Bias-Variance Analysis for Ensembling Regularized Multiple Criteria Linear Programming Models^{*}

Peng Zhang¹, Xingquan Zhu², and Yong Shi^{1,3}

¹ FEDS Research Center, Chinese Academy of Sciences, Beijing, 100190, China

² Dep. of Computer Sci. & Eng., Florida Atlantic University, Boca Raton, FL, 33431, USA

³ College of Inform. Science & Technology, Univ. of Nebraska at Omaha, Nebraska, USA
zhangpeng04@gmail.com, xzhu3@fau.edu, yshi@gucas.ac.cn

Abstract. Regularized Multiple Criteria Linear Programming (RMCLP) models have recently shown to be effective for data classification. While the models are becoming increasingly important for data mining community, very little work has been done in systematically investigating RMCLP models from common machine learners' perspectives. The missing of such theoretical components leaves important questions like whether RMCLP is a strong and stable learner unable to be answered in practice. In this paper, we carry out a systematic investigation on RMCLP by using a well-known statistical analysis approach, bias-variance decomposition. We decompose RMCLP's error into three parts: bias error, variance error and noise error. Our experiments and observations conclude that RMCLP's error mainly comes from its bias error, whereas its variance error remains relatively low. Our observation asserts that RMCLP is stable but not strong. Consequently, employing boosting based ensembling mechanism RMCLP will mostly further improve the RMCLP models to a large extent.

Keywords: Bias variance decomposition, RMCLP, Ensemble.

1 Introduction

Due to the strong application-driven nature, the fields of data mining and optimization are becoming heavily intermingled than ever [1]. Experts used to traditionally work in optimization have not gradually shifted to the data mining field, or vice versa. In 2001, Shi et al. [2] proposed a multiple criteria linear programming (MCLP) model for classification. The promising results of MCLP motivate widely applications in business intelligence. Most recently, based on MCLP, a much more sophisticated model called Regularized Multiple criteria Linear Programming (RMCLP) is proposed [3] and reported exciting results on some UCI benchmark datasets. On the other hand, empirical studies on some other UCI benchmark datasets also show significant drop of accuracy of RMLCP. These observations motivate us to step further to look into

^{*} This research has been partially supported by a grant from National Natural Science Foundation of China (#90718042, #70621001, #70531040, #70501030, #10601064, #70472074, #60674109), National Natural Science Foundation of Beijing #9073020, 973 Project #2004CB720103, Ministry of Science and Technology, China and BHP Billiton Co., Australia.

the inherent characteristics of RMCLP in three aspects: (1) whether RMCLP is a strong classifier which is supposed to achieve low prediction errors on any testing sets; (2) whether RMCLP is a stable classifier with very little fluctuation of its performance; and (3) which ensemble method is eligible to enhance RMCLP's performance.

Bias-variance analysis is a powerful tool to study learning algorithms [4,5,6,7], mainly because that bias-variance decomposition can provide first hand evidence on the stability and the weakness of the classifiers built from the learning algorithms. To a specific learning algorithm $\mathcal{L}$, bias-variance analysis decomposes its error into three parts: bias error, variance error and noise error. Bias error comes from the inherent shortage of learners. For example, a linear learner will have a high bias error on a non-linear dataset. Variance error is related to the learner's stability on different samples. When the sample is changed, the learner probably will generate a different model which has different prediction accuracy. This shifting of performance with the changing of sample is called the variance error. Noise error is assumed to exist in every sample. For example, two examples may have the same attribute values while having different labels, and this discrepancy generates the noise error. A strong learner is supposed to have low bias error and variance error. For a given variance error, a strong learner is more likely to have a low bias error. On the other hand, a stable classifier is supposed to have low variance error, which means the learner fluctuates very little on different training samples. Besides applications in studying learning algorithms, bias variance decomposition also provides a rationale to develop ensemble methods [7]. If a learner is a weak learner (the opposite side of strong learner), boosting (a well-known ensemble mechanics) can be used to transform the weak learner into a strong learner by combing models trained on weighted training instances [8]. In contrast, if a learner is an unstable one, bagging (another well-known ensemble mechanics) can be used to transform it into a stable learner by equally combining each base learner together [9].

In this paper, to investigate why and how RMCLP works, we decompose RMCLP model's error by bias variance decomposition. Moreover, by observing where the error mainly comes from, we can develop an appropriate ensemble method to enhance RMCLP performance. For instance, if bias error accounts for the heaviest part of the entire error while variance error is low, we can assert that RMCLP is a stable but weak classifier, and boosting method can be used to improve its performance. On the other hand, if variance error takes the heaviest part of the entire error while bias error is low, we can say that RMCLP is a strong but unstable classifier, and under this circumstance, bagging can be exploited as the ensemble method.

The rest of this paper is organized as follows: in the next section, we give a short introduction of RMCLP model. In the third section, we introduce the bias variance decomposition method. In the fourth section, we introduce the ensemble methods and describe the bagging and adaboosting algorithms. In the fifth section, we carry out bias variance decomposition of RMCLP on synthetic datasets. In the last section, we finish our paper with several conclusions.

2 Regularized Multiple Criteria Linear Programming (RMCLP)

In this section, we will introduce the two groups RMCLP model. Since RMCLP model derives from the original MCLP model, we will introduce MCLP model first.

2.1 Multiple Criteria Linear Programming (MCLP) Model

MCLP model [11] is originally introduced for linearly separating two-group data sets. Assume a two-group data set A which has n instances $A = \{A_1, A_2, \dots, A_n\}$, we define a boundary vector b to distinguish the first group G_1 and the second group G_2 by following the rules that, if an example $A_i \in G_1$, then $A_i x < b$; otherwise, $A_i x \geq b$. To formulate the criteria functions and complete constraints for data separation, some other variables need to be introduced. We define external measurement α_i to be the overlapping distance between boundary b and a training instance, say A_i . When A_i is wrongly classified, α_i will be equal to $|A_i x - b|$. We also define internal measurement β_i to be the distance of A_i from its adjusted boundary b^* . When A_i is correctly classified, distance β_i will equal to $|A_i x - b^*|$, where $b^* = b + \alpha_i$ or $b^* = b - \alpha_i$. To separate the two groups as far as possible, we design two objective functions which minimize the overlapping distances and maximize the distances between classes. Suppose $\|\alpha\|_p^p$ denotes for the relationship of all overlapping α_i while $\|\beta\|_q^q$ denotes for the aggregation of all distances β_i . The final correctly classified instances is depended on simultaneously minimize $\|\alpha\|_p^p$ and maximize $\|\beta\|_q^q$. By choosing $p=q=1$, we get the linear combination of these two objective functions as follows:

$$(MCLP) \quad \text{Minimize} \quad w_\alpha \sum_{i=1}^n \alpha_i - w_\beta \sum_{i=1}^n \beta_i \quad (1)$$

Subject to:

$$A_i x - \alpha_i + \beta_i - b = 0, \forall A_i \in G_1$$

$$A_i x + \alpha_i - \beta_i - b = 0, \forall A_i \in G_2$$

where A_i is given, x and b are unrestricted, α_i and $\beta_i \geq 0$.

2.2 RMCLP Model

A lot of empirical studies have shown that MCLP is a powerful tool for classification. However, there is no theoretical work on whether MCLP always can find an optimal solution under different kinds of training samples. To go over this difficulty, recently, Shi et.al [3] proposed a RMCLP model by adding two regularized items $x^T H x / 2$ and $\alpha^T Q \alpha / 2$ on MCLP as follows:

$$\text{Minimize} \quad \frac{1}{2} x^T H x + \frac{1}{2} \alpha^T Q \alpha + d^T \alpha - c^T \beta \quad (2)$$

Subject to:

$$A_i x - \alpha_i + \beta_i = b, \forall A_i \in G_1;$$

$$A_i x + \alpha_i - \beta_i = b, \forall A_i \in G_2;$$

$$\alpha_i, \beta_i \geq 0.$$

where $H \in R^{r \times r}$, $Q \in R^{n \times n}$ are symmetric positive definite matrices. $d^T, c^T \in R^n$. The RMCLP model is a convex quadratic program. Theoretically studies [3] have shown that RMCLP can always find a global optimal solution. The algorithm of RMCLP is shown in Algorithm 1.

Algorithm 1. Building RMCLP model

Input: The data set $X = \{x_1, x_2, \dots, x_n\}$, training percentage p

Output: RMCLP model (w, b)

Begin

1. Randomly select $p \times |x|$ instances as the training set **TR**, the remained instances are combined as the testing set **TS**;
2. Choose appropriate parameters of (H, D, d, c) ;
3. Apply the MCLP model (1) to compute the optimal solution $W^* = (w_1, w_2, \dots, w_n)$ as the direction of the classification boundary;
4. Output $y = \text{sgn}(wx - b)$ as the model.

End

3 Bias Variance Decomposition

Notations. Consider a two groups classification problem with 0/1 loss function. Assume there is a set $D = \{D_1, D_2, \dots, D_n\}$ of learning sets $D_j = \{(x_1, t_1), \dots, (x_m, t_m)\}$, with $t_j \in C = \{-1, 1\}$, and $x_j \in X$ has d attributes. The estimates of all the errors are performed on a test set T separated from the training set D . The learning algorithm is denoted by L . A classifier f_i built on D_i using L is denoted as $f_i = L(D_i)$. The predicted label of instance x_k by f_i is denoted as $y_k = f_i(x_k)$.

Measuring Bias and Variance. Bias-variance analysis provides a powerful tool to study learning algorithms and can be used to properly design ensemble methods well tuned to the properties of a specific base learner. Historically, the bias-variance analysis was borrowed from regression problems where squared-loss is often used [4]. Recently, Domingo proposed a unified framework [6] of bias-variance decomposition of error which can be used for an arbitrary loss function. According to Domingo's decomposition, the expected loss of a learner $\mathcal{L}$ on a test instance x can be decomposed as a noise error $N(x)$, a bias error $B(x)$ and a variance error $V(x)$ as follow,

$$E[L(\mathcal{L}, x)] = c_1 N(x) + B(x) + c_2 V(x) \quad (3)$$

where c_1 and c_2 are coefficients decided by the loss function. A good classifier should have low bias, in which case the expected loss will approximately equal the variance. To calculate (4), we give the definitions of *optimal prediction* and *main prediction* first. In presence of noise, instance x_i and instance x_j ($i \neq j$) may share the same attribute values but having different labels. We define the *optimal prediction* y^* on instance x to be the most frequent observed label t as

$$y^* = \operatorname{argmax}_{t \in C} p(t | x). \quad (4)$$

We then define the *main prediction* y^m to be the most frequent class label that each base learner gives to x as follow,

$$y^m = \operatorname{argmax}(p(c_1 | x), p(c_2 | x)). \quad (5)$$

By defining y^* and y^m , we can calculate the noise error of instance x by counting the number of discrepancies between class label t and y^* in (7),

$$N(x) = \sum_{t \in C} \mathbb{I}(t \neq y^*) p(t | x), \quad (6)$$

where $\mathbb{I}(z) = 1$ if z is true, otherwise 0. Then the bias error can be calculated by

$$B(x) = \left| \frac{y_m - t}{2} \right| = \begin{cases} 1, & y^m \neq y^* \\ 0, & y^m = y^* \end{cases} \quad (7)$$

The variance error is the discrepancies between each base classifier's prediction and the main prediction, which can be denoted as

$$V(x) = \frac{1}{s} \sum_{i=1}^s \|f(x_i) - y^m\| \quad (8)$$

where s is the number of base classifiers. Then the average bias, variance, and noise errors over the entire set of the examples in the test set $\mathcal{T}$ can be denoted as :

$$E_x[N(x)] = \frac{1}{n} \sum_{i=1}^n N(x_i) = \frac{1}{n} \sum_{i=1}^n \sum_t \mathbb{I}(t_{x_i} \neq y^*) p(t | x_i) \quad (9)$$

$$E_x[B(x)] = \frac{1}{n} \sum_{i=1}^n B(x_i) = \frac{1}{n} \sum_{i=1}^n \left| \frac{y^m - t_i}{2} \right|, \quad (10)$$

and

$$E_x[V(x)] = \frac{1}{n} \sum_{i=1}^n V(x_i) = \frac{1}{ns} \sum_{j=1}^s \sum_{i=1}^n \mathbb{I}(y^m \neq f(x_i)). \quad (11)$$

where n is the number of examples in test set $\mathcal{T}$. Finally, the average loss on all the examples is the algebraic sum of the average bias, variance and noise errors as follow,

$$E_x[L(t, y)] = E_x[N(x)] + E_x[B(x)] + E_x[V(x)]. \quad (12)$$

4 Ensemble Methods

Ensemble of classifiers is one of the main research topics in machine learning. Empirical studies have shown that ensemble are often much more accurate than the individual base learner that makes them up in classification and regression problems. Two typical methods of ensemble strategy is bagging and boosting. Bagging can improve the performance of a learner by reducing its variance error, thus unstable learner such as C4.5 can achieve better prediction accuracy. In contrast, boosting can be used to

enhance the performance of a weak learner which is slightly better than random guess. By using boosting, a weak classifier can be lifted into a strong classifier. As we discussed above, bias variance decomposition offers a rationale to develop ensemble methods for RMCLP. If RMCLP is an unstable classifier with its error mainly coming from its variance, bagging can be used to transform RMCLP to be a stable classifier. On the other hand, if RMCLP is a weak classifier with its error mainly coming from its bias, boosting method can be used to enhance it into a strong classifier. Algorithm 2 shows the algorithm of Bagging and Algorithm 3 exhibits the Adaboosting algorithm (a refined version of boosting) [10].

Algorithm 2. Bagging Algorithm

Input: an unstable learner L , number of bags b ,
training set $Tr = \{(x_1, y_1), \dots, (x_n, y_n)\}$

Output: a stable learner $f(x)$

Begin

1. for $i = 1 \dots b$

Randomly select n samples from X with replacement to form a bag B_i ;

Build a classifier using L on B_i to get classifier $C_i = L(B_i)$

end for

2. Combine the n base classifiers C_i to form an ensemble classifier:

$$f(x) = \sum_{i=1}^n C_i(x)$$

3. Output $f(x)$.

End

Algorithm 3. AdaBoosting Algorithm

Input: a weak Learner L , maximal iteration number T ,
training set $Tr = \{(x_1, y_1), \dots, (x_n, y_n)\}$

Output: a strong learner $f(x)$

Begin

1. Initialize distribution of Tr by $D_1(i) = 1/N$;

2. for $t = 1 \dots T$

2.1 Train L using D_t and get $f_t = L(D_t)$

2.2 Calculate w_t

2.3 update distribution by $D_{t+1}(i) = \frac{D_t(i) e^{-y_i w_t f_t(x_i)}}{Z_t}$, where Z_t is a normalizer.

end for

3. Output the ensemble learner (strong learner): $f(x) = \text{sgn}(\sum_{i=1}^T w_i f_i(x))$

End

5 Bias Variance Decomposition to Develop Ensemble Method for RMCLP

We generate four synthetic datasets with levels of noise 0%, 5%, 10%, 15% respectively. Each dataset is a 2-dimensional 2-class problem with 600 instances, 300 for each class. 80% instances will be used for training and the remained 20% will be used as testing. All of the generated instances comply with the Gaussian distribution $x \sim N(\mu, \Sigma)$, where μ is the mean vector and Σ is the covariate matrix. The classification boundary is defined as a linear boundary $\|x\|=1$. Instances that satisfy $\|x\|<1$ will be assigned the class label “-1” while agree with $\|x\|\geq 1$ will be assigned the class label “+1”. To simulate noisy instances, we randomly pick up 0.0%, 0.25%, 5%, 7.5% instances from each class and then assign them the opposite class label. In the following experiments, we will do bias-variance decomposition on these four datasets to investigate whether RMCLP is a stable and strong classifier.

Table 1 reports the 10-folder cross validation of bias variance results on the synthetic datasets. The first column lists three measurements, average error, bias error and variance error respectively. The second column shows the different parameters of RMCLP. The 3th to 6th column lists the experiment results under different training samples. In this experiments, we will do the following observations: (1) by using different parameters of (H, D, d, c) , we examine whether RMCLP varies significantly different on different parameters; (2) by comparing the bias and variance errors, we get a conclusion where the error mainly comes from, and thus we can define RMCLP as a strong or weak classifier, a stable or unstable classifier. However, we don't report the noise error $N(x)$ here because noises error is independent of classifiers but only related to the nature of dataset. Additionally, we give approximate values to bias error by subtract variance error from the average error. This approximation is reasonable because it won't change their orders when compared. All of the numeric results are listed in Table 1.

From the results of average errors, we can observe that the error rates have slight changes when the parameters keep changing. On the 0% noise dataset, only when $H>D$, the error goes up from 0.003 to 0.005. On the other three datasets, these similar tendencies have been observed. Thus we can say that RMCLP is slightly impacted by different parameters.

From the variance error, we can observe that compared to the average errors and bias errors, the variance errors take a low percentage of the whole error. For example, on 0% noise dataset, most of the variance errors stay 0, that is to say, no changes have been observed. On the other three datasets, the variance error are almost 10^{-1} lower than the bias error. Thus it is safe to say that RMCLP is a stable classifier, and bagging will help little on RMCLP model.

From the bias error, we can see that on the four datasets, bias errors take part of most of the average errors. For example, on the 10% noise dataset, when $H=D$, $d=c$, the average error is 0.091, while the bias errors is 0.084, which takes up 92.3% of the whole error. Thus it is also safe to say that RMCLP is not a strong classifier, and boosting can be used to enhance RMCLP into a strong classifier.

Table 1. Results of Bias variance analysis

Measures	parameters	0%	5%	10%	15%
Avg. Error	H=D, d=c	0.003	0.050	0.091	0.116
	H=D, d>c	0.003	0.050	0.091	0.117
	H=D, d<c	0.003	0.050	0.090	0.116
	H>D, d=c	0.005	0.047	0.087	0.106
	H<D, d=c	0.003	0.048	0.085	0.116
Bias	H=D, d=c	0.003	0.042	0.084	0.107
	H=D, d>c	0.003	0.041	0.084	0.108
	H=D, d<c	0.003	0.042	0.085	0.108
	H>D, d=c	0.004	0.046	0.081	0.094
	H<D, d=c	0.003	0.045	0.082	0.107
Variance	H=D, d=c	0.000	0.008	0.007	0.009
	H=D, d>c	0.000	0.009	0.007	0.009
	H=D, d<c	0.000	0.008	0.005	0.008
	H>D, d=c	0.001	0.001	0.006	0.012
	H<D, d=c	0.000	0.003	0.003	0.009

6 Conclusions

In this paper, we applied bias-variance decomposition to RMCLP learning algorithms for the purposes of gaining a thorough understanding of RMCLP's stability and weakness for classification. Our theoretical and empirical studies have concluded that : (1) The parameter setting (H, D, d, c) has very limited impact on the performance of the RMCLP models; (2) The variance of the RMCLP models accounts for only a small part of the total testing error, which indicates that RMCLP is a stable classifier; (3) The testing error of RMCLP mostly comes from its bias, which suggests that RMCLP doesn't belong to the strong learner category; (4) Bagging has a limited effect on RMCLP as it is a stable learner; (5) By using boosting method (especially adaboosting), we can transform RMCLP into a strong classifier. In the future, we will use this bias variance decomposition to test RMCLP model on UCI benchmark datasets to further validate our conclusions.

References

1. Bennett, K.P., Parrado-Hernandez, E.: The Interplay of Optimization and Machine Learning Research. *Journal of Machine Learning Research* 7, 1265–1281 (2006)
2. Shi, Y., Wise, M., Luo, M., Lin, Y.: Data mining in credit card portfolio management: a multiple criteria decision making approach. *Multiple Criteria Decision Making in the New Millennium*, 427–436 (2001)
3. Shi, Y., Tian, Y., Chen, X., Zhang, P.: A Regularized Multiple Criteria Linear Program for Classification. In: *ICDM Workshops 2007*, pp. 253–258 (2007)
4. Kong, E.B., Dietterich, T.G.: Error-correcting output coding corrects bias and variance. In: *Proc. of the 12th ICML Conference* (1996)
5. Domingos, P.: A Unified Bias-Variance Decomposition and its Applications. In: *Proc. of the seventeenth International Conference on Machine Learning*, Stanford, CA, pp. 231–238. Morgan Kaufmann, San Francisco (2000)

6. Domingos, P.: A Unified Bias-Variance Decomposition for Zero-One and Squared Loss. In: Proc. of the Seventeenth National Conference on Artificial Intelligence, Austin, TX. AAAI Press, Menlo Park (2000)
7. Valentini, G., Dietterich, T.G.: Bias-Variance Analysis of Support Vector Machines for the Development of SVM-Based Ensemble Methods. *Journal of Machine Learning Research* 5, 725–775 (2004)
8. Breiman, L.: Bagging predictors. *Journal of Machine Learning* 24, 123–140 (1996)
9. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55(1), 119–139 (1997)
10. Polikar, R.: Ensemble Based Systems in Decision Making. *IEEE Circuits and Systems Magazine* 6(3), 21–45 (2006)
11. Peng, Y., Kou, G., Chen, Z., Shi, Y.: Cross-Validation and Ensemble Analyses on Multiple-Criteria Linear Programming Classification for Credit Cardholder Behavior. In: International Conference on Computational Science, pp. 931–939 (2004)

Appendix

In this appendix we give the deduction of the original bias variance decomposition in regression problems with squared-loss function.

Assume the genuine function we want to approximate is $f = f(x)$. There is a training sample $D = \{(x_i, t_i)\}_{i=1}^N$, where $x_i \in X$, for instance $X = \mathbb{R}^d$, $d \in N$ and $t_i \in T$ with $T = \mathbb{R}$. Besides, D contains a noise level ε with expectation of 0, that is to say, $t = f + \varepsilon$ and $E(\varepsilon) = 0$. A learning function $g = g(w, x)$ is used to approximate the genuine function f , where w is the parameters of g and x is the training sample. To a specific x_i , g gives a corresponding value $y_i = g(w, x_i)$. Thus, the mean squared loss of g on the whole training dataset D is

$$MSE = \frac{1}{|D|} \int_{x \in D} L(g, x) dx = \frac{1}{N} \sum_{i=1}^N (t_i - y_i)^2.$$

To investigate whether g is a good regression model on D , we calculate the expectation of the mean squared loss by

$$E\{MSE\} = E\left\{\frac{1}{N} \sum_{i=1}^N (t_i - y_i)^2\right\} = \frac{1}{N} E\left\{\sum_{i=1}^N (t_i - y_i)^2\right\}. \quad (13)$$

To get $E\{MSE\}$, we calculate $E[(t_i - y_i)^2]$ as follows:

$$\begin{aligned} E[(t_i - y_i)^2] &= E[(t_i - f_i + f_i - y_i)^2] \\ &= E[(t_i - f_i)^2 + (f_i - y_i)^2 + 2(t_i - f_i)(f_i - y_i)] \\ &= E(t_i - f_i)^2 + E(f_i - y_i)^2 + 2E[(t_i - f_i)(f_i - y_i)] \\ &= E(\varepsilon_i^2) + E(f_i - y_i)^2 + 2\left[E(t_i f_i) - E(t_i y_i) - E(f_i^2) + E(f_i y_i)\right] \\ &= E(\varepsilon_i^2) + E(f_i - y_i)^2 + 2\left[E(t_i f_i) - E(t_i y_i) - E(f_i^2) + E(f_i y_i)\right] \end{aligned} \quad (14)$$

As we discussed above, $t = f + \varepsilon$ and $E(\varepsilon) = 0$, thus we have the following two equations:

$$E(t_i f_i) = E(f_i^2 + \varepsilon_i f_i) = E(f_i^2), \quad (15)$$

$$E(t_i y_i) = E(f_i y_i + \varepsilon_i y_i) = E(f_i y_i). \quad (16)$$

Meanwhile

$$\begin{aligned} E(f_i - y_i)^2 &= E(f_i - E(y_i) + E(y_i) - y_i)^2 \\ &= E(f_i - E(y_i))^2 + E(E(y_i) - y_i)^2 + 2E((f_i - E(y_i))(E(y_i) - y_i)) \\ &= E(f_i - E(y_i))^2 + E(E(y_i) - y_i)^2 + 2(E(f_i)E(y_i) - E(f_i y_i) - E^2(y_i) + E^2(y_i)) \\ &= E(f_i - E(f_i))^2 + E(E(f_i) - y_i)^2 \end{aligned} \quad (17)$$

Consequently, by combining (14), (15), (16) and (17), we can get the result of (13) as follows:

$$\begin{aligned} E\{MSE\} &\propto E[(t_i - y_i)^2] \\ &= E(\varepsilon_i^2) + E(f_i - y_i)^2 \\ &= \underbrace{E(\varepsilon_i^2)}_{noise} + \underbrace{E(f_i - E(y_i))^2}_{Bias^2} + \underbrace{E(E(y_i) - y_i)^2}_{Variance} \end{aligned}$$

We can see that under this decomposition, the average error of a learner g is composed of its noise, bias and variance errors. Noise error is the inherent characteristics of the training sample that we can do nothing to improve. Bias error is the error between the target function f and the average output of learning algorithm g . Variance error is the error between the individual output of g and the average output of g .