SENSITIVITY ANALYSIS OF PREDICTIVE DATA ANALYTIC MODELS TO
ATTRIBUTES

by

James Chiou

A Thesis Submitted to the Faculty of
The College of Computer Science and Engineering
In Partial Fulfillment of the Requirements for the Degree of
Master of Science

Florida Atlantic University
Boca Raton, Florida
December 2014
SENSITIVITY ANALYSIS OF PREDICTIVE DATA ANALYTIC MODELS TO
ATTRIBUTES

by

James Chiou

This thesis was prepared under the direction of the candidate’s thesis advisor, Dr. Xingquan “Hill” Zhu, Department of Computer & Electrical Engineering and Computer Science, and has been approved by the members of his supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

SUPERVISORY COMMITTEE:

Xingquan Zhu
Thesis Advisor, Ph.D.

Dingding Wang, Ph.D.

Taghi M. Khoshgoftaar, Ph.D.

Nurgun Erdol, Ph.D.
Chair, Department of Computer & Electrical Engineering and Computer Science

Mohammad Ilyas, Ph.D.
Dean, College of Engineering and Computer Science

Deborah L. Floyd, Ed.D.
Interim Dean, Graduate College

11/18/2014
Date
ACKNOWLEDGMENT

I like to thank Dr. Zhu for giving me advice for my thesis. With his help, support and guidance, I am able to finish my analysis and complete my writing. I also thank Dr. Zhu for reviewing my writing as well.

Besides my advisor, I would like to thank the rest of my thesis committee: Dr. Dingding Wang, Dr. Taghi M. Khoshgoftaar and Dr. Hanqi Zhuang. I also thank my family and friends for their support to finish my thesis.
A classification algorithm represents a rich set of tools, which train a classification model from a given training and test set, to classify previously unseen test instances. Although existing methods have studied classification algorithm performance with respect to feature selection, noise condition, and sample distributions, our existing studies have not addressed an important issue on the classification algorithm performance relating to feature deletion and addition. In this thesis, we carry out sensitive study of classification algorithms by using feature deletion and addition. Three types of classifiers: (1) weak classifiers; (2) generic and strong classifiers; and (3) ensemble classifiers are validated on three types of data (1) feature dimension data, (2) gene expression data and (3) biomedical document data. In the experiments, we continuously add redundant features to the training and test set in order to observe the classification algorithm performance,
and also continuously remove features to find the performance of the underlying classifiers. Our studies draw a number of important findings, which will help data mining and machine learning community under the genuine performance of common classification algorithms on real-world data.
SENSITIVITY ANALYSIS OF PREDICTIVE DATA ANALYTIC MODELS TO ATTRIBUTES

LIST OF TABLES.................................................................................................................. x
LIST OF FIGURES.................................................................................................................. xi

CHAPTER 1 ................................................................................................................................. 1
PREFACE .................................................................................................................................... 1

1.1 Classification Algorithms ............................................................................................... 1
1.2 Feature Selection ............................................................................................................. 3
1.3 Irrelevant Features and Relevant Features ...................................................................... 4
1.4 Sensitivity Study of Classification Algorithms to Relevant and Irrelevant Features ......... 6
1.5 Summary of the Contribution ........................................................................................ 8
1.6 Structure of the Thesis .................................................................................................. 9

CHAPTER 2 ................................................................................................................................. 11
RELATED WORKS ..................................................................................................................... 11

2.1 Classification Algorithms ............................................................................................... 11
2.2 Feature Selection Methods ............................................................................................ 20
2.3 Classification Algorithm Performance with Respect to Different Data ....................... 22

2.3.1 Classification Algorithm Performance with Respect to Noisy Data ....................... 22
2.3.2 Classification Algorithm Performance with Respect to Data Imbalance ................. 24

CHAPTER 3 ................................................................................................................................. 27
EMPIRICAL STUDY SETTINGS ............................................................................................. 27
3.1 Classifiers.......................................................................................................................... 27
  3.1.1 Weak Classifiers.................................................................................................................. 27

  3.1.2 Generic and Strong Classifiers.............................................................................................. 28

  3.1.3 Ensemble Classifiers............................................................................................................. 28

3.2 Dataset Summarization............................................................................................................. 29
  3.2.1 Benchmark Datasets............................................................................................................ 29

3.3 Synthetic Feature Generation................................................................................................... 30
  3.3.1 Feature Deletion .................................................................................................................. 34
  3.3.2 Feature Addition.................................................................................................................... 35

CHAPTER 4 ................................................................................................................................. 37
CLASSIFIER SENSITIVITY STUDY ON FEATURE DIMENSION ............................................. 37
  4.1 Experiment.............................................................................................................................. 37
  4.2 Results..................................................................................................................................... 37

  4.3 Classifier Sensitivity Study on Weak Classifiers..................................................................... 54
  4.3.1 Summary on Weak Classifiers ........................................................................................... 55

  4.4 Classifier Sensitivity Study on Generic and Strong Classifiers.............................................. 57
  4.4.1 Summary on Generic and Strong Classifiers ....................................................................... 57

  4.5 Classifier Sensitivity Study on Ensemble Classifiers............................................................... 59
  4.5.1 Summary on Ensemble Classifiers ...................................................................................... 60

  4.6 Summary on Weak, Generic and Strong and Ensemble Classifiers ........................................ 62

CHAPTER 5 ................................................................................................................................. 66
CLASSIFIER SENSITIVITY STUDY ON BIOMEDICAL HIGH DIMENSIONAL DATA ................. 66
  5.1 Purpose.................................................................................................................................... 66

  5.2 Introduction to Gene Expression Data...................................................................................... 66
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>Introduction to Biomedical Document Data</td>
<td>70</td>
</tr>
<tr>
<td>5.4</td>
<td>Experiment</td>
<td>72</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Performance for Gene Expression Data</td>
<td>73</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Performance for Biomedical Document Data</td>
<td>81</td>
</tr>
<tr>
<td>5.5</td>
<td>Classifier Sensitivity Study on Gene Expression Data</td>
<td>90</td>
</tr>
<tr>
<td>5.5.1</td>
<td>Summary on Gene Expression Data</td>
<td>90</td>
</tr>
<tr>
<td>5.6</td>
<td>Classifier Sensitivity Study on Biomedical Document Data</td>
<td>93</td>
</tr>
<tr>
<td>5.6.1</td>
<td>Summary on Biomedical Document Data</td>
<td>93</td>
</tr>
<tr>
<td>6</td>
<td>CHAPTER 6</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>CONCLUSION</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>REFERENCES</td>
<td>99</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table III-1: Information Relating to Feature Dimension Datasets .................................. 30
Table IV-1: Change in Performance for Weak Classifiers .............................................. 56
Table IV-2: Change in Performance for Generic and Strong Classifiers ......................... 58
Table IV-3: Change in Performance for Ensemble Classifiers ....................................... 60
Table V-1: Gene Expression and Biomedical Document Datasets ................................. 73
Table V-2: Change in Performance for Gene Expression Data ...................................... 91
Table V-3: Change in Performance for Biomedical Document Data .............................. 94
LIST OF FIGURES

Figure II-1: Diagram for Decision Tree ........................................................................ 13
Figure II-2: Diagram for k-NN Classification ................................................................ 14
Figure II-3: Diagram for SVM’s Linear Classification .................................................. 15
Figure II-4: Diagram for Decision Stump .................................................................... 17
Figure III-1: Bar Graph for Pima Dataset .................................................................... 32
Figure III-2: Flowchart for 10 folds Validation ............................................................ 34
Figure III-3: Flowchart for Feature Deletion ............................................................... 35
Figure III-4: Flowchart for Feature Addition ............................................................... 36
Figure IV-1: Graphs for Performance of Classifiers (Accuracy) ..................................... 39
Figure IV-2: Graphs for Performance of Classifiers (AUC) ........................................... 41
Figure IV-3: The Legend for Feature Dimension Data ................................................... 42
Figure IV-4: Results for Weak Classifiers (Accuracy) .................................................. 44
Figure IV-5: Results for Weak Classifiers (AUC) ......................................................... 46
Figure IV-6: Results for Generic and Strong Classifiers (Accuracy) ............................. 48
Figure IV-7: Results for Generic and Strong Classifiers (AUC) .................................... 50
Figure IV-8: Results for Ensemble Classifiers (Accuracy) .......................................... 52
Figure IV-9: Results for Ensemble Classifiers (AUC) .................................................. 54
Figure IV-10: Overall Performance for Classifiers (Feature Dimension) ...................... 64
Figure V-1: Diagram for DNA ................................................................. 68
Figure V-2: Diagram for the Process of Gene Expression ......................... 69
Figure V-3: Classifiers’ Performance with DLBCL-NIH-ALL_240x7399 (Accuracy) . 74
Figure V-4: Classifiers’ Performance with lung-harvard_203x12600 (Accuracy) ........ 75
Figure V-5: Classifiers’ Performance with DLBCL-NIH-ALL_240x7399 (AUC) ....... 76
Figure V-6: Classifiers’ Performance with lung-harvard_203x12600 (AUC) ............. 78
Figure V-7: The Legend for Gene Expression and Biomedical Document Data .......... 78
Figure V-8: Performance Change with DLBCL-NIH-ALL_240x7399 (Accuracy) ...... 79
Figure V-9: Performance Change with lung-harvard_203x12600 (Accuracy)............. 80
Figure V-10: Performance Change with DLBCL-NIH-ALL_240x7399 (AUC) ........... 80
Figure V-11: Performance Change with lung-harvard_203x12600 (AUC) ............... 81
Figure V-12: Classifiers’ Performance with oh5.wc (Accuracy) ............................. 82
Figure V-13: Classifiers’ Performance with oh15.wc (Accuracy) .......................... 84
Figure V-14: Classifiers’ Performance with oh5.wc (AUC) ............................... 85
Figure V-15: Classifiers’ Performance with oh15.wc (AUC) .............................. 86
Figure V-16: Performance Change with oh5.wc (Accuracy) ............................... 87
Figure V-17: Performance Change with oh15.wc (Accuracy) .............................. 88
Figure V-18: Performance Change with oh5.wc (AUC) ................................. 89
Figure V-19: Performance Change with oh15.wc (AUC) ................................. 90
Figure V-20: Bar Graphs for Classifiers’ Overall Performance (Gene Expression) .... 92
Figure V-21: Bar Graphs for Classifiers’ Overall Performance .......................... 95
(Biomedical Document)
CHAPTER 1

PREFACE

1.1 Classification Algorithms

In data mining, classification is a function that categorizes instances to different class values. Each instance represents a row of data which consist of attributes and class label. The attribute defines the properties of the dataset. The function of attribute is to influence the quality of the dataset. A class label consists of categorical value that represents the decision of a particular instance. An example of classification is making decision to buy contact lens [1]. The customer can either purchase soft or hard contact lenses. A typical data relating to contact lenses would consist of features such as age, spectacle prescription, astigmatism, and tear production rate. Each feature consists of categorical values that describe the prescription of contact lens. For example, the feature age would consist of values like young, pre-presbyopic and presbyopic. The class label for contact lenses data is recommended lenses. The categorical values for recommended lenses would be hard and soft. In data mining, contact lenses data can be stored in a file called dataset. Each row of dataset represents an instance that has a class value and various feature values. A good classification can help us to develop a set of rules that determine whether person should prescribe hard or soft contact lenses.
Classification algorithms represent a rich set of tools that trains a classification model with training samples. This training helps to classify previously unseen test instances. The training samples refer to training set that is used for ten folds cross validation. This training set is used in conjunction with test data set during ten folds cross validation. There are many applications for classification algorithms. In today’s world, companies are using classification algorithms in the area of fault diagnosis, loan approval, and bankruptcy protection. The benefits for using classification algorithms are making quick decision and reducing the reliance on experts. For example, in the area of loan approval, some of loan applicants have credit rating status in the border line of bad credit. A good classification algorithm can identify loan applicants as a good candidate for loan even though they have bad credit rating. This utilization of classification algorithm helps lenders to increase their revenues by making quick decision.

The classification model, which is trained by classification algorithms, is called classifier, which utilizes mathematical function to classify unknown instances. Normally, a dataset, which consists of attributes and class label, is the input of classifier.

There are four types of classification in data mining. They are binary, multi-class, multi-labeled, and hierarchical. In binary classification, there are two class values. One is usually referred as positive and the other one is referred as negative. Multi-class classification consists of class labels with more than two class values. For example, the identification of the iris type in a three-class dataset which is very popular in pattern recognition. The class label for iris type has three types. They are Iris Setosa, Iris Versicolor, and Iris Virginica [4]. Multi-labeled is a classification whose input is classified into multiple class variables. For example, the categorization of images such as
beaches and sunsets utilize multi-labeled classification. Hierarchical classification consists of input that is classified into one class label. This class label can be divided into subclasses or grouped into super classes. The applications for hierarchical classification are text classification, bioinformatics, and protein function prediction.

1.2 Feature Selection

Feature selection is a set of methods which select a subset of features from the original dataset. There are three benefits using feature selections. First, these methods will save time while training the dataset. Second, these methods, which is trained from the selected feature, will outperform the models trained from the whole set of the features. Third, feature selections reduce overfitting. Overfitting generally occurs when a model is excessively complex, such as having too many parameters relative to the number of observations. A model which has been overfitted will generally have poor predictive performance, as it can exaggerate minor fluctuations in the data [6].

Because features used to represent training instances, which play vital role to help build good classification models, many methods exist to find good features. The current feature selection methods mainly exploit two approaches. They are individual (feature) evaluation and subset evaluation (Blum and Langley, 1997; Guyon and Elisseeff, 2003). Individual evaluation ranks features according to their importance in differentiating instances of different classes [5]. It analyzes all features in an instance of data and assigns a weighted value to each feature. Therefore, irrelevant features will likely have similar ranking. And also, the feature with highest weighted value will be selected as most relevant attribute. Subset evaluation utilizes ranking feature such as information gain or relief in order to determine highest ranking attribute. The goal of subset
evaluation is searching for optimal subset from a dataset such as gene expression data. In order words, the subset of features with highest ranking will be selected.

Currently, there are four algorithms of feature selection that are currently being used. These are wrapper, filter, embedded and hybrid method. The wrapper method relies on the performance of a specific classifier to evaluate the quality of feature set. In wrapper method, every subset of features is used to train a model. It is a method that offers the best performance. However, the cost of computation is large with increase amount of features. There are three processes of wrapper method. First, wrapper method analyzes the features in the dataset. This process is accomplished by searching space of features. Second, wrapper method determines accuracy for each feature with one learning algorithm. Third, features are added or removed base on the value of the accuracy. The filter model separates feature selection from the classifier and selects subsets of features from the original dataset [5]. Filter method is faster than wrapper method. It uses measurement such as correlation, distance and consistency measurement to find good subset from entire feature. The advantage of filter method is that it utilizes feature ranking which is less computational intensive than wrapper method. Embedded method is a catch-all group of techniques which perform feature selection as part of the model construction process. Embedded method interacts with learning algorithm. For example, Recursive Feature Elimination (RFE) is an embedded method which interacts with support vector machine. The Hybrid method evaluates features by using independent measure to find the best subset and then using a learning algorithm to find the new subset that is better than the last one.

1.3 Irrelevant Features and Relevant Features
Relevant features refer to attributes that have significant roles in the dataset. Their values can help us to determine the expected class value of an instance. Removing a relevant feature can reduce the accuracy of a classifier. There are two types of relevant feature. They are weak relevant feature and strong relevant feature. Weak relevant feature refers to attribute that has less impact to the class value of an instance. Weak relevant feature becomes strong relevant feature if the subset of strong features is removed from the dataset. Strong relevant feature refers to attribute that has great impact to the accuracy of a classifier. Strong relevant feature is referred as predominant attribute. Irrelevant features refer to features that are not relevant features. Usually, irrelevant features have no influence to the value of class label. When a dataset has many irrelevant features, the accuracy remains unchanged when deleting a few numbers of irrelevant features.

There are two algorithms that help to determine the relevancy of each feature. These are Focus and Relief algorithms. Focus algorithm starts out with an empty feature set. It carries out search until it finds the minimum combination of features. The Relief algorithm assigns weight to each feature. The feature that meets user’s expectation will be selected. However, this method will increase computational time. These algorithms can work with other classifiers such as C4.5 and Naïve Bayes.

Although existing methods have studied classification algorithm performance with respect to feature selection, noise condition, and sample distributions, our existing studies have not addressed an important issue on the classification algorithm performance to relevant and irrelevant features. More specifically, the collection of the training data is often an independent process separated from the data mining model training circle. It is
common to find that many training data may contain irrelevant (or random) features, whereas one or multiple strongly relevant features are excluded (missed) during the data collection process. Given a training data set, if a number of irrelevant (or random) features exist in the data, how will the classification algorithm behave and which methods will be mostly affected because of the existence of such irrelevant features? If one or multiple important/relevant features are excluded (or missed) during the data collection process, how will the classification algorithm behave and which methods will be most robust, even if important features were excluded during the data collection process. The answers to the above questions are fundamental because in a real-world data mining environments, the ground-truth of the data characteristics often remains unknown, and understanding the behaviors of our classification algorithms with respect to realistic data environments can provide firsthand experience and knowledge to the whole community.

1.4 Sensitivity Study of Classification Algorithms to Relevant and Irrelevant Features

In this thesis, the investigation of performance of classifiers is developed regarding to sensitivity of relevant and irrelevant features. The purpose is to empirically validate which methods are more robust to noisy data. It is important to know that robustness is a very important element that determines whether the classifier can handle the sensitivity of data. If the classifier is robust, then it is insensitive to data corruptions and suffers less from the impact of noise [18]. Robustness is considered more important than performance results when dealing with noisy data, because it allows one to know a priori the expected behavior of a learning method against noise where characteristics of noise are unknown [18].
There are three types of noise that can exist in noisy dataset. They are uniform noise, variable noise and importance sampled noise. Uniform noise has same noise level on all attribute. Variable noise has different noise levels on attribute. Importance sampled noise has noise level selected by attribute importance.

In this thesis, the attribute, with the impact of different types of noise levels, is called asymmetrical attribute. Asymmetric attribute is asymmetric means noise levels in training data that are significantly different than noise levels when a classifier is deployed in the field. It is common to replicate field noise by creating redundant attribute [8]. Attribute importance is the predominant attribute with largest noise level. Reverse attribute importance is predominant attribute with smallest noise. Attribute importance is determined by Ranker search method, which implements information gain as evaluation function.

In this thesis, two methods are developed to investigate the performance of the classifiers with various datasets. The performance is measured in accuracy and AUC. These variables are referred as performance metrics. The first method will add a redundant feature, which is asymmetrical attribute, continuously until the dataset has five (feature dimension data) or fifty (gene expression or biomedical document data) features. In order to add redundant feature to the dataset, the redundant attribute must be same as any attribute in the dataset except the class label. The purpose of this method is to investigate whether the performance of the classifier is robust to the noise which it will be replicated by adding redundant feature. Redundant feature is labeled as asymmetrical attribute. The problem of adding redundant feature in the dataset will become relevant when all five redundant attributes are irrelevant attributes. This result will lead to bias
when investigating the performance of the classifier. Therefore, the use of ten time and ten folds is implemented to reduce the bias. And also, the redundant feature is assigned a value that is similar to the value of attribute in the dataset. The value for the attribute can be nominal or numeric. The value for redundant attribute will be randomly generated which it will be based on the min or max value of the attribute in the dataset.

The second method will delete predominant feature continuously until the original dataset has lost five (feature dimension data) or has lost fifty (gene expression or biomedical document data) attributes or has only one left. The class label will be excluded in this method. The purpose of deleting the attribute is to investigate whether the classifier is sensitive to elimination of predominant feature. The predominant feature can be functioned as noise that has great impact to the quality of the dataset. This is important because that the new technique can be developed to eliminate noise. This new technique can utilize the classifier that is robust to noise. The problem of deleting predominant attribute is determining which attribute in the dataset is predominant. Usually, most algorithms that determine the predominant attribute have shown inconsistency when it used at various datasets.

1.5 Summary of the Contribution

In this thesis, three groups of classifiers are being observed. The datasets will be input to the program that evaluates performance of the classifier. There will be 34 datasets and couple datasets with high dimensionality that will be used for analysis. The output that shows performance of classifiers is split into three groups. The first group represents generic and strong classifiers. They are Naive Bayes, decision tree, k-NN and SVM. The second group represents weak classifiers. The weak classifiers are
HyperPipes, Reptree and decision stump. The last group represents ensemble classifiers. The ensemble classifiers are Bagging, AdaBoost and Random Forest. In this thesis, there will be two methods that evaluate the importance of feature. They are information gain and ReliefF. Since these two methods show the importance of attributes differently, therefore, separate outputs for ReliefF and information gain are created. These separate outputs give better analysis and comparison while observing the performance of classifiers.

For each dataset that will be analyzed, the program will first determine number of instances and attributes in the dataset. The program will determine the accuracy and AUC for weak, generic and strong, and ensemble classifiers. Once all of performance metrics are determined, the program will generate graphs relating to performance of classifiers.

In this thesis, there are three observations while adding or deleting a feature. These observations are:

- The classifiers are more sensitive to feature deletion than feature addition.
- In gene expression data, the performance of classifiers is the same while deleting or adding a feature.
- In biomedical document data, the performance of classifiers decreases as the number of feature deletion increases.

1.6 Structure of the Thesis

The remaining chapters will discuss the performance of the classifiers with various scenarios. In chapter 2, the thesis will discuss the related works about the
classifiers. In chapter 3, the thesis will discuss datasets that will be used in empirical studies. In chapter 4, the thesis will discuss the sensitivity of classifier relating to feature dimension. In chapter 5, the thesis will discuss the sensitivity of classifier relating to high dimensional data. And finally, chapter 6 concludes the thesis.
CHAPTER 2

RELATED WORKS

2.1 Classification Algorithms

Naive Bayes is a simple probabilistic classifier based on Bayes’ Theorem [22]. It provides a practical learning algorithm by making multiple predictions. It predicts multiple hypotheses which are weighted by probabilities. Bayes Theorem, which consists of strong independent assumptions, is an equation with manipulation of conditional probabilities and it assumes that conditions are independent to each other [23]. The equation below is the basis of Bayes’ Theorem.

\[
p(C|F_1, \ldots, F_n) = \frac{1}{Z} p(C) \prod_{i=1}^{n} p(F_i|C)
\]

Where the evidence \( Z = p(F_1, \ldots, F_n) \) is a scaling factor dependent only on \( F_1, \ldots, F_n \).

The application of utilizing Naïve Bayes is text documents. Naïve Bayes is best utilized with large training set. And also, the attributes are conditionally independent in a datafile.
Decision tree implements information in tree like structure in order to make sound decision [21]. The advantages of decision trees are easy to understand, easy to generate rules relating to the dataset, and no priori assumption about the data. Disadvantage of decision tree is overfitting. One of the algorithms that builds a decision tree is C4.5 [24]. C4.5 generates a decision tree by splitting the nodes from the top and split other attributes recursively. The algorithms stop when there is no more data to be split up. C4.5 builds a decision tree that correctly classifies the largest number of examples. C4.5 is a decision tree where instance is filtered through the top of the node. Figure II-1 below is the decision tree that is created for the weather problem. This decision tree makes prediction whether the person should play or not play outside [1]. There are four attributes that have great influence about the decision to play. These are outlook, humidity, temperature and wind. The decision tree selects outlook, which has highest information gain ratio, to be the top of the tree. Then, the tree recursively calculates information gain ratio for the rest of the attributes in order to determine next predominant attribute.
Figure II-1: An example of a decision tree. A decision tree consists of a number of internal nodes (i.e., rectangle boxes) and decision nodes or leaf nodes (i.e., the circles). Each rectangle box represents an attribute or a feature, and the arrow (including the value associated to each arrow) indicates the possible values of each attributes. For a given training data set, one can use the attributes to split the dataset into subsets for decisions. In order to make a decision, one can traverse the tree from the root to the leaf nodes, and make corresponding decisions (i.e., yes or no).

K-NN, which stands for K-nearest neighbor’s algorithm, is an algorithm that uses for classification and regression [25]. K-NN is a non-parametric method which the representation of data can be varied in size [26]. K-NN utilizes distance function to determine class values. The input consists of the k closest training examples in the feature space. The classification of an instance is based on k number of nearest objects in space. For example, in order determine the class value for an instance when k = 1, the classification will find one object that is closest to the instance, which it is represented as a point in space. If the object is labeled as HIGH, then the class value of the instance is
assigned as HIGH. Figure II-2 below shows the classification of k-NN when the value of k is three. The instance is represented as red square. There are two types of objects in feature space. They are black circles and light green triangles. Since there two green triangles and one black circle are closest to red square. Therefore, the instance should be classified as HIGH.

**Figure II-2** – An example of k-NN classification. Black circles and green triangles are training examples. The class value for black circle is low and the class value for green triangles is high. The red square represents an instance which will be classified as low or high. The input value for k-NN classification is k which represents the number of closest examples. When k = 3, the algorithm will find three training examples (two green triangles and one black circle that are in dotted circle), which are closest to red square. Therefore, red square is classified as HIGH.

SMO, which means sequential minimal optimization, is an SVM algorithm for solving quadratic programming. SVM, which is support vector machine, is a supervised
learning model for classification. SVM performs linear classification and non-linear classification. In linear classification, SVM model creates a hyperplane that separate two training examples into two classes. Then, the Maximum Margin is established in order to prevent misclassification in training examples. In non-linear classification, training examples consist of two mixed classes that cannot be separated by hyperplane. Therefore, a kernel trick is performed to transform the training examples in original feature space into 3D feature space. Then, SVM model can apply hyperplane to separate training examples. In figure II-3 below, the graph demonstrates that SVM model performs linear classification. Please note a maximum margin is established in order to achieve maximum performance in linear classification.

Figure II-3 – An example of SVM’s linear classification. The graph demonstrates SVM’s linear classification which consists of X and Y axis with hyperplane (line B) and two maximum margins (line A & C). Hyperplane, which exists in an n-dimensional space, is a flat subset with dimension n-1 [56]. Hyperplane divides the space into one-
half. The maximum margins (line A & C) separate two training examples. The training examples are black circles and white circles.

In a group that represents weak classifiers, there are HyperPipes, Reptree and decision stump. HyperPipes is a classification algorithm that can operate very fast and process datasets with a large quantity of feature values [20]. HyperPipes only handle values that are discretized. Discretization refers to attributes with numeric values are converted to categorical values. In general, discretization of numerical attributes is quite important for data-mining algorithms. For a dataset with numeric values, a range of values needs to be established in order for the attribute to be discretized. HyperPipes works best in a dataset that has few instances with minimum amount of features with values. In order to test the hypothesis of HyperPipes, the method, which is Randomized Decimation HyperPipes (RDH), is used. RDH has the capability of adjusting the training sets in order to simulate the sparseness of the dataset. In other words, RDH allows user to set the level of sparseness of a dataset. For example, RDH has a parameter that specifies the amount of deletion of instances of entire dataset. The parameter is called minimal deletion parameter. The range of minimal deletion parameter is between 0 and 1. Therefore, if minimal deletion parameter is 0.1, ninety percent of the training data will be eliminated. The problem of using single parameter in RDH is that the algorithm cannot handle the dataset with high complexity such as genomic dataset. To alleviate the problem, a second parameter is used. This parameter specifies the number of features that will be selected by FSS (feature subset selection). This selection will help to select instances that have optimal subset and eliminate the rest of instances that do not have relevant features. Zainab Abu Deeb, Thomas Devine and Zongyu Geng perform RDH experiments that are done in 10 times and 10 folds validation. The results show that
RDH performs better with increased sparseness of data. The use of RDH will increase the intelligence of HyperPipes while maintaining the level of simplicity. Feature subset selector improves the learning scheme such as RDH (randomized decimation HyperPipes). The performance of classifier is better. However, it has no effect on HyperPipes.

Reptree is a fast decision tree learner. It builds a decision or regression tree using information gain or variance. In order words, the method calculates which attribute has highest information gain attribute. Therefore, the first level of decision tree will have the highest information gain.

Decision Stump is a machine learning model consisting of one level decision tree [28]. It is a decision tree with a main node that is connected with two nodes. It is simple algorithm that analyzes one important attribute. In other words, the decision stump makes decision based on one feature. Therefore, decision stump can be called as one rule. In figure II-4, the graph shows an example of decision stump.

![Figure II-4](https://via.placeholder.com/150)

**Figure II-4** – An example of a decision stump. Decision stump is a type of a decision tree with top node (blue square) and two bottom nodes (green squares). The top node
(blue square) evaluates the main input. The bottom nodes (green squares) are the outputs based on the input of petal width, which is evaluated at the top node.

The graph shows petal width is the input of decision stump. If the input is no, the class value is Iris Versicolor. If the input is yes, the class value is Iris Virginica.

In a group that represents ensemble classifiers, there are Bagging, AdaBoost and Random Forest. Ensemble classifiers refer to supervised learning algorithm that utilizes multiple learning algorithms in order to achieve better performance in accuracy. Bagging is called bootstrap aggregating. Bootstrap aggregating is a strategy that utilizes random dataset that is deriving from the original dataset. It randomly samples with replacement and combines with majority vote. The use of random dataset helps to improve the performance of the classifier. Therefore, bagging is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression.

AdaBoost is a machine learning meta-algorithm formulated by Yoav Freund and Robert Schapire who won the prestigious “Golden Prize” in 2003 for their work [30]. AdaBoost, which stands for adaptive boosting, is a meta-algorithm that uses other learner to improve performance. Usually, AdaBoost uses combination of weak learners such as decision stump in order to improve the performance of the AdaBoost. According to general theory of Vapnik and Chervonenkis, there are three criteria in order for AdaBoost to be effective [43]. First, the classifier should be trained enough training examples. Second, the classifier should have less training error. Third, the classifier should be easy to use. The theory itself is the starting point of understanding AdaptBoost. The theory has stated that weak classifier, which will be used as combination of classifiers for
AdaptBoost, should have better accuracy than random guesses. The weak classifier should satisfy the following condition: \( \varepsilon_t \leq \frac{1}{2} - \gamma \); where \( \gamma > 0 \). The condition, which is called weak learning condition, should have error that is below \( \frac{1}{2} \).

The following steps are the algorithms of AdaptBoost [43]. First, given a number of instances from a dataset, a class label is assigned as 1 or -1. Second, create ten partitions of smaller subset from the original dataset. These partitions will train the weak classifier such as decision stump. Third, assign weak hypothesis, which can be 1 or -1 to weak classifier. Fourth, determine weighted sum error which is determined by the equation below.

\[
\varepsilon_t = \sum_i w_{i,t} e^{-y_i h(x_i)}
\]

Fifth, use weighted sum error to determine the weighted value. The equation is listed below.

\[
\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \varepsilon_t}{\varepsilon_t} \right)
\]

And the last, the final hypothesis for AdaBoost is calculated by the equation below.

\[
H(x) = \text{sign} \left( \sum_{t=1}^T \alpha_t h(x) \right)
\]

Random Forest is an ensemble classifier consists of collection of tree based classifiers. Random Forest utilizes the technique of bagging, which creates training and test set in order to train the tree based classifiers. Random Forest usually accepts two parameters. Those two parameters are number of features in a node and number of trees
in the forest [19]. The first parameter specifies the number of features that define random selection feature. The random selection feature, which feature is randomly selected for candidate splitting, prevents correlation between features. The decreasing correlation prevents overfitting. The second parameter specifies the number of trees that will be used as a classifier. The tree will grow maximum extent without pruning. Each tree will have classification that gives a vote to the performance of the classifier. Therefore, the output of the Random Forest is decided by majority vote. Random Forest is robust to noise and it offers the best performance because it does not overfit. The result, surprisingly, shows that Random Forest has better performance than discriminant analysis, support vector machines and neural networks. The application of Random Forest can be applied to a set of multi-temporal SAR images using decision tree as a base classifier [16].

2.2 Feature Selection Methods

Originally, ReliefF is derived from Relief, which is a filter algorithm. Relief is one of the most successful preprocessing algorithms [15]. It is context sensitive and can correctly estimate the quality of attributes in problems. Relief can select splits at intermediate nodes in decision tree learning and also inductive logic programming [15]. As the classification algorithms become more complex, the extension of Relief algorithm is developed. Igor Kononenko proposed an extension of Relief algorithm which it is called ReliefF. ReliefF is an instance-based filtering method which the determination of importance of attributes is depended on the number of instances and values of class label in the dataset [15]. Currently, ReliefF can estimate the rank of attributes that can be value of numeric, nominal and missing values. ReliefF algorithm can also handle multi-
class classification problem. ReliefF evaluates the ranking of an attribute by repeatedly sampling an instance and considering the value of the given attribute.

The following steps are the algorithm of ReliefF. First, the weight values for all the attributes from the dataset are set to zero. Second, randomly select an instance from a subset of instances and designate it as R. Third, find the nearest instance that has same value of class label R and name it as Nearest Hit H. Fourth, find the instance that has different value of class label than R. Then, label the instance as the Nearest Miss M. Fifth, use quality estimator, which accepts the input values of R, H and M, to update the all the weight values for the attributes from the dataset. The algorithm stops until certain numbers of randomly selected instances have been reached. The algorithm for ReliefF is similar to Relief except it handles multiple class labels. And also, the input values for quality estimator, which are H and M, are determined by the average of all contributions of all nearest hits and misses.

Feature subset selection is a feature selection technique that reduces the size of features. There are two types of feature subset selector. They are ranking feature selectors and subset selector. The ranking feature selectors rank the features and select the n top features [20]. Subset Selector finds the optimal subset. The applications for feature selection are e-mail filtering and drug discovery problems. Feature subset selection extracts subset of features by removing redundant or irrelevant features from the dataset. The advantage of feature subset selection is no relevant feature is lost.

Howley et al (2006) have investigated the effect of PCA on dataset with high dimensionality. PCA is a dimensionality reduction technique that generates a linear
combination of the original attribute set [17]. In other words, PCA creates an optimal subset from original dataset. PCA, which stands for principal component analysis, captures linear dependencies among attributes of dataset. The results show that PCA combines with classifier have better result in accuracy with a high dimensionality dataset.

Guvenir has developed VFI5 (Voting Feature Intervals) algorithm, which is a feature selection that utilizes voting mechanism [12]. The applications for VFI5 are diagnosis of Erythematous-Squamous diseases and arrhythmia analysis of ECG signals. VFI5 can achieve high accuracy and it is not sensitive to removal of irrelevant features. The reason is that the voting mechanism in VFI5 prevents the side effect of irrelevant features. The voting mechanism gives the same weight value for all irrelevant features in the domain.

Section 2.3: Classification Algorithm Performance with Respect to Different Data Characteristics.

This section will discuss the performance of classification algorithm with respect to noisy data. Then, the section will discuss the performance of classification algorithm with respect to data imbalance.

2.3.1: Classification Algorithm Performance with Respect to Noisy Data.

According to Mannino, there are currently five studies relating to noise level in training data. First study is implementing non representative noise level in training data. This implementation will measure how noise affects performance. Second study is improving the quality of field data. This process is called cleaning field data. Third study is the measurement of tolerance of asymmetrical noise by popular classification
algorithms. Forth study is relating to observation of interaction of training dataset size and asymmetrical attribute noise. The result shows that there is more sensitivity to asymmetrical attribute noise on small training set than medium and large training set. Fifth study is relating to impact of noise variation. The result shows that the impact of asymmetrical attribute noise is directed toward predominant attribute.

Mannino’s investigation involves the implementation of explicit noise handling. Explicit noise handling utilizes the addition of noise in a controlled manner. The result shows that explicit noise handling has the same expected performance but lower variance than traditional techniques using noisy training data [8].

There are number of experiments relating to C4.5 experiment with MCS, which stands for Multiple Classifier System. Study shows that boosting is more accurate than bagging and randomization on low noise level [18]. Bagging works better when noise level increases. Two factors are considered in the experiment: the performance and robustness. Saez, Galar, Luengo and Herrerra have conducted experiment with a post-pruning, which discards unreliable parts from growing [18]. Since the process is based on statistical analysis, the data can be unreliable because they depend on training data. Therefore, when the noise level is high, C4.5 may obtain poor performance. Saez, Galar, Luengo & Herrerra have also conducted experiment relating to MCSs. They made eight observations relating to MCSs [18]. First, the behavior of MCSs with heterogeneous classifier is relied on individual classifier that is part of MCSs. Second, only classifier that is robust to noise should be selected. Third, noise with different levels affects the performance of classifier differently. In general, uniform class noise affects the performance of classifier more than uniform attribute noise. Fourth, regarding to class
noise, MCSs perform better with uniform class noise. Fifth, regarding to attribute noise, MCSs perform better with less disruptive attribute noise scheme. Sixth, attribute or class noise affects MCSs differently. Seventh, the performance of MCSs is the average of all classifiers that implemented in MCSs. Eighth, a majority vote scheme is a popular method for MCSs. They concluded that MCSs, which have combination of classifiers that complement each other, should be noise tolerant. The overall performance for MCSs should have better generalized capabilities [18].

Therefore, based on Saez, Galaar, Luengo & Herrerra’s experiments, the results show that MCSs do not improve their performance of single classification algorithms when dealing with noisy data. In other words, the performance of MCSs will be the average of all the classifiers that are implemented in MCSs. The only factors that improve the performance of MCSs are noise type and noise level. However, it will not exceed the performance of single classifier. They also demonstrated that the performance of MCSs with heterogeneous classifiers is depended on performance of each classifier. They also state that MCSs perform better with class noise than attribute noise. On the final note, they pointed out that MCSs’ robustness is the average robustness of all classifiers that are implemented in the MCSs.

2.3.2: Classification Algorithm Performance with Respect to Data Imbalance.

Zhu, Bao and Qiu have conducted experiment that utilizes L3B approach. L3B is a bagging approach that combines weak classifiers such as decision stump, OneR and SuperPipes. The purpose of this experiment is to see the performance of L3B with the employment of 30 datasets and 2 high-dimensional gene expression data. Since the features of high dimensional dataset grow exponentially, it is important for this
experiment to determine if L3B can handle the complexity of data. The experiment also compares the performance of L3B with SB, which is simple bagging method. Bagging represents a classifier ensemble framework which relies on a set of trained base classifiers to predict the class label of a test instance [8].

Rakkrit Duangsoithong and Terry Windeatt utilized ten times and ten folds validation in order to validate the accuracy of classifiers. These classifiers are Naïve Bayes, multilayer perception, support vector machine and decision tree. The number of classifiers in bagging is varied from 10, 20, 50, 100, 250 and 500 classifiers [11]. Results show that removing irrelevant features improves accuracy and reduces computational time. Results also show that removing redundant feature not only reduces computational time but also reduces accuracy of the ensemble.

The performance of ReliefF degraded significantly when the dataset is imbalanced. Therefore, the dataset needs to be examined for possibility of data imbalance. Currently, there are three algorithms that handle data imbalance. They are higher weight ReliefF, ReliefF with differential minority repeat, and ReliefF with balanced minority repeat. Those three algorithms are derived from ReliefF algorithm. The higher weight ReliefF can handle data imbalance from the dataset. The algorithm assigns weight value to each attribute. Therefore, it can handle minority class more effectively than the original ReliefF algorithm [15]. ReliefF with differential minority repeat is the method that modifies the dataset so that it becomes balanced so that the minority class examples are not neglected while calculating the weight for the attributes [15]. ReliefF with balanced minority repeat splits entire instance space into minority and
majority instance spaces. This splitting will create a final dataset that consists of number of balanced datasets of equal number of minority and majority examples.
CHAPTER 3

EMPIRICAL STUDY SETTINGS

3.1: Classifiers

This section will discuss three groups of classifiers that will be used for empirical study. They are weak, generic and strong and ensemble classifiers. First section will discuss the setting of weak classifiers. Second section will discuss the setting of generic and strong classifiers. Third section will discuss the setting of ensemble classifiers.

3.1.1: Weak Classifiers

Weak classifiers are designed to have better performance than random guesses. There are couple reasons to use weak classifiers. First, weak classifiers are easy to use because they are fast decision learner. For example, decision stump requires one relevant feature in order to classify the instances. Second, weak classifiers don’t require a lot of computational cost. Because weak classifiers are fast decision learner, therefore, the training of weak classifiers doesn’t require a lot of time. Third, weak classifiers can be used as combination of classifiers which offer optimal performance in data mining.

In this thesis, the weak classifiers consist of HyperPipes, Reptree and decision stump. Three graphs will compare weak classifiers’ performance in accuracy and AUC.
In the graph, HP represents the line for HyperPipes. REP represents the line Reptree. DS represents the line decision stump.

### 3.1.2: Generic and Strong Classifiers

The goal of generic and strong classifiers is to have better performance than weak classifiers. For example, the implementation of SVM (support vector machine) is to achieve maximum accuracy by separating two classes of data with the use of hyperplane.

In this thesis, generic and strong classifiers are Naïve Bayes, decision tree, k-NN and SVM. Three graphs will compare each classifier’s performance in accuracy and AUC. In the graph, Naïve Bayes stands for NB. Decision tree represents the line for J48. k-NN represents the line for IBK. SMO represents the line for SVM.

### 3.1.3: Ensemble Classifiers

Ensemble classifiers utilize a combination of classifiers in order to make decision. Ensemble classifiers have a mechanism called majority vote that can make a decision based on the performance of each classifier. Typically, the performance of ensemble classifier is better than a single classifier. The combination of classifiers usually consists of weak classifier. The reason of selecting weak classifiers is that weak classifiers offer better performance.

In ensemble classifiers, there are Bagging, AdaBoost and Random Forest. The performance of classifiers will be measured in accuracy and AUC. In the graph, BAG stands for the line for Bagging. ADA represents the line for AdaBoost. RF represents the line for Random Forest.
3.2: Dataset Summarization

The following subsection discusses datasets that will be used for benchmarking.

3.2.1: Benchmark datasets

The purpose of benchmarking with 34 datasets is to get consistent values in performance metrics and standard deviation for each classifier. Table III-1 shows the table that consists of listing of dataset.

<table>
<thead>
<tr>
<th>File name</th>
<th># of Instances</th>
<th># of Attribute</th>
<th>Missing Value</th>
<th>Area</th>
<th>Number of Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>annealing</td>
<td>798</td>
<td>38</td>
<td>Yes</td>
<td>Physical</td>
<td>6</td>
</tr>
<tr>
<td>arrhythmia</td>
<td>452</td>
<td>279</td>
<td>Yes</td>
<td>Life</td>
<td>16</td>
</tr>
<tr>
<td>audiology</td>
<td>226</td>
<td>69</td>
<td>Yes</td>
<td>Life</td>
<td>24</td>
</tr>
<tr>
<td>autos</td>
<td>205</td>
<td>26</td>
<td>Yes</td>
<td>N/A</td>
<td>6</td>
</tr>
<tr>
<td>balance-scale</td>
<td>625</td>
<td>4</td>
<td>No</td>
<td>Social</td>
<td>3</td>
</tr>
<tr>
<td>breast-cancer</td>
<td>286</td>
<td>9</td>
<td>Yes</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>breast-w</td>
<td>699</td>
<td>10</td>
<td>Yes</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>car</td>
<td>1728</td>
<td>6</td>
<td>No</td>
<td>N/A</td>
<td>4</td>
</tr>
<tr>
<td>cmc</td>
<td>1473</td>
<td>9</td>
<td>No</td>
<td>Life</td>
<td>3</td>
</tr>
<tr>
<td>colic</td>
<td>368</td>
<td>27</td>
<td>Yes</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>cylinder-bands</td>
<td>512</td>
<td>40</td>
<td>Yes</td>
<td>Physical</td>
<td>2</td>
</tr>
<tr>
<td>dermatology</td>
<td>366</td>
<td>34</td>
<td>Yes</td>
<td>Life</td>
<td>6</td>
</tr>
<tr>
<td>diabetes</td>
<td>768</td>
<td>8</td>
<td>None</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>ecoli</td>
<td>336</td>
<td>7</td>
<td>None</td>
<td>Life</td>
<td>8</td>
</tr>
<tr>
<td>flags</td>
<td>194</td>
<td>29</td>
<td>None</td>
<td>Life</td>
<td>8</td>
</tr>
<tr>
<td>glass</td>
<td>214</td>
<td>9</td>
<td>None</td>
<td>Physical</td>
<td>7</td>
</tr>
<tr>
<td>hepatitis</td>
<td>155</td>
<td>19</td>
<td>Yes</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>hypothyroid</td>
<td>3772</td>
<td>29</td>
<td>Yes</td>
<td>Life</td>
<td>4</td>
</tr>
<tr>
<td>iris</td>
<td>150</td>
<td>4</td>
<td>No</td>
<td>Life</td>
<td>3</td>
</tr>
<tr>
<td>labor</td>
<td>57</td>
<td>16</td>
<td>No</td>
<td>Social</td>
<td>2</td>
</tr>
<tr>
<td>liver-disorders</td>
<td>345</td>
<td>6</td>
<td>No</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>lymph</td>
<td>148</td>
<td>18</td>
<td>No</td>
<td>Life</td>
<td>4</td>
</tr>
<tr>
<td>mushroom</td>
<td>8124</td>
<td>22</td>
<td>No</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>pima</td>
<td>767</td>
<td>8</td>
<td>No</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>segment</td>
<td>2310</td>
<td>19</td>
<td>No</td>
<td>N/A</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>-----</td>
<td>------</td>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>sick</td>
<td>3772</td>
<td>29</td>
<td>Yes</td>
<td>Life</td>
<td>2</td>
</tr>
<tr>
<td>sonar</td>
<td>208</td>
<td>60</td>
<td>No</td>
<td>Physical</td>
<td>2</td>
</tr>
<tr>
<td>soybean</td>
<td>683</td>
<td>35</td>
<td>Yes</td>
<td>Life</td>
<td>19</td>
</tr>
<tr>
<td>splice</td>
<td>3190</td>
<td>61</td>
<td>No</td>
<td>Life</td>
<td>3</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>958</td>
<td>9</td>
<td>No</td>
<td>Game</td>
<td>2</td>
</tr>
<tr>
<td>vehicle</td>
<td>946</td>
<td>18</td>
<td>N/A</td>
<td>N/A</td>
<td>4</td>
</tr>
<tr>
<td>vote</td>
<td>435</td>
<td>16</td>
<td>Social</td>
<td>Yes</td>
<td>2</td>
</tr>
<tr>
<td>vowel</td>
<td>528</td>
<td>10</td>
<td>N/A</td>
<td>N/A</td>
<td>11</td>
</tr>
<tr>
<td>zoo</td>
<td>101</td>
<td>17</td>
<td>No</td>
<td>Life</td>
<td>7</td>
</tr>
</tbody>
</table>

Table III-1 – A table with listing of feature dimension datasets. Each row has a detail information about the dataset. The columns for each row are number of instances, number of attributes, missing values, area and number of class.

3.3: Synthetic Feature Generation

In order to discuss feature deletion and addition, it is important to know about ten folds cross validation. Ten folds cross validation begins by partitioning the original dataset into ten small datasets with equal number of instances. The number of instances per partition is called partition length. The program will shuffle the order of instances and redistribute each instance to each partition. The redistribution stops when all partition has its partition length reaches maximum. Any leftover instances that have not yet redistributed to the partition will be discarded. After the original dataset is partitioned into ten smaller subsets, the program will randomly select one subset as test set and the rest of subsets as training sets. This process will count as one fold validation and the process stops when the validation has evaluated ten times. The key advantage of this method is that all partitions are guaranteed to be used for both training and test set. Each validation will evaluate all classifier’s performance in accuracy and AUC. The accuracy is a measurement that determines how many instances in a dataset are correctly classified. Accuracy is sometime called confidence. The factors of determining accuracy of classical algorithms are depended on adequate training data, the input data characteristics,
and the feature selections. In order to understand AUC, it is important to understand F-score. F-score is harmonic mean of precision and recall [41]. Precision is defined as the number of correctly classified positive examples divided by the number of examples labeled by the system as positive. Recall is calculated by the number of correctly classified positive examples divided by the number of positive examples in the data [7]. The calculation of precision and recall is influenced by four factors. These four factors are true positive, true negative, false positive and false negative. True positive is the number of correctly recognized class examples. True negative is the number of correctly recognized examples that do not belong to the class. False positive is the number of incorrectly instances assigned to the class. False negative is the number of instances that are not recognized as class examples. Below equation is the calculation for F-score.

\[ F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \]

F-score is often used for measuring performance of search classification, document classification and query classification. AUC is the area under the ROC curve. ROC, which stands for Receiver Operating Characteristic, is a graphical plot which illustrates the performance of a binary classifier system [42]. It can also be viewed as a linear transformation of Youden Index. The purpose of using AUC is to measure TPR (true positive rate) and FPR (false positive rate) for each classifier. This helps experimenter to understand how many instances are correctly classified and how many instances should be incorrectly classified. Figure III-2 below shows the flowchart of ten folds validation with feature deletion and addition.
Before the program proceeds to perform feature deletion, the method ReliefF and information gain will determine the predominant feature. These methods will rank the attribute from low to high. Although there is differences how ReliefF and information gain rank the importance of attributes, nevertheless, their methods will be used in this thesis which will give broad perspective about the performance of classifier. Below bar graph (figure III-1) shows the ranking of attributes for Pima dataset.

Figure III-1 – The bar graph above shows ranking of all attributes in Pima.arff. The graph utilized information gain method to determine the importance of attributes. The values on y axis represent ranking value. The names on the x axis represents attribute name. The attribute att2 has the highest ranking and attribute att3 has the lowest ranking.
Begin N-fold Validation

Original Dataset

10 partitions

\[ S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4 \rightarrow S_5 \rightarrow S_6 \rightarrow S_7 \rightarrow S_8 \rightarrow S_9 \rightarrow S_{10} \]

\( S_{10} \)

Train Classifiers

Get Accuracy

Training Set

Add Redundant Attribute

\( T' \)

Train Classifiers

Get Accuracy

Compare Result

Feature Addition Complete?

No

Yes

Train''

Delete Predominant Attribute

\( T'' \)

Train Classifiers

Get Accuracy

Feature Deletion Complete?

No

Yes

Done?

Exit

33
**Figure III-2** – The flow chart above shows the process of ten folds validation with feature deletion and addition. The original dataset is split into 10 partitions (smaller dataset). Then, one of the partitions is assigned as test set. The rest of partitions are combined as training set. The processes of feature deletion and addition are applied to training and test set. The training and test set will be evaluated with a list of classifiers. The accuracy of each classifier is determined after each evaluation.

### 3.3.1: Feature Deletion

The process of feature deletion lowers prediction about the dataset. Usually, the most important attribute in the dataset, which is predominant attribute, is selected for feature deletion. In this thesis, the method ReliefF or information gain will be used to determine the predominant attribute.

The goal of feature deletion is to study the sensitivity of classifiers. The process of feature deletion begins by deleting the predominant attribute from training and test set after the predominant attribute is determined. After the attribute is deleted along with its values, the performance of each classifier is evaluated. Then, the method will evaluate the next highest importance of attribute. After the next highest relevant attribute is determined, the process deletes the attribute from training and test data set. The process stops when it has deleted enough attributes from training and test set. If the training and test set have only one attribute left, the process will stop deleting the feature and begin to evaluate the performance of classifiers. Figure III-3 below shows the flow chart of feature deletion.
Figure III-3 – The flow charts shows the process of feature deletion with method information gain and Relieff. The diamond represents the decision to use either information gain or Relieff. The output of this process will delete the predominant attribute from training and test set.

3.3.2: Feature Addition

The feature addition is process of adding a redundant feature in the training and test set. The goal of feature addition is to observe the sensitivity of classifiers while adding redundant features.

The method will randomly select one attribute from original dataset as an inserted attribute. The attribute can be nominal or numerical. The program will generate an
attribute name for the new inserted attribute. Then, the process will assign a value, which is randomly generated, to this new attribute. This value is based on the range of original attribute’s value. For each random attribute that is inserted in training or test set, the performance of classifiers is evaluated. The method will continue until it has inserted enough random attributes to test or training set. Figure III-4 shows the flow chart of feature addition.

Figure III-4 – The flow chart shows the process of feature addition. This process randomly selects an attribute from the dataset. The output of process will add a redundant attribute to the dataset.
CHAPTER 4

CLASSIFIER SENSITIVITY STUDY ON FEATURE DIMENSION

4.1: Experiment

In this chapter, we will observe the sensitivity of weak, generic and strong, and ensemble classifiers with feature deletion and addition. The purpose of this experiment is to determine if there are classifiers that are robust to feature deletion and addition. The performance metrics that are being used in this experiment will be accuracy and AUC. In order to understand the sensitivity of classifiers, a 10 runs 10 folds validation is used in order to determine the performance of each classifier with 34 datasets. During each fold validation, there will be five feature deletion and addition that are being applied to the training and test set. After completion of 10 runs 10 folds validation, there will be graphs that analyze the performance of classifiers in weak, generic and strong, and ensemble classifiers.

4.2: Results

The graphs (Figure IV-1 to IV-2), which are created by Jfreechart, compare classifiers’ performance with dataset (Soybean). For feature deletion, each graph utilizes information gain or ReliefF in order to determine predominant attribute. Therefore, there
will be two graphs for each performance metric. In each graph, the values of x axis indicate the number of feature deletion or addition. The positive number on x axis represents the number of attributes that is added to the dataset. For example, in a particular x axis where x=2, it means that two redundant attributes (x=2) are added to training and test set. Redundant attribute is a dummy attribute that is similar to one of the attributes from the dataset. In order to create a dummy attribute, we randomly select the attribute from the dataset and create a copy of selected attribute. This copy will be designated as redundant attribute. The value of redundant attribute is assigned by a random number, which is based on range of values from the selected attribute. The negative number on x axis represents the number of attributes that is deleted from the dataset. For example, in a particular x axis where x=-3, it means that three predominant attributes (x=-3) are deleted from the training and test set. The values on y axis represent performance metrics such as accuracy and AUC. The range of values on y axis is from -1 to 1. For example, in a particular y axis that represents the value of accuracy where y = .9100 and x=2, it means that the accuracy is .91 after two redundant attributes are added to training and test set.

Figure IV-1 consists of graphs that show the performance of weak, generic and strong, and ensemble classifiers in accuracy. Figure IV-2 consists of graphs that show performance of weak, generic and strong, and ensemble classifiers in AUC.
Figure IV-1 – Graphs that compare the performance of weak, generic and strong, and ensemble classifiers in accuracy. The classifiers that belong to weak classifiers are HyperPipes (HP), decision stump (DS), and Reptree (REP). The classifiers that belong to generic and strong classifiers are Naive Bayes (NB), decision tree (J48), sequential
minimal optimization (SMO) and k-NN (IBK). The classifiers that belong to ensemble classifiers are Bagging (BAG), AdaBoost (ADA), and Random Forest (RF). The performance is measured in accuracy (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure IV-1(A) and IV-1 (B) display performance of weak classifiers. Figure IV-1(C) and IV-1(D) display the performance of generic and strong classifiers. Figure IV-1(E) and IV-1(F) display the performance of ensemble classifiers.

(A) Weak – Information Gain

(B) Weak - ReliefF

(C)Generic – Information Gain

(D) Generic - ReliefF
Figure IV-2 – Graphs that compare the performance of weak, generic and strong, and ensemble classifiers in AUC. The classifiers that belong to weak classifiers are HyperPipes (HP), decision stump (DS), and Reptree (REP). The classifiers that belong to generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimization (SMO) and k-NN (IBK). The classifiers that belong to ensemble classifiers are Bagging (BAG), AdaBoost (ADA), and Random Forest (RF). The performance is measured in accuracy (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure IV-2(A) and IV-12(B) display performance of weak classifiers. Figure IV-2(C) and IV-2(D) display the performance of generic and strong classifiers. Figure IV-2(E) and IV-2(F) display the performance of ensemble classifiers.

There will be graphs that compare the performance of weak, generic and strong, and ensemble classifiers. Figure IV-3 below shows the legend for the all scatter graphs for weak, generic and strong, and ensemble classifiers. The legend shows the listing of datafiles, which are represented in a shape with color.
Figure IV-3 – the legend for the scatter graphs. The scatter graphs show the changes in performance of weak, generic and strong, and ensemble classifiers after feature deletion or addition. The legend shows the list of datasets that will be used to observe the sensitive of weak, generic and strong, and ensemble classifiers.

The graphs from figure IV-4 to IV-9 are scatter graphs that are created by Excel. Each graph shows a classifier’s change in performance in 34 datasets with information gain or ReliefF. The values on y axis represent can be accuracy or AUC. The values on x axis represent the number of feature deletion or addition. The positive number on x axis indicates the number of feature addition. The negative number on y axis indicates the number of feature deletion. The values on y axis are always zero when x=0. The reason is that the dataset is the original dataset without feature deletion or addition. Therefore, there is no change in performance.

When the feature is added or deleted from the dataset, the graph will compare the difference in accuracy between 0 on x axis and 1 on x axis. For example, given a particular x-axis value, say x=1, if the corresponding y-axis value is positive, it means that adding one attribute (x=1) to the dataset results in a classifier whose accuracy is
better than the accuracy of the classifier trained from the original dataset (i.e., the dataset without adding or deleting any attribute).

Figure IV-4 shows overall performance in weak classifiers where values on the y-axis represent accuracy. The graphs from Figure IV-5 summarize the performance of weak classifiers where values on the y-axis represent AUC.

(A) HP - Information Gain

(B) HP - ReliefF
Figure IV-4 – Graphs show the sensitivity of weak classifiers after feature deletion and addition. Each symbol represents the dataset (see Figure IV-3). There are 34 symbols in each graph. The values on y axis measure the change in accuracy. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure IV-4(A) measures the sensitivity of HyperPipes (HP) with information gain. Figure IV-4(B) measures the sensitivity of HyperPipes (HP) with ReliefF. Figure IV-4(C) measures the sensitivity of decision stump (DS) with information gain. Figure IV-4(D) measures the sensitivity of decision stump (DS) with ReliefF. Figure IV-4(E) measures the sensitivity of Reptree (REP) with information gain. Figure IV-4(F) measures the sensitivity of Reptree (REP) with ReliefF.
Figure IV-5 – Graphs show the sensitivity of weak classifiers after feature deletion and addition. Each symbol represents the dataset (see Figure IV-3). There are 34 symbols in each graph. The values on y axis measure the change in AUC. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure IV-5(A) measures the sensitivity of HyperPipes (HP) with information gain. Figure IV-5(B) measures the sensitivity of HyperPipes (HP) with ReliefF. Figure IV-5(C) measures the sensitivity of decision stump (DS) with information gain. Figure IV-5(D) measures the sensitivity of decision stump (DS) with ReliefF. Figure IV-5(E) measures the sensitivity of Reptree (REP) with information gain. Figure IV-5(F) measures the sensitivity of Reptree (REP) with ReliefF.

Figure IV-6 show overall performance of generic and strong classifiers where values on y axis represent accuracy. The graphs from Figure IV-7 summarize the performance of generic and strong classifiers where values on y axis represent AUC.
Figure IV-6 – Graphs show the sensitivity of generic and strong classifiers after feature deletion and addition. Each symbol represents the dataset (see Figure IV-3). There are 34 symbols in each graph. The values on y axis measure the change in accuracy. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure IV-6(A) measures the sensitivity of Naïve Bayes (NB) with information gain. Figure IV-6(B) measures the sensitivity of Naïve Bayes (NB) with ReliefF. Figure IV-6(C) measures the sensitivity of decision tree (J48) with information gain. Figure IV-6(D) measures the sensitivity of decision tree (J48) with ReliefF. Figure IV-6(E) measures the sensitivity of sequential minimal
optimization (SMO) with information gain. Figure IV-6(F) measures the sensitivity of sequential minimal optimization (SMO) with ReliefF. Figure IV-6(G) measures the sensitivity of k-NN (IBk) with information gain. Figure IV-6(H) measures the sensitivity of k-NN (IBk) with ReliefF.
Figure IV-7 – Graphs show the sensitivity of generic and strong classifiers after feature deletion and addition. Each symbol represents the dataset (see Figure IV-3). There are 34 symbols in each graph. The values on y axis measure the change in AUC. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure IV-7(A) measures the sensitivity of Naïve Bayes (NB) with information gain. Figure IV-7(B) measures the sensitivity of Naïve Bayes (NB) with ReliefF. Figure IV-7(C) measures the sensitivity of decision tree (J48) with information gain. Figure IV-7(D) measures the sensitivity of decision tree (J48) with ReliefF. Figure IV-7(E) measures the sensitivity of sequential minimal
optimization (SMO) with information gain. Figure IV-7(F) measures the sensitivity of sequential minimal optimization (SMO) with ReliefF. Figure IV-7(G) measures the sensitivity of k-NN (IBk) with information gain. Figure IV-7(H) measures the sensitivity of k-NN (IBk) with ReliefF.

Figure IV-8 show overall performance in ensemble classifiers where values on y axis represent accuracy. The graphs from Figure IV-9 summarize the performance of ensemble classifiers where values on y axis represent AUC.
Figure IV-8 – Graphs show the sensitivity of ensemble classifiers after feature deletion and addition. Each symbol represents the dataset (see Figure IV-3). There are 34 symbols in each graph. The values on y axis measure the change in accuracy. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure IV-8(A) measures the sensitivity of Bagging (BAG) with information gain. Figure IV-8(B) measures the sensitivity of Bagging (BAG) with ReliefF. Figure IV-8(C) measures the sensitivity of AdaBoost (ADA) with information gain. Figure IV-8(D) measures the sensitivity of AdaBoost (ADA) with ReliefF. Figure IV-8(E) measures the sensitivity of Random Forest (RF)
with information gain. Figure IV-8(F) measures the sensitivity of Random Forest (RF) with ReliefF.

(A) BAG - Information Gain

(B) BAG - ReliefF

(C) ADA - Information Gain

(D) ADA - ReliefF
Figure IV-9 – Graphs show the sensitivity of ensemble classifiers after feature deletion and addition. Each symbol represents the dataset (see Figure IV-3). There are 34 symbols in each graph. The values on y axis measure the change in AUC. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure IV-9(A) measures the sensitivity of Bagging (BAG) with information gain. Figure IV-9(B) measures the sensitivity of Bagging (BAG) with ReliefF. Figure IV-9(C) measures the sensitivity of AdaBoost (ADA) with information gain. Figure IV-9(D) measures the sensitivity of AdaBoost (ADA) with ReliefF. Figure IV-9(E) measures the sensitivity of Random Forest (RF) with information gain. Figure IV-9(F) measures the sensitivity of Random Forest (RF) with ReliefF.

4.3: Classifier Sensitivity Study on Weak Classifiers

The results show that weak classifiers are sensitive to deleting relevant features in most of the datasets. In some datasets, the performance of weak classifiers changes significantly when removing the first and second relevant feature of the dataset. Based on the graphs from figure IV-4, removing a relevant feature decreases the accuracy of the weak classifiers. In some cases, deleting first two features decreases the accuracy of the weak classifiers dramatically. And also, in some datasets, dropping a relevant feature improves the accuracy of weak classifiers. The interpretations are supported by
information gain and ReliefF, which are the methods that rank the importance of attributes from the dataset. The sensitivity of weak classifiers is small when adding a feature. The results (figure IV-4) show that the accuracy drops slightly when adding a redundant feature.

The results (figure IV-5) show that AUC for Reptree changes significantly when deleting a relevant feature. The values for AUC tend to be decreased when performing feature deletion. When adding a feature, the values for AUC decrease slightly. The results (figure IV-5) also show that AUC for HyperPipes and decision stump change significantly when adding or dropping the first feature. The values for AUC are increased when the feature is added or deleted. The interpretations are supported by information gain and ReliefF.

4.3.1: Summary on Weak Classifiers

<table>
<thead>
<tr>
<th></th>
<th>-5</th>
<th>-4</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>+1</th>
<th>+2</th>
<th>+3</th>
<th>+4</th>
<th>+5</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>0.0287</td>
<td>0.0235</td>
<td>-0.0161</td>
<td>-0.0090</td>
<td>-0.0025</td>
<td>0.6149</td>
<td>0.0181</td>
<td>0.0181</td>
<td>0.0181</td>
<td>0.0181</td>
<td>0.0181</td>
</tr>
<tr>
<td></td>
<td>16/18/0</td>
<td>16/18/0</td>
<td>17/17/0</td>
<td>19/15/0</td>
<td>18/16/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HP</td>
<td>-0.0934</td>
<td>-0.0802</td>
<td>-0.0623</td>
<td>-0.037</td>
<td>-0.0054</td>
<td>0.7039</td>
<td>0.0195</td>
<td>0.018</td>
<td>0.0168</td>
<td>0.0154</td>
<td>0.014</td>
</tr>
<tr>
<td></td>
<td>12/22/0</td>
<td>13/21/0</td>
<td>14/20/0</td>
<td>15/19/0</td>
<td>19/15/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REP</td>
<td>-0.0676</td>
<td>-0.0559</td>
<td>-0.0422</td>
<td>-0.0256</td>
<td>0</td>
<td>0.7931</td>
<td>0.019</td>
<td>0.0156</td>
<td>0.0127</td>
<td>0.0105</td>
<td>0.0081</td>
</tr>
<tr>
<td></td>
<td>9/25/0</td>
<td>11/23/0</td>
<td>13/21/0</td>
<td>12/22/0</td>
<td>16/18/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(A)

<table>
<thead>
<tr>
<th></th>
<th>-5</th>
<th>-4</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>+1</th>
<th>+2</th>
<th>+3</th>
<th>+4</th>
<th>+5</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>0.0315</td>
<td>0.0369</td>
<td>0.0417</td>
<td>0.0461</td>
<td>0.049</td>
<td>0.7201</td>
<td>0.0377</td>
<td>0.0454</td>
<td>0.0483</td>
<td>0.0497</td>
<td>0.0504</td>
</tr>
<tr>
<td></td>
<td>19/15/0</td>
<td>20/14/0</td>
<td>20/14/0</td>
<td>20/14/0</td>
<td>21/13/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HP</td>
<td>0.0023</td>
<td>0.0078</td>
<td>0.0129</td>
<td>0.0175</td>
<td>0.0206</td>
<td>0.7527</td>
<td>0.0165</td>
<td>0.0198</td>
<td>0.0207</td>
<td>0.021</td>
<td>0.021</td>
</tr>
<tr>
<td></td>
<td>16/18/0</td>
<td>17/17/0</td>
<td>17/17/0</td>
<td>17/17/0</td>
<td>17/17/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REP</td>
<td>-0.0266</td>
<td>-0.0209</td>
<td>-0.0153</td>
<td>-0.0103</td>
<td>-0.0057</td>
<td>0.7806</td>
<td>-0.0001</td>
<td>-0.0008</td>
<td>-0.0015</td>
<td>-0.0021</td>
<td>-0.0028</td>
</tr>
<tr>
<td></td>
<td>14/20/0</td>
<td>14/20/0</td>
<td>15/19/0</td>
<td>16/18/0</td>
<td>16/18/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(B)
Table IV-1 – The table above shows overall changes from weak classifiers’ original values that are listed under column 0. Table IV-1(A) shows a table that consists of values in accuracy. Figure IV-1(B) shows a table that consists of values in AUC. The negative numbers on top of column indicate the number of feature deletion. The positive numbers on top of column indicate the number of feature addition. Each cell that is under negative numbers or positive numbers has two smaller cells. The cell on top indicates the average change in value from the original value in specific classifier. The cell on bottom indicates how many datasets have better accuracy than original value (win), how many datasets have worse accuracy than original value (lose), and how many datasets have same accuracy than original value (tie). The format is expressed W/L/T.

The tables, which are in table IV-1, show overall changes from the classifiers’ original value after feature deletion and addition. The format win/lose/tie is explained in table IV-1. Table IV-1(A) shows decrease in accuracy after feature deletion and increase in accuracy after feature addition. The numbers, in the format of win/lose/tie, support the fact that almost half of the datasets has decreased their accuracy after feature deletion and addition. In table IV-1(B), the table shows classifier DS and HP have increased their AUC when feature is deleted or added. However, classifier REP shows decrease in AUC when feature is deleted or added. These observations are supported by the number with the format of win/lose/tie.

The graphs (figure IV-4) show that the accuracy for HiperPipes (HP) classifier has dropped significantly after deleting two features. While using information gain for feature deletion, only 24 datasets are getting worse. While using ReliefF, only 20 datasets are getting worse. As for decision stump (DS), the classifier has dropped significantly after deleting a feature. While using information gain method, only 13 datasets are getting worse. And also, while using ReliefF method, only 11 datasets are getting worse. The accuracy for Reptree (REP) classifier has dropped significantly after removing a feature. While using information gain, only 27 datasets are getting worse. And also, while using reliefF method, only 27 datasets are getting worse.
The overall results show that performance of weak classifiers in accuracy drops significantly when deleting a relevant feature. The performance of weak classifiers does not change significantly when adding a redundant attribute.

4.4: Classifier Sensitivity Study on Generic and Strong Classifiers

The results show that generic and strong classifiers are sensitive to feature deletion. Based on the graphs from figure IV-6, deleting a relevant feature decreases the accuracy of the generic and strong classifiers. In some cases, dropping any relevant features increases the accuracy of the generic and strong classifiers dramatically. The interpretations are supported by information gain and ReliefF. The sensitivity of generic and strong classifiers is small when adding a redundant feature except the classifier k-NN. Figure IV-6(G) and IV-6(H) show that in some datasets, adding a redundant feature decreases accuracy significantly.

As for AUC (figure IV-7), the values for Naïve Bayes and decision tree improve significantly when adding or deleting the first feature. Generally, the value for AUC is decreased when the feature is added or deleted.

In some instances, the result also shows that AUC for SVM and k-NN decreases dramatically when deleting the first relevant feature. The interpretations are supported by information gain and ReliefF.

4.4.1: Summary on Generic and Strong Classifiers

<table>
<thead>
<tr>
<th></th>
<th>-5</th>
<th>-4</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>+1</th>
<th>+2</th>
<th>+3</th>
<th>+4</th>
<th>+5</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>-0.0823</td>
<td>-0.0698</td>
<td>-0.0563</td>
<td>-0.0345</td>
<td>-0.0137</td>
<td>0.7763</td>
<td>0.0214</td>
<td>0.0207</td>
<td>0.02</td>
<td>0.0194</td>
<td>0.0188</td>
</tr>
<tr>
<td>J48</td>
<td>-0.0753</td>
<td>-0.0652</td>
<td>-0.0478</td>
<td>-0.0253</td>
<td>-0.0087</td>
<td>0.835</td>
<td>0.0226</td>
<td>0.0209</td>
<td>0.0194</td>
<td>0.0179</td>
<td>0.0164</td>
</tr>
</tbody>
</table>
Table IV-2 – The table above shows overall changes from generic and strong classifiers’ original values that are listed under column 0. Table IV-2(A) shows a table that consists of values in accuracy. The negative numbers on top of column indicate the number of feature deletion. The positive numbers on top of column indicate the number of feature addition. Each cell that is under negative numbers or positive numbers has two smaller cells. The cell on top indicates the average change in value from the original value in specific classifier. The cell on bottom indicates how many datasets have better accuracy than original value (win), how many datasets have worse accuracy than original value (lose), and how many datasets have same accuracy than original value (tie). The format is expressed W/L/T.

Table IV-2 shows overall changes in original value for the generic and strong classifiers after feature deletion and addition. In table IV-2(A), the table shows decrease in accuracy after feature deletion. As for feature addition, there is increase in accuracy except for classifier k-NN (IBk). The numbers, win/lose/tie, support the fact that almost half of the datasets has decreased their accuracy after feature deletion and addition for classifier IBk. In table IV-2(B), the table shows that the values for AUC have decreased after feature deletion or addition.
The graphs (figure IV-6) show that the accuracy for Naïve Bayes classifier has reduced significantly after deleting two features. While using information gain, only 32 datasets are getting worse. And also, while using reliefF, only 27 datasets are getting worse. As for decision tree (J48), the accuracy is decreased significantly after deleting two features. While using information gain, only 23 datasets are getting worse. And also, while using reliefF, only 28 datasets are getting worse. For classifier k-NN (IBk), the accuracy has dropped greatly after adding two redundant features. While using information gain, only 30 datasets are getting worse. And also, only 30 datasets are getting worse while using reliefF.

Study shows that generic and strong classifiers are sensitive to feature deletion and feature addition. The accuracy is generally decreased after deleting a relevant feature. As for feature addition, the accuracy decreases significantly after adding a redundant feature. However, the changes do not have the same impact as feature deletion.

4.5: Classifier Sensitivity Study on Ensemble Classifiers

The results show that ensemble classifiers are sensitive to removing relevant features in most of the dataset. In some datasets, the performance of ensemble classifiers is very sensitive to dropping the first and second relevant feature of the dataset. Based on the graphs from figure IV-8, deleting a relevant feature decreases the accuracy of the ensemble classifiers. In some cases, removing first two features decreases the accuracy of the ensemble classifiers dramatically. And also, in some datasets, dropping a relevant feature improves the accuracy of ensemble classifiers. The interpretations are supported
by information gain and ReliefF, which are the methods that rank the importance of attributes from the dataset.

The sensitivity of ensemble classifiers is significant when adding a feature. However, the changes do not have greater impact than dropping a relevant feature. Figure IV-8(C) and IV-8(D) show that the Adaboost’s accuracy drops significantly when adding the first redundant feature.

Overall, the results (figure IV-9) show that values of AUC change significantly when dropping the first feature. The values for AUC tend to be decreased when performing feature deletion or addition.

### 4.5.1: Summary on Ensemble Classifiers

<table>
<thead>
<tr>
<th></th>
<th>-5</th>
<th>-4</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>+1</th>
<th>+2</th>
<th>+3</th>
<th>+4</th>
<th>+5</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BAG</td>
<td>-0.0678</td>
<td>-0.056</td>
<td>-0.0446</td>
<td>-0.0245</td>
<td>-0.0055</td>
<td>0.8244</td>
<td>0.0165</td>
<td>0.0115</td>
<td>0.0066</td>
<td>0.0027</td>
<td>0.0001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8/26/0</td>
<td>9/25/0</td>
<td>12/22/0</td>
<td>11/23/0</td>
<td>13/21/0</td>
<td>18/16/0</td>
<td>19/15/0</td>
<td>18/16/0</td>
<td>18/16/0</td>
<td>18/16/0</td>
<td>18/16/0</td>
<td></td>
</tr>
<tr>
<td>ADA</td>
<td>-0.1015</td>
<td>-0.0857</td>
<td>-0.073</td>
<td>-0.0464</td>
<td>-0.0301</td>
<td>0.8743</td>
<td>-0.012</td>
<td>-0.0199</td>
<td>-0.0262</td>
<td>-0.0312</td>
<td>-0.0358</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8/26/0</td>
<td>9/25/0</td>
<td>10/24/0</td>
<td>12/22/0</td>
<td>12/22/0</td>
<td>15/19/0</td>
<td>14/20/0</td>
<td>12/22/0</td>
<td>12/22/0</td>
<td>12/22/0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF</td>
<td>-0.1014</td>
<td>-0.0889</td>
<td>-0.0751</td>
<td>-0.0546</td>
<td>-0.0325</td>
<td>0.9013</td>
<td>-0.0208</td>
<td>-0.0275</td>
<td>-0.0343</td>
<td>-0.0405</td>
<td>-0.9013</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9/25/0</td>
<td>9/25/0</td>
<td>10/24/0</td>
<td>12/22/0</td>
<td>13/21/0</td>
<td>13/21/0</td>
<td>13/21/0</td>
<td>12/22/0</td>
<td>12/22/0</td>
<td>11/23/0</td>
<td>0/34/0</td>
<td></td>
</tr>
</tbody>
</table>

**Table IV-3** – The table above shows overall changes from ensemble classifiers’ original values that are listed under column 0. Table IV-3(A) shows a table that consists of values in accuracy. Table IV-3(B) shows a table that consists of values in AUC. The negative numbers on top of column indicate the number of feature deletion. The positive numbers on top of column indicate the number of feature addition. Each cell that is under negative values indicates the change in value from the original value.
numbers or positive numbers has two smaller cells. The cell on top indicates the average change in value from the original value in specific classifier. The cell on bottom indicates how many datasets have better accuracy than original value (win), how many datasets have worse accuracy than original value (lose), and how many datasets have same accuracy than original value (tie). The format is expressed W/L/T.

Table IV-3 shows overall changes from ensemble classifiers’ original value after feature deletion and addition. In table IV-3(A), the table shows decrease in accuracy after feature deletion and addition. The numbers, in the format of win/lose/tie, support the fact that almost half of the datasets has decreased their accuracy after feature deletion and addition. In table IV-3(B), the table shows that ensemble classifiers have reduced their AUC after feature deletion. And also, ensemble classifiers have improved their AUC after feature addition.

For Bagging, adding a redundant feature to the 34 datasets decreases accuracy slightly except for dataset (Vote). The accuracy drops significantly after the second redundant attribute is added to the datasets. After second attribute is deleted from the 34 datasets, 31 datasets show decrease in accuracy with Bagging and 30 datasets show decrease in accuracy with AdaBoost.

For Random Forest, the graph shows that dataset (Iris) dropped significantly in accuracy after three features being deleted. As for dataset (Ecoli), the graph shows a major drop in accuracy after four features being removed.

Overall, Bagging has better accuracy than AdaBoost and Random Forest after adding or deleting a feature. In fact, the accuracy for Bagging has improved. AdaBoost and Random Forest have nearly same performance in accuracy after adding or deleting a feature. However, AdaBoost has better accuracy than Random Forest.
Study shows that ensemble classifiers are sensitive to feature deletion and less sensitive to feature addition. There is decrease in accuracy when dropping a relevant feature. As for feature addition, the accuracy is decreased when adding a redundant feature. The changes can be considered negligible because the variance can be the factor that explains minor drop in accuracy while adding a redundant feature.

4.6: Summary on Weak, Generic and Strong and Ensemble Classifiers

(A) Feature Deletion – Accuracy
(B) Feature Addition - Accuracy

(C) Feature Deletion - AUC
Figure IV-10 – The bar graphs that show summary of win, lose and tie. The legend on the right side shows number of feature deletion or addition. Positive number indicates the number of feature addition. Negative number indicates the number of feature deletion. The values of y axis represent the total number of win, lose or tie. The values on x axis represent categorical value of win, lose or tie. There are three categorical values in a weak, generic and strong, or ensemble group. Figure IV-10(A) and IV-10(B) consist of summary of win, lose and tie relating to accuracy. Figure IV-10(C) and IV-10(D) consist of summary of win, lose and tie relating to AUC.

General Summary:

The study shows that classifiers in three groups are sensitive to feature deletion. The performance in accuracy tends to be decreased when deleting any feature. Base on figure IV-10, the analysis shows that classifiers in three groups are sensitive to feature deletion. The performance in accuracy tends to be decreased when deleting any feature. Figure IV-10(A) shows that there are more loses than wins in feature deletion. This result means that there is more lost than gain in accuracy.

As for feature addition, the study shows that weak and ensemble classifiers are less sensitive to feature addition. For ensemble classifiers, there is significantly decrease
in accuracy while adding a redundant feature. Figure IV-10(B) indicates that there are more loses than gain in accuracy for ensemble classifiers. The changes in performance for feature addition do not have same impact as feature deletion.

Itemized Summary:

- Bagging is the most robust classifier because it achieves high accuracy and it is less sensitive to feature deletion or addition. In fact, Bagging has improved its accuracy as a result of feature deletion or addition.
- IBk classifier is the most sensitive of all classifiers because it has highest drop in accuracy after feature deletion or addition.
- The performance of ensemble classifiers is better than other classifiers because they are less sensitive in feature deletion and addition.
CHAPTER 5

CLASSIFIER SENSITIVITY STUDY ON BIOMEDICAL HIGH DIMENSIONAL DATA

5.1: Purpose

In this chapter, experiments are conducted to study the sensitivity of classifiers relating to gene expression and biomedical document data. Since each dataset relating to biomedical high dimensional data typically consists over thousands of attributes, therefore, there will be 50 feature deletions and 50 feature additions in order to observe the sensitivity of classifiers. The feature deletions and additions will be applied to training and test set. The performance of classifiers will be analyzed with graphs such as Jfreechart and Excel’s scatter graphs. And also, there will be tables that will determine if feature deletion or addition actually improves the performance of classifiers.

5.2: Introduction to Gene Expression Data

In order to understand gene expression data, the basic understanding of cell will be discussed in this thesis. Cell is the basic biological unit of all known living organisms. The phospholipids, which are a class of lipids, separate two different cells. The nucleus
of the cell contains genetic information that is stored in DNA, which stands for deoxyribonucleic acid. DNA contains deoxyribose sugars which are the basic units of carbohydrates. Carbohydrates give energy and they are very important molecule for humans.

DNA is a nucleic acid that is polymers of nucleotides. Each nucleotide composes of organic molecules such as a pentose, a phosphate group and nitrogenous base. Nitrogenous base can be adenine, guanine, cytosine and thymine. A combination of nucleotides can form a single strand of nucleic acid. Each nucleotide in a single strand of nucleic acid can form base pair with other nucleotide from other strand of nucleic acid. A base pair is the unit of DNA. A hydrogen bond allows these base pair to be bonded together. A base pair can be A-T pair (adenine and thymine) and G-C pair (guanine and cytosine). Therefore, each DNA has two strands of nucleic acid that are arranged in helix shape with hydrogen bonds holding them together.

Gene is a fragment of DNA that can be activated to synthesize protein or perform other functional groups. A protein is a long chain of amino acid that performs various living organisms’ functions such as catalyzing metabolic reaction and replicating DNA. Most of proteins are enzymes that speed up biochemical reaction. In order to synthesize a specific enzyme, some genes are expressed and other genes are not expressed. The structure of gene consists of head, introns and exons. Introns contain code that is not
responsible for protein synthesis. Exons contain code that is responsible for protein synthesis.

The combination of DNA and other protein molecule that fit in the nucleus of the cell is called chromatin. Chromatin packages DNA with histones, a special protein, in order to fit in the nucleus of the cell. And also chromatin strengthens DNA during mitosis, in which cell is divided into two. The replication of DNA occurs before the division of the cell. Chromatin can further condense, in which DNA is condensed 10,000 times onto itself. This new structure is called chromosome. The chromosome is packaged in the nucleus of the cell. In human cells, there are 23 pairs of chromosomes. One set of human chromosomes determines the sex trait. Figure V-1 shows where DNA is derived from the cell in eukaryote. Eukaryote is a cell with nucleus and organelles in a membrane.

![Diagram](image)

**Figure V-1** – A diagram [57] that shows where DNA (deoxyribonucleic acid) is coming from. The DNA is condensed into a complex structure called chromosome. The chromosome is packaged in the nucleus. The diagram shows that the nucleus is part of cell in eukaryote. Eukaryote is a cell with nucleus and organelles in a membrane.

RNA, which is ribonucleic acid, compos of nucleic acid and it serves the role of expression of genes, coding and decoding. One type of the RNAs that involves protein
synthesis is called messenger RNA, which is mRNA. The synthesis of mRNA is carried out by enzyme called RNA polymerase.

The process of translation can be described in four steps. First, RNA polymerase binds to promoter region of DNA. Second, promoter region of DNA sends signals to RNA polymerase to make DNA to create complementary strands of RNA. Third, introns are eliminated thru splicing and exons are joined together to form mRNA. Fourth, mRNA will attach to ribosome, a site responsible for protein synthesis. Then, mRNA will be used as protein production. How many can mRNA proteins make are depended on the stability of mRNA and also its cellular condition.

The flow of genetic information within biological system can be described in three processes. They are replication, transcription and translation. These processes are central dogma of molecular biology. The process of replication involves the duplication of DNA. The process of transcription involves RNA synthesis. The process of translation involves protein synthesis. Figure V-2 below shows the process of gene expression. DNA is transcribed into RNA and then RNA is translated into protein.

Figure V-2 – the diagram [32] shows the processes of gene expression, which is flow of genetic information. After replication of DNA, the gene can be expressed into RNA
(transcription) which it can be translated into protein (translation). These processes are central dogma of molecular biology.

The purpose of classifying gene expression data is to identify specific genes, which are expressed to create specific cell, in human’s DNA. There are many benefits of identifying genes. One of the benefits is early detection of cancer cell. This benefit helps people, who will have the cancer in later age, to seek early treatment.

5.3: Introduction to Biomedical Document Data

The biomedical document refers the information relating to biomedical and molecular biology domain [51]. In general, biomedical document consists of medical research relating to human development. There are many examples of biomedical document. They are identification of protein or gene, association of gene clusters, and extraction of protein interaction [55].

There are a lot of interests in text mining in the area of biomedical document because most of publications are available in the database. Text mining is the technology that discovers the patterns and trends from huge collections of unstructured text. Text mining utilizes basic technology such as natural language processing, information retrieval, information extraction and data mining. The use of text mining helps people to pre-screen the document in order to find the area of interests. The keys to achieve max performance in retrieving biomedical document are medical subject heading, descriptors, keywords relating to authors and indexers [55]. One example of biomedical documents that can be used for categorization is literature relating to proteins with gene ontology codes. The codes can be abbreviated as GO code. This helps the researcher to sort enormous amount of documents.
The classification relating to biomedical data is challenging due to enormous amount biomedical terminology and the large volume of published work. One of the examples which database contains biomedical document is PIR-PSD. PIR-PSD stands for protein sequence database created by protein information resources. PIR-PSD contains comprehensive information about protein sequences and now it is merged with another source of information, which is UniProt [52].

IBM’s MedTAKMI, an extension of TAKMI, mines collection of document with biomedical categories. This tool helps people to extract knowledge about specific topic after scanning few documents. MedTAKMI is capable of mining 11 million biomedical journal abstracts [53].

There are number of work relating to text mining. First, Hearst proposed a system for predicting the function of unknown genes base on biomedical document. Second, Swanson also proposed a system that will discover the knowledge from the biomedical literature. Therefore, there are numerous researches involve in the area of concept extraction, relationship extraction and network/pathway construction for protein-protein interaction.

There are problems relating text mining. First, there are vast amount of information in the literature. Second, the most of existing systems with biomedical documents are non-interactive which makes it difficult to apply data mining to the system. The reason that the system is non-interactive is that extracted concepts and relationships are provided in a fix way [53].
In order to solve this problem, IBM develops a new text-mining system, which is called MedTAKMI [53]. This system is capable of handling a large amount of biomedical document in an interactive manner. It helps the user to apply data mining with trial and error approach. MedTAKMI has two primary subtasks that extract information relating to genes, proteins and diseases. These methods are entity extraction and relation extraction. Entity extraction involves dictionary lookup. Entity extraction refers the concept of gene, protein and other biomedical terminology. MedTAKMI has developed a large domain dictionary that consists of 2 million biomedical entities. Relation extraction refers the extraction of relationship among these entities. For example, A inhibits B or A activates B is an entity relation. IBM’s MedTAKMI uses syntactic information from shallow parsing and full parsing to extract binary and ternary relationships. Binary relationship consists of relationship between noun and verb. Ternary consists of two nouns and a verb relationship.

5.4: Experiment

For gene expression data, there are two datasets that will be used for studies. They are DLBCL-NIH-ALL_240x7399.arff and lung-harvard_203x12600.arff. For biomedical document data, there are two datasets that will be used for the studies. They are oh5.wc.arff and oh15.wc.arff. Table V-1 below shows the information about datasets from gene expression and biomedical document data.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Number of Instances</th>
<th>Number of Attributes</th>
<th>Number of Class Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLBCL-NIHALL_240x7399</td>
<td>240</td>
<td>7400</td>
<td>2</td>
</tr>
<tr>
<td>Lung-harvard_203x12600</td>
<td>203</td>
<td>12601</td>
<td>5</td>
</tr>
</tbody>
</table>

(A)
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Number of Instances</th>
<th>Number of Attributes</th>
<th>Number of Class Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oh5.wc</td>
<td>918</td>
<td>3013</td>
<td>10</td>
</tr>
<tr>
<td>Oh15.wc</td>
<td>913</td>
<td>3101</td>
<td>10</td>
</tr>
</tbody>
</table>

(B)

**Table V-1** – Table V-1(A) shows the listing of gene expression datasets and table V-1(B) shows the listing of biomedical document datasets. The first column shows the name of datasets. The second column indicates the number of instances for the dataset. The third column shows the number of attributes for the dataset. The last column shows the number of class variables for the dataset.

### 5.4.1: Performance for Gene Expression Data

Figure V-3 and V-4 consist of graphs that show the performance (accuracy) of weak, generic and strong, and ensemble classifier. The evaluation for classifiers in figure V-3 is based on dataset DLBCL-NIH-ALL_240x7399 and the evaluation for classifiers in figure V-4 is based on dataset lung-harvard_203x12600.

(A) Weak - Information Gain

(B) Weak - ReliefF
Figure V-3 – The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with DLBCL-NIH-ALL_240x7399. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimization (SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in accuracy (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-3(A) shows the sensitivity of weak classifiers with information gain. Figure V-3(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-3(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-3(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-3(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-3(F) shows the sensitivity of ensemble classifiers with ReliefF.
Figure V-4 – The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with lung-harvard_203x12600. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimization (SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in accuracy (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-4(A) shows the sensitivity of weak classifiers with information gain. Figure V-4(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-4(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-4(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-4(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-4(F) shows the sensitivity of ensemble classifiers with ReliefF.

Figure V-5 consists of graphs that show performance of weak, generic and strong, and ensemble classifiers in AUC for dataset DLBCL-NIH-ALL_240x7399. Figure V-6
consists of graphs that show the performance of weak, generic and strong and ensemble classifiers in AUC for dataset lung-harvard_203x12600.

Figure V-5 – The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with DLBCL-NIH-ALL_240x7399. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimization
(SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in AUC (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-5(A) shows the sensitivity of weak classifiers with information gain. Figure V-5(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-5(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-5(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-5(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-5(F) shows the sensitivity of ensemble classifiers with ReliefF.
**Figure V-6** – The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with lung-harvard_203x12600. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimization (SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in AUC (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-6(A) shows the sensitivity of weak classifiers with information gain. Figure V-6(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-6(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-6(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-6(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-6(F) shows the sensitivity of ensemble classifiers with ReliefF.

Figure V-8 consists of graphs that summarize the performance of weak, generic and strong, and ensemble classifiers while using DLBCL-NIH-ALL_240x7399 dataset. And also, the graphs from figure V-10 summarize the performance of weak, generic and strong, and ensemble classifiers while using lung-harvard_203x12600. Figure V-7 is the legend for the scatter graphs. The legend shows the listing of classifiers, which are represented in a shape with color. For example, the classifier AdaBoost is represented in orange circle.

**Figure V-7** – the legend shows the list of classifiers for the scatter graphs. The scatter graphs show the changes in performance of weak, generic and strong and ensemble classifiers after feature deletion or addition.
Figure V-8 – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with DLBCL-NIH-ALL_240x7399 dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in Accuracy. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-8(A) measures the sensitivity of ten classifiers with information gain. Figure V-8(B) measures the sensitivity of ten classifiers with ReliefF.
Figure V-9 – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with lung-harvard_203x12600 dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in Accuracy. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-9(A) measures the sensitivity of ten classifiers with information gain. Figure V-9(B) measures the sensitivity of ten classifiers with ReliefF.

The graphs from Figure V-10 summarize the performance of weak, generic and strong, and ensemble classifiers in AUC with DLBCL-NIH-ALL_240x7399 dataset. And also, the graphs from figure V-11 summarize the performance of weak, generic and strong, and ensemble classifiers in AUC with lung-harvard_203x12600 dataset.

Figure V-10 – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with DLBCL-NIH-ALL_240x7399 dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in AUC. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-10(A) measures the sensitivity of ten classifiers with information gain. Figure V-10(B) measures the sensitivity of ten classifiers with ReliefF.
Figure V-11 – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with lung-harvard_203x12600 dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in AUC. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-11(A) measures the sensitivity of ten classifiers with information gain. Figure V-11(B) measures the sensitivity of ten classifiers with ReliefF.

5.4.2: Performance for Biomedical Document Data

Figure V-12 and V-13 consist of graphs that show the performance (accuracy) of weak, generic and strong, and ensemble classifier. The evaluation for classifiers in figure V-12 is based on dataset oh5.wc and the evaluation for classifiers in figure V-13 is based on oh15.wc.
The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with oh5.wc. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimization (SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in
accuracy (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-12(A) shows the sensitivity of weak classifiers with information gain. Figure V-12(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-12(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-12(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-12(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-12(F) shows the sensitivity of ensemble classifiers with ReliefF.
(E) Ensemble – Information Gain          (F) Ensemble - ReliefF

Figure V-13 – The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with oh15.wc. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimisation (SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in accuracy (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-13(A) shows the sensitivity of weak classifiers with information gain. Figure V-13(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-13(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-13(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-13(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-13(F) shows the sensitivity of ensemble classifiers with ReliefF.

Figure V-14 consists of graphs that show the performance of weak, generic and strong, and ensemble classifiers in AUC for dataset oh5.wc. Figure V-15 consists of graphs that show the performance of weak, generic and strong and ensemble classifiers in AUC for oh15.wc.
Figure V-14 – The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with oh5.wc. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB), decision tree (J48), sequential minimal optimization (SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in AUC (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-14(A) shows the sensitivity of weak classifiers with information gain. Figure V-14(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-14(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-14(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-14(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-14(F) shows the sensitivity of ensemble classifiers with ReliefF.
Figure V-15 – The graphs measure the sensitivity of weak, generic and strong and ensemble classifiers with oh15.wc. The weak classifiers are HyperPipes (HP), decision stump (DS) and Reptree (REP). The generic and strong classifiers are Naïve Bayes (NB),
decision tree (J48), sequential minimal optimization (SMO) and k-NN (IBk). The ensemble classifiers are Bagging (BAG), AdaBoost (ADA) and Random Forest (RF). The graphs utilize information gain and ReliefF. The performance is measured in AUC (y axis). The positive number (x axis) indicates the number of feature addition. The negative number (x axis) indicates the number of feature deletion. Figure V-15(A) shows the sensitivity of weak classifiers with information gain. Figure V-15(B) shows the sensitivity of weak classifiers with ReliefF. Figure V-15(C) shows the sensitivity of generic and strong classifiers with information gain. Figure V-15(D) shows the sensitivity of generic and strong classifiers with ReliefF. Figure V-15(E) shows the sensitivity of ensemble classifiers with information gain. Figure V-15(F) shows the sensitivity of ensemble classifiers with ReliefF.

The graphs from Figure V-16 summarize the performance (accuracy) of weak, generic and strong, and ensemble classifiers with oh5.wc dataset. And also, the graphs from figure V-17 summarize the performance (accuracy) of weak, generic and strong, and ensemble classifiers with oh15.wc dataset.

**Figure V-16** – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with oh5.wc dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in Accuracy. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-16(A)
measures the sensitivity of ten classifiers with information gain. Figure V-16(B) measures the sensitivity of ten classifiers with ReliefF.

Figure V-17 – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with oh15.wc dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in Accuracy. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-17(A) measures the sensitivity of ten classifiers with information gain. Figure V-17(B) measures the sensitivity of ten classifiers with ReliefF.

The graphs from Figure V-18 summarize the performance (AUC) of weak, generic and strong, and ensemble classifiers with oh5.wc dataset. And also, the graphs from figure V-19 summarize the performance (AUC) of weak, generic and strong, and ensemble classifiers with oh15.wc dataset.
Figure V-18 – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with oh5.wc dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in AUC. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-18(A) measures the sensitivity of ten classifiers with information gain. Figure V-18(B) measures the sensitivity of ten classifiers with ReliefF.
Figure V-19 – Graphs show the sensitivity of weak, generic and strong and ensemble classifiers with oh15.wc dataset. Each symbol represents the classifiers (see Figure V-7). There are 10 symbols in each graph. The values on y axis measure the change in AUC. The positive number on x axis shows the number of feature addition. The negative number on the x axis shows the number of feature deletion. Figure V-19(A) measures the sensitivity of ten classifiers with information gain. Figure V-19(B) measures the sensitivity of ten classifiers with ReliefF.

Section 5.5: Classifier Sensitivity Study on Gene Expression Data

The results show that all classifiers are not sensitive to feature deletion and addition for those two datasets except for SMO and J48. For the dataset lung-harvard_203x12600, SMO and J48 are more sensitive to feature deletion and addition. Overall, SMO is sensitive to feature deletion and addition.

The results show that AUC for weak, generic and strong, and ensemble classifiers do not change significantly when adding or deleting a feature. For the classifier likes HyperPipes, decision stump or k-NN, there are significant change in AUC when deleting or adding the first feature.

5.5.1: Summary on Gene Expression Data

<table>
<thead>
<tr>
<th></th>
<th>-50</th>
<th>-40</th>
<th>-30</th>
<th>-20</th>
<th>-10</th>
<th>0</th>
<th>+10</th>
<th>+20</th>
<th>+30</th>
<th>+40</th>
<th>+50</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>-0.0358</td>
<td>-0.0383</td>
<td>-0.0383</td>
<td>-0.0311</td>
<td>-0.0311</td>
<td>0.6906</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
</tr>
<tr>
<td>HP</td>
<td>-0.0021</td>
<td>-0.0021</td>
<td>-0.0021</td>
<td>-0.001</td>
<td>-0.001</td>
<td>0.7883</td>
<td>-0.0001</td>
<td>-0.0005</td>
<td>-0.0007</td>
<td>-0.0008</td>
<td>-0.0006</td>
</tr>
<tr>
<td></td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
</tr>
<tr>
<td>REP</td>
<td>-0.0077</td>
<td>-0.0077</td>
<td>-0.0077</td>
<td>-0.004</td>
<td>-0.0056</td>
<td>0.7854</td>
<td>-0.0018</td>
<td>-0.0024</td>
<td>-0.0034</td>
<td>-0.004</td>
<td>-0.0042</td>
</tr>
<tr>
<td></td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
</tr>
</tbody>
</table>

(A)

<table>
<thead>
<tr>
<th></th>
<th>-50</th>
<th>-40</th>
<th>-30</th>
<th>-20</th>
<th>-10</th>
<th>0</th>
<th>+10</th>
<th>+20</th>
<th>+30</th>
<th>+40</th>
<th>+50</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>-0.0042</td>
<td>-0.0052</td>
<td>-0.0031</td>
<td>-0.001</td>
<td>0.001</td>
<td>0.7802</td>
<td>0</td>
<td>0.0001</td>
<td>-0.0004</td>
<td>-0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td></td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
</tr>
<tr>
<td>J48</td>
<td>-0.0015</td>
<td>-0.0015</td>
<td>0.0058</td>
<td>-0.001</td>
<td>0.001</td>
<td>0.8317</td>
<td>-0.0011</td>
<td>-0.0024</td>
<td>-0.0025</td>
<td>-0.0024</td>
<td>-0.0032</td>
</tr>
<tr>
<td></td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
</tr>
</tbody>
</table>

90
Table V-2 – The table above shows overall changes from weak, generic and strong, and ensemble classifiers’ original values that are listed under column 0. The values represent accuracy. Table V-2(A) shows the sensitivity of weak classifiers. Table V-2(B) shows the sensitivity of generic and strong classifiers. Table V-2(C) shows the sensitivity of ensemble classifiers. The negative numbers on top of column indicate the number of feature deletion. The positive numbers on top of column indicate the number of feature addition. Each cell that is under negative numbers or positive numbers has two smaller cells. The cell on top indicates the average change in value from the original value in specific classifier. The cell on bottom indicates how many datasets have better accuracy than original value (win), how many datasets have worse accuracy than original value (lose), and how many datasets have same accuracy than original value (tie). The format is expressed W/L/T.

Table V-2 shows overall changes from the classifiers’ original value after feature deletion and addition. In table V-2(A), the table shows that accuracies for weak classifiers have either decreased slightly or remained unchanged after feature deletion or addition. In table V-2(B), the table shows that accuracies for generic and strong classifiers have either decreased slightly or remained unchanged after feature deletion or addition. In table V-2(C), the table shows that accuracies for ensemble classifiers have either decreased slightly or remained unchanged after feature deletion or addition.
The bar graphs that show summary of win, lose and tie. The legend on the right side shows number of feature deletion or addition. Positive number indicates the number of feature addition. Negative number indicates the number of feature deletion. The values of y axis represent the total number of win, lose or tie. The values on x axis represent categorical value of win, lose or tie. There are three categorical values in a weak, generic and strong, or ensemble group. Figure V-20(A) and V-20(B) consist of summary of win, lose and tie relating to accuracy.
In general, the accuracies for weak, generic and strong, and ensemble classifiers do not change significantly when deleting or adding the features. The graphs from figure V-20 support the fact that accuracy does not change after feature deletion and addition.

5.6: Classifier Sensitivity Study on Biomedical Document Data

The results show that all classifiers are sensitive to feature deletion for dataset oh5.wc and oh15.wc except for HyperPipes. In general, the results of feature deletion reduce the accuracy. The results show that the values for AUC change significantly when adding or deleting the first feature.

5.6.1: Summary on Biomedical Document Data

<table>
<thead>
<tr>
<th></th>
<th>-50</th>
<th>-40</th>
<th>-30</th>
<th>-20</th>
<th>-10</th>
<th>0</th>
<th>+10</th>
<th>+20</th>
<th>+30</th>
<th>+40</th>
<th>+50</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>-0.0478</td>
<td>-0.0429</td>
<td>-0.044</td>
<td>-0.0401</td>
<td>-0.0198</td>
<td>0.2643</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>HP</td>
<td>-0.0264</td>
<td>-0.0237</td>
<td>-0.0201</td>
<td>-0.0146</td>
<td>-0.0088</td>
<td>0.8159</td>
<td>-0.0001</td>
<td>-0.0001</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>REP</td>
<td>-0.2709</td>
<td>-0.2385</td>
<td>-0.2248</td>
<td>-0.1932</td>
<td>-0.1239</td>
<td>0.7772</td>
<td>-0.0081</td>
<td>-0.0128</td>
<td>-0.0154</td>
<td>-0.0173</td>
<td>-0.0189</td>
</tr>
<tr>
<td></td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>-50</th>
<th>-40</th>
<th>-30</th>
<th>-20</th>
<th>-10</th>
<th>0</th>
<th>+10</th>
<th>+20</th>
<th>+30</th>
<th>+40</th>
<th>+50</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>-0.0479</td>
<td>-0.0426</td>
<td>-0.0399</td>
<td>-0.0316</td>
<td>-0.0165</td>
<td>0.8566</td>
<td>0.0002</td>
<td>0.0007</td>
<td>0.0008</td>
<td>0.0009</td>
<td>0.0012</td>
</tr>
<tr>
<td></td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
</tr>
<tr>
<td>J48</td>
<td>-0.1616</td>
<td>-0.1393</td>
<td>-0.1341</td>
<td>-0.1242</td>
<td>-0.0739</td>
<td>0.8401</td>
<td>-0.0041</td>
<td>-0.0046</td>
<td>-0.0055</td>
<td>-0.0058</td>
<td>-0.0063</td>
</tr>
<tr>
<td></td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
</tr>
<tr>
<td>SMO</td>
<td>-0.0544</td>
<td>-0.0511</td>
<td>-0.0462</td>
<td>-0.039</td>
<td>-0.0261</td>
<td>0.8651</td>
<td>-0.0041</td>
<td>-0.0107</td>
<td>-0.0195</td>
<td>-0.024</td>
<td>-0.0335</td>
</tr>
<tr>
<td></td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
</tr>
<tr>
<td>IBk</td>
<td>-0.0113</td>
<td>-0.0102</td>
<td>-0.014</td>
<td>-0.0129</td>
<td>-0.0093</td>
<td>0.5742</td>
<td>-0.0112</td>
<td>-0.0165</td>
<td>-0.0247</td>
<td>-0.0347</td>
<td>-0.0534</td>
</tr>
<tr>
<td></td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>-50</th>
<th>-40</th>
<th>-30</th>
<th>-20</th>
<th>-10</th>
<th>0</th>
<th>+10</th>
<th>+20</th>
<th>+30</th>
<th>+40</th>
<th>+50</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAG</td>
<td>-0.181</td>
<td>-0.1706</td>
<td>-0.1522</td>
<td>-0.1223</td>
<td>-0.0753</td>
<td>0.8442</td>
<td>-0.0136</td>
<td>-0.0205</td>
<td>-0.0248</td>
<td>-0.0299</td>
<td>-0.0332</td>
</tr>
<tr>
<td></td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>1/1/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
</tr>
<tr>
<td>ADA</td>
<td>-0.194</td>
<td>-0.1869</td>
<td>-0.1616</td>
<td>-0.1204</td>
<td>-0.0734</td>
<td>0.8788</td>
<td>-0.0114</td>
<td>-0.0169</td>
<td>-0.0218</td>
<td>-0.0261</td>
<td>-0.0297</td>
</tr>
</tbody>
</table>
Table V-3 – The table above shows overall changes from weak, generic and strong, and ensemble classifiers’ original values that are listed under column 0. The values represent accuracy. Table V-3(A) shows accuracy changes in weak classifiers. Table V-3(B) shows accuracy changes in generic and strong classifiers. Table V-3(C) shows accuracy changes in ensemble classifiers. The negative numbers on top of column indicate the number of feature deletion. The positive numbers on top of column indicate the number of feature addition. Each cell that is under negative numbers or positive numbers has two smaller cells. The cell on top indicates the average change in value from the original value in specific classifier. The cell on bottom indicates how many datasets have better accuracy than original value (win), how many datasets have worse accuracy than original value (lose), and how many datasets have same accuracy than original value (tie). The format is expressed W/L/T.

Table V-3 shows overall changes from the classifiers’ original value after feature deletion and addition. The table V-3(A) shows that weak classifiers have decreased in accuracy after feature deletion. The table also shows that weak classifiers have remained unchanged in accuracy after feature addition. Table V-3(B) shows that accuracies for generic and strong classifiers have either decreased slightly or remained unchanged after feature deletion or addition. Table V-3(C) shows that ensemble classifiers have decreased in accuracy after feature deletion or addition. These observations are supported by the number with the format of win/lose/tie.

<table>
<thead>
<tr>
<th></th>
<th>0/2/0</th>
<th>0/2/0</th>
<th>0/2/0</th>
<th>0/2/0</th>
<th>0/2/0</th>
<th>1/1/0</th>
<th>1/1/0</th>
<th>1/1/0</th>
<th>1/1/0</th>
<th>1/1/0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF</td>
<td>-0.071</td>
<td>-0.066</td>
<td>-0.0558</td>
<td>-0.0528</td>
<td>-0.0311</td>
<td>0.7714</td>
<td>-0.0061</td>
<td>-0.0199</td>
<td>-0.0341</td>
<td>-0.0453</td>
</tr>
<tr>
<td>0/2/0</td>
<td></td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td></td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
<td>0/2/0</td>
</tr>
</tbody>
</table>
Figure V.21 – The bar graphs that show overall summary of win, lose and tie. The legend on the right side shows number of feature deletion or addition. Positive number indicates the number of feature addition. Negative number indicates the number of feature deletion. The values of y axis represent the total number of win, lose or tie. The values on x axis represent categorical value of win, lose or tie. There are three categorical values in a weak, generic and strong, or ensemble group. Figure V-21(A) and V-21(B) consist of summary of win, lose and tie relating to accuracy.
In general, the performance of weak, generic and strong, and ensemble classifiers in accuracy changes significantly when using feature deletion. The accuracy tends to be decreased after deleting or adding a feature. The graphs from figure V-21 support the fact that there is more loss than gain in accuracy.
CHAPTER 6

CONCLUSION

In this thesis, we study the sensitivity of classifier’s performance with respect to attributes. It is important to understand which classifier has the best performance after feature deletion or addition. Feature deletion involves the process of deleting a predominant feature that is determined by information gain and ReliefF. Feature addition involves the process of adding a redundant attribute to training and test set. If the dataset has few numbers of attributes in a dataset, the classifier’s performance may change after adding a redundant attribute. With a dataset consists of at least a thousand attributes, adding a redundant attribute might not have impact to the performance of a classifier. We have conducted the experiment that consists of deleting and adding five features to each dataset. There are 34 datasets that are being selected for this experiment. For high dimensional data such as gene expression and biomedical document, the experiment involves adding and deleting fifty features. When a dataset is deleted or added, a ten folds validation is performed in order to gather the result relating to classifier’s performance metrics. After gathering the results, we can use graphs to interpret the sensitivity of classifiers.
The study shows that the classifier is more sensitive to feature deletion than feature addition. The reason is that the process of feature deletion involves the removal of relevant feature and the process of feature addition involves inserting a redundant attribute, which is usually an irrelevant attribute.

With gene expression data, which dataset has at least over a thousand attributes, the performance of classifiers remains the same when deleting or adding a feature. For feature deletion, the reason why the performance of classifier remains the same is that it is difficult to determine importance of attribute with information gain and ReliefF. The difference between high ranking value and low ranking value is nearly the same. Therefore, removing a high ranking value may not affect the performance of classifiers. For feature addition, a dataset with large amount of irrelevant attributes creates a greater chance of inserting irrelevant feature to the dataset. Therefore, feature addition rarely affects the sensitivity of classifiers.

With biomedical document data, which has great number of attributes that exceeds over one thousand, the performance of classifiers decreases as the number feature deletion is increased. Unlike gene expression data, ReliefF and information gain can accurately determine the importance of attribute.

A new development of algorithm, that determines relevancy of gene expression data, is needed in order to determine the performance of classifier. This method should be capable of determining a set of attributes that have the highest relevancy in gene expression data. However, a separate study is needed in order to test this new method.
REFERENCES


[35]. I. Rish, An Empirical Study of the Naïve Bayes classifier, T.J. Watson Research Center.


