NYU-CS-6923-Final-Exam

Minimizing Errors

Jiannliang Tsay (jlt245)

Theory

Learning By Definition

A machine learning algorithm is an algorithm that is capable of learning from data. But what exactly do we mean here "**learning**"? Mitchell [1] has provided a succinct definition:

> ***Definition***: A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

In this report, what we are interested is "the performance measure P" - to evaluate the capabilities of a machine learning algorithm through the design of quantitative measurement of its performance.

The central challenge in machine learning is that our algorithm must perform well on "new", "unseen" data besides those training data which our model was trained from. Although the choice of performance measure may seem straightforward and objective, it is often difficult to choose a performance measure that corresponds well to the desired behavior of the model. It may also involve the trade-off process that we try to optimize the procedure of learning process.

MSE – mean squared error

The goal of the machine learning algorithm to solve a task, as previously described, not only performs well on the training set but also performs well on generalization. The **generalization** means the ability to perform well on future, unobserved data.

The field of statistics has given us many tools to achieve such goal, to generalize a learning process or algorithm. A most commonly-used measure is called **mean squared error (MSE)**:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))$$

where $\hat{f}(x_i)$ is the prediction that $\hat{f}$ gives for the *i*th observation and y_i is the true response for the *i*th observation. [8]

As seen, the MSE will be small if the predicted responses are very close to the true responses, and will be large if for some observations, the predicted and true responses differ substantially.

The above MSE formula is computed using the training data for fitting the model, i.e. the Training MSE. But in general, we may have test MSE over the test data. What we are interested in the accuracy of the predictions is minimizing the test MSE.

In practice, the computation of the training MSE is relatively easy, but estimating test MSE is somehow more difficult as there may be the case of no test data available at all.

Bias, Variance and the bias-variance Tradeoff

In statistics, the **bias** (or **bias function**) of an estimator is the difference between this estimator's expected value and the true value of the parameter being estimated. Therefore, bias leads to an under or overestimate of the true value. An estimator or decision rule with zero bias is called **unbiased**. Otherwise the estimator is said to be **biased**. [12]

Assume our predicted model is $\hat{f}(X)$ for any inputs X. We want to predict a new data $X = x_0$ and estimate the expected performance measure $MSE(x_0)$:

$MSE(x_0) = E[(y_0 - \hat{f}(x_0))^2]$

$= E[((y_0 - E[\hat{f}(x_0)]) + (E[\hat{f}(x_0)] - \hat{f}(x_0)))^2]$

$= E[(y_0 - E[\hat{f}(x_0)])^2 + 2(E[\hat{f}(x_0)] - \hat{f}(x_0))(y_0 - E[\hat{f}(x_0)])$

$+ (E[\hat{f}(x_0)] - \hat{f}(x_0))^2]$

$= E[(E[\hat{f}(x_0)] - \hat{f}(x_0))^2] + (y_0 - E[\hat{f}(x_0)])^2$

$= \boldsymbol{Var}[\hat{f}(x_0)] + \boldsymbol{Bias}^2(\hat{f}(x_0))$

The **variance** of a random variable X is the expected value of the squared deviation from the mean of X, u = E[X] such that: Var(X) = E [$(X - u)^2$]. [13]

Here u is $E\left[\hat{f}(x_0)\right]$ in our case.

The bias of $\hat{f}(x_0)$ is $y_0 - E[\hat{f}(x_0)]$ that the difference between expected value and true value.

In short,

$$\text{MSE} = \text{variance} + \text{bias}^2$$

This is called the **bias-variance tradeoff**. What it means is that it might be wise to use a biased estimator to minimize the mean squared error by reducing the variance. [3]

In practice, bias includes "measurement bias", "sampling bias" and "estimation bias". Measurement bias is mainly due to faulty measuring procedures and does not go away with sampling effort. Sampling bias is due to unrepresentative sampling of the target population. It does not go away either with sampling effort. What interests us is the **estimation bias** that refers to an estimation method for which the average of repeated estimates deviates from the true value. Thus, estimation bias is due to the estimator itself being biased and it should go down with increasing sampling effort. This feature is what we desire for an estimator if we can tune down the estimation bias.

Precision

Precision is defined as the opposite of variance, referring to the absence of random error which is due to **measurement error**, **sample variation** and **estimation variance**. Therefore, **precision** is a measure of the statistical variance of an estimation procedure or in sampling cases, the data spread due to the statistical variability present in the sample. In the measurement error case, precision arises from the variance produced by the measurement procedure.

The most common precision measure is the variance as described before. Other precision measures are like standard deviation, range etc.

Accuracy

Bias and precision combine to define the performance of an estimator. The more biased and the less precise an estimator is, the worse its overall ability to make an accurate estimation. **Accuracy** is thus defined as the overall distance between the estimated, observed values and the true value.

One of the common accuracy measure is the MSE. This is the major point of this report. I will explore the possibilities of improving the accuracy through each algorithm.

Regression

In this report, I would like to illustrate the idea of bias-variance tradeoff from regression first. It is easier to create the over-fitting situations.

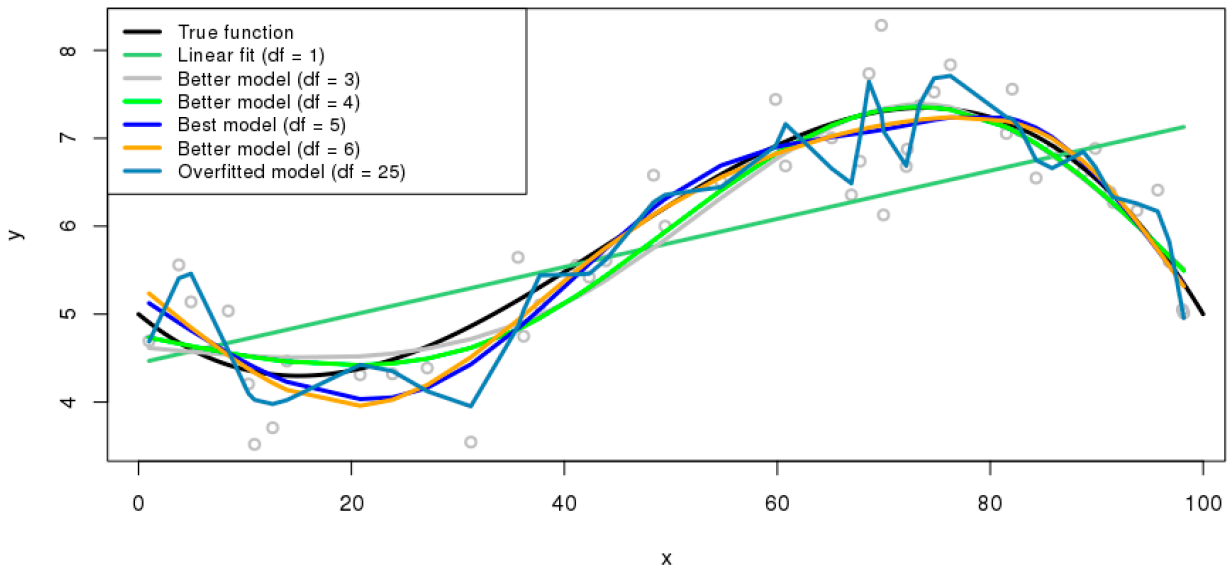
Linear Model Regression

R package spline contains lm () that can be used to create the situation of over-fitting. In this experiment, I will do the over-fitting and inspect the changes of bias and variance.

R script

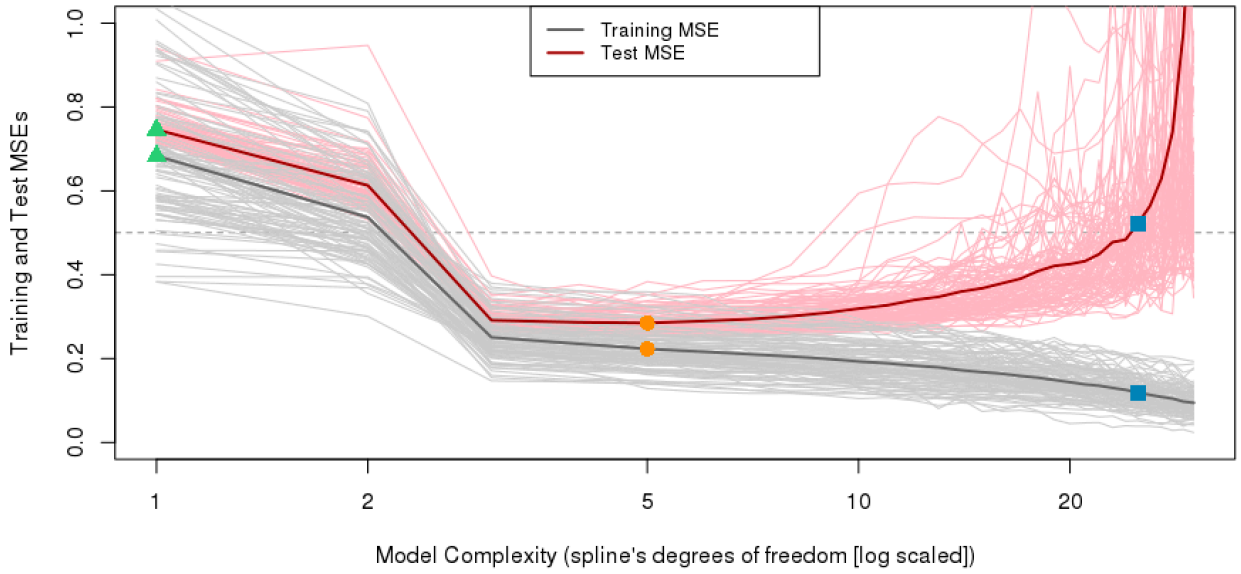
Run jlt_245_final_exam_lm_regression. R

Fig. 1 The linear model fitting curves v.s. the true function (black)



The figure 1 shows the best fit and over-fitted models.

Fig. 2 Higher degrees cause over fit, then conduct higher Test MSE.



It is obvious that training MSE will keep decreasing as model complexity increases. However, the test MSE will go up at over-fitting.

Fig. 3 Overfitting generates high variance.

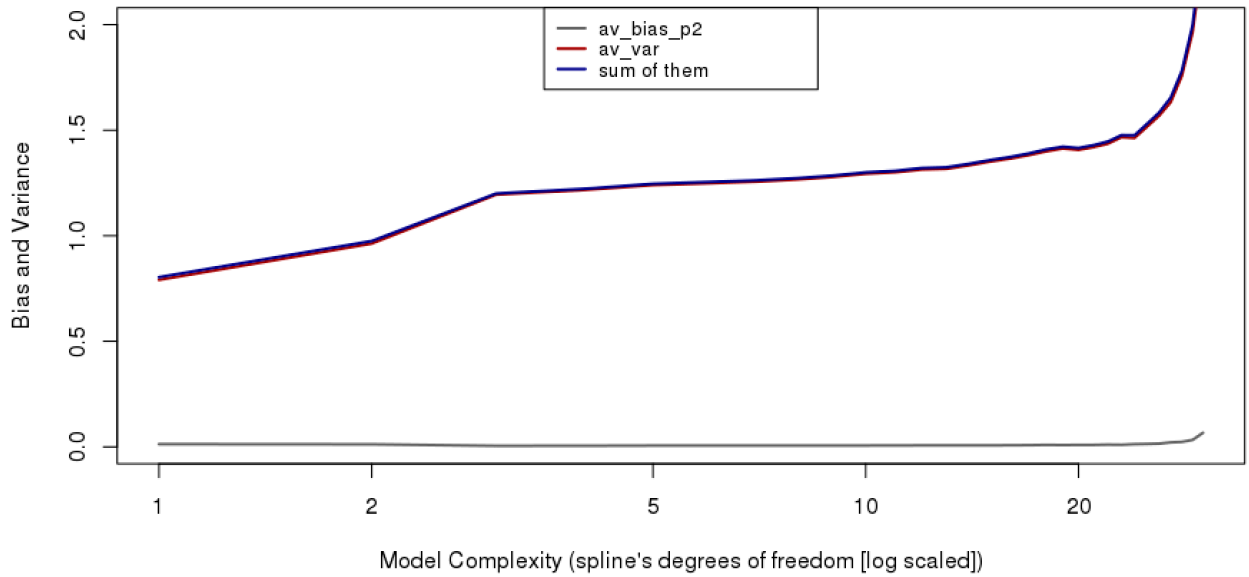


Fig. 3 demonstrates the high variance when overfitting occurs.

Random Forest Regression

Continue the same simulation data, the RF model is chosen from ntree = <degree> as the model complexity.

R script

Run jlt_245_final_exam_rf_regression. R

Fig. 4 The fitting curves by ntree numbers for RF algorithm.

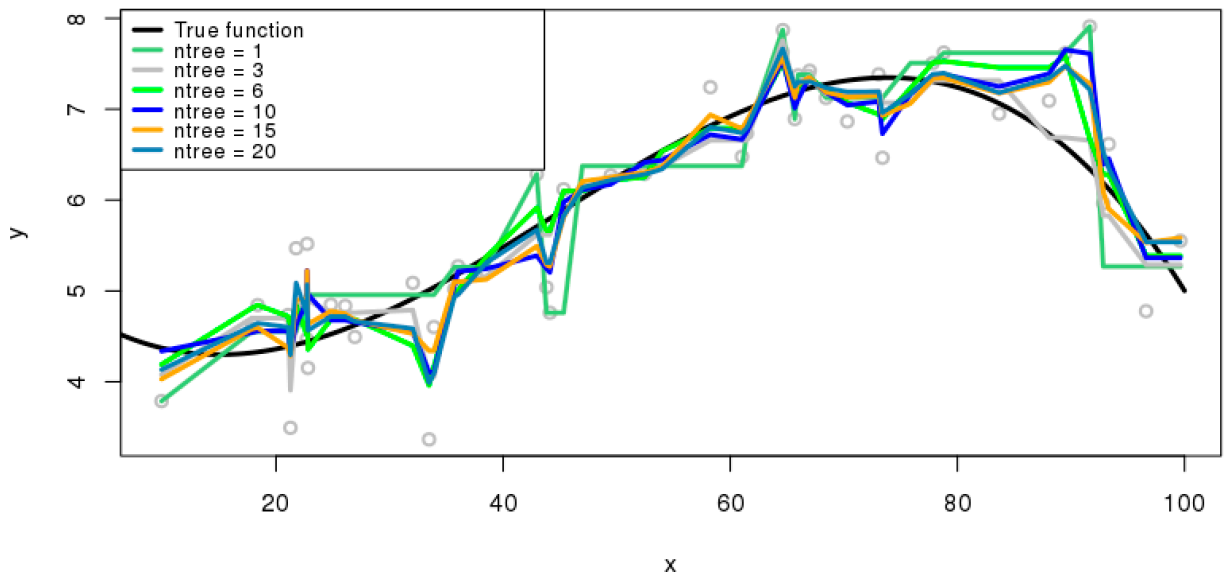
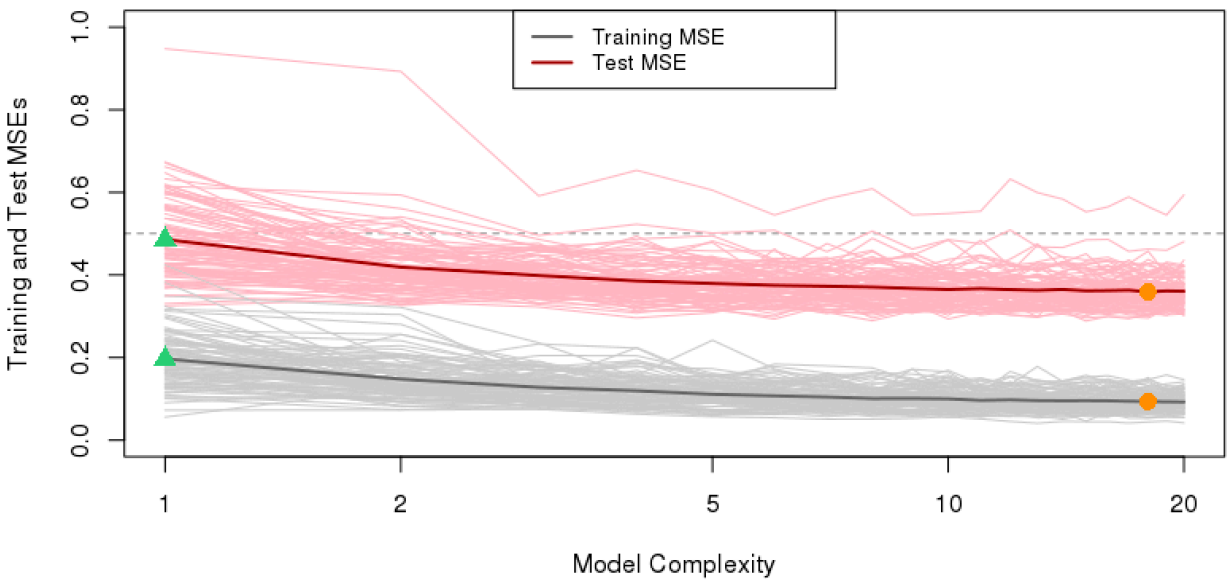
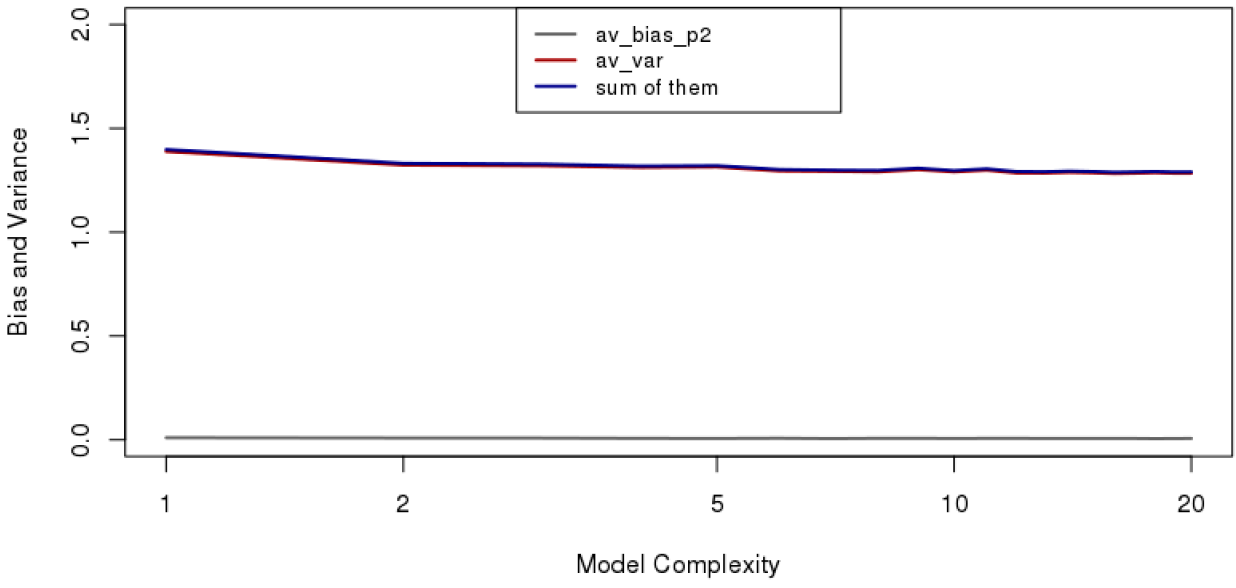


Fig. 5 The Training and Test MSEs of Random Forest



It is understandable that for RF, the tree node increases, the better fitting to the estimator. In Fig. 6 below, also shows the variance is stably decreasing when ntree, the tree nodes increase.

Fig. 6 Average Bias and Variance



Classification

In the classification experiment, I used the dataset[13] from UCI, which

GLM Classification

Generalized linear models (GLM) are fit using the **glm( )** function. The form of the **glm** function is

glm(*formula***, family=***familytype* **(link=***linkfunction***), data=)**

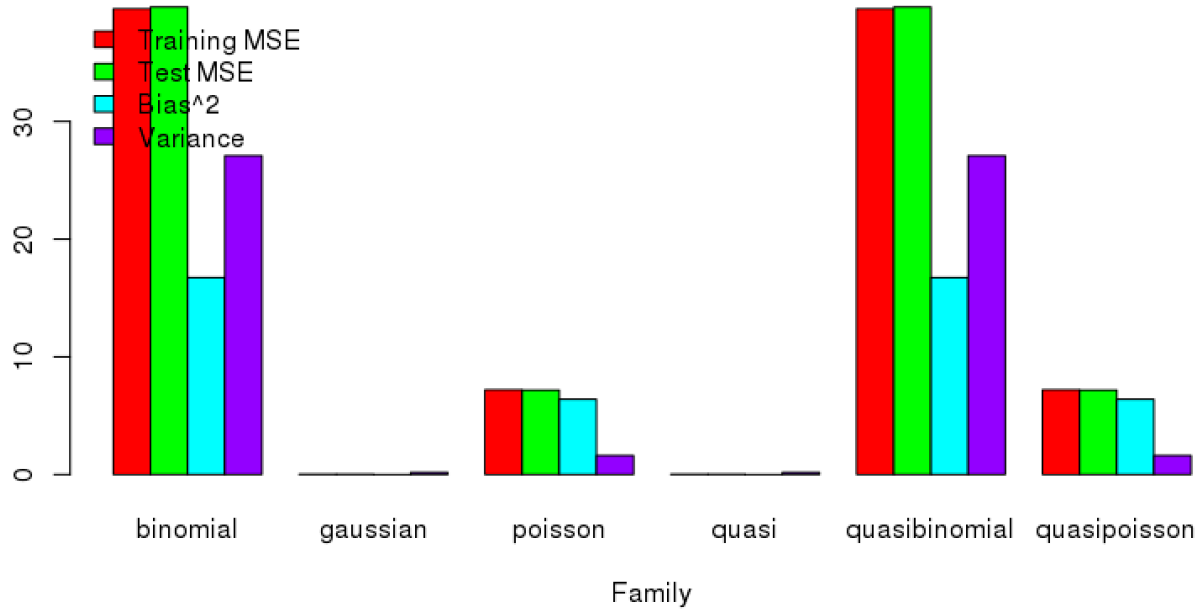
Family	**Default Link Function**
binomial	(link = "logit")
gaussian	(link = "identity")
Gamma	(link = "inverse")
inverse.gaussian	(link = "1/mu^2")
poisson	(link = "log")
quasi	(link = "identity", variance = "constant")
quasibinomial	(link = "logit")
quasipoisson	(link = "log")

In this experiment, I would demonstrate the Bias and Variance for each family type. Of course, this result is based on the dataset I used.

R script

Run jlt_245_final_exam_glm_classification. R

Fig. 7 the MSE, Bias and Variance of each GLM family type.



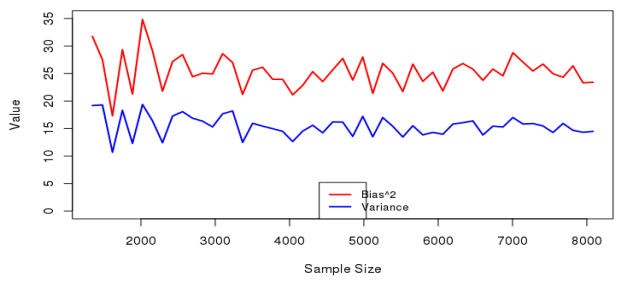
Sampling Effect

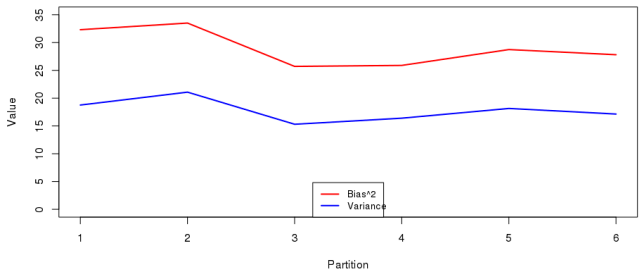
Using binomial family of GLM as the experiment of sampling. Fig. 10 is the test with different sample size, the bais^2 and variance

R script

Run jlt_245_final_exam_glm_classification_sampling. R

Fig. 8 the variation of variance and bias due to sampling in GLM.





GLM seems to show the nice ability of minimizing bias and variation through model selection and sampling.

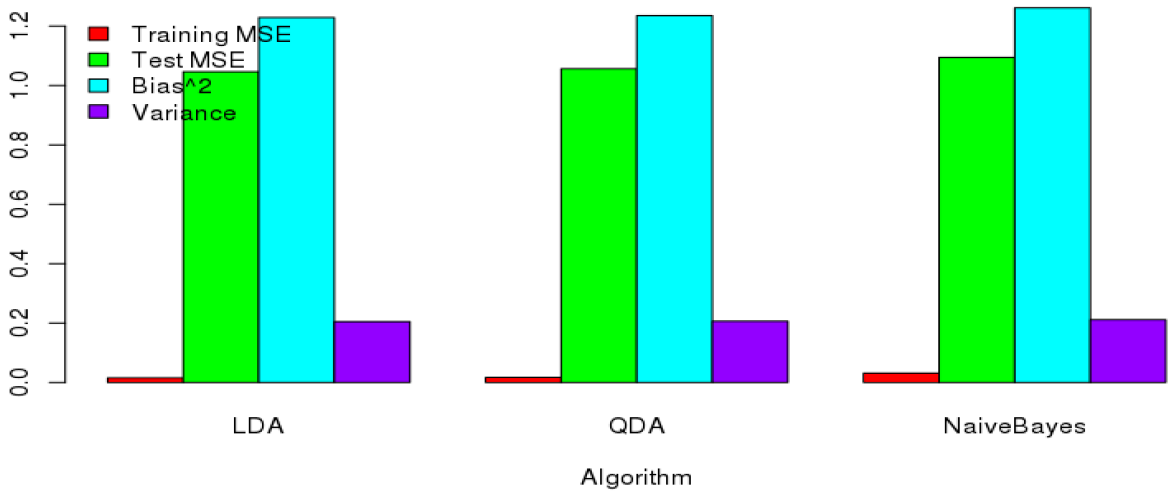
LDA, GDA and Naive Bayes Classification

These three classifier seems very insensitive to bias and variance variation. In R packages, they always achieve very good and stable variance and bias.

R script

Run jlt_245_final_exam_lda_qda_naivebayes_classification. R

Fig. 9 The bias and variance measures of LDA, QDA and NaiveBayes.

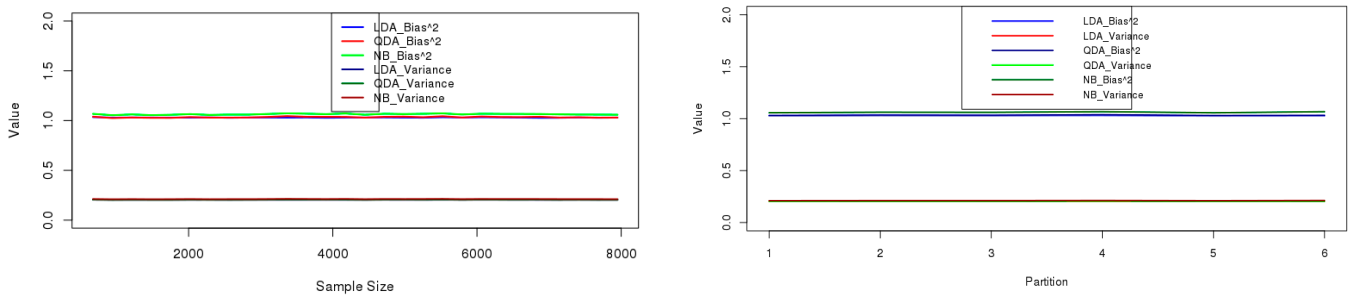


Sampling Effect

R script

Run jlt_245_final_exam_lda_qda_naivebayes_classification_sampling. R

Fig. 10 The bias and variance of the three algos v.s. sampling size and partition



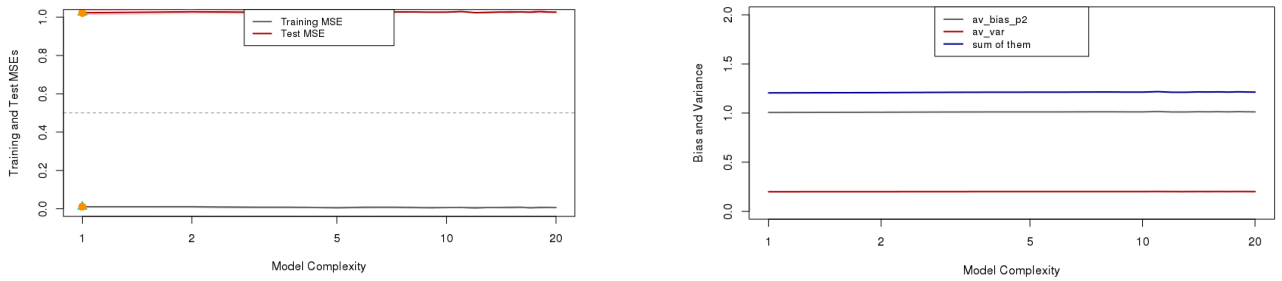
Random Forest Classification

In RF experiment, since the dataset is two classes situation, the ntree nodes that I used to provide the model complexity seems to have no impact on MSE.

R script

Run jlt_245_final_exam_rf_classification. R

Fig. 11 MSE seems no change due to the two classes test case.

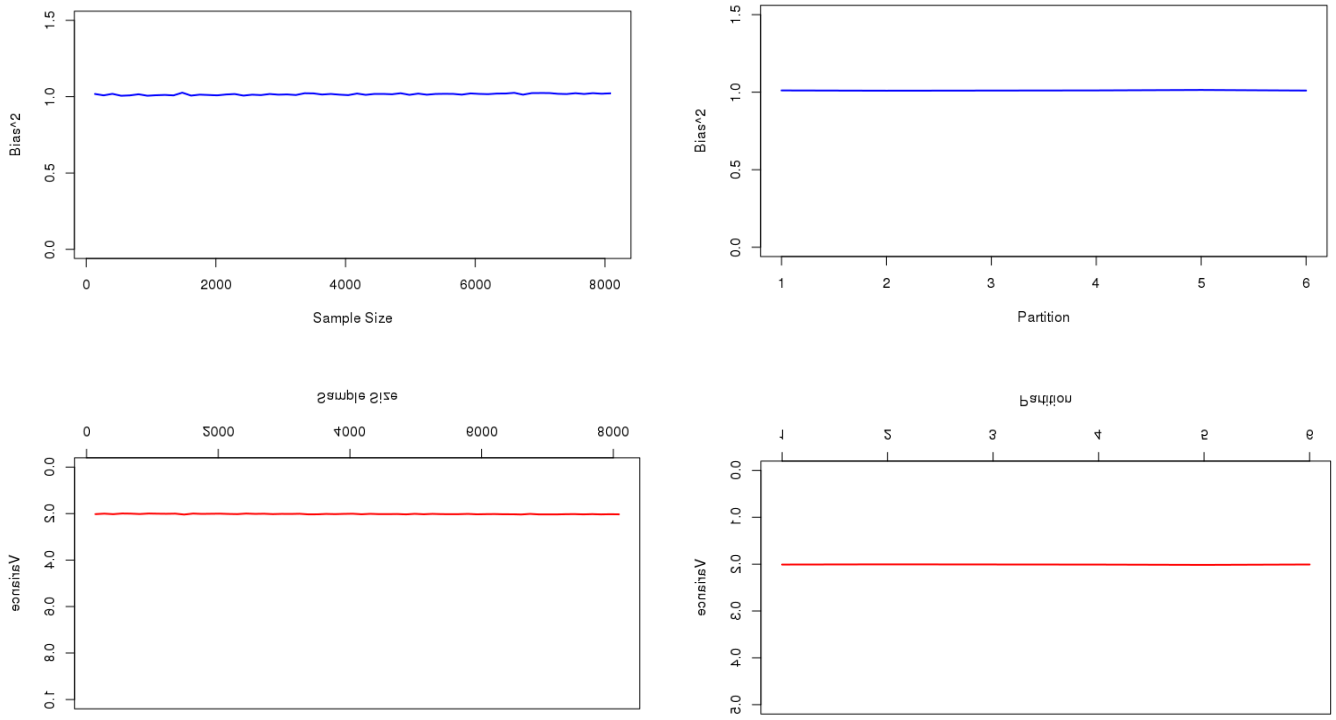


Sampling Effect

R script

Run jlt_245_final_exam_rf_classification_sampling. R

Fig. 12 rarely changed bias and variance in sampling



SVM Classification

Each kernel function of SVM are explored. [16]

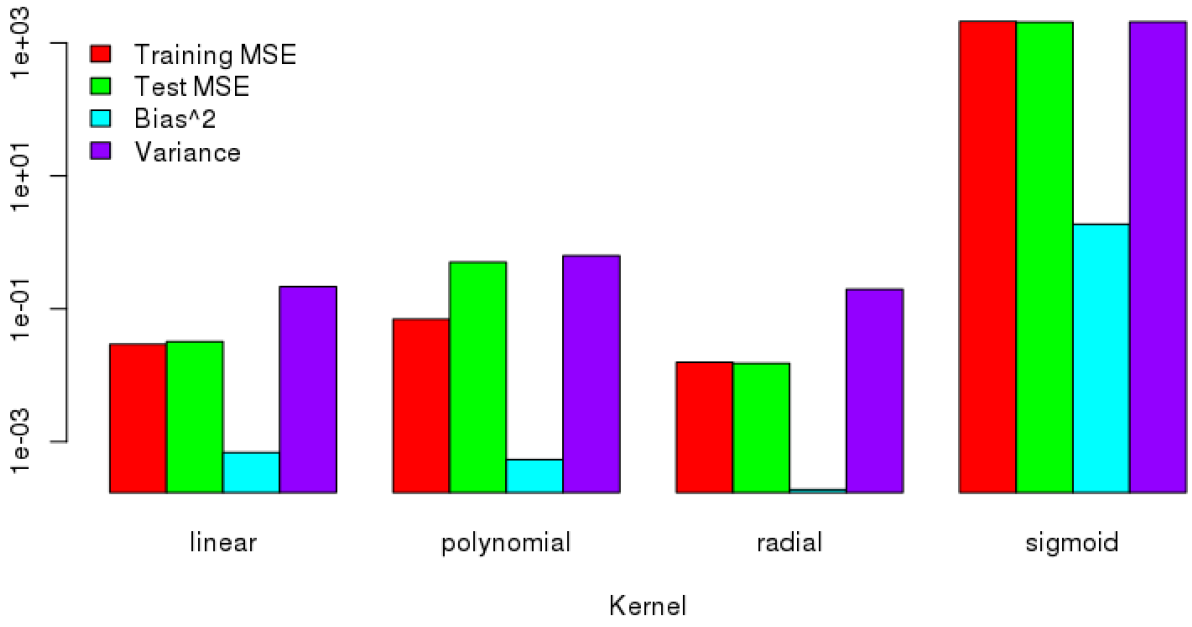
`kernel`	the kernel used in training and predicting. You might consider changing some of the following parameters, depending on the kernel type. **linear:** $u'v$ **polynomial:** $(\gamma u'v + coef0)^{degree}$ **radial basis:** $e^(-\gamma \lvert u-v \rvert^2)$ **sigmoid:** $tanh(\gamma u'v + coef0)$

I only used the default values for each kernel function for the tests. Sigmoid kernel is hard to manage. It generates high MSE, but I think I just did not use it right.

R script

Run jlt_245_final_exam_svm_classification. R

Fig. 13 Variation of MSE in each kernel.

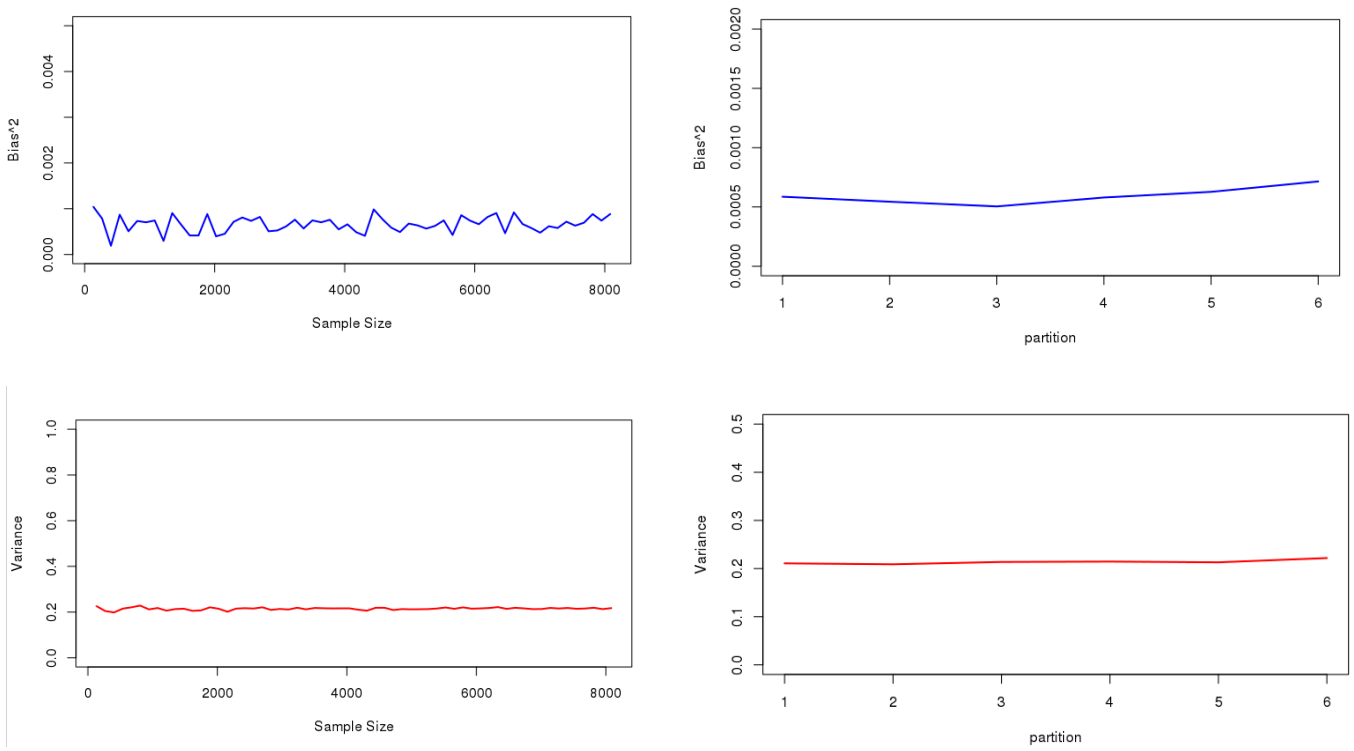


Sampling Test

R script

Run jlt_245_final_exam_svm_classification_sampling. R

Fig. 14 Some variation of bias are seen in sampling test for kernel “linear”.



Conclusion

Accuracy measure is related to bias and variance measures which combines into MSE. To achieve better accuracy performance, the reduction of bias or variance is the approach.

Overfitting conducts high variance. Model complexity and small training dataset are two major reasons to cause overfitting. A proper model to best fit the test data is what Occam’s razor principle is about. Small training dataset may conduct overfitting too as it does not provide enough information for the algorithm to do the best fit.

Sampling and resampling are the ways of reducing bias. N-folds, cross validation and resampling are the techniques.

In my experiments, GLM is the most capable of tuning bias and variance thus minimize them.

Random Forest seems a high bias but low variance situations. Although I did not find ways of exploring the variety of bias in the RF experiment. I guess most of R packages for ML has been optimized to achieve a stable bias-variance situation. So, it does not give out the flexibility to create those situations of bias variation.

More tree nodes in RF can achieve better performance of regression in my tests. In general, overfitting is generally caused by over growing the trees in random forests. But the R package that I used seems to restrict such possibility. I will get errors if I grow the tree nodes too much.

SVM does create bias variation due to sampling efforts as in SVM sampling experiments. Since their values are low, I doubt such technique is useful in the practices.

In non-linear kernel case of SVM, the smoothness of the kernel function may impact the complexity of the classifier, hence on the risk of over-fitting. The performance of the SVM can be sensitive to the selection of the regularization and kernel parameters and it is possible to get over-fitting in tuning the performance measurements. [18] Unfortunately, I am unable to go through the variation of those parameters in my tests.

LDA, QDA and Naïve Bayes are quite stable in terms of bias and variance variation due to sampling. I did not see the ability of tuning them.

Resampling [17] like “bootstrap”, “jackknife” can estimate the precision, variances by random replacement of training data subset. Both can significantly achieve better performance.

Appendix

All R sources are located at https://github.com/billtsay/final-exam.

Reference:

[1] Machine Learning (1977), Tom M. Mitchell, https://docs.google.com/file/d/0ByVUs_BZLlUyRi1zT1VjZEJGcTA/edit

[2] Deep Learning (2016), Ian Goodfellow, Yoshua Bengio, and Aaron Courville.

[3] Machine Learning. A Probabilistic Perspective (2012), Kevin P. Murphy.

[4] Fundamentals of Machine Learning for Predictive Data Analytics (2015), John D. Kelleher, Brian Mac Namee, Aoife D'Arcy.

[5] Ensemble Methods Foundations and Algorithms (2012), Zhi-Hua Zhou

[6] Computer Age Statistical Inference (2016), Bradley Efron, Trevor Hastie

[7] The Elements of Statistical Learning (2009), Trevor Hastie, Robert Tibshirani, Jerome Friedman.

[8] An Introduction to Statistical Learning (2013), Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani.

[9] https://www.quantstart.com/articles/The-Bias-Variance-Tradeoff-in-Statistical-Machine-Learning-The-Regression-Setting

[10] http://www.milanor.net/blog/cross-validation-for-predictive-analytics-using-r/

[11] https://www.rdocumentation.org/packages/splines/versions/3.3.2

[12] https://en.wikipedia.org/wiki/Bias_of_an_estimator

[13] https://en.wikipedia.org/wiki/Variance

[14] http://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+

[15] https://github.com/billtsay/final-exam

[16] https://cran.r-project.org/web/packages/e1071/e1071.pdf

[17] https://en.wikipedia.org/wiki/Resampling_(statistics)

[18] http://www.jmlr.org/papers/volume11/cawley10a/cawley10a.pdf