論文 / 著書情報
Article / Book Information

題目(和文)	協調能動学習
Title(English)	Collaborative active learning
著者(和文)	NeilRouben
Author(English)	Rubens Neil
出典(和文)	学位:博士(学術), 学位授与機関:東京工業大学, 報告番号:甲第7719号, 授与年月日:2009年3月26日, 学位の種別:課程博士, 審査員:杉山 将
Citation(English)	Degree:Doctor (Academic), Conferring organization: Tokyo Institute of Technology, Report number:甲第7719号, Conferred date:2009/3/26, Degree Type:Course doctor, Examiner:
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Collaborative Active Learning

Neil Rubens

Supervisor: Masashi Sugiyama

Department of Computer Science
Graduate School of Information Science and Engineering
Tokyo Institute of Technology
Japan

December 2008

Abstract

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. The goal of this dissertation is to ensure that the acquired training data is useful.

The training data (needed by the supervised learning) may not be available in advance. A common way of acquiring the training data is by specifying the inputs and obtaining the corresponding outputs from the target function. However, obtaining training output values often incurs a *cost* (in terms of money, effort, time, availability, etc.). In order to ensure that obtained training data is useful, training input points (for which the output values will be obtained) are selected as to maximize the accuracy of the learned function (referred to as *active learning* (AL)). The task of AL is often performed in isolation, i.e. the function's *model* (e.g., the type and number of basis functions) is selected and then the training data is obtained for it. However, we are often not sure whether a given model would allow us to approximate the target function well and therefore may also consider *other* candidate models. In addition, the training data from *other* target functions may also be available at no cost (referred to as *collaborative settings*). Therefore we investigate how active learning could be performed not in isolation, but in *collaboration* with multiple candidate models (Active Learning with Model Selection) and data from multiple target functions (Active Learning in Collaborative Settings).

Active Learning with Model Selection

The task of selecting the function's model (e.g., the type and number of basis functions) shares the same goal as AL – improving the accuracy of the learned model. If AL and *model selection* (MS) are performed at the same

time, the accuracy of the learned function could be further improved. We propose a method that allows us to combine AL with MS and as a result to further improve the effectiveness of AL.

AL with MS can not be directly solved by simply combining standard AL methods and MS methods in a *batch* manner due to the AL/MS dilemma: In order to select training input points by an existing AL method, a model must be fixed (i.e., MS has been performed). On the other hand, in order to select the model by a standard MS method, the training input points must be fixed and corresponding training output values must be gathered (i.e., AL has been performed). Hence, we propose to perform AL for all candidate models. This allows all the models to contribute to the optimization of the training data and thus we can hedge the risk of overfitting to a single (possibly inferior) model. The proposed method compares favorably with an alternative approach of iteratively performing active learning and model selection in a sequential manner.

Active Learning in Collaborative Settings

Traditional active learning methods can be applied but are not necessarily well suited to the collaborative settings. These methods tend to select the training points as to improve the model's parameters. However, improved parameters may not necessarily translate into improved accuracy, since in collaborative settings most of the data points are missing, and the number of training data points is much smaller than the number of model's parameters.

We proposed a new active learning method that is aimed at *directly* improving the accuracy of the output value estimation, by analyzing the effect of the new training points on the estimates of the output values. We have applied the proposed method to a practical task of learning the user's movie preferences and achieved a favorable performance.

Conclusion

We have shown that by not treating active learning as an isolated problem, it is possible to further improve its effectiveness. We looked at two approaches that allow us to achieved this. First, performing AL with MS allows us to take into consideration multiple candidate models and as a result to improve the effectiveness of AL for a finally selected model. Second, by utilizing the training data from other target functions, we are able to improve the effectiveness of AL for the target function of interest.

Contents

List of Figures

List of Tables

Nomenclature

$f(\boldsymbol{x})$	:	learning target function
$\mathcal{D}$	:	domain of $f(\boldsymbol{x})$
$\boldsymbol{x}_i$	:	i-th training input point
y_i	:	i-th training output value
$\widehat{y}_i$	:	i-th estimate of the output value
ϵ_i	:	i-th noise
$(\boldsymbol{x}_i, y_i)$	:	i-th training sample
n	:	the number of training samples
$p(\boldsymbol{x})$	:	training input density
$q(\boldsymbol{x})$	:	test input density
$\varphi_i(\boldsymbol{x})$	:	i-th basis function
α_i, β_i	:	i-th parameter
b	:	the number of basis functions
G	:	generalization error
$\triangle G$	:	change of generalization error
J	:	estimate of $\triangle G$
$\boldsymbol{X}$	:	design matrix
$\boldsymbol{X}^*$	:	test matrix

Part I

Preliminaries

Chapter 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" (Babbage, 1864) summarizes well the importance of training data in the learning process. The goal of this dissertation is to ensure that the acquired training data is useful.

1.1 Active Learning

In some cases, the training data is provided beforehand. It is referred to as *passive learning*, since the training data is simply passed to the supervised learning method and a learned function is then obtained (see Figure 1.1). We are interested in the other cases, where the training data (needed by the supervised learning) is not provided in advance and therefore needs to be acquired. We can obtain training data by specifying inputs (or select inputs in cases where obtaining inputs at the arbitrary location is not possible)

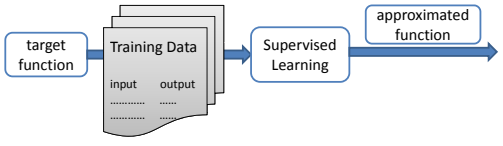


Figure 1.1: Passive Learning.

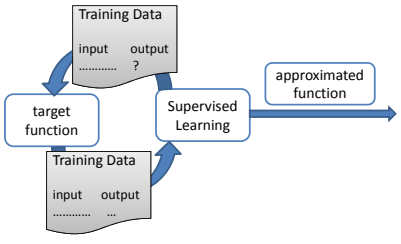
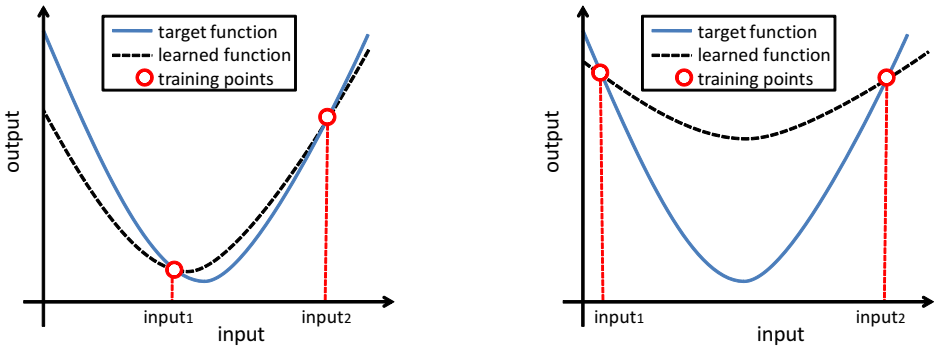


Figure 1.2: Active Learning.



(a) (b)

Figure 1.3: Dependence of accuracy of the learned function on the choice of training input points. Inputs must be chosen by an AL algorithm *before* the output values are obtained.

and obtaining the corresponding outputs from the target function. However, obtaining output values often incurs a *cost* (in terms of money, effort, time, availability, etc.). For example, asking a user to evaluate a movie is costly in terms of the user's effort; evaluating effectiveness of a drug candidate requires clinical trial and is costly in terms of both money and time. In order to ensure that obtained training data is useful, input points (for which the output values will be obtained) are selected as to maximize the accuracy of the learned function (referred to as *active learning* (AL), see Figure 1.2). What makes the AL task challenging is that we have to predict the improvement in the accuracy of the learned function with regard to the input point *before* its output value is obtained, since once the output value is obtained it incurs a cost (see Figure 1.3).

1.1.1 Illustrative Example

In this illustrative example (see Figure 1.4) we attempt to provide a little bit of intuition behind active learning. Let us describe its settings. The input is 2-dimensional. The output value is an integer, for visualization purposes it is mapped to a color by the following colormap (see Figure 1.4). The outputs of target function are shown for reference purpose in Figure 1.4, but in practice the outputs of target function are not known. We want to select a training point that may allow us to accurately estimate the output values of the test points. We are given a choice of the training input points: a, b, c, d. The predicted output values for each of the training point choices are shown in Figure 1.5. If we add the training point (a), it does not affect

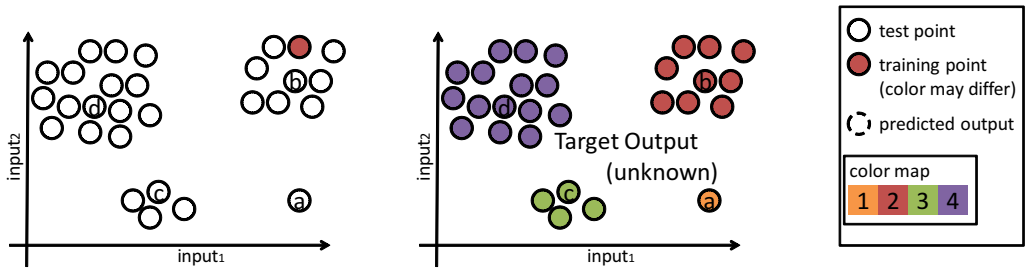


Figure 1.4: Active learning: illustrative example

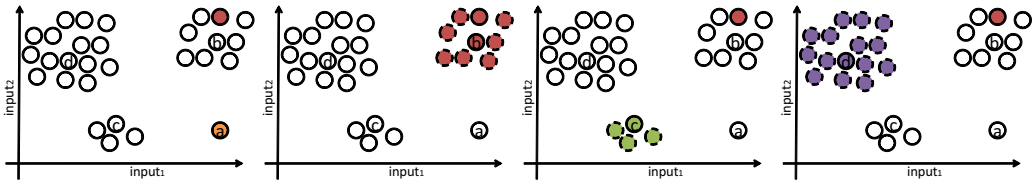
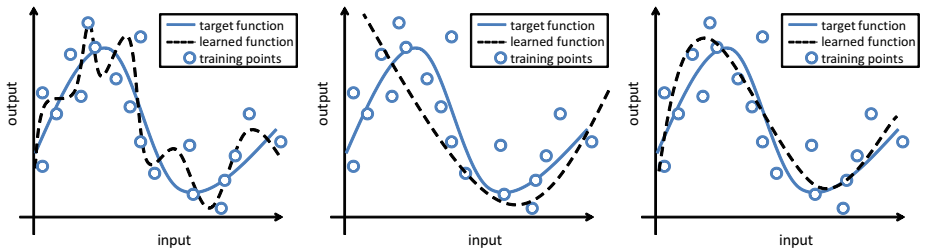


Figure 1.5: Active learning: predicted output values

our predictions (no new predictions could be made). If we add the training point (b), we can predict the values for the points in the same area. However these predictions could have been made before the training point (b) was added since we already had a training point in that area. If the training point (c) is added, we are able to make new predictions but only for 3 test points. If we add the training point (d), we are able to make predictions for a large number of test points that are in the same area. In this example, selecting the point (d) and obtaining its output values (i.e. adding it to the training set) allows to improve the accuracy of predictions the most. From this example we may define a heuristic AL policy – obtain training samples that are in the same area with many test points, for which the output values could not be currently predicted.

1.1.2 Objectives

The above example makes active learning seem like a simple problem. However, consider that now we are trying to learn the user's preferences for 10,000 movies, where the number of inputs characterizing the movie is large (e.g. genre, directors, actors, reviews of critics, users comments, etc.). The function used to learn the user's preferences is based on a *model* (e.g., the type and number of basis functions), but we are not sure which model is the appropriate one (Goldberg et al., 2001). However, the user is willing to rate only 10 movies. Selecting movies to be rated which may allow to accurately



(a) Too Complex (b) Too Simple (c) Appropriate

Figure 1.6: Dependence of accuracy of the learned function on the model.

learn the user's preferences for the rest of 9,990 movies becomes a task that is far from trivial.

The problems and their settings are becoming increasingly complex. Simply applying traditional AL methods may not allow to address some of the complicated problems that need to be solved. The task of AL is often performed in isolation (Fedorov, 1972; MacKay, 1992b; Cohn et al., 1996; Fukumizu, 2000; Wiens, 2000; Kanamori & Shimodaira, 2003; Sugiyama, 2006), i.e. the function's model is selected and then the training data is obtained for it. However, we are often not sure whether a given model would allow us to approximate the target function well and therefore may also consider *other* candidate models. In addition, the training data from *other* target functions may also be available at no cost (referred to as *collaborative settings*). Therefore we investigate how active learning could be performed not in isolation, but in *collaboration* with:

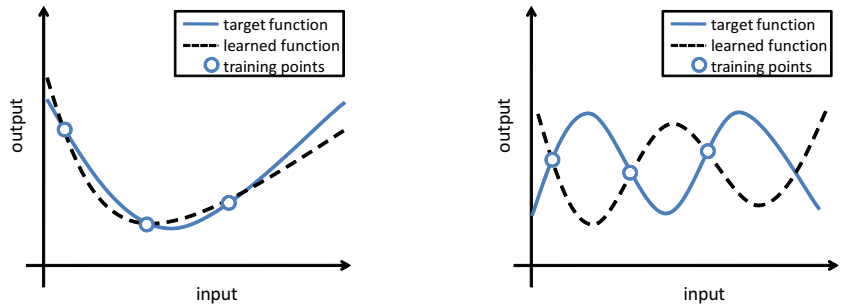
- Multiple candidate models (Active Learning with Model Selection).
- Data from multiple target functions (Active Learning in Collaborative Settings).

In the following sections we provide an overview of aspects of performing the task of active learning in a collaborative manner.

1.2 Active Learning with Model Selection

In this section, we look at how performing active learning with regard to model selection may allow to improve the accuracy of the learned function.

A concept of model has many different meanings. We refer to *model* as a set of functions with some common characteristic (e.g. function's complexity). The characteristics of the functions that may differ are often referred



Training input points that are good for learning one model, are not necessary good for the other.

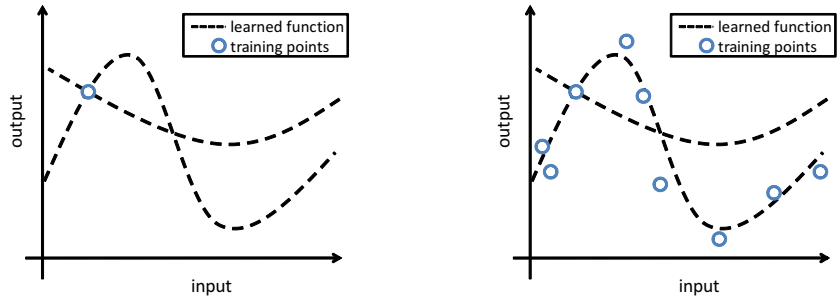
Figure 1.7: Dependence of active learning on model selection.

to as parameters (e.g. polynomial coefficients). Then given a model and training data the task of supervised learning is to find parameters that may allow to accurately approximate the target function.

Many of the works address the issue of active learning with respect to a single model (Fedorov, 1972; MacKay, 1992b; Cohn et al., 1996; Fukumizu, 2000; Wiens, 2000; Kanamori & Shimodaira, 2003; Sugiyama, 2006). However, we are often not sure whether a given model would allow us to approximate the target function well (see Figure 1.6). If the model is too simple in comparison with the target function, then the learned function may not be capable of approximating the target function (under-fit). On the other hand, if the model is too complex it may start trying to approximate irrelevant information (e.g. noise that may be contained in the output values) which will cause for the learned function to over-fit the target function. Therefore we may choose to have a number of candidate models. Model selection (MS) may allow us to select a model of appropriate complexity (Akaike, 1974; Rissanen, 1978; Schwarz, 1978; Craven & Wahba, 1979; Shimodaira, 2000; Sugiyama & Müller, 2005). So ideally, we would like to choose the model of appropriate complexity by a MS method and to choose the most useful training data by an AL method. However simply combining active learning with model selection may not be possible due to the following dilemma:

- To select training input points by a standard AL method, a model must be fixed (i.e., MS has been performed, see Figure 1.7).
- To select the model by a standard MS method, the training input points must be fixed and corresponding training output values must be gathered (i.e., AL has been performed, see Figure 1.8).

We refer to the above dilemma as AL/MS dilemma.



Unable to determine which model is more appropriate (MS), until training points have been obtained (AL).

Figure 1.8: Dependence of model selection on active learning.

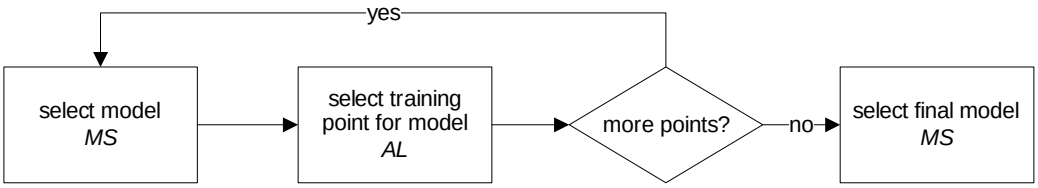


Figure 1.9: Sequential active learning.

1.2.1 Importance

Traditional active learning approaches such as sequential active learning and batch active learning are not able to cope with the AL/MS dilemma well; in the following sections we provide details.

Sequential Active Learning In the sequential active learning approach, the training points and models are selected incrementally (Figure 1.9). That is, the model is selected, then a training point is obtained for the selected model, and so on. Although this approach is intuitive, it may perform poorly due to the *model drift* – the chosen model varies through the learning process. As the number of training points increases, more complex models tend to fit data better and are therefore selected (see Figure 1.10). Since the location of optimal training input points depends on the model, the training input points chosen in the early stages could be less useful for the model selected in the end of the learning process (see Figure 1.11).

1.2.1.1 Batch Active Learning

In the batch active learning approach, all the training input points are selected for an initially chosen model (see Figure 1.12). Since the target model

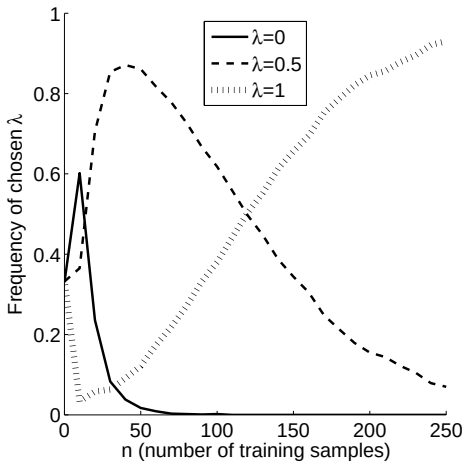


Figure 1.10: Relation between the model selection (y-axis), the number of training samples (x-axis), and the model's complexity (simpler models are indicated by smaller λ). Details are provided in Chapter 6.

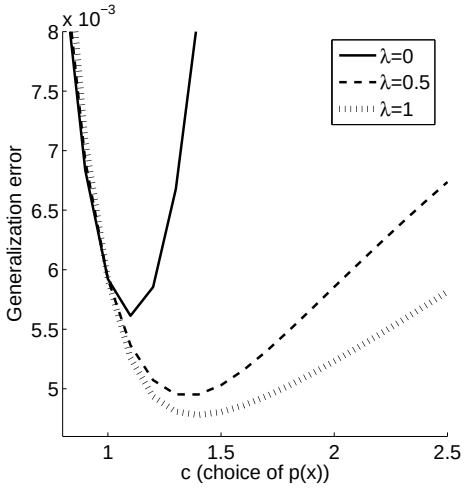


Figure 1.11: Relation between the model's accuracy (y-axis), the location of training input points (x-axis), and the model's complexity (simpler models are indicated by smaller λ). Details are provided in Chapter 6.

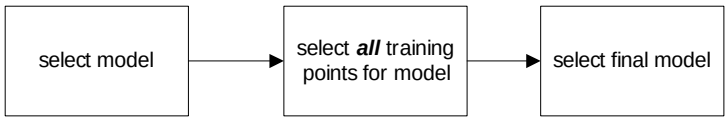


Figure 1.12: Batch active learning.

is fixed through the learning process, this approach does not suffer from the model drift and it works optimally (in terms of AL) if the initially chosen model agrees with the finally chosen model. However, choosing an appropriate initial model before having any single training sample may not be possible without prior knowledge – which is usually unavailable. For this reason, it may be difficult to obtain a good performance by the batch approach.

1.2.2 Contribution

We proposed an active learning approach that selects the training input points with respect to multiple candidate models, by optimizing the training data for all model candidates (see Figure 1.13). This allows all the mod-

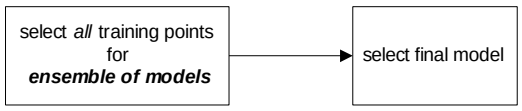


Figure 1.13: Ensemble active learning.

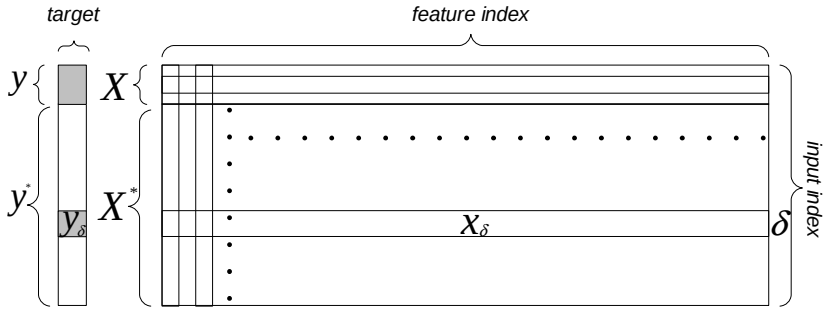


Figure 1.14: Representation of active learning in a matrix form.

els to contribute to the optimization of the training data and thus we can hedge the risk of overfitting to possibly inferior models. We have shown that performing active learning in *collaboration* with model selection allows us to further improve the effectiveness of active learning. However, ignoring the dependency between active learning and model selection by existing methods results in overfitting to a possibly inferior model for the batch AL, or due to the model drift gathering only a small portion of points for each of the candidate models, which in the end may not be well suited for any of the candidate models.

1.3 Active Learning in Collaborative Settings

Let us represent the problem of active learning in a matrix form as following (see Figure 1.14). Training input points correspond to the row vectors in $\mathbf{X}$. Test input points correspond to row vectors in $\mathbf{X}^*$. Corresponding output values are elements of vectors $\mathbf{y}$ and in $\mathbf{y}^*$. Only the output values in $\mathbf{y}$ are assumed to be known. The goal of active learning is to select a new training input point $\mathbf{x}_\delta$ from $\mathbf{X}^*$ (δ refers to an arbitrary row index of $\mathbf{X}^*$), as to maximize the accuracy of the learned function. After the input point $\mathbf{x}_\delta$ is selected, the point is added to the training set with the with corresponding output value y_δ.

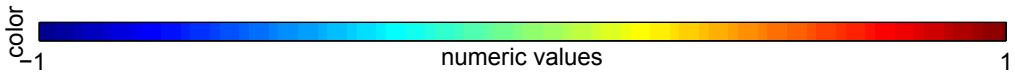


Figure 1.15: Colormap.

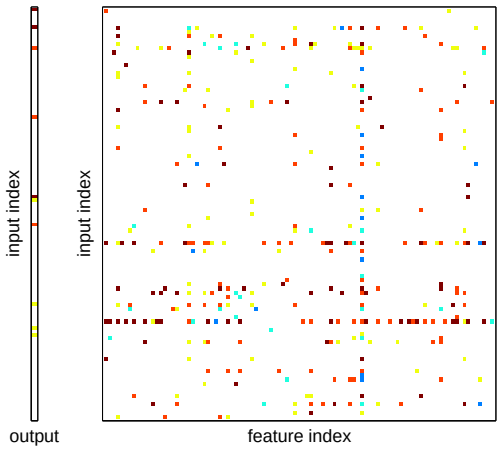


Figure 1.16: Example of collaborative settings.

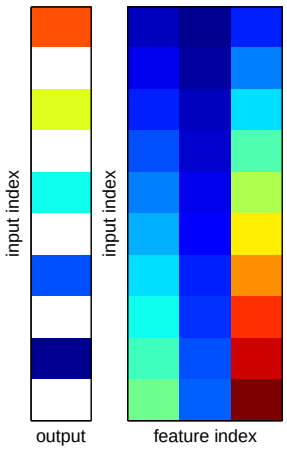


Figure 1.17: Example of traditional settings.

1.3.1 Traditional Settings

In traditional settings, we learn an approximation of the target function based on the training data consisting of inputs and the corresponding outputs from the *target* function. The number of training points is often assumed to be large enough in order to obtain an accurate approximation of the target function. We show an example of traditional settings in Figure 1.17. For visualization purposes numeric data values are scaled and mapped through the color map, missing values are indicated by the white color (see Figure 1.15).

1.3.2 Collaborative Settings

In collaborative settings, while the cost is still high for obtaining the training points from the *target* function, the training points from *other target* functions are available at no cost. For example, while rating a movie may require a substantial effort from a user, the same movie could have already been rated by many other users. We want to utilize the training data of other

functions, in order to improve the accuracy of active learning for the target function of interest. We may represent this problem in a same way as for traditional settings, where matrix entries correspond to the output values of other functions (e.g. $x_{i,j}$ corresponds to the rating of a movie i by a user j). The target output values are then approximated in terms of output values of other functions. Collaborative settings could be considered a special case of traditional settings with the following characteristics (see Figure 1.16):

- matrix is sparse (most of the matrix entries are missing)
- the number of inputs and the number features (dimensions of the matrix) are large
- the number of training points is much smaller than the number of inputs and features

1.3.3 Importance

The number of domains, where the data from multiple functions is available is increasing. Let us provide some of the examples of collaborative data:

- Recommender Systems

  MovieLens dataset: 1×10^6ratings of $6,040$ users for $3,900$ movies, 95.4% of the matrix values are missing (GroupLens, University of Minnesota, 2005).

  NetFlix dataset: 100×10^6 ratings from $480,000$ users for $17,000$ movies, 98.8% of the matrix entries are missing (NetFlix Inc., 2007).

- Information Retrieval

  Million Query Track TREC dataset: 25×10^6 web pages, $10,000$ queries, for each query $1,000$ web pages are judged, 99.996% of matrix entries are missing (Allan, 2007).

- Bioinformatics

  Beilstein database: 9.8×10^6 substances, and 10.3×10^6 chemical reactions between substances (L. Domokos & Wittig, 1986).

  Comparative Toxicogenomics Database: interactions between: $59,000$ chemicals, 1.2×10^6 gene and protein sequences, $83,000$ taxonomic terms, and $6,000$ human diseases (Mattingly CJ, 2004).

The proliferation of collaborative settings is in part due to the advances in:

- Data Acquisition: It is becoming cheaper and easier to acquire data. For example, the web logs that record users activity on the Internet could be obtained at very little cost; the cost of DNA sequencing has been rapidly decreasing, etc. However the data available for each function often stays limited, which makes active learning relevant.
- Data Storage: The cost of storing data has been decreasing.
- Data Sharing: Sharing data has become easier and cheaper due to the increased bandwidth of the Internet.
- Data Processing: Advances in computer hardware and software allow for large quantities of data to be processed inexpensively.

Active learning approaches could be divided into two categories: model-based active learning and model-independent active learning. Let us briefly describe these approaches and point out their potential limitations.

Model-based Active Learning In model-based active learning training points are selected as to improve the accuracy of the parameters of a learned function(Nakamura et al., 2003; Boutilier et al., 2003; Yu et al., 2006; Sugiyama, 2006). However, due to the specific characteristics of collaborative settings, improving the function's parameters may not necessarily result in improved accuracy for the output estimates. For example, assume a model estimates output values as a weighted sum input features. Improving the weight of the feature for which all the values are missing will have no direct effect on the accuracy of the learned function. In addition, for each model the corresponding active learning method may need to be derived. This could be a difficult and time consuming task.

Model-independent Active Learning In model-independent active learning, training points are selected based on the properties of the training points (Kohrs & Merialdo, 2001; Rashid et al., 2002). For example, uncertainty of the output value is a property often used in model-independent methods. If the output value of a training point is certain, there is little need for obtaining it. However, the relation between the properties of the point and the accuracy of output estimates may also depend on other factors. Selecting training point based on the uncertainty property alone may not be effective. For example, outlier points are often characterized by high uncertainty, but may not improve the accuracy of the output estimates. In addition, model-independent AL methods are often heuristic and therefore may lack a solid justification.

a) Adding a training point influences *many* output estimates.

b) Adding a training point influences only a *few* output estimates.

Before a training point was added output estimates are denoted as $\widehat{y}_t$, after as $\widehat{y}_{t+1}$.

Figure 1.18: Intuition for the proposed method.

1.3.4 Contribution

Model-based active learning approaches attempt to improve the accuracy of output estimates by selecting training points that may allow to improve the function's parameters. Model-independent active learning approaches attempt to identify properties of the training points that may positively affect the accuracy of the output estimates. A famous statistician Vladimir Vapnik noted that: "When solving a problem of interest, do not solve a more general problem as an intermediate step. Try to get the answer that you really need but not a more general one." (Vapnik, 1998). Motivated by this, we base the proposed method on the relation between the changes in the output estimates (caused by adding a new training point) and the accuracy of the output estimates (see Figure 1.18). The intuition for our method is as following: assuming that overall the improvement in the output estimates is positive (i.e. as more training points are added, the estimates become more accurate), adding an influential point will potentially improve the accuracy of the output estimates. We provide a more solid derivation and justification in Part III. Concentrating on directly improving the accuracy of output estimates allows the proposed criterion to achieve a good performance in collaborative settings. It is a model-independent active learning approach, since in only relies on the estimates of the output values. However, it could be justified/interpreted with regards to the *model-based* methods such as residual, training set-based and transductive methods[1].

We have shown that utilizing additional data may allow us to further improve the effectiveness of active learning. However, the peculiarities of the data must be taken into account by the active learning methods in order to be effective.

(a) Ratings for these movies would provide only limited information about the user's preferences.

(b) Ratings for these movies would provide a more comprehensive information about the user's preferences.

1) Images are obtained from Amazon® and used in accordance with the 'fair use' copyright license.

2) All movies mentioned are assumed to be copyrighted by the corresponding copy holders.

3) In case of any violations, author is solely responsible for bringing the works into compliance

Figure 1.19: Differences in the 'usefulness' of the training data.

Application

We have applied the proposed active learning method to the task of learning a user's movie preferences. Let us provide a description of this application.

In order to be able to make personalized recommendations, we need to know the user's preferences for at least some of the movies. However, a user is often willing to rate only a limited number of movies. Hence, obtaining user's movie ratings could be considered costly. While the cost of obtaining a movie's rating could be the same, the usefulness of the obtained data may vary significantly. For example, ratings of three similar movies 'Star Wars' I, II, and III, may provide a very limited information about the user's preferences (see Figure 1.19(a)). On the other hand, we may learn a lot more from the ratings of movies from different genres, e.g. a science fiction movie 'Star Wars I', a romantic movie 'Pretty Woman' and an action movie 'Die Hard'.

Proposed active learning method works well for the illustrative example above (Figure 1.19). As the first movie 'Star Wars I' is selected. It is a representative movie of the science fiction genre, i.e. people that like this movie tend to favor other movies in the science fiction genre, and the other way around. The rating of 'Star Wars I' captured well the user's preferences for the genre, so the rest of the 'Star Wars' sequels were no longer presented for rating. The rest of the movies (see Figure 1.19(b)) are both representative of their genres. Since the romance genre contains more movies, the movie 'Pretty Woman' was presented first, followed by a comedy 'Knocked Up'.

We have performed a number of numerical experiments by using a popular MovieLens dataset (GroupLens, University of Minnesota, 2005). The proposed method compares favorably with traditional active learning meth-

[1] shown in linear regression settings

ods.

1.4 Organization of Dissertation

This dissertation addresses the issue of performing active learning in collaboration with model selection, and data from multiple target functions. The rest of the dissertation is organized as following. In Part II we provide details about the aspect of performing active learning with model selection. In Part III we describe the details about the aspect of performing active learning in collaborative settings. Finally in Part IV we discuss the importance of not treating active learning as an independent task, and discus a number of issues to be investigated in the future works.

Part II

Active Learning with Model Selection

Chapter 2

Introduction

Overview Optimally designing the location of training input points (active learning) and choosing the best model (model selection) are two important components of supervised learning and have been studied extensively. However, these two issues seem to have been investigated separately as two independent problems. If training input points and models are simultaneously optimized, the generalization performance would be further improved. We propose a new approach called *ensemble active learning* for solving the problems of active learning and model selection at the same time. We demonstrate by numerical experiments that the proposed method compares favorably with alternative approaches such as iteratively performing active learning and model selection in a sequential manner.

When we are allowed to choose the location of training input points in supervised learning, we want to optimize them so that the generalization error is minimized. This problem is called *active learning* (AL) and has been studied extensively (Fedorov, 1972; MacKay, 1992b; Cohn et al., 1996; Fukumizu, 2000; Wiens, 2000; Kanamori & Shimodaira, 2003; Sugiyama, 2006). On the other hand, *model selection* (MS) is another important issue in supervised learning: a model (e.g., the type and number of basis functions, regularization parameter, etc.) is optimized so that the generalization error is minimized (Akaike, 1974; Rissanen, 1978; Schwarz, 1978; Craven & Wahba, 1979; Shimodaira, 2000; Sugiyama & Müller, 2005).

Although AL and MS share a common goal of minimizing the generalization error, they seem to have been studied separately as two independent

problems. If AL and MS are performed at the same time, the generalization performance would be further improved. We call the problem of simultaneously optimizing training input points and models *active learning with model selection* (ALMS). However, ALMS can not be directly solved by simply combining standard AL methods and MS methods in a batch manner due to the AL/MS dilemma: In order to select training input points by an existing AL method, a model must be fixed (i.e., MS has been performed). On the other hand, in order to select the model by a standard MS method, the training input points must be fixed and corresponding training output values must be gathered (i.e., AL has been performed).

A standard approach to coping with the AL/MS dilemma is the *sequential approach*, i.e., iteratively performing AL and MS in an online manner (MacKay, 1992a). Although this approach is intuitive, it can perform poorly due to the *model drift*—the chosen model varies through the online learning process. Since the location of optimal training input points depends on the model, the training input points chosen in early stages could be less useful for the model selected in the end of the learning process.

An alternative approach to solving the ALMS problems is to choose all the training input points for an initially chosen model, which we refer to as the *batch approach*. Since the target model is fixed through the learning process, this approach does not suffer from the model drift and it works optimally (in terms of AL) if the initially chosen model agrees with the finally chosen model. However, choosing an appropriate initial model *before* having any single training sample may not be possible without prior knowledge—which is usually unavailable. For this reason, it may be difficult to obtain a good performance by the batch approach.

The weakness of the batch approaches actually lies in the fact that the training input points chosen by an AL method are *overfitted* to the initially chosen model; the training input points optimized for the initial model could be poor if a different model is chosen later. To alleviate this problem, we propose a new approach called *ensemble active learning* (EAL). The main idea of EAL is to perform AL not for a *single* model, but for *all* models at hand. This allows us to hedge the risk of overfitting to a single (possibly inferior) model. We experimentally show the EAL method significantly outperforms other approaches.

Chapter 3

Problem Formulation

In this section, we formulate the supervised learning problem.

3.1 Linear Regression

Let us consider the regression problem of learning a real-valued function $f(\boldsymbol{x})$ defined on $\mathcal{D}(\subset \mathbb{R}^d)$ from training samples

$$\{(\boldsymbol{x}_i, y_i) \mid y_i = f(\boldsymbol{x}_i) + \epsilon_i\}_{i=1}^n, \tag{3.1}$$

where d is the dimension of the input vector $\boldsymbol{x}$, n is the number of training samples, and $\{\epsilon_i\}_{i=1}^n$ are independent and identically distributed (*i.i.d.*) noise with mean zero and unknown variance σ^2 (see Figure 3.1). We draw the training input points $\{\boldsymbol{x}_i\}_{i=1}^n$ from a distribution with density $p(\boldsymbol{x})$, which we would like to optimize by an AL method.

We employ the following linear regression model for learning:

$$\widehat{f}(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i \varphi_i(\boldsymbol{x}), \tag{3.2}$$

where $\{\varphi_i(\boldsymbol{x})\}_{i=1}^b$ are fixed linearly independent functions, $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_b)^\top$ are parameters to be learned, and $^\top$ denotes the transpose of a vector/matrix. We define the generalization error G of a learned function $\widehat{f}(\boldsymbol{x})$ by the expected squared error for *test* input points. We assume that the test input points are drawn independently from a distribution with density $q(\boldsymbol{x})$. Then G is expressed as

$$G = \int_{\mathcal{D}} \Big(\widehat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\Big)^2 q(\boldsymbol{x}) d\boldsymbol{x}. \tag{3.3}$$

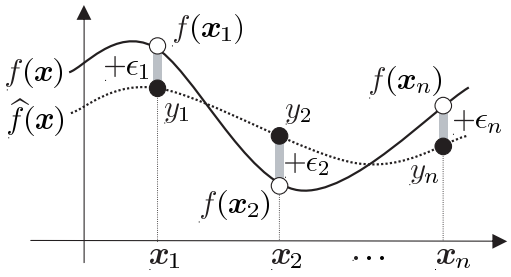


Figure 3.1: Regression problem.

As in the AL literature (Fukumizu, 2000; Wiens, 2000; Kanamori & Shimodaira, 2003; Sugiyama, 2006), we assume that $q(\boldsymbol{x})$ is known or its reasonable estimate is available.

Since we discuss the MS problem, i.e., choosing the number and type of the basis functions $\{\varphi_i(\boldsymbol{x})\}_{i=1}^b$, we can not generally assume that the model is correctly specified. Thus, the target function $f(\boldsymbol{x})$ is not necessarily of the form (3.2), but is expressed as follows (see Figure 3.2):

$$f(\boldsymbol{x}) = g(\boldsymbol{x}) + \delta r(\boldsymbol{x}), \tag{3.4}$$

where $g(\boldsymbol{x})$ is the optimal approximation to $f(\boldsymbol{x})$ within the model (3.2):

$$g(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i^* \varphi_i(\boldsymbol{x}). \tag{3.5}$$

$\boldsymbol{\alpha}^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_b^*)^\top$ is the unknown optimal parameter under G:

$$\boldsymbol{\alpha}^* = \operatorname*{argmin}_{\boldsymbol{\alpha}} G. \tag{3.6}$$

$\delta r(\boldsymbol{x})$ in Eq.(**??**) is the residual, which is orthogonal to $\{\varphi_i(\boldsymbol{x})\}_{i=1}^b$ under $q(\boldsymbol{x})$: for $i = 1, 2, \ldots, b$,

$$\int_{\mathcal{D}} r(\boldsymbol{x})\varphi_i(\boldsymbol{x})q(\boldsymbol{x})d\boldsymbol{x} = 0. \tag{3.7}$$

The function $r(\boldsymbol{x})$ governs the nature of the model error, and δ is the possible magnitude of this error. In order to separate these two factors, we further impose the following normalization condition on $r(\boldsymbol{x})$:

$$\int r^2(\boldsymbol{x})q(\boldsymbol{x})d\boldsymbol{x} = 1. \tag{3.8}$$

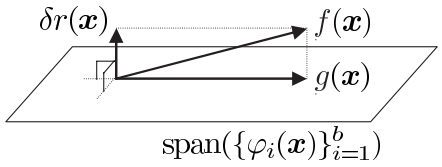


Figure 3.2: Orthogonal decomposition of $f(\boldsymbol{x})$.

Under this setting, the expected generalization error can be decomposed into the model error δ^2, bias B, and variance V:

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} G = \delta^2 + B + V, \tag{3.9}$$

where $\mathbb{E}_{\boldsymbol{\epsilon}}$ denotes the expectation over the noise $\{\epsilon_i\}_{i=1}^n$ and

$$B = \int_{\mathcal{D}} \left(\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{f}(\boldsymbol{x}) - g(\boldsymbol{x}) \right)^2 q(\boldsymbol{x}) d\boldsymbol{x}, \tag{3.10}$$

$$V = \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \int_{\mathcal{D}} \left(\widehat{f}(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{f}(\boldsymbol{x}) \right)^2 q(\boldsymbol{x}) d\boldsymbol{x}. \tag{3.11}$$

3.2 Parameter Learning

A standard parameter learning method in the regression scenario would be ordinary least squares (OLS). OLS is asymptotically unbiased and efficient in standard cases. However, the AL scenario is a typical case of *covariate shift* (Shimodaira, 2000)—the training input distribution is different from the test input distribution: $p(\boldsymbol{x}) \neq q(\boldsymbol{x})$. Under covariate shift, OLS is not asymptotically unbiased anymore; instead, the following *adaptive importance-weighted least-squares* (AIWLS)[1] is shown to work well (Shimodaira, 2000):

$$\min_{\boldsymbol{\alpha}} \left[\sum_{i=1}^n \left(\frac{q(\boldsymbol{x}_i)}{p(\boldsymbol{x}_i)} \right)^{\lambda} \left(\widehat{f}(\boldsymbol{x}_i) - y_i \right)^2 \right], \tag{3.12}$$

where λ ($0 \leq \lambda \leq 1$) is a tuning parameter. λ is called the *flattening parameter* since it flattens the importance weights.

AIWLS has the following property: When $\lambda = 0$, AIWLS is reduced to OLS; thus it is biased but has smaller variance. When $\lambda = 1$, AIWLS is asymptotically unbiased but has a larger variance. In practice, an intermediate λ often produces good results since it can control the tradeoff between

[1] We may further add a regularizer to AIWLS.

the bias and variance. Therefore, we have to choose λ appropriately by an MS method for improving generalization performance (Shimodaira, 2000; Sugiyama & Müller, 2005; Sugiyama et al., 2007).

Let $\boldsymbol{X}$ be the *design matrix*, i.e., $\boldsymbol{X}$ is the $n \times b$ matrix with the (i,j)-th element

$$X_{i,j} = \varphi_j(\boldsymbol{x}_i). \tag{3.13}$$

Let $\boldsymbol{D}$ be the diagonal matrix with the i-th diagonal element being the importance weight of the i-th sample:

$$D_{i,i} = \frac{q(\boldsymbol{x}_i)}{p(\boldsymbol{x}_i)}. \tag{3.14}$$

Then the AIWLS estimator $\widehat{\boldsymbol{\alpha}}$ is analytically given by

$$\widehat{\boldsymbol{\alpha}} = \boldsymbol{L}\boldsymbol{y}, \tag{3.15}$$

where

$$\boldsymbol{L} = (\boldsymbol{X}^\top \boldsymbol{D}^\lambda \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{D}^\lambda, \tag{3.16}$$

$$\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^\top. \tag{3.17}$$

3.3 Active Learning (AL)

AL is the problem of optimizing the training input density $p(\boldsymbol{x})$ so that the generalization error is minimized[2]:

$$\min_p G(p). \tag{3.18}$$

In order to perform AL, the inaccessible generalization error G has to be estimated. Note that in batch AL, we have to estimate G *before* training samples is observed (cf. MS in Chapter 3.4). Since our model (3.2) could be misspecified (see Chapter 3.1), we can not reliably use the traditional OLS-based AL method (Fedorov, 1972; Cohn et al., 1996; Fukumizu, 2000).

Recently, a novel generalization error estimator $\widehat{G}^{(AL)}$ for AL has been developed, which is shown to be reliable even if the model (3.2) is not correctly specified (Sugiyama, 2006):

$$\widehat{G}^{(AL)} = \mathrm{tr}(\boldsymbol{U}\boldsymbol{L}\boldsymbol{L}^\top), \tag{3.19}$$

[2]Precisely, this is the *batch* active learning problem where all training input points are designed at once in the beginning.

where $\boldsymbol{U}$ is the b-dimensional square matrix with the (i,j)-th element

$$U_{i,j} = \int_{\mathcal{D}} \varphi_i(\boldsymbol{x})\varphi_j(\boldsymbol{x})q(\boldsymbol{x})d\boldsymbol{x}. \tag{3.20}$$

Note that $\sigma^2\widehat{G}^{(AL)}$ corresponds to the variance term V (see Eq.(3.11)). When the model is approximately correct (i.e., the model error δ asymptotically vanishes) and $\lambda \to 1$ (i.e., $\widehat{\boldsymbol{\alpha}}$ is asymptotically unbiased), $\widehat{G}^{(AL)}$ is shown to be a consistent estimator of the expected generalization error (up to the model error δ^2):

$$\sigma^2\widehat{G}^{(AL)} = \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} G - \delta^2 + o_p(n^{-1}), \tag{3.21}$$

where $o_p(\cdot)$ denotes the asymptotic order in probability. A sketch of its proof is reviewed in Appendix A.1. This justifies the use of Eq.(3.19) as an AL criterion.

3.4 Model Selection (MS)

MS is the problem of optimizing a model M so that the generalization error is minimized:

$$\min_{M} G(M). \tag{3.22}$$

In the current setting, the model M refers to the number and type of the basis functions $\{\varphi_i(\boldsymbol{x})\}_{i=1}^b$; the flattening parameter λ in Eq.(3.12) is also included in the model. In order to perform MS, the inaccessible generalization error G has to be estimated. Note that in the MS problem, it is usually assumed that the training samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ have already been observed (Akaike, 1974; Rissanen, 1978; Schwarz, 1978; Craven & Wahba, 1979; Shimodaira, 2000; Sugiyama & Müller, 2005). Thus $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ are used for estimating the generalization error.

The current situation contains a shift in input distributions, i.e. $p(\boldsymbol{x}) \neq q(\boldsymbol{x})$. Therefore, standard MS methods such as Akaike's information criterion (Akaike, 1974) and cross-validation (Craven & Wahba, 1979) have strong bias and thus are not reliable (Zadrozny, 2004; Sugiyama & Müller, 2005; Sugiyama et al., 2007). Recently, a novel generalization error estimator for MS has been proposed, which possesses proper unbiasedness even under covariate shift (Sugiyama & Müller, 2005):

$$\widehat{G}^{(MS)} = \langle \boldsymbol{U}\boldsymbol{L}\boldsymbol{y}, \boldsymbol{L}\boldsymbol{y}\rangle - 2\langle \boldsymbol{U}\boldsymbol{L}\boldsymbol{y}, \boldsymbol{L}_1\boldsymbol{y}\rangle + 2\widehat{\sigma^2}\mathrm{tr}(\boldsymbol{U}\boldsymbol{L}\boldsymbol{L}_1^\top), \tag{3.23}$$

where

$$\widehat{\sigma^2} = \frac{\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{L}_0\boldsymbol{y}\|^2}{n-b}. \tag{3.24}$$

$\boldsymbol{L}_0$ and $\boldsymbol{L}_1$ denote $\boldsymbol{L}$ computed with $\lambda = 0$ and $\lambda = 1$, respectively. $\widehat{G}^{(MS)}$ is shown to satisfy

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{G}^{(MS)} = \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} G - C + \mathcal{O}_p(\delta n^{-\frac{1}{2}}), \tag{3.25}$$

where

$$C = \int_{\mathcal{D}} f(\boldsymbol{x})^2 q(\boldsymbol{x}) d\boldsymbol{x}. \tag{3.26}$$

A sketch of its proof is reviewed in Appendix A.2. This means that, $\widehat{G}^{(MS)}$ is an exact unbiased estimator of the expected generalization error (up to the constant C) if the model is correctly specified (i.e., the model error δ is zero); for misspecified models, it is an asymptotic unbiased estimator in general, where the asymptotic error is proportional to the model error δ. This justifies the use of Eq.(3.23) as an MS criterion.

3.5 Active Learning with Model Selection (ALMS)

The problems of AL and MS share the common goal—minimizing the generalization error (see Eqs.(3.18) and (3.22)). However, they have been studied separately as two independent problems so far. If AL and MS are performed at the same time, the generalization performance would be further improved. We call the problem of simultaneously optimizing training input points and models *active learning with model selection* (ALMS):

$$\min_{p,M} G(p, M). \tag{3.27}$$

This is the problem we would like to solve.

Chapter 4

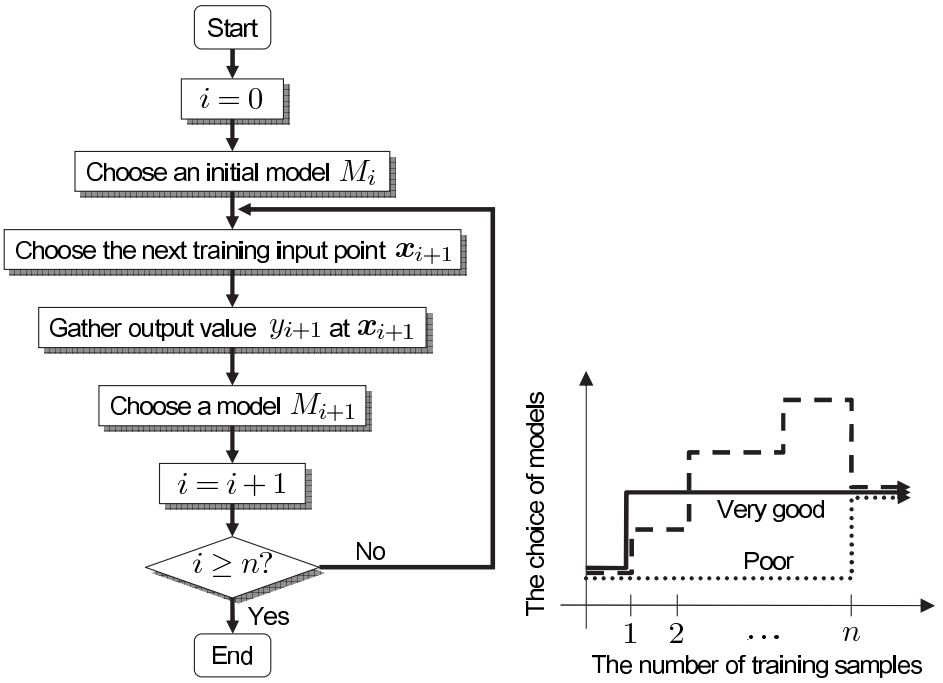
Related Work

In this section, we discuss strengths and limitations of existing approaches to solving the ALMS problem (3.27).

4.1 Direct Approach and the AL/MS Dilemma

A naive and direct solution to the ALMS problem would be to simultaneously optimize $p(\boldsymbol{x})$ and M. However, this direct approach may not be possible by simply combining existing AL methods and MS methods in a batch manner due to the *AL/MS dilemma*: when selecting the training input density $p(\boldsymbol{x})$ with existing AL methods, the model M must have been fixed (Fedorov, 1972; MacKay, 1992b; Cohn et al., 1996; Fukumizu, 2000; Wiens, 2000; Kanamori & Shimodaira, 2003; Sugiyama, 2006). On the other hand, when choosing the model M with existing MS methods, the training input points $\{\boldsymbol{x}_i\}_{i=1}^n$ (or the training input density $p(\boldsymbol{x})$) must have been fixed and the corresponding training output values $\{y_i\}_{i=1}^n$ must have been gathered (Akaike, 1974; Rissanen, 1978; Schwarz, 1978; Craven & Wahba, 1979; Shimodaira, 2000; Sugiyama & Müller, 2005). For example, the AL criterion (3.19) can not be computed without fixing the model M and the MS criterion (3.23) can not be computed without fixing the training input density $p(\boldsymbol{x})$.

If training input points that are optimal for all model candidates exist, it is possible to perform AL and MS at the same time without regard to the AL/MS dilemma: choose the training input points $\{\boldsymbol{x}_i\}_{i=1}^n$ for some model M by a standard AL method (e.g., Eq.(3.19)), gather corresponding output values $\{y_i\}_{i=1}^n$, and perform MS using a standard method (e.g., Eq.(3.23)). It is shown that such common optimal training input points exist for a class of correctly specified trigonometric polynomial regression models (Sugiyama & Ogawa, 2003). However, the common optimal training input points may not



(a) Diagram (b) Transition of chosen models

Figure 4.1: Sequential approach.

exist in general and thus the range of application of this approach is limited.

4.2 Sequential Approach

A standard approach to coping with the above AL/MS dilemma for arbitrary models would be the *sequential approach* (MacKay, 1992a), i.e., in an iterative manner, a model is chosen by an MS method and the next input point (or a small portion) is optimized for the chosen model by an AL method (see Figure 4.1(a)).

In the sequential approach, the chosen model M_i varies through the online learning process (see the dashed line in Figure 4.1(b)). We refer to this phenomenon as the *model drift*. The model drift phenomenon could be a weakness of the sequential approach since the location of optimal training input points depends on the target model in AL; thus a good training input point for one model could be poor for another model (see Chapter 6.1 for illustrative examples). Depending on the transition of the chosen models, the sequential approach can work very well. For example, when the transition of the model is the solid line in Figure 4.1(b), most of the training input points

are chosen for the finally selected model M_n and the sequential approach has an excellent performance. However, when the transition of the model is the dotted line in Figure 4.1(b), the performance becomes poor since most of the training input points are chosen for other models. Note that we can *not* control the transition of the model properly since we do not know a priori which model will be chosen in the end. Therefore, the performance of the sequential approach is unstable which could be critical in some application domains such as medical or financial data analysis.

Another issue that needs to be taken into account in the sequential approach is that the training input points are not i.i.d. in general—the choice of the $(i+1)$-th training input point $\boldsymbol{x}_{i+1}$ depends on the previously gathered samples $\{(\boldsymbol{x}_j, y_j)\}_{j=1}^{i}$. Since standard AL and MS methods require the i.i.d. assumption for establishing their statistical properties such as consistency or unbiasedness, they may not be directly employed in the sequential approach (Fedorov, 1972; MacKay, 1992b; Cohn et al., 1996; Fukumizu, 2000; Akaike, 1974; Rissanen, 1978; Schwarz, 1978; Craven & Wahba, 1979; Shimodaira, 2000; Bach, 2007). The AL criterion (3.19) and MS criterion (3.23) also suffer from the violation of the i.i.d. condition, and loose their consistency and unbiasedness. However, this problem can be settled by slightly modifying the criteria, which is an advantage of the AL criterion (3.19) and MS criterion (3.23): Suppose we draw u input points from $p^{(i)}(\boldsymbol{x})$ in each iteration (let $n = uv$, where v is the number of iterations). If u tends to be infinity, simply redefining the diagonal matrix $\boldsymbol{D}$ as follows makes $\widehat{G}^{(AL)}$ and $\widehat{G}^{(MS)}$ still consistent and asymptotically unbiased:

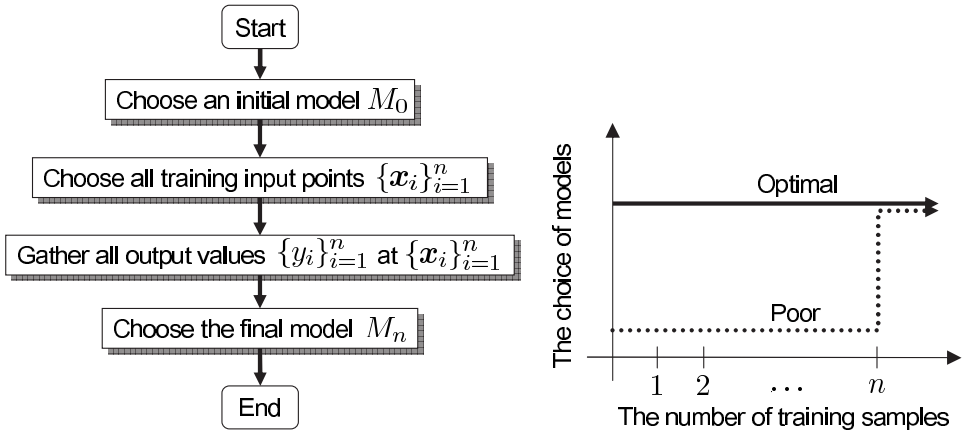
$$D_{k,k} = \frac{q(\boldsymbol{x}_k)}{p^{(i)}(\boldsymbol{x}_k)}, \tag{4.1}$$

where $k = (i-1)u + j$, $i = 1, 2, \ldots, v$, and $j = 1, 2, \ldots, u$.

4.3 Batch Approach

An alternative approach to ALMS is to choose all the training input points for an initially chosen model M_0. We refer to this approach as the *batch approach* (see Figure 4.2(a)). Due to the batch nature, this approach does not suffer from the model drift (cf. Figure 4.1(b)); the batch approach can be optimal in terms of AL if an initially chosen model M_0 agrees with the finally chosen model M_n (see the solid line in Figure 4.2(b)).

In order to choose the initial model M_0, we may need a generalization error estimator that can be computed before observing training samples—for example, the generalization error estimator (3.19). However, this does not



(a) Diagram (b) Transition of chosen models

Figure 4.2: Batch approach.

work well since Eq.(3.19) only evaluates the variance of the estimator (see Eq.(3.11)); thus using Eq.(3.19) for choosing the initial model M_0 simply results in always selecting the simplest model from the candidates. Note that this problem is not specific to the generalization error estimator (3.19), but is common to most generalization error estimators since it is generally not possible to estimate the bias of the estimator (see Eq.(3.10)) before observing training samples. Therefore, in practice, we may have to choose the initial model M_0 *randomly*. If we have some prior preference of models, $P(M)$, we may draw the initial model according to it; otherwise we just have to choose the initial model M_0 randomly from the uniform distribution.

Due to the randomness of the initial model choice, the performance of the batch approach may be unstable (see the dashed line in Figure 4.2(b)).

Chapter 5

Proposed Approach: Ensemble Active Learning (EAL)

In the previous section, we pointed out potential limitations of existing approaches. In this section, we propose a new ALMS method that can cope with the above limitations.

The weakness of the batch approach lies in the fact that the training input points chosen by an AL method are *overfitted* to the initially chosen model—the training input points optimized for the initial model could be poor if a different model is chosen later.

We may reduce the risk of overfitting by not optimizing the training input distribution *specifically* for a single model, but by optimizing it for *all* model candidates (see Figure 5.1). This allows all the models to contribute to the optimization of the training input distribution and thus we can hedge the risk of overfitting to a single (possibly inferior) model. Since this approach could be viewed as applying a popular idea of *ensemble learning* to the problem of AL, we refer to the proposed approach as *ensemble active learning* (EAL).

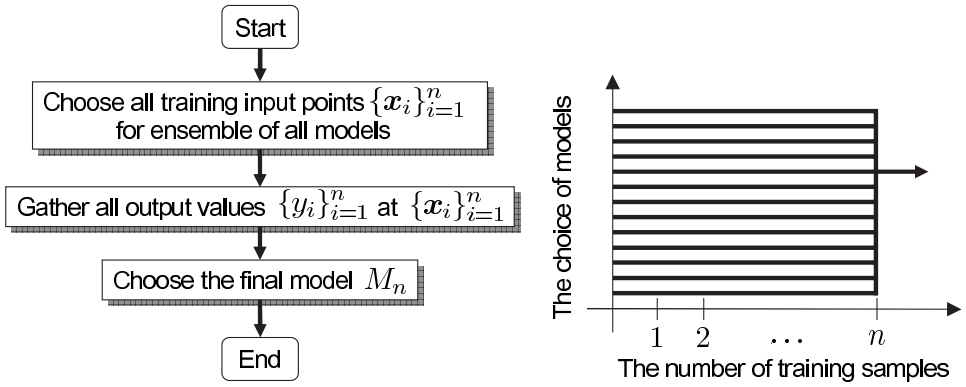
This idea could be realized by determining the training input density $p(\boldsymbol{x})$ so that the *expected* generalization error over *all* model candidates is minimized:

$$\min_{p} \widehat{G}^{(EAL)}(p), \tag{5.1}$$

where

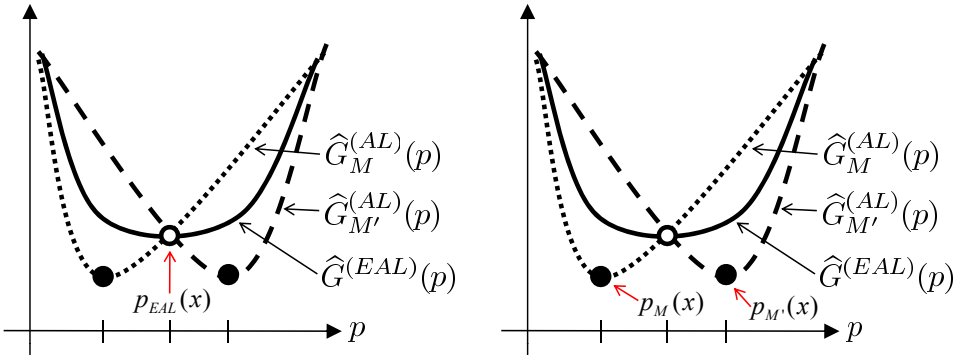
$$\widehat{G}^{(EAL)}(p) = \sum_{M} \widehat{G}_{M}^{(AL)}(p) P(M). \tag{5.2}$$

$\widehat{G}_{M}^{(AL)}$ denotes the value of $\widehat{G}^{(AL)}$ for a model M and $P(M)$ is the prior preference of the model M. If no prior information on the goodness of the models is available, the uniform prior may be simply used. In Chapter 6, we



(a) Diagram (b) Transition of chosen models

Figure 5.1: The proposed ensemble approach.



(a) $p(x)$ selected by EAL (b) $p(x)$ selected by Model EAL

Figure 5.2: Selection of training input densities by EAL and Model EAL.

experimentally show that this ensemble approach significantly outperforms the sequential and batch approaches.

5.1 Alternative approach: Model ensemble active learning

In the above approach, we select the training input density $p(\boldsymbol{x})$ based on an ensemble of AL criteria (we denote this criterion as $\widehat{G}^{(EAL)}(p)$). However, it is also possible to implement the ensemble idea in AL by allowing each model M to choose n_M training input points (see Figure 5.2), where

$$n_M \propto P(M) \tag{5.3}$$

and

$$\sum_M n_M = n. \tag{5.4}$$

More specifically, we let each model M select the training input density $p_M(\boldsymbol{x})$ by

$$p_M(\boldsymbol{x}) = \operatorname*{argmin}_p \widehat{G}_M^{(AL)}(p), \tag{5.5}$$

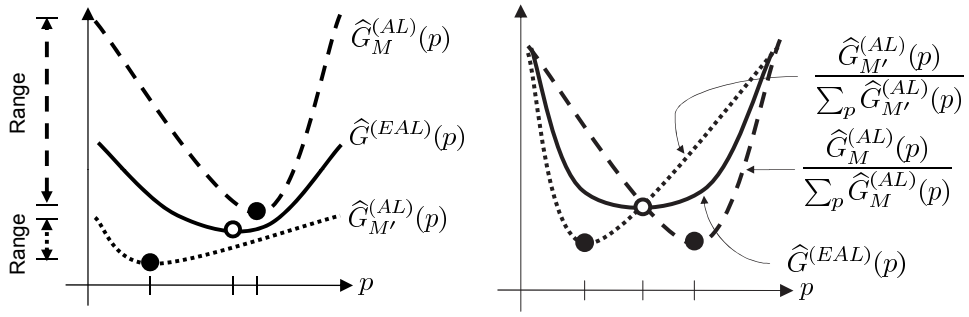
where $\widehat{G}_M^{(AL)}(p)$ is computed based on n_M training input points (see Eq.(3.19)). Then each model M draws n_M training input points from $p_M(\boldsymbol{x})$.

In active learning settings, the number of training points that we can obtain is often assumed to be small. Moreover each model M can select a only portion of the total training input points according to the chosen density $p_M(\boldsymbol{x})$. As pointed out in (Sugiyama, 2006), it may not always be possible to obtain the training input points in an arbitrary location, since training input points may not be present at the desired locations. We may first create provisional input points following the determined training input density, and then choose the input points from the pool of unlabeled samples that are closest to the provisional input points. Since the number of training points chosen by each model could be small, the actual density of the training input points $p_M(\boldsymbol{x})$ could be different from the desired one. As the result, the selected points may not be optimal for the corresponding models. Our preliminary experiments on the benchmark dataset showed that this alternative idea does not work well. For this reason, we omit further details.

5.2 Normalization

The proposed criterion Eq. (5.1) tries to choose the training input density that is good for all model candidates. However, it can still suffer from the overfitting problem. To illustrate this, let us consider a simple situation where we have two model candidates M and M' with equal prior preferences. Suppose $\widehat{G}_M^{(AL)}(p)$ has a larger *range* than $\widehat{G}_{M'}^{(AL)}(p)$. Here, the 'range' means the difference between the maximum and minimum of the AL criterion (see Figure 5.3(a)). The figure shows that the minimum of the simple unnormalized average $\widehat{G}^{(EAL)}(p)$ is dominated by $\widehat{G}_M^{(AL)}(p)$; thus the training input density $p(\boldsymbol{x})$ optimized by Eq.(5.1) still overfits to the (inferior) model M.

The above example illustrates that the possible overfitting problem of EAL comes not from the *mean* of the AL criterion over training input densities, but from the *range* of the AL criterion. Motivated by this observation, we propose normalizing the range of the AL criterion $\widehat{G}^{(AL)}(p)$, which ensures



(a) Unnormalized EAL (b) Normalized EAL

Figure 5.3: The effect of normalization in EAL.

that all the model candidates have *equal* influence on the choice of the training input density. More specifically, we define the normalized EAL approach as follows.

$$\min_p \widehat{G}^{(nEAL)}(p), \tag{5.6}$$

where

$$\widehat{G}^{(nEAL)}(p) = \sum_M \frac{\widehat{G}_M^{(AL)}(p)}{\sum_{p'} \widehat{G}_M^{(AL)}(p')} P(M). \tag{5.7}$$

By the normalization, we can equalize the influence of all models (see Figure 5.3(b)).

We can also consider a variant of the ensemble approach where the magnitude of each AL criterion $\widehat{G}^{(AL)}(p)$, is normalized:

$$\min_p \widehat{G}^{(nEAL)}(p), \tag{5.8}$$

where

$$\widehat{G}^{(nEAL)}(p) = \sum_M \frac{\widehat{G}_M^{(AL)}(p)}{\sum_{p'} \widehat{G}_M^{(AL)}(p')} P(M). \tag{5.9}$$

This variant may have an effect of reducing the influence of poor models.

However, our preliminary experiments on the benchmark datasets (6.2) showed that the use of the normalization schemes does not make a big difference. Therefore, for the numerical experiments we simply use the unnormalized approach Eq. (5.1).

Chapter 6

Numerical Experiments

In this section, we quantitatively compare the proposed EAL method with other approaches through numerical experiments.

6.1 Toy Datasets

Here we illustrate how the ensemble (Chapter 5), batch (Chapter 4.3), and sequential (Chapter 4.2) methods behave using a toy one-dimensional dataset.

Let the input dimension be $d = 1$ and the target function $f(x)$ be the following third order polynomial (see the top graph of Figure 6.1):

$$f(x) = 1 - x + x^2 + r(x), \tag{6.1}$$

where, for $\delta = 0.05$,

$$r(x) = \delta \frac{z^3 - 3z}{\sqrt{6}} \quad \text{with} \quad z = \frac{x - 0.2}{0.4}. \tag{6.2}$$

Let the test input density $q(x)$ be the Gaussian density with mean 0.2 and standard deviation 0.4, which is assumed to be known in this illustrative simulation. We choose the training input density $p(x)$ from a set of Gaussian densities with mean 0.2 and standard deviation $0.4c$, where

$$c = 0.8, 0.9, 1.0, \ldots, 2.5. \tag{6.3}$$

These density functions are illustrated in the bottom graph of Figure 6.1. We add i.i.d. Gaussian noise with mean zero and standard deviation 0.3 to the training output values.

Let the number of basis functions be $b = 3$ and the basis functions be

$$\varphi_i(x) = x^{i-1} \quad \text{for} \quad i = 1, 2, \ldots, b. \tag{6.4}$$

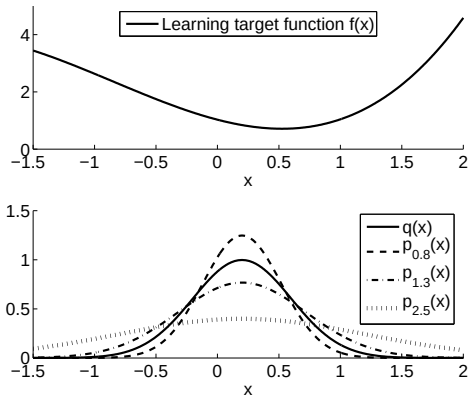


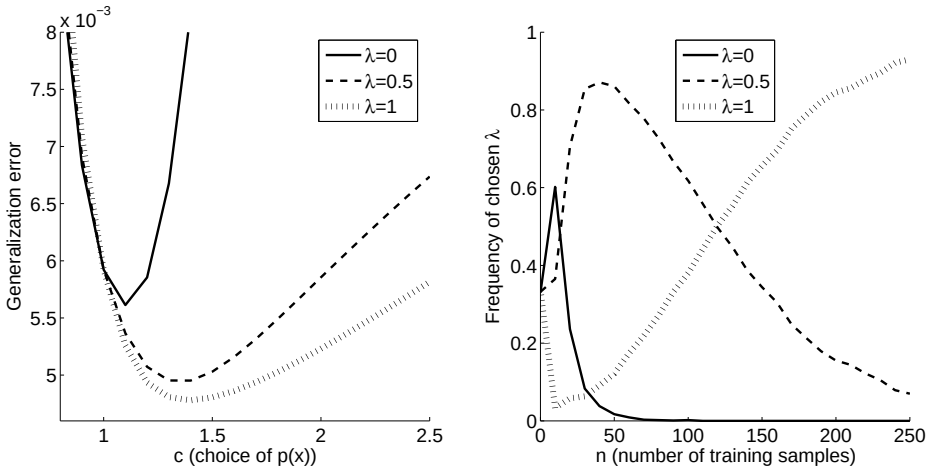
Figure 6.1: Target function, training input density $p_c(x)$, and test input density $q(x)$.

Note that the target function $f(x)$ is not realizable since the model is the second order polynomial. For illustration purposes, we use the above fixed basis functions and focus on choosing the flattening parameter λ by MS; λ is selected from

$$\lambda = 0, 0.5, 1. \tag{6.5}$$

First, we investigate the dependency between the goodness of the training input density (i.e., c) and the model (i.e., λ). For each λ and each c, we draw training input points $\{x_i\}_{i=1}^{100}$ and gather output values $\{y_i\}_{i=1}^{100}$. Then we learn the parameter $\widehat{\boldsymbol{\alpha}}$ by AIWLS and compute the generalization error G. The mean G over 500 trials as a function of c for each λ is depicted in Figure 6.2(a). This graph underlines that the best training input density c could strongly depend on the model λ, implying that a training input density that is good for one model could be poor for others. For example, when the training input density is optimized for the model $\lambda = 0$, $c = 1.1$ would be an excellent choice. However, $c = 1.1$ is not so suitable for other models $\lambda = 0.5, 1$. This figure illustrate a possible weakness of the batch method: when an initially chosen model is significantly different from the finally chosen model, the training input points optimized for the initial model could be less useful for the final model and the performance is degraded.

Next, we investigate the behavior of the sequential approach. In our implementation, 10 training input points are chosen at each iteration. Figure 6.2(b) depicts the transition of the frequency of chosen λ in the sequential learning process over 500 trials. It shows that the choice of models varies over the learning process; a smaller λ (which has smaller variance thus low complexity) is favored in the beginning, but a larger λ (which has larger variance thus higher complexity) tends to be chosen as the number of training



(a) The mean generalization error over 500 trials as a function of training input density c for each λ (when $n = 100$).

(b) Frequency of chosen λ over 500 trials as a function of the number of training samples.

Figure 6.2: Simulation results for the toy dataset.

samples increases. Figure 6.2 illustrates a possible weakness of the sequential method: the target model drifts during the sequential learning process (from small λ to large λ) and the training input points designed in an early stage (for $\lambda = 0$) could be poor for the finally chosen model ($\lambda = 1$).

Finally, we investigate the generalization performance of each method when the number of training samples to gather is

$$n = 50, 100, 150, 200, 250. \tag{6.6}$$

Table 6.1 describes the means and standard deviations of the generalization error obtained by the sequential, batch, and ensemble methods; as a baseline, we also included the result of passive learning, i.e., the training input points $\{x_i\}_{i=1}^n$ are drawn from the test input density $q(x)$ (or equivalently $c = 1$). The table shows that all three ALMS methods tend to outperform passive learning. However, the improvement of the sequential method is not so significant, which would be caused by the model drift phenomenon (see Figure 6.2). The batch method also does not provide significant improvement due to the overfitting to the randomly chosen initial model (see Figure 6.2(a)). On the other hand, the proposed ensemble method does not suffer from these problems and works significantly better than the other methods—the best method in terms of the mean generalization error and comparable methods by the *Wilcoxon signed rank test* at the significance level 5% (Henkel, 1979) are marked by '$\circ$' in the table.

Table 6.1: Means and standard deviations of generalization error for the toy dataset when the number of basis functions b is fixed at 3 and the flattening parameter λ is chosen from $\{0, 0.5, 1\}$ by MS. All values in the table are multiplied by 10^3. The best method in terms of the mean generalization error and comparable methods according to the Wilcoxon signed rank test at the significance level 5% are marked by '$\circ$'.

n	Passive	Sequential	Batch	Ensemble
50	10.63 ± 8.33	7.98 ± 4.57	8.04 ± 4.39	$^\circ 7.59 \pm 4.27$
100	5.90 ± 3.42	5.66 ± 2.75	5.73 ± 3.01	$^\circ 5.15 \pm 2.49$
150	4.80 ± 2.38	4.40 ± 1.74	4.61 ± 1.85	$^\circ 4.13 \pm 1.56$
200	4.21 ± 1.66	3.97 ± 1.54	4.26 ± 1.63	$^\circ 3.73 \pm 1.25$
250	3.79 ± 1.31	3.46 ± 1.00	3.88 ± 1.41	$^\circ 3.35 \pm 0.95$

We also conducted similar experiments when the flattening parameter is fixed at $\lambda = 1$ and the maximum order of polynomials b is chosen by MS; b is selected from

$$b = 2, 3, 4. \tag{6.7}$$

The results are summarized in Table 6.2, showing that the proposed EAL method still works well.

6.2 Benchmark Datasets

Finally, we evaluate whether the advantages of the proposed method are still valid under more realistic settings. For this purpose, we use regression benchmark datasets provided by DELVE (Rasmussen et al., 1996): *Bank-8fm, Bank-8fh, Bank-8nm, Bank-8nh, Kin-8fm, Kin-8fh, Kin-8nm, Kin-8nh, Pumadyn-8fm, Pumadyn-8fh, Pumadyn-8nm, Pumadyn-8nh, Abalone* and *Boston*. Let N be the number of samples: $N = 8192$ for the Bank, Kin, and Pumadyn datasets, $N = 4177$ for the Abalone dataset, and $N = 596$ for the Boston dataset. Each sample consists of d-dimensional input points and 1-dimensional output values, where $d = 8$ for the Bank, Kin, and Pumadyn datasets, $d = 7$ for the Abalone dataset, and $d = 13$ for the Boston dataset. For convenience, every attribute is normalized into $[0, 1]$.

Suppose we are given all N *input* points (i.e., unlabeled samples). Note that output values are kept unknown at this point. From this pool of unlabeled samples, we choose $n = 200$ input points $\{\boldsymbol{x}_i\}_{i=1}^n$ for training and

Table 6.2: Means and standard deviations of generalization error for the toy dataset when the flattening parameter λ is fixed at 1 and the number of basis functions b is chosen from $\{2, 3, 4\}$ by MS. All values in the table are multiplied by 10^3. The best method in terms of the mean generalization error and comparable methods according to the Wilcoxon signed rank test at the significance level 5% are marked by '$\circ$'.

n	Passive	Sequential	Batch	Ensemble
50	26.82 ± 30.60	18.97 ± 24.08	16.36 ± 21.43	$\circ 16.24 \pm 21.54$
100	11.59 ± 17.41	12.91 ± 21.19	7.99 ± 14.59	$\circ 7.21 \pm 13.50$
150	9.38 ± 16.23	$\circ 9.44 \pm 18.32$	5.67 ± 12.16	$\circ 5.60 \pm 12.18$
200	6.90 ± 13.75	6.92 ± 15.77	$\circ 3.37 \pm 7.39$	$\circ 3.78 \pm 8.75$
250	4.01 ± 8.49	5.40 ± 13.89	2.43 ± 4.50	$\circ 2.40 \pm 4.49$

observe the corresponding output values $\{y_i\}_{i=1}^n$. The task is to predict the output values of all N samples.

In this experiment, the test input density $q(\boldsymbol{x})$ is unknown. So we estimate it using the uncorrelated multi-dimensional Gaussian model:

$$q(\boldsymbol{x}) = \frac{1}{(2\pi\widehat{\gamma}_{MLE}^2)^{\frac{d}{2}}} \exp\left(-\frac{\|\boldsymbol{x} - \widehat{\boldsymbol{\mu}}_{MLE}\|^2}{2\widehat{\gamma}_{MLE}^2}\right), \tag{6.8}$$

where $\widehat{\boldsymbol{\mu}}_{MLE}$ and $\widehat{\gamma}_{MLE}$ are the maximum likelihood estimates of the mean and standard deviation obtained from all N unlabeled samples. We select the training input density $p(\boldsymbol{x})$ from the set of uncorrelated multi-dimensional Gaussian densities with mean $\widehat{\boldsymbol{\mu}}_{MLE}$ and standard deviation $c\widehat{\gamma}_{MLE}$, where

$$c = 0.5, 0.6, 0.7, \ldots, 1.5. \tag{6.9}$$

Let the basis functions be Gaussian functions with center $\boldsymbol{t}_i$ and width h: for $i = 1, 2, \ldots, b$,

$$\varphi_i(\boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{t}_i\|^2}{2h^2}\right). \tag{6.10}$$

The centers $\{\boldsymbol{t}_i\}_{i=1}^b$ are randomly chosen from the pool of unlabeled samples. In this experiment, we fix the number of basis functions at $b = 100$ and choose λ from

$$\lambda = 0, 0.5, 1. \tag{6.11}$$

For the Bank, Pumadyn, Abalone, and Boston datasets, the standard deviation h of Gaussian basis functions is chosen from from

$$h = 0.4, 0.8, 1.2, 1.6, 2.0, \tag{6.12}$$

Table 6.3: Means and standard deviations of the generalization error for the DELVE datasets. All values are normalized by the mean generalization error of the passive learning method. The best method in terms of the mean generalization error and comparable methods according to the Wilcoxon signed rank test at the significance level 5% are marked by '$\circ$'.

Dataset	Passive	Sequential	Batch	Ensemble
Bank-8fm	1.0000 ± 1.2545	$.5162 \pm .4892$	$.4139 \pm .0797$	$\circ.4059 \pm .0742$
Bank-8fh	$1.0000 \pm .4054$	$.5473 \pm .2111$	$.4495 \pm .1354$	$\circ.4391 \pm .0945$
Bank-8nm	$1.0000 \pm .6972$	$.6375 \pm .1618$	$.5676 \pm .1187$	$\circ.5572 \pm .1128$
Bank-8nh	$1.0000 \pm .2976$	$.6153 \pm .1574$	$.5326 \pm .1196$	$\circ.5202 \pm .1010$
Kin-8fm	$1.0000 \pm .3488$	$.8228 \pm .2067$	$\circ.7654 \pm .1634$	$\circ.7596 \pm .1570$
Kin-8fh	$1.0000 \pm .3922$	$.7759 \pm .2011$	$.7217 \pm .1530$	$\circ.7164 \pm .1494$
Kin-8nm	$1.0000 \pm .2380$	$.8495 \pm .1673$	$.7756 \pm .1158$	$\circ.7710 \pm .1138$
Kin-8nh	$1.0000 \pm .3629$	$.8676 \pm .2165$	$\circ.8094 \pm .1537$	$\circ.8053 \pm .1409$
Pumadyn-8fm	$1.0000 \pm .2540$	$\circ.8273 \pm .4020$	$.9332 \pm .7677$	$.9320 \pm .7998$
Pumadyn-8fh	$1.0000 \pm .1999$	$.8059 \pm .1703$	$.7440 \pm .2154$	$\circ.7169 \pm .2058$
Pumadyn-8nm	$1.0000 \pm .2044$	$.8286 \pm .1545$	$.7963 \pm .1913$	$\circ.7768 \pm .1824$
Pumadyn-8nh	$1.0000 \pm .2062$	$.8399 \pm .1393$	$.7744 \pm .1546$	$\circ.7634 \pm .1626$
Abalone	1.0000 ± 5.0032	$.2089 \pm .9677$	$\circ.0974 \pm .7885$	$\circ.0552 \pm .2810$
Boston	1.0000 ± 13.1612	$.0141 \pm .1738$	$\circ.0008 \pm .0048$	$\circ.0007 \pm .0035$

and for the Kin datasets, h is chosen from

$$h = 2, 3, 4. \tag{6.13}$$

We again compare the proposed ensemble method with the batch, sequential, and passive methods. In this simulation, we can not create the training input points in an arbitrary location because we only have N samples in the pool. Here, we first create provisional input points following the determined training input density, and then choose the input points from the pool of unlabeled samples that are closest to the provisional input points. In this simulation, the expectation over the test input density $q(\boldsymbol{x})$ in the matrix $\boldsymbol{U}$ (see Eq.(3.20)) is calculated by the empirical average over all N unlabeled samples since the true test error is also calculated as such. For each dataset, we run this simulation 500 times, by changing the template points $\{\boldsymbol{t}_i\}_{i=1}^b$ in each run (thus the choice of training input points is also changed in each trial).

Table 6.3 describes the mean and standard deviation of the generalization error by each method. All the values are normalized by the mean generalization error of the passive learning method for better comparison. In the table, the best method in terms of the mean generalization error and comparable methods according to the *Wilcoxon signed rank test* at the significance level 5% (Henkel, 1979) are marked by '$\circ$'. This shows that all three ALMS methods perform better than the passive learning method. Among them, the proposed ensemble method tends to work significantly better than the other methods.

Based on the simulation results, we conclude that the proposed ensemble approach is useful in ALMS scenarios; thus it could be a promising alternative to the sequential approach, which used to be *de facto* standard.

Chapter 7

Summary

So far, the problems of active learning (AL) and model selection (MS) have been studied as two independent problems, although they both share a common goal of minimizing the generalization error. We suggested that by simultaneously performing AL and MS — which we called *active learning with model selection* (ALMS), a better generalization capability could be achieved. We pointed out that the sequential approach, which would be a common approach to ALMS, can perform poorly due to the model drift phenomenon (Chapter 4.2). A batch approach does not suffer from the model drift problem, but it is hard to choose the initial model appropriately and therefore we argued that the batch approach is not reliable in practice (Chapter 4.3). To overcome the limitations of the sequential and batch approaches, we proposed a new approach called *ensemble active learning* (EAL), which performs AL not only for a single model, but for an ensemble of models (Chapter 5). We have demonstrated through simulations that the proposed EAL method compares favorably with other approaches (Chapter 6).

In theory, IWLS is asymptotically unbiased as long as the support of training and test input distributions are equivalent. However, in practical situations with finite samples, IWLS may not work properly if these two distributions are less overlapped. It is important to theoretically investigate how robust IWLS is when training and test input distributions are significantly different.

In real applications, we are often given unlabeled samples and want to choose the best samples to label. Such a situation is referred to as *pool-based* scenarios. In our experiments, we estimated the input density from the unlabeled samples and showed that the proposed approach significantly outperforms passive learning. However, it would be more promising to develop a method that can directly deal with a finite number of unlabeled samples.

Although we focused on regression problems, the idea of EAL is applicable in any supervised learning scenarios, given that a suitable *batch* AL method is available. This implies that, in principle, it is possible to extend the proposed EAL method to classification problems. However, to the best of our knowledge, there is no reliable batch AL method in classification tasks. Therefore, developing an ALMS method for classification is still a challenging open problem, which needs to be investigated.

We focused on a linear regression model, which is categorized as a regular model in statistics (White, 1982). On the other hand, a lot of attention has been paid to non-regular models such as neural networks (Watanabe, 2001). Fukumizu (2000) proposed an active learning method for neural networks which prunes irrelevant components during the on-line learning process. An interesting future direction to pursue along this line is how to cope with the model drift issues in neural network learning.

Part III

Active Learning in Collaborative Settings

Chapter 8

Introduction

Overview In this chapter, we address the task of active learning for linear regression models in collaborative settings. The goal of active learning is to select training points that would allow accurate prediction of test output values. We propose a new active learning criterion that is aimed at *directly* improving the accuracy of the output value estimation by analyzing the effect of the new training points on the estimates of the output values. The advantages of the proposed method are highlighted in collaborative settings – where most of the data points are missing, and the number of training data points is much smaller than the number of the parameters of the model.

In standard supervised learning settings, we try to learn (approximate) a target function from the data consisting of inputs and the corresponding outputs from the target function (Hastie et al., 2001). Recently, collaborative settings are becoming more common(Goldberg et al., 2001; Adomavicius & Tuzhilin, 2005). In standard settings, it is assumed that all of the data comes from the same function; whereas in *collaborative settings* there is not enough data from the target function to obtain a reliable approximation. However, in collaborative settings, the data from many other functions are available. Utilizing the data from other functions allows us to obtain a better approximation of the target function and/or to reduce the cost of data acquisition. For example, in recommender systems domain, it is common for a user to rate only a small portion of the items. We can approximate the preferences of the target user by analyzing the ratings of other users. In the domain of information retrieval, we can retrieve relevant documents for a specific

user's query by analyzing the documents retrieved by other users. In the domain of bioinformatics, we can predict the properties of a given compound by analyzing the properties of other compounds.

For the task of *active learning*, it is assumed that in order to better approximate the function, we can select inputs for which the output values will be obtained. For example, in order to better learn a user's preferences in the domain of recommender systems, we can ask the user to express preferences for the selected item. However, the degree to which a training point allows us to approximate the function varies. For example, rating a popular item may not be useful for approximating the user's preferences since most users assign a positive rating to a popular item. On the other hand, rating an item which is representative of many other items and whose rating is uncertain may allow us to better approximate the user's preferences. Therefore, choosing samples carefully may allow us to obtain a better approximation of the true function.

There have been a number of works addressing the task of active learning in collaborative settings. Many of the works concentrate on the domain of recommender systems, due to the availability of datasets and the widespread use of collaborative methods (often referred to as collaborative filtering). Nakamura et al. (Nakamura et al., 2003) considered the collaborative filtering problem as an image restoration problem; where a user's preferences are restored from the items for which the user has expressed preferences. Selecting the item to be rated is then formulated as finding a restriction operator (an operator that restricts the components to those having observed values) that minimizes the expected squared error. Boutiler et al. (Boutilier et al., 2003) selected the item to rate based on the expected value of information with respect to the explicit probabilistic model; where the value of information is expressed as the expected improvement in the decision quality after the item is rated. In (Goldberg et al., 2001; Dasgupta et al., 2003), the items to be rated are selected so that they would allow the system to assign the user to a certain stereotype.

Linear regression based methods are often used in collaborative filtering (Adomavicius & Tuzhilin, 2005). However, there are few works on active learning in collaborative settings for linear regression based methods (Yu et al., 2006; Adomavicius & Tuzhilin, 2005). Standard active learning methods for linear regression models (Chan, 1981; John & Draper, 1975; Dette & Studden, 1993) are not necessarily well suited for collaborative settings due to the small sample size. As a result, the performance of these methods may be poor. Motivated by this, we develop an active learning method that takes the peculiarities of collaborative settings into account.

Chapter 9

Problem Formulation

9.1 Linear Regression in Collaborative Settings

Let us formulate the task of function approximation for collaborative settings in a linear regression form. As illustrated in Figure 9.1, we want to approximate the output values $\mathbf{y}$ of the target function through the linear combination of the output values of other functions (corresponding to the column vectors of matrix $\mathbf{X}$) weighted by the parameters $\boldsymbol{\beta}$:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{9.1}$$

$\mathbf{X} \in \mathbb{R}^{t \times p}$, where t is the number of inputs and p is the number of functions; $\mathbf{y} \in \mathbb{R}^t$, parameters $\boldsymbol{\beta} \in \mathbb{R}^p$, and $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_t)$ normally distributed i.i.d. noise with mean zero and unknown variance σ^2. We can obtain the least squares estimator $\widehat{\boldsymbol{\beta}}$ of the parameter values as:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}, \tag{9.2}$$

where $^\top$ denotes the transpose. However, in the collaborative settings the matrix $\mathbf{X}$ is often sparse, so the matrix $\mathbf{X}^\top\mathbf{X}$ is singular and is not invertible. To cope with this, we add a regularization constant $\alpha\mathbf{I}$ to $\mathbf{X}^\top\mathbf{X}$ (where the value of α is positive and is small e.g. $\alpha = 0.1$). This ensures that $\mathbf{X}^\top\mathbf{X} + \alpha\mathbf{I}$ is full rank (invertible), and improves numerical stability. Parameters $\widehat{\boldsymbol{\beta}}$ could now be expressed as:

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X} + \alpha\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}. \tag{9.3}$$

We can approximate the output values $\mathbf{y}^*$ of the test inputs by estimates $\widehat{\mathbf{y}}$ as:

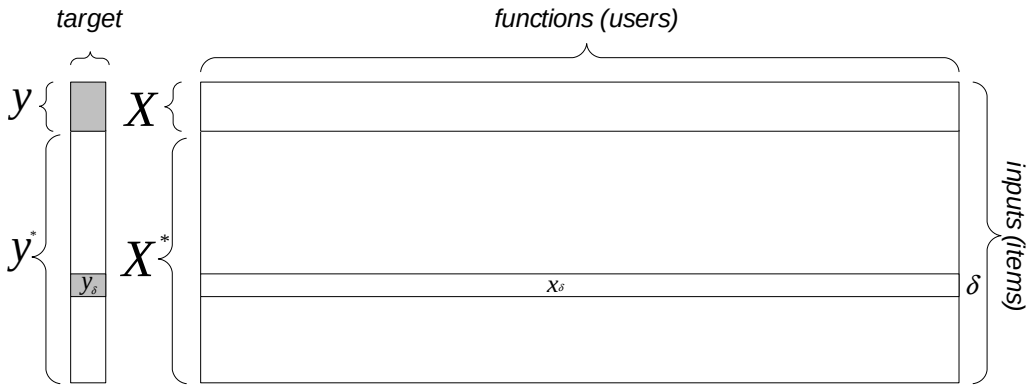


Figure 9.1: A matrix entry corresponds to the output value of a function for an input (for recommender systems it corresponds to a rating of an item by a user). The matrix is sparse (most of the entries are missing). The task is to select an input δ for which the output value of the target function y_δ will be provided, so as to better approximate the output values $\mathbf{y}^*$.

$$\widehat{\mathbf{y}} = \mathbf{X}^* \widehat{\boldsymbol{\beta}}, \tag{9.4}$$

where $\mathbf{X}^*$ are the output values of the functions for the test inputs. We measure how well $\widehat{\mathbf{y}}$ approximates $\mathbf{y}^*$ by the generalization error G:

$$G(\widehat{\mathbf{y}}) = \|\widehat{\mathbf{y}} - \mathbf{y}^*\|^2 . \tag{9.5}$$

9.2 Active Learning (AL)

We consider the following task. We are allowed to sequentially select for which inputs the output values (of the target function) are obtained. We want to select an input δ, so that obtaining and adding its output value y_δ to the existing output values $\mathbf{y}$ minimizes the generalization error G:

$$argmin_\delta G. \tag{9.6}$$

Chapter 10

Related Work

An information matrix is typically used for identifying inputs, obtaining output values for which, allows us to reduce the generalization error. The inverse of the information matrix $\mathbf{A}$:

$$\mathbf{A}^{-1} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1}, \tag{10.1}$$

allows us to estimate the variances of the approximated parameters $\widehat{\boldsymbol{\beta}}$:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}$$

The covariance of parameters could be expressed as:

$$Cov(\widehat{\boldsymbol{\beta}}) = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top Cov(\mathbf{y}) \left(\mathbf{X}^\top \mathbf{X}\right)^{-1},$$

by assuming that $Cov(\mathbf{y}) = \boldsymbol{\sigma}^2 \mathbf{I}$ we can rewrite the above equation further as:

$$\begin{aligned} Cov(\widehat{\boldsymbol{\beta}}) &= \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \left(\mathbf{X}^\top \mathbf{X}\right) \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \\ &= \left(\mathbf{X}^\top \mathbf{X}\right)^{-1}. \end{aligned}$$

The active learning task could then be formulated as the minimization of the parameter's estimation error based on a particular optimality criterion of the information matrix.

A-optimal The A-optimal design (Chan, 1981) seeks to minimize the trace of the inverse of the information matrix i.e.:

$$\min tr\mathbf{A}^{-1}. \tag{10.2}$$

This criterion results in minimizing the average variance of the estimates of the regression parameters $\widehat{\boldsymbol{\beta}}$.

D-optimal The D-optimal design (John & Draper, 1975) seeks to maximize the determinant of the information matrix i.e.:

$$\max |\mathbf{A}| . \tag{10.3}$$

This criterion results in maximizing the differential Shannon information content of the parameter estimates.

E-optimal The E-optimal design (Dette & Studden, 1993) seeks to minimize the 2-norm of the inverse of the information matrix i.e.:

$$\min \left\| \mathbf{A}^{-1} \right\|_2 . \tag{10.4}$$

This criterion results in maximizing the minimum eigenvalue of the information matrix.

Transductive Experimental Design The transductive experimental design (Yu et al., 2006) seeks to maximize the trace of the information matrix with respect to test points i.e.:

$$\max tr \left(\mathbf{X}^* \mathbf{X}^\top \mathbf{A}^{-1} \mathbf{X} \mathbf{X}^{*\top} \right) . \tag{10.5}$$

This criterion results in finding representative training points that span a linear space for retaining most of the information of the test points.

Chapter 11

Proposed Method

11.1 Motivations

The methods described in Section 10 tend to indirectly improve the estimates of output values $\widehat{\mathbf{y}}$ by improving the estimates of parameters $\widehat{\boldsymbol{\beta}}$. However, in collaborative settings, this may not necessarily be efficient for the reasons outlined bellow:

- The ultimate goal is to obtain good estimates $\widehat{\mathbf{y}}$ of the output values $\mathbf{y}$, and not necessarily good estimates $\widehat{\boldsymbol{\beta}}$ of the parameters $\boldsymbol{\beta}$.
- Traditional optimal design methods tend to assume that the bias is sufficiently small, and concentrate on minimizing the variance. However, in the current settings, the value of bias is not necessarily small, since the number of training points is much smaller than the number of parameters. So reducing the variance and ignoring the bias may not necessarily be an effective way of minimizing the generalization error.
- In the current settings, the size of $\mathbf{y}$ is smaller than the size of $\boldsymbol{\beta}$, so optimizing estimates of $\mathbf{y}$ (instead of estimates of $\boldsymbol{\beta}$) may be more computationally efficient.

11.2 Method

The generalization error measures how well the estimated output values approximate the true output values. Traditionally, active learning methods attempt to improve the generalization error indirectly by identifying training points that will improve the estimates of the parameters (used in turn for obtaining the estimated output values). However, in collaborative settings,

improving the parameter estimates may not necessarily be an effective way of reducing the generalization error (as discussed in Chapter Section 11.1). 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. Let us denote the input of the next training point 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.:

$$argmin_\delta G_{t+1} = argmax_\delta(G_t - G_{t+1}). \tag{11.1}$$

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 existing ratings $\mathbf{y}$. 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 $\widehat{\mathbf{y}}_{t+1}$:

$$\begin{aligned}
\triangle G =& G_t - G_{t+1} \\
=& \|\widehat{\mathbf{y}}_t - \mathbf{y}^*\|^2 - \|\widehat{\mathbf{y}}_{t+1} - \mathbf{y}^*\|^2 \\
=& \|\widehat{\mathbf{y}}_t\|^2 - 2\langle\widehat{\mathbf{y}}_t, \mathbf{y}^*\rangle + \|\mathbf{y}^*\|^2 - \|\widehat{\mathbf{y}}_{t+1}\|^2 + 2\langle\widehat{\mathbf{y}}_{t+1}, \mathbf{y}^*\rangle - \|\mathbf{y}^*\|^2 \\
=& \|\widehat{\mathbf{y}}_t\|^2 - 2\langle\widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}, \mathbf{y}^*\rangle - \|\widehat{\mathbf{y}}_{t+1}\|^2 .
\end{aligned}$$

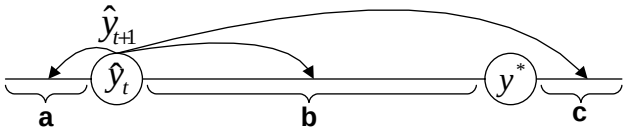


Figure 11.1: Location of the estimate of the output value $\widehat{y}$ after the training point δ is added to the training set (making the number of training points equal to $t+1$).

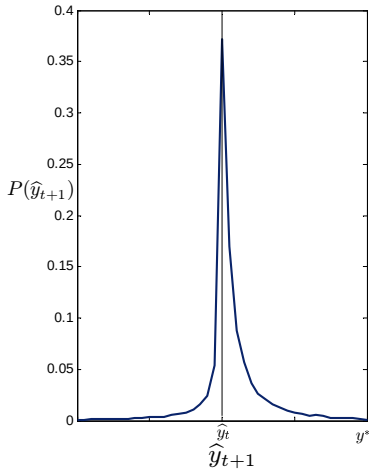


Figure 11.2: Distribution of $\widehat{y}_{t+1}$ in relation to y^* and $\widehat{y}_{t+1}$ (Chapter 12.3).

Defining $\boldsymbol{\epsilon} = \mathbf{y}^* - \widehat{\mathbf{y}}_{t+1}$, we have

$$\begin{aligned}
&\|\widehat{\mathbf{y}}_t\|^2 - 2\left\langle \widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}, \mathbf{y}^* \right\rangle - \|\widehat{\mathbf{y}}_{t+1}\|^2 \\
&= \|\widehat{\mathbf{y}}_t\|^2 - 2\left\langle \widehat{\mathbf{y}}_t, \widehat{\mathbf{y}}_{t+1} \right\rangle + \|\widehat{\mathbf{y}}_{t+1}\|^2 + 2\left\langle \widehat{\mathbf{y}}_{t+1} - \widehat{\mathbf{y}}_t, \boldsymbol{\epsilon} \right\rangle \\
&= \|\widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}\|^2 + 2\left\langle \widehat{\mathbf{y}}_{t+1} - \widehat{\mathbf{y}}_t, \boldsymbol{\epsilon} \right\rangle . \qquad (11.2)
\end{aligned}$$

Note that this decomposition is different from the standard bias-variance decomposition. Let us denote the first term of the above Eq. (11.2) by T_1:

$$T_1 = \|\widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}\|^2 ,$$

and the second term by T_2:

$$T_2 = 2\left\langle \widehat{\mathbf{y}}_{t+1} - \widehat{\mathbf{y}}_t, \boldsymbol{\epsilon} \right\rangle .$$

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 the values in $\mathbf{y}^*$, since $\boldsymbol{\epsilon} = \mathbf{y}^* - \widehat{\mathbf{y}}_{t+1}$ and $\mathbf{y}^*$ is not accessible. 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 11.1. 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 Figure 11.2, we show the distribution of the location of $\widehat{y}_{t+1}$ relative to $\widehat{y}_t$ and y^* (plotted from the data from the numerical experiment described in Chapter 12.3). 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. Since T_1 is less prone to error and is more likely to be applicable, we use it as an estimator of $\triangle G$ and define the active learning criterion J as:

$$J(\delta) = \left\| \widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1} \right\|^2 . \tag{11.3}$$

The training point is then selected as:

$$argmax_\delta J(\delta). \tag{11.4}$$

11.3 Criterion Formulation in Linear Regression Settings

Let us formulate the proposed criterion for the linear regression settings (Chapter Section 9.1) as:

$$J(\delta) = \left\| \widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1} \right\|^2 = \left\| \mathbf{X}^* \left(\widehat{\boldsymbol{\beta}}_t - \widehat{\boldsymbol{\beta}}_{t+1} \right) \right\|^2 , \tag{11.5}$$

where the least-squares estimators $\widehat{\boldsymbol{\beta}}_t$, $\widehat{\boldsymbol{\beta}}_{t+1}$ of the parameter values are obtained by using Eq. (9.3). Let

$$\mathbf{A} = \mathbf{X}^\top \mathbf{X} + \alpha \mathbf{I}, \tag{11.6}$$

where $\alpha\mathbf{I}$ is a regularization parameter (where $0 < \alpha \ll 1$), this ensures that matrix $\mathbf{A}$ is invertible. We can rewrite the parameter estimate $\widehat{\boldsymbol{\beta}}_t$ as:

$$\widehat{\boldsymbol{\beta}}_t = \mathbf{A}^{-1}\mathbf{X}^\top \mathbf{y}. \tag{11.7}$$

The parameters $\widehat{\boldsymbol{\beta}}_{t+1}$, after the output value for the input δ was added could be expressed as:

$$\begin{aligned}\widehat{\boldsymbol{\beta}}_{t+1} &= (\mathbf{A} + \mathbf{x}_\delta \mathbf{x}_\delta^\top)^{-1}(\mathbf{X}^\top \mathbf{y} + \mathbf{x}_\delta y_\delta) \\ &= (\mathbf{A} + \mathbf{x}_\delta \mathbf{x}_\delta^\top)^{-1}\mathbf{X}^\top \mathbf{y} + (\mathbf{A} + \mathbf{x}_\delta \mathbf{x}_\delta^\top)^{-1}\mathbf{x}_\delta y_\delta. \end{aligned} \tag{11.8}$$

We can rewrite the first term of Eq. (11.8) by using the Woodbury formula (Woodbury, 1950) as:

$$(\mathbf{A} + \mathbf{x}_\delta \mathbf{x}_\delta^\top)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{x}_\delta \mathbf{x}_\delta^\top \mathbf{A}^{-1}}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta}.$$

Then expand the terms of Eq. (11.8) as:

$$(\mathbf{A} + \mathbf{x}_\delta \mathbf{x}_\delta^\top)^{-1}\mathbf{X}^\top \mathbf{y} = \mathbf{A}^{-1}\mathbf{X}^\top \mathbf{y} - \frac{\mathbf{A}^{-1}\mathbf{x}_\delta \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{X}^\top \mathbf{y}}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta} \tag{11.9}$$

and

$$(\mathbf{A} + \mathbf{x}_\delta \mathbf{x}_\delta^\top)^{-1}\mathbf{x}_\delta y_\delta = \mathbf{A}^{-1}\mathbf{x}_\delta y_\delta - \frac{\mathbf{A}^{-1}\mathbf{x}_\delta \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta y_\delta}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta}. \tag{11.10}$$

The difference between the parameter estimates could then be expressed as:

$$\begin{aligned}\widehat{\boldsymbol{\beta}}_{t+1} - \widehat{\boldsymbol{\beta}}_t =& \mathbf{A}^{-1}\mathbf{x}_\delta y_\delta - \frac{\mathbf{A}^{-1}\mathbf{x}_\delta \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta y_\delta}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta} - \frac{\mathbf{A}^{-1}\mathbf{x}_\delta \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{X}^\top \mathbf{y}}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta} \\ =& \mathbf{A}^{-1}\mathbf{x}_\delta y_\delta \frac{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta - \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta} - \frac{\mathbf{A}^{-1}\mathbf{x}_\delta \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{X}^\top \mathbf{y}}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta} \\ =& \frac{\mathbf{A}^{-1}\mathbf{x}_\delta (y_\delta - \mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t)}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta}. \end{aligned}$$

The difference between the output values could now be expressed as:

$$\widehat{\mathbf{y}}_{t+1} - \widehat{\mathbf{y}}_t = \mathbf{X}^* \frac{\mathbf{A}^{-1}\mathbf{x}_\delta (y_\delta - \mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t)}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta}.$$

The proposed criterion is then formulated as:

$$\begin{aligned} J(\delta) &= \|\widehat{\mathbf{y}}_{t+1} - \widehat{\mathbf{y}}_t\|^2 \\ &= \left(\frac{y_\delta - \mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta} \right)^2 \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{X}^{*\top}\mathbf{X}^*\mathbf{A}^{-1}\mathbf{x}_\delta. \end{aligned} \tag{11.11}$$

11.4 Interpretation

Let us look at a possible interpretation of the proposed criterion (Eq. (11.11)) and its relation to existing active learning criterions. Let us rewrite the criterion as:

$$\begin{aligned} J(\delta) &= (y_\delta - \mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t)^2 \frac{\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{X}^{*\top} \mathbf{X}^* \mathbf{A}^{-1} \mathbf{x}_\delta}{(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2} \\ &= J_R \frac{J_S}{J_P}, \end{aligned} \tag{11.12}$$

where

$$J_R = (y_\delta - \mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t)^2, \tag{11.13}$$

$$J_S = \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{X}^{*\top} \mathbf{X}^* \mathbf{A}^{-1} \mathbf{x}_\delta, \tag{11.14}$$

$$J_P = (1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2. \tag{11.15}$$

The term $J_R = (y_\delta - \mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t)^2$ represents the residual value, i.e. the squared error between the actual output value y_δ and its estimate $\mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t$. The $\mathbf{x}_\delta$ with larger residual value is then favored by the term J_R. Taking the residual value into account corresponds to the residual-based active learning methods(**?**).

The next term J_S may be rewritten further as:

$$\begin{aligned} J_S &= \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{X}^{*\top} \mathbf{X}^* \mathbf{A}^{-1} \mathbf{x}_\delta \\ &= \sum_{\mathbf{x}_t \in \boldsymbol{X}^*} \left(\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_t \right)^2, \end{aligned} \tag{11.16}$$

where $\boldsymbol{X}^*$ denotes a set of row vectors of the matrix $\mathbf{X}^*$. By noticing that $\mathbf{x}_\delta \in \boldsymbol{X}^*$, we can further rewrite the above term as:

$$J_S = (\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2 + \sum_{\mathbf{x}_t \in \boldsymbol{X}^* \setminus \mathbf{x}_\delta} \left(\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_t \right)^2. \tag{11.17}$$

The part $\frac{J_S}{J_P}$ of the proposed criterion could now be rewritten as:

$$\begin{aligned} \frac{J_S}{J_P} &= \frac{(\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2 + \sum_{\mathbf{x}_t \in \boldsymbol{X}^* \setminus \mathbf{x}_\delta} \left(\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_t \right)^2}{(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2} \\ &= \frac{(\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2}{(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2} + \frac{\sum_{\mathbf{x}_t \in \boldsymbol{X}^* \setminus \mathbf{x}_\delta} \left(\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_t \right)^2}{(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2} \\ &= J_O + J_T, \end{aligned} \tag{11.18}$$

where

$$J_O = \frac{(\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta)^2}{(1+\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta)^2}\,, \tag{11.19}$$

$$J_T = \frac{\sum_{\mathbf{x}_t \in \boldsymbol{X}^* \setminus \mathbf{x}_\delta} \left(\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_t\right)^2}{(1+\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta)^2}. \tag{11.20}$$

In order to interpret the meaning of the terms J_O and J_T, let us eigen-decompose the matrix $\mathbf{X}^\top\mathbf{X}$ into its eigenvalues and eigenvectors as:

$$\mathbf{X}^\top\mathbf{X} = \sum_{i=1}^{p} \lambda_i \boldsymbol{\varphi}_i \boldsymbol{\varphi}_i^\top, \tag{11.21}$$

where λ_i are eigenvalues such that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m > \lambda_{m+1} = \ldots = \lambda_d = 0$ and associated eigenvectors $\boldsymbol{\varphi}_i$, and m is the rank of the matrix $\mathbf{X}^\top\mathbf{X}$. We can also rewrite the matrix $\mathbf{A}$ as:

$$\begin{aligned}\mathbf{A} &= \mathbf{X}^\top\mathbf{X} + \alpha\mathbf{I} \\ &= \sum_{i=1}^{p} \lambda_i \boldsymbol{\varphi}_i \boldsymbol{\varphi}_i^\top + \alpha\mathbf{I}.\end{aligned} \tag{11.22}$$

Let us examine the conditions under which the value of J_O increases. Let $\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta = a$, then we can rewrite J_O as:

$$J_O = \frac{a^2}{(a+1)^2}. \tag{11.23}$$

The value of J_O is non-negative and is monotone increasing with respect to a. So let us examine under which conditions the value of a is large. We can rewrite a by using Eq. (11.22) as:

$$\begin{aligned}a &= \sum_{i=1}^{p} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)^2 \frac{1}{\lambda_i + \alpha} \\ &= \sum_{i=1}^{m} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)^2 \frac{1}{\lambda_i + \alpha} + \frac{1}{\alpha} \sum_{i=m+1}^{p} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)^2.\end{aligned} \tag{11.24}$$

Since α is set to a value close to zero (Eq. (11.6)), the $\frac{1}{\alpha}\sum_{i=m+1}^{p} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)^2$ part dominates in the above equation. We may then approximate a as:

$$a \approx \frac{1}{\alpha} \sum_{i=m+1}^{p} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)^2 . \tag{11.25}$$

The value of $\frac{1}{\alpha}\sum_{i=m+1}^{p}\left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)^2$ is large when we choose $\mathbf{x}_\delta$ that belongs to the null space of $\mathbf{X}^\top\mathbf{X}$ spanned by $\{\boldsymbol{\varphi}_i\}_{i=m+1}^{p}$. This is equivalent to $\mathbf{x}_\delta$ being orthogonal to the *training space* (the range of $\mathbf{X}^\top\mathbf{X}$ spanned by $\{\boldsymbol{\varphi}_i\}_{i=1}^{m}$). So the J_O part of the criterion favors $\mathbf{x}_\delta$ that is 'not close' to the training space. This would be related to variance-based AL methods (John & Draper, 1975; Chan, 1981; Dette & Studden, 1993; Sugiyama & Ogawa, 2000).

Let us examine the conditions under which the value of J_T increases. Let us try to simplify the formulation of the J_T. By using the fact that the denominator $(1+\mathbf{x}_\delta^\top\mathbf{A}^{-1}\mathbf{x}_\delta)^2 \geq 1$, we may obtain the lower bound of J_T as:

$$J_T \geq \sum_{\mathbf{x}_t \in \boldsymbol{X}^* \setminus \mathbf{x}_\delta} \left(\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_t\right)^2 . \tag{11.26}$$

We can rewrite the $\mathbf{x}_\delta^\top\mathbf{A}^{-1}\mathbf{x}_t$ part of the above equation as:

$$\begin{aligned}
\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_t &= \sum_{i=1}^{p} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)\left(\mathbf{x}_t^\top \boldsymbol{\varphi}_i\right)\frac{1}{\lambda_i+\alpha} \\
&= \sum_{i=1}^{m} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)\left(\mathbf{x}_t^\top \boldsymbol{\varphi}_i\right)\frac{1}{\lambda_i+\alpha} + \frac{1}{\alpha}\sum_{i=m+1}^{p} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)\left(\mathbf{x}_t^\top \boldsymbol{\varphi}_i\right) .
\end{aligned} \tag{11.27}$$

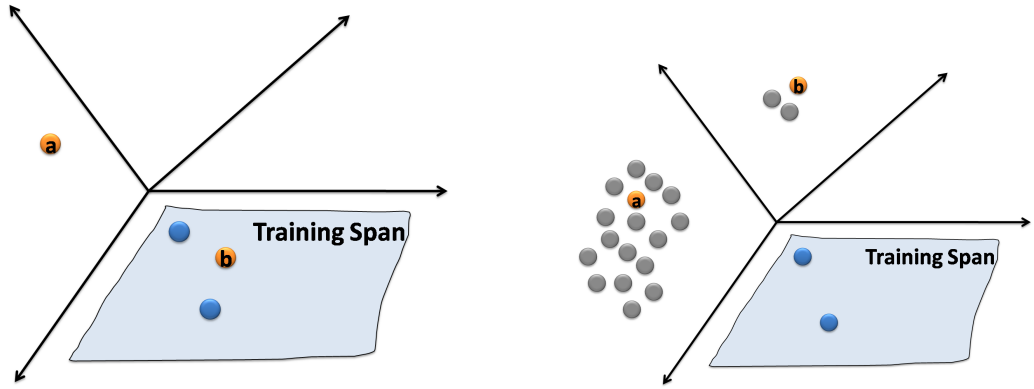
Since α is set to a value close to zero (Eq. (11.6)), the $\frac{1}{\alpha}\sum_{i=m+1}^{p}\left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)\left(\mathbf{x}_t^\top \boldsymbol{\varphi}_i\right)$ part dominates in the above equation. We may then approximate the above equation as:

$$\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_t \approx \frac{1}{\alpha}\sum_{i=m+1}^{p} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)\left(\mathbf{x}_t^\top \boldsymbol{\varphi}_i\right) , \tag{11.28}$$

and may now approximate the lower bound of the term J_T as:

$$\begin{aligned}
J_T &\geq \sum_{\mathbf{x}_t \in \boldsymbol{X}^* \setminus \mathbf{x}_\delta} \left(\mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_t\right)^2 \\
&\approx \sum_{\mathbf{x}_t \in \boldsymbol{X}^* \setminus \mathbf{x}_\delta} \left(\frac{1}{\alpha}\sum_{i=m+1}^{d} \left(\mathbf{x}_\delta^\top \boldsymbol{\varphi}_i\right)\left(\mathbf{x}_t^\top \boldsymbol{\varphi}_i\right)\right)^2 .
\end{aligned} \tag{11.29}$$

The term J_T favors $\mathbf{x}_\delta$ whose projection onto the null space of $\mathbf{X}^\top\mathbf{X}$ is 'close' to the projections of the vectors $\boldsymbol{X}^* \setminus \mathbf{x}_\delta$ onto the null space. Taking the test



(a) J_O favors points that are not in the training space. Point (a) is then favored.

(b) J_T favors points that are close to the test points in the null space. Point (a) then is favored.

: training point
: test point
: point under consideration
Points are pictured in the eigen-space.

Figure 11.3: Criterion Interpretation.

points into account is related to the transductive active learning method (Yu et al., 2006).

11.5 Implementation Considerations

In this section we address the details that need to be considered for the implementation of the proposed active learning criterion.

Criterion Estimation We are not able to calculate the value of the proposed criterion directly since the output value of the sample y_δ is not known. Let us denote by $J(\delta|\, y_\delta = r)$ the value of the criterion $J(\delta)$ when $y_\delta = r$. We may then approximate the value of the criterion as:

$$J(\delta) \approx \sum_r P(y_\delta = r) J(\delta|\, y_\delta = r),$$

Since we assume no prior knowledge of $P(y_\delta = r)$, we approximate it by the non-informative uniform distribution.

Handling Missing Data In collaborative settings, the data matrix is assumed to be very sparse, so proper handling of missing values (e.g. items that the user provided no ratings for) is important. We handle missing values in the following way. First, we calculate the mean output value of a function and use it to centralize the output values of the given function. Missing values are then assigned a value of zero, which corresponds to the mean output value of the function.

Chapter 12

Numerical Experiments

12.1 Experiment Settings

Let us describe the settings that are common to the experiments. We have selected a popular collaborative dataset MovieLens (GroupLens, University of Minnesota, 2005) for the numerical experiments. The MovieLens dataset consists of approximately 1 million ratings for $3,900$ movies by $6,040$ users. We randomly select 100 users that have each rated at least 100 items. For each user, we randomly select 50 points (items) as potential training points and use the rest of the points as a test set. All of the users' output values (ratings) are withheld. For each user, training points are selected in a sequential manner by an active learning algorithm. After the training point is selected, its output value is revealed and the point is added to the training set. For the random active learning method, training points are selected following the uniform distribution. For all of the applicable active learning methods, the value of α (of the regularization parameter in Eq. (11.6)) is set to 0.1.

12.2 Effect of Active Learning

In this experiment, we investigate the effect of active learning (training point selection) on the generalization error. We use a random active learning algorithm to sequentially select 20 training points for each user. At each step, we record the change in the generalization error $\triangle G$. Results of the experiment are presented in Figure 12.1. In line with the expectations, the effect of training point selection on generalization error is large when the number of training points is small. As the number of training points increases, the effect of training point selection decreases, and flattens out after the number

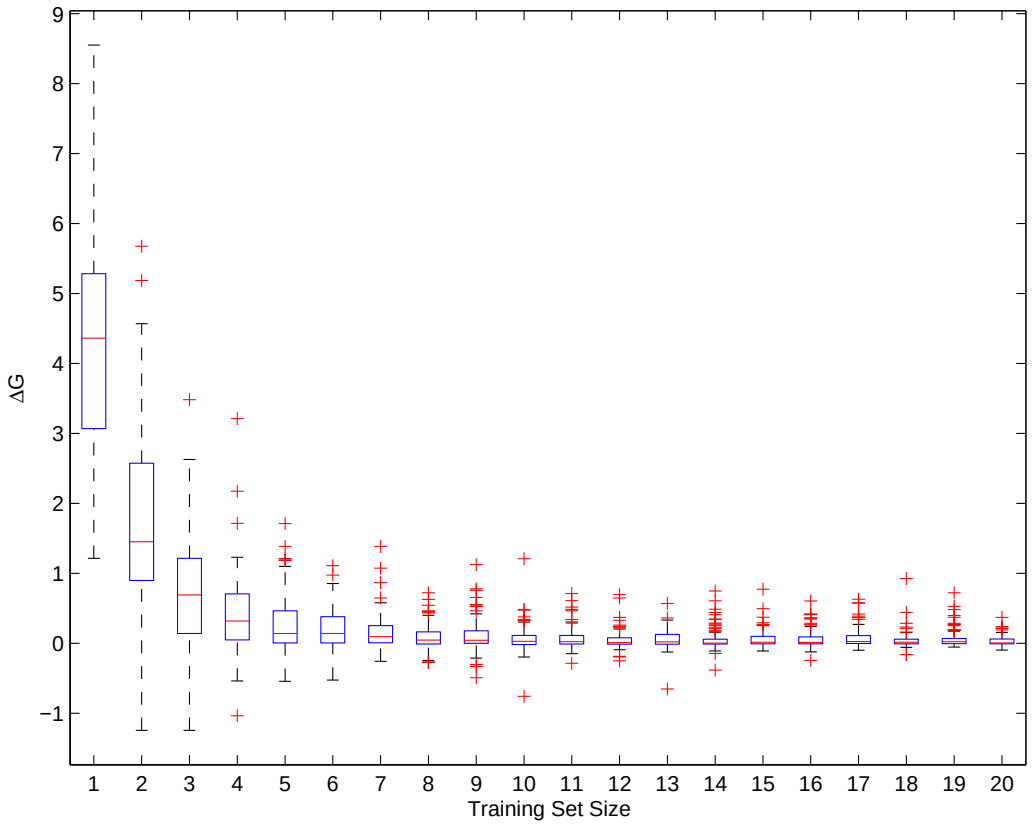


Figure 12.1: Effect of the training point selection (active learning) on the generalization error with respect to the training set size (Chapter 12.2).

of training points is larger than 10. In order to better distinguish the effect of active learning on generalization error, we limit the size of the training set to 10 for the rest of the experiments.

12.3 Validity of Assumptions

In this experiment, we investigate whether the assumptions that the proposed algorithm relies upon are satisfied. As discussed in Chapter 11.2, the proposed criterion relies on the value of the error (of the output estimate) not increasing, and the output estimate not overshooting the true value. We use the experiment settings described in Chapter 12.1 and the random active learning method. For each run, we record the values of $\widehat{y}_t$, $\widehat{y}_{t+1}$ and $y^*$and then plot the distribution of the $\widehat{y}_{t+1}$ normalized by $\widehat{y}_t - y^*$. From results shown in Figure 11.2, we can see that deterioration of the estimate

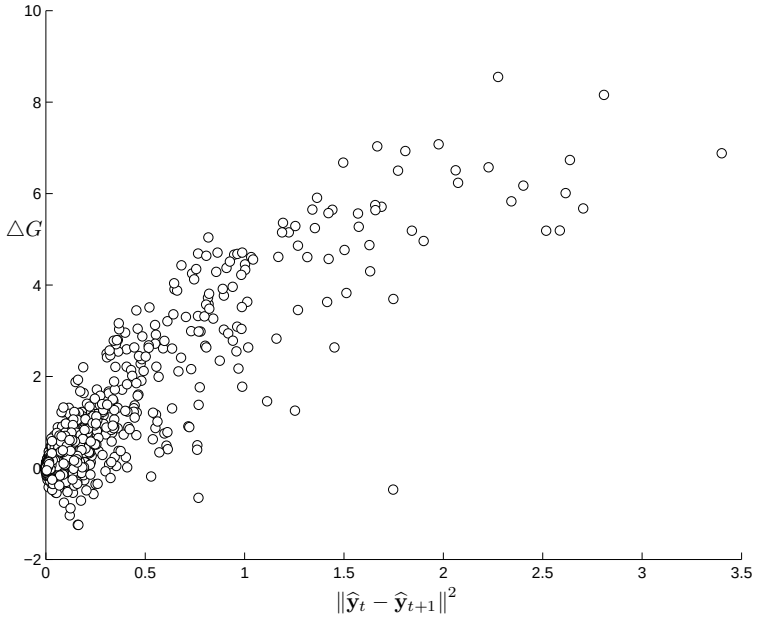


Figure 12.2: Relation between the value of $T_1 = \|\widehat{\mathbf{y}}_t - \widehat{\mathbf{y}}_{t+1}\|$ and the value that it tries to approximate $\triangle G$ (Chapter 12.4).

and overshooting of the true value occurs with the low probability and the value of the resulting error is likely to be small. Therefore, due to only a mild violation of the assumptions, the proposed criterion is still likely to be accurate.

12.4 Criterion Accuracy Evaluation

In this experiment, we evaluate how accurately the proposed criterion estimates the change in the generalization error (caused by the addition of the new training point). We use the experiment settings described in Chapter 12.1 and the random active learning method. For each run, we record the actual values of the proposed criterion T_1, and the value that it estimates $\triangle G$. Results are presented in Figure 12.2. The term T_1 models $\triangle G$ well, except in a relatively rare situations where $\triangle G < 0$. However, for the active learning task, we are interested in the training points that improve the model i.e. $\triangle G > 0$. Therefore for the task of active learning, the proposed criterion could be considered a good predictor of $\triangle G$.

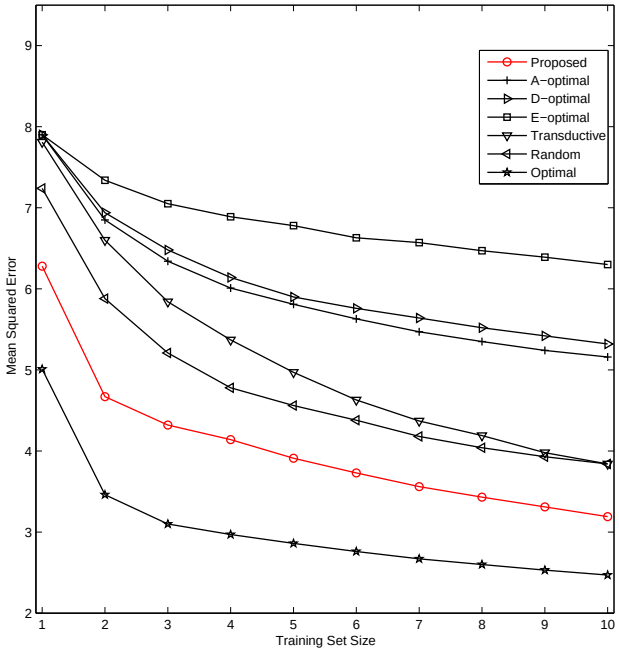


Figure 12.3: Evaluation of active learning criterions (Chapter 12.5).

12.5 Comparison with existing Active Learning algorithms

In this experiment, we evaluate how the proposed method compares with existing methods (Section 10), a random active learning method, and an optimal method (that selects the best possible training sample that results in the largest possible reduction of the generalization error). In order to calculate the value of the proposed criterion, the estimate of the output value y_δ is required. We assume no prior knowledge, and estimate the expected value of the criterion by assuming a uniform distribution over the output values of the input δ (Chapter 11.5). Results are presented in Figure 12.3. Existing methods appear to perform poorly under collaborative settings, since most of the existing methods (with the exception of transductive active learning at the later stages) are outperformed by the random method. Existing algorithms do not specifically consider collaborative settings, and this appears to be detrimental to their performance. The proposed algorithm has the best performance (at the statistical significance level of 95%); hence it appears to be a promising active learning method for collaborative settings.

Chapter 13

Summary

Utilizing the training data from other targets is beneficial. However, the inability to cope with sparse settings appears to be detrimental to the performance of traditional active learning methods. Traditional methods tend to select training points at to improve the estimates of the function's parameters. Due to the sparseness of data, improved parameter estimates do not necessary translate into improved output estimates, and as a result the performance may suffer.

We proposed an active learning criterion that aims to *directly* improve the estimates of the output values. To accomplish this, we select new training points to be added to the training set, that cause a large change in the estimates of output values. As a result, the proposed criterion favors training points that are not represented by the existing training set and are representative of the test points (Chapter 11.4). In experiments, the proposed approach performs favorably under collaborative settings in comparison with traditional approaches. An important distinction of the proposed approach is that while it is model-independent, it could be interpreted/justified with regards to the *model-based* AL methods such as residual, training set-based and transductive AL methods. From an integration standpoint the proposed method also provides an important advantage. It relies only on the estimates of the output values. Existing systems are capable by design of providing the estimates of the output values. Therefore the active learning capability could be added to existing systems without the need to modify any of its existing components.

Part IV

Conclusion and Future Work

Chapter 14

Conclusion

The main message of this dissertation is that by not treating the problem of active learning in *isolation* its effectiveness could be further improved. We show two aspects through which this could be accomplished.

One approach is to perform active learning with regards to another task that is dependent on AL and/or AL is dependent on that task. We have selected the task of model selection, since AL depends on MS, and MS depends on AL. Active learning depends on model selection, since in order to select training points (AL) the model should be fixed (i.e. MS should have been performed). In turn, model selection depends on active learning, since in order to select the model (MS) the training points should have been obtained (i.e. AL must have been performed). Although AL (Fedorov, 1972; MacKay, 1992b; Cohn et al., 1996; Fukumizu, 2000; Wiens, 2000; Kanamori & Shimodaira, 2003; Sugiyama, 2006) and MS (Akaike, 1974; Rissanen, 1978; Schwarz, 1978; Craven & Wahba, 1979; Shimodaira, 2000; Sugiyama & Müller, 2005) share a common goal of minimizing the generalization error, simply combining existing AL and MS methods in a batch manner (where all the training points selected at once) may not be possible due to the circular dependency between AL and MS as mentioned above. An alternative way of combining AL and MS in an sequential manner (where for MS and AL are performed iteratively for each of the training points) can perform poorly due to the identified model drift phenomenon (Chapter 4.2). To overcome the limitations of combining AL with MS of the sequential and batch approaches, we proposed a new approach that performs AL with regards to MS. The proposed approach selects training points that are suited not only for a single model (which may be inferior in the batch approach, or drifting in the sequential approach), but for *all* of the candidate models (since we may not know which model is better until all of the training points have been obtained). In addition, the proposed approach selects training points in a batch manner, which allows

to avoid the model drift. These properties allow the proposed approach to compare favorably with other approaches as demonstrated by the numerical experiments.

We have shown the merits of combining AL with another dependent task. It is important to look at the different ways in which the tasks could be combined. In addition, it is necessary to study and analyze the possible interactions between the tasks. Some of the interactions may be negative and should be avoided or suppressed (e.g. model drift in the sequential AL). Other interactions are beneficial and if possible should be incorporated into the new method (e.g. if an *appropriate* model is selected, performing AL in a batch manner could be optimal).

Another approach is to utilize additional data that is related to the task at hand. We concentrated on utilizing the training data from other target functions. Utilizing this data has proved to be beneficial and allowed us to further improve the effectiveness of active learning. However, there a number of issues that need to be considered in order for the active learning task to be effective. The peculiarities of the data need to be taken into account. We have shown that simply applying traditional methods to different settings may be ill-suited. For example, the inability of the traditional AL methods to cope with sparse data was shown to be detrimental to their performance. It may be tempting to utilize additional data resources, but it should be done with caution. Adding additional resource may result in increasing the complexity of the function's model, since more features would need to be considered. A model of the increased complexity may require a larger number training points, which may be counter productive. Therefore when adding new data resource, it should be ensured that adding more features will result in improved accuracy. In addition, at first sight some data may seem irrelevant for the task at hand, but after careful analysis may prove to be rather useful. For example, for the task of learning the user's preferences it may be beneficial to utilize the data from social networks. While this data is not directly related to predicting the preferences of the user, it may allow to identify the people that are in some way related to the user. Related users may have some interests in common and this may allow to further improve the accuracy of the predictions. In conclusion, performing active learning by utilizing additional data is beneficial, but should be performed with proper care.

We have showed that by not treating active learning as an independent problem its effectiveness could be further improved. More importantly the following dissertation raises many new questions e.g.: are there any other task that could be combined, what other resources could be utilized and how? We provide details about possible future directions of current research

in the next chapter, and are planning to address them in the near future.

Chapter 15

Future Work

This dissertation raises a number of new research questions. In the following sections we provide a brief sketch of possible future directions.

15.1 Model Selection with Active Learning

We have shown in Part II that it is possible to perform active learning with model selection. A natural question arises: is it possible to perform model selection with active learning? We may want to to select training data points (perform AL) as to better estimate the appropriate model (MS). The motivations for this method are similar to those of AL/MS. Implementation however may differ significantly.

A hint to how it could be implemented may lie in one of the early works on active learning referred to as Query by Committee (QBC) (Seung et al., 1992). The idea behind QBC is that training points for which the estimates of the candidates models differs the most are selected. The goal of QBC is to select informative training points. However, it may also be viewed as a form of model selection. The candidate models that are no longer capable of reliably predicting the output estimates may need to be discarded from the pool of model candidates.

15.2 Coping with Model Drift

If it was not for the model drift problem, sequential approach appears to be a perfect candidate that may allow to combine AL with MS as well as to combine MS with AL. An interesting approach would be to attempt to estimate drifting of the model. We may create profiles of model drifting for various settings. Then we may compare the current settings and the drifting

behavior to the existing profiles and see which one it matches the best. This may allow us to avoid obtaining training points for simpler models, which may not be as useful for the finally selected model.

15.3 Multiple Objective Active Learning

A common object of AL is to minimize the overall generalization error. However, in some domains (e.g. recommender systems) we may consider slightly different objectives. We may still want to minimize the generalization error, but perhaps only for the items for which high ratings are predicted, since a user may simply disregard any of the items for which the predicted ratings are bellow neutral. Therefore the accuracy of predictions for these items may not be relevant. In addition when an item is presented to a user it could be done due to several objectives: learning user's preferences, tempting a user to buy it, etc. We may consider using a method that may allow to incorporate multiple objectives in to active learning, perhaps Pareto's method (Zitzler & Thiele, 1999) may be suited for this task.

15.4 Similarity Estimation in Collaborative Settings

In our research, we considered linear regression settings, where output values are estimated as a linear combination of the input's features. Where the parameter/weight of each feature is often estimated by the least squares method. Another approach is to estimate output values based on the outputs of similar entities. Where similarity metric measures how similar any given entities are to one another (Balcan, 2006). If the entities are identical, the similarity is maximal. Let us consider memory-based collaborative filtering algorithms (Resnick et al., 1994; Nakamura & Abe, 1998; Delgado & Ishii, 1999; Adomavicius & Tuzhilin, 2005). Ability to identify similar users or items is crucial, since very accurate predictions could be provided by utilizing the elicited preferences of similar entities. Let us briefly sketch the settings of the problem. The rating matrix is denoted as X. The target function is denoted as $f(x)$ and its approximation as $\widehat{f}(x)$. The entry of the matrix $x_{i,j} = f_j(x_i)$, i.e. the rating of the item i by the user j. We may obtain the estimate of the output value as following:

$$\widehat{f}(x_i) = aggr_{f_s \in \hat{S}} f_s(x_i).$$

Where $\hat{S} = \{f_i\}_{i=1}^{m}$ is the set of functions that are most similar to f (as measured by the sim function) and that have realized (not approximated) output value of x_i. The aggregation function $aggr$ is commonly expressed as:

$$f(x_i) = \frac{1}{\sum_{f_s \in \hat{S}} sim(f, f_s)} \sum sim(f, f_s) \times f_s(x).$$

In collaborative filtering similarity sim is often measured by Pearson correlation as:

$$sim(f_t, f_{t'}) = \frac{\sum_{x \in X_{tt'}} (f_t(x) - \bar{f}_t)(f_{t'}(x) - \bar{f}_{t'})}{\sqrt{\sum_{x \in X_{tt'}} (f_t(x) - \bar{f}_t)^2 \sum_{x \in X_{tt'}} (f_{t'}(x) - \bar{f}_{t'})^2}},$$

where $X_{tt'}$ is the set of items that have realizations for both f_t and $f_{t'}$ i.e. $X_{tt'} = \{x \in X | \, x \in X_t, \, x \in X_{t'}\}$.

Estimating similarity in collaborative setting is challenging, since most of the values are missing. Let us assume that there are two users with very similar preferences. However, unless users have evaluated common items their similarity may be undefined. We may consider performing dimensionality reduction of the item dimension first and then we may be able to better estimate the similarity. In addition, since the size of the matrix may be significantly reduced, it may allow improving the computational complexity of similarity approximation.

Let us consider another issue. Assume there are two users that have the same preferences for comedies, but opposite preferences for romantic movies. Their similarity may not be very high, even though they have exactly the same preferences for comedies. When making predictions for a comedy, it would make sense to consider their similarity to be high. We may again find useful to perform dimensionality reduction based on the items dimension. This way when we make a prediction for an item, we may calculate the similarity between the users based on the dimension to which the item belongs (e.g. a comedy dimension (if dimensionality is reduced based on genre)). Using this approach may allow to further improve the accuracy of output estimates.

Acknowledgments

I would like to express my gratitude to Professor Sugiyama for introducing me to the wonderful field of machine learning and his continuous support and guidance. I am grateful to the members of my PhD committee[1]: Professor Akiyama, Professor Murata, Professor Sato, Professor Tokunaga; for their guidance and support. I would like to express my thanks to the members (current and former) of Sugiyama laboratory, my friends and family. I am thankful for the financial support of MEXT, which allowed me to study in Japan.

[1] in alphabetical order

Part V

Appendix

Appendix A

Active Learning with Model Selection

A.1 Proof of Eq.(3.21)

First, we show the consistency (3.21) when $\lambda = 1$. A simple calculation yields that the bias B and the variance V can be expressed as

$$B = \langle \boldsymbol{U}(\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*), \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^* \rangle, \tag{A.1}$$

$$V = \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \langle \boldsymbol{U}(\widehat{\boldsymbol{\alpha}} - \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{\boldsymbol{\alpha}}), \widehat{\boldsymbol{\alpha}} - \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{\boldsymbol{\alpha}} \rangle. \tag{A.2}$$

Let

$$\boldsymbol{z}_g = (g(\boldsymbol{x}_1), g(\boldsymbol{x}_2), \dots g(\boldsymbol{x}_n))^\top, \tag{A.3}$$

$$\boldsymbol{z}_r = (r(\boldsymbol{x}_1), r(\boldsymbol{x}_2), \dots r(\boldsymbol{x}_n))^\top. \tag{A.4}$$

By definition, it holds that

$$\boldsymbol{z}_g = \boldsymbol{X}\boldsymbol{\alpha}^*. \tag{A.5}$$

Then we have

$$\begin{aligned} \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^* &= \boldsymbol{L}(\boldsymbol{z}_g + \delta \boldsymbol{z}_r) - \boldsymbol{\alpha}^* \\ &= (\tfrac{1}{n}\boldsymbol{X}^\top \boldsymbol{D}\boldsymbol{X})^{-1} \tfrac{1}{n} \boldsymbol{X}^\top \boldsymbol{D}(\boldsymbol{X}\boldsymbol{\alpha}^* + \delta \boldsymbol{z}_r) - \boldsymbol{\alpha}^* \\ &= \delta (\tfrac{1}{n}\boldsymbol{X}^\top \boldsymbol{D}\boldsymbol{X})^{-1} \tfrac{1}{n} \boldsymbol{X}^\top \boldsymbol{D}\boldsymbol{z}_r. \end{aligned} \tag{A.6}$$

By the law of large numbers (Rao, 1965), we have

$$\begin{aligned}\lim_{n\to\infty}[\tfrac{1}{n}\boldsymbol{X}^\top\boldsymbol{D}\boldsymbol{X}]_{i,j} &= \lim_{n\to\infty}\left(\frac{1}{n}\sum_{k=1}^{n}\frac{q(\boldsymbol{x}_k)}{p(\boldsymbol{x}_k)}\varphi_i(\boldsymbol{x}_k)\varphi_j(\boldsymbol{x}_k)\right)\\ &= \int_{\mathcal{D}}\frac{q(\boldsymbol{x})}{p(\boldsymbol{x})}\varphi_i(\boldsymbol{x})\varphi_j(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}\\ &= \mathcal{O}_p(1). \end{aligned} \tag{A.7}$$

Furthermore, by the central limit theorem (Rao, 1965), it holds for sufficiently large n,

$$\begin{aligned}[\tfrac{1}{n}\boldsymbol{X}^\top\boldsymbol{D}\boldsymbol{z}_r]_i &= \frac{1}{n}\sum_{k=1}^{n} r(\boldsymbol{x}_k)\varphi_i(\boldsymbol{x}_k)\frac{q(\boldsymbol{x}_k)}{p(\boldsymbol{x}_k)}\\ &= \int_{\mathcal{D}} r(\boldsymbol{x})\varphi_i(\boldsymbol{x})\frac{q(\boldsymbol{x})}{p(\boldsymbol{x})}p(\boldsymbol{x})d\boldsymbol{x} + \mathcal{O}_p(n^{-\frac{1}{2}})\\ &= \mathcal{O}_p(n^{-\frac{1}{2}}), \end{aligned} \tag{A.8}$$

where the last equality follows from Eq.(3.7). Given $\boldsymbol{U} = \mathcal{O}_p(1)$, we have

$$B = \mathcal{O}_p(\delta^2 n^{-1}). \tag{A.9}$$

Since

$$\begin{aligned}\boldsymbol{L}\boldsymbol{L}^\top &= (\tfrac{1}{n}\boldsymbol{X}^\top\boldsymbol{D}\boldsymbol{X})^{-1}\tfrac{1}{n^2}\boldsymbol{X}^\top\boldsymbol{D}^2\boldsymbol{X}(\tfrac{1}{n}\boldsymbol{X}^\top\boldsymbol{D}\boldsymbol{X})^{-1}\\ &= \mathcal{O}_p(n^{-1}), \end{aligned} \tag{A.10}$$

we have

$$\begin{aligned}V &= \sigma^2\mathrm{tr}(\boldsymbol{U}\boldsymbol{L}\boldsymbol{L}^\top)\\ &= \mathcal{O}_p(n^{-1}). \end{aligned} \tag{A.11}$$

Thus, if $\delta = o_p(1)$,

$$\begin{aligned}\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} G &= B + V + \delta^2\\ &= o_p(n^{-1}) + \sigma^2\widehat{G}^{(AL)} + \delta^2, \end{aligned} \tag{A.12}$$

which results in Eq.(3.21). We may establish the same argument if λ is asymptotically one.

A.2 Proof of Eq.(3.25)

We show the asymptotic unbiasedness (3.25). A simple calculation yields that the generalization error G is expressed as

$$G = \langle \boldsymbol{U}\widehat{\boldsymbol{\alpha}}, \widehat{\boldsymbol{\alpha}} \rangle - 2\langle \boldsymbol{U}\widehat{\boldsymbol{\alpha}}, \boldsymbol{\alpha}^* \rangle + C. \tag{A.13}$$

Since

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \langle \boldsymbol{U}\widehat{\boldsymbol{\alpha}}, \boldsymbol{\alpha}^* \rangle = \langle \boldsymbol{U}\boldsymbol{L}(\boldsymbol{z}_g + \delta \boldsymbol{z}_r), \boldsymbol{L}_1 \boldsymbol{z}_g \rangle, \tag{A.14}$$

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \langle \boldsymbol{U}\widehat{\boldsymbol{\alpha}}, \boldsymbol{L}_1 \boldsymbol{y} \rangle = \langle \boldsymbol{U}\boldsymbol{L}(\boldsymbol{z}_g + \delta \boldsymbol{z}_r), \boldsymbol{L}_1(\boldsymbol{z}_g + \delta \boldsymbol{z}_r) \rangle + \sigma^2 \mathrm{tr}(\boldsymbol{U}\boldsymbol{L}\boldsymbol{L}_1^\top), \tag{A.15}$$

we have

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} G - C - \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{G}^{(MS)} = 2\langle \boldsymbol{U}\boldsymbol{L}(\boldsymbol{z}_g + \delta \boldsymbol{z}_r), \delta \boldsymbol{L}_1 \boldsymbol{z}_r \rangle + 2(\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{\sigma^2} - \sigma^2)\mathrm{tr}(\boldsymbol{U}\boldsymbol{L}\boldsymbol{L}_1^\top). \tag{A.16}$$

Eqs.(A.7) and (A.8) imply

$$\boldsymbol{L}_1 \boldsymbol{z}_r = \mathcal{O}_p(n^{-\frac{1}{2}}). \tag{A.17}$$

Thus the first term in the right-hand side of Eq.(A.16) is of $\mathcal{O}_p(\delta n^{-\frac{1}{2}})$. Since

$$\mathrm{tr}(\boldsymbol{U}\boldsymbol{L}\boldsymbol{L}_1^\top) = \mathcal{O}_p(n^{-1}), \tag{A.18}$$

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \widehat{\sigma^2} = \sigma^2 + \frac{\delta^2 \|\boldsymbol{G}\boldsymbol{z}_r\|^2}{\mathrm{tr}(\boldsymbol{G})}, \tag{A.19}$$

where $\boldsymbol{G} = \boldsymbol{I} - \boldsymbol{X}\boldsymbol{L}_0$ and $\boldsymbol{I}$ is the identity matrix, the second term in the right-hand side of Eq.(A.16) is of $\mathcal{O}_p(\delta^2 n^{-1})$. This establishes Eq.(3.25).

Appendix B

Active Learning in Collaborative Settings

B.1 Relation between the A-Optimal and the proposed criterion

Let us look at the relation between the A-Optimal and the proposed criterion. The A-optimal design (Chan, 1981) seeks to minimize the trace of the inverse of the information matrix i.e.:

$$\min tr\mathbf{A}^{-1}. \tag{B.1}$$

Lets us rewrite the above criterion with the respect to selecting which $\mathbf{x}_\delta$ to add to the training set. Let us denote the information matrix $\mathbf{A}$ after $\mathbf{x}_\delta$ is added to the training set as $\mathbf{A}'$:

$$\mathbf{A}' = \mathbf{X}\mathbf{X}^\top + \mathbf{x}_\delta \mathbf{x}_\delta^\top. \tag{B.2}$$

The inverse of the information matrix $\mathbf{A}'$ could be written then as:

$$(\mathbf{A}')^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{x}_\delta \left(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta\right)^{-1} \mathbf{x}_\delta^\top \mathbf{A}^{-1}, \tag{B.3}$$

and the corresponding trace could be written as:

$$\begin{aligned} tr(\mathbf{A}')^{-1} &= tr\mathbf{A}^{-1} - tr\mathbf{A}^{-1}\mathbf{x}_\delta \left(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta\right)^{-1} \mathbf{x}_\delta^\top \mathbf{A}^{-1} \\ &= trA^{-1} - \frac{\mathbf{x}_\delta^\top \mathbf{A}^{-2}\mathbf{x}_\delta}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1}\mathbf{x}_\delta}. \end{aligned}$$

Let us note the objective of A-optimal is to minimize the $tr(\mathbf{A}')^{-1}$. Therefore it would favor $\mathbf{x}_\delta$ with a large value of:

$$\frac{\mathbf{x}_\delta^\top \mathbf{A}^{-2} \mathbf{x}_\delta}{1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta}. \tag{B.4}$$

The proposed criterion is formulated as:

$$J(\delta) = (y_\delta - \mathbf{x}_\delta^\top \widehat{\boldsymbol{\beta}}_t)^2 \frac{\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{X}^{*\top} \mathbf{X}^* \mathbf{A}^{-1} \mathbf{x}_\delta}{(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2}.$$

It in turn favors $\mathbf{x}_\delta$ with a large value of:

$$\frac{\mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{X}^{*\top} \mathbf{X}^* \mathbf{A}^{-1} \mathbf{x}_\delta}{(1 + \mathbf{x}_\delta^\top \mathbf{A}^{-1} \mathbf{x}_\delta)^2}. \tag{B.5}$$

While the exact relation between the terms Eq. (B.4), (B.5) is not clear, there is appears to be a strong similarity between them.

Bibliography

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*, 734–749.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *AC-19*, 716–723.

Allan, J., K. E. (2007). Million query track 2007 overview. *Text REtrieval Conference (TREC)*.

Babbage, C. (1864). *Passages from the life of a philosopher*. Longman and Co.

Bach, F. R. (2007). Active learning for misspecified generalized linear models. In B. Schölkopf, J. Platt and T. Hoffman (Eds.), *Advances in neural information processing systems 19*, 65–72. Cambridge, MA: MIT Press.

Balcan, M.-F., B. A. (2006). On a theory of learning with similarity functions. *International Conference on Machine Learning*.

Boutilier, C., Zemel, R., & Marlin, B. (2003). Active collaborative filtering. *Proceedings of the Nineteenth Annual Conference on Uncertainty in Artificial Intelligence* (pp. 98–106).

Chan, N. (1981). *A-optimality for regression designs* (Technical Report). Stanford University, Department of Statistics.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*, *4*, 129–145.

Craven, P., & Wahba, G. (1979). Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerische Mathematik*, *31*, 377–403.

Dasgupta, S., Lee, W., & Long, P. (2003). A theoretical analysis of query selection for collaborative filtering.

Delgado, J., & Ishii, N. (1999). Memory-based weighted-majority prediction for recommender systems. *Proc. ACM SIGIR 99 Workshop Recommender Systems: Algorithms and Evaluation.*

Dette, H., & Studden, W. J. (1993). Geometry of e-optimality. *Annals of Statistics*, *21*, 416–43.

Fedorov, V. V. (1972). *Theory of optimal experiments.* New York: Academic Press.

Fukumizu, K. (2000). Statistical active learning in multilayer perceptrons. *IEEE Transactions on Neural Networks*, *11*, 17–26.

Goldberg, K., Roeder, T., Gupta, D., & Perkins, C. (2001). Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, *4*, 133–151.

GroupLens, University of Minnesota (2005). Movielens data set. http://movielens.umn.edu.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The elements of statistical learning: Data mining, inference, and prediction.* New York: Springer.

Henkel, R. E. (1979). *Tests of significance.* Beverly Hills: SAGE Publication.

John, R. C. S., & Draper, N. R. (1975). D-optimality for regression designs: A review. *Technometrics*, *17*, 15–23.

Kanamori, T., & Shimodaira, H. (2003). Active learning algorithm using the maximum weighted log-likelihood estimator. *Journal of Statistical Planning and Inference*, *116*, 149–162.

Kohrs, A., & Merialdo, B. (2001). Improving collaborative filtering for new users by smart object selection. *Proceedings of International Conference on Media Features (ICMF).*

L. Domokos, C. J., & Wittig, G. (1986). Data in beilstein-online. *Microchimica Acta.*

MacKay, D. J. C. (1992a). Bayesian interpolation. *Neural Computation*, *4*, 415–447.

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

Mattingly CJ, Colby GT, R. M. F. J. B. J. (2004). Promoting comparative molecular studies in environmental health research: an overview of the comparative toxicogenomics database (ctd). *The Pharmacogenomics Journal*.

Nakamura, A., & Abe, N. (1998). Collaborative filtering using weighted majority prediction algorithms. *Proc. 15th Intl Conf. Machine Learning*.

Nakamura, A., Kudo, M., & Tanaka, A. (2003). Collaborative filtering using restoration operators. *Proceedings of Principles and Practice of Knowledge Discovery in Databases, PKDD* (pp. 339–349). Springer.

NetFlix Inc. (2007). Netflix data set. http://www.netflixprize.com.

Rao, C. R. (1965). *Linear statistical inference and its applications*. New York: Wiley.

Rashid, A. M., Albert, I., Cosley, D., Lam, S. K., McNee, S. M., Konstan, J. A., & Riedl, J. (2002). Getting to know you: learning new user preferences in recommender systems. *IUI '02: Proceedings of the 7th international conference on Intelligent user interfaces* (pp. 127–134). New York, NY, USA: ACM Press.

Rasmussen, C. E., Neal, R. M., Hinton, G. E., van Camp, D., Revow, M., Ghahramani, Z., Kustra, R., & Tibshirani, R. (1996). The DELVE manual.

Resnick, P., Iacovou, N., Suchak, M., Bergstorm, P., & Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work* (pp. 175–186). Chapel Hill, North Carolina: ACM.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*, 465–471.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*, 461–464.

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. *Proceedings of the Fifth Workshop on Computational Learning Theory* (pp. 287–294). Morgan Kaufmann.

Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, *90*, 227–244.

Sugiyama, M. (2006). Active learning in approximately linear regression based on conditional expectation of generalization error. *Journal of Machine Learning Research*, *7*, 141–166.

Sugiyama, M., Krauledat, M., & Müller, K.-R. (2007). Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, *8*, 985–1005.

Sugiyama, M., & Müller, K.-R. (2005). Input-dependent estimation of generalization error under covariate shift. *Statistics & Decisions*, *23*, 249–279.

Sugiyama, M., & Ogawa, H. (2000). Incremental active learning for optimal generalization. *Neural Computation*, *12*, 2909–2940.

Sugiyama, M., & Ogawa, H. (2003). Active learning with model selection — Simultaneous optimization of sample points and models for trigonometric polynomial models. *IEICE Transactions on Information and Systems*, *E86-D*, 2753–2763.

Vapnik, V. N. (1998). *Statistical learning theory.* New York: Wiley.

Watanabe, S. (2001). Algebraic analysis for nonidentifiable learning machines. *Neural Computation*, *13*, 899–933.

White, H. (1982). Maximum likelihood estimation of misspecified models. *Econometrica*, *50*, 1–25.

Wiens, D. P. (2000). Robust weights and designs for biased regression models: Least squares and generalized M-estimation. *Journal of Statistical Planning and Inference*, *83*, 395–412.

Woodbury, M. A. (1950). *Inverting modified matrices* (Technical Report). Statistical Research Group, Princeton University.

Yu, K., Bi, J., & Tresp, V. (2006). Active learning via transductive experimental design. *Proceedings of the 23rd Int. Conference on Machine Learning ICML '06* (pp. 1081–1088). New York, NY, USA: ACM.

Zadrozny, B. (2004). Learning and evaluating classifiers under sample selection bias. *Proceedings of the Twenty-First International Conference on Machine Learning.* New York, NY: ACM Press.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE-EC*, *3*, 257.