ASSEMBLY INDUCED FAILURE CLASSIFICATION AND PREDICTION

PAKYADA MANEEWAN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN DATA STORAGE TECHNOLOGY
INTERNATIONAL COLLEGE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2016

KMITL-2016-IC-M-005-002

Thesis Certification

International College

King Mongkut's Institute of Technology Ladkrabang

Thesis Title	Assembly Induced Failure Classification And Prediction
Student	Ms. Pakyada Maneewan
Student ID.	54600711
Degree	Master of Engineering
Program	Data Storage Technology
Thesis Advisor	Asst.Prof.Dr. Siridech Boonsang
Thesis Reference Number	KMITL-2016-IC-M-005-002

EXAMINERS	SIGNATURES
Assoc.Prof. Dr. Pitikhate Sooraksa	
Asst.Prof.Dr. Siridech Boonsang	
Asst.Prof.Dr. Chanon Warisarn	
Asst.Prof.Dr. Anakkapon Saenthon	
Asst.Prof.Dr. Thavida Maneewarn	

Date 2 June 2016 **Time** 13.30 - 15.30

Place International College, 8th floor, 55th Anniversary Chalermprakiat Building

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

(Assoc.Prof.Dr. Supat Kittiratsatcha)

Dean

2 June 2016

หัวข้อวิทยานิพนธ์	การจำแนกและพยากรณ์การเสียของแผ่นบันทึกข้อมูลที่เกิดจากขณะประกอบฮาร์ดดิสก์
นักศึกษา	นางสาวภัคญดา มณีวรรณ
รหัสประจำตัว	54600711
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีการบันทึกข้อมูล
พ.ศ.	2559
อาจารย์ที่ปรึกษาวิทยานิพนธ์	ผศ.ดร.ศิริเดช บุญแสง

บทคัดย่อ

การเสียของแผ่นข้อมูลจากขณะประกอบฮาร์ดดิสก์ เป็นประเภทหนึ่งของการเสียในแผ่นข้อมูลในขณะที่กำลังประกอบฮาร์ดดิกส์ในห้องปลอดฝุ่น โดยปรกติแล้วหลังจากประกอบฮาร์ดดิสก์เสร็จจะมีขั้นตอนการตรวจสอบคุณภาพรวมถึงการตรวจสอบการเสียของแผ่นข้อมูลทุก ๆ ประเภทด้วย แต่เนื่องด้วยเวลาที่ใช้ในการทดสอบจนกระทั่งรู้การเสียของแผ่นข้อมูลทุกประเภทใช้เวลานานมากอีกทั้ง การเสียของแผ่นข้อมูลประเภทนี้มีความจำเป็นต้องรู้ให้เร็วที่สุดเพื่อที่จะสามารถแก้ไขข้อผิดพลาดในห้องปลอดฝุ่นได้ทันท่วงทีเพื่อลดค่าใช้จ่ายที่ตามมา ดังนั้นจึงมีความจำเป็นมากที่รู้ว่ามีการเสียของแผ่นข้อมูลประเภทนี้เกิดขึ้นให้เร็วที่สุดเท่าที่จะเป็นไปได้ วัตถุประสงค์ของการศึกษานี้คือหาโมเดลของการจำแนกที่มีอยู่ในปัจจุบันที่สามารถให้ค่าความถูกต้องสูงที่สุดที่มาจำแนกการเสียของแผ่นข้อมูลขณะประกอบฮาร์ดดิสก์ โดยแบบจำลองที่สนใจศึกษาคือ นิวรอนเน็ตเวิร์ค, นาอีฟเบย์, ต้นไม้ตัดสินใจ และ การสุ่มต้นไม้ในป่า โดยการทดลองจะมีการเพิ่มจุดทดสอบตรวจหาการเสียของแผ่นข้อมูลขนาดย่อมในจุดที่เร็วที่สุดเท่าที่จะเป็นไปได้เพื่อจะได้มาซึ่งขนาดที่ใหญ่ที่สุดของการเสียของแผ่นข้อมูลแต่ละพื้นผิว และจำนวนจุดที่เสียในแผ่นข้อมูลแต่ละแผ่น ซึ่งข้อมูลนี้จะนำไปเป็นข้อมูลที่ใช้ในแต่ละโมเดล โดยที่แต่ละโมเดลที่ทำการทดลองจะใช้จำนวนข้อมูลต่างกันคือจากจำนวนฮาร์ดดิสก์ ตั้งแต่ 100, 200, ..., 800 ตัว แล้วทำการคำนวนเพื่อนำไปสู่ค่าความถูกต้องของแต่ละโมเดล ผลการทดลองพบว่าค่าความถูกต้องของแต่ละแบบจำลองเริ่มคงที่ตั้งแต่จำนวนข้อมูล 600 เป็นต้นไป และ ต้นไม้ตัดสินใจเป็นโมเดลที่ให้ค่าความถูกต้องในการจำแนกมากที่สุดจากการใช้ข้อมูลของขนาดที่ใหญ่ที่สุดของการเสียของแผ่นข้อมูลแต่ละพื้นผิว

Thesis Title:	Assembly Induced Failure Classification and Prediction
Student:	Pakyada Maneewan
Student ID:	54600711
Degree:	Master of Engineering
Program:	Data Storage Technology
Year:	2016
Thesis Advisor:	Asst.Prof.Dr Siridech Boonsang

ABSTRACT

Assembly Induced Defect (AID) is a defect induced during the hard disk drive assembly process, occurring in a clean room and is captured later during the testing process. As the time is needed to detect all flaw types, including AID, in a regular defect scan sequence is considered too long and costly to reject the drive; it would be beneficial for the hard disk drive factory to be able to capture the AID at the early stages of the test process. The engineer responding at Hard Disk Drive (HDD) assembly process can correct the assembly fault quicker thus reducing time to failure. The objective of this study is to identify the best existing classification model which can produce the highest accuracy to capture this failure. The classification models in this study are Neural Network, Naïve Bayes, Random Forest and Decision Tree. An additional defect scan sequence, for the data pre-processing, is added at the earliest possible state in test process to identify the required information which are the input data elements of the selected model. All selected models are measured based on different input data, maximum defect size per surface or defect count by head-zone, and data sizes starting from 100, 200 until 800. The results indicate that the detection accuracy would be stable at sample size 600 to 800. Hence, the Decision Tree model is selected as it can provide the highest accuracy based on maximum defect size as input information.

ACKNOWLEDGEMENT

I am grateful to Seagate Technology (Thailand) Ltd and College of Data Storage Innovation, King Mongkut's Institute of Technology Ladkrabang for both financial and experimental equipment support.

I am gratefully thanks to Asst.Prof.Dr Siridech Boonsang, my advisor, for the valuable recommendation of my study.

Furthermore, I am thankfully to my beloved mother, Pattranid Kosintanapat for her 100% accuracy to encourage me and asked me to study for knowledge in purpose.

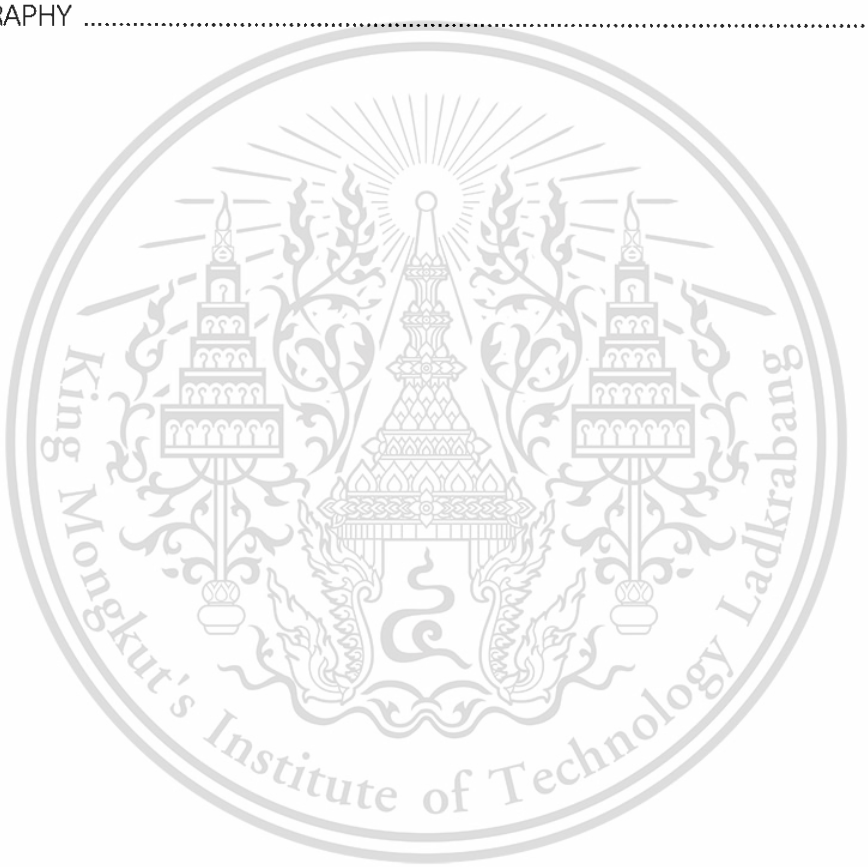
Finally, I thank you to my Seagate's colleague who did support on large quantity experiment.

Pakyada Maneewan

CONTENTS

CONTENTS (Cont.)

CONTENTS (Cont.)



LIST OF FIGURES

LIST OF FIGURES (Cont.)

LIST OF FIGURES (Cont.)

LIST OF TABLES

LIST OF TABLES (Cont.)

CHAPTER 1

INTRODUCTION

1.1 Backgrounds and Problem Statement

In the HDD industry, there are 2 major processes required before ship drive to customer. One is clean room process and another one is backend process.

For clean room process, most of operation is mechanical assembly of hard disk drive. It takes place in clean room which is required class 100 cleanliness (the maximum particles 0.5 um or larger in a cubic meter is 100)[1] to prevent particle damage the hard disk drive part. The HDAs are comprised of sophisticated components which are all assembled by automation machines with several processes starting from Basedesk load, LVCM install, LVCM screw install until HDA clean room exit as shown in Figure 1.1.

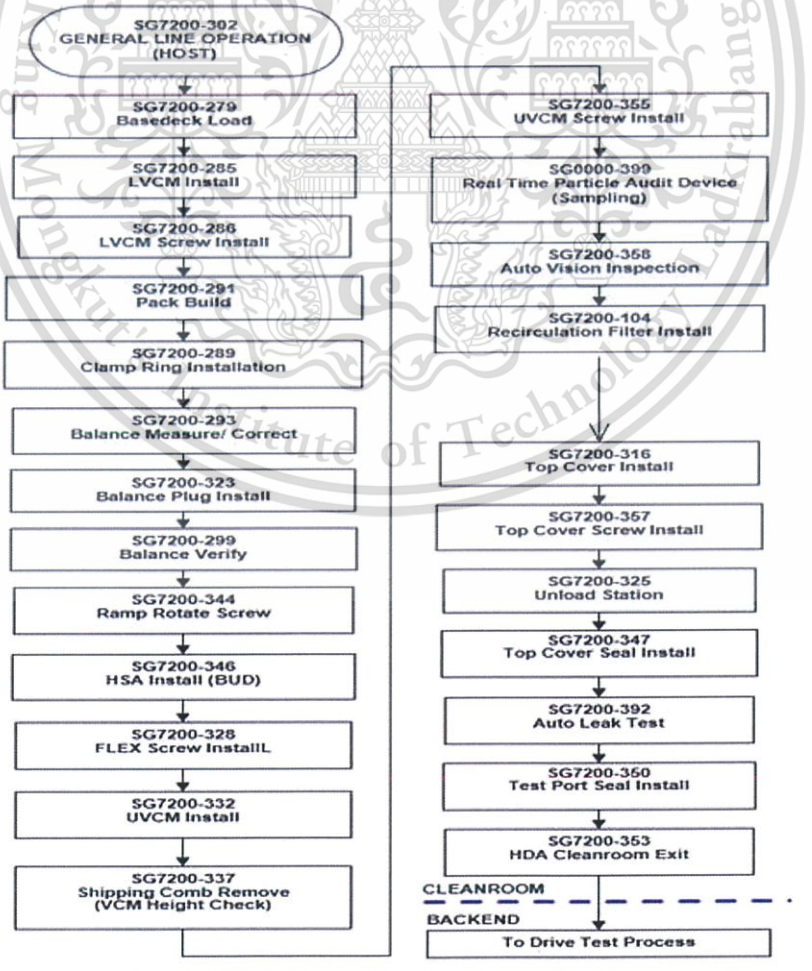


Figure 1.1 Clean room process flow.

For backend process, the sequence starting with PWA operation which installs PCBA (Printed Circuit Board Assembly) into HDA, and then the full assembled component hard disk drives are brought to test in the tester which consist of several test sequence for instant MDW (Multi Disk Write), VBAR (Variable Bit Aspect Ratio) and AFH (Adaptive Fly Height) for vertically aligned across all the disc measuring, drive capacity measuring and individual head fly height measuring respectively. Finally, all of passer drives are pack and ready to ship to customer as shown in Figure 1.2.

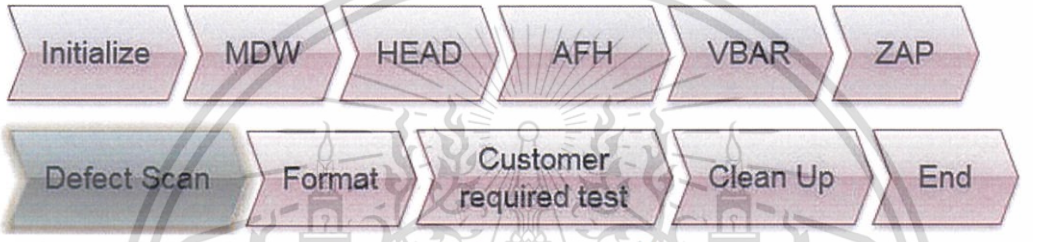


Figure 1.2 Test process flow.

In the test process, "Defect Scan" is one of the test process sequences which can identify all of defect parameters such as the defect size, defect amplitude and defect address in media. This sequence can either reject the drive when defect parameters exceed the threshold target or disable surface of defect location when defect parameters is still in the acceptable range. There are several required test sequences before Defect scan test such as MDW, AFH or VBAR. Therefore, the estimate time from beginning test process until Defect Scan is about 80 hr.

The defect captured in Defect Scan would come from either incoming media or be induced by assembly in factory for example during the DSP (Disc Separator Plate) installation. If there are some misalignments or errors, the edge of DSP will easily come in contact with the disc media and damage it as shown in Figure 1.3 and 1.4. This defect is called "Assembly Induced Defect". The shape of this defect is similar to DSP shape. Currently, failure analysis team can identify this defect type by classifying from defect shape, defect location and their amplitude. There are two possible disposition from this kind of defect are fail at Defect Scan test or pass Defect scan with marking disable location usage.

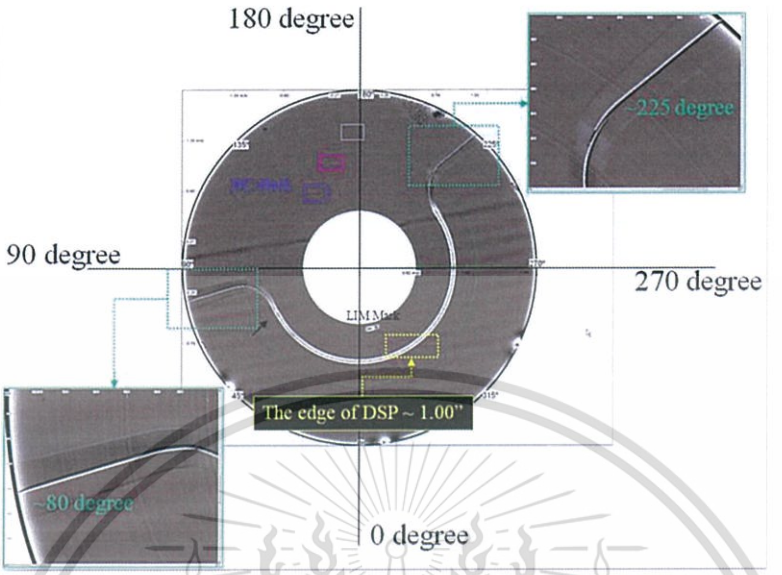


Figure 1.3 Disc Separator Plate with Disc Media.

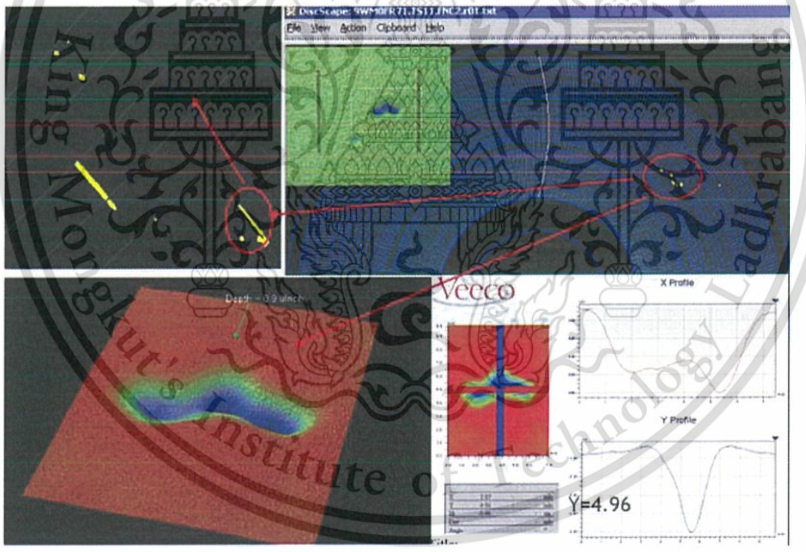


Figure 1.4 Assembly Induced Defect Sample.

Furthermore, the possibility of this defect to grow after being mapped in media optimization test is very high. Thus, the right disposition should be to detect and reject this defect as soon as possible. We should not mark the defect, so called defect mapping in HDD industry, and let it pass to the following test sequence to avoid the quality issues later on. It is highly benefit for Hard Disk Drive factory to be able to detect this Assembly Induced Defect at the early stage of the test process because the assembly line can be quickly alerted on the problem and solve it faster for cost reduction.

1.2 Objective

To identify the best existing classification model for the prediction of Assembly Induced Defect based on both maximum defect size per surface and defect count per surface information. The models selected in this study are Neural Network (NN), Naïve Bayes, Decision Tree and Random Forest.

1.3 Scope of work

1.3.1 To identify the earliest stage for adding special defect scan to faster acquire defect information.

1.3.2 To identify the classification model which provides the highest accuracy with the accuracy greater than 95%.

1.4 Expected Benefits

1.4.1 Apply the model into normal test process to predict the Assembly Induced Defect and reject it.

1.4.2 Being able to solve the defect induced in assembly process faster.

CHAPTER 2

LITERATURE REVIEW

The media defect which was occurred during HDA assembly could be captured in Defect Scan test process. The previous studies which are associated with the classification and prediction have presented their accuracy. Most of the study's accuracy were promising, however some studies show the classifier does not be the major contribution to the accuracy. The studies are as follows;

In 2006, Tyndall and Khurshoudov [2] introduced the model to predict the reliability failure of HDI (Head Disk Interference) based on temperature, air pressure and humidity variation. The HDI failure signal can be predicted by the clearance value. The lower clearance value, the higher chance HDI failure occurs.

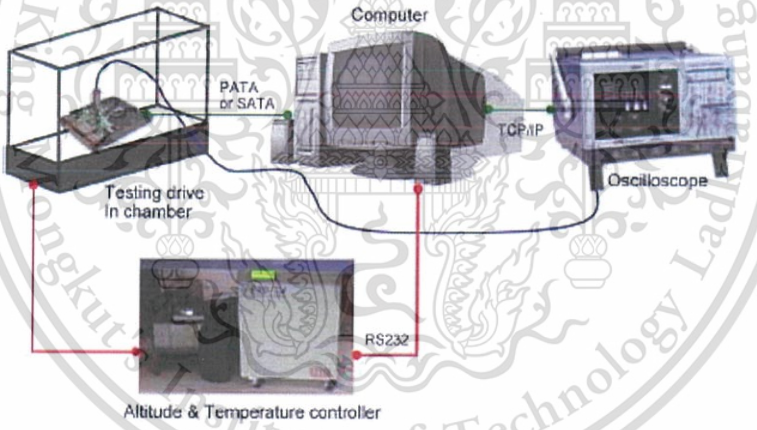


Figure 2.1 Diagram of altitude clearance measurement [2].

The experiment was set up as shown in Figure 2.1. The computer controlled the altitude and temperature controller to input different altitude and environment temperature in vacuum testing drive chamber. Then the actual clearance was presented in oscilloscope. The results from the study found higher temperature generates the lower clearance, the higher altitude (lower pressure) and the higher humidity generates same result that is lower clearance. The prediction model of temperature 60 °C and 0 °C by varying altitude was presented in Figures 2.2 and 2.3 respectively.

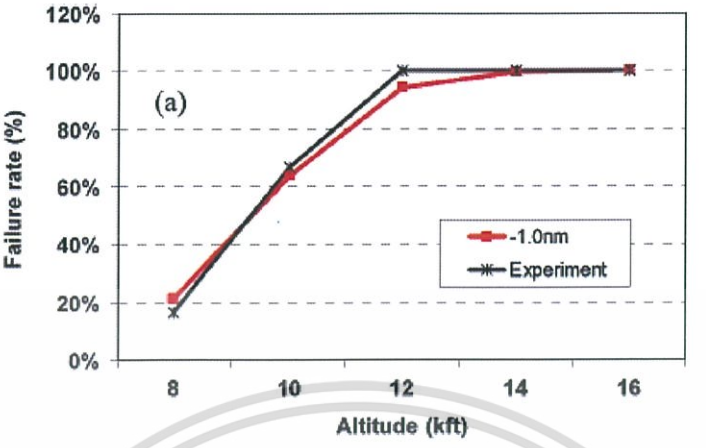


Figure 2.2 The prediction model at temperature 60C, clearance limit -1.0 nm against the experiment chart [2].

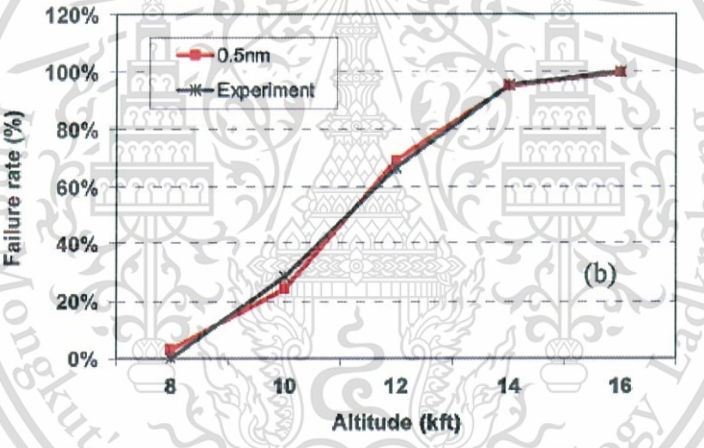


Figure 2.3 The prediction model at temperature 0C, clearance limit 0.5 nm against the experiment chart [2].

In 2010, Farhoodi and Yari [3] studied the accuracy of Persian text document classification based on the existing supervised learning algorithms. According to the many studied regarding text classification with positive results, the selected algorithms are Support Vector Machine (SVM) and K-Nearest Neighbor (KNN) and the experiment flow is shown in Figure 2.4. Since the Persian text document is unlikely English, text preprocessing, document indexing and keywords selection is required before applied into the classification algorithm. The precision results of Support Vector Machine and K-Nearest Neighbor algorithm were presented in Tables 2.1 and 2.2 respectively. Both algorithms can generate the acceptable accuracy which are 0.8918 and 0.9738 of Support Vector Machine (SVM) and K-Nearest Neighbor respectively. Anyway, the KNN is better performance than SVM.

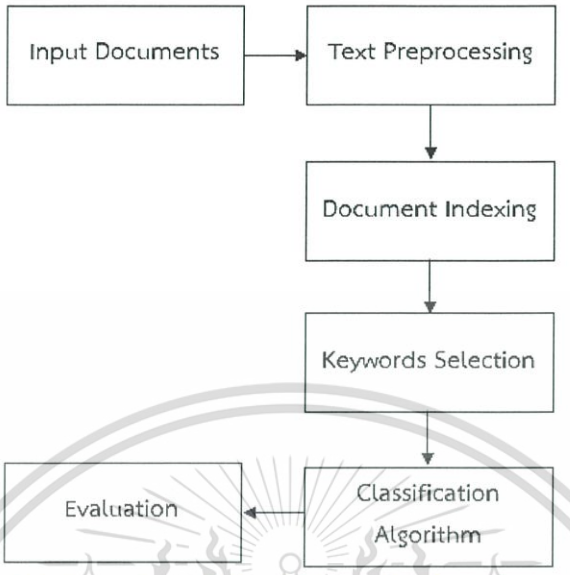


Figure 2.4 Overview of the Categorization Process [3].

Table 2.1 The accuracy by applying Support Vector Machine (SVM) algorithm [3].

Kernel Function	Precision	Recall	F-Measure
Linear	0.7865	0.8750	0.8284
Polynomial	0.8918	0.9895	0.9381
Rbf	0.8715	1	0.9313
Quadratic	0.8127	0.8652	0.8381
Mlp	0.7865	0.8750	0.8284

Table 2.2 The accuracy by applying K-Nearest Neighbor (KNN) algorithm [3].

K = 3, Distance Metric = cosine, Rule = nearest		
Number of features	Correct Rate (Precision)	Error Rate
10	0.6975	0.3025
100	0.8950	0.1050
500	0.9450	0.0550
1000	0.9475	0.0525
2000	0.9613	0.0388
3000	0.9700	0.0300
4000	0.9738	0.0262
5000	0.9712	0.0288

In 2011, Ramangkul and Ponsawad [4] classified the hard disk drive failure modes which are CND/NPF, Firmware, Head Related, Mechanical, Media or Test process by using the classification model: Support Vector Machine (SVM), Neural Network (NN), Bayes Network and C5.0 based on data sets of S.M.A.R.T (Self-Monitoring and Reporting Technology). The results illustrated in Table 2.3 presented the best model for this study generating the highest accuracy is C5.0.

Table 2.3 The accuracy of different models based on same data sets [4].

Model	Accuracy (Testing)
C5.0	93.03%
Neural Network	56.25%
Bayes Network	53.00%
SVM	42.03%

In 2013, Chug and Dhall [5] studied the of both supervised learning and unsupervised learning algorithm to classify the NASA MDP data sets as shown in table 2.4. The selected supervised algorithms are Random Forest, Naïve Bayes (NB) and Decision Trees (J48). Besides, the selected unsupervised algorithms are K-means Clustering, Hierarchical Clustering and Make Density Based Clustering. The accuracy of supervised learning and unsupervised learning algorithm were provided in Tables 2.5 and 2.6 respectively. The Random Forest Algorithm is the best supervised learning and K-means clustering algorithm is the best unsupervised algorithm to achieve the highest accuracy on data set of interest.

Table 2.4 NASA MDP data sets [5].

Name	Language	Total KLOC	No. of Modules	%Defective Modules
CMI	C	20	505	10
JMI	C	315	10878	19
KCI	C++	43	2107	15
KC3	Java	18	458	9
KC4	Perl	25	125	49
MC1	C & C++	63	9466	0.7
MC2	C	6	161	32
MW1	C	8	403	8
PC1	C	40	1107	7
PC2	C	26	5589	0.4
PC3	C	40	1563	10
PC4	C	36	1458	12
PC5	C++	164	17186	3

Table 2.5 The accuracy of supervised algorithm [5].

	Model	Recall	Precision	F-Measure	ROC	MAE	RMSE	RAE	Accuracy
CM1	J48	.855	.836	.844	.594	0.1722	0.3645	79.63%	85.46%
	NB	.823	.832	.827	.694	0.1784	0.4164	82.51%	82.26%
	RF	.858	.823	.836	.725	0.1849	0.3235	82.50%	85.75%
KC1	J48	.842	.813	.821	.674	0.2069	0.3661	78.88%	84.16%
	NB	.824	.816	.820	.791	0.1759	0.4135	67.06%	82.44%
	RF	.851	.837	.842	.827	0.1863	0.3274	71.03%	85.06%
KC3	J48	.805	.790	.797	.624	0.2264	0.4260	76.08%	80.50%
	NB	.785	.773	.779	.636	0.2176	0.4549	73.13%	78.50%
	RF	.780	.750	.762	.698	0.2530	0.3805	85.01%	78.00%
MC1	J48	.994	.992	.992	.827	0.0106	0.0783	72.28%	99.36%
	NB	.941	.990	.963	.892	0.0592	0.2404	100.27%	94.11%
	RF	.996	.995	.995	.891	0.0082	0.0664	55.58%	99.55%
MC2	J48	.630	.617	.621	.585	0.3930	0.5779	86.59%	62.99%
	NB	.732	.728	.707	.717	0.2755	0.5191	60.70%	73.22%
	RF	.630	.613	.618	.633	0.3882	0.4863	85.52%	62.99%
PC1	J48	.909	.893	.899	.719	0.1201	0.2863	80.70%	90.90%
	NB	.883	.892	.887	.768	0.1156	0.3377	77.63%	88.27%
	RF	.914	.889	.897	.806	0.1209	0.2595	81.20%	91..43%
PC2	J48	.987	.980	.984	.448	0.0280	0.1120	100.80%	98.73%
	NB	.955	.984	.968	.878	0.0445	0.2046	100.48%	95.45%
	RF	.989	.980	.984	.657	0.0181	0.1050	87.64%	98.86%
PC3	J48	.858	.843	.849	.655	0.1622	0.3572	74.24%	85.77%
	NB	.358	.847	.414	.743	0.6347	0.7719	100.49%	35.82%
	RF	.869	.850	.857	.814	0.1704	0.3027	77.99%	86.73%
PC4	J48	.896	.896	.896	.773	0.1127	0.3030	50.64%	89.56%
	NB	.869	.857	.862	.825	0.1339	0.3492	60.16%	86.91%
	RF	.901	.893	.896	.908	0.1397	0.2679	62.78%	90.06%
PC5	J48	.974	.971	.972	.805	0.0326	0.1522	56.75%	97.35%
	NB	.963	.966	.964	.834	0.0369	0.1912	64.13%	96.30%
	RF	.975	.974	.975	.950	0.0302	0.1300	52.56%	97.52%

Table 2.6 The accuracy of unsupervised algorithm [5].

	Algorithm	Time Taken (sec)	Cluster Instance (0 & 1)	No. of iterations	Within cluster sum of squared error	Incorrectly clustered instances	Log Likelihood
CM1	KM	0.08	14% & 86%	10	205.2676	17.7326%	-
	HC	0	99% & 1%	-	-	12.2093%	-
	MDBC	0.13	19% & 81%	10	205.2676	21.5116%	-122.89047
KC1	KM	0.39	17% & 83%	19	203.6514	18.7023%	-
	HC	0.03	100% & 0%	-	-	15.5057%	-
	MDBC	0.5	22% & 78%	19	203.6514	20.3244%	-63.77387
KC3	KM	0.08	83% & 18%	12	174.7175	22.5%	-
	HC	0	96% & 5%	-	-	18.5%	-
	MDBC	0.06	78% & 22%	12	174.7175	24%	-111.19398
MC1	KM	2.45	25% & 75%	9	5120.6531	24.8141%	-
	HC	0.13	100% & 0%	-	-	0.7546%	-
	MDBC	1.72	23% & 77%	9	5120.6531	23.0678%	-102.36315
MC2	KM	0.02	10% & 90%	3	115.7140	30.7087%	-
	HC	0.02	97% & 3%	-	-	31.4961%	-
	MDBC	0.03	13% & 87%	3	115.7140	29.1339%	-129.82322
PC1	KM	0.13	34% & 66%	6	324.2966	37.1542%	-
	HC	0.08	100% & 0%	-	-	7.7734%	-
	MDBC	0.09	36% & 64%	6	324.2966	36.3636%	-139.90952
PC2	KM	0.38	10% & 90%	9	492.6716	10.7256%	-
	HC	0.31	100% & 0%	-	-	1.0726%	-
	MDBC	0.28	60% & 40%	9	492.6716	39.6485%	-121.15875
PC3	KM	0.24	70% & 30%	7	414.2303	35.6444%	-
	HC	0.16	100% & 0%	-	-	12.5333%	-
	MDBC	0.17	64% & 36%	7	414.2303	38.6667%	-149.61368
PC4	KM	0.61	64% & 36%	16	703.9360	45.8899%	-
	HC	0.39	100% & 0%	-	-	12.7949%	-
	MDBC	0.39	38% & 62%	16	703.9360	33.1665%	-126.11086
PC5	KM	2.82	38% & 62%	5	3689.6454	35.2097%	-
	HC	1.94	100% & 0%	-	-	2.941%	-
	MDBC	2.29	39% & 61%	5	3689.6454	35.892%	-126.46888

In 2014, Shepperd Bowes and Hall [6] studied the largest factor from researcher group, data set family, input metrics and classifier family influencing to the defect-prone software predictive performance. They apply ANOVA model and meta-analysis to 42 primary studies consisting of 600 instances with their classifier as shown in Table 2.7. The results as shown in Table 2.8 surprise the researchers that the classifier family has lowest effect on the predictive performance while the researcher group has highest affection which is 31%. So, they suggested the way to overcome bias from researcher by do the blind analysis, develop better reporting protocols and require intergroup studying.

Table 2.7 Frequency of instances by classifier family [6].

	Instances	Percent
DecTree	172	28.7
Regression	136	22.7
Bayes	124	20.7
CBR	77	12.8
Search	41	6.8
ANN	28	4.7
SVM	17	2.8
Benchmark	5	0.8

Table 2.8 Partial Eta-Squared Values for the ANOVA Model [6].

Factor	Partial η^2	Significance
Researcher Group	31.01%	$p < 0.0001$
Dataset Family	31.00%	$p < 0.0001$
Input Metrics	12.44%	$p < 0.0001$
Classifier Family	8.23%	$p < 0.0001$

In 2014, Shepperd Bowes and Hall [6] studied the largest factor from researcher group, data set family, input metrics and classifier family influencing to the defect-prone software predictive performance.

However, there are many classification studies. Some studies provided their own prediction model while many studies utilized a prediction model based on published algorithms such as Neural Network or Support Vector machine. Also, the Assembly Induced defect is considered specific defect type, having unique shape, and there is no study regarding this defect classification and prediction.

In this thesis, we studied the possible parameter related to AID and the existing possible classification algorithms to classify this defect. Finally, we identify the best existing classification model for the AID prediction based on the highest accuracy. It also could be applied in the normal practice of HDD test process. Hence, the defect can be efficiently classified beforehand during the test

CHAPTER 3

THEORY

All of classification algorithms have same purpose which can think and provide the result like human being. This chapter introduces the 4 classification algorithms which are Neural Network, Naïve Bayes, Decision Tree and Random Forest. The concept of each one are as follow.

3.1 Neural Network

The neural networks concept has got the inspiration from the complexity of animal brain consisting of the multiple sets of neurals. Figure 4.1 presents the real neural and artificial neural model. For the real neural, it uses abandon dendrites for input information congregation and becomes a nonlinear response. Axon brings them to other neurals when reaches the threshold. For artificial neural model, each input (x_i) is combined via a combination function, for instant summation function (Σ), before produce the output (y) for further downstream to other neurals

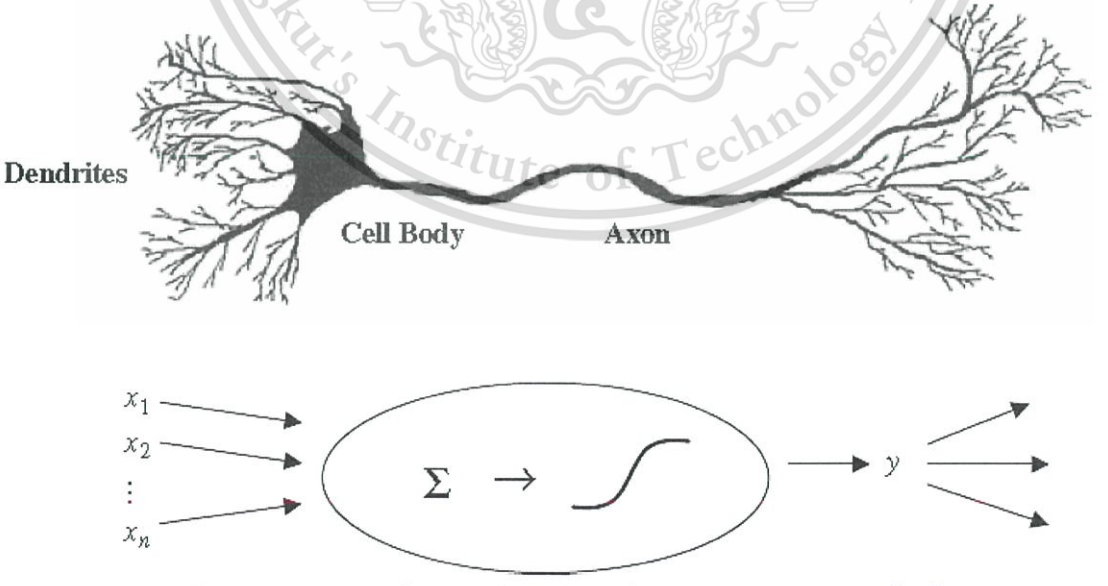


Figure 3.1 Actual neural and artificial neural model [11].

3.1.1 Single layer perceptron networks

The single layer perceptron networks can be presented in model of figure 3.2. Each output consists of the summation of each input multiply with their weight [7].

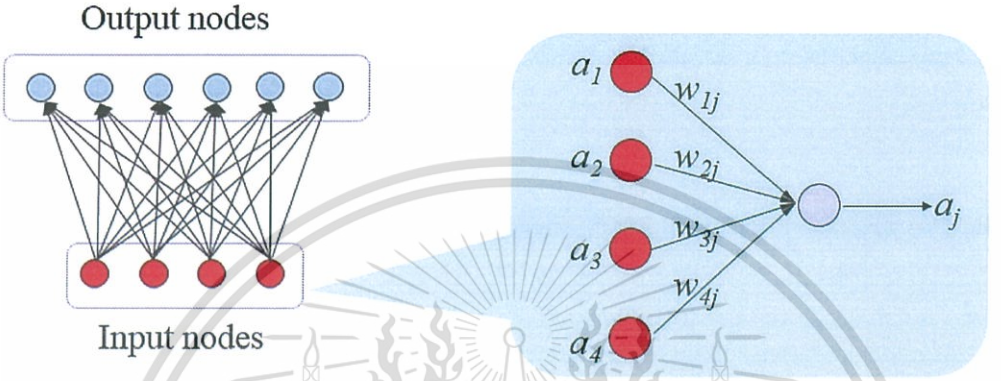


Figure 3.2 Single layer perceptron chart [8].

The output from model in figure 3.2 can also be described in equation (3.1) below [8].

$$a_j = g \sum_{i=1}^{n} (w_{ij} a_i) \tag{3.1}$$

Where w_{ij} is the connection weight of branch (i,j), a_i is the input data of node i, and g is the activation function.

The number of input nodes depend on the number of input data components and the activation function can be threshold function (3.2) or sigmoid function (3.3). If the output is discrete output: 'yes' or 'no', the output function is the threshold function. On the other hand, if the output is continuous output, the output function is sigmoid function.

$$f(x) = \begin{cases} 1, x > 1 \\ 0, x \leq T \end{cases} \tag{3.2}$$

Where T is the threshold level

$$f(x) = \frac{1}{1 + e^{\alpha x}} \tag{3.3}$$

3.1.2 Multilayer perceptron Networks

Since the single threshold unit could not solve all of the problem such as XOR case of Figure 3.3 example, the multilayer perceptron networks was introduced.

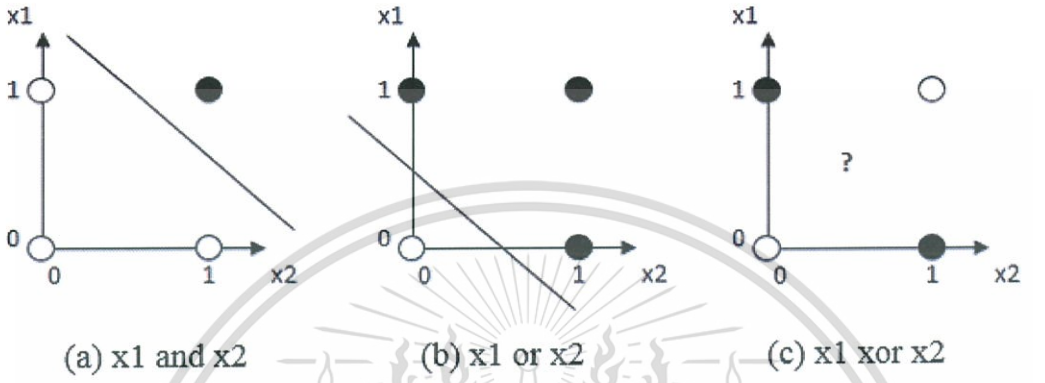


Figure 3.3 (a) result from 'and' function can be classified by linear function, (b) result from 'or' function can be classified by linear function, and (c) there is no linear function can classified result from 'xor' function [8].

The multilayer perceptron can be descripted in Figure 3.4 which consists of many hidden layer. The output of each layer which is the input of next layer can also be calculated by formula (3.1).

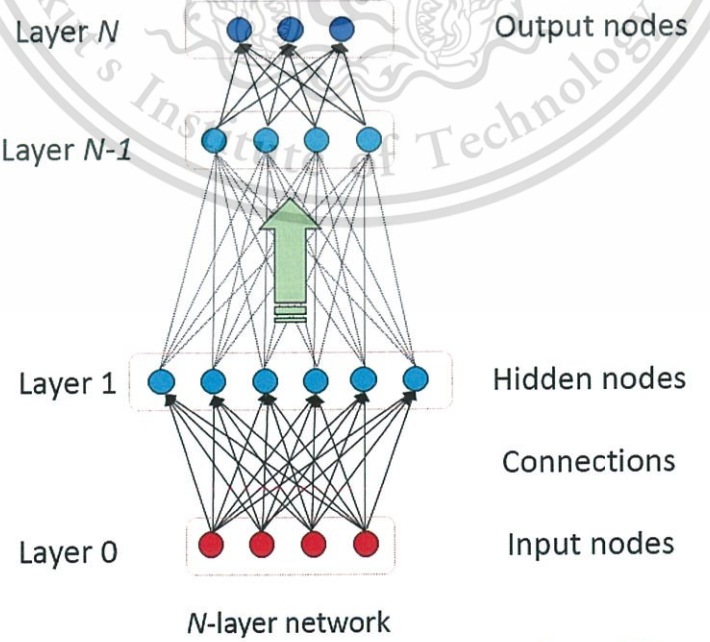


Figure 3.4 The multilayer perceptron chart [8].

3.1.3 Back-propagation

Neural network requires a large training set of known results for learning method. Once the output value from each observation is provided from the output node, it compares to the actual value and the errors are calculated. The sum of square error as equation (3.4) is used to measure the effective of output prediction. The objective is to minimize the SSE for predict function [11].

$$SSE = \sum_{Records} \sum_{Output\ nodes} (actual - output)^2 \tag{3.4}$$

3.1.4 Gradient Descent Method

This method can help to minimize SSE by providing the direction for each weight adjustment. The method can be provided from equation (3.5) [11].

$$\nabla SSE(W) = \left[\frac{\partial SSE}{\partial w_0}, \frac{\partial SSE}{\partial w_1}, \ldots, \frac{\partial SSE}{\partial w_m}\right] \tag{3.5}$$

Where $w_0, w_1, .., w_m$ is weight of each node and SSE is sum of square error. This gradient descent method can be illustrated in Figure 3.5. If the current weight value $w_{current}$ is w_{1L} with negative slope value, increasing weight value will be the right method to minimize SSE purpose. On the other hand, if the current weigh value $w_{current}$ is w_{1R} with positive slope value, decreasing weight value will be the right method to minimize SSE purpose. Hence, the weight can be adjusted according to equations (3.6) and (3.7).

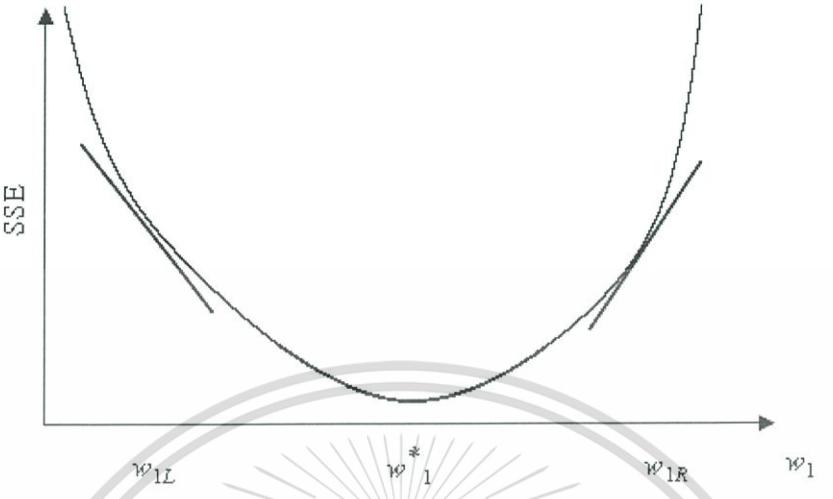


Figure 3.5 The slope of SSE for weight adjustment [11].

$$w_{new} = w_{current} + \Delta w_{current} \tag{3.6}$$

where

$$\Delta w_{current} = -\eta \frac{\partial \text{SSE}}{\partial w_{current}} \tag{3.7}$$

The η is the learning rate which have value between 0 and 1

This Gradient Descent method likes the blinding man walk into the direction against their slope to reach the destination, consequently, the weak point of Gradient Descent method is the destination could be at local minima which does not be the real minimum point rather than global minimum. According to figure 3.6, A and C is the local minima while B is the global minimum.

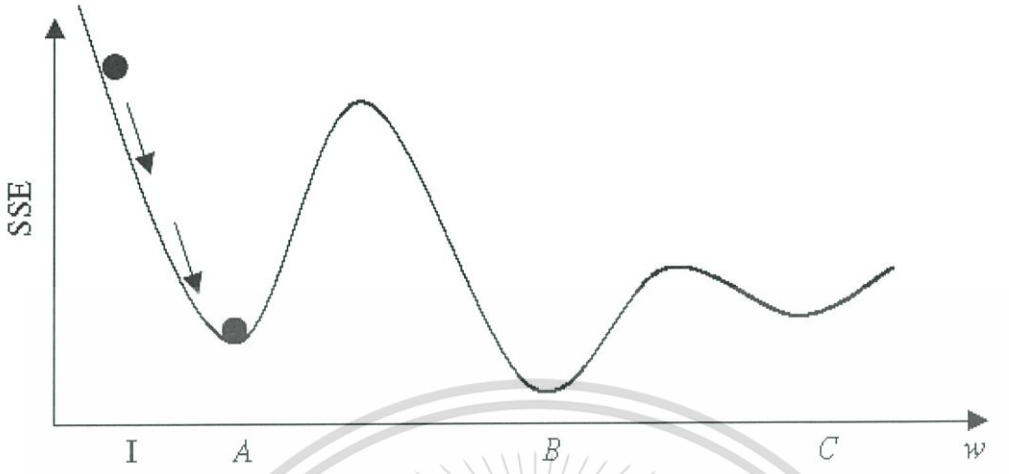


Figure 3.6 The local minimum and Global minimum [11].

3.1.5 The training perceptron network.

This method for perceptron network training can be illustrated in step a to e as below.

a) Insert the input ($a_1, a_2, \ldots$) into the network.
b) Calculate network output.
c) Measurement error according to equation (3.2).
d) Adjust weight according to equation (3.4).
e) Back to step a) until the error is in the acceptable value.

3.2 Naïve Bayes

The classification model is presented in form of probability function based on Bayes's theorem illustrated in section 3.2.3. For section 3.2.1 and 3.2.2, these explain the meaning of each significant probability variable applied in Bayes's formula.

3.2.1 Conditional Probability

The conditional probability, P(A|B), is the probability of A given that the event B has occurred. The conditional probability formula is denoted by equation (3.8) [9].

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{3.8}$$

Where $P(A \cap B)$ is the probability of event A is happen with event B, and $P(B)$ is the unconditional probability of event B occur.

3.2.2 Probability Density Function.

The probability density function of X denoted f(x) is the resulting of probability of all possible x value. If x is discrete, the f(x) will be histogram. Otherwise, f(x) will be continuous chart as shown in Figure 3.9. The area under probability density function chart is 1.

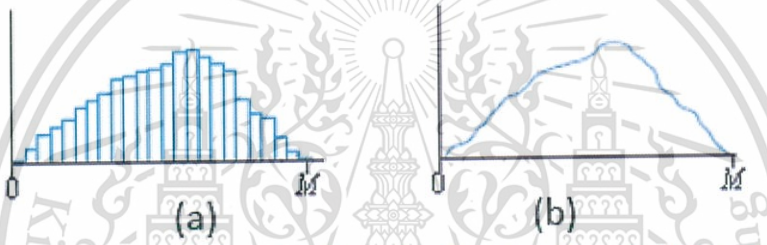


Figure 3.7 (a) The discrete probability density function. (b) The continuous probability density function so called density curve [9].

Hence, the definition of probability density function of x taken on a value between a and b is the area under the density function and can be defined in formula (3.9)[9].

$$P(a \leq X \leq b) = \int_a^b f(x)dx \tag{3.9}$$

3.2.3 Bayes Theorem and Naïve Bayes.

The Bayes' formula consists of both conditional probability and probability density function as shown in equation (3.10) and (3.11)[10].

$$P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_j)}{p(x)} \tag{3.10}$$

Or can be explained in form of English as

$$posterior = \frac{likelihood\ x\ prior}{evidence} \tag{3.11}$$

Where $P(\omega_j|x)$ is probability of ω_j given that the event x, $p(x|\omega_j)$ is probability density function of ω_j given that the event x which expressed in equation (3.12), and p(x) is probability density function of x.

$$p(x) = \sum_{j=1}^{N} p(x|\omega_j)P(\omega_j) \tag{3.12}$$

This Bayes' Theorem can be applied to Naïve Bayes' relation as in equation (3.13).

$$p(\omega_k|x) \ \alpha \ \prod_{i=1}^{d} p(x_i|\omega_k) \tag{3.13}$$

3.3 Decision Tree

The prediction model can be presented in form of tree structure consisting of decision nodes and branches. Each node is connected by a branch and extending downward until completing in leaf nodes as shown in Figure 3.8. The Decision Trees can classify only discrete target attribute and require large numbers of data set training [11][12].

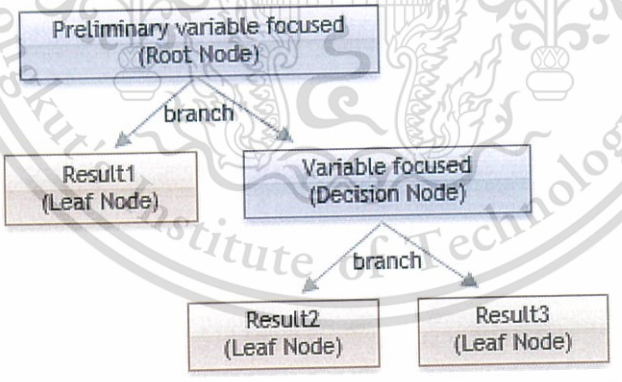


Figure 3.8 The decision tree structure [11][12].

3.3.1 Attribute node selection criteria

The attribute node selection criteria of decision tree model can be identified based on the information gain value. The information gain of node (D) can be described in equation (3.14)[11]. The node providing maximum information gain is selected to be the node of each tree level.

$$Gain(A) = Info(D) - Info_A(D) \tag{3.14}$$

The $Info(D)$ and $Info_A(D)$ can be calculated from equations (3.15)-(3.16) here.

$$Info(D) = \sum_{i=1}^{m} p_i \log_2(p_i) \tag{3.15}$$

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} * Info(D_j) \tag{3.16}$$

Where p_i is the probability of each overall result. The $\frac{|D_j|}{|D|}$ is the weight at j^{th} partition and $Info(D_j)$ is the information from sub value of node D.

3.3.2 Branch split criteria

For attribute contained discrete value, the branch splitting criteria can be identified based on element of each node. For attribute contained continuous value, the branch splitting criteria can be identify based on the split-point value. The split-point value of attribute A required sort element in increasing order. Then their mid-point is considered to be the best split-point which can be calculated from equation (3.17)[11].

$$split\ point = \frac{a_i + a_{i+1}}{2} \tag{3.17}$$

3.4 Random Forest

The Random Forest (RF) the algorithm applying several weak decision trees onto a training dataset for the optimal solution. The prediction model can be presented in form of the average of multiple trees structure. Each tree can be created as following sequences [13].

a) At training data set, create sample of bootstrap Z^* with N sizes.

b) Use recursive repeating generation of each random-forest tree T_b until the minimum node size n_m is obtained.

c) Repeat a to b for number of trees setting (B). After that the set of multiple trees $\{T_b\}_1^B$ is taken place. The regression function can be presented in equation (3.18).

$$f_{rf}^B(\mathrm{x}) = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \tag{3.18}$$

d) The selected classification is highest vote of each class prediction from the forest as in equation (3.19).

$$C_{rf}^B(\mathrm{x}) = majority\ vote\{C_b(x)\}_1^B \tag{3.19}$$

where $C_b(x)$ is the class prediction the random-forest at b^{th}.

CHAPTER 4
RESEARCH METHODOLOGY

This research methodology consists of three steps including the measurement tool, data pre-processing, and the validation process. The new software written by python language and Rapid minor application were used in the data-preparation and analysis process. This chapter will present the step of measurement tool and the setting up procedure, the assumption on material and method, how to collect and prepare the desired data, how to evaluate based on various condition and the expectation results of this experiment. The procedure to measure the accuracy will be presented.

4.1 The measurement tool

4.1.1 Hardware measurement.

The hardware measurement used in this experiment is hard disk drive tester as shown in Figure 4.1. The chamber test consists of many slots to test each individual drive. At the start of validation, each drive is inserted into a test slot by an automated system. Then the slot starts testing that drive. The slot voltage, slot environment temperature and fan speed setting are same as in normal drive factory test process, because the validation have to use same environment as real process. The drive is performed based on the sequence provided in software which will provide detail in Section 4.1.2.

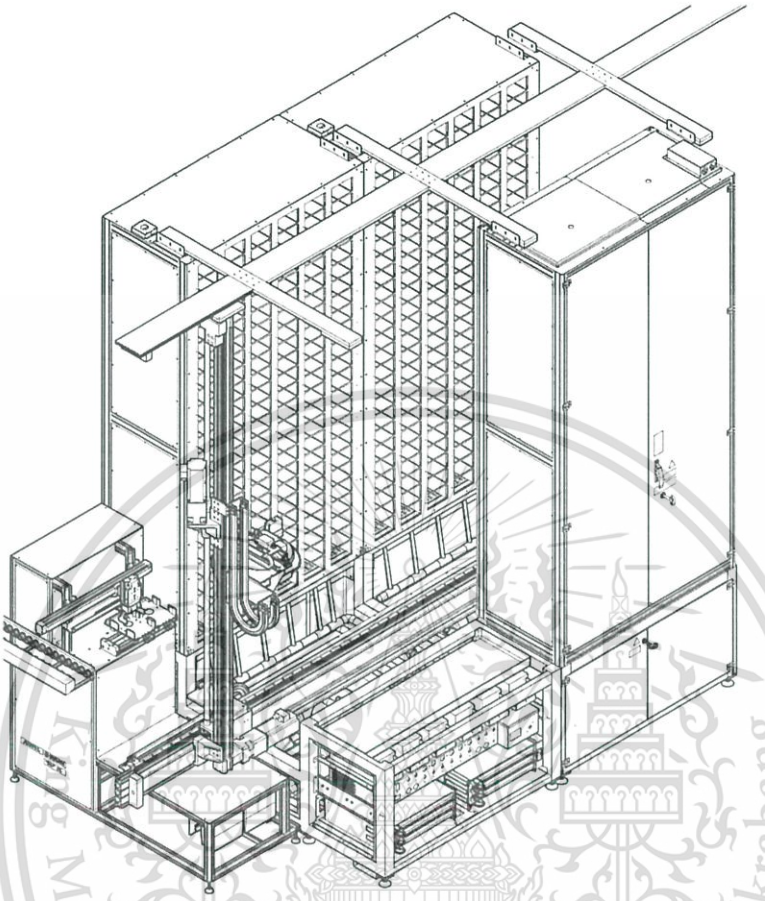


Figure 4.1 Hard disk drive tester.

4.1.2 Software measurement.

For the original test script, there are several sequence test such as initialization, MDW until Defect Scan test which can capture all of defects including defect count by head zone and maximum defect size per surface. These information is necessary to be used as input into each model later on. However, the time to test Defect Scan, 80 hr., is considered too long to obtain the information and detect the problem. Therefore, the validated script was modified from the original used in normal factory by adding new special defect scan sequence at the earliest of possible state of the HDD test as shown in Figure 4.2. In this case, it has been added at the time about 19 hours. There are 2 reasons of this additional sequence. One is the information required for analysis, defect count by head zone and maximum defect size per surface, can be retrieved in normal defect scan test which is performed at the time of 80 hours. The time is considered too long to get failure information and detect the problem. And the other is we plan to put the selected

model based on the analysis results here to identify non-AID or AID for sooner reject AID drive.

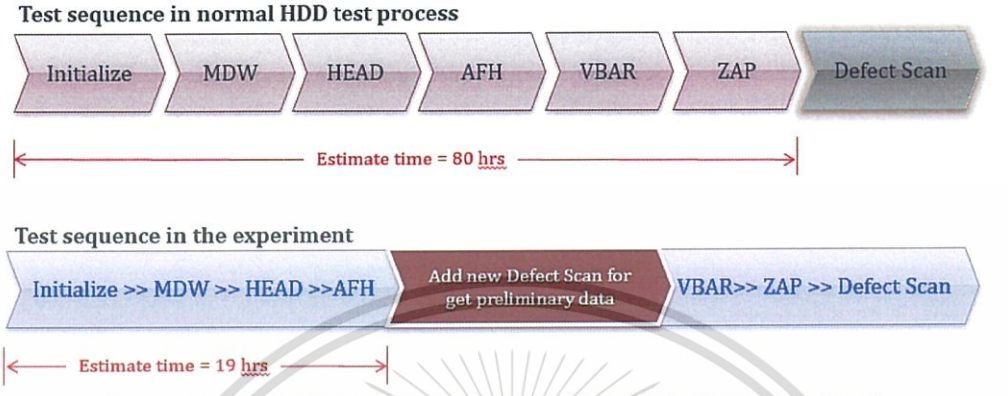


Figure 4.2 Additional defect scan for quick data obtained.

4.1.3 Hard Disk Drive material.

With complete both equipment and modified test script, the drive can be operated to collect the information in this experiment. These are composed of 8 groups. Each group has 100, 200... 800 pcs consisting of AID and non-AID drives as detail in table 4.1. After each drive complete the validation, the result of all test sequence such as MDW, HEAD calibrate, AFH or new defect scan were accumulated into one text document. It is called the "result file". Therefore, number of result file is equal to the number of validated drive which has the maximum to 800 files from the group with maximum drive quantities testing.

Table 4.1 Detail of validated drives of each group.

Group	Sample size	non-AID	AID
1	100	50	50
2	200	124	76
3	300	224	76
4	400	324	76
5	500	424	76
6	600	524	76
7	700	624	76
8	800	724	76

4.2 Data Pre-processing

Since the result files contain whole of drive information and consist of many text document, the data pre-processing is required to retrieve only the interested information and gather into 1 document of each group. The information focused on is defect count by head zone and maximum defect size per surface and be part of the result file. The new software specifically written for this study was created. Because the focused data are in different format, it requires totally 2 new software. One is for retrieving defect count by head by zone information and another one is for retrieving defect size information. Both can filter each of result file, retrieve the target information and translate into each record of excel file. Then, the final excel file is comprised of the multiple record which is the information of drive in each group. For instant, the number of record in 100 pcs group is 100 record and each record contain the interested information of each drive.

Defect Count Info:

PHY_HD_NO	ZN_NO	LOG_HD_NO	DEF_CNT	STATUS
0	0	0	10	1
0	1	0	0	1
0	2	0	0	1
0	3	0	2	1
0	4	0	1	1
0	5	0	25	1
0	6	0	0	1
0	7	0	26	1
0	8	0	5	1
0	9	0	0	1
0	10	0	0	1
0	11	0	8	1
0	12	0	1	1
0	13	0	0	1
0	14	0	0	1

Figure 4.3 The raw data of defect count by head-zone from result file.

Defect Size info:

PHY_HD_NO	START_TRK	HD_LGC_PSN	ENDING_TRK	DEF_SIZE	STATUS
9	50788	9	50796	14	P
0	51020	0	51026	11	P
8	55262	8	55272	21	P
5	60024	5	60034	26	P
6	64276	6	64282	21	P
7	64322	7	64326	9	P
0	88702	0	88706	11	P
9	93350	9	93356	12	P
2	94056	2	94064	21	P
1	96668	1	96676	38	P
0	105716	0	105720	7	P
5	108782	5	108788	11	P

Figure 4.4 The raw data of size of defect from result file.

For defect count of each head zone information, it is provided in part of the result file as in figure 4.3. The data of each head and zone are in DEF_CNT column, as a result, this special program can directly parse the information and translate it into excel file before identify AID or non-AID of each record.

For the maximum defect count information, it is provided in part of the result file as in figure 4.4. Unfortunately, there is no exactly data for maximum defect count per surface to parse but the result file provides only the size of each defect in DEF_SIZE column. Consequently, the special program must calculate the maximum defect size after parse the information. After that it is translated into excel format and latter label drive status into each record as AID or non-AID.

SN	Hd0_Zn0	Hd0_Zn1	Hd0_Zn2	Hd0_Zn3	...	Hd9_Zn15	Hd9_Zn16	Hd9_Zn17	Drive Status
Drive no.1	0	3	0	2	...	1	0	0	non-AID
Drive no.2	32	535	2954	372	...	437	0	394	non-AID
Drive no.3	0	13	0	5	...	0	332	620	AID
Drive no.4	52	46	26	5	...	19	0	4	non-AID
Drive no.5	0	3	8	2	...	0	0	2	AID
Drive no.6	0	3	1	2	...	8	0	3	non-AID
:	:	:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:	:	:
Drive no.100	0	4	175	11	...	0	0	2	non-AID

Figure 4.5 The translated data in Microsoft excel document of defect count by head zone from 100 sample sizes.

SN	Hd0	Hd1	Hd2	Hd3	...	Hd7	Hd8	Hd9	Drive Status
Drive no.1	0	0	0	16	...	41	41	0	non-AID
Drive no.2	8	15	10	47	...	18	15	9	non-AID
Drive no.3	26	0	10	0	...	2071	11	82	AID
Drive no.4	7	0	0	0	...	0	0	26	non-AID
Drive no.5	16	14	473	42	...	4	10	0	AID
Drive no.6	34	64	8	19	...	0	0	12	non-AID
:	:	:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:	:	:
Drive no.100	0	49	8	0	...	0	14	37	non-AID

Figure 4.6 The translated data in Microsoft excel document of maximum defect size by surface from 100 sample sizes.

After new excel format generated, the prepared data are presented in excel format together with AID or non-AID label. Figure 4.5 presents the translated data example of defect count of each head zone from the 100 sample sizes group. The product used to validate has total 10 headers and 18 zones, as a result, the translated data consists of defect count value from head 0 zone 0 until head 9 zone 17 of all 100 sample drives. Besides, figure 4.6 presents the translated data example of maximum defect size by surface from the 100 sample sizes group. Since the information is required by surface, the maximum defect size data, calculated from special parse program, are presented from head 0 until head 9 in translated data only. These 2 prepared data are ready for further analysis.

4.3 The validation process

The experimental process consists of operate HDD based on special test sequence in prepared tester, gather the result file of each drive, translate to excel file of each information and analysis based on interested classification model with various conditions. The figure 4.7 presents the experimental condition flow. The prepared data is the first process consisting of 16 translated excel files which are the defect count by head zone of 100 pcs, 200 pcs... 800 pcs, the maximum defect size by surface of 100 pcs, 200 pcs... 800 pcs. Both defect count by head zone and maximum defect size by surface of same quantity group are retrieved from same drive. For example, the defect count by head by zone of 100 pcs group come from same source as defect size of 100 pcs group.

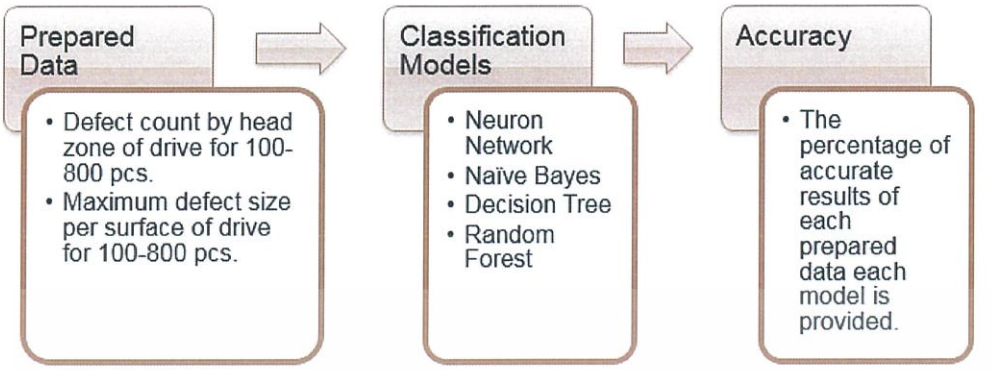


Figure 4.7 The experimental condition flow.

Next, each of translated data fed into the 4 different classification algorithms: Neural Network, Naïve Bayes, Decision Tree and Random Forest. In the classification model process, the input data are divided into 2 groups supporting 2 micro processes. The first micro process is training process which use the data of first group as learning material (called training sets) based on the selected model and then the unique prediction model formula is provided. This formula is applied to the data in the second group (called testing sets) for micro testing process and then the experimental accuracy are calculated and presented [14]. The ratio between training and testing data sets called validation numbers. In this case, 4 and 10 value are used. The objective for different number of validation is we expect to see how the ratio between training group and testing group effect on the accuracy. The definition of number of validation (k) is the number of divided data sets used in both the training process and testing process. The k-1 sub data sets are applied as training data set while another sub data set is used as testing data set.

According to the experimental condition, the experimental accuracy are 128 values which come from 16 (number of prepared data) x 4 (number of classification model) x 2 (validation 4 and 10). The assumption would see higher accuracy on higher data set of 800 sets and higher validation numbers. Eventually, the model generating the highest accuracy greater than 95% is selected to be the best model of this AID prediction.

CHAPTER 5
EXPERIMENTAL RESULTS

This chapter presents the prediction model of all target classification algorithms: Neural Network, Naïve Bayes, Decision Tree and Random Forest. Also, their experimental accuracies. The classification algorithm providing the highest accuracy and greater than 95% is considered to be the best algorithm. Moreover, we can see the trend of accuracy with different sample size to identify the minimum sample size for the accurate stabilization.

For all of the experimental accuracies, these can be segregated into 16 criteria groups based on 2 different input information: defect count by head zone, defect size per surface, 2 different validation number: 4 or 10, 4 different classification algorithm: Neural Network, Naïve Bayes, Decision Tree and Random Forest. Also, each criteria condition group has 8 sub-criteria by sample size starting from 100, 200...800. The results of each criteria which presented in appendix B are comprised of percentage accuracy of each class (class recall), the percentage accuracy of each prediction (class precision) and the total accuracy.

5.1 The results from Neural Network algorithm.

5.1.1 The parameter setting from Neural Network algorithm.

This parameters setting of Neural Network algorithm consist of training cycles: the cycle count for weight learning, learning rate: the amount of weight change of each stop, momentum: the fraction added of previous update to the current one to prevent local maxima and error threshold: the threshold value to stop weight adjustment. The value settings of each one are presented in Table 5.1.

Table 5.1 The parameters setting of Neural Network algorithm.

Parameters	Value
Training Cycles	500
Learning rate	0.3
Momentum	0.2
Error Threshold	1.00E-05

5.1.2 The prediction model from Neural Network algorithm.

The selected prediction model from Neural Network algorithm based on the highest accuracy of all criteria which are maximum defect size input, 800 sample sizes and number of validation 10 is presented in this section. The workflow to be Neural Network prediction model can be described in diagram of Figure 5.1. According to Figure 5.1, the input training data are used to calculate the coefficient of each node of Figure 5.2. Then this formula is applied into testing data set to come out the accuracy.

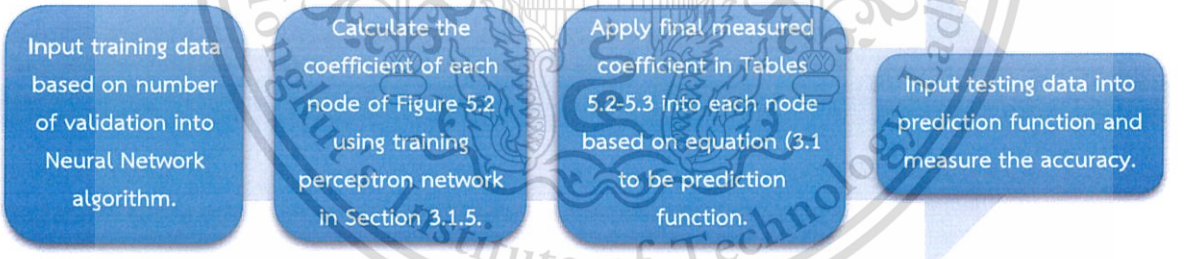


Figure 5.1 The diagram describes Neural Network workflow.

According to figure 5.2 and equation 3.1, it consists of the node of input layer and hidden layer linked to output data. The figure represents the Neural Network prediction model which consists of 11 input nodes. Each node input has different coefficient based on Table 5.2 to come out each hidden node. For example, hidden node 1 come from the equation 5.1. In the similar method, the output nodes, AID or non-AID can be calculated from hidden node which has coefficient from Table 5.3

$$hidden\ node1 = 2.188hd_0 - 2.702hd_1 + \cdots - 2.983hd_9 \quad (5.1)$$

where hd_0, hd_1,..., hd_9 represent the maximum defect count value of each header.

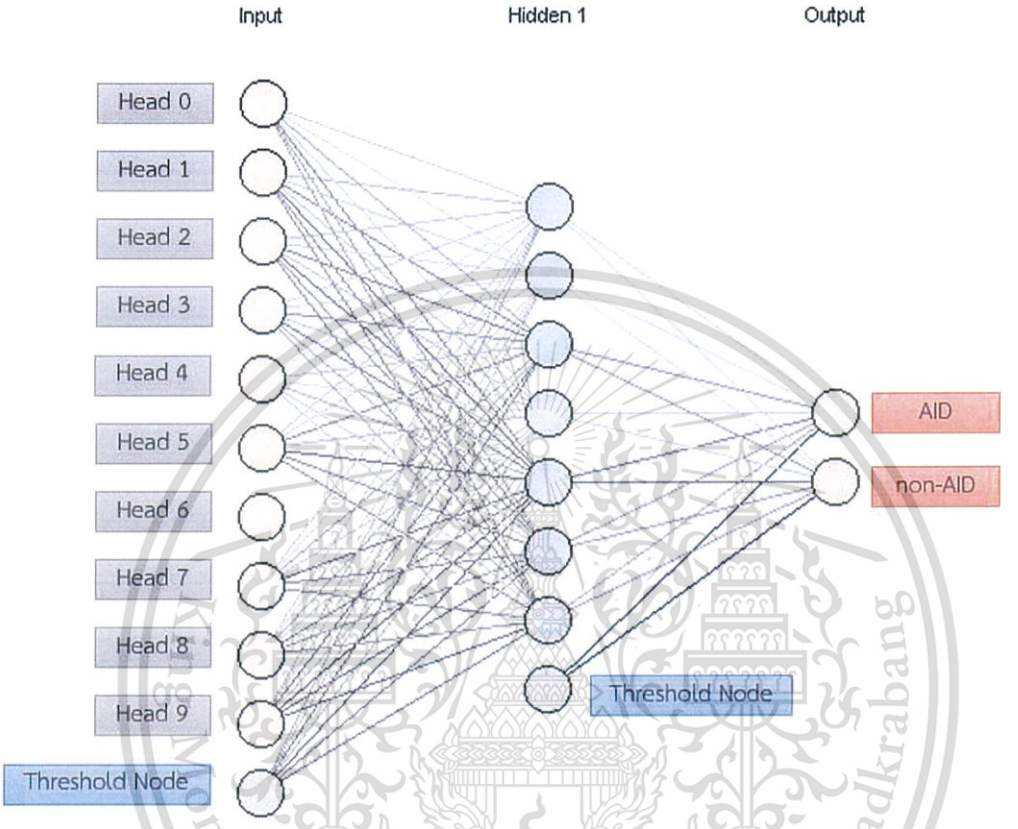


Figure 5.2 The multilayer perceptron model for AID prediction based on defect size input.

Table 5.2 Weight value of hidden node layer.

Node	Hidden Node1	Hidden Node2	Hidden Node3	Hidden Node4	Hidden Node5	Hidden Node6	Hidden Node7
Head 0	2.188	1.762	3.684	2.283	4.712	3.608	3.6
Head 1	-2.702	-2.236	-4.67	-2.775	-6.061	-4.561	-4.567
Head 2	-2.442	-1.925	-4.628	-2.616	-5.984	-4.547	-4.614
Head 3	2.104	1.71	3.784	2.338	4.875	3.664	3.691
Head 4	-1.541	-1.332	-2.479	-1.577	-3.066	-2.411	-2.406
Head 5	2.556	2.14	4.32	2.643	5.518	4.178	4.317
Head 6	-0.301	-0.246	-0.589	-0.316	-0.725	-0.571	-0.574
Head 7	2.529	2.116	4.073	2.575	5.205	4.114	4.008
Head 8	-2.305	-1.928	-3.887	-2.363	-4.992	-3.803	-3.773
Head 9	-2.983	-2.25	-5.657	-3.264	-7.092	-5.682	-5.683
Threshold	-3.332	-2.876	-5.292	-3.448	-6.57	-5.275	-5.262

Table 5.3 Weight value of output node layer.

Node	Non-AID	AID
Hidden Node 1	1.928	-1.931
Hidden Node 2	1.51	-1.518
Hidden Node 3	3.687	-3.702
Hidden Node 4	2.066	-2.069
Hidden Node 5	5.027	-4.998
Hidden Node 6	3.64	-3.644
Hidden Node 7	3.647	-3.658
Threshold	-9.917	9.921

5.1.3 The experimental accuracies from Neural Network algorithm.

The experimental accuracies from Neural Network algorithm of all criteria can be analyzed as in Figure 5.3. According to chart in Figure 5.3, all of input criteria provided the similar trend and the accuracy starting to stable at data size 600 onward with value about 95%. The defect size input provides significantly higher accuracy than defect count input while the different of number of validation provides similar results.

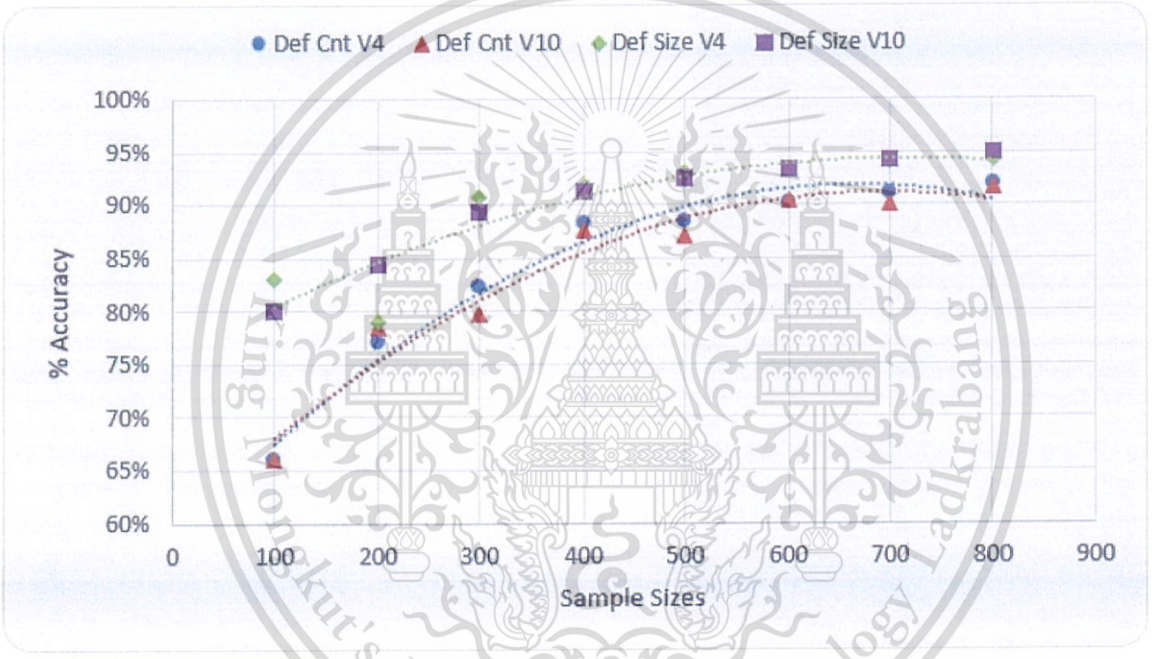


Figure 5.3 The accuracy trend from Neural Network algorithm.

5.1.4 The Sum Squared Error from Neural Network algorithm.

The sum squared error (SSE) fit curve against the training cycle trend is presented in Figure 5.4. According to Figure 5.4, the SSE values reduced when the training cycle increasing and start to stable at the training cycle reaching 380 onward with SSE value about 32.8. This final SSE value is considered high and lead to the results of accuracy lower than the target 95%. This high SSE value causes from the large range of the input data between 0 to 2 million. Moreover, the SSE trend of training cycle 500 onward close to 28 which similar to the SSE at training cycle 500. This can imply that increasing training cycle does not much effect on their accuracy based on the model condition as Table 5.1.

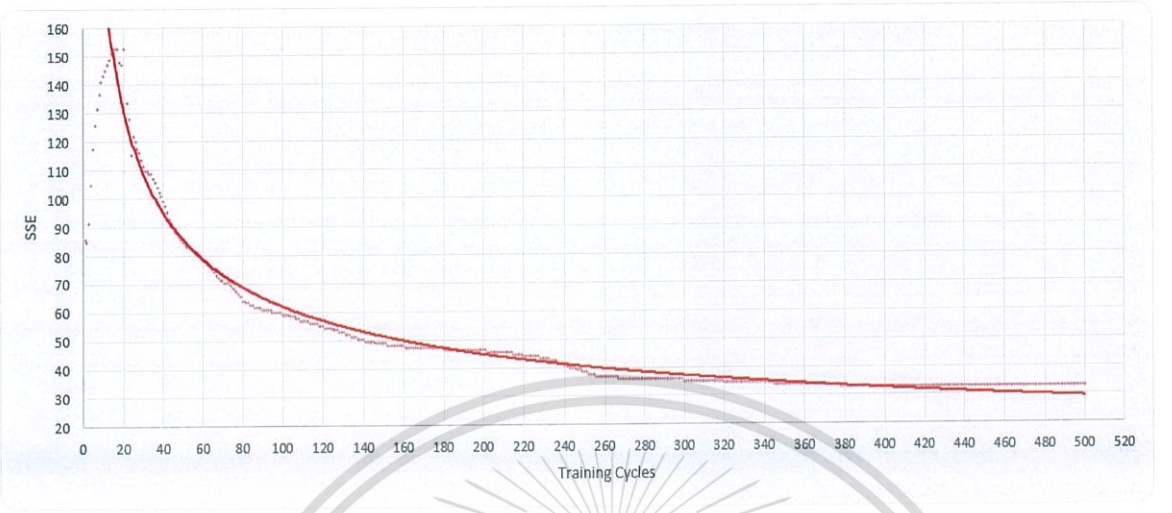


Figure 5.4 The Sum Squared Error (SSE) fit curve against the training cycle.

To achieve higher accuracy based on Neural Network algorithm, the more complexity of Neural Network condition is required. For instant, normalize the input data to reduce the range of defect size input before apply into training process, increase the number of node in the hidden later or increase the number of hidden layer. These can cause the higher CPU usage of HDD tester and lead to the higher the prediction test time when implement in the real test process.

5.2 The results from Naïve Bayes algorithm.

5.2.1 The prediction model from Naïve Bayes algorithm.

The selected prediction model from Naïve Bayes algorithm based on the highest accuracy of all criteria which are maximum defect size input, 800 sample sizes and number of validation 10 is presented in this section. The workflow to be Naïve Bayes prediction model can be described in diagram of Figure 5.5. According to Figure 5.5, the probability density function of each header given each output can be calculated based on training data set. These can be applied into testing data set to calculate P(AID|x) and P(non-AID|x) based on equation (3.11). The highest value is the prediction result.

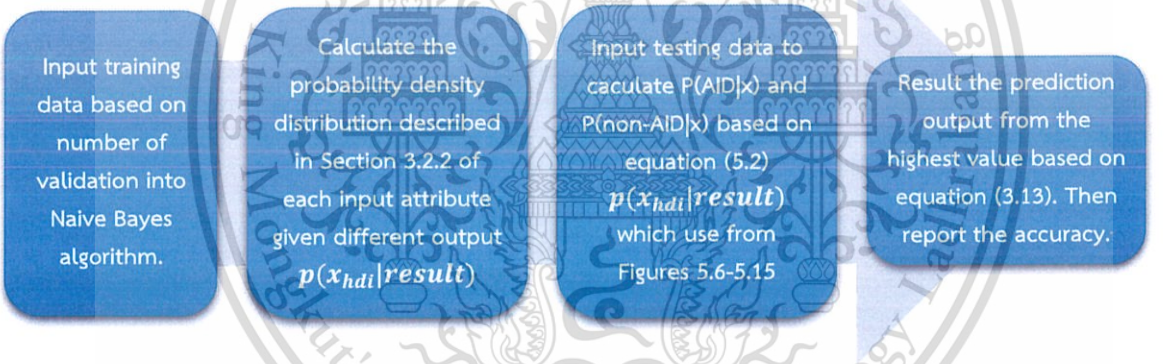


Figure 5.5 The diagram describes Naïve Bayes workflow.

To illustrated how P(AID|x) and P(non-AID|x) calculation, it can be described in step of a-c below.

a) Calculate P(AID|x) based on equation 3.11. In this case can be expressed in equation 5.2 below.

$$(AID|x) = p(x_{hd0}|AID) * p(x_{hd1}|AID) * \ldots * p(x_{hd9}|AID) \quad (5.2)$$

where x is maximum defect size, and $p(x_{hdi}|AID)$ can be retrieved from Figures 5.5-5.14 in blue line.

b) Similar to a, calculate P(non-AID|x) based on equation 3.13.

c) Predict the output: AID or non-AID based on the maximum value.

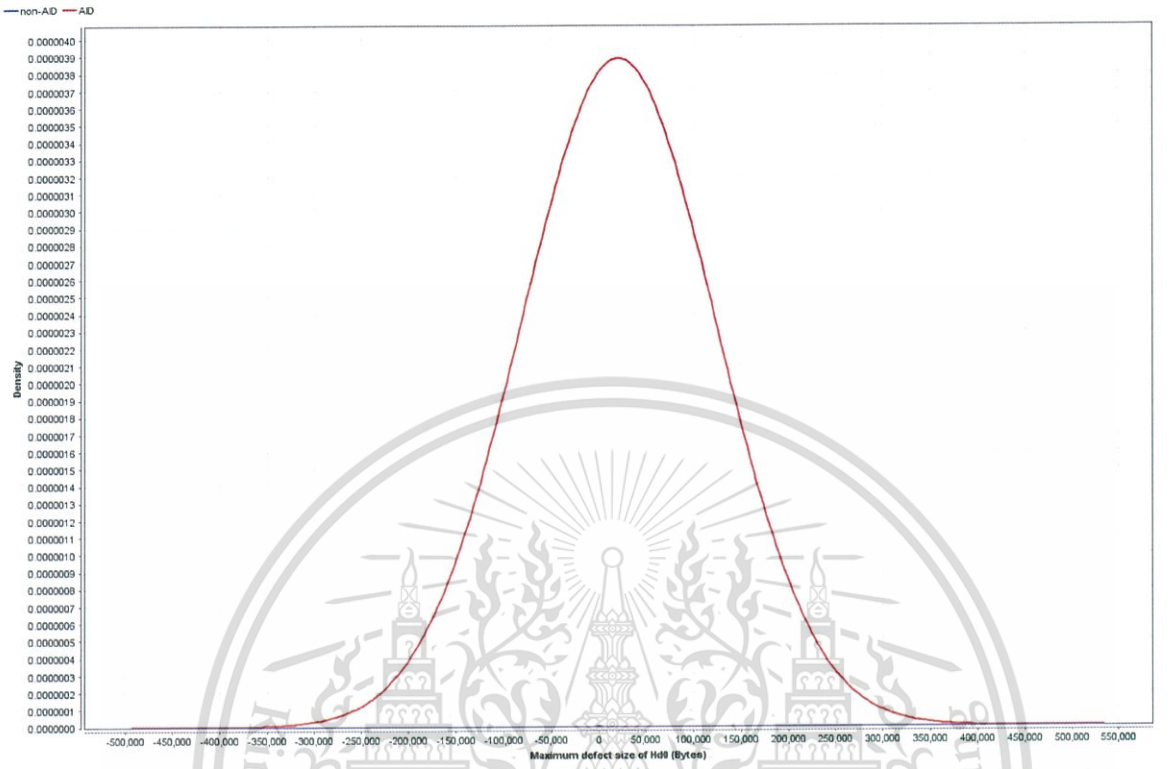


Figure 5.6 The probability density of each output (AID and non-AID) of head 0.

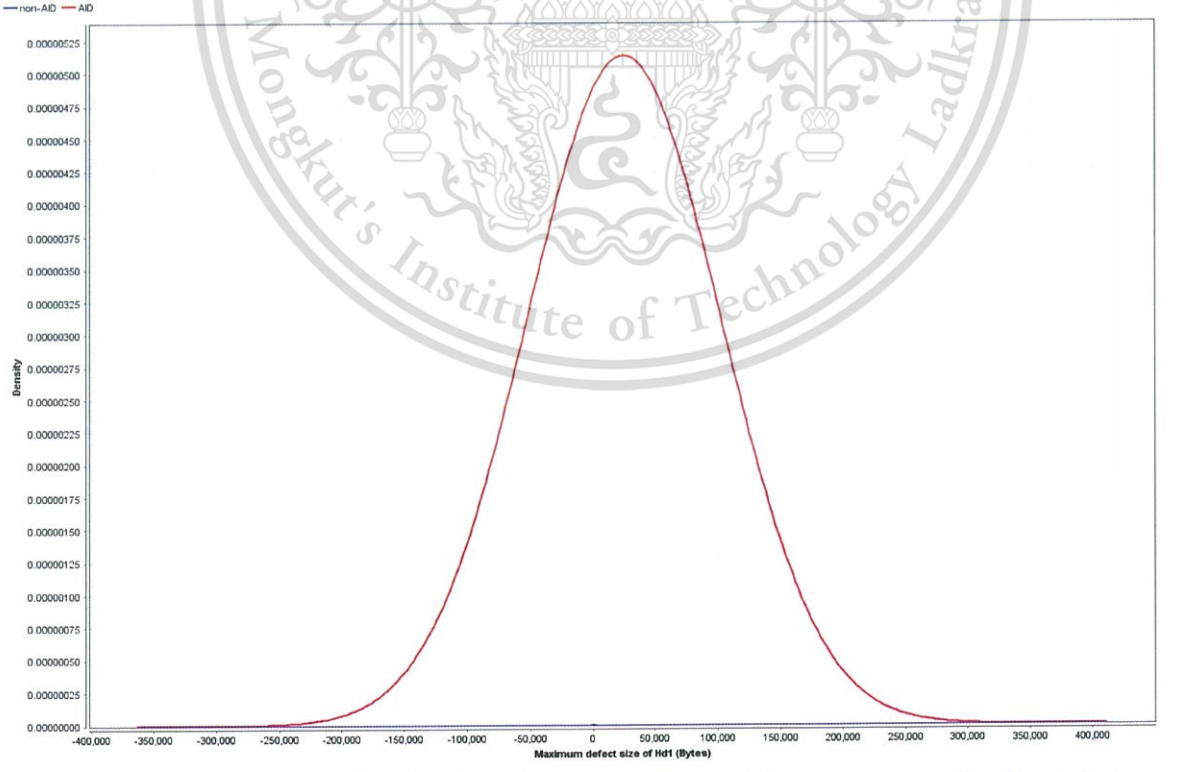


Figure 5.7 The probability density of each output (AID and non-AID) of head 1.

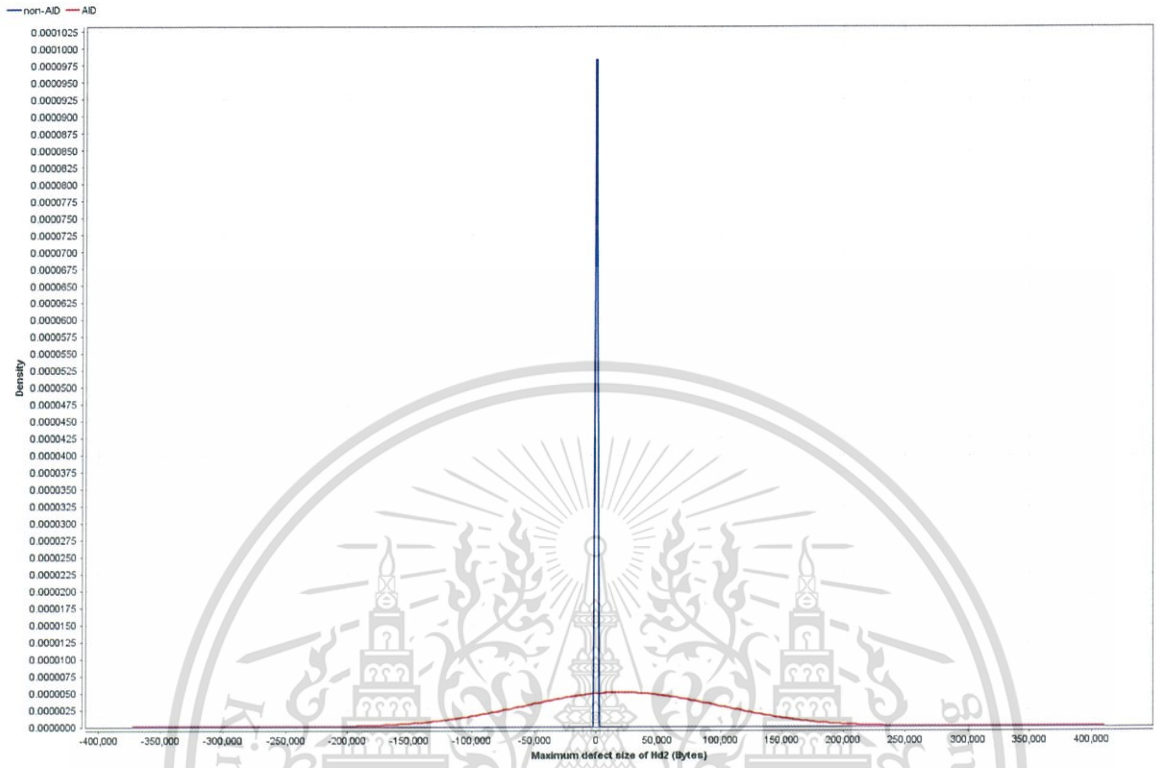


Figure 5.8 The probability density of each output (AID and non-AID) of head 2.

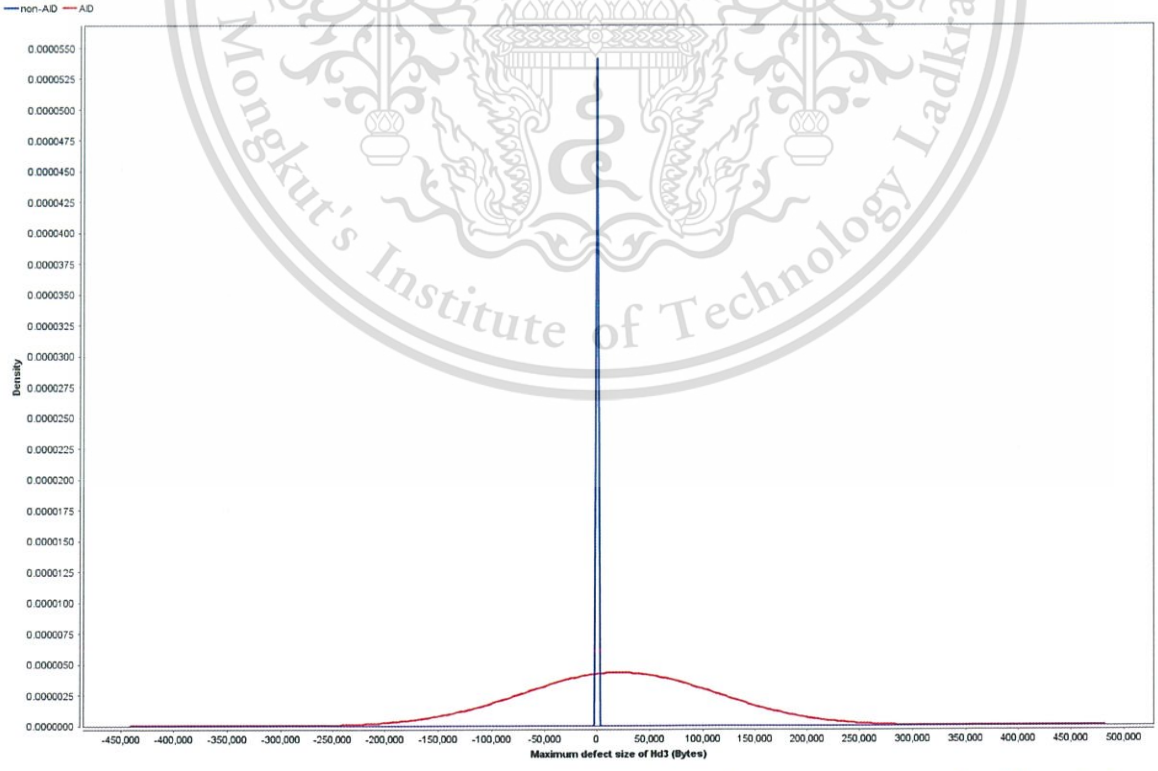


Figure 5.9 The probability density of each output (AID and non-AID) of head 3.

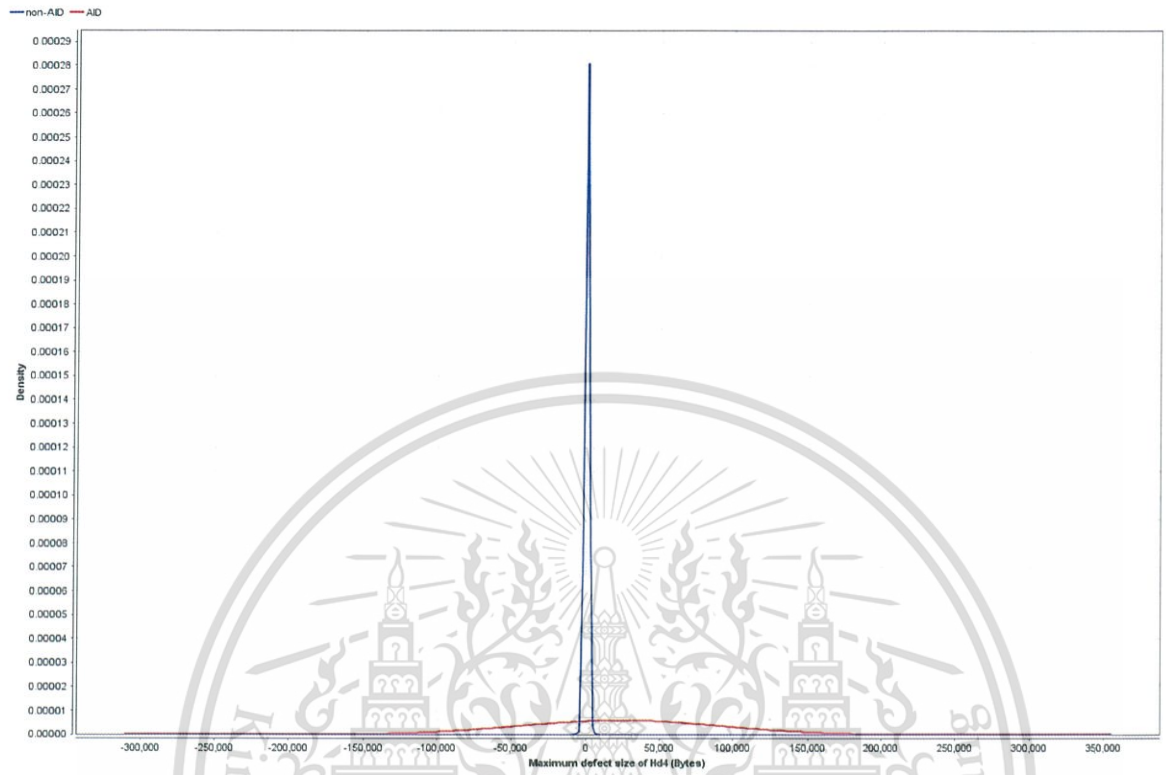


Figure 5.10 The probability density of each output (AID and non-AID) of 4.

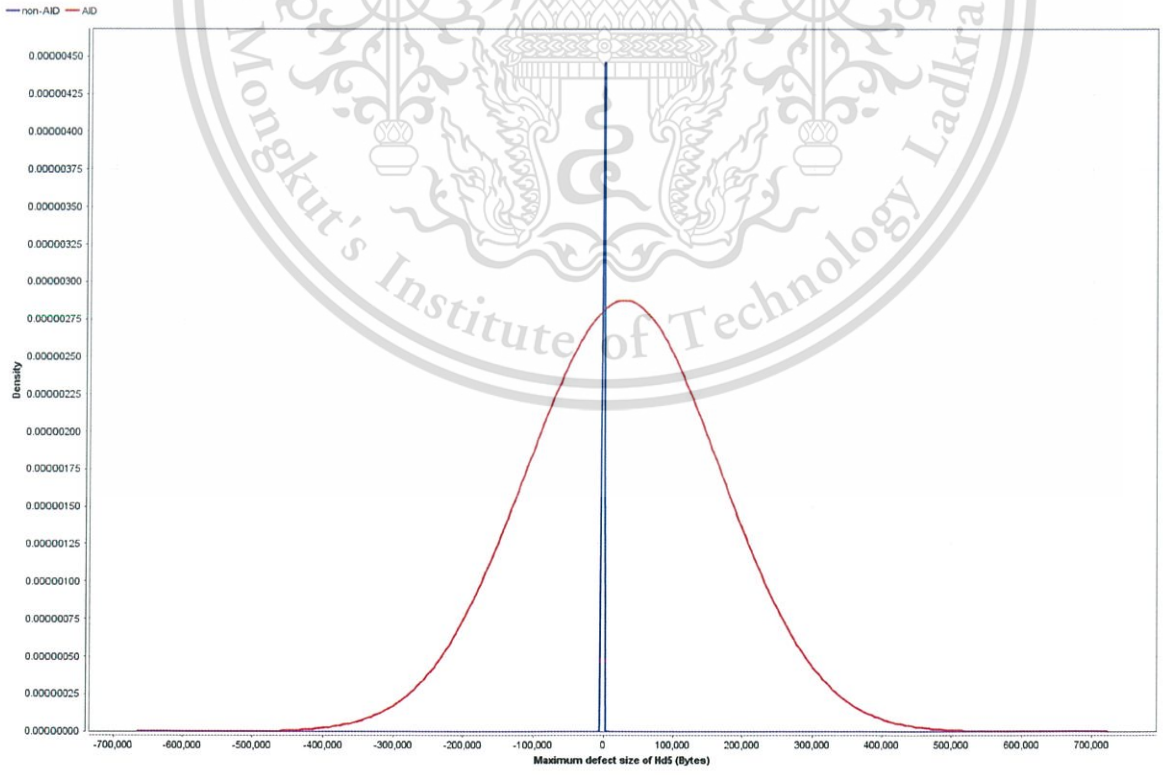


Figure 5.11 The probability density of each output (AID and non-AID) of 5.

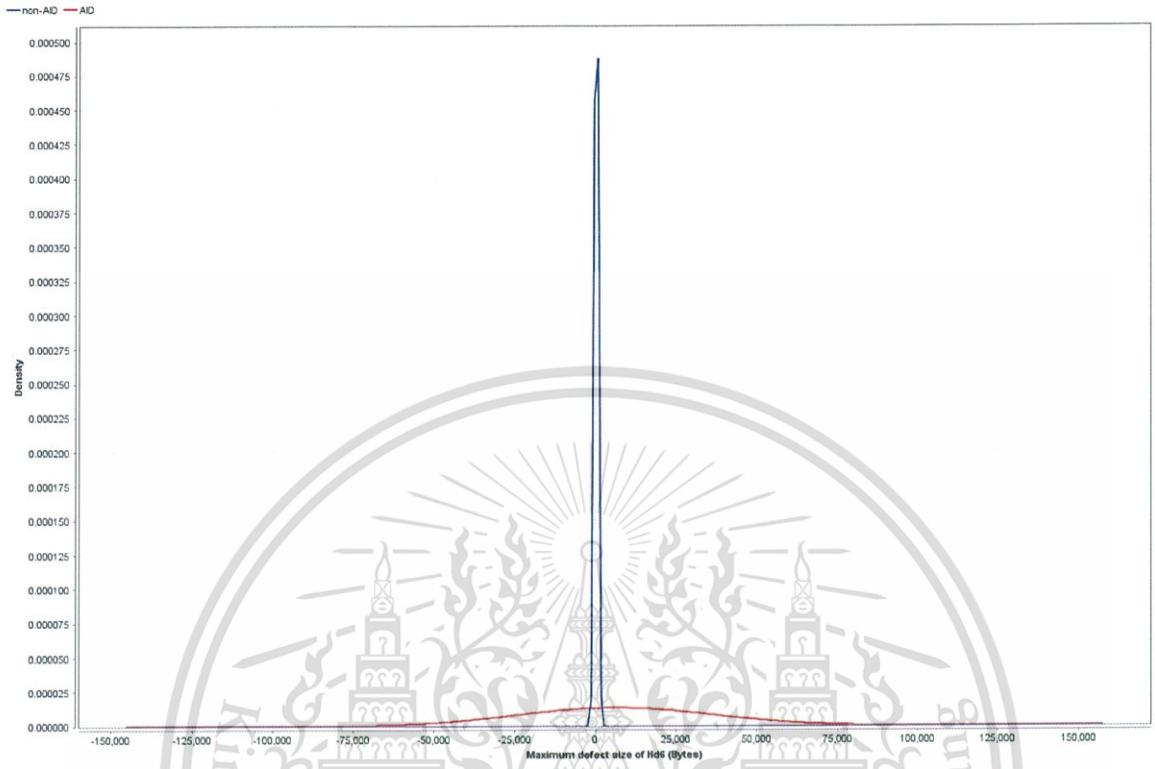


Figure 5.12 The probability density of each output (AID and non-AID) of 6.

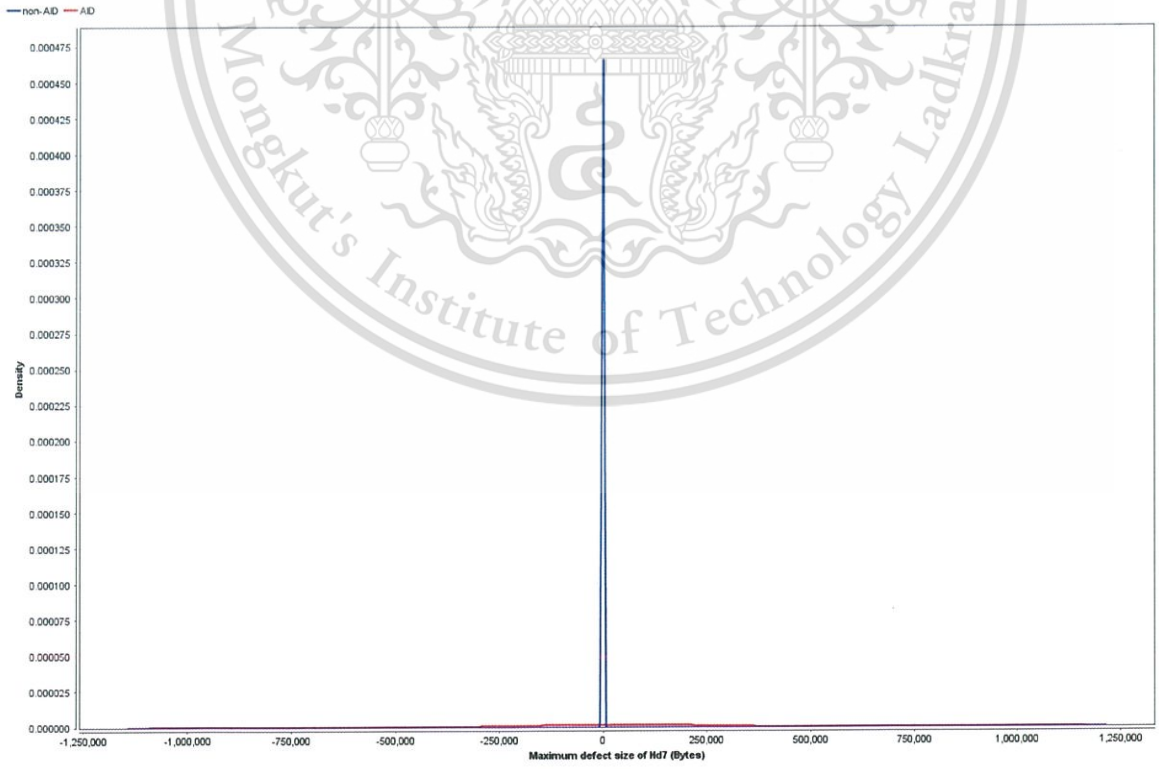


Figure 5.13 The probability density of each output (AID and non-AID) of 7.

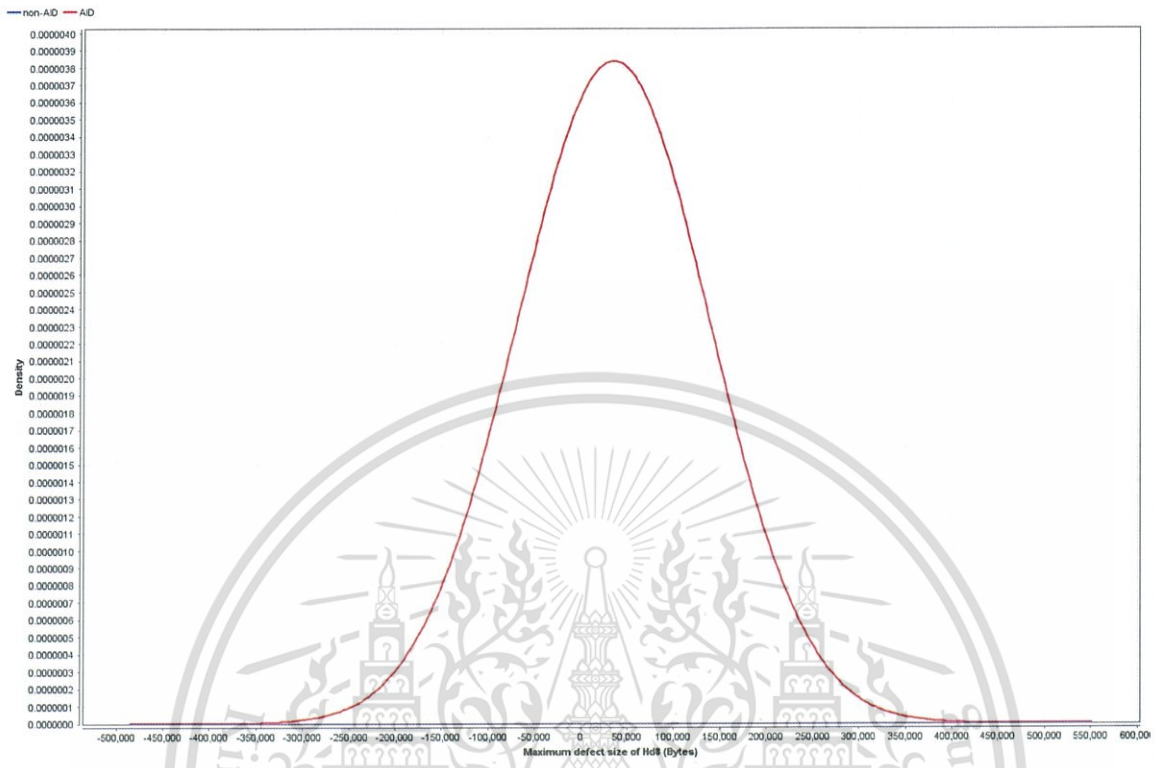


Figure 5.14 The probability density of each output (AID and non-AID) of 8.

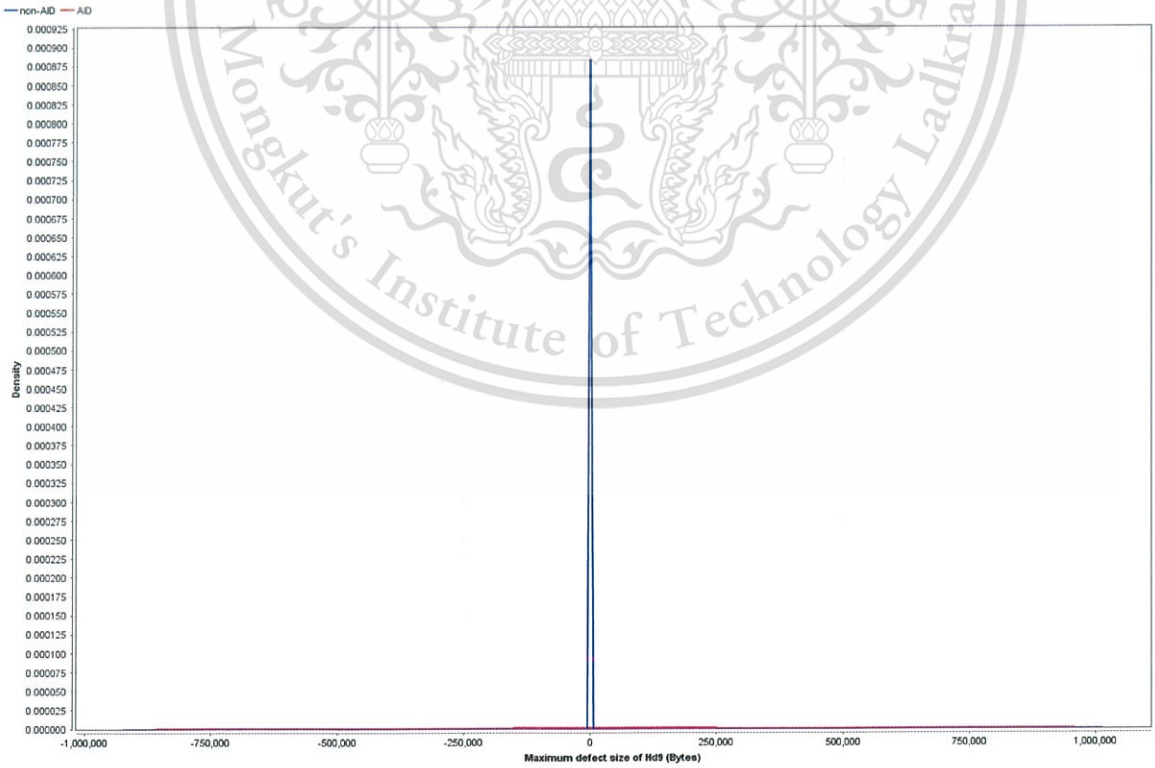


Figure 5.15 The probability density of each output (AID and non-AID) of 9.

5.2.2 The experimental accuracies from Naïve Bayes algorithm.

The experimental accuracies from Naïve Bayes algorithm of all criteria can be analyzed as in Figure 5.16. According to chart in Figure 5.16, the defect count input provides significantly poor accuracy while defect size input provides high accuracy about 96-97%.

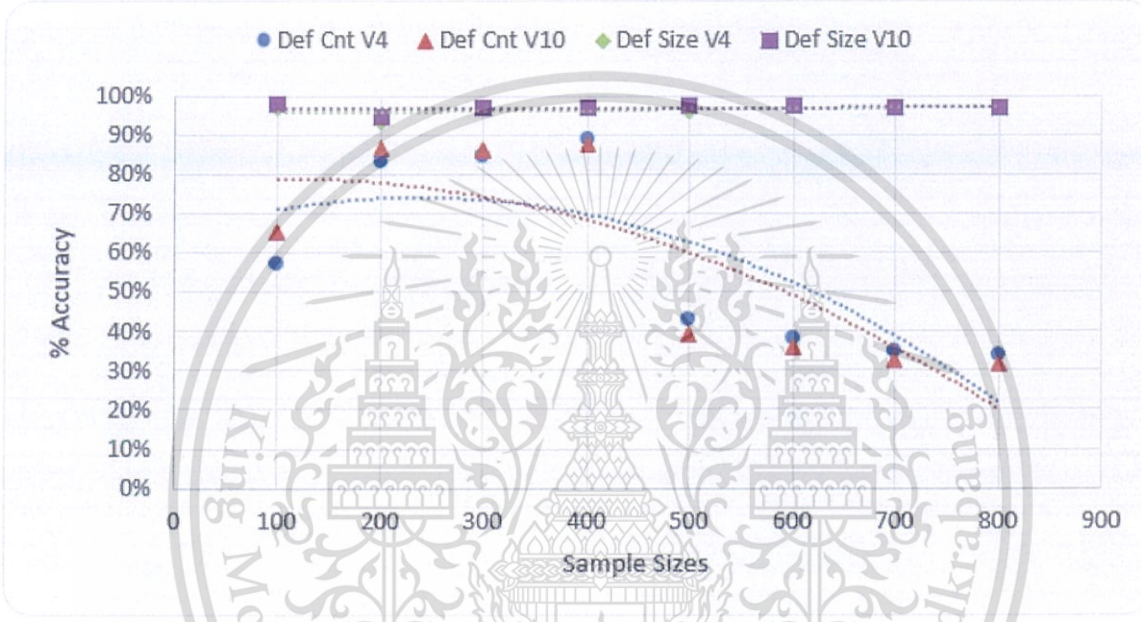


Figure 5.16 The accuracy trend from Naïve Bayes algorithm.

5.3 The results from decision tree algorithm.

5.3.1 The parameter setting from decision tree algorithm.

The parameters setting of Decision Tree algorithm consist of maximal depth: the maximal tree level to restrict the size of decision tree, confidence: confidential level, minimal gain: used for spit the node when its gain over this value, minimal leaf size: the size of leaf node, minimal size for split: node are split if size greater than this parameter and number of pre-pruning alternatives: the number of alternative nodes tried for splitting. The value settings of each one are presented in Table 5.4.

Table 5.4 The parameters setting of Decision Tree algorithm.

Parameters	Value
Maximal depth	20
Confidence	0.25
Minimal gain	0.1
Minimal leaf size	2
Minimal size for split	4
Number of prepruning alternatives	3

5.3.2 The prediction model from Decision Tree algorithm.

The selected prediction model from Decision Tree algorithm based on the highest accuracy of all criteria which are maximum defect size input, 800 sample sizes and number of validation 4 is presented in this section. The workflow to be Decision Tree prediction model can be described in diagram of Figure 5.17. According to Figure 5.17, each node of decision tree model can be selected based on "Information Gain" value which can be calculated from equations 3.12-3.13. It will continue measure until the single result receive or the tree reaches the maximum depth which is 20. The result of prediction model is presented in Figure 5.18.

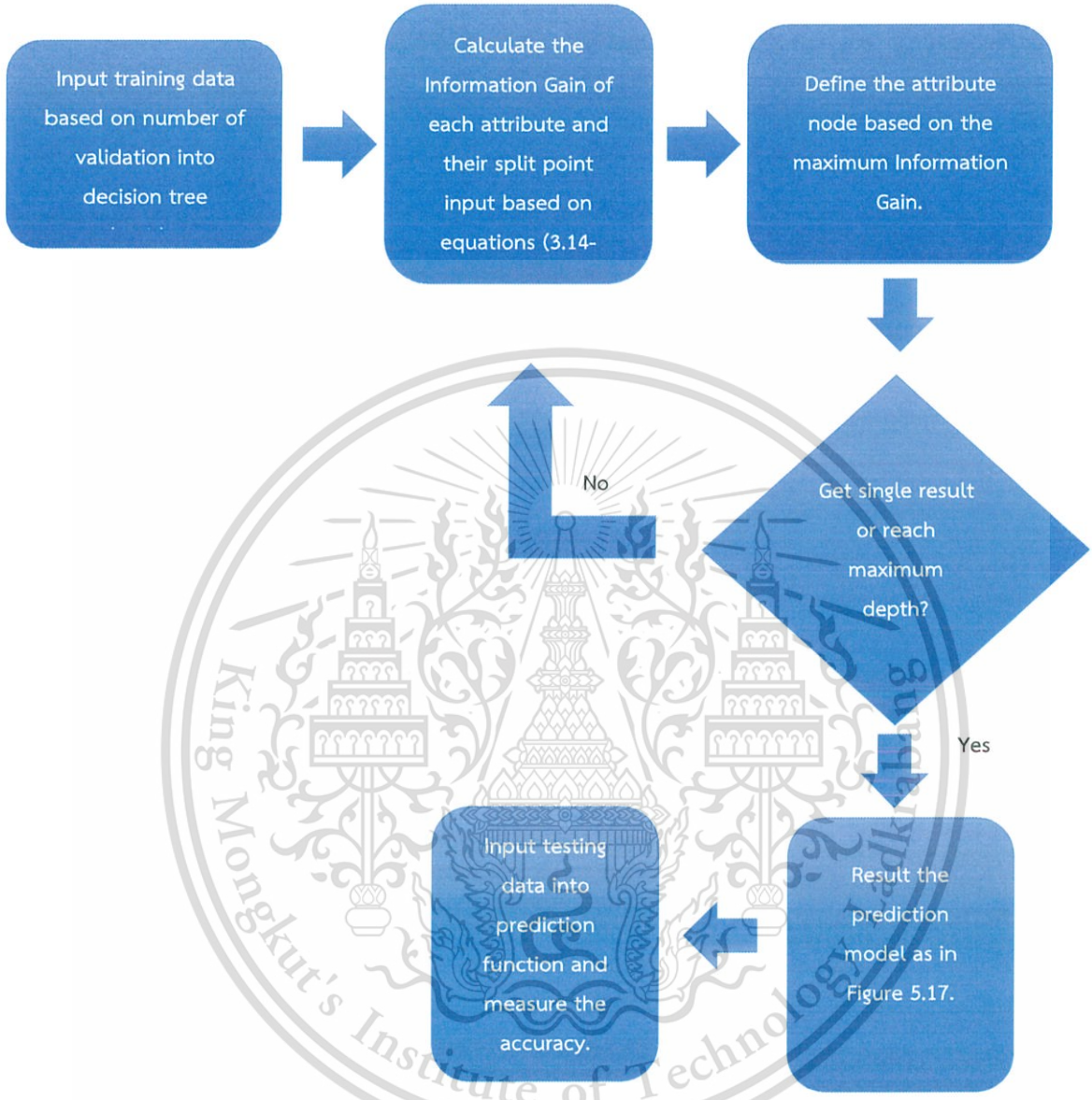


Figure 5.17 The diagram describes Decision Tree workflow.

According to Figure 5.18, Head 9 provides the maximum information gain, as a result it was selected to be root node. All of decision nodes can classify AID class but leaf nodes can classify both AID and non-AID classes. At root node, it can predict the result by classify the maximum defect size of head 9. If the value greater than 11238, the prediction result will be AID. Otherwise, it will consider on maximum defect size of head 1. If the value greater than 4242.5, the prediction result will be AID. Otherwise, it will consider next node until reach to leaf node.

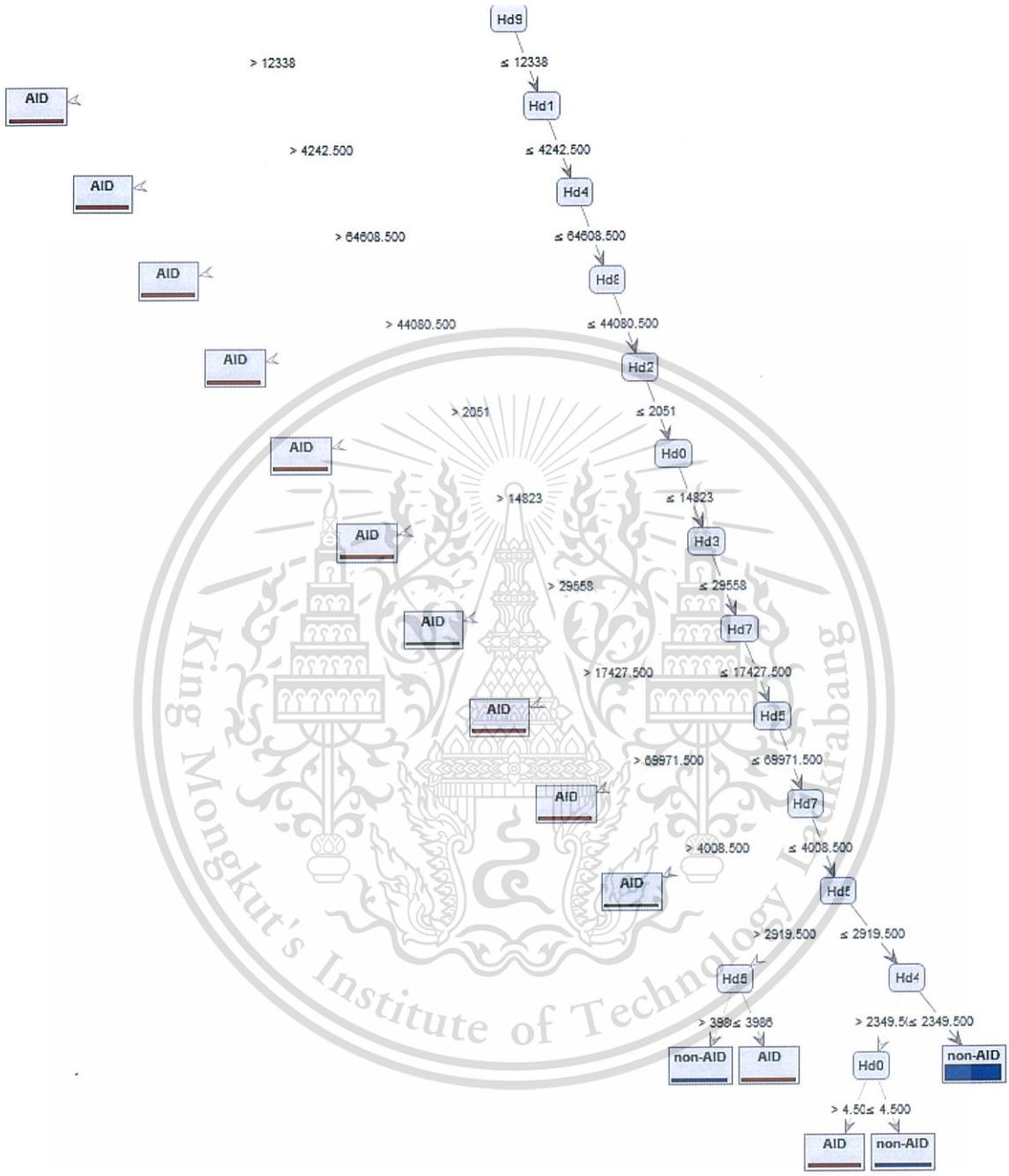


Figure 5.18 The prediction model of decision tree algorithm.

5.3.3 The experimental accuracies from Decision Tree algorithm.

The experimental accuracies from Decision Tree algorithm of all criteria can be analyzed as in Figure 5.19. According to chart in Figure 5.19, the results similar to Neural Network algorithm but provide higher accuracy. All of input criteria provided the similar trend and the accuracy starting to stable at data size 600 onward with value about 97%. The defect size input provides significantly higher accuracy than defect count input while the different of number of validation provides similar results.

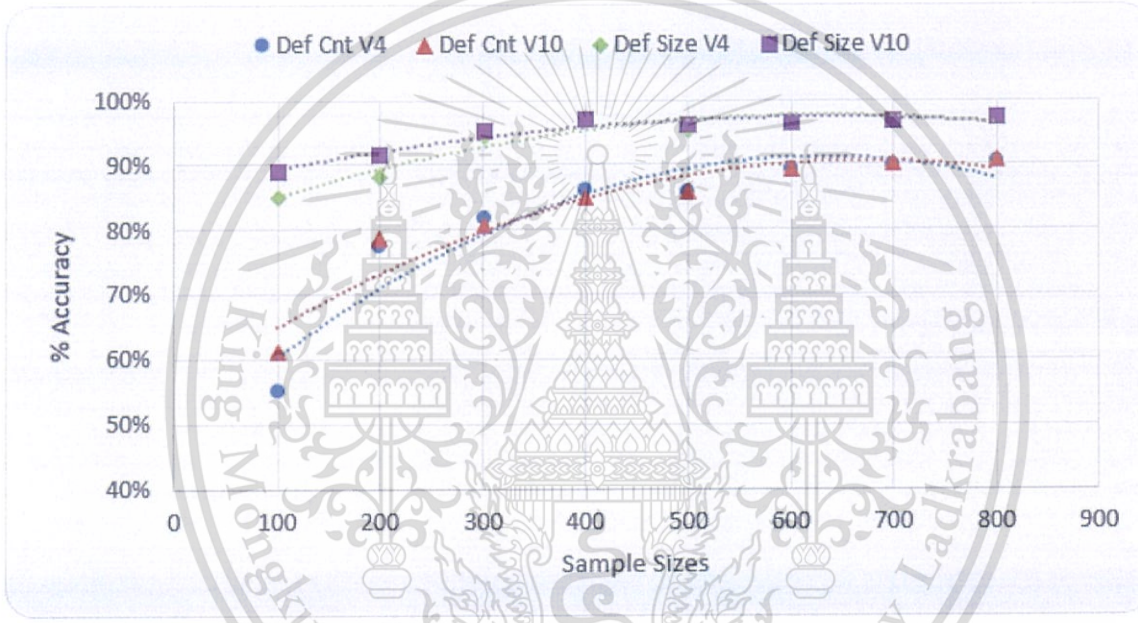


Figure 5.19 The accuracy trend from Decision Tree algorithm.

5.4 The results from Random Forest algorithm.

5.4.1 The parameter setting from Random Forest algorithm.

The parameters setting of Random Forest algorithm similar to Decision Tree algorithm but have additional number of trees parameter. These consist of number of trees: specify number of random tree generate, maximal depth: the maximal tree level to restrict the size of decision tree, confidence: confidential level, minimal gain: used for spit the node when its gain over this value, minimal leaf size: the size of leaf node and minimal size for split: node are split if size greater than this parameter. The value settings of each one are presented in Table 5.5.

Table 5.5 The parameters setting of Random Forest algorithm.

Parameters	Value
Number of trees	10
Maximal depth	20
Confidence	0.25
Minimal gain	0.1
Minimal leaf size	0.1
Minimal size for split	4

5.4.2 The prediction model from Random Forest algorithm.

The selected prediction model from Decision Tree algorithm based on the highest accuracy of all criteria which are maximum defect size input, 800 sample sizes and number of validation 4 is presented in this section. The workflow to be Decision Tree prediction model can be described in diagram of Figure 5.20. According to Figure 5.20, the multiple tree was built based on random selected subset from training data set. The prediction trees are presented in Figure 5.21-5.30. After that, apply the testing data set into all 10 prediction trees the final prediction result come from the maximum vote from those 10 trees.

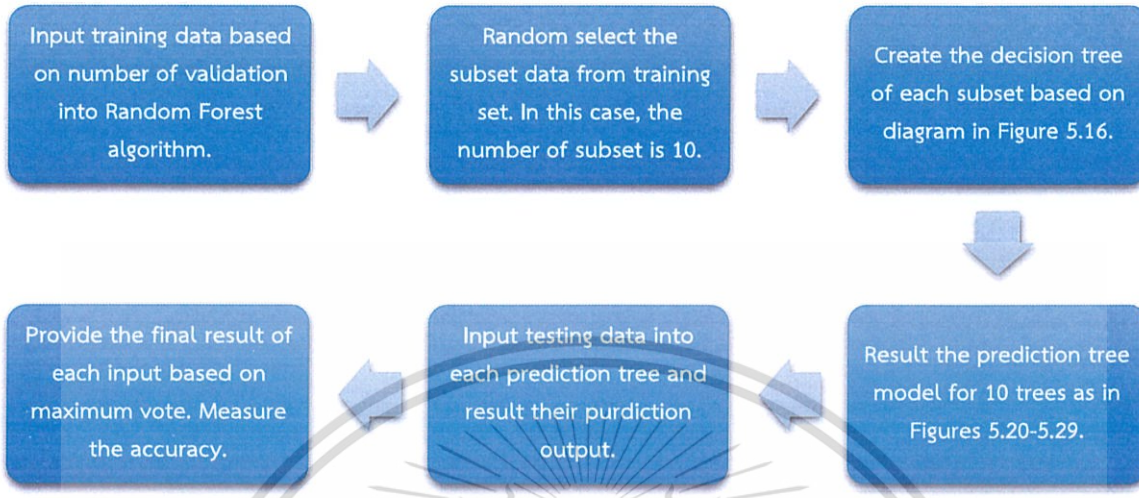


Figure 5.20 The diagram describes Random Forest workflow.

To illustrate this prediction model from Figures 5.21-5.30, we can measure the output: AID or non-AID based on maximum defect size input from each tree. The maximum vote output will be the prediction result from this model. For example tree Models 1-8 provide the non-AID result while tree Models 9-10 provide AID result. The final prediction output will be non-AID as it has the higher vote from the model.

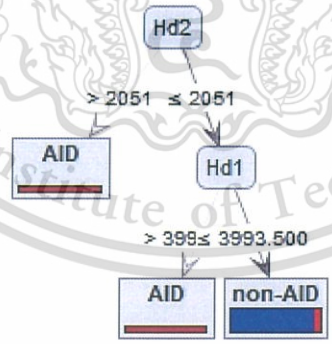


Figure 5.21 The tree model 1 from Random Forest algorithm.

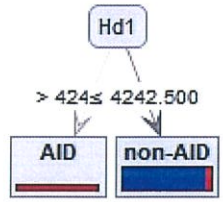


Figure 5.22 The tree model 2 from Random Forest algorithm.

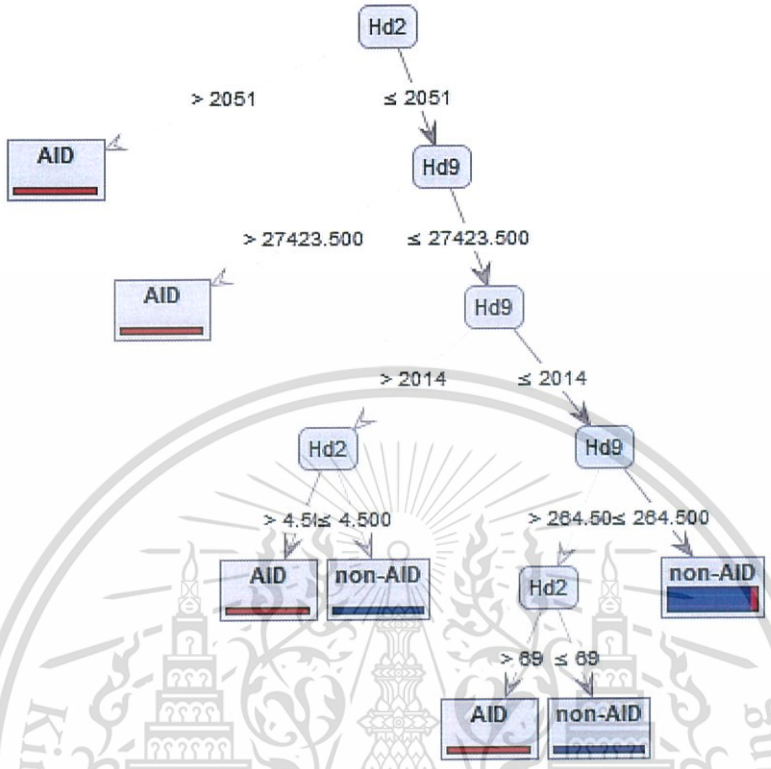


Figure 5.23 The tree model 3 from Random Forest algorithm.

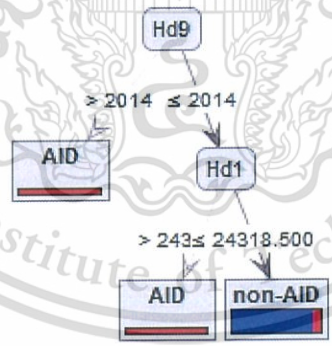


Figure 5.24 The tree model 4 from Random Forest algorithm.

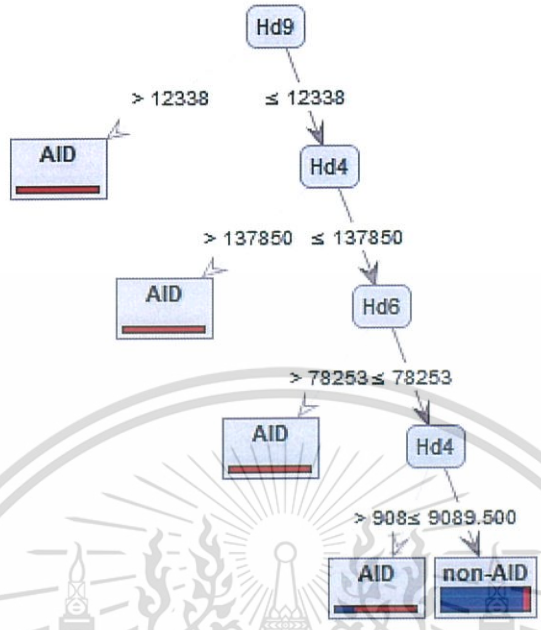


Figure 5.25 The tree model 5 from Random Forest algorithm.

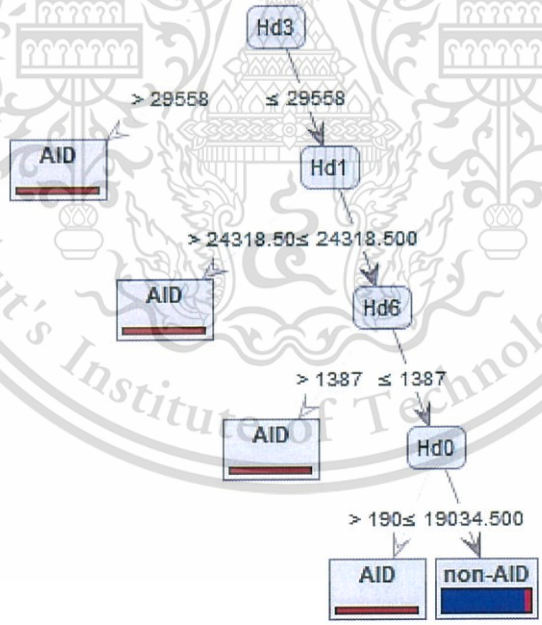


Figure 5.26 The tree model 6 from Random Forest algorithm.

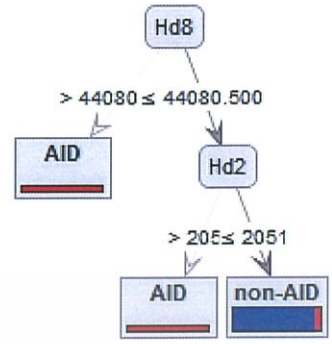


Figure 5.27 The tree model 7 from Random Forest algorithm.

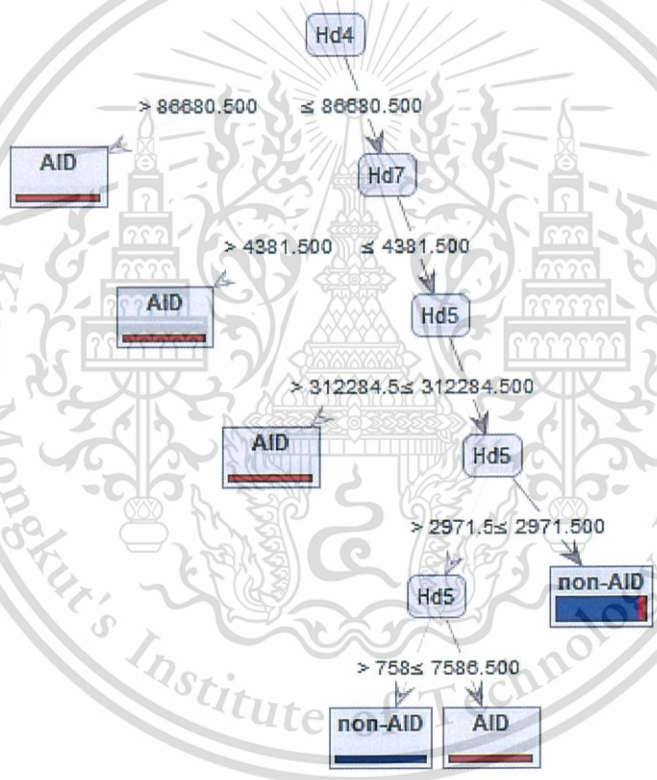


Figure 5.28 The tree model 8 from Random Forest algorithm.

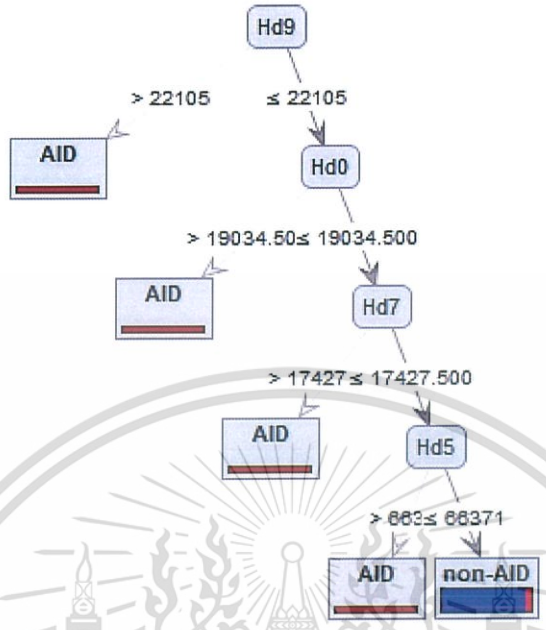


Figure 5.29 The tree model 9 from Random Forest algorithm.

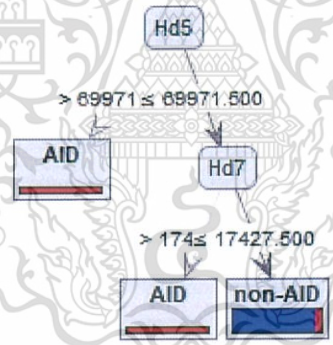


Figure 5.30 The tree model 10 from Random Forest algorithm.

5.4.3 The experimental accuracies from Random Forest algorithm.

The experimental accuracies from Random Forest algorithm of all criteria can be analyzed as in Figure 5.31. According to chart in Figure 5.31, the results of all criteria provide the similar trend and the accuracy starting to stable at data size 700 onward with value about 90-91%. The defect size input provides slightly better accuracy than the defect count input while the different of number of validation provides similar results.

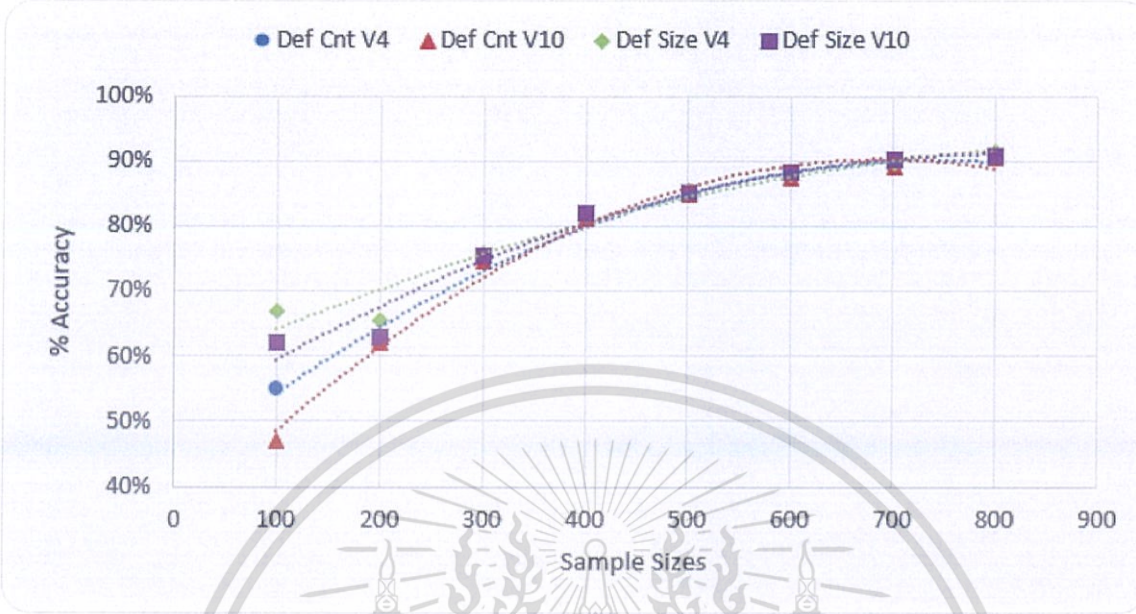


Figure 5.31 The accuracy trend from Random Forest algorithm.

5.5 The performance comparison of studied algorithm.

The experimental accuracies of all algorithm can be re-analyzed into Figures 5.32-5.33 and Table 5.6 reports the accuracies of the 800 sample data size.

According to Figures 5.32-5.33, the experimental accuracies of defect count by head zone with different validation numbers, it illustrates that the varying number of validations does not correlate with the accuracy, whereas the size of data sets does. Besides, the accuracies begin stabilize at the sample size of 600 with the values about 90%. However, the Naïve Bayes algorithm generates poor accuracy.

According to Figures 5.34-5.35, the experimental accuracies of maximum defect size per surface, it also confirms the validation number does not relate to accuracy. Also, the accuracy starts stabilize at the sample size of about 600 with the value about 96%.

According to Table 5.6, the green bold hi-lights are the accuracy performances greater than 95%, as analyzed by Naïve Bayes and Decision Tree algorithms with number of data sets at 800 and the maximum defect size per surface input. The classification algorithm model that generated the highest accuracy of 97.38% comes from the Decision Tree algorithm which has 4 validations, 800 data sets, and maximum defect size per surface as data input.

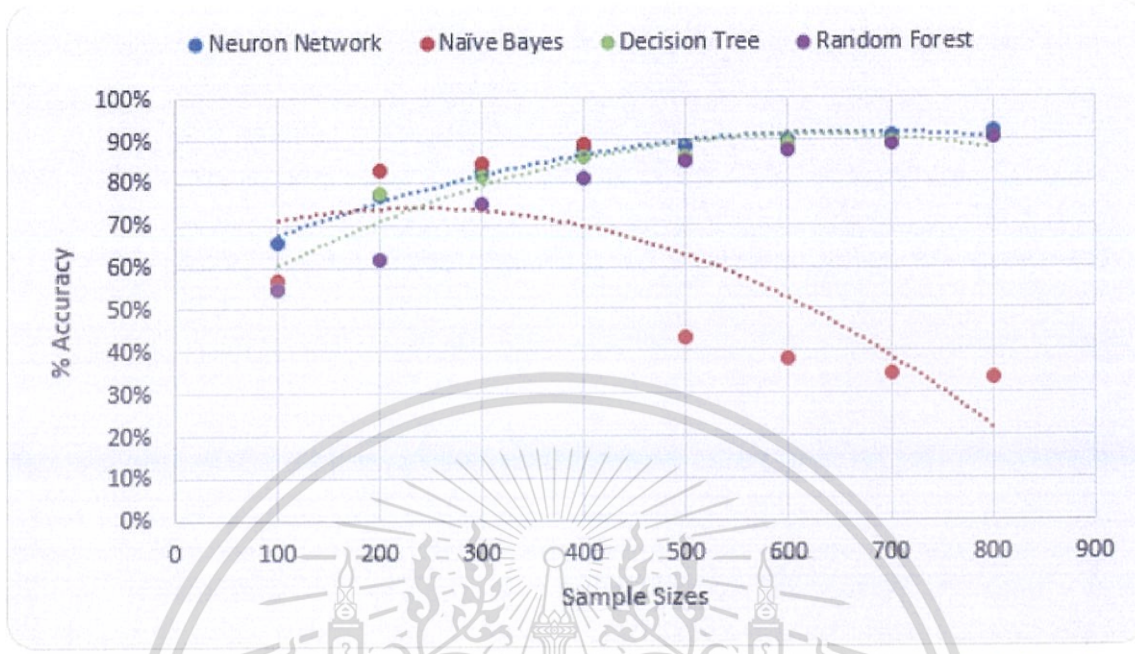


Figure 5.32 The accuracy trend of defect count by head zone with validation 4.

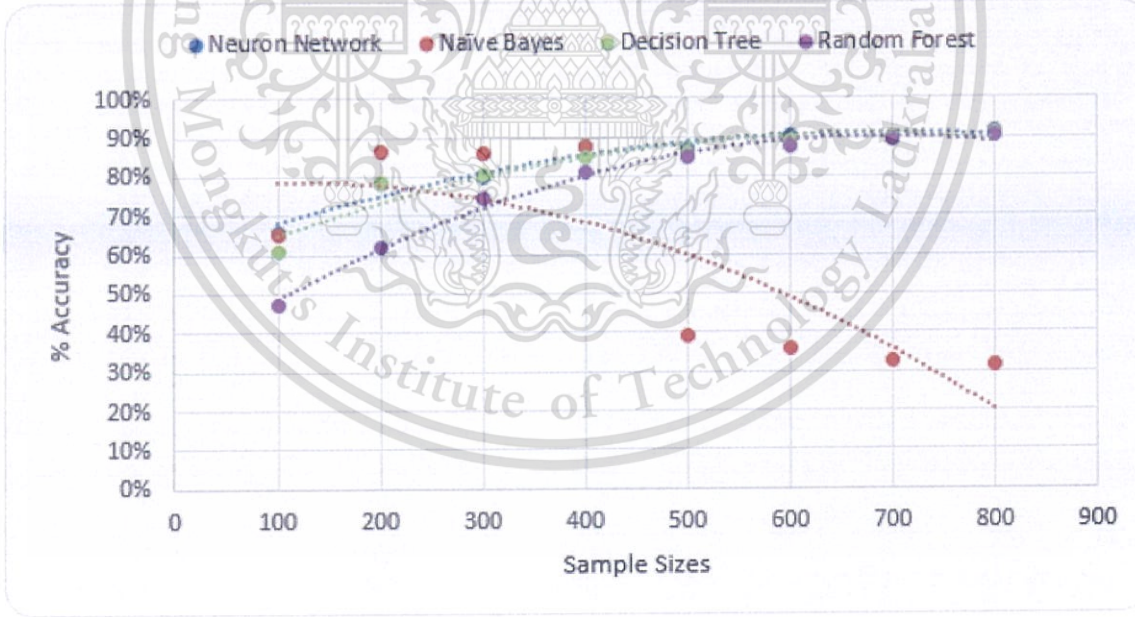


Figure 5.33 The accuracy trend of defect count by head zone with validation 10.

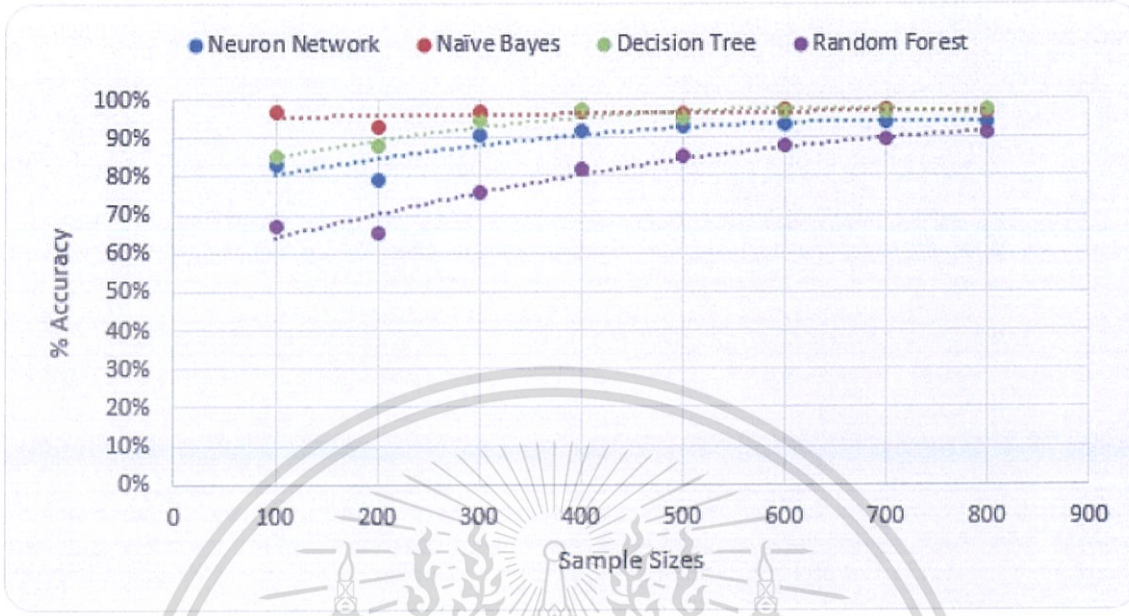


Figure 5.34 The accuracy trend of defect size per surface with validation 4.

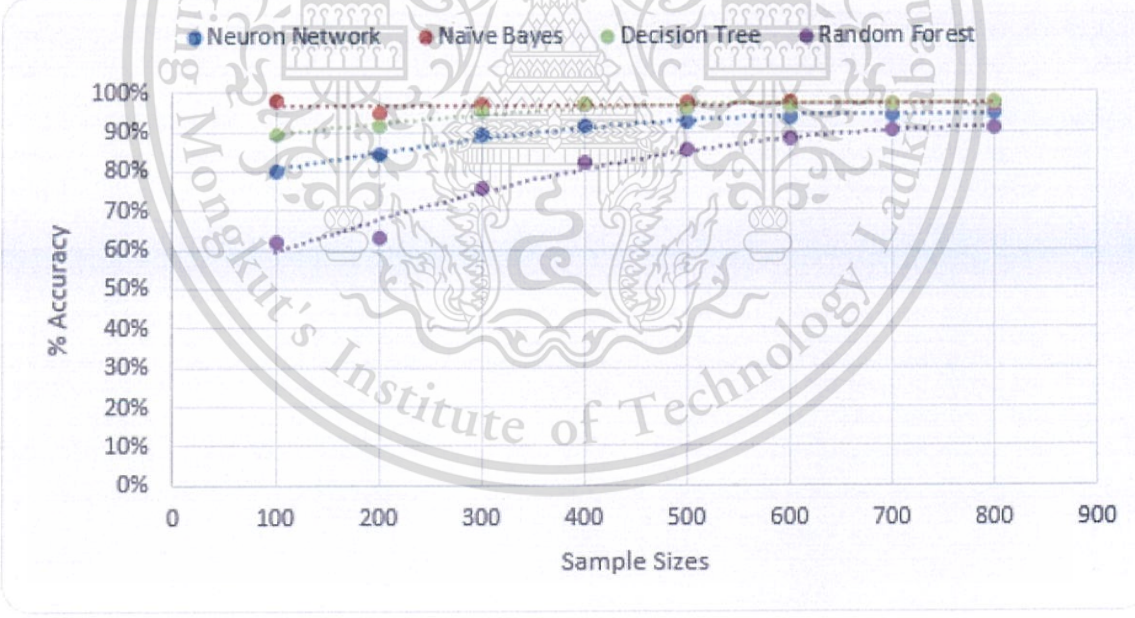


Figure 5.35 The accuracy trend of defect size per surface with validation 10.

Table 5.6 The experimental accuracies at maximum sample size 800.

Model	Input	The accuracy of Number of	
		No. Validation 4	No. Validation 10
Neuron Network	Defect Count	91.88%	91.62%
	Defect Size	94.25%	94.88%
Naïve Bayes	Defect Count	33.75%	31.38%
	Defect Size	**96.75%**	**96.75%**
Decision Tree	Defect Count	90.38%	90.75%
	Defect Size	**97.38%**	**97.37%**
Random Forest	Defect Count	90.50%	90.50%
	Defect Size	90.75%	90.75%

CHAPTER 6
CONCLUSIONS and FUTURE WORKS

6.1 Conclusions

This study aims to identify the Assembly Induced Defect (AID) based on the best prediction model and the sample size that start to get the stabilize accuracy. The selected model for this study are Neural Network, Naïve Bayes, Decision Tree and Random Forest algorithm. The input are the suspect defect parameters influence on AID. These are defect count by head zone and maximum defect size by surface. The output are the prediction model based on each classification algorithm and their accuracy. The best model can be determined by the highest accuracy.

For the experimental preparation, the hard disk drive tester are reserved for the experiment with same environment setting as in actual factory runs. Also, the group of hard disk drive was prepared to run the experiment. The HDD sample size was selected based on population 100000, confidential level 95% and confidential interval 4 which provide the output about 600 pcs. Therefore, it started from 100, 200... 800. As the input parameter contain in the text file of each drive and the time to test Defect Scan is too long, the data pre-processing is required by introducing an additional defect scan test and translating raw data from multiple text files into one desired excel file. According to data reformatting, it can be easily fed into the select classification model for efficient analysis. The ratio between AID and non-AID of each group is critical. If the data of one class have too small, it is high opportunity to get the bias accuracy. Because, it is not enough information for training and testing.

For the experimental validation, the data of each group are fed into each classification algorithm for training and testing before providing the accuracy. The Rapid Miner application was used to measure the prediction model and their accuracy. During the application measure the prediction model, the running time to come out the prediction model and accuracy from Neural Network algorithm with defect count by head zone input is too long while others model spent a few time. The reason are the Neural

Network Algorithm has a complicate learning method before come out the model and there are many input attributes which is 170 attributes. Therefore, the training time will be considered for the best prediction model in case of there are more than 2 algorithms providing the highest accuracy.

Based on the prediction accuracy results, the defect size input is right information for AID prediction. The accuracy starts to stable at sample size 600 as minimum while validation number between 4 and 10 generates similar result. Furthermore, the Decision Tree model generating the highest accuracy is the best model to predict AID and it could be applied in the normal practice of HDD test process for AID rejection. This is very helpful in cost reduction for the HDD industry.

6.2 Future works

This prediction model can be applied only the initial hard disk drive product which has 10 headers and 17 zones. It could not be applied into the future product because of others HDD model because of TPI, BPI, number of head, and number of zone difference. It would be better if:

1. The prediction model should automatically utilize to all future product. It would have the number of head, number of zone, capacity per surface as input parameters. The defect input parameter could be maximum defect size per surface based on the accuracy of this study.
2. The accuracy could be greater than 99% to achieve the highest factory benefit.

REFERENCES

[1] Cleanroom classification ISO 14644-1 cleanroom standards [Internet]; 2015 [Cited 2016 Jan]. Available from: https://en.wikipedia.org/wiki/Cleanroom.

[2] S.C.Lee, W.Tyndall, and A.Khurshudov, "Hard disk drive reliability modelling and failure prediction", Asia-Pacific Magnetic Recording Conference, 2006, pp. 1-2.

[3] M. Farhoodi, and A. Yari K, "Applying machine learning algorithms for automatic Persian text classification", Advanced Information Management and Service (IMS) conference, 2010, pp. 318-323.

[4] T. Ramangkul, and J. Ponsawad, "Hard disk drive failure mode prediction from SMART attribute using data mining method", DST-CON, 2011, pp338-340.

[5] A. Chug, and S. Dhall "Software defect prediction using supervised learning algorithm and unsupervised learning algorithm", IET Conference Publications, 2013, pp. 173-179.

[6] M. Shepperd, D. Bowes, and T. Hall, "Researcher Bias: The use of Machine Learning in Software Defect Prediction" IEEE Transactions on Software Engineering, 2014, pp603-616.

[7] H. Adeli, and S. Huang, "Machine Learning Neural Networks, Genetic Algorithms, and Fuzzy Systems", John Wiley & Sons Ltd., 1995.

[8] S. Russell, and P. Norvig. "Artificial Intelligence: A Modern Approach 3rd Edition", Pearson Education, Inc., 2010.

[9] J.L. Devore. "Probability and Statistics for Engineering and the sciences 8th Edition," Brooks/Cole, Cengage Learning., 2009.

[10] R. O. Duda, Peter E. Hart and David G. Stork. "Pattern Classification 2nd Edition" John Wiley & Sons Ltd., 2001.

[11] J.Han and M.Kamber. "Data Mining Concepts and Techniques", Elsevier Inc., 2006.

[12] D.T. Larose. "Discovering Knowledge in Data: An Introduction to Data Mining", John Wiley & Sons Ltd., 2005.

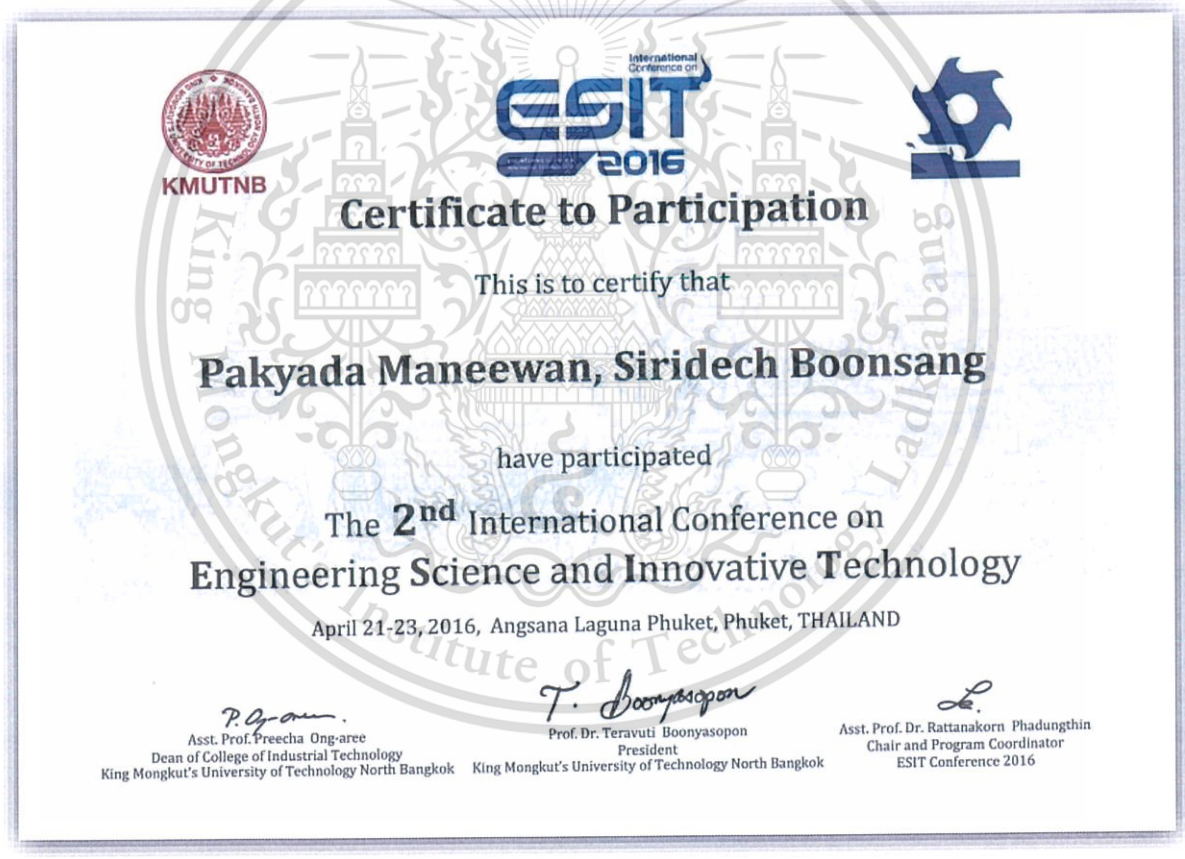
[13] T. Hastie, R. Tibshirani and J. Friedma. "The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer Science & Business Media, 2009.

[14] H.G. Schaathun, "Machine Learning in Image Steganalysis", John Wiley & Sons Ltd., 2012.

APPENDIX A

PUBLICATION

This research has been published and presented in the 2nd International Conference on Engineering Science and Innovative Technology (ESIT 2016) at Angsana Laguna Phuket, Phuket, Thailand during April21-23, 2016.



KMUTNB

International Conference on ESIT 2016

Certificate to Participation

This is to certify that

Pakyada Maneewan, Siridech Boonsang

have participated

The 2nd International Conference on

Engineering Science and Innovative Technology

April 21-23, 2016, Angsana Laguna Phuket, Phuket, THAILAND

Asst. Prof. Preecha Ong-aree
Dean of College of Industrial Technology
King Mongkut's University of Technology North Bangkok

Prof. Dr. Teravuti Boonyasopon
President
King Mongkut's University of Technology North Bangkok

Asst. Prof. Dr. Rattanakorn Phadungthin
Chair and Program Coordinator
ESIT Conference 2016

Paper ID101

The 2nd International Conference on Engineering Science and Innovative Technology (ESIT 2016), Phuket, Thailand, April 21 – 23 2016

Assembly Induced Failure Classification and Prediction

Pakyada Maneewan[1,2,*] and Siridech Boonsang[2,*]

Abstract

Assembly Induced Defect (AID) is a kind of the defect induced during the hard disk drive assembly process in clean room and it is captured during the in testing process. The objective of this study is to identify the best existing classification model which can produce the highest accuracy to capture this failure. The classification models in this study are Neuron Network, Naïve Bayes, Random Forest and Decision Tree. As the time needed to detect all flaw types including AID, in a regular defect scan sequence is considered too long and costly to reject the drive, it would be beneficial for the hard disk drive factory to be able to capture the AID at the early stages of the test process. Because the engineer responding at Hard Disk Drive (HDD) assembly process can correct the assembly fault quicker. Therefore, an additional defect scan sequence, for the data pre-processing, is added at the earliest possible state in test process to identify the required information. These are defect count by head-zone and maximum defect size per surface which are the input data elements of the selected model. All selected models are measured based on different input data, maximum defect size per surface or defect count by head-zone, and data sizes starting from 100, 200 until 800. Next, the percentage of accuracy with each criterion is analyzed and compared. The results indicate that the detection accuracy would be stable at sample size 600 to 800. Hence, the Decision Tree model is selected as it can provide the highest accuracy based on maximum defect size as input information.

Keywords: Classification model, Defect prediction, Decision Tree, Naïve Bayes, Neuron Network, Supervised Learning

[1] Seagate Technology (Thailand) ltd.

[2] Department of Data Storage Technology, College of International, King Mongkut's Institute of Technology Ladkrabang

[*] pakyada.maneewan@seagate.com, and siridech@gmail.com

1. Introduction

In Hard Disk Drive (HDD) industry, "Defect Scan" is one of the test process sequences in HDD testing process which can identify the defect size, amplitude and location in media. The defect would cause from either incoming media or induced during assembly in factory. One kind of such defect is called "Assembly Induced Defect" and it is created during hard disk drive assembly in clean room process. If the assembly machine has some misalignment or error, machine part can come in contact with the disc media and cause damage, as shown in Fig.1. Furthermore, possibility of this defect to grow after being mapped during media optimization test is very high. Thus, the right disposition should be to detect and reject this defect as soon as possible. We should not mark the defect, defect mapping in HDD industry term, and let it pass to the following test sequence to avoid the quality issues later on.

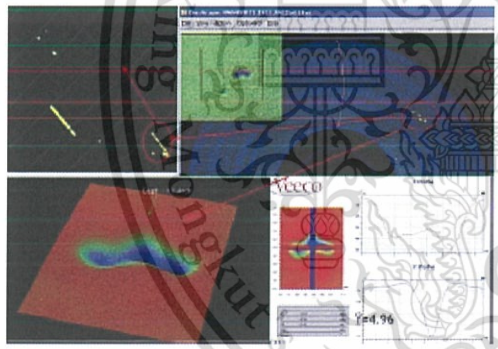


Fig. 1. Assembly Induced Defect Sample

The previous studies which are associated with the classification and prediction have presented accurate results. Most of the study's accuracy were promising, however some studies were not. Tyndall and Khurshoudov [1] introduced their model to predict the reliability failure from air pressure, temperature and humidity variation. Farhoodi and Yari [2] used both SVM (Support Vector Machine) and KNN (K-Nearest Neighbor) model to classify Persian Text. The outcomes of both algorithms were highly accurate. However, KNN generated better performance. For software defect prediction, Chug and Dhall [3] declared that the Random Forest Algorithm is the best supervised learning and K-means clustering algorithm is the best unsupervised algorithm to achieve the highest accuracy on data set of interest. On the other hand, Shepperd Bowes and Hall [4] applied ANOVA model to predict the software defect. After meta-analysis, they could not identify the proper model to predict software defects.

However, the Assembly Induced defect is considered specific defect type, having unique shape, and there is no study regarding this defect classification and prediction. The purpose of this research was to identify the best existing classification model for the AID prediction. It could be applied in the normal practice of HDD test process. Hence, the defect can be efficiently classified beforehand during the test.

2. Experimental Setup

The equipment used in experiment is HDD tester which can measure drive values based on the sequence identified in test script. The hardware setting of tester would be same as normal setting in HDD test process because we would like to validate in same environment as the real process. The test sequence in test script is modified by adding the special defect scan sequence at the earliest of possible states of the HDD test as shown in Fig. 2. In this case, it has been added at the time about 19 hours. There are 2 reasons of this additional sequence. One is the information required for analysis, defect count by head zone and maximum defect size per surface, retrieved in normal defect scan test which is performed at the time of 80 hours. However, the time is considered too long to get failure information and detect the problem. And the best model will be applied at the end of this sequence test for AID or non-AID identification.

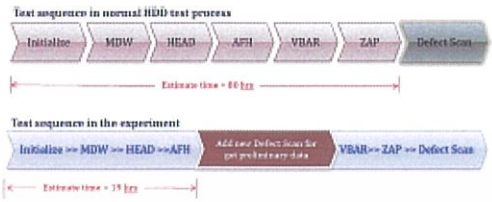


Fig. 2. Additional defect scan for quick data obtained.

The drives provided as the material in this experiment are composed of 8 groups. Each group has 100, 200... 800 pcs consisting of AID and non-AID drives. All drives were validated in standard test machine and the new test sequence. After all drive complete validate, the results of all test sequence including the additional test sequence were accumulated into one text document for each drive called the "result file". Therefore, the maximum result file would be 800 files from the group with maximum drive quantities testing.

Since the information focused on is defect count by head zone and maximum defect size per surface and be part of the result file, it is required that a special computer program specifically be written for this study to filter and translate from text result file to Microsoft excel format. For defect count, the raw data in result file provide the defect count by head and zone in DEF_CNT column as shown in Fig. 3. This special program can directly parse the information and translate it into excel file. For the maximum defect count information, as the raw data in result file provide only the size of each defect in DEF_SIZE column as shown in Fig. 4, there is no information of maximum defect size per surface. Therefore, the program must calculate the maximum defect size before translate to excel format. Therefore, the translated data are piled up into 2 excel files from all result files. One is defect count by head-zone of all result files and another one is maximum defect size per surface of all result files. Subsequently, more data must be labeled to the translated data: AID or non-AID.

Defect Count infor

PHY_HD_NO	ZN_NO	LOG_HD_NO	DEF_CNT	STATUS
0	0	0	10	1
0	1	0	0	1
0	2	0	0	1
0	3	0	2	1
0	4	0	1	1
0	5	0	25	1
0	6	0	0	1
0	7	0	26	1
0	8	0	5	1
0	9	0	0	1
0	10	0	0	1
0	11	0	8	1
0	12	0	1	1
0	13	0	0	1
0	14	0	0	1

Fig. 3. The raw data of defect count by head-zone.

Defect Size infor

PHY_HD_NO	START_TRK	HD_LOG_PSN	ENDING_TRK	DEF_SIZE	STATUS
9	50788	9	50796	14	P
9	51020	9	51026	11	P
8	55262	8	55272	21	P
5	60029	5	60034	26	P
6	64276	6	64282	21	P
7	64322	7	64326	9	P
0	88702	0	88706	11	P
3	93350	3	93356	12	P
2	94056	2	94064	21	P
1	96669	1	96676	38	P
0	103716	0	103720	7	P
1	108782	1	108788	11	P

Fig. 4. The raw data of size of defect.

3. Classification Algorithms

The classification algorithms selected for AID defect prediction are Neuron Network, Naïve Bayes, Decision Tree and Random Forest algorithms. The inputs are data sets which contain both maximum defect size and defect count per surface of the results "AID" and "non-AID" results.

3.1 Neuron Network

The prediction model can be presented in form of activation function which consists of nodes associated with directed link with unique weight [5][6] as show in Fig. 5.

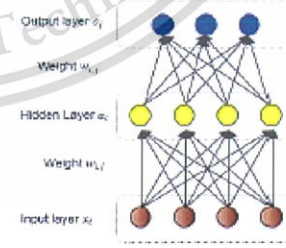


Fig. 5. A multi-layer perceptron [6].

The activation function appears as equation 1[6].

$$a_j = g(\sum_{i=0}^{n} w_{i,j} a_i) \quad (1)$$

Where $w_{i,j}$ is weight of each link and a_i is hidden result from vector input x_i.

3.2 Naïve Bayes

The prediction model can be presented in form of probability function based on Bayes's Theorem as in equation 2 [7]. The defect prediction is one of the most popular classified by this model

$$P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_i)}{p(x)} \quad (2)$$

Where $P(\omega_j|x)$ is probability mass function of event ω_j based on x, $p(x|\omega_j)$ is probability density function of ω_j given x, $P(\omega_j)$ is probability of event ω_j and $p(x)$ is probability density function of x [8].

3.3 Decision Tree

The prediction model can be presented in form of tree structure consisting of decision nodes and branches. Each node is connected by a branch and extending downward until completing in leaf nodes as shown in Fig. 6. Decision Trees can classify only discrete target attribute and require large numbers of data set training [9].

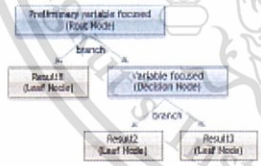


Fig. 6. Decision Tree structure.

3.4 Random Forest

The prediction model can be presented in form of the average of multiple trees structure. Each tree can be created as following sequences [10].

a) At training data set, create sample of bootstrap Z^* with N sizes.

b) Use recursive repeating generation of each random-forest tree (T_b) until the minimum node size n_{ms} is obtained.

c) Repeat a to b for number of trees setting (B). After that the set of multiple trees $\{T_b\}_1^B$ is taken place. The regression function can be presented in equation 3.

$$f_{rf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \quad (3)$$

d) The selected classification is highest vote of each class prediction from the forest as in equation 4.

$$C_{rf}^B(x) = majority\ vote\{C_b(x)\}_1^B \quad (4)$$

Where $C_b(x)$ is the class prediction the random-forest at b^{th}.

4. Data Analysis with Classification Algorithm

To run the experiment, the translated data are fed into the 4 different classification algorithms: Neuron Network, Naïve Bayes, Decision Tree and Random Forest with different 8 data sets: 100, 200, 300... 800, and different input data: maximum defect size per surface or maximum defect count per surface as shown in Fig. 7. Based on each classification algorithm, the input data are divided into 2 groups. The first group is used as learning material (called training sets) based on the selected model and then the unique prediction model formula is provided. This formula is applied to the data in the second group (called testing sets) and then the accurate results are calculated and presented [11]. The ratio between training and testing data sets calls the number of validation numbers. In this case, 2 values are selected for this study, 4 and 10. The assumption would see higher accuracy on higher data set of 800 sets and higher validation numbers. Eventually, the model generating the highest accuracy greater than 95% is selected.

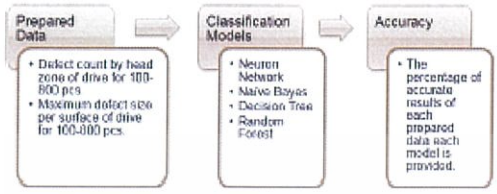


Fig. 7. The validation condition flow

Number of Validation (k) is the number of divided data sets used in both the training process and testing process. The k-1 sub data sets are applied as training data set while another sub data set is used as testing data set.

5. Results and Discussion

The accurate results of each algorithm are presented in Fig 8-11 and Table 1 reports the accurate results of the 800 sample data size.

According to Fig. 8-9, the accurate results of defect count by head zone with different validation numbers, it illustrates that the varying number of validations does not correlate with the accurate results, whereas the size of data sets does. Besides, the accuracies begin stabilize at the sample size of 600 with the values about 90%. However, the Naïve Bayes algorithm generates poor accuracy.

According to Fig. 10-11, the accurate results of maximum defect size per surface, it also confirms the validation number does not relate to accuracy. Also, the accuracy starts stabilize at the sample size of about 600 with the value about 96%.

According to **Table1**, the green bold hi-lights are the accuracy performances greater than 95%, as analyzed by Naïve Bayes and Decision Tree algorithms with number of data sets at 800 and the maximum defect size per surface input. The classification algorithm model that generated the highest accuracy of 97.38% comes from the Decision Tree algorithm which has 4 validations, 800 data sets, and maximum defect size per surface as data input.

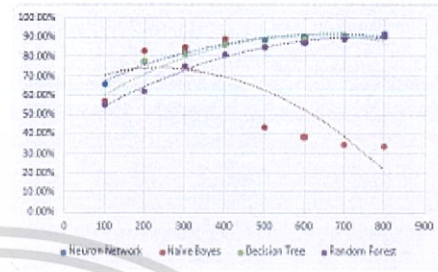


Fig. 8. The accuracy of defect count with validation 4.

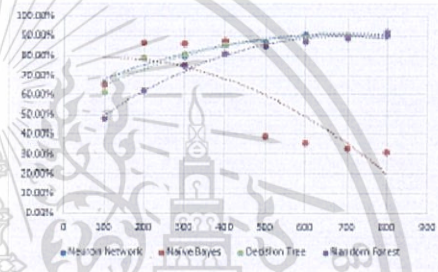


Fig. 9. The accuracy of defect count with validation 10.

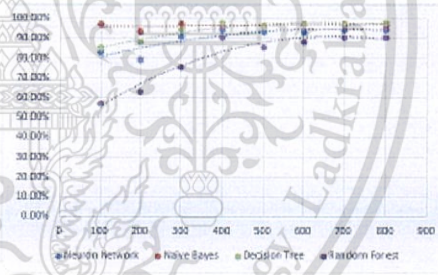


Fig. 10. The accuracy of defect size with validation 4.

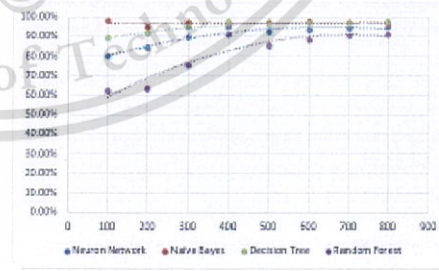


Fig. 11. The accuracy of defect size with validation 10.

Table1 The accurate results at maximum sample size 800.

Model	Input	The accuracy	
		No. Validation 4	No. Validation 10
Neuron Network	Defect Count	91.88%	91.62%
	Defect Size	94.25%	94.88%
Naïve Bayes	Defect Count	33.62%	31.00%
	Defect Size	96.75%	96.75%
Decision Tree	Defect Count	90.38%	90.75%
	Defect Size	97.38%	97.37%
Random Forest	Defect Count	90.50%	90.50%
	Defect Size	90.62%	90.75%

5. Conclusion

The Assembly Induced failure prediction has been created based on defect count by head/zone and maximum defect size per surface. The data pre-processing is required by introducing an additional defect scan test and translating raw data from multiple text files into one desired excel file. According to data reformatting, they can be easily fed into the select classification model for efficient analysis.

Based on the prediction accuracy results, it starts stabilize at sample size 600 as minimum while validation number between 4 and 10 generates similar result. Furthermore, the Decision Tree model generating the highest accuracy is the best model to predict AID and it could be applied in the normal practice of HDD test process for AID rejection. This is very helpful in cost reduction for the HDD industry.

6. Acknowledgement

This research was made successful by both financial and experimental equipment support given by Seagate Technology (Thailand) Ltd and College of Data Storage Innovation, King Mongkut's Institute of Technology Ladkrabang.

7. Reference

[1] S.C.Lee, W.Tyndall, and A.Khurshudov, "Hard Disk Drive Reliability Modelling and Failure Prediction", Asia-Pacific Magnetic Recording Conference, 2006, pp. 1-2.

[2] M. Farhoodi, and A. Yari K, "Applying Machine Learning Algorithms for Automatic Persian Text Classification", Advanced Information Management and Service (IMS) conference, 2010, pp. 318-323.

[3] A. Chug, and S. Dhall "Software Defect Prediction Using Supervised Learning Algorithm and Unsupervised Learning Algorithm", IET Conference Publications, 2013, pp. 173-179.

[4] M. Shepperd, D. Bowes, and T. Hall, "Researcher Bias: The use of Machine Learning in Software Defect Prediction" IEEE, 2014, pp603-616.

[5] Hojjat Adeli, and Shih-Lin Huang, *"Machine Learning Neural Networks, Genetic Algorithms, and Fuzzy Systems"*, John Wiley & Sons Ltd., 1995.

[6] Stuart Russell, and Peter Norvig, *"Artificial Intelligence: A Modern Approach 3rd Edition"*, Pearson Education, Inc., 2010.

[7] Rechard O. Duda, Peter E. Hart and David G. Stork. *"Pattern Classification 2nd Edition"* John Wiley & Sons Ltd., 2001.

[8] Jay L. Devore, *"Probability and Statistics for Engineering and the sciences 8th Editio,"* Brooks/Cole, Cengage Learning., 2009.

[9] Daniel T. Larose, *"Discovering Knowledge in Data: An Introduction to Data Mining"*, John Wiley & Sons Ltd., 2005.

[10] Trevor Hastie, Robert Tibshirani and Jerome Friedma, *"The Elements of Statistical Learning: Data Mining, Inference, and Prediction"*, Springer Science & Business Media, 2009.

[11] Hans Georg Schaathun, *"Machine Learning in Image Steganalysis"*, John Wiley & Sons Ltd., 2012.

APPENDIX B

THE ACCURACIES of ALL CRITERIA

Table B.1 The accuracy of defect count input with number of validation 4 from Neural Network algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	43	27	61.43%	66.00%
	Predict AID	7	23	76.67%	+/-7.21%
	Class recall	86.00%	46.00%		(mikro: 66.00%)
200	Predict non-AID	120	42	74.07%	77.00%
	Predict AID	4	34	89.47%	+/-5.92%
	Class recall	96.77%	44.74%		(mikro: 77.00%)
300	Predict non-AID	222	51	81.23%	82.33%
	Predict AID	2	25	92.59%	+/-2.73%
	Class recall	99.11%	32.89%		(mikro: 82.33%)
400	Predict non-AID	322	45	87.74%	88.25%
	Predict AID	2	31	93.94%	+/-2.86%
	Class recall	99.38	40.79		(mikro: 82.25%)
500	Predict non-AID	419	53	88.77%	88.40%
	Predict AID	5	23	82.14%	+/-1.20%
	Class recall	98.82%	30.26%		(mikro: 88.40%)
600	Predict non-AID	517	52	90.86%	90.17%
	Predict AID	7	24	77.42%	+/-1.44%
	Class recall	98.66%	31.58%		(mikro: 90.17%)
700	Predict non-AID	611	50	92.44%	91.00%
	Predict AID	13	26	66.67%	+/-1.91%
	Class recall	97.92%	34.21%		(mikro: 91.00%)
800	Predict non-AID	716	57	92.63%	91.88%
	Predict AID	8	19	70.37%	+/-0.82%
	Class recall	98.90%	25.00%		(mikro: 91.88%)

Table B.2 The accuracy of defect count input with number of validation 4 from Naïve Bayes algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	29	22	56.86%	57%
	Predict AID	21	28	57.14%	+/-5.92%
	Class recall	58.00%	56.00%		(mikro: 57.00%)
200	Predict non-AID	109	19	85.16%	83%
	Predict AID	15	57	79.17%	+/-7.00%
	Class recall	87.90%	75.00%		(mikro: 83.00%)
300	Predict non-AID	217	39	84.77%	84.67%
	Predict AID	7	37	84.09%	+/-2.75%
	Class recall	96.88%	48.68%		(mikro: 84.67%)
400	Predict non-AID	313	33	90.46%	89.00%
	Predict AID	11	43	79.63%	+/-2.35%
	Class recall	96.60%	56.58%		(mikro: 89.00%)
500	Predict non-AID	164	25	86.77%	43.00%
	Predict AID	260	51	16.40%	+/-6.73%
	Class recall	38.68%	67.11%		(mikro: 43.00%)
600	Predict non-AID	176	22	88.89%	38.33%
	Predict AID	348	54	13.43%	+/-5.61%
	Class recall	33.59%	71.05%		(mikro: 38.33%)
700	Predict non-AID	190	22	89.62%	34.86%
	Predict AID	434	54	11.07%	+/-3.64%
	Class recall	30.45%	71.05%		(mikro: 34.86%)
800	Predict non-AID	212	18	92.17%	33.75%
	Predict AID	512	58	10.18%	+/-2.36%
	Class recall	29.28%	76.32%		(mikro: 33.75%)

Table B.3 The accuracy of defect count input with number of validation 4 from Decision Tree algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	33	28	54.10%	55%
	Predict AID	17	22	56.41%	+/-11.09%
	Class recall	66.00%	44.00%		(mikro: 55.00%)
200	Predict non-AID	116	37	75.82%	78%
	Predict AID	8	39	82.98%	+/-4.56%
	Class recall	93.55%	51.32%		(mikro: 77.50%)
300	Predict non-AID	219	50	81.41%	81.67%
	Predict AID	5	26	83.87%	+/-1.73%
	Class recall	97.77%	34.21%		(mikro: 81.67%)
400	Predict non-AID	319	51	86.22%	86.00%
	Predict AID	5	25	83.33%	+/-1.41%
	Class recall	98.46%	32.89%		(mikro: 86.00%)
500	Predict non-AID	413	60	87.32%	85.80%
	Predict AID	11	16	59.26%	+/-1.04%
	Class recall	97.41%	21.05%		(mikro: 85.80%)
600	Predict non-AID	515	56	90.19%	89.17%
	Predict AID	9	20	68.97%	+/-1.28%
	Class recall	98.28%	26.32%		(mikro: 89.17%)
700	Predict non-AID	613	61	90.95%	89.71%
	Predict AID	11	15	57.69%	+/-0.57%
	Class recall	98.24%	19.74%		(mikro: 89.71%)
800	Predict non-AID	713	66	91.53%	90.38%
	Predict AID	11	10	47.62%	+/-0.89%
	Class recall	98.48%	13.16%		(mikro: 90.38%)

Table B.4 The accuracy of defect count input with number of validation 4 from Random Forest algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	26	21	55.32%	55%
	Predict AID	24	29	54.72%	+/-5.20%
	Class recall	52.00%	58.00%		(mikro: 55.00%)
200	Predict non-AID	124	76	62.00%	62%
	Predict AID	0	0	0.00%	+/-0.00%
	Class recall	100.00%	0.00%		(mikro: 62.00%)
300	Predict non-AID	224	76	74.67%	74.67%
	Predict AID	0	0	0.00%	+/-0.00%
	Class recall	100.00%	0.00%		(mikro: 74.67%)
400	Predict non-AID	324	76	81.00%	81.00%
	Predict AID	0	0	0.00%	+/-0.00%
	Class recall	100.00%	0.00%		(mikro: 81.00%)
500	Predict non-AID	424	76	84.80%	84.80%
	Predict AID	0	0	0.00%	+/-0.00%
	Class recall	100.00%	0.00%		(mikro: 84.80%)
600	Predict non-AID	524	76	87.33%	87.33%
	Predict AID	0	0	0.00%	+/-0.00%
	Class recall	100.00%	0.00%		(mikro: 87.33%)
700	Predict non-AID	624	76	89.14%	89.14%
	Predict AID	0	0	0.00%	+/-0.00%
	Class recall	100.00%	100.00%		(mikro: 89.14%)
800	Predict non-AID	724	76	90.50%	90.50%
	Predict AID	0	0	0.00%	+/-0.00%
	Class recall	100.00%	0.00%		(mikro: 90.50%)

Table B.5 The accuracy of defect count input with number of validation 10 from Neural Network algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	36	20	64.29%	66%
	Predict AID	14	30	68.18%	+/-14.28%
	Class recall	72.00%	60.00%		(mikro: 66.00%)
200	Predict non-AID	122	41	74.85%	79%
	Predict AID	2	35	94.59%	+/-7.09%
	Class recall	98.39%	46.05%		(mikro: 78.50%)
300	Predict non-AID	205	42	83.00%	79.67%
	Predict AID	19	34	64.15%	+/-11.20%
	Class recall	91.52%	44.74%		(mikro: 79.67%)
400	Predict non-AID	321	47	87.23%	87.50%
	Predict AID	3	29	90.62%	+/-2.96%
	Class recall	99.07%	38.16%		(mikro: 87.50%)
500	Predict non-AID	414	55	88.27%	87.00%
	Predict AID	10	21	67.74%	+/-3.26%
	Class recall	97.64%	27.63%		(mikro: 87.00%)
600	Predict non-AID	516	50	91.17%	90.33%
	Predict AID	8	26	76.47%	+/-3.06%
	Class recall	98.47%	34.21%		(mikro: 90.33%)
700	Predict non-AID	615	61	90.98%	90.00%
	Predict AID	9	15	62.50%	+/-2.12%
	Class recall	98.56%	19.74%		(mikro: 90.00%)
800	Predict non-AID	717	60	92.28%	91.62%
	Predict AID	7	16	69.57%	+/-1.48%
	Class recall	99.03%	21.05%		(mikro: 91.62%)

Table B.6 The accuracy of defect count input with number of validation 10 from Naïve Bayes algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	38	23	62.30%	65%
	Predict AID	12	27	69.23%	+/-17.46%
	Class recall	76.00%	54.00%		(mikro: 65.00%)
200	Predict non-AID	115	18	86.47%	87%
	Predict AID	9	58	86.57%	+/-6.73%
	Class recall	92.74%	76.32%		(mikro: 86.50%)
300	Predict non-AID	216	34	86.40%	86.00%
	Predict AID	8	42	84.00%	+/-7.72%
	Class recall	96.43%	55.26%		(mikro: 86.00%)
400	Predict non-AID	309	35	89.83%	87.50%
	Predict AID	15	41	73.21%	+/-2.24%
	Class recall	95.37%	53.95%		(mikro: 87.50%)
500	Predict non-AID	140	21	86.96%	39.00%
	Predict AID	284	55	16.22%	+/-6.40%
	Class recall	33.02%	72.37%		(mikro: 39.00%)
600	Predict non-AID	157	18	89.71%	35.83%
	Predict AID	367	58	13.65%	+/-5.12%
	Class recall	29.96%	76.32%		(mikro: 35.83%)
700	Predict non-AID	174	23	88.32%	32.43%
	Predict AID	450	53	10.54%	+/-4.43%
	Class recall	27.88%	69.74%		(mikro: 32.43%)
800	Predict non-AID	195	20	90.70%	31.38%
	Predict AID	529	56	9.57%	+/-5.20%
	Class recall	26.93%	73.68%		(mikro: 31.37%)

Table B.7 The accuracy of defect count input with number of validation 10 from Decision Tree algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	44	33	57.14%	61%
	Predict AID	6	17	73.91%	+/-11.36%
	Class recall	88.00%	34.00%		(mikro: 61.00%)
200	Predict non-AID	117	36	76.47%	79%
	Predict AID	7	40	85.11%	+/-6.34%
	Class recall	94.35%	52.63%		(mikro: 78.50%)
300	Predict non-AID	216	50	81.20%	80.67%
	Predict AID	8	26	76.47%	+/-4.90%
	Class recall	96.43%	34.21%		(mikro: 80.67%)
400	Predict non-AID	318	54	85.48%	85.00%
	Predict AID	6	22	78.57%	+/-4.61%
	Class recall	98.15%	28.95%		(mikro: 85.00%)
500	Predict non-AID	415	62	87.00%	85.80%
	Predict AID	9	14	60.87%	+/-2.75%
	Class recall	97.88%	18.42%		(mikro: 85.80%)
600	Predict non-AID	517	58	89.91%	89.17%
	Predict AID	7	18	72.00%	+/-2.27%
	Class recall	98.66%	23.68%		(mikro: 89.17%)
700	Predict non-AID	617	63	90.74%	90.00%
	Predict AID	7	13	65.00%	+/-1.69%
	Class recall	98.88%	17.11%		(mikro: 90.00%)
800	Predict non-AID	715	65	91.67%	90.75%
	Predict AID	9	11	55.00%	+/-0.61%
	Class recall	98.76%	14.47%		(mikro: 90.75%)

Table B.8 The accuracy of defect count input with number of validation 10 from Random Forest algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	26	29	47.27%	47%
	Predict AID	24	21	46.67%	+/-7.81%
	Class recall	52.00%	42.00%		(mikro: 47.00%)
200	Predict non-AID	124	76	62.00%	62%
	Predict AID	0	0	0.00%	+/-2.45%
	Class recall	100.00%	0.00%		(mikro: 62.00%)
300	Predict non-AID	224	76	74.67%	74.67%
	Predict AID	0	0	0.00%	+/-1.63%
	Class recall	100.00%	0.00%		(mikro: 74.67%)
400	Predict non-AID	324	76	81.00%	81.00%
	Predict AID	0	0	0.00%	+/-1.22%
	Class recall	100.00%	0.00%		(mikro: 81.00%)
500	Predict non-AID	424	76	84.80%	84.80%
	Predict AID	0	0	0.00%	+/-0.98%
	Class recall	100.00%	0.00%		(mikro: 84.80%)
600	Predict non-AID	524	76	87.33%	87.33%
	Predict AID	0	0	0.00%	+/-0.82%
	Class recall	100.00%	0.00%		(mikro: 87.33%)
700	Predict non-AID	624	76	89.14%	89.14%
	Predict AID	0	0	0.00%	+/-0.70%
	Class recall	100.00%	100.00%		(mikro: 89.14%)
800	Predict non-AID	724	76	90.50%	90.50%
	Predict AID	0	0	0.00%	+/-0.61%
	Class recall	100.00%	0.00%		(mikro: 90.50%)

Table B.9 The accuracy of defect size input with number of validation 4 from Neural Network algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	50	17	74.63%	83%
	Predict AID	0	33	100.00%	+/-3.32%
	Class recall	100.00%	66.00%		(mikro: 83.00%)
200	Predict non-AID	124	42	74.70%	79%
	Predict AID	0	34	100.00%	+/-3.00%
	Class recall	100.00%	44.74%		(mikro: 79.00%)
300	Predict non-AID	224	28	88.89%	90.67%
	Predict AID	0	48	100.00%	+/-4.00%
	Class recall	100.00%	63.16%		(mikro: 75.00%)
400	Predict non-AID	324	33	90.76%	91.75%
	Predict AID	0	43	100.00%	+/-1.79%
	Class recall	100.00%	56.58%		(mikro: 90.67%)
500	Predict non-AID	424	35	92.37%	93.00%
	Predict AID	0	41	100.00%	+/-2.62%
	Class recall	100.00%	53.95%		(mikro: 93.00%)
600	Predict non-AID	524	40	92.91%	93.33%
	Predict AID	0	36	100.00%	+/-1.25%
	Class recall	100.00%	47.37%		(mikro: 93.33%)
700	Predict non-AID	623	41	93.83%	94.00%
	Predict AID	1	35	97.22%	+/-0.64%
	Class recall	99.84%	46.05%		(mikro: 94.00%)
800	Predict non-AID	723	45	94.14%	94.25%
	Predict AID	1	31	96.88%	+/-0.90%
	Class recall	99.86%	40.79%		(mikro: 94.25%)

Table B.10 The accuracy of defect size input with number of validation 4 from Naïve Bayes algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	49	2	96.08%	97%
	Predict AID	1	48	97.96%	+/-1.73%
	Class recall	98.00%	96.00%		(mikro: 97.00%)
200	Predict non-AID	114	4	96.61%	93%
	Predict AID	10	72	87.80%	+/-3.32%
	Class recall	91.94%	94.74%		(mikro: 93.00%)
300	Predict non-AID	216	1	99.54%	97.00%
	Predict AID	8	75	90.36%	+/-2.19%
	Class recall	96.43%	98.68%		(mikro: 97.00%)
400	Predict non-AID	312	2	99.37%	96.75%
	Predict AID	11	74	87.06%	+/-1.30%
	Class recall	0.966	0.9737		(mikro: 96.75%)
500	Predict non-AID	409	4	99.03%	96.20%
	Predict AID	15	72	82.76%	+/-1.04%
	Class recall	96.46%	94.74%		(mikro: 96.20%)
600	Predict non-AID	510	3	99.42%	97.17%
	Predict AID	14	73	83.91%	+/-0.87%
	Class recall	97.33%	96.05%		(mikro: 97.17%)
700	Predict non-AID	612	8	98.71%	97.14%
	Predict AID	12	68	85.00%	+/-1.21%
	Class recall	98.08%	89.47%		(mikro: 97.14%)
800	Predict non-AID	705	7	99.02%	96.75%
	Predict AID	19	69	78.41%	+/-0.56%
	Class recall	97.38%	90.79%		(mikro: 96.75%)

Table B.11 The accuracy of defect size input with number of validation 4 from Decision Tree algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	48	13	78.69%	85%
	Predict AID	2	37	94.87%	+/-4.36%
	Class recall	96.00%	74.00%		(mikro: 85.00%)
200	Predict non-AID	114	14	89.06%	88%
	Predict AID	10	62	86.11%	+/-3.74%
	Class recall	91.94%	81.58%		(mikro: 88.00%)
300	Predict non-AID	221	14	94.04%	94.33%
	Predict AID	3	62	95.38%	+/-6.28%
	Class recall	98.66%	81.58%		(mikro: 94.33%)
400	Predict non-AID	321	8	97.57%	97.25%
	Predict AID	3	68	95.77%	+/-1.30%
	Class recall	99.07%	89.47%		(mikro: 97.25%)
500	Predict non-AID	415	14	96.74%	95.40%
	Predict AID	9	62	87.32%	+/-1.04%
	Class recall	97.88%	81.58%		(mikro: 95.40%)
600	Predict non-AID	517	13	97.55%	96.67%
	Predict AID	7	63	90.00%	+/-1.70%
	Class recall	98.66%	82.89%		(mikro: 96.67%)
700	Predict non-AID	618	17	97.32%	96.71%
	Predict AID	6	59	90.77%	+/-1.30%
	Class recall	99.04%	77.63%		(mikro: 96.71%)
800	Predict non-AID	716	13	98.22%	97.38%
	Predict AID	8	63	88.73%	+/-0.54%
	Class recall	98.90%	82.89%		(mikro: 97.38%)

Table B.12 The accuracy of defect size input with number of validation 4 from Random Forest algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	49	32	60.49%	67%
	Predict AID	1	18	94.74%	+/-9.54%
	Class recall	98.00%	36.00%		(mikro: 67.00%)
200	Predict non-AID	124	69	62.25%	66%
	Predict AID	0	7	100.00%	+/-1.66%
	Class recall	100.00%	9.21%		(mikro: 65.50%)
300	Predict non-AID	223	72	75.59%	75.67%
	Predict AID	1	4	80.00%	+/-0.58%
	Class recall	99.55%	5.26%		(mikro: 75.67%)
400	Predict non-AID	324	73	81.61%	81.75%
	Predict AID	0	3	100.00%	+/-0.83%
	Class recall	100.00%	3.95%		(mikro: 81.75%)
500	Predict non-AID	423	72	85.45%	85.40%
	Predict AID	1	4	80.00%	+/-0.66%
	Class recall	99.76%	5.26%		(mikro: 85.40%)
600	Predict non-AID	524	73	87.77%	87.83%
	Predict AID	0	3	100.00%	+/-0.55%
	Class recall	100.00%	3.95%		(mikro: 87.83%)
700	Predict non-AID	623	71	89.77%	89.71%
	Predict AID	1	5	83.33%	+/-0.7%
	Class recall	99.84%	6.58%		(mikro: 89.71%)
800	Predict non-AID	724	74	90.73%	90.75%
	Predict AID	0	2	100.00%	+/-0.25%
	Class recall	100.00%	2.63%		(mikro: 90.75%)

Table B.13 The accuracy of defect size input with number of validation 10 from Neural Network algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	50	20	71.43%	80%
	Predict AID	0	30	100.00%	+/-10.00%
	Class recall	100.00%	60.00%		(mikro: 80.00%)
200	Predict non-AID	124	31	80.00%	85%
	Predict AID	0	45	100.00%	+/-4.72%
	Class recall	100.00%	59.21%		(mikro: 84.50%)
300	Predict non-AID	224	32	87.50%	89.33%
	Predict AID	0	44	100.00%	+/-5.54%
	Class recall	100.00%	57.89%		(mikro: 89.33%)
400	Predict non-AID	324	35	90.25%	91.25%
	Predict AID	0	41	100.00%	+/-4.51%
	Class recall	100.00%	53.95%		(mikro: 91.25%)
500	Predict non-AID	424	38	91.77%	92.40%
	Predict AID	0	38	100.00%	+/-2.50%
	Class recall	100.00%	50.00%		(mikro: 92.40%)
600	Predict non-AID	524	40	92.91%	93.33%
	Predict AID	0	36	100.00%	+/-1.83%
	Class recall	100.00%	47.37%		(mikro: 93.33%)
700	Predict non-AID	624	41	93.83%	94.14%
	Predict AID	0	35	100.00%	+/-1.74%
	Class recall	100.00%	46.05%		(mikro: 94.14%)
800	Predict non-AID	723	40	94.76%	94.88%
	Predict AID	1	36	97.30%	+/-1.53%
	Class recall	99.86%	47.37%		(mikro: 94.88%)

Table B.14 The accuracy of defect size input with number of validation 10 from Naïve Bayes algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	49	1	98.00%	98%
	Predict AID	1	49	98.00%	+/-4.00%
	Class recall	98.00%	98.00%		(mikro: 98.00%)
200	Predict non-AID	116	3	97.48%	95%
	Predict AID	8	73	90.12%	+/-5.22%
	Class recall	93.55%	96.05%		(mikro: 94.50%)
300	Predict non-AID	218	3	98.64%	97.00%
	Predict AID	6	73	92.41%	+/-3.48%
	Class recall	97.32%	96.05%		(mikro: 97.00%)
400	Predict non-AID	313	2	99.37%	96.75%
	Predict AID	11	74	87.06%	+/-2.25%
	Class recall	96.06%	97.37%		(mikro: 96.75%)
500	Predict non-AID	413	3	99.28%	97.20%
	Predict AID	11	73	86.90%	+/-1.60%
	Class recall	97.41%	96.05%		(mikro: 97.20%)
600	Predict non-AID	511	3	99.42%	97.33%
	Predict AID	13	73	84.88%	+/-2.49%
	Class recall	97.52%	96.05%		(mikro: 97.33%)
700	Predict non-AID	612	9	98.55%	97.00%
	Predict AID	12	67	84.81%	+/-1.49%
	Class recall	98.08%	88.16%		(mikro: 97.00%)
800	Predict non-AID	707	9	98.74%	96.75%
	Predict AID	17	67	79.76%	+/-1.87%
	Class recall	97.65%	88.16%		(mikro: 96.75%)

Table B.15 The accuracy of defect size input with number of validation 10 from Decision Tree algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	48	9	84.21%	89%
	Predict AID	2	41	95.35%	+/-9.43%
	Class recall	96.00%	82.00%		(mikro: 89.00%)
200	Predict non-AID	116	9	92.80%	92%
	Predict AID	8	67	89.33%	+/-4.50%
	Class recall	93.55%	88.16%		(mikro: 91.50%)
300	Predict non-AID	220	10	95.65%	95.33%
	Predict AID	4	66	94.29%	+/-4.00%
	Class recall	98.21%	86.84%		(mikro: 95.33%)
400	Predict non-AID	321	9	97.27%	97.00%
	Predict AID	3	67	95.71%	+/-2.45%
	Class recall	99.07%	88.16%		(mikro: 97.00%)
500	Predict non-AID	416	12	97.20%	96.00%
	Predict AID	8	64	88.89%	+/-2.00%
	Class recall	98.11%	84.21%		(mikro: 96.00%)
600	Predict non-AID	515	12	97.72%	96.50%
	Predict AID	9	64	87.67%	+/-2.83%
	Class recall	98.28%	84.21%		(mikro: 96.50%)
700	Predict non-AID	615	14	97.77%	96.71%
	Predict AID	9	62	87.32%	+/-0.65%
	Class recall	98.56%	81.58%		(mikro: 96.71%)
800	Predict non-AID	716	13	98.22%	97.37%
	Predict AID	8	63	88.73%	+/-1.31%
	Class recall	98.90%	82.89%		(mikro: 97.38%)

Table B.16 The accuracy of defect size input with number of validation 10 from Random Forest algorithm.

Drive Quantity	Predict Group	True non-AID	True AID	Class Precision	Total Accuracy
100	Predict non-AID	46	34	57.50%	62%
	Predict AID	4	16	80.00%	+/-7.48%
	Class recall	92.00%	32.00%		(mikro: 62.00%)
200	Predict non-AID	122	71	63.21%	64%
	Predict AID	2	5	71.43%	+/-5.50%
	Class recall	98.39%	6.58%		(mikro: 63.50%)
300	Predict non-AID	224	74	75.17%	75.33%
	Predict AID	0	2	100.00%	+/-1.63%
	Class recall	100.00%	2.63%		(mikro: 75.33%)
400	Predict non-AID	324	72	81.82%	82.00%
	Predict AID	0	4	100.00%	+/-1.87%
	Class recall	1	0.0526		(mikro: 82.00%)
500	Predict non-AID	423	73	85.28%	85.20%
	Predict AID	1	3	75.00%	+/-1.33%
	Class recall	99.76%	3.95%		(mikro: 85.20%)
600	Predict non-AID	523	70	88.20%	88.17%
	Predict AID	1	6	85.71%	+/-1.74%
	Class recall	99.81%	7.89%		(mikro: 88.17%)
700	Predict non-AID	624	68	90.17%	90.29%
	Predict AID	0	8	100.00%	+/-1.25%
	Class recall	100.00%	10.53%		(mikro: 90.29%)
800	Predict non-AID	724	74	90.73%	90.75%
	Predict AID	0	0	100.00%	+/-0.61%
	Class recall	100.00%	2.63%		(mikro: 90.75%)

APPENDIX C

PYTHON SOURCE CODE

Data Pre-processing: defect count by head zone extraction

```
import sys, string, csv, time, os, re
Location = "C:\\DriveE\\Master\\Thesis\\Experiment\\All_FAIL_AID" #Destination address


outputFile = "P109Resultfx.xls"  #Extract file name
hdr =
['SN','Hd0_Zn0','Hd0_Zn1','Hd0_Zn2','Hd0_Zn3','Hd0_Zn4','Hd0_Zn5','Hd0_Zn6','Hd0_Zn7',
'Hd0_Zn8','Hd0_Zn9','Hd0_Zn10','Hd0_Zn11','Hd0_Zn12','Hd0_Zn13','Hd0_Zn14','Hd0_Zn15',
'Hd0_Zn16','Hd0_Zn17','Hd1_Zn0','Hd1_Zn1','Hd1_Zn2','Hd1_Zn3','Hd1_Zn4','Hd1_Zn5','Hd1
_Zn6','Hd1_Zn7','Hd1_Zn8','Hd1_Zn9','Hd1_Zn10','Hd1_Zn11','Hd1_Zn12','Hd1_Zn13','Hd1_Z
n14','Hd1_Zn15','Hd1_Zn16','Hd1_Zn17','Hd2_Zn0','Hd2_Zn1','Hd2_Zn2','Hd2_Zn3','Hd2_Zn4
','Hd2_Zn5','Hd2_Zn6','Hd2_Zn7','Hd2_Zn8','Hd2_Zn9','Hd2_Zn10','Hd2_Zn11','Hd2_Zn12','
Hd2_Zn13','Hd2_Zn14','Hd2_Zn15','Hd2_Zn16','Hd2_Zn17','Hd3_Zn0','Hd3_Zn1','Hd3_Zn2','H
d3_Zn3','Hd3_Zn4','Hd3_Zn5','Hd3_Zn6','Hd3_Zn7','Hd3_Zn8','Hd3_Zn9','Hd3_Zn10','Hd3_Zn
11','Hd3_Zn12','Hd3_Zn13','Hd3_Zn14','Hd3_Zn15','Hd3_Zn16','Hd3_Zn17','Hd4_Zn0','Hd4_Z
n1','Hd4_Zn2','Hd4_Zn3','Hd4_Zn4','Hd4_Zn5','Hd4_Zn6','Hd4_Zn7','Hd4_Zn8','Hd4_Zn9','H
d4_Zn10','Hd4_Zn11','Hd4_Zn12','Hd4_Zn13','Hd4_Zn14','Hd4_Zn15','Hd4_Zn16','Hd4_Zn17',
'Hd5_Zn0','Hd5_Zn1','Hd5_Zn2','Hd5_Zn3','Hd5_Zn4','Hd5_Zn5','Hd5_Zn6','Hd5_Zn7','Hd5_Z
n8','Hd5_Zn9','Hd5_Zn10','Hd5_Zn11','Hd5_Zn12','Hd5_Zn13','Hd5_Zn14','Hd5_Zn15','Hd5_Z
n16','Hd5_Zn17','Hd6_Zn0','Hd6_Zn1','Hd6_Zn2','Hd6_Zn3','Hd6_Zn4','Hd6_Zn5','Hd6_Zn6',
'Hd6_Zn7','Hd6_Zn8','Hd6_Zn9','Hd6_Zn10','Hd6_Zn11','Hd6_Zn12','Hd6_Zn13','Hd6_Zn14','
Hd6_Zn15','Hd6_Zn16','Hd6_Zn17','Hd7_Zn0','Hd7_Zn1','Hd7_Zn2','Hd7_Zn3','Hd7_Zn4','Hd7
_Zn5','Hd7_Zn6','Hd7_Zn7','Hd7_Zn8','Hd7_Zn9','Hd7_Zn10','Hd7_Zn11','Hd7_Zn12','Hd7_Zn
13','Hd7_Zn14','Hd7_Zn15','Hd7_Zn16','Hd7_Zn17','Hd8_Zn0','Hd8_Zn1','Hd8_Zn2','Hd8_Zn3
','Hd8_Zn4','Hd8_Zn5','Hd8_Zn6','Hd8_Zn7','Hd8_Zn8','Hd8_Zn9','Hd8_Zn10','Hd8_Zn11','H
d8_Zn12','Hd8_Zn13','Hd8_Zn14','Hd8_Zn15','Hd8_Zn16','Hd8_Zn17','Hd9_Zn0','Hd9_Zn1','H
d9_Zn2','Hd9_Zn3','Hd9_Zn4','Hd9_Zn5','Hd9_Zn6','Hd9_Zn7','Hd9_Zn8','Hd9_Zn9','Hd9_Zn1
0','Hd9_Zn11','Hd9_Zn12','Hd9_Zn13','Hd9_Zn14','Hd9_Zn15','Hd9_Zn16','Hd9_Zn17','Statu
s'] #Column header
status = 'non-AID' #For non-AID result file extraction.
state_data = ''
P109_data = ''

import xlwt

ezxf = xlwt.easyxf

def write_xls(file_name, sheet_name, headings, data):
    book = xlwt.Workbook()
    sheet = book.add_sheet(sheet_name)
    rowx = 0
    for colx, value in enumerate(headings):
        sheet.write(rowx, colx, value)
    for row in data:
        rowx += 1
        for colx, value in enumerate(row):
            sheet.write(rowx, colx, value)
    book.save(file_name)

def prep_data_P109(state_name = 'INIT'):

    for file_round,the_file in enumerate(os.listdir(Location)):
        each_raw = []
        each_raw.append(the_file[:8])

        this_file = os.path.join(Location,the_file)
```

Data Pre-processing: defect count by head zone extraction

(cont.)

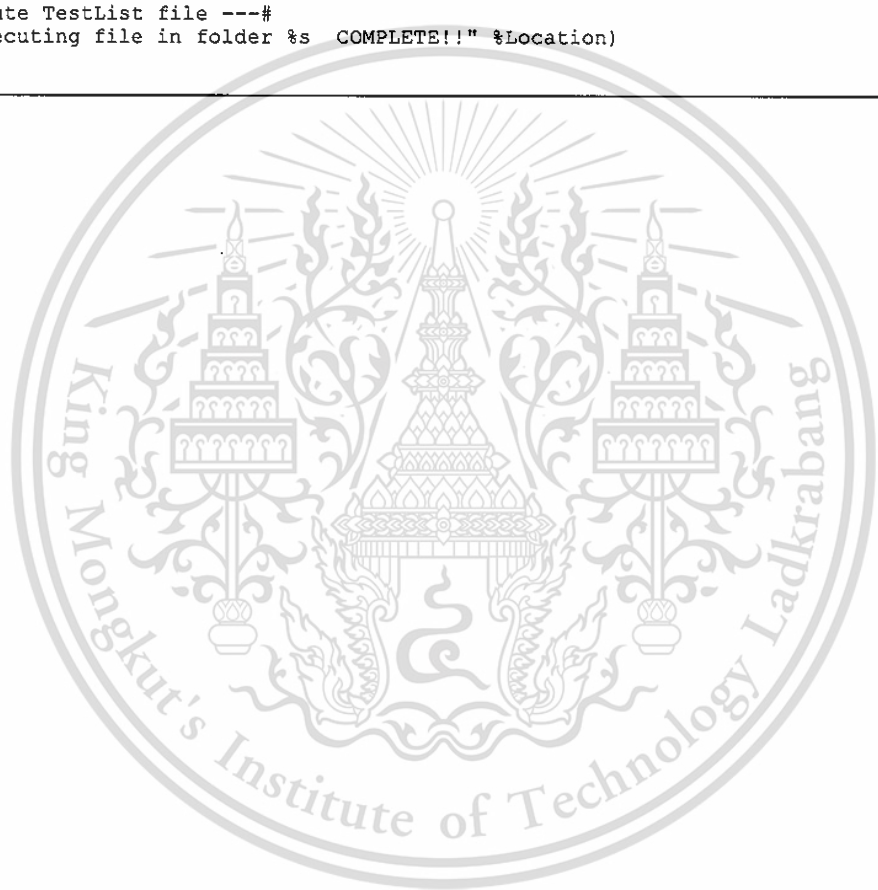
```
        if os.path.isdir(this_file):
            dirCnt = dirCnt + 1
            print "sub directory feature is not support in this version"
        else:
            print "Extracting %s \n" %the_file
            file = open(this_file,'r')
            log_data = file.read()
            file.close()
            state_match = re.search("^.*?%s : BEGIN.*?\n(?P<data>.*?)(\*\*\*\*\* FAULT
DETECTED \*\*\*\*\*|: COMPLETE.*?\n){1}"%state_name,log_data, re.M | re.DOTALL)
            if state_match:
                state_data = state_match.group('data')
                P109match =
re.search("P109_SUM_HD_ZONE:.*?\n(?P<data>.*?)P109_TOT_VFYD_UNVFYD_FLAW",state_data,
re.M | re.DOTALL)
                if P109match:
                    P109_data = P109match.group('data')
                    pat = re.compile("(?P<HD_PHYS_PSN>[\d.]+)[
\t]+(?P<DATA_ZONE>[\d.]+)[ \t]+(?P<HD_LGC_PSN>[\d.]+)[ \t]+(?P<VFYD_ERRS>[\d.]+)[
\t]+(?P<UNVFYD_ERRS>[\d.]+)[ \t]+(?P<HD_STATUS>[\d.]+)")
                    for line in P109_data.splitlines():
                        match = pat.search(line)
                        if match:
                            each_raw.append(int(match.group('UNVFYD_ERRS')))
                    each_raw.append(status)
                else:
                    print 'No P109 table match!!!'
            else:
                print 'No state match!!!'
        All_data.append(each_raw)

All_data = []
prep_data_P109(state_name = 'AID_SCREEN')
#write_xls(os.path.join(os.path.dirname(__file__),outputFile) ,"Sheet name" ,hdr ,
[['D','E','F'],['G','H','I']])
write_xls(os.path.join(os.path.dirname(__file__),outputFile) ,"Sheet name" ,hdr ,
All_data)

#--- execute TestList file ---#
print("executing file in folder %s  COMPLETE!!" %Location)
```

Data Pre-processing: maximum defect size per surface extraction.

```
import sys, string, csv, time, os, re
Location = "C:\\Users\\336764\\Documents\\Master\\Thesis\\fix" #Destination address
outputFile = "fix.xls"
hdr = ['SN','Hd0','Hd1','Hd2','Hd3','Hd4','Hd5','Hd6','Hd7','Hd8','Hd9','Status']
#Column header
status = 'AID'
state_data = ''
P109_data = ''

import xlwt

ezxf = xlwt.easyxf

def write_xls(file_name, sheet_name, headings, data):
    book = xlwt.Workbook()
    sheet = book.add_sheet(sheet_name)
    rowx = 0
    for colx, value in enumerate(headings):
        sheet.write(rowx, colx, value)
    for row in data:
        rowx += 1
        for colx, value in enumerate(row):
            sheet.write(rowx, colx, value)
    book.save(file_name)

def prep_data_P117(state_name = 'INIT'):

    for file_round,the_file in enumerate(os.listdir(Location)):
        each_raw = []
        each_raw.append(the_file[:8])

        this_file = os.path.join(Location,the_file)
        if os.path.isdir(this_file):
            dirCnt = dirCnt + 1
            print "sub directory feature is not support in this version"
        else:
            print "Extracting %s \n" %the_file
            file = open(this_file,'r')
            log_data = file.read()
            file.close()
            state_match = re.search("^.*?%s : BEGIN.*?\n(?P<data>.*?)(\*\*\*\*\* FAULT
DETECTED \*\*\*\*\*|: COMPLETE.*?\n){1}"%state_name,log_data, re.M | re.DOTALL)
            if state_match:
                state_data = state_match.group('data')
                P117match = re.search("P117_MEDIA_SCREEN:.*?\n(?P<data>.*?)F I N I S H
E D   Testing st",state_data, re.M | re.DOTALL)
                if P117match:
                    P117_data = P117match.group('data')
                    P117_dict = {}
                    for i in range(10):
                        P117_dict[i] = [0]
                    print P117_dict
                    pat = re.compile("(?P<HD_PHYS_PSN>[\d.]+)[
\t]+(?P<BEGINNING_TRK>[\d.]+)[ \t]+(?P<SCRATCH_ID>[\d.]+)[ \t]+(?P<HD_LGC_PSN>[\d.]+)[
\t]+(?P<ENDING_TRK>[\d.]+)[ \t]+(?P<SCRATCH_LENGTH>[\d.]+)[ \t]+(?P<DEFECTS>[\d.]+)[
\t]+(?P<TOTAL_BYTES>[\d.]+)[ \t]+(?P<SCREEN_PF_FLAG>[\S.]+)")
                    for line in P117_data.splitlines():
                        match = pat.search(line)
                        if match:

P117_dict[int((match.group('HD_PHYS_PSN')))].append(int(match.group('TOTAL_BYTES')))
                    for j in range(10):
                        each_raw.append(int(max(P117_dict[j])))
                    each_raw.append(status)
```

Data Pre-processing: maximum defect size per surface extraction. (cont.)

```
                else:
                    print 'No P109 table match!!!'
            else:
                print 'No state match!!!'
        All_data.append(each_raw)

All_data = []
prep_data_P117(state_name = 'AID_SCREEN')
write_xls(os.path.join(os.path.dirname(__file__),outputFile) ,"Sheet name" ,hdr ,
All_data)

#--- execute TestList file ---#
print("executing file in folder %s  COMPLETE!!" %Location)
```



APPENDIX D

APPLICATION USAGE

The application usage based on RapidMiner Studio Basic version 6.5

1. Open the application.
2. File > New Process

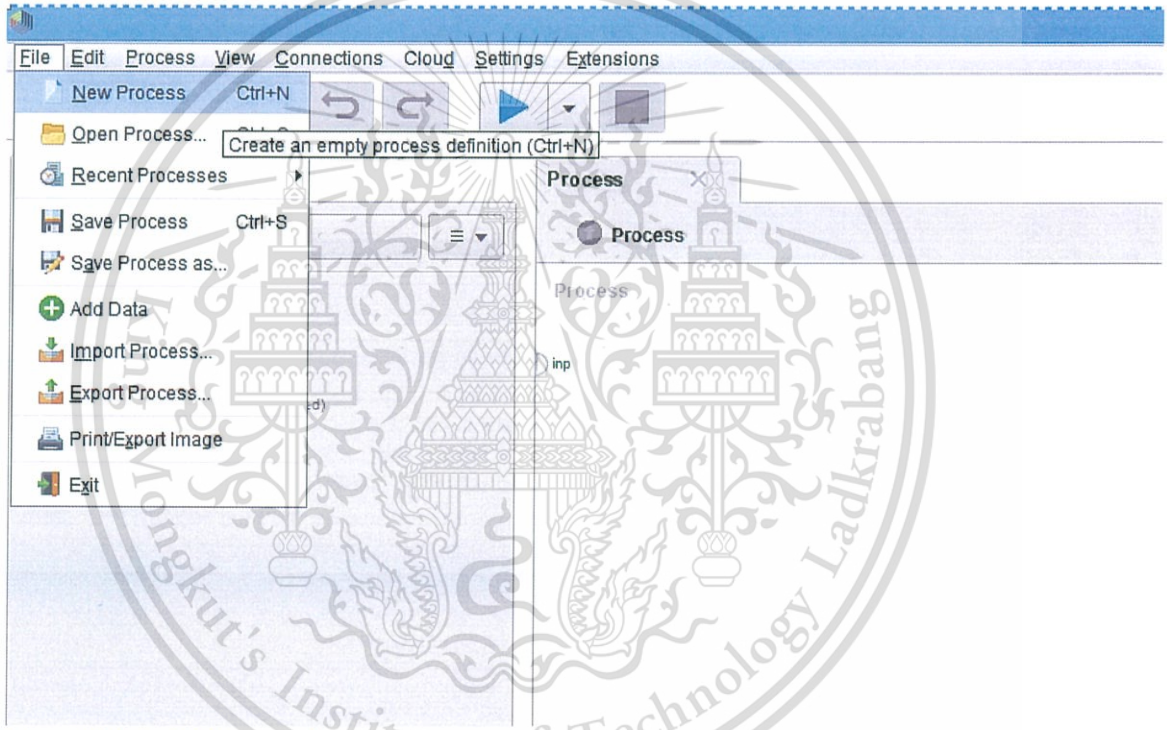


Figure D.1 RapidMiner main page.

3. Import > Data > Read Excel.

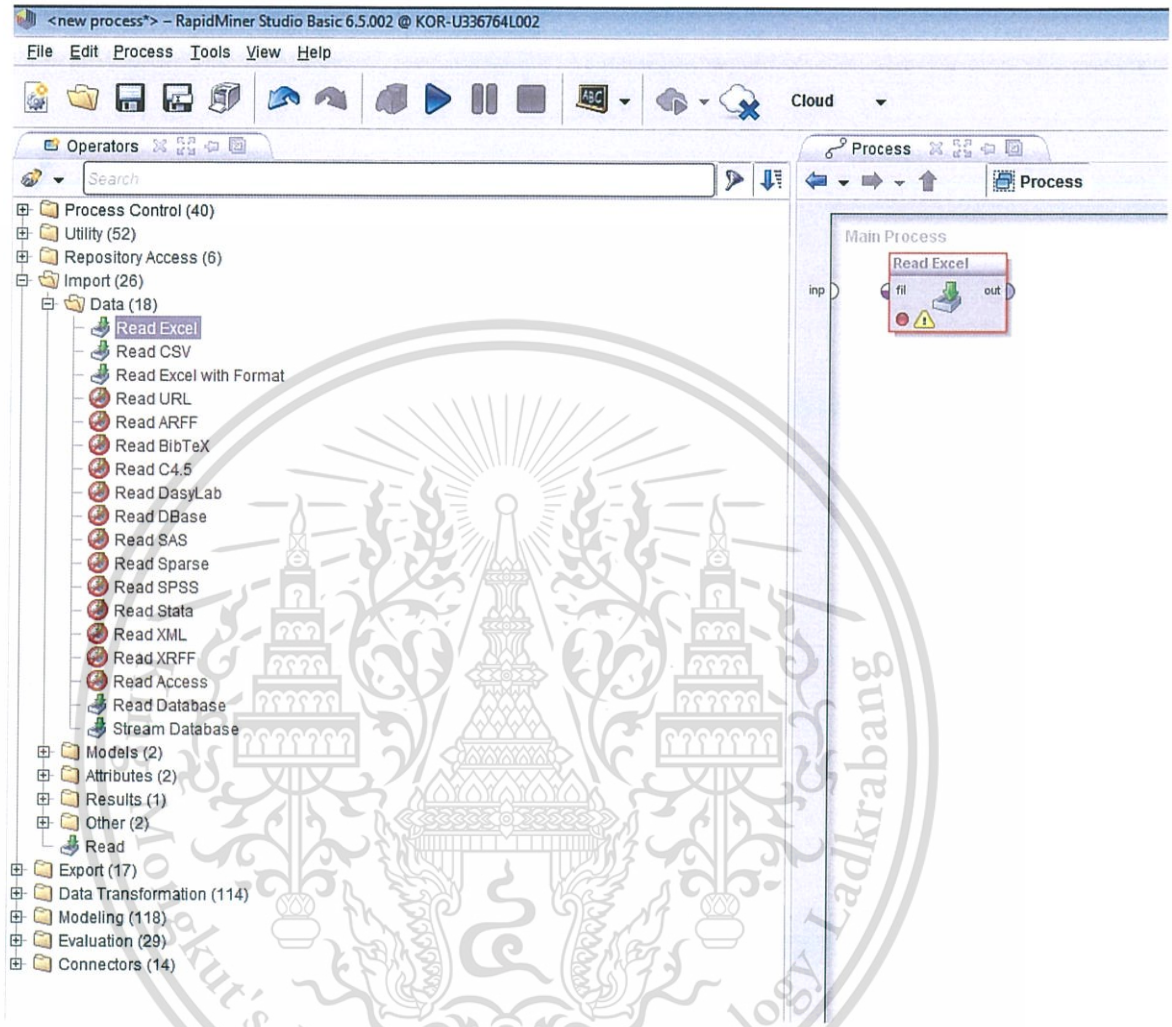


Figure D.2 Read Excel input.

4. Evaluation > Validation > %X-Validation.

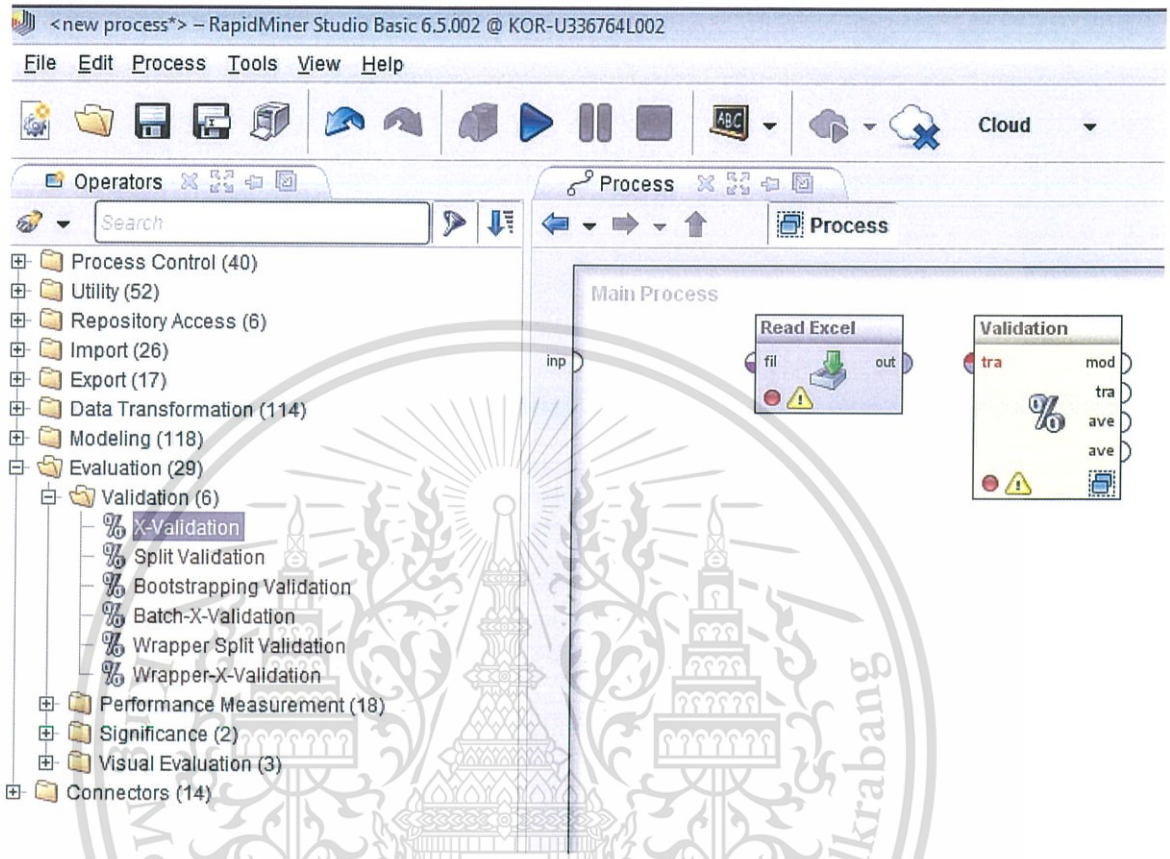


Figure D.3 X-Validation.

5. Connect link in as shown in figure C.4.

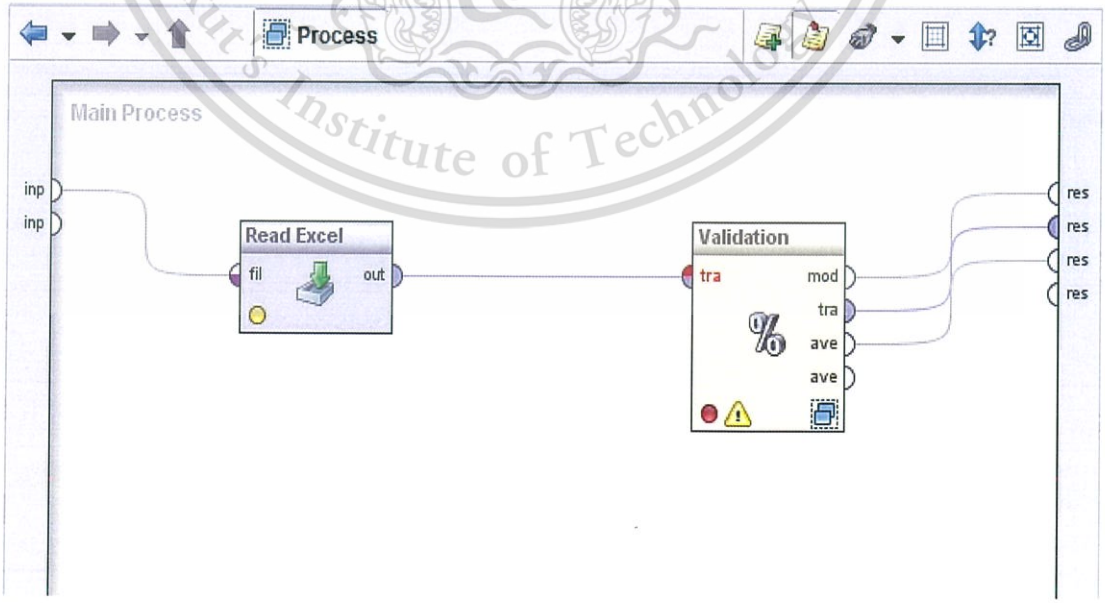


Figure D.4 Main connection.

6. Select input data by Read Excel>Import Configuration Wizard.

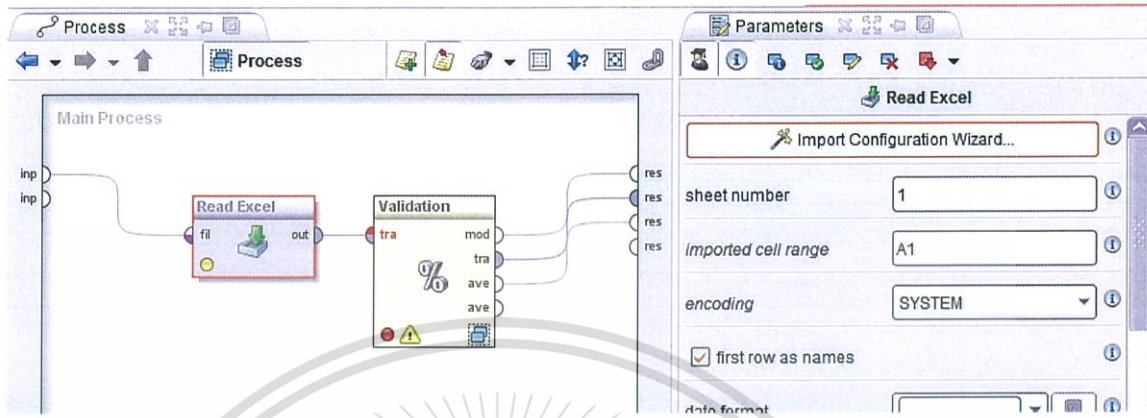


Figure D.5 Import Configuration Wizard.

7. Select the input data from target address then click 'Next'.

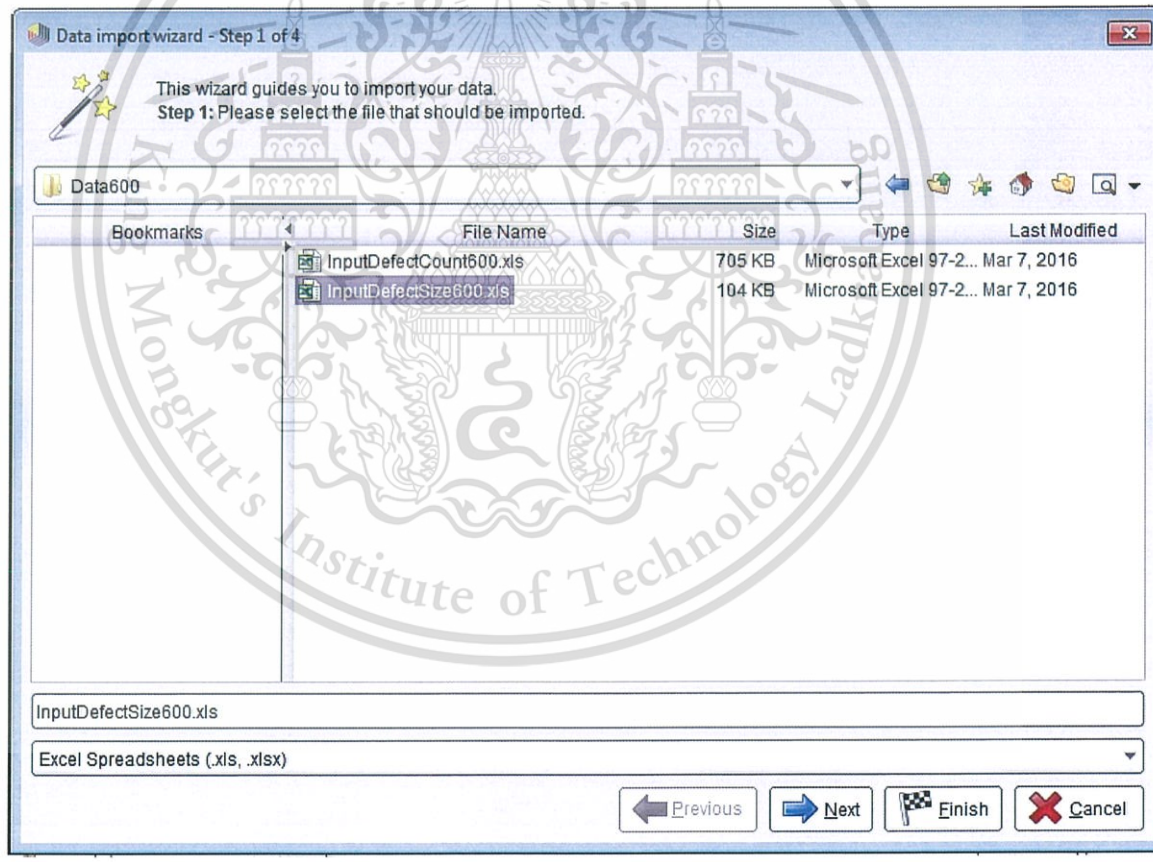
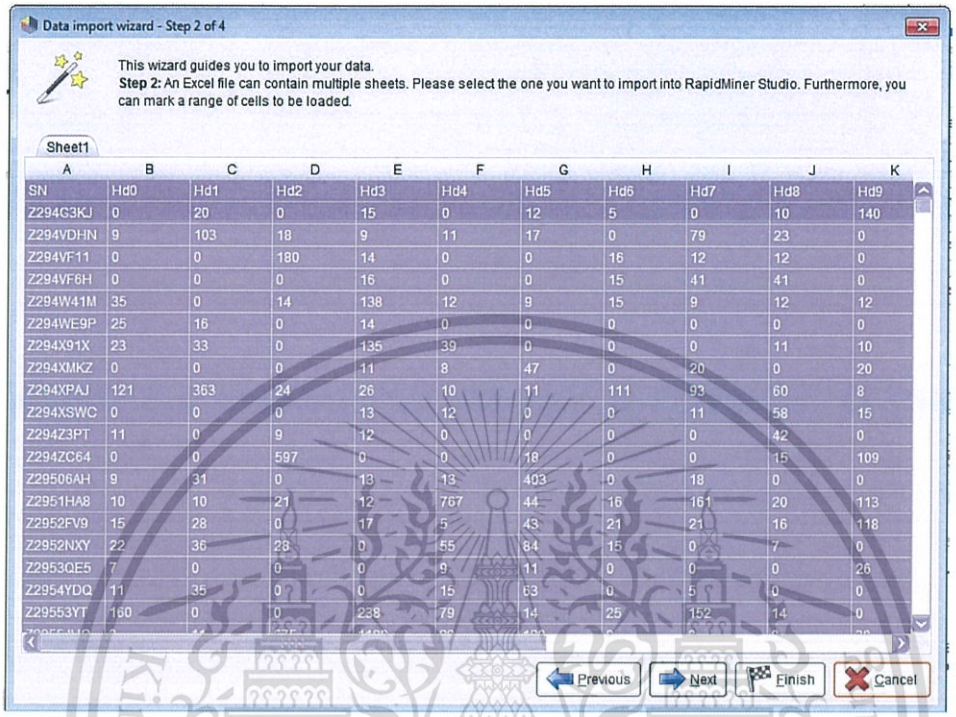


Figure D.6 Select Input File.

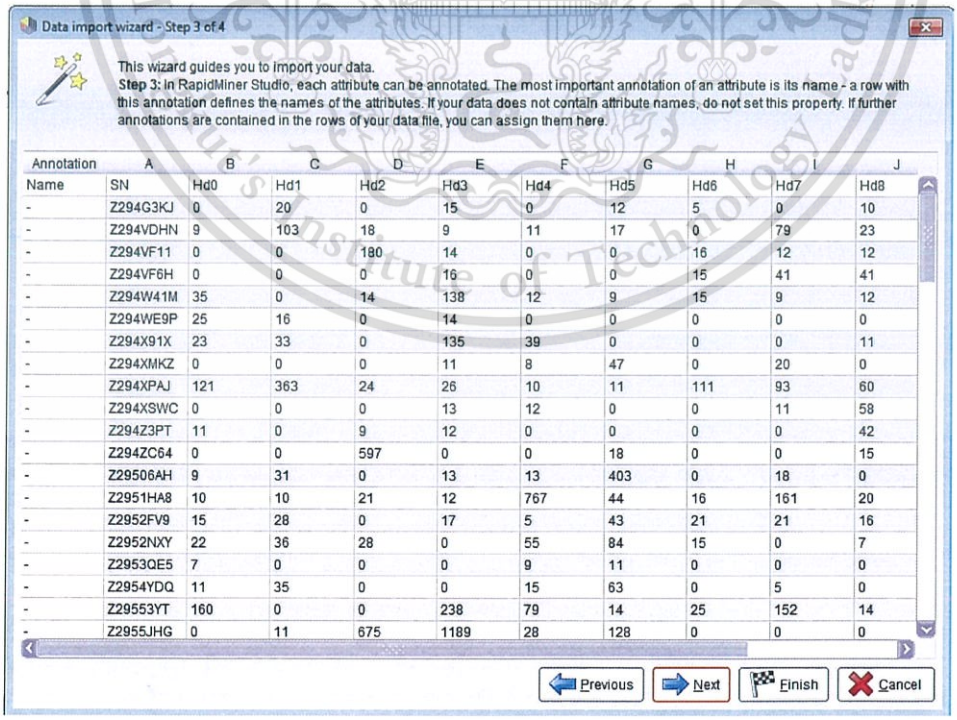
8. Click ‘Next’.



A	B	C	D	E	F	G	H	I	J	K
SN	Hd0	Hd1	Hd2	Hd3	Hd4	Hd5	Hd6	Hd7	Hd8	Hd9
Z294G3KJ	0	20	0	15	0	12	5	0	10	140
Z294VDHN	9	103	18	9	11	17	0	79	23	0
Z294VF11	0	0	180	14	0	0	16	12	12	0
Z294VF6H	0	0	0	16	0	0	15	41	41	0
Z294W41M	35	0	14	138	12	9	15	9	12	12
Z294WE9P	25	16	0	14	0	0	0	0	0	0
Z294X91X	23	33	0	135	39	0	0	0	11	10
Z294XMKZ	0	0	0	11	8	47	0	20	0	20
Z294XPAJ	121	363	24	26	10	11	111	93	60	8
Z294XSWC	0	0	0	13	12	0	0	11	58	15
Z294Z3PT	11	0	9	12	0	0	0	0	42	0
Z294ZC64	0	0	597	0	0	18	0	0	15	109
Z29506AH	9	31	0	13	13	403	0	18	0	0
Z2951HA8	10	10	21	12	767	44	16	161	20	113
Z2952FV9	15	28	0	17	5	43	21	21	16	118
Z2952NXY	22	36	28	0	55	84	15	0	7	0
Z2953QE5	7	0	0	0	9	11	0	0	0	26
Z2954YDQ	11	35	0	0	15	63	0	5	0	0
Z29553YT	160	0	0	238	79	14	25	152	14	0

Figure D.7 Input file set up 1.

9. Click ‘Next’.



Annotation	A	B	C	D	E	F	G	H	I	J
Name	SN	Hd0	Hd1	Hd2	Hd3	Hd4	Hd5	Hd6	Hd7	Hd8
-	Z294G3KJ	0	20	0	15	0	12	5	0	10
-	Z294VDHN	9	103	18	9	11	17	0	79	23
-	Z294VF11	0	0	180	14	0	0	16	12	12
-	Z294VF6H	0	0	0	16	0	0	15	41	41
-	Z294W41M	35	0	14	138	12	9	15	9	12
-	Z294WE9P	25	16	0	14	0	0	0	0	0
-	Z294X91X	23	33	0	135	39	0	0	0	11
-	Z294XMKZ	0	0	0	11	8	47	0	20	0
-	Z294XPAJ	121	363	24	26	10	11	111	93	60
-	Z294XSWC	0	0	0	13	12	0	0	11	58
-	Z294Z3PT	11	0	9	12	0	0	0	0	42
-	Z294ZC64	0	0	597	0	0	18	0	0	15
-	Z29506AH	9	31	0	13	13	403	0	18	0
-	Z2951HA8	10	10	21	12	767	44	16	161	20
-	Z2952FV9	15	28	0	17	5	43	21	21	16
-	Z2952NXY	22	36	28	0	55	84	15	0	7
-	Z2953QE5	7	0	0	0	9	11	0	0	0
-	Z2954YDQ	11	35	0	0	15	63	0	5	0
-	Z29553YT	160	0	0	238	79	14	25	152	14
-	Z2955JHG	0	11	675	1189	28	128	0	0	0

Figure D.8 Input file set up 2.

10. Select ID.

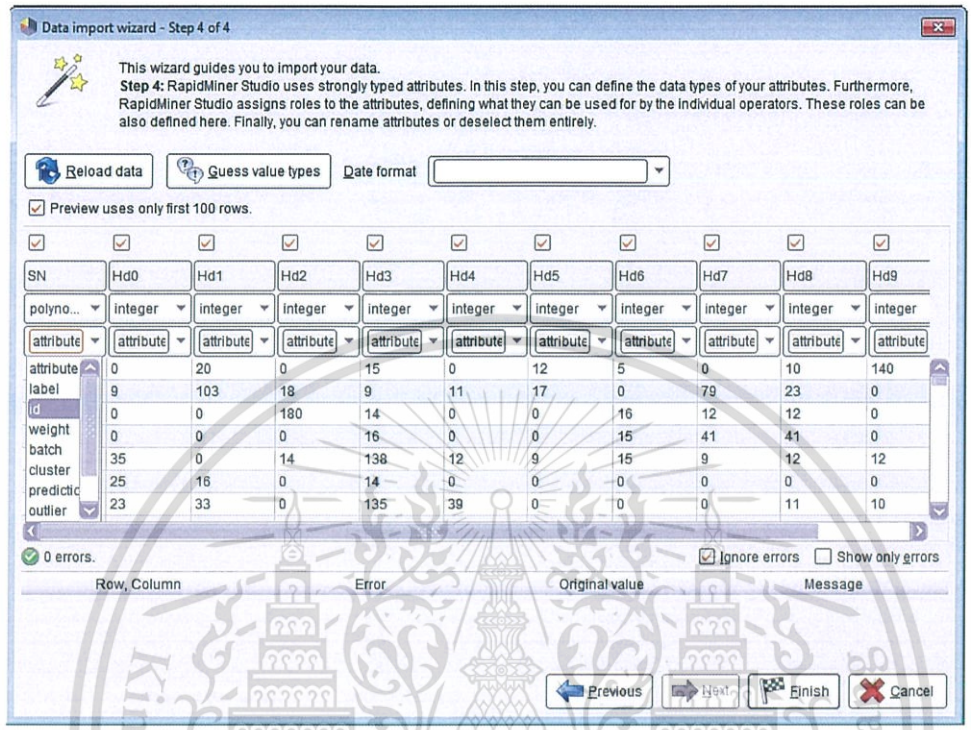


Figure D.9 Input file set up 3.

11. Select label then click 'Finish'.

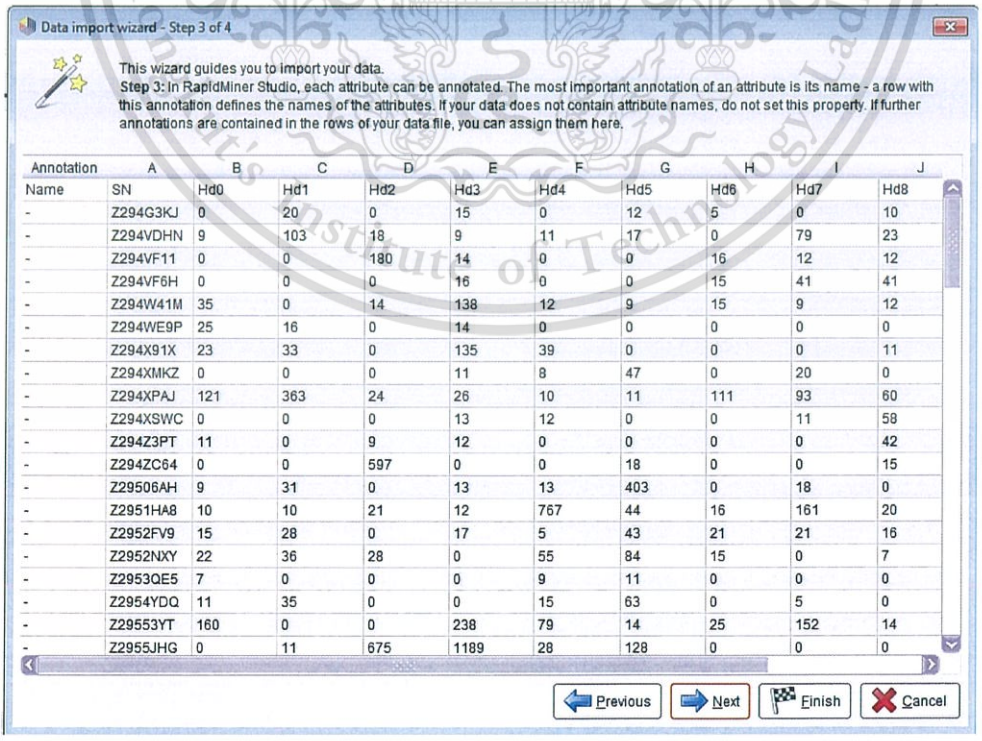


Figure D.10 Input file set up 4.

12. Double click at Validation.

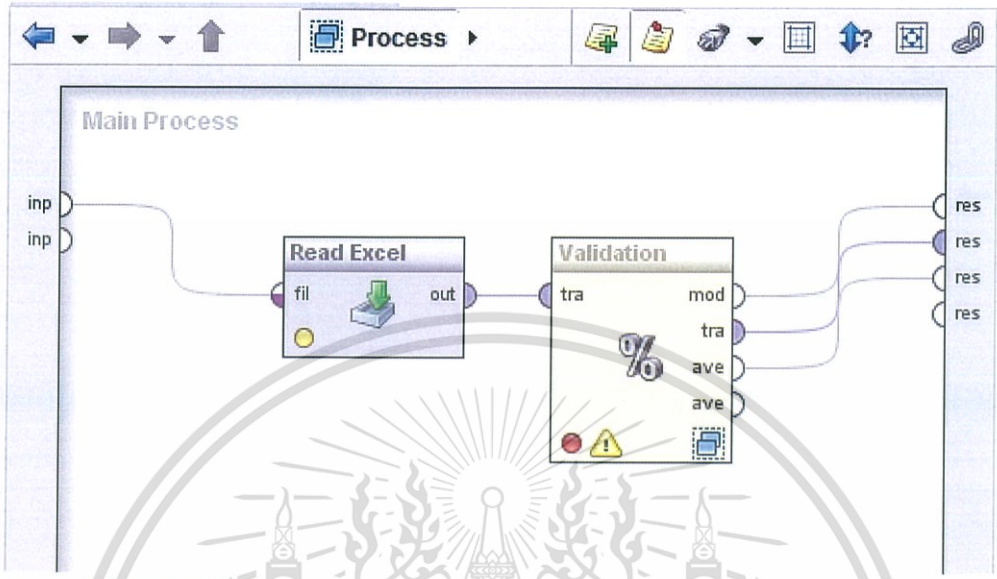


Figure D.11 % Validation main page.

13. Modelling > Classification and Regression. Then select the interested model and connect the link.

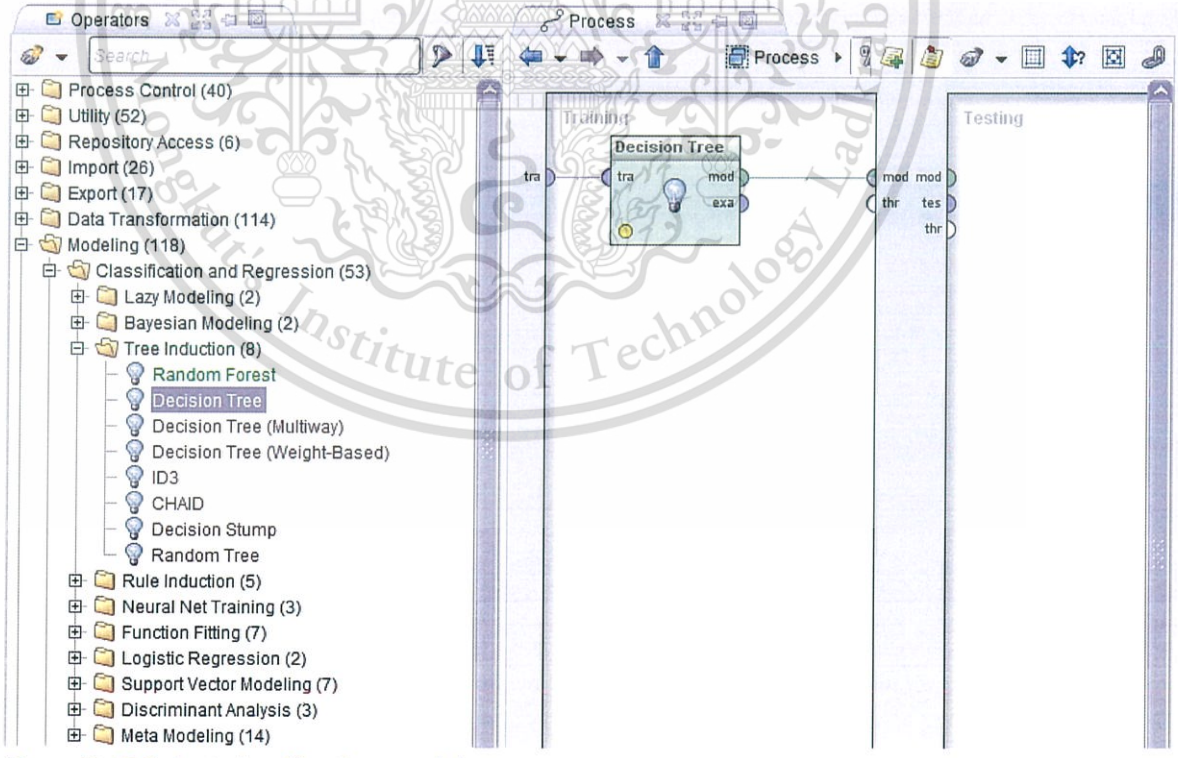


Figure D.12 Select classification model.

14. Modelling > Model Application > Apply Model.

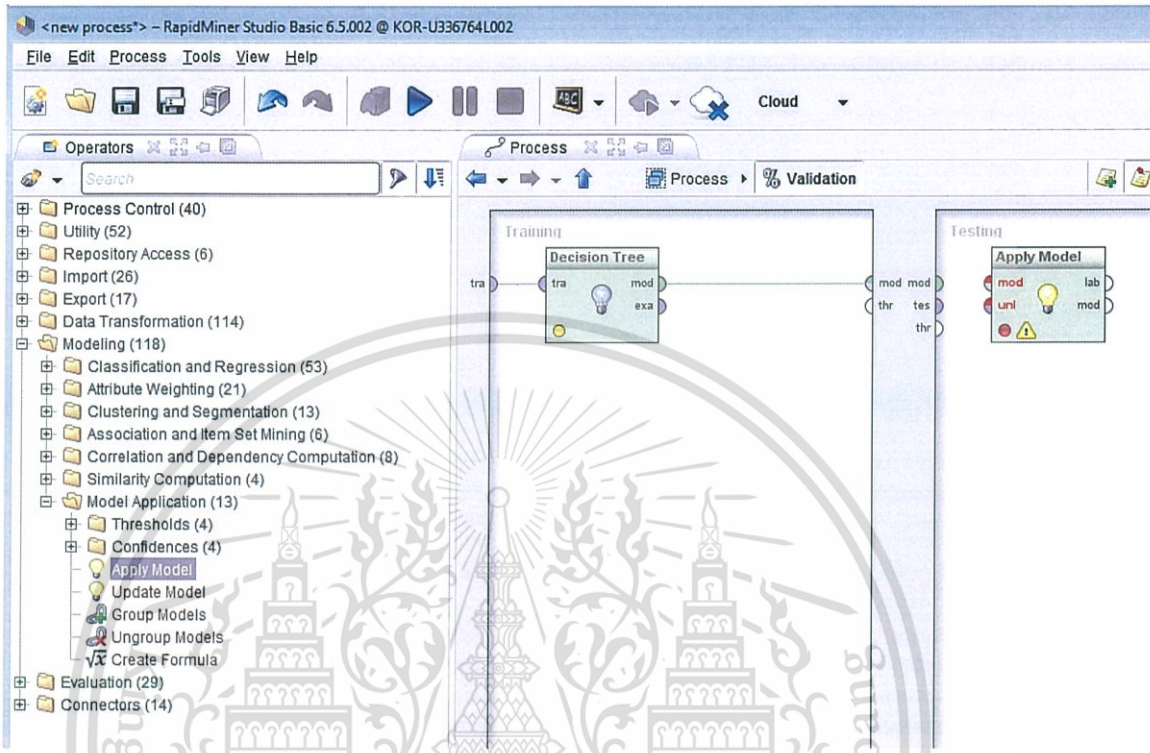


Figure D.13 Apply Model selection.

15. Evaluation > Performance Measurement > %Performance. Then connect the link as shown in fiture C.14.

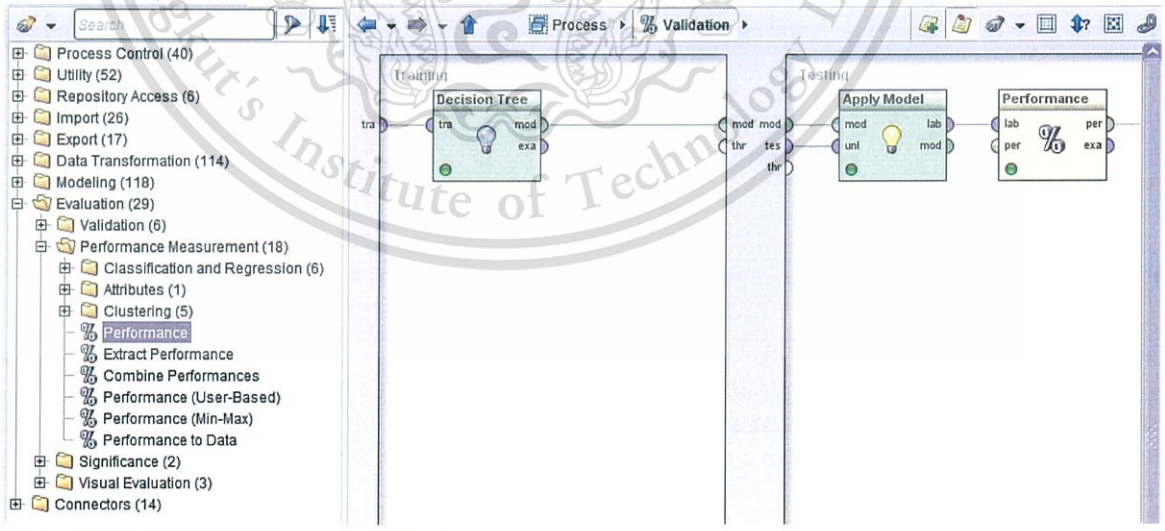


Figure D.14 % Validation connecting.

16. Select number of validation.

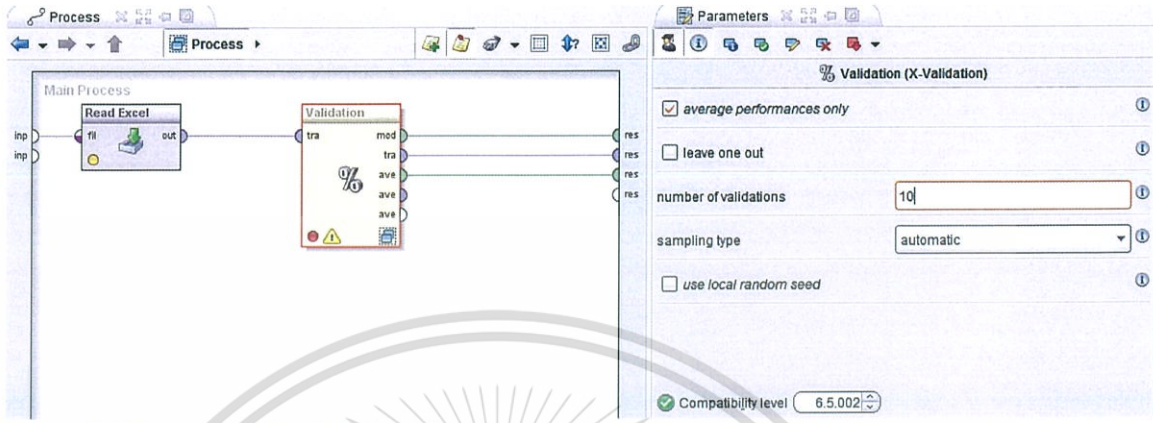


Figure D.15 Number of validation selecting.

17. Click Run. The accuracy are presented as shown in figure C.16.

Table View Plot View

accuracy: 96.50% +/- 2.83% (mikro: 96.50%)

	true non-AID	true AID	class precision
pred. non-AID	515	12	97.72%
pred. AID	9	64	87.67%
class recall	98.28%	84.21%	

Figure D.16 Example of accuracy.

AUTHOR BIOGRAPHY

Name-Surname: Ms. Pakyada Maneewan

Date of Birth: September 22nd, 1983

Present Address: 525/10, Moo 2, Tambol Nongkratum, Amphur Muang, Nakornratchasima, Thailand 30000

Education: 2001-2004: Bachelor degree in Computer Engineering, Khon Kaen University.

Scholarships: 2011-2012 Scholarship for study in Master of Engineering in Data Storage Technology (English program) by NSTDA, KMITL and Seagate Technology (Thailand) Ltd.

Publications: Pakyada M. Siridech B., "ASSEMBLY INDUCED FAILURE CLASSIFICATION AND PREDICTION", ESIT 2016, Engineering Science and Innovative Technology 2016, Thailand, April 21-23, 2016.

Experience:

2005-2007 Seagate Technology (Thailand) Ltd.
- Back End Drive Process engineer

2007-Present Seagate Technology (Thailand) Ltd.
- Asia Engineering Firmware/Test