Active Learning for Black-box Models

Neil Rubens[*], Vera Sheinman[†], Toshio Okamoto[*], Maomi Ueno[*]

Abstract

Active learning refers to the settings in which a machine learning algorithm (learner) is able to select data from which it learns, and by doing so aims to achieve a better accuracy (e.g. by avoiding obtaining training data that is redundant or unimportant). Active learning is particularly useful in cases where obtaining training data is costly.

A common assumption is that an active learning algorithm is fully aware of the details of an underlying learning algorithm for which it obtains the data. However, in many real world settings, obtaining precise details of the learning algorithm may not be feasible, making the underlying algorithm in essense a black box – no knowledge of internal workings of the algorithm, where only the inputs and corresponding outputs are acessible. We note that accuracy will improve only if the learner's outputs change. Motivated by this, we select a training point that is expected to cause many changes in the learner's outputs, in the anticipation that the resulting changes will be for the better.

1 Introduction

The goal of supervised learning is to learn a function that allows to accurately predict the output for previously unseen inputs. A function is learned from the training data, consisting of inputs and outputs from the unknown target function. A popular phrase in computer science 'Garbage in, Garbage Out' summarizes well the importance of the training data in the learning process. Obtaining output values (labeling) often incurs a cost (in terms of money, effort, time, availability, etc.). While the cost of obtaining an output value may be the same, the degree to which a training point allows us to approximate the function varies (Figure 1). The goal of active learning (AL) is to select input points to label as to maximize the accuracy of the learned function. What makes the AL task challenging is that we have to predict the improvement in the accuracy of the learned function with regards to the input point before its output value is obtained. Since once the output value is obtained it incurs a cost.

A common assumption is that an active learning algorithm is fully aware of the details of an underlying learning algorithm for which it obtains the data. However, in many real world settings obtaining precise details of the learning algorithm may not

[*]Graduate School of Information Systems, University of Electro-Communications, Tokyo, Japan
[†]Japan Institute for Educational Measurement, Tokyo, Japan

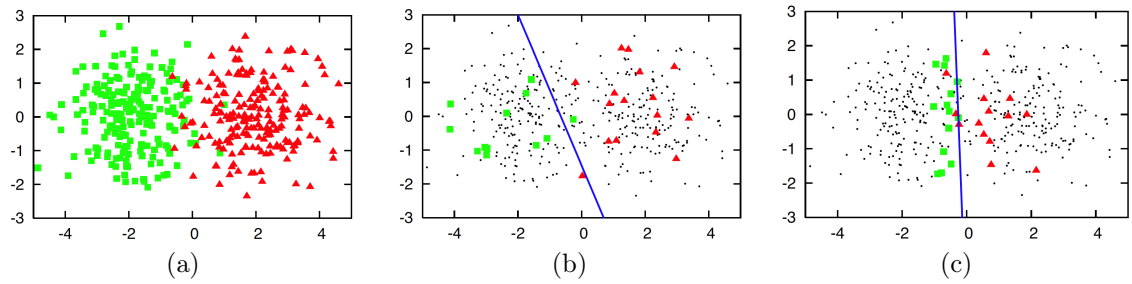

Figure 1: Utilizing training points selected by AL method (1c), allows to more accurately predict the true values (1a), in comparison with selecting training points randomly (1b) [2].

be feasible, e.g. learning algorithm could be very complex, consisting of many models and modules that are developed independently. Even if model details are available, developing active learning algorithms for complex models could be very difficult. In addition, active learning criterion may need to be reformulated each time the underlying model changes.

2 Problem Formulation

Supervised Learning Let us define the problem of active learning in a more formal manner. An input variable is cosindered to be a multi-dimensional data point and is denoted by a vector $\mathbf{x} \in \mathbb{R}^p$, where p is a number of attributes/features. The set of all points is denoted by $\mathcal{X}$. The target function that we are trying to approximate is denoted by f, and its output value (also referred to as label) is denoted as $f(\mathbf{x}) = y \in \mathbb{R}$. The set of training input points is denoted by $\mathcal{X}^{(Train)}$, and these points along with their corresponding output values are referred to as a *training set*, i.e. $\mathcal{T} = \{(\mathbf{x}_i, y_i\}_{\mathbf{x}_i \in \mathcal{X}^{(Train)}}$. The task of supervised learning is, given a training set, to learn an estimate $\widehat{f}$ of the target function f by, the estimated output value is denoted as $\widehat{y} = \widehat{f}(\mathbf{x})$. We measure how accurately the learned function predicts the true output values by the generalization error estimated based on the test set: $G(\widehat{f}) = \sum_{\mathbf{x} \in \mathcal{X}^{(Test)}} \mathcal{L}\left(f(\mathbf{x}), \widehat{f}(\mathbf{x})\right) P(x)$,where $\mathcal{X}^{(Test)}$ refers to the *test set*, and prediction errors are quantified by a loss function $\mathcal{L}$.

Active Learning We consider that we are allowed to sequentially select which inputs will be labeled. The active learning criterion aims to estimate the usefulness of labeling an input $\mathbf{x}$ (and adding it to the training set $\mathcal{T}$). In our case usefulness is defined as the minimization of generalization error with respect to the training set. The active learning criterion could then be formulated as: $\widehat{G}(\mathbf{x}) = \widehat{G}(\mathcal{X}^{(Train)} \cup \{\mathbf{x}\})$. For example, if we consider labeling a point $\mathbf{x}_j$ or a point $\mathbf{x}_k$, then we would estimate their usefulness by an active learning criterion, i.e. $\widehat{G}(\mathbf{x}_j)$ and $\widehat{G}(\mathbf{x}_k)$, and select the one that will result in a smaller generalization error. Note that we need to estimate the usefulness of labeling the point without knowing its actual label. To distinguish a candidate point

to be labled from the other points we refer to it as $\mathbf{x}_\delta$. The goal of active learning can then be stated as selecting an input point $\mathbf{x}$ to be labeld, so that after adding it to the training set the the generalization error will be minimized: $argmin_{\mathbf{x}}\widehat{G}(\mathbf{x})$.

Black-box Settings In black-box settings the details of learned function $\widehat{f}$ are not accessible, however its output estimates $\widehat{y} = \widehat{f}(\mathbf{x})$ are accessible.

3 Proposed Method

In out previous work [1], we have investigated the efficiency of the proposed criterion when the underlying model was assumed to be known (a linear regression model in our case). In this work, we consider black-box setttings, i.e. the underlying model is unknown. Model-based approaches tend to aim at reducing the model error (i.e. the error of model parameters), which is hoped would result in the improvement of predictive error. However, in black-box settings no information about the undelying model is assumed to be available. Therefore many of the traditional active learning methods are not applicable in these settings. On the other hand, the output estimates are easily accessible. Motivated by this we aim at developing an active learning that utilizes the information contained within the output estimates.

Method The generalization error measures how well the estimated output values approximate the true output values. We note that in the calculation of the generalization error, the true output values are not affected by the addition of the new training point, while the estimates of the output values do change. Therefore, we propose to estimate the effect of a new training point on the value of the generalization error in terms of changes in the estimates of the output values.

First, let us reformulate the goal of minimizing the generalization error in terms of the changes in its value that adding a training point causes. Let us denote the generalization error when the number of training points is equal to t by G_t, the index of the next training point $\mathbf{x}_\delta$ by δ; and the generalization error after the output value y_δ is obtained by G_{t+1}. Let us express G_{t+1} as $G_{t+1} = G_t - (G_t - G_{t+1})$. The value of G_t is fixed in advance (since we are considering a sequential scenario). The value of G_{t+1} depends on the choice of δ. In order for G_{t+1} to be minimized the difference between generalization errors G_t and G_{t+1} needs to be maximized i.e.: $min_\delta G_{t+1} = G_t - max_\delta(G_t - G_{t+1})$. So the original task of minimizing the generalization error could be reformulated as maximizing the difference between the generalization errors G_t and G_{t+1} i.e.: $argmin_\delta G_{t+1} = argmax_\delta(G_t - G_{t+1})$. Let us denote $\widehat{\mathbf{y}}_t$ as the estimates of output values when the number of training samples is equal to t; and $\widehat{\mathbf{y}}_{t+1}$ as the estimates of output values after the value of y_δ was obtained and added to the training set. Let us rewrite the difference between generalization errors G_t and G_{t+1} (also referred to as $\triangle G$) in terms of a difference between $\widehat{\mathbf{y}}_t$ and $\widehat{\mathbf{y}}_{t+1}$: $\triangle G = G_t - G_{t+1} = \|\widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}\|^2 + 2\left\langle \widehat{\mathbf{y}}_{t+1} - \widehat{\mathbf{y}}_t,\, \mathbf{y} - \widehat{\mathbf{y}}_{t+1} \right\rangle$. Let us denote the first term by $T_1 = \|\widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}\|^2$, and the second term by $T_2 = 2\left\langle \widehat{\mathbf{y}}_{t+1} - \widehat{\mathbf{y}}_t,\, \mathbf{y} - \widehat{\mathbf{y}}_{t+1} \right\rangle$. Note that this decomposition is different from the standard bias-variance decomposition.

The value of $\triangle G$ could not be calculated directly since the true output values $\mathbf{y}$ are not accessible. Estimating the value of term T_2 relies on the estimate of all of the values in $\mathbf{y}$. In the current settings, the number of training samples is small, so the estimate of $\mathbf{y}$ is likely to be unreliable. However, estimating the value of term T_1 requires only the estimate of a single value y^*_δ, so the estimate of T_1 is less likely to be error-prone than the estimate of T_2.

Let us investigate if T_1 alone is a good predictor of $\triangle G$. Let us consider three possible cases of the location of $\widehat{y}_{t+1}$ (an element of $\widehat{\mathbf{y}}_{t+1}$) in relation to the corresponding elements $\widehat{y}_t$ and y, as illustrated in Figure 2. In case (b), adding a training point improves the estimate of the true output value. In this case, maximizing T_1 also maximizes $\triangle G$. In case (a), adding a training point deteriorates the estimate of the true output value. In case (c), adding a training point causes the estimate to overshoot the true output value. In both cases (a) and (c) maximizing T_1 does not maximize $\triangle G$. In [1], we have empirically shown that case (b) is much more frequent than cases (a) and (c). Even when cases (a) and (c) do occur, the probability of the output estimate significantly deteriorating is low. T_1 appeas to be a promossing AL criterion (Figure 3).

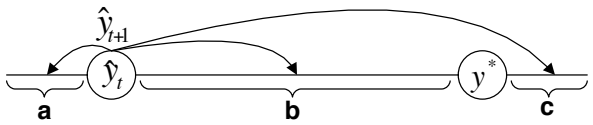

Figure 2: $\widehat{y}$ after the training point δ is added to the training set (making the number of training points equal to $t+1$).

References

[1] N. Rubens, R. Tomioka, and M. Sugiyama. Output divergence criterion for active learning in collaborative settings. *IPSJ Transactions on Mathematical Modeling and its Applications*, 2009.

[2] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

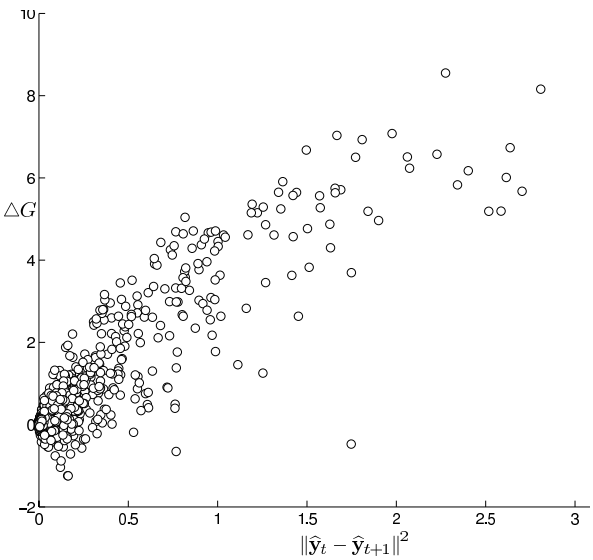

Figure 3: Emperical Evaluation. Most importantly, high values of $\|\widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}\|^2$ should correspond to high values of $\triangle G$, since those are the points that will be chosen.