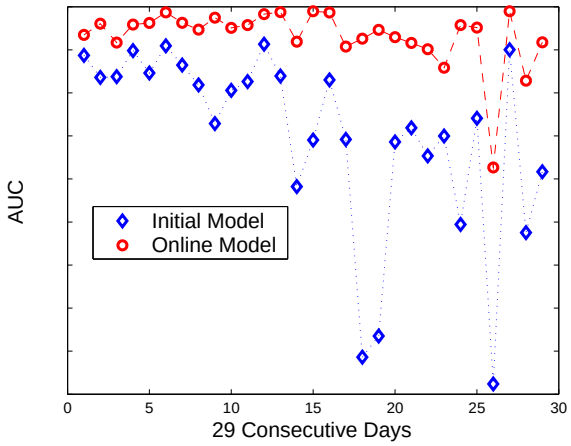

Figure 3: Initial vs. Online model on daily AUC.

γ	AUC	NLP	WL
1.0	0.04161	0.5373	0.7116
0.99999	0.04217	0.5451	0.7135
0.99995	0.04312	0.5676	0.7428
0.9999	0.04388	0.5750	0.7468
0.9995	0.03832	0.4426	0.5871
0.999	0.03065	0.2857	0.3816

Table 1: Online learning performance with decay.

6.1.3 Pattern Change

To illustrate the existence of concept drift or pattern change in our data set, we compared the initial model against an online classifier on the evaluation data. The initial model was built using the first day's data and was kept unchanged in the test. The online classifier was updated as in Algorithm 1 but without selective labeling in the simulation, i.e., every label is revealed in time. We present the comparisons in Figure 3 as empirical evidence. We observed that the initial model suffers by performance degradation especially after 10 days indicating a change in the data. The online classifier is capable of learning pattern changes in a timely manner to handle the changing data over time. The existence of concept drifting in the data stream is further verified in Section 6.3.

6.1.4 Memory Decay Factor

We also compared online classifiers with different decay factors on the UGC evaluation data. The decay factor was varied from 1.0 to 0.999. In Table 1, we report three performance metrics for four online classifiers. Note that in this study, selective labeling was not involved. We revealed sample labels to the classifier for online model update after every one-step-ahead prediction. We observed that the online classifier with a decay factor of 0.99999 or 0.9999 yields better performance than the online classifier without memory decay. The best performance is achieved with decay factor

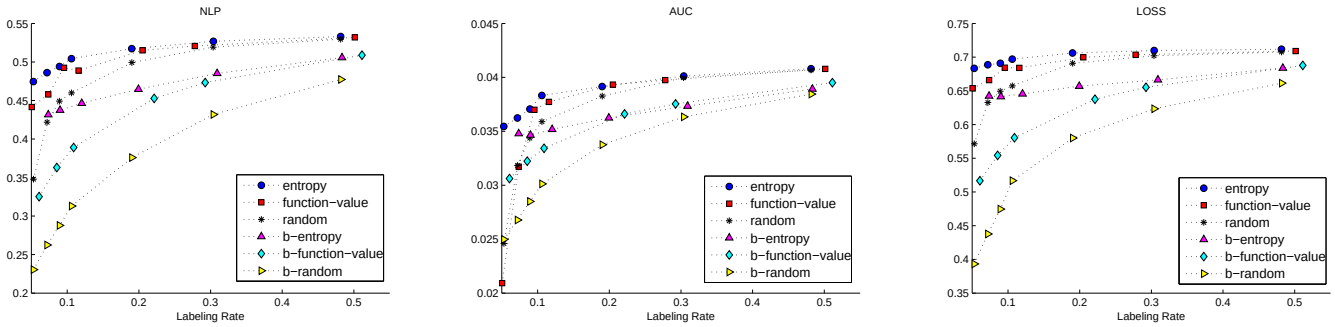

Figure 4: Generalization performance at different labeling rates.

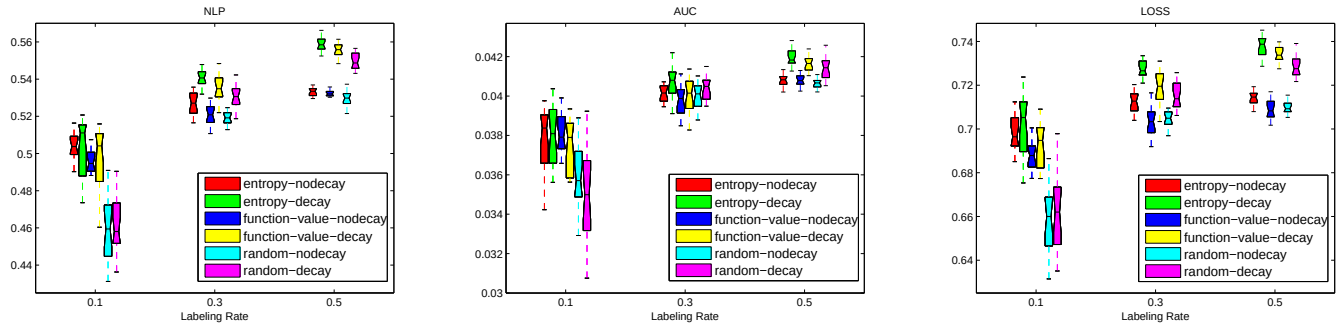

Figure 5: Generalization performance with and without the decay factor. For each different labeling rate in each plot, the three labeling strategies from left to right are entropy-based, function-value-based, and random; the results with and without decaying for each strategy are listed side-by-side. The box has lines at the lower quartile, median, and upper quartile values.

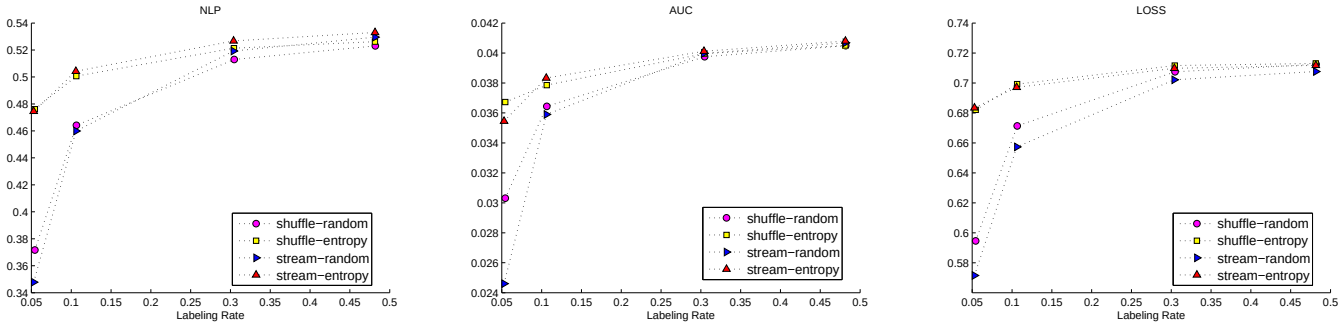

Figure 6: Generalization performance on shuffled data vs. on data stream, at different labeling rates.

0.9999. When the factor goes down to 0.9995, the performance becomes worse than that without memory decay. We also observed that the three metrics, NLP, AUC and WL, are consistently correlated. The benefits we observed from the decay factor also support the conjecture that there are temporal changes of the spam pattern or input distribution in the one-month UGC data stream.

6.2 Online Active Learning

We tested six candidate models of the online probit classifier. These six models were realized by combining three active learning strategies (entropy-based criteria, the function-value-based criteria, and random selection), with two bias types (unbiased and biased). The six candidate models were initialized by the same first-day model. The labeling rate in entropy-based algorithms is controlled by the tolerance parameter ϵ. We varied ϵ from 0.1 to 0.0025, and the resulting label rate decreased from 50% to 5%. The labeling rate in

function-value-based algorithms is controlled by the parameter b. We varied b from 4 to 0.2 and the resulting label rate decreased from 50% to 5% as well. We explicitly specified selection rates in the random models at the rates realized by the entropy models. We report our results averaged over 20 trials on the evaluation data.

Figure 4 presents the results (for various metrics) of each model *without decay* on the evaluation data. Each graph presents generalization results of the six models at different labeling rates ranging from 5% to 50%, averaged over 20 trials.

Figure 5 presents the results (for various metrics) for the unbiased models with decay factor 0.9999 on the evaluation data stream. Each boxplot present generalization results of the six models at three labeling rates, 10%, 30% and 50%, averaged over 20 trials.

The three biased models yield inferior performance, compared to their unbiased counterparts. In general the unbiased models converge faster to the optimal than the biased models. In Figure 4, the optimal performance is bounded by the online learning result without decay factor $\gamma = 1$ given in Table 1. The gap between the biased and unbiased random model emphasizes the well-known fact that the learning rate is critical in online sequential learning. Around a labeling rate of 0.7, the biased entropy model can catch up with the unbiased random model. Within the three biased models, we found that the function-value and entropy models are consistently better than the random model at all labeling rates. When the labeling rate is below 20%, the entropy model performs significantly better than the function-value model. Within the three unbiased models, we found that the function-value and entropy models are consistently better than the random model. When the labeling rate is above 20%, although the function-value and entropy models are usually better than the random model, the difference is insignificant. The entropy model performs significantly better than the function-value model when the labeling rate is below 10%, as shown in Figure 5. In Figure 5, we observed that the active learning algorithms with memory decay are capable of achieving superior performance on the three metrics, especially when the labeling rate is greater than 30%. The boxplot graphs in Figure 5 also visualize performance results of the three unbiased active learning models for the 20 trials. When the labeling rate is 10%, the entropy-based model without decay is significantly better than the function-value-based model on both NLP and LOSS metrics.

6.3 Shuffled Data

In the last set of experiments, we compared active learning results on shuffled data with those on the data stream in its original order. We randomly shuffled the evaluation data to generate a new data stream, and then report the performance of the entropy model and the random model on the shuffled data. Both models are run without decay for the purpose of comparison. We present the generalization performance averaged over 20 trials, at different labeling rates ranging from 5% to 50%. The performances of the random model on the two data sets reveal the intrinsic difficulty of the two learning tasks. Since the random model consistently yields better metrics on the shuffled data when the labeling rate is below 30%, we believe that classification on the original data stream is more difficult than on the shuffled data. The underlying reason might be due to something similar to Example 2, but it is hard to know for sure. The learning tasks become even harder as the label rate decreases. On the two tasks, the entropy-based active learning approach performs consistently better than the random model. We also observed that the entropy model yields comparable results on both cases. If we look at the 0-1 loss, while the shuffled problem seems to be significantly harder given a random sampling of data, it seems that the active learning algorithm does equally well on both problems. This demonstrates that the lift obtainable in the original online problem is higher than that obtained in the shuffled problem. Thus, in this dynamic environment, active learning provides more benefits than in the stationary analogue of the problem.

7. CONCLUSIONS

In this work, we studied online active learning in dynamic problems with potentially adversarial concept drifts. It is shown, using real UGC data from a news portal at Yahoo!, that active learning is powerful for reducing labeling efforts in dynamic problems. Furthermore, motivated by practical challenges in spam detection from user generated content, we proposed and tested an online Bayesian algorithm that can handle sample weighing and forgetting, both of which are required by online active learning. Empirical results show the effectiveness of the algorithm.

The promising results in the paper raise a few important questions. First, it is interesting to analyze regret or mistake bounds for online active learning algorithms, similar to those developed for stationary problems (e.g., [4]), when there is concept drift. Obviously, little can be hoped for if arbitrary concept drifts are allowed, so reasonable assumptions are necessary. Second, we haven't studied how to adapt the value of γ in online model updates, which is an important question in practice. Third, pool-based and stream-based scenarios represent two extremes of practical situations. In reality, it is often possible, and even advantageous, to keep a small amount of unlabeled data as to-be-labeled candidates for the data stream case, so that the algorithm can revisit not-so-old examples to decide whether or not their labels are required. It is not straightforward to ensure unbiasedness in such hybrid architectures.

For future work, it is interesting to carry out controlled experiments on simulated datasets in which the concept drift process is predefined, to verify that online active learning is more powerful in dynamic environments. It is also worth comparing our approach to stochastic gradient decent. Finally, the labeling rate in online active learning is indirectly controlled through tolerance parameters in a highly nonlinear fashion, which may lead to prohibitive workload for labeling occasionally. The impact of realistic constraints in label acquisition needs to be investigated as well.

8. ACKNOWLEDGEMENTS

We appreciate the support from Pramod Khincha's engineer team in data collection.

9. REFERENCES

[1] M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proceedings of the Twenty-Third International Conference on Machine Learning (ICML-06)*, pages 65–72, 2006.

[2] A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML-09)*, pages 49–56, 2009.

[3] C. I. Bliss. The calculation of the dosage-mortality curve. *Annals of Applied Biology*, 22:134–167, 1935.

[4] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear classification. *Journal of Machine Learning Research*, 7:1205–1230, 2006.

[5] D. A. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[6] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[7] I. Dagan and S. P. Engelson. Committee-based sampling for training probabilistic classifiers. In *Proceedings of the Twelfth International Conference on Machine Learning (ICML-95)*, pages 150–157, 1995.

[8] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2–3):133–168, 1997.

[9] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning (ICML-10)*, pages 13–20, 2010.

[10] Y. Guo and R. Greiner. Optimistic active-learning using mutual information. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 823–829, 2007.

[11] D. J. Hand. Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine Learning*, 77:103–123, 2009.

[12] N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 609–616, 2002.

[13] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR-94)*, pages 3–12, 1994.

[14] D. J. C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

[15] T. P. Minka. *A family of algorithms for approximate Bayesian inference*. Ph.D. thesis, Massachusetts Institute of Technology, January 2001.

[16] M. A. Newton and A. E. Raftery. Approximate Bayesian inference by the weighted likelihood bootstrap. Technical Report No. 199, Department of Statistics, University of Washington, March 1991.

[17] M.-S. Oh and J. O. Berger. Adaptive importance sampling in Monte Carlo integration. *Journal of Statistical Computation and Simulation*, 41:143–168, 1992.

[18] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML-01)*, pages 441–448, 2001.

[19] C. Sawade, N. Landwehr, S. Bickel, and T. Scheffer. Active risk estimation. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning (ICML-10)*, pages 951–958, 2010.

[20] B. Settles. Active learning literature survey. Technical Report 1648, Department of Computer Sciences, University of Wisconsin-Madison, January 2009.

[21] H. S. Seung, M. Opper, and N. Tishby. Query by committee. In *Proceedings of the Fifth Annual Conference on Computational Learning Theory (COLT-92)*, pages 287–294, 1992.

[22] X. Wang, C. van Eeden, and J. V. Zidek. Asymptotic properties of maximum weighted likelihood estimators. *Journal of Statistical Planning and Inference*, pages 37–54, 2004.

[23] T. Zhang and F. J. Oles. A probability analysis on the value of unlabeled data for classification problems. In *Proceedings of Seventeenth International Conference on Machine Learning (ICML-00)*, pages 1191–1198, 2000.

[24] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.