NOISE REMOVAL FROM ATTRIBUTE-GROUPS FOR CLASSIFICATION

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Angshu Kar

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Program:
Computer Science

November 2012

Fargo, North Dakota

North Dakota State University

Graduate School

Title

NOISE REMOVAL FROM ATTRIBUTE-GROUPS FOR CLASSIFICATION

By

Angshu Kar

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Anne Denton

Chair

Dr. Sanku Mallik

Dr. Kendall Nygard

Dr. Vasant Ubhaya

Approved by Department Chair: Dr. Kenneth Magel

11-20-2012

Date

Kenneth Magel

Signature

ABSTRACT

In this work, we present a novel algorithm that considers attributes from different experimental sources as separate groups for the purpose of classification. We remove noise from each of these groups, combine them, and then run the classifier on the grouped data. Examples are considered to be noise if they do not contribute to the prediction but, rather, degrade the quality of the classification result. As part of this work, we identify a measure that appropriately labels noise without knowledge of the class labels. Our method shows that the classification result is better when run on such filtered, grouped data than when run on the entire grouped data. In this work, we have considered time-series data because of their noisy nature. Our approach can be viewed as unsupervised feature-subset selection in grouped attributes and at the level of each instance individually.

ACKNOWLEDGEMENTS

The author expresses appreciation and deep gratitude to his adviser, Dr. Anne Denton, for her continued encouragement and invaluable advice, without which this work would not have been completed. It is not often that one finds an adviser who is so supportive and who always finds time for listening to the little problems which arise in the course of performing research. Special thanks are due to the committee members, Dr. Kendall Nygard, Dr. Vasanth Ubhaya, and Dr. Sanku Mallik, for their support. Special thanks go to my parents, brothers and family who always showed me the right direction. My thanks also go to my friends. Last, but not least, I would like to thank my wife, Arundhati, for her support during the past few years. Her encouragement until the end is what made this thesis possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER ONE. INTRODUCTION

1.1. Overview

Real-world data typically suffer from noise that may influence decisions. In this work, we present an algorithm to combine attributes from different experiments and to remove noise from these attribute-groups to improve the classification of experimental data. Time-series data are a common type of data used by data miners [Keogh and Kasetty (2002)]. In this thesis, we work with time-series data. A time series is a sequence of data points that are measured at successive time intervals. Time-series data mining is a very important problem in data-mining research [Yang and Wu (2006)]. Particularly, the problem of handling noise in time-series data is an open issue to tackle. Over the past decade, there has been a wave of interest in data-mining time series, with researchers attempting to index, cluster, classify, and mine association rules from increasing massive sources of data.

It has been seen that the data-mining algorithms become less effective in databases with a large number of attributes. One way to approach this problem is to reduce the amount of data before applying the mining process. In particular, pre-processing of feature selection, applied to the data before mining, has been shown to be promising because it can eliminate the irrelevant attributes that cause the mining tools to become inefficient and ineffective. At the same time, it can preserve the classification quality of the mining algorithm (determined by F-measure [Yang and Wu (2006)]). There are some feature-selection algorithms reported in the literature that have been used on time-series data [Weiss and Provost (2001), Mörchen (2003)]. Some of them are effective, but very costly in computational time (e.g., wrappers methods) [Yoon et al. (2005)], and others are fast, but less effective for the feature-selection task (e.g., filter methods) [van der

Walt and Barnard (2006)]. Specifically, wrapper methods, although effective in eliminating irrelevant and redundant attributes, are very slow because they apply the mining algorithm many times, changing the number of attributes during each execution as they follow some search-and-stop criteria. Filter methods are more efficient; they use some form of correlation measure between individual attributes and the class [Mörchen (2003)]. However, because they measure the relevance of each isolated attribute, they cannot detect if redundant attributes exist or if a combination of two (or more) attributes, apparently irrelevant when analyzed independently, are indeed relevant. In this thesis, we propose a method to consider attributes from different sources as grouped, which helps to distinguish between the relevant and the redundant sets of features. We use a measure to evaluate the relevance of an attribute block instead of a single attribute.

1.2. Motivation and Contribution

The goal of this classification algorithm is to classify *m* new examples, $E' = \{e_{n+1}, \ldots, e_{n+m}\}$, that are characterized by t_g time points per experiment (which are the attributes in this case and where g denotes the number of experiments; i.e., t_1 is the number of time points in experiment 1; t_2 is the number of time points in experiment 2; etc.). To generate this classifier, we have a set of *n* training samples, $E = \{e_1,\ldots,e_n\}$, characterized by t_g time points, $D_g = \{T_1,\ldots,T_{Dg}\}$, and the class label, $L = \{c_1,\ldots,c_n\}$, to which they belong. We know that not all time points (for each experiment) are necessarily beneficial for classification. Instead of using all available time points, we selectively choose experiments to use for classification. The main advantage here is that our selective choice of experiments results in reducing the noise to improve the classification results.

It is often thought that the relevant attributes are independent of each other and carry separate information. However, the independence assumption may not be true always. One

possible reason could be the fact that, for gene-expression experiments, the overall experimental condition remains the same and varies only in the time lapse after the beginning of the experiment. Therefore, the question that comes to our mind is whether some of the *t* attributes are redundant for learning the classification rule. To respond to this question, we come up with a new algorithm to identify those time points (that we call "noise") and to remove them from the data before running the classifier. In this research, we take each experiment; analyze whether it can be marked as noise for the given instance; and, if so, remove all the time points for that experiment before we train the classifier. Basically, we are always removing a given experiment when we conduct the comparison of data classification using Naïve Bayes classification with the original datasets and the ones with the noise blocked. The results demonstrate that our algorithm outperforms the detection rate when compared to classification done with the entire feature set.

1.3. Problem Statement and Approach

Time-series data are a common data type used by data miners. In this work, we are attempting to improve the classification of time-series data (created by combining multiple sources of information). Individual sources can be looked upon as a set of attributes (time points), each describing the same example with a common index such as the Gene ID. Specifically, our data consist of different time-series experiments conducted with the same set of genes to create a combined data source. The task is then to use the classifier (e.g., Bayesian) that predicts the class labels. Most classic machine-learning algorithms do not work well for time series. In particular, the high dimensionality and high-feature correlation present in time-series data have been viewed as an interesting research challenge. In addition to dimensionality and feature correlation, if high noise is considered, it complicates the classification. By "noise," we mean data that degrade the classification performance. To tackle these problems, we have come

up with a novel algorithm to filter part of the data (described as noise) before running the classification task. Our approach is described in the following schematic representation (Figure 1.1). Figure 1.1 shows how we combine data from different time-series experiments conducted on the same subject; e.g., in each dataset, we have data from a set of time-series experiments conducted on the same set of yeast genes. In Figure 1.1, Time Series Data 1 comes from Experiment 1 which is a time-series experiment conducted on a set of genes. Similarly, Time Series Data 2 and Time Series Data 3 come from independent experiments conducted on the same set of genes.

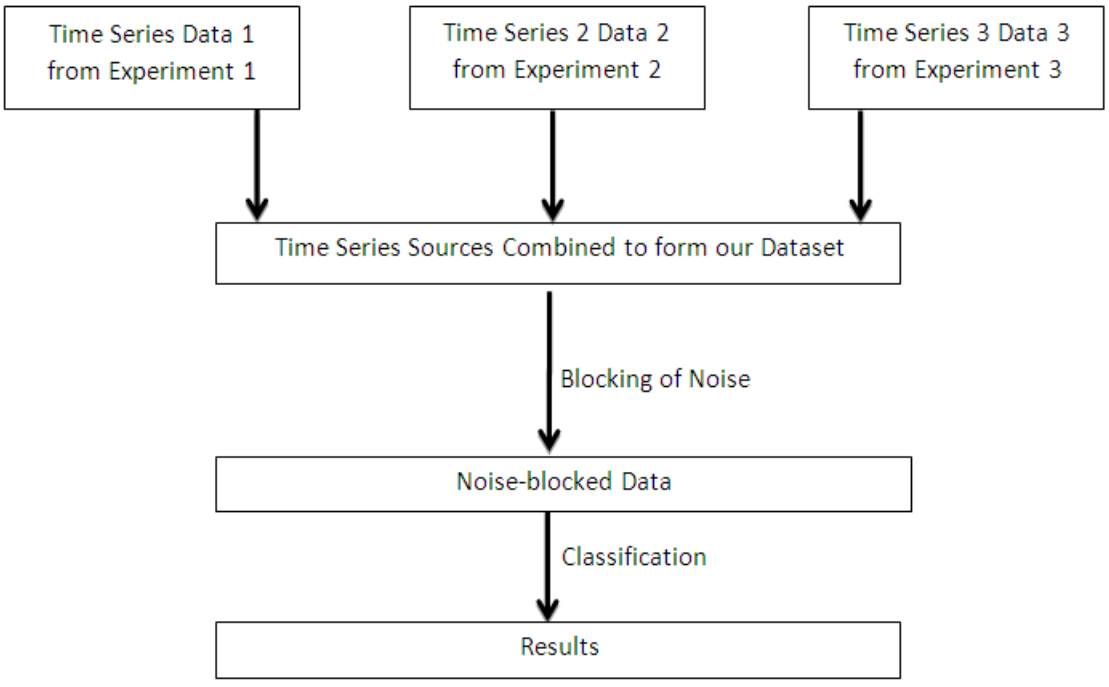

Figure 1.1. Schematic Representation of Our Approach to the Problem.

We combine all three of these data groups to form our dataset where we run our noise-blocking algorithm to remove the data points we consider noise. Then, we run the classifier on

the noise-blocked data. We compare the results of this classification with the findings for when the classifier is run on the entire dataset (without blocking noise).

Figure 1.2 represents how our final dataset looks. It is the dataset on which we run our algorithm. We see that g1, g2, and g3 are the common genes on which the three experiments (Experiment 1, Experiment 2, and Experiment 3) were conducted with the same set of yeast genes. The three experiments might have different numbers of time points. Each dataset has many time-series-based microarray readings, and each row signifies the record for a particular yeast gene. Each gene in the training data belongs to a particular class label (which represents a particular biological function). Here, the class labels are binary (i.e., They are either 0 or 1.), and the binary lables tell whether a particular gene participates in that function. Class label 0 signifies that the gene did not participate in cell-cycle regulation while 1 means it took part in that function. In Figure 1.2, we see that, given the genes of the combined dataset, we predict the class label for a new gene that is denoted by g100.

Yeast Gene	Experiment 1		Experiment 2			Experiment 3			Class Label
	t1	t2	t1	t2	t3	t1	t2	t3	
g1	x	x	x	x	x	x	x	x	0
g2	x	x	x	x	x	x	x	x	**1**
g3	x	x	x	x	x	x	x	x	**1**
: :									. .
g100	..	..	..	..	..	..	..	..	?

Figure 1.2. Schematic Representation of Combined Dataset.

As we see from Figure 1.1, our algorithm only considers the relevant part of the data (by blocking the redundant attributes) and improves the classification problem. The blocking technique is equivalent to treating the noise as missing values. Using this noise-blocked dataset,

we perform the Naïve Bayes classification and examine the results. The process is described in more detail in Figure. 1.3.

In Figure 1.3, we see that our dataset is composed of attributes from different experiments conducted on the same set of genes. For the sake of illustration, we mark the 100th gene (g_{100}) as a test gene (representing the test set). The rest of the genes constitute the set used to train the classifier (the training set). In Step 2, we merge the three datasets. After merging the attributes from the different experiments, our algorithm blocks the genes identified as noise and they are not used in training the classifier.

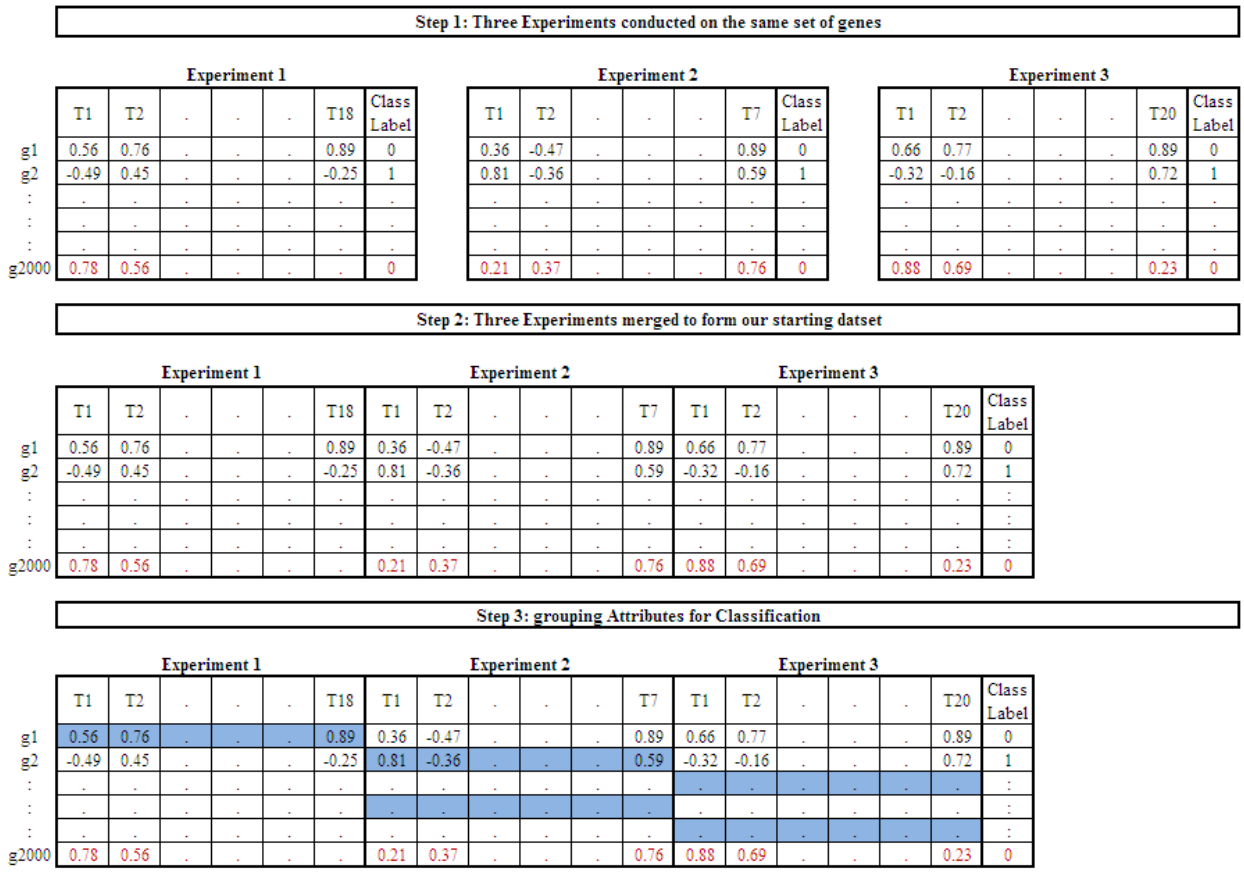

Figure 1.3. Schematic Representation of Attribute Grouping.

We see that we block all the attributes (Here attributes are time points.) for a particular gene for a given experiment (denoted by blue rows). The dataset we obtain from Step 3 in Figure 1.3 illustrates the novelty of how we group the attributes together for classification. As we see, this dataset contains less data than the one in Step 2. We show that the classification with the dataset in Step 3 is better than using the entire dataset (without grouping) as in Step 2, even though it has more instances.

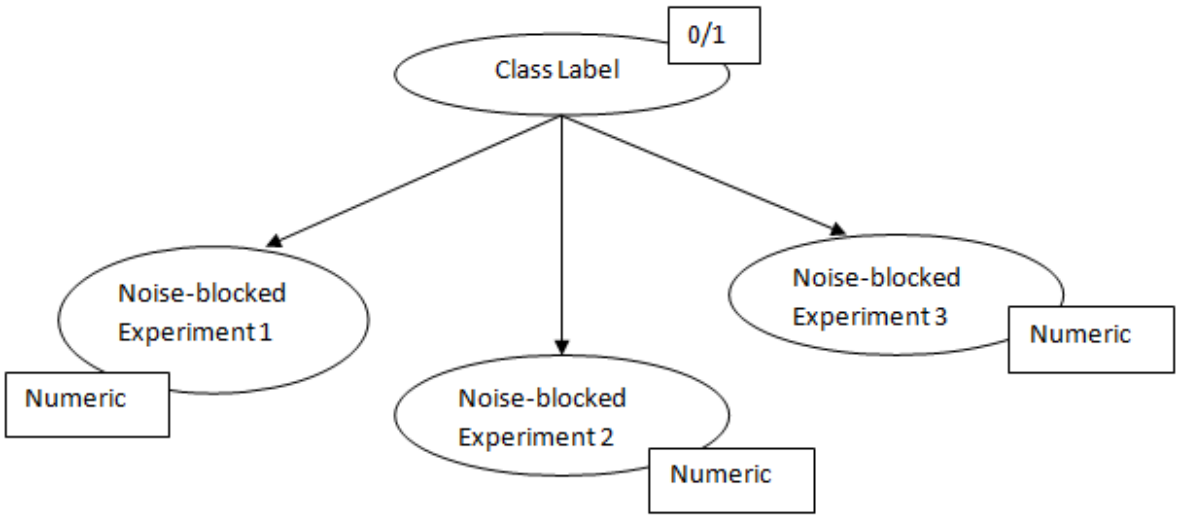

Figure 1.4. Schematic Representation of Naïve Bayes Classification.

In Figure 1.4, we see how our classifier works. We use the noise-blocked data from the experiments and run the classifier using them. We show that the results are better than when we run the classifier on the unblocked datasets.

1.4. Outline

The organization of this paper is as follows: Chapter 2 describes the method of Classification, with an emphasis on Naïve Bayes Classification along with its significance for the

current thesis. It then goes on to talk about the Rare Class Classification which we used. It also gives an overview of Feature Subset Selection. It describes the process of outlier detection. This chapter also defines the Time-Series Data upon which our experiments are based. Finally, it talks about how we evaluate our algorithm.

Chapter 3 describes Related Work. It discusses Time-Series Classification and Classification in the Presence of a Rarity because we deal with rare data in this thesis. We also review feature selection and noise detection as well as discuss the combination of multiple sources for data-mining purposes.

Chapter 4 discusses Our Approach. It defines experiments, the concept of noise, and how we identify noise. The chapter then introduces our proposed metric. We also discuss the evaluation procedure. The metrics for evaluating the results are defined, and their significance is discussed. The metrics used are mainly f-measure and balanced error rate.

Chapter 5 discusses the experimental procedure and the results and then talks about their significance. It compares the results given in the plots and talks about which technique gives the best results. Chapter 6 talks about possible future work in this field.

CHAPTER TWO. DEFINITIONS AND BACKGROUND

In this chapter, we deal with a brief description of each concept used in this work. We talk in details about classification here.

2.1. Classification

Classification is the process of predicting some category (called the class) of a dataset (measured on the test data or the test set) by building a model based on some predictor variables (or features, measured by selecting a subset of the data called the training data or the training set). For example, in evaluating a store location, the success of a store may be determined by its neighborhood quality and the weather of the locality. A company is interested in identifying localities with ideal neighborhood quality and weather conditions. A model based on the values of all available attributes (neighborhood quality and weather) is built to classify each item into a particular class (whether the locality is suitable for business or not). The goal of classification is to analyze the training set and to develop a description for each class using the attributes presented in the data. There are numerous classification algorithms, such as decision trees [Mitchell (1997)], Bayesian classifiers, Support Vector Machines [Mitchell (1997)], etc. In this work, we deal with the Naïve Bayes Classifier for binary classification. (i.e., The class can take only 2 values; we call them the target class and the major class which are also called positive and negative class, respectively.) We also deal with a particular type of classification, known as Rare Class Classification [Weiss (2004)], which is described in Section 2.1.2. The task of classification is described in Figure 2.1. We see that the classifier is trained using the training data and run on the test data. The outcomes of classification are the predicted class labels. In this task, we train the classifier using the instances g1, g2, g3,… while we test using instances g100, g101, g102,…, i.e., how well we can predict the class labels for g100, g101. and g102. We

compare the predicted class labels with existing class labels to evaluate the classifier's performance.

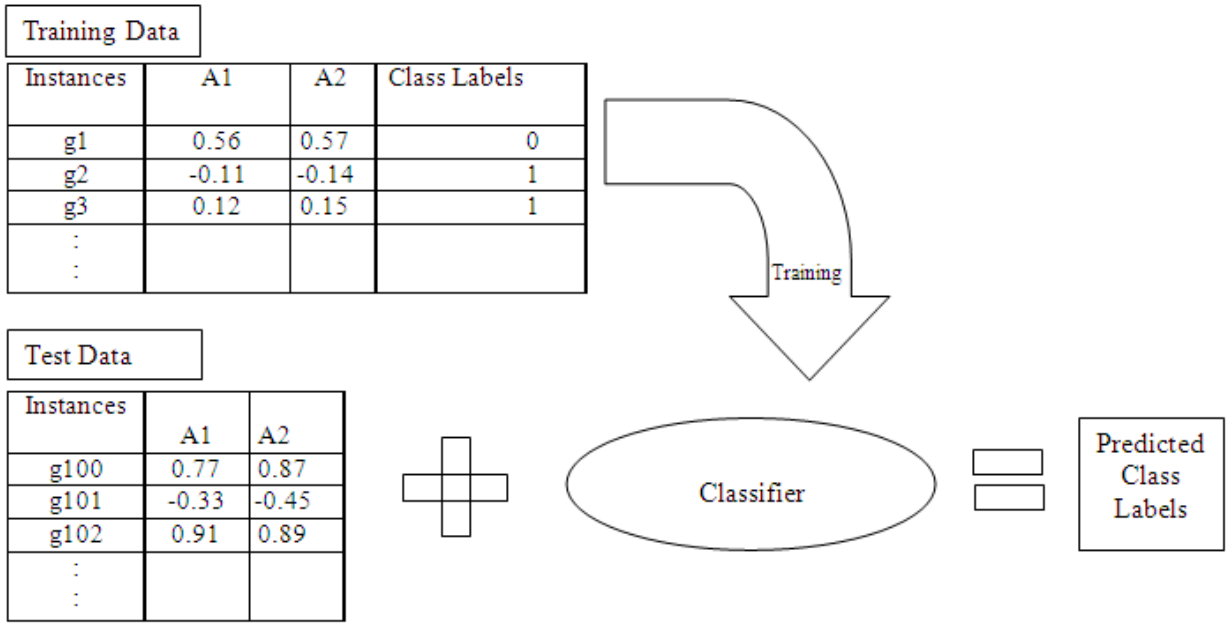

Figure 2.1. Schematic Representation of Classification (where A1 and A2 are Attributes).

2.1.1. Naïve Bayes Classification

The Naïve Bayes Classifier is the most popular among the probabilistic classifiers, and it is used in many applications, such as bioinformatics, insurance, medical, etc. The goal in Bayesian classification [Domingos and Pazzani (1997)] is to compute the probability of C_i for a given point, X_j, where $X_j = (X_j^1, X_j^2 \ldots X_j^d)$ is a point in d dimensions. Then, among all classes, we choose the one that has the largest probability and classify X_j as being of that class. That is, we predict the class of X_j as arg max $\{P(C_i|X_j)\}$. To compute $P(C_i|X_j)$, we simply invert the probability using the Bayes theorem:

$$P(C_i|X_j) = \frac{P(X_j \mid C_i) \times P(C_i)}{P(X_j)} \qquad (2.1)$$

In Naïve Bayes, we make the naive assumption that attributes are all independent, and under this independence assumption, $P(C_i|X_j)$ can be reduced to the equation below:

$$P(C_i|X_j) = \prod_{a=1}^{d} P(X_j^a \mid C_i), \qquad (2.2)$$

where X_j^a is the value of X_j in the a^{th} dimension.

For numeric data, we assume that each dimension is normally distributed, and we, thus, have to estimate σ and μ for each class, C_i, separately, directly from the data. Once the mean and variance per class, Cj, for each dimension a (namely, σ_j^a and μ_j^a), are known, we compute

$$P(X_j^a|C_i) = N(X_j^a \mid \sigma_j^a, \mu_j^a) = \frac{1}{\sqrt{2\pi}\sigma_i^a} e^{-\frac{(x_i^a - \mu_i^a)^2}{2(\sigma_i^a)^2}} \qquad (2.3)$$

2.1.2. Rare Class Classification

A challenging complication in classification arises when the number of target class members is far outnumbered by the number of the other class (major class members in the training set). This situation is the scenario in rare class classification [Kubat et al. (1997), Kubat and Matwin (1997), Japkowicz (2000), Cieslak and Chawla (2008), Kotsiantis et al. (2006)]. In these scenarios, the number for the target class is outnumbered by the other class. Figure 2.2 shows a rare-class classification scenario. In Figure 2.2, we see that the class being predicted is present in only 5% of the input data. Say that there are 100 genes in our input dataset and that we are predicting the function (class label) Lipid Metabolism; then, 5 of those 100 genes participate in Lipid Metabolism. The problem with this rarity is that the rare objects cannot be located under greedy-search heuristics [Weiss (2004)]; e.g., the *k-NN* (*Nearest Neighbor*) classifier may incorrectly classify many cases from the target class because the nearest neighbors of these cases

are examples belonging to the major class. In a situation where the noise is very high, the probability of the target class' nearest neighbor being noise is likely to be high. In this work, we deal with rarity as low as 3%.

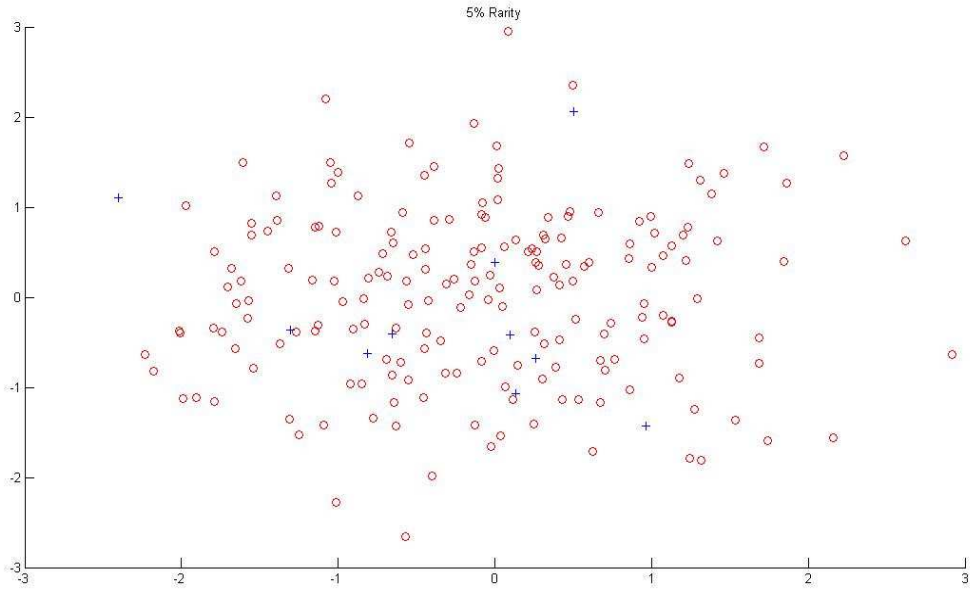

Figure 2.2. Graphical Representation of Rarity. Here, only 5% of the data points belong to our target class (+) while the rest belong to the major class.

2.2. Feature Subset Selection

Any data-mining task is usually preceded by data preprocessing. Feature-subset selection is such a process that selects a subset of original features. The optimality of the feature-subset selection algorithm is measured by an evaluation criterion. The evaluation criterion broadly falls into three categories:

- Filter model which relies on general characteristics of the data to evaluate feature subsets
- Wrapper model which requires one mining algorithm to evaluate and
- Hybrid model that uses a combination of both

2.3. Noise/Outlier Detection

In data mining, there are certain objects that are irrelevant or weakly relevant to a particular data-analysis method [Xiong et al. (2006)]. In this thesis, we call them noise and try to identify them as observations that deviate greatly from the other observations with respect to some measure. This noise is then removed and the remaining data are evaluated on classification task. (Here, Naïve Bayes classification is used.) The amount of noise to be removed is taken as a user parameter. This work deals with experimentation that has a wide range of noise levels, ranging from 10% to as much as 90% of the experimental data.

2.4. Time-Series Data

Time-series data are a series of observations over a period of time. Mathematically, each observation is recorded at a specific time, *t*. A time-series is given by Yoon et al. (2005):

$$t_{x,i} \; ; \; [i=1,\ldots,N; \; x=1,\ldots,M] \qquad (2.4)$$

and is a set of observation made serially through time, where *i* is the index of the measurement at time point *t* and *x* is the sample or the instance index. A time series is usually represented by the $M \times N$ matrix, where *M* is the number of instances and *N* is the number of observations. In this work, we will only be dealing with numeric time-series data. We will drop the term *x* according to convenience (and without loss of practicality), and in that case, it will mean that the series is for any given instance. An example of 3 × 7 time series is shown in Figure 2.3. We see that there are three time series and that the observations (experiments) were recorded at seven time points (t1, t2, … , t7). The value of each time series is plotted along the Y-axis.

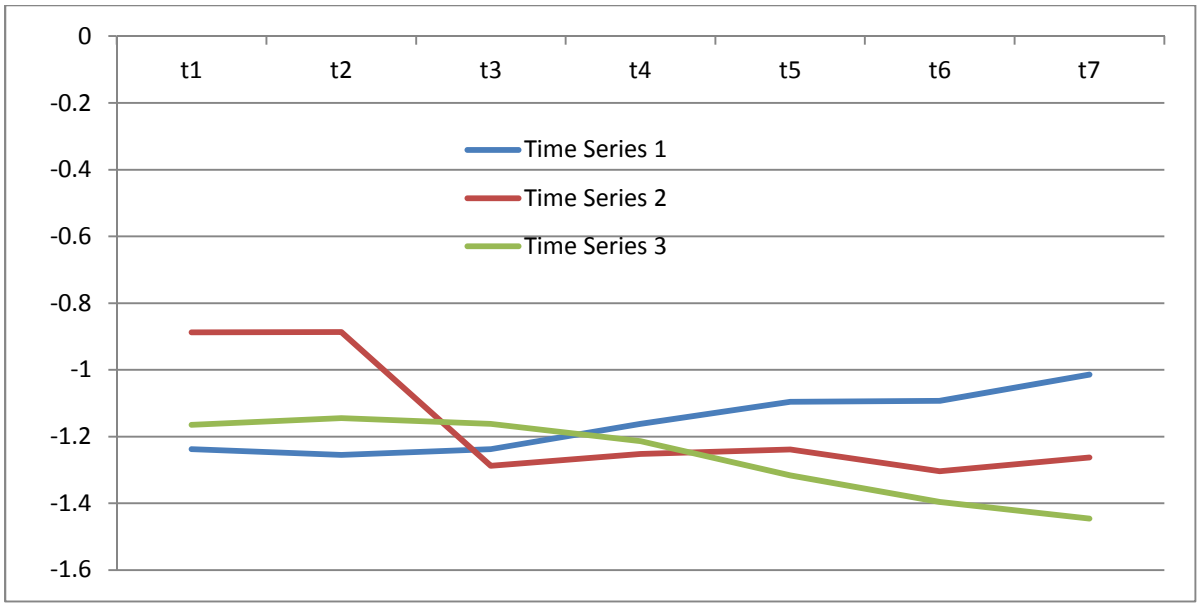

Figure 2.3. A Time Series. The figure shows three time series each having seven time points.

2.4.1. Random-Walk Time Series

A time series, t_i, is a random walk [Denton (2005)] if

$$t_i = t_{i-1} + e_t \quad , \qquad (2.5)$$

where e_t is called the white-noise time series (dependent on the gene) and $E\{e_t\} = 0$, $\text{var}\{e_t\} = \sigma^2$, for all t, $\text{cov}\{e_t, e_s\} = 0$, for all $t \neq s$. By white-noise time series, we mean $\{e_t\}$ is a normally distributed sequence of values corresponding to time t.

2.5. Evaluation

A popular way to evaluate the performance of classifiers is based on the confusion-matrix analysis [Provost et al. (1998)]. The comparison parameters used here are mostly dependent on four values for each experimental result. These values are true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The explanation of each is given as follows.

- TP = the number of times when the prediction of a gene expression in the training set is 1 while expression of the same gene in the testing set is 1

- TN = the number of times when the prediction of a gene expression in the training set is 0 while expression of the same gene in the testing set is 0
- FP = the number of times when the prediction of a gene expression in the training set is 0 while expression of the same gene in the testing set is 1
- FN = the number of times when the prediction of a gene expression in the training set is 1 while the expression of the same gene in the testing set is 0.

Table 2.1 illustrates a confusion matrix for our binary class problem having target class and major class values. We use this table to compare all our results while running to various classifiers on various datasets. The tables presented in this chapter will tell us, at a glance, how many true positives and true negatives are predicted by each classifier. Using the data in the tables, we shall find accuracy, specificity, sensitivity, and precision for the experiments.

From such a matrix, a large number of widely used metrics are derived to measure the performance of learning systems, such as

error rate, defined as

$$\frac{FP + FN}{TP + FN + FP + TN}$$

and accuracy (1-error rate), defined as

$$\frac{TP + TN}{TP + FN + FP + TN}$$

Table 2.1. Confusion Matrix for a Binary Problem

	Predicted Target Class	Predicted Major Class
Actual Target Class	True Positive (TP)	False Positive (FP)
Actual Major Class	False Negative (FN)	True Negative (TN)

However, when there is a huge amount of noise, the use of such measures might lead to misleading conclusions. In our scenario, noise is defined as the examples that are degrading the classification task, and we are classifying data which have as low as 3% rarity. Now, for example, it is easy to say that the classifier has an accuracy of 97% (or an error rate of 3%) in a scenario where the major-class proportion corresponds to 97% of the examples by simply predicting every new example as belonging to the major class. However, predicting the rare class is of primary interest in this problem scenario. Let us take the example of a scenario for predicting 3% of the cancer patients. A cancer patient diagnosed as healthy might be a fatal error while a healthy patient diagnosed as having cancer is considered a much less serious mistake. We use two evaluation metrics in this problem scenario: F-Measure (FM) and Balanced Error Rate (BER).

2.5.1. F-Measure (FM)

This metric, denoted by *F*, is used widely by the information retrieval community [Joshi (2002)]. It is defined with respect to a given class. The general form of *F* with respect to a class, c, is as follows:

$$F_\lambda = \frac{1}{\lambda \frac{1}{R} + (1-\lambda)\frac{1}{P}}, \quad 0 \le \lambda \le 1 \quad , \qquad (2.6)$$

where P is the precision ($P = \frac{TP}{TP+FP}$), R is the recall ($R = \frac{TP}{TP+FN}$), and λ is a parameter. We are interested in the classifier performance for the both normal and rare classes [Joshi (2002)]; hence, we use F_λ as the metric. We choose the value of $\lambda = 0.5$, giving recall and precision equal weights. Henceforth, we refer F_λ simply as $FM = 2*R*P / (R + P)$. Note, that this expression for FM becomes the harmonic mean of R and P.

2.5.2. Balanced Error Rate (BER)

This statistic [Chen and Wasikowski (2008)] looks at the performance of the classifier on both classes. It is defined as the average of the error rates for the two classes. If the classes are balanced, the BER is equal to the global error rate. Global error rate is commonly used for rare-class classification. The statistic is given by

$$BER = \frac{1}{2}\left(\frac{FP}{FP+TP} + \frac{FN}{FN + TN}\right) \qquad (2.7)$$

CHAPTER THREE. RELATED WORK

3.1. Time-Series Classification

Several machine-learning approaches have been developed [Keogh et al. (2003), Roddick et al. (2002)] to solve the time-series classification problem. We are also aware [Verleysen and Francois (2005)] how high-dimensional time series can be difficult to use for classification. Geurts (2001) has shown an algorithm to combine local patterns to improve classification.

3.2. Classification in the Presence of a Rarity

When we classify data with one or more classes being rare, it is called a rare-class classification or a class-imbalance problem. There has been extensive research done in this domain. Weiss and Provost (2001) have shown that there are two basic methods for dealing with class imbalance: i) under-sampling, where the major class elements are eliminated, and ii) over-sampling which multiplies the values in the minority class. Kubat et al. (1997) also use a novel technique of one-sided sampling while Liu et al. (2006) use a sequential under-sampling. Drummond and Holte (2003) have shown that under-sampling is better than over-sampling. Weiss (2004) shows different types of rarity and how noise affects rarity. Also, different papers [Al-Shahib et al. (2006), Chen et al. (2008)] describe how feature selection can help to improve classifier performance with a class-imbalance scenario.

3.3. Feature-Subset Selection

Feature-subset selection is a crucial step in classification. Several workers [Hall and Holmes (2003), Guyon and Elisseeff (2003), Blum and Langley (1997), etc.] have talked about different attribute-selection techniques and the two basic categories: wrapper and filter. We see in Bo and Jonassen (2002) how gene pairs are selected and evaluated as well as how gene sets

are formed for classification. There are also rough set-based feature selection algorithms which uses attribute dependency [Han et al. (2005)]. There are some unsupervised feature-selection algorithms [Varshavsky et al. (2006)] that use the variance of data collected for each feature. There are papers [Mitra et al. (2002), Yoon et al. (2005)] that also use hill-climbing approach and feature similarity for feature-selection. Hall (2000) also uses correlation between attributes for feature selection. Ding and Peng (2003) use mutual information to select the feature subset. There has been work on feature selection in microarray data [Xing et al. (2001)] using a series of filters.

3.4. Noise Detection/Minority Detection/Outlier Detection

The extensive survey by Hodge and Austin (2004) on outlier detection talks about three basic approaches to outlier detection: unsupervised clustering, where we find outliers with no prior knowledge of the data; supervised classification, where we have pre-labeled data and model the outliers; and semi-supervised, where there is a mixture of the other two methods. Keogh et al. (2005) find time-series discords which are subsequences of a longer time series and are maximally different from all the rest of the time-series subsequences. Chandola et al. (2009) describe a window-based technique to find discords in time series. Ando (2007) talks about finding atypical objects in the dataset using information theoretic approaches. Deng et al. (2004) find informative genes based on a non-parametric rank-sum test method while Chen et al. (2007) introduce a method which ranks irrelevant features which have little effect on classification under uniform noise. Cheng et al. (2009) do random walk-based anomaly detection in graphs. There are certain papers [Muthukrishnan et al. (2004)] that mine values (deviants) where removal leads to an improved, compressed representation of the remaining items while some other papers bring in the concept of time-series subsequences (shapelets) [Ye and Keogh (2009)]

that maximally represent a class. There has also been research on finding unusual patterns in time series [Keogh et al. (2002)] based on suffix trees. We see from previous research [Zhu and Wu (2004), Cheng et al. (2009)] how noisy attributes can affect classification performance. Cheng et al. (2009) introduce a graph-based local anomaly detection technique in time-series data using dependence among variables.

3.5. Naïve Bayes Classification

The Naïve Bayes classifier is a remarkably successful, yet simple, classifier [Rish et al. (2001)]. It assumes that attributes are statistically independent. Although it is unrealistic [Jakulin and Bratko (2003)], it works well in most classification scenarios. Keller et al. (2000) describe how effective the Naïve Bayes classifier is on DNA expression data after feature-subset selection.

3.6. Mixing Multiple Sources

In the works of Costa et al. (2002), we see that, when multiple time-series, gene-expression data are combined, it results in better clustering. Workers such as Rish et al. (2001) and Kundaje et al. (2005) have shown how time-series data and motif data can be combined to obtain better clustering results. This idea has also been used in the classification world (e.g., in classifier fusion). Multiple data sources for classification, where the different data sources involve the same set of genes, have been used before. In this work, we are combine those data sources that most help with classification.

CHAPTER FOUR. OUR APPROACH

4.1. Overview

In our data, the columns of the data matrix correspond to time points (or attributes) along some biological process. Intuitively, there will be some homogeneity along the attributes. As a part of data preprocessing, we have removed the time series which deviate from the other time series in that group based on our metric.

4.2. Concept of Groups or Experiments in Data

We introduce the concept of groups or experiments in our data related to the concept used by Costa et al. (2002). A group or experiment (henceforth, used interchangeably) is a subset of the attributes that is generated from a related source. A group of experiments is further illustrated in Table 4.1.

Table 4.1. Concept of Experiments

	Experiment 1			Experiment 2				Experiment 3				
	t1	t2	t3	t1	t2	t3	t4	t1	t2	t3	t4	t5
Instance 1												
Instance 2												
.....												

For example, we see in Table 4.1 that each instance consists of 12 time points and 3 groups. Experiment 1 has 3 time points denoted by t_1-t_3; Experiment 2 has 4 time points denoted by t_1-t_4; and Experiment 3 has 5 time points, t_1-t_5. Thus, we can say that each instance is described by a vector of experiments and that each experiment is a collection of attributes. In this work, we analyze each group independently to detect noise.

4.3. Noise Identification

The basic data preprocessing step we used here is called first-order differencing. The values of the first-order difference for a series (of N attributes*)*, t_1, t_2,t_3 ... t_N, are given by a new series

t'_1, t'_2 ... t'_{N-1},

where $t'_1 = t_2 - t_1$; $t'_2 = t_3 - t_2$; ... $t'_{N-1} = t'_N - t'_{N-1}$.

The operation of getting $t'_i = t_i - t_{i-1}$ is called the first difference.

4.3.1. Our Metric

The statistic that we use to measure the relevance of an instance, *x*, in the time-series data is the normalized sum squares of first differences (SSFD) and is given by

$$SSFD_x = \frac{1}{(n-1)} \sum_{i=1}^{n} \left(t_{x,i} - t_{x,i-1}\right)^2 = \frac{1}{(n-1)} \sum_{i=1}^{n} t'^{\,2}_{x,i} \qquad (4.1)$$

where n is the number of time points in the group under consideration. The main motivation behind using this metric is identifying highly fluctuating time series in the group. If the SSFD value is high, we can conclude that the series is highly fluctuating. Mathematically, from equation (2), we have

$$\left(t_i - t_{i-1}\right)^2 = e_t^2$$

$$\Rightarrow \sum_n \left(t_i - t_{i-1}\right)^2 = \sum_n e_t^2$$

$$\Rightarrow \sum_n \left(t_i - t_{i-1}\right)^2 = \sum_n e_t^2 - n\mu^2 \text{, (since, } E\{e_t\} = 0 \text{ or } \mu = 0\text{)}$$

$$\Rightarrow \frac{1}{n-1} \sum_n \left(t_i - t_{i-1}\right)^2 = \frac{1}{n-1} \sum_n e_t^2 - \frac{n}{n-1}\mu^2$$

$$\Rightarrow \frac{1}{n-1}\sum_{n}\left(t_i - t_{i-1}\right)^2 = \text{var}\{\varepsilon_t\} = \sigma^2 \qquad (4.2)$$

Hence, we see that value of SSFD basically denotes the variance of the white noise (which is gene dependent). Minimizing SSFD basically means reducing the variance in the subsequence for which we are calculating the SSFD.

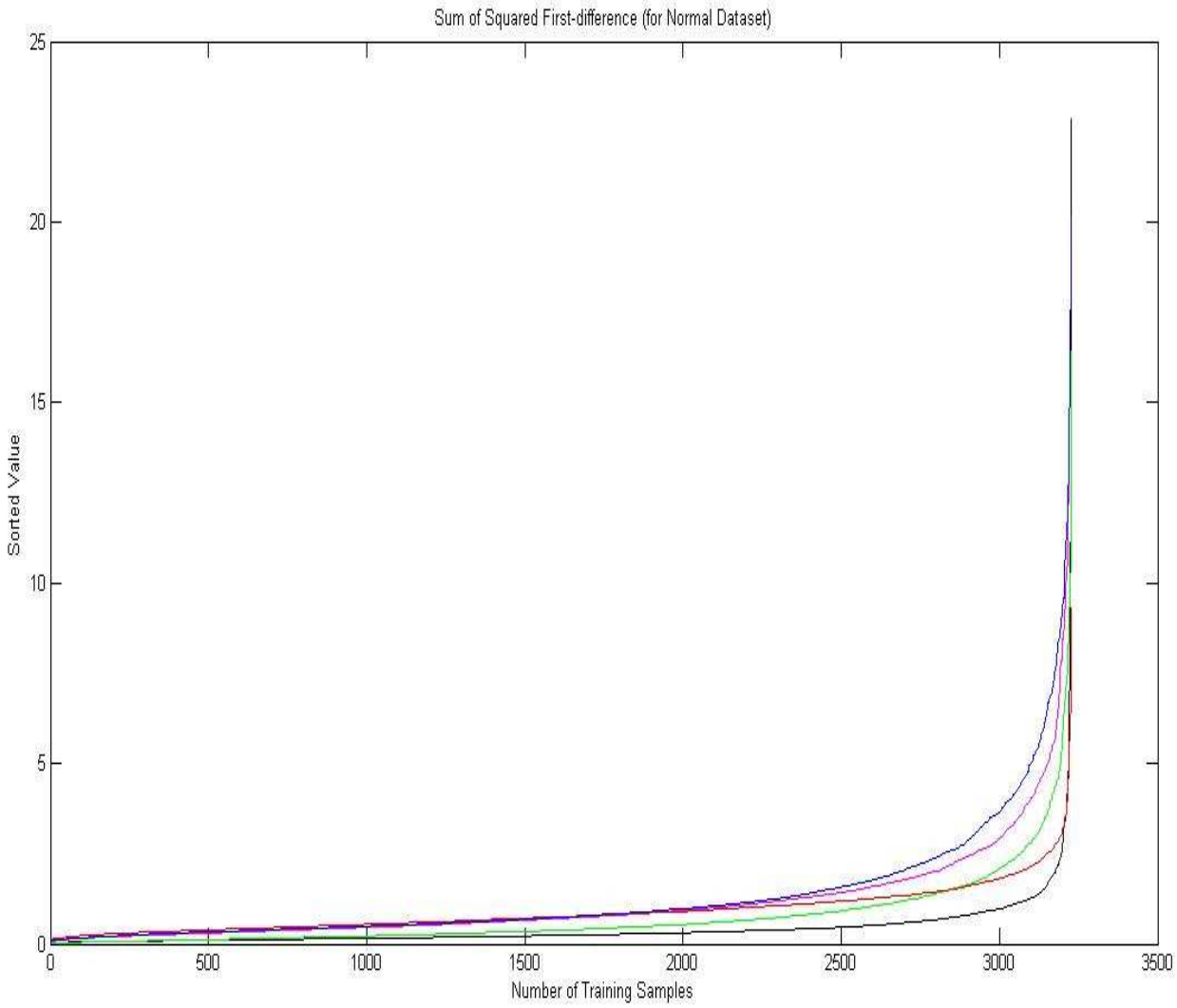

Figure 4.1. Sorted SSFD Values.

Figure 4.1 shows the sorted SSFD values. We calculate the SSFD for each gene in each experiment and then plot the sorted SSFD values (Y-axis) against the number of genes (X-axis)

used to calculate the value. We see that, for high SSFD, a small change in the number of training genes brings a huge change in the SSFD value. From the figure, we can say that, if we want to eliminate about 1,200 noisy samples for a given group, we need to choose the cutoff SSFD value corresponding to 2,000 which will be around 1.5. We use this idea to block experiments for individual genes.

We see that the measure is helping us detect time series (for a given instance) which are different from other instances in a given experiment. The time-series which are different from the other instances are defined as noise because they deviate significantly from the normal behavior displayed by the group [Cheng et al. (2009)]. A representation of the measure is shown in Figure 4.2. We have generated some examples to show how the SSFD measure helps to differentiate high-fluctuating time series from the rest. We see that the noise detected by the SSFD method is behaving differently from the other time series in the figure; i.e., time series 4, 5, and 6 have highly fluctuating behavior when compared to time series 1, 2, 3, and 7.

Also, we can assume that the groups are uncorrelated because they come from completely different experimental sources.

Now, we use this SSFD measure to detect noise in an unsupervised method. Again, under this method, we use this measure in two types of datasets: Rare and Normal. The goal of our research is not to show anything specific to rare class classification. We did this division of data to see how our algorithm performs under rare class scenario i.e. when the number of examples for the target class is low. We see that even in rare class scenario our algorithm outperforms the classification results that we get when we classify the entire dataset (versus the classification results that we obtain using the noise-blocked dataset that we get using our noise detection algorithm).

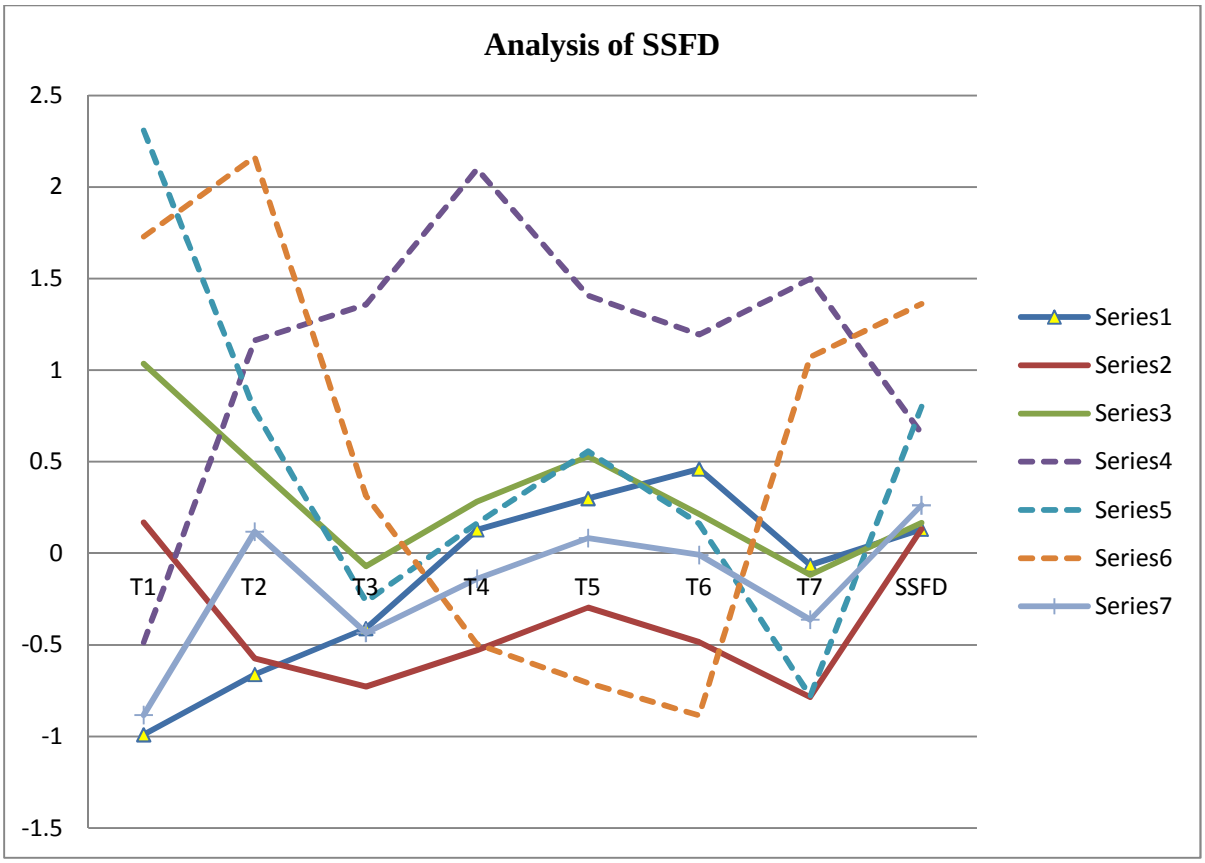

Figure 4.2. Noise Detection Based on SSFD. (Series 4, 5, and 6 are noise represented by broken lines.)

4.3.2. Our Method

In our algorithm, we use an unsupervised method for identifying noise. By unsupervised method, we mean we do not have any prior knowledge of the class distribution while identifying the noise; i.e., we do not know which instance belongs to the class of interest. We get the input dataset, detect the noise, and block them off. (Blocking off is the same as treating them as missing values.) Then, using this noise-blocked dataset, we perform the Naïve Bayes classification and examine the results. The steps of the algorithm are described in Table 4.2. (The program is provided in Appendix A.)

Table 4.2. Algorithm for Our Method

Data: Time-Series Data [m × n]

Parameter: Percentage, p, of noise to be eliminated per group

Result: The classification result (measured by F-Measure and the BER).

1. Z-Normalize the dataset along each time point (column-wise)
2. Calculate the SSFD of each instance per group for data
3. Based on p, find the cutoff value, c, above which everything is considered noise
4. Mark all instances above c as noise
5. Block the instances identified in Step 5
6. Run the classifier on these noise-blocked data

For illustration, we randomly choose a time point, say time point 2, from Experiment 1 and plot the time point against SSFD, as shown in Figure 4.3. There are 67% of the genes plotted in the figure which constitute the training set after the data are normalized. The blue hexagons in the figure indicate the target class that we want to predict. We notice that most of the genes are concentrated below SSFD = 0.5. Given that the user wants to eliminate 60% of the training instances (i.e., p=60%), we find that the corresponding SSFD cutoff value is 0.15. Hence, we eliminate all the instances where SSFD is greater than 0.15, as shown in Figure 4.4. This subset is used to train the classifier.

As we described in Figure 4.3, we want to eliminate 60% of the genes from that particular experiment (i.e., Experiment 1) and see from Figure 4.3 that the cutoff needs to be 0.15. We generate Figure 4.4 after eliminating (blocking) 60% of the genes. From Figure 4.4, we can see that any genes above 0.15 have been eliminated. We use these data for training the

classifier. From Figure 4.4, we see that the class labels (0 and 1) are much better separated than in Figure 4.3, which helps in classification.

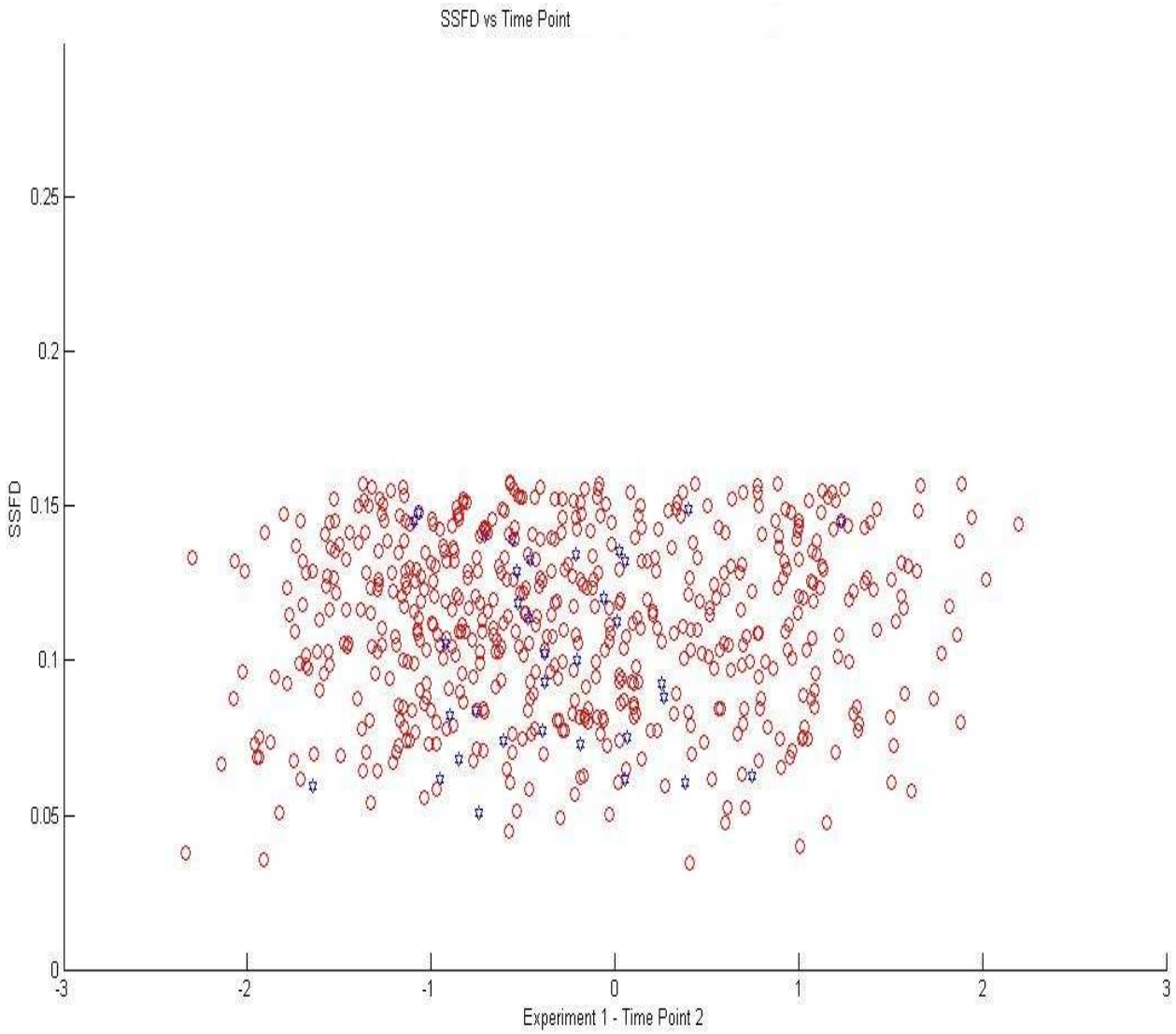

Figure 4.3. Data After Columnwise Z-Normalization.

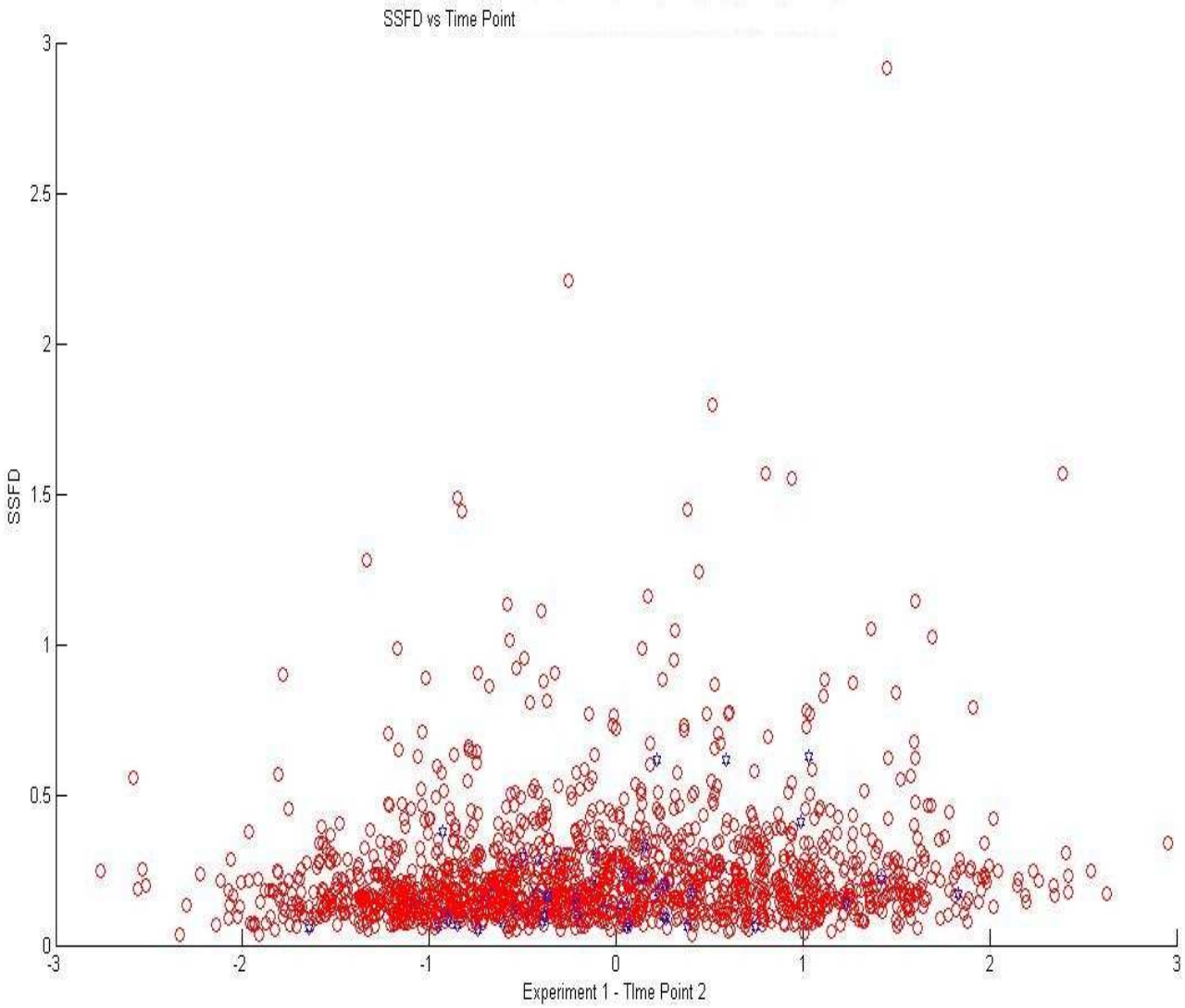

Figure 4.4. The Example Dataset (After Noise Blocking). The blue hexagons indicate the target class.

4.4. Classification

The Naïve Bayes method (implemented as a customized Naïve Bayes in Matlab) is used on the training set, described above, and the results are predicted on the test set. We use a threefold cross validation in this work.

4.5. Evaluation

We evaluate the performance of our algorithm by comparing the result of the Naïve Bayes classification on the entire dataset versus the dataset generated after removal of the

instances. As discussed earlier, we use the F-Measure and BER to evaluate our result which takes into account the class distribution. We see that our algorithm outperforms the result when run with all the features. The results are further discussed in Chapter 5. We show that our attribute-grouping algorithm appears to separate the two classes better, helping the classification.

CHAPTER FIVE. EXPERIMENTS AND RESULTS

5.1. Our Data

We used two datasets (Tables 5.2 and 5.3) in this work. We created them from the Stanford Microarray Database (SMD) [Hubble et al. (2009)]. We called them Dataset 1a and 1b. Time-series microarrays captured expression profiles of a gene at discrete time points. Microarray experiments are routinely performed to measure the expression of genes. Thousands of DNA samples are arrayed on glass slides and labeled cDNA or genomic DNA from control and experimental samples are competitively hybridized to the array. Images of the slides are then processed to produce a data file that contains dozens of values per spot for several thousands of spots. The main information for each spot is the ratio between the experimental and the control samples. We created the datasets by merging 5 time-series datasets for yeast (Saccharomyces Cerevisiae) available in SMD: cell cycle, nutrient effects, sporulation, starvation, and stress. Each dataset was generated by measuring the expression level of every gene at a sequence of time points. The number of time points used in each of them is shown in Table 5.1. We replaced the missing values with 0s and replaced multiple readings on the same gene by their means. The reason why these readings were assumed to be 0 is that, if the red by green ratio is assumed to be 1, the logarithmic value of red by green would then be 0. The expression pattern of a gene in each experiment is a vector where the i^{th} component is the expression level of the gene at the i^{th} time point. Thus, Dataset 1 is a matrix with a row for every gene and a column for every time point; it has 4,889 genes and 69 time points.

Table 5.1. Distribution of Time Points Among Groups in Dataset 1

Experiment or Group	Number of Time Points
Cell cycle	25
Nutrient effects	17
Sporulation	7
Starvation	10
Stress	10
	69

5.1.1. Class Labels for Dataset 1

We used the class labels from the Go-Slim data available from the Saccharomyces Genome Database (SGD) [Hong et al. (2008)]. Initially, we got 113 classes but filtered out the ones which had less than equal to 3% rarity. The final dataset had 44 classes.

5.1.2. Dataset 1a and Dataset 1b

For the purpose of analyzing the classification performance of our algorithm on the dataset with an extremely rare target class, we divided our Dataset 1 into two datasets: Dataset 1a and Dataset 1b (shown in Figures 5.1 and 5.2, respectively). We created these two datasets with the goal of evaluating the algorithm under different experimental conditions. We created one dataset with 5% of the major class element, and we used the rest of the data to create our second dataset. Dataset 1a had 95% or more of the major class elements while dataset 1b had 70-95% of the major class elements. Hence, Dataset 1a is the rarest. Both the datasets, along with class labels, are shown in Tables 5.2 and 5.3. In Figures 5.1 and 5.2, we plot two random time points from Experiment 2 (2 and 10) and Experiment 3 (3 and 7), respectively, to show the data distribution. From Table 5.2, we see that the rarity (i.e., the percentage of the target class) ranges from 3.17% to 4.93%. From Table 5.3, we see that the rarity ranges from 5.42% to 31.60%.

Table 5.2. Dataset 1a

Class ID	Class	Instances	From Table 5.1		Instances of the Target Class	Percentage of Target Class
			Number of Experiments or Groups	Number Of Time Points		
1	Golgi apparatus	4889	5	69	155	3.17%
2	RNA binding	4889	5	69	158	3.23%
3	Enzyme regulator activity	4889	5	69	163	3.33%
4	Signal transduction	4889	5	69	170	3.48%
5	Carbohydrate metabolism	4889	5	69	180	3.68%
6	Molecular function	4889	5	69	181	3.70%
7	DNA binding	4889	5	69	184	3.76%
8	Amino acid and derivative metabolism	4889	5	69	186	3.80%
9	Oxidoreductase activity	4889	5	69	188	3.85%
10	Vacuole	4889	5	69	189	3.87%
11	Cytoskeleton	4889	5	69	193	3.95%
12	Morphogenesis	4889	5	69	199	4.07%
13	Plasma membrane	4889	5	69	201	4.11%
14	Cytoskeleton organization and biogenesis	4889	5	69	204	4.17%
15	Chromosome	4889	5	69	204	4.17%
16	Lipid metabolism	4889	5	69	212	4.34%
17	Nucleolus	4889	5	69	217	4.44%
18	Ribosome biogenesis and assembly	4889	5	69	230	4.70%
19	Ribosome	4889	5	69	230	4.70%
20	Mitochondrial envelope	4889	5	69	241	4.93%

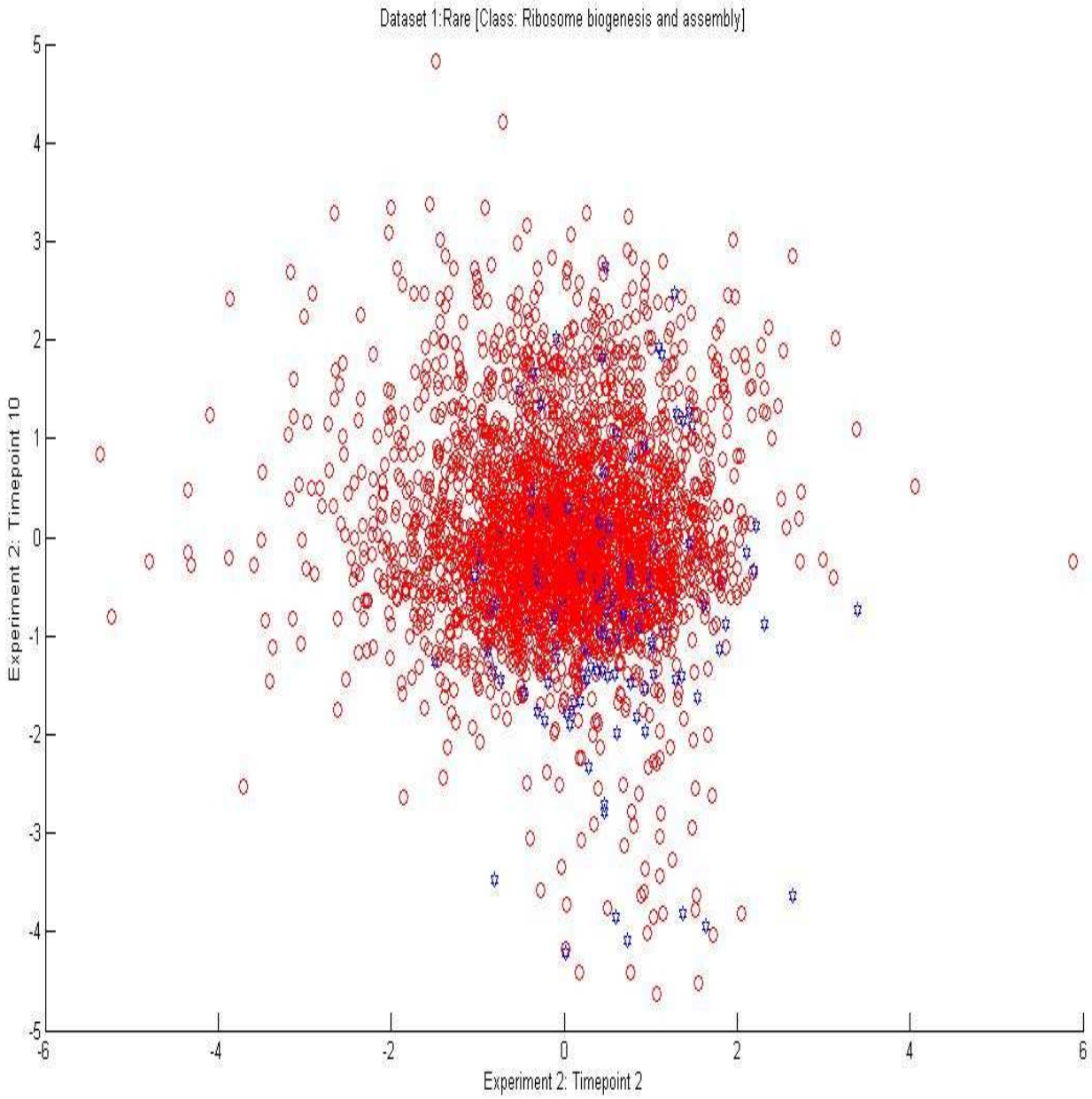

Figure 5.1. Dataset 1a (After Columnwise Z-Normalization). Blue hexagons denote target class.

Figure 5.1 represents the z-normalized data for Experiment 2. The blue hexagons denote the class to be predicted, or the target class (i.e., for the class Ribosome biogenesis and assembly), and it is plotted against a time point (here, time point 10).

Table 5.3. Dataset 1b

Class ID	Class	Instances	From Table 5.1		Instances of the Target Class	Percentage of Target Class
			Number Of Experiments or Groups	Number of Time Points		
1	Endomembrane system	4889	5	69	265	5.42%
2	Cellular component	4889	5	69	270	5.52%
3	Vesicle-mediated transport	4889	5	69	286	5.85%
4	Cell cycle	4889	5	69	293	5.99%
5	Transcription regulator activity	4889	5	69	299	6.12%
6	Transporter activity	4889	5	69	299	6.12%
7	Structural molecule activity	4889	5	69	299	6.12%
8	Endoplasmic reticulum	4889	5	69	323	6.61%
9	Transferase activity	4889	5	69	334	6.83%
10	Response to stress	4889	5	69	391	8.00%
11	Protein binding	4889	5	69	391	8.00%
12	Hydrolase activity	4889	5	69	413	8.45%
13	Membrane	4889	5	69	428	8.75%
14	Protein biosynthesis	4889	5	69	429	8.77%
15	RNA metabolism	4889	5	69	448	9.16%
16	DNA metabolism	4889	5	69	461	9.43%
17	Transcription	4889	5	69	480	9.82%
18	Protein modification	4889	5	69	507	10.37%
19	Transport	4889	5	69	594	12.15%
20	Organelle organization	4889	5	69	641	13.11%
21	Biological_process	4889	5	69	675	13.81%
22	Mitochondrion	4889	5	69	759	15.52%
23	Cytoplasm	4889	5	69	1536	31.42%
24	Nucleus	4889	5	69	1545	31.60%

We observe from Figure 5.1 that the target class is extremely rare. From Table 5.2, we see that it is Class ID = 18 and has a 4.70% presence in the entire dataset. From Table 5.3, we

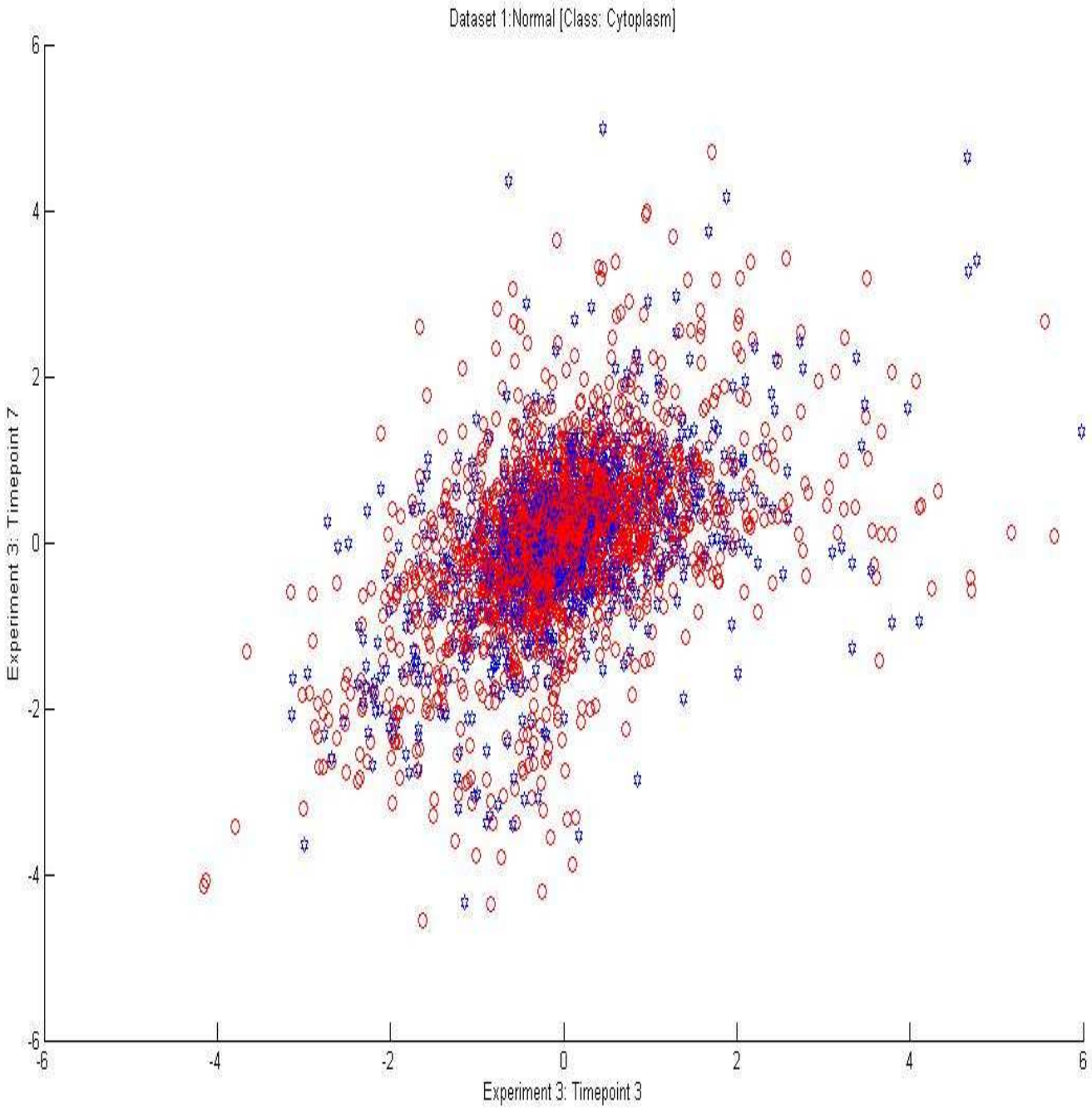

Figure 5.2. Dataset 1b (After Columnwise Z-Normalization). Blue hexagons denote target class.

see that the dataset has rare classes such as the Endomembrane system with 5.42% rarity while it has classes such as Nucleus which is in almost one-third (31.60%) of the data.

Figure 5.2 represents the z-normalized data for Experiment 3. The blue hexagons denote the class to be predicted (i.e., for the class Cytoplasm), and one time point in the Cytoplasm class (time point 7) is plotted against another time point (here, time point 3) in the same class. We observe from Figure 5.2 that the class to be predicted is not rare and that it is in over 30% of the total genes.

5.2. Data Preprocessing

The initial step before the time series are mined is to normalize them so that similar patterns can be effectively identified. In the raw data, there are usually large disparities in the expression level or the logarithmic ratio of red by green among the genes. For normalization, the bias (vertical shift) has been removed by subtracting the mean value of the time series per time point. Then, "rescaling" has been done by dividing the time series by their standard deviation, again per time point. The resulting time series before and after this normalization are depicted in Figure 5.1. The subtraction of the mean value, in conjunction with division by the standard deviation, is called z-normalization and is a necessary step for the identification of similar patterns. We divide the dataset into training data and test data in 2:1 ratio, respectively.

5.3. Experimental Results

The results of the three experiments using the two datasets (Table 5.2 and Table 5.3) are explained in the following sections. We show the results using our method. For a given p, we have the F-Measure and BER for all class labels in the dataset. We use the mean of the F-Measures and BERs for all those classes for comparison. In all the figures, we have plotted the change in mean F-Measure and mean BER for the noise-blocked dataset when compared to the

entire normalized dataset; i.e., the baseline performance is that of a Naïve Bayes classifier using all the features. The term "mean" denotes the average over all the classes. For plotting, we have used the following measure:

$$\frac{\text{Mean F-Measure for noise-blocked data} - \text{Mean F-Measure for entire data}}{\text{Mean F-Measure for entire data}}$$

Similarly, we calculated the BER change as follows:

$$\frac{\text{Mean BER for noise-blocked data} - \text{Mean BER for entire data}}{\text{Mean BER for entire data}}$$

These measures are plotted it in Figures 5.3 and 5.4.

We also see, for a fixed training-set size, to what extent different training-set class distributions affect classifier performance. We vary the training-set class distributions for all datasets as described in the following sections.

5.3.1. Results for Our Method

The classification results for Dataset 1a are shown in Figure 5.3. The figure shows that the best classification result is obtained at the level of blocking 20% of the noise. Obviously, it results in an improvement of over 1.5% in the F-Measure, and the BER has also decreased. We can notice that the error rate slowly rises as we eliminate more instances as noise (as they are information), and at 60% blocking, we have the F-Measure decreasing and BER increasing, which denotes underperformance in classification. Hence, the optimal percentage of noise to be eliminated for this dataset is 20%.

When we do classification on the Dataset 1b, we see that the best performance is when the percentage of noise eliminated is 40% (Figure 5.4). The improvement is because, at that level of blocking, we have more than a 3% increase in the F-Measure while the BER remains same. In

this algorithm, we increase p, the parameter that determines how many genes we want to eliminate per experiment, by increments of 10%; run the classifier; and record the result.

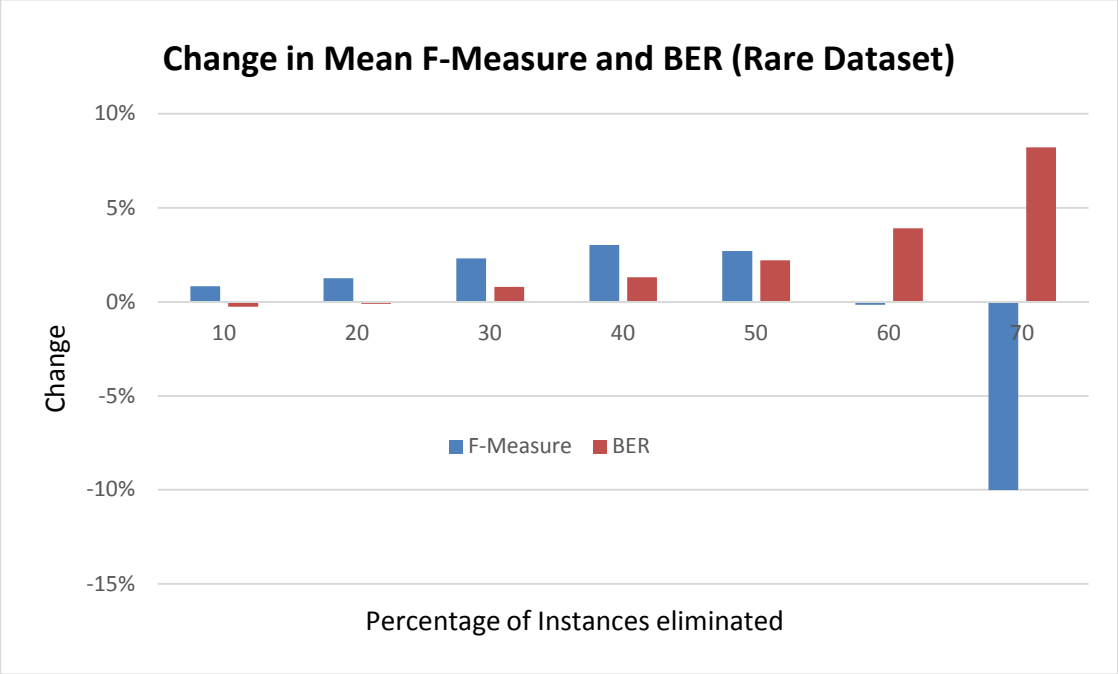

Figure 5.3. Classification Results for Dataset 1a.

What we expect is that the F-Measure will first increase with an increasing p, until reaching a maximum, and will then drop. This drop can be attributed to the fact that, at that point, we are eliminating information needed to train the classifier instead of just redundant attributes. In this example, we see from p=60% that the error rate starts and the F-Measure starts to decrease. Hence, we can say that the classifier performs best at 40%, i.e., when 40% of the genes have been eliminated per experiment.

5.4. Discussion

We address the issue of finding and eliminating noise in time-series data to improve Bayesian classification using groups of attributes. Our algorithm performs feature reduction on two datasets by eliminating noise from the groups. For each group, we found that there is an

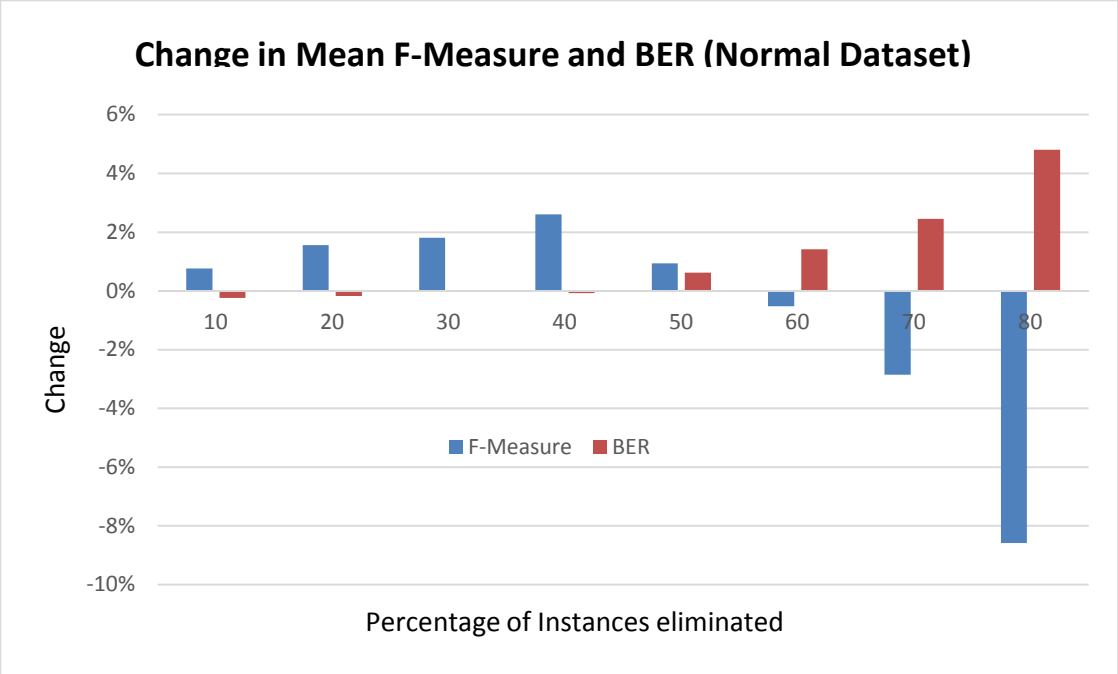

Figure 5.4. Classification Results for Dataset 1b.

optimal noise-blocking level where the grouping by our algorithm outperforms the classification results performed on the entire feature set. The success of our algorithm could be attributed to the ability to identify and block the widely varying changes, which helped the Bayesian classifier [Rish et al. (2001)]. If we apply our algorithm in the unsupervised manner, we can block as much as 40% of the training data per group and, yet, achieve equal or better classification results. We see that, for unsupervised learning, it is not appropriate simply to choose the entire dataset for training even with a rare-class classification scenario. We have shown that we don't need to use all the data to get good classification results. Also, we have shown a technique for using grouped data from different experiments for feature selection and classification. Especially in the biological domain, we believe that using groups of attributes in the feature-selection process may inspire new ways of modeling time-series learning.

CHAPTER SIX. CONCLUSION AND FUTURE WORK

We described an algorithm to select genes using groups of attributes to optimize the classification performance for time-series data. We compared the performance of the Naïve Bayes method when applied to the entire dataset with that when applied to the noise-blocked dataset. We tried our experiments on two different datasets and found similar results. Our algorithm was based on a preprocessing technique of first-order difference which likely reduces the dependency between adjacent attributes. Our motivation for performing the preprocessing technique was to improve the classification performance by discarding redundant features called noise.

Our results show that eliminating noise will likely enhance the classifier performance and that smaller subsets can be created. The smaller subsets are preferable as long as the BER does not increase significantly or the F-Measure does not fall much over 1%. In this research, we also provide insight about why one distribution might be better than another for training the classifier. We have also shown how combining different experiments (from different sources) can lead to affect learning. Our future work will focus on the following aspects:

- Apply and improve more existing classification algorithms, besides the Naïve Baye,s to the results of our algorithm to see whether the classifier can be improved.
- Try to get rid of parameter p, the percentage of genes to be eliminated from each experiment, in the algorithm. We can probably do that task by incorporating a search and finding the optimal value of p.
- We can attempt to develop novel classification algorithms by integrating this concept of noise elimination into the process of learning. Something such as a classifier will pick only non-noisy examples to train itself.

- We can analyze the biological significance of noise (or the relevant attributes) for each experiment.

We can work on finding some other preprocessing techniques that may lead to even better results. Instead of taking successive differences, we can try measures such as standard deviation and can eliminate genes ranked on that basis. We can also use some intrinsic properties of the time series to come up with a metric for noise elimination.

REFERENCES

1. Al-Shahib, A., Breitling, R., and Gilbert, D. (2006): *Feature Selection and the Class Imbalance Problem in Predicting Protein Function from Sequence,* Applied Bioinformatics, 5(2):124.

2. Ando, S. (2007): *Clustering Needles in a Haystack: An Information Theoretic Analysis of Minority and Outlier Detection.* 7th IEEE Int. Conf. on Data Mining, pages 13-22.

3. Asuncion, A. and Newman, D. J. (2007): *UCI Machine Learning Repository* [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science.

4. Batista, G. E. A. P. A., Prati, R. C., and Monard, M. C. (2004): *A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data*. ACM SIGKDD Explorations Newsletter, 6:20-29.

5. Batista, G. E. A. P. A., Carvalho, A. C. P. L. F., and Monard, M. C. (2000): *Applying One-Sided Selection to Unbalanced Datasets*. In Proceedings of the Mexican International Conference on Artificial Intelligence: Advances in Artificial Intelligence, pages 315-325.

6. Blum, A. L. and Langley, P. (1997): *Selection of Relevant Features and Examples in Machine Learning.* Artificial Intelligence, 97(1-2).

7. Bo, T. H. and Jonassen, I. (2002): *New Feature Subset Selection Procedures for Classification of Expression Profiles*. Genome Biology, 2002;3(4).

8. Chandola, V., Banerjee A., and Kumar V. (2009): *Anomaly Detection - A Survey*, ACM Computing Surveys, 41(3):Article 15.

9. Chen L., Goldgof, D. B., Hall L. O., and Eschrich S. (2007): *Noise-Based Feature Perturbation as a Selection Method for Microarray Data*, In Proceedings of Third International Symposium: Bioinformatics Research and Applications (ISBRA), pages 237-247.

10. Chen, X. and Wasikowski, M (2008): *FAST: A ROC-based Feature Selection Metric for Small Samples and Imbalanced Data Classification Problems*. Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 124-132.

11. Cheng, H., Tan, P-N., Potter, C., and Klooster, S (2009): *Detection and Characterization of Anomalies in Multivariate Time Series.* Proceedings of SIAM International Conference on Data Mining.

12. Cieslak, D. A. and Chawla, N. V. (2008): *Start Globally, Optimize Locally, Predict Globally: Improving Performance on Imbalanced Data*. Eighth IEEE International Conference on Data Mining, pages 143-152.

13. Costa, I. G., Carvalho, F. de A. T. de, and Souto, M. C. P. de (2002): *A Symbolic Approach to Gene Expression Time Series Analysis.* IEEE Brazilian Symposium on Neural Networks, pages 25-30.

14. Costa, I. G., Carvalho, F. de A. T. de, and Souto, M. C. P. de (2002): *A Symbolic Approach to Gene Expression Time Series Analysis.* IEEE Brazilian Symposium on Neural Networks, pages 25-30.

15. Deng, L., Pei, J., Ma, J., and Lee, D.L (2004): *A Rank Sum Test Method for Informative Gene Discovery*. Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 410-419.

16. Deng, L., Pei, J., Ma, J., and Lee, D.L (2004): *A Rank Sum Test Method for Informative Gene Discovery*. Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 410-419.

17. Denton, A. (2005): *Kernel-Density-Based Clustering of Time Series Subsequences Using a Continuous Random-Walk Noise Model*. Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM), Houston, pages 122-129.

18. Denton, A. M (2004): *Density-Based Clustering of Time Series Subsequences.* Proc. of the 3rd Workshop on Mining Temporal and Sequential Data (TDM 04) in conj. with the 10th ACMSIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle.

19. Denton, A. M., Besemann, C A., and Dorr, D. H.(2009): *Pattern-Based Time-Series Subsequence Clustering Using Radial Distribution Function,* Knowl Inf Syst 18:1-27.

20. Denton, A. M., and Kar, A. (2007): *Finding Differentially Expressed Genes Through Noise Elimination.* Proc. of the Workshop on Data Mining for Biomedical Informatics in conjunction with the 6th SIAM International Conference on Data Mining, Minneapolis, MN.

21. Ding, A. and Peng, H. (2003): *Minimum Redundancy Feature Selection from Microarray Gene Expression Data.*

22. Domingos, P. and Pazzani, M. (1997): *On the Optimality of the Simple Bayesian Classifier Under Zero-One Loss.* Machine Learning, 29(2-3): 103-130.

23. Drummond, C. and Holte, R. C. (2003): *C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling Beats Over-Sampling.* Proceedings of the ICML workshop on learning from imbalanced datasets.

24. Dy, J. G. and Brodley, C. E. (2004): *Feature Selection for Unsupervised Learning*, Journal of Machine Learning Research, 5:845-889.

25. Folleco, A., Khoshgoftaar, T. M., and Napolitano, A. (2008): *Comparison of Four Performance Metrics for Evaluating Sampling Techniques for Low Quality Class-Imbalanced Data.* Seventh International Conference on Machine Learning and Applications

26. Geurts, P. (2001): *Pattern Extraction for Time Series Classification.*

27. Guyon, I. and Elisseeff, A. (2003): *An Introduction to Variable and Feature Selection.* Journal of Machine Learning Research, 3:1157-1182

28. Hall, M. A. (2000): *Correlation-Based Feature Selection for Discrete and Numeric Class Machine Learning*. Proceedings of the Seventeenth International Conference on Machine Learning, pages 359-366.

29. Hall, H. A. and Holmes, G. (2003): *Benchmarking Attribute Selection Techniques for Discrete Class Data Mining.* IEEE Trans. on Knowledge and Data Engineering.

30. Han, J., Sanchez, R., and Hu, X. T. (2005): *Feature Selection Based on Relative Attribute Dependency: An Experimental Study.* Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part I, ser. Lecture Notes in Artificial Intelligence, 3641:214-223.

31. Hodge, V. and Austin, J. (2004): *A Survey of Outlier Detection Methodologies.* Artificial Intelligence Review, 22(2).

32. Hong, E. L., Balakrishnan, R., Dong, Q., Christie, K. R., Park, J., Binkley, G., Costanzo, M. C., Dwight, S. S., Engel, S. R., Fisk, D. G., Hirschman, J. E., Hitz, B. C., Krieger, C. J., Livstone, M. S., Miyasato, S. R., Nash, R. S., Oughtred, R., Skrzypek, M. S., Weng, S., Wong, E. D., Zhu, K. K., Dolinski, K., Botstein, D., and Cherry, J. M. (2008): *Gene Ontology Annotations at SGD: New Data Sources and Annotation Methods,* Nucleic Acids Res. Jan; 36 (Database issue):D577-581.

33. Hubble, J., Demeter, J., Jin, H., Mao, M., Nitzberg, M., Reddy, T. B., Wymore, F., Zachariah, Z. K., Sherlock, G., Ball, C. A. (2009): *Implementation of Gene Pattern Within the Stanford Microarray Database,* Nucleic Acids Res. Jan; 37 (Database Issue):D898-901.

34. Jakulin, A. and Bratko, I. (2003): *Analyzing Attribute Dependencies*. 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD),

35. Japkowicz, N. (2000): *The Class Imbalance Problem: Significance and Strategies.* Proceedings of the International Conference on Artificial Intelligence (ICAI), pages 111-117.

36. John, G. H., Kohavi, R., and Pfleger, K. (1994): *Irrelevant Features and the Suset Selection Problem.* Machine Learning: Proceedings of the 11th Int. Conf., (eds.) Cohen, W.W. and Harsh, H., pp. 121 – 129.

37. Joshi, M. V. (2002): *On Evaluating Performance of classifiers for Rare Classes*. IEEE International Conference on Data Mining (ICDM), pages 641-644.

38. Keller, A. D., Schummer, M., Le Hood and Ruzzo, W.L. (2000): *Bayesian Classification of DNA Array Expression Data.*

39. Keogh, E., Lin, J., and Fu, A. (2005): *HOT SAX: Efficiently Finding the Most Unusual Time Series Subsequence.* In Proc. of the 5th IEEE International Conference on Data Mining.

40. Keogh, E., Lin, J., and Truppel, W. (2003): *Clustering of Time Series Subsequences is Meaningless: Implications for Previous and Future Research,*

41. Keogh, E., Lonardi, S., and Chiu, B. (2002): *Finding Surprising Patterns in a Time Series Database in Linear Time and Space.* In Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 550-556.

42. Keogh, E. and Kasetty, S. (2002): *On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration.* In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23 – 26.

43. Kotsiantis, S., Kanellopoulos, D., and Pintelas, P. (2006): *Handling Imbalanced Datasets*: A Review GESTS International Transactions on Computer Science and Engineering.

44. Kubat, M., Holte, R., and Matwin, S. (1997): *Learning When Negative Examples Abound*. Proceedings of the 9th European Conference on Machine Learning (ECML), 1224:146-153.

45. Kubat, M., and Matwin, S. (1997): *Addressing the Curse of Imbalanced Training Sets: One-Sided Selection.* In Proceedings of the Fourteenth International Conference on Machine Learning, pages 179-186.

46. Kundaje, A., Middendorf, M., Gao, F., Wiggins, C., and Leslie, C. (2005): *Combining Sequence and Time Series Expression Data to Learn Transcriptional Modules.* IEEE/ACM Transaction on Computational Biology and Bioinformatics, 2:194-202.

47. Liu, X., Wu, J., and Zhou, Z. (2006); *Exploratory Under-Sampling for Class-Imbalance Learning*. In IEEE Sixth International Conference on Data Mining, pages 965-969.

48. Mitchell, T. M. (1997): *Machine Learning*, The McGraw-Hill Companies, Inc.

49. Mitra, P., Murthy, C. A., and Pal, S. K. (2002): *Unsupervised Feature Selection Using Feature Similarity*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(3).

50. Monard, M. C., and Batista, G. E .A. P. A. (2003): *Learning with Skewed Class Distributions.* CADERNOS DE COMPUTACAO XX

51. Mörchen, F. (2003): *Time Series Feature Extraction for Data Mining Using DWT and DFT.*

52. Muthukrishnan, S., Shah, R., and Vitter, J. S. (2004): *Mining Deviants in Time Series Data Streams.* In proceedings of 16th International Conference on Scientific and Statistical Database Management (SSDBM'04), pages 41-50.

53. Pajares, R. G., Benitez, J. M., and Palmero, G. S. (2008). *Feature Selection for Time Series Forecasting: A Case Study.* In Eighth International Conference on Hybrid Intelligent Systems, pages 555-560.

54. Provost, F. and Fawcett, T. (1997): *Analysis and Visualization of Classifier Performance: Comparison Under Imprecise Class and Cost Distributions.* Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, pages 43-48.

55. Provost, F., Fawcett, T., and Kohavi, R. (1998): *The Case Against Accuracy Estimation for Comparing Classifier*. Proceedings of the Fifteenth International Conference on Machine Learning (ICML), pages 445-453.

56. Rish, I.. Hellerstein, J., and Thathachar, J. (2001): *An Analysis of Data Characteristics that Affect Naïve Bayes Performance.*

57. Roddick, J. F., and Spiliopoulou, M. (2002): *A Survey of Temporal Knowledge Discovery Paradigms and Methods,* Transactions on Data Engineering, 14:750-767.

58. Simon, G. J., Kumar, V., and Zhang, Z. (2007): *Estimating False Negatives for Classification Problems with Cluster Structure*. The Proceedings of SIAM International Conference on Data Mining.

59. Søndberg-madsen, N., Thomsen, C., and Peña, J.M. (2003): *Unsupervised Feature Subset Selection.* Proceedings of the Workshop on Probabilistic Graphical Models for Classification, pages 71-82.

60. van der Walt, C. and Barnard, E. (2006): *Data Characteristics that Determine Classifier Performance.* 17th Annual Symposium of the Pattern Recognition Association of South Africa.

61. van Hulse, J. D., Khoshgoftaar, T. M., and Huang, H. (2007): *The Pair Wise Attribute Noise Detection Algorithm,* Knowledge and Information Systems, 11(2).

62. Varshavsky, R., Gottlieb, A., Linial, M., and Horn, D. (2006): *Novel Unsupervised Feature Filtering of Biological Data,* Bioinformatics, 22(14).

63. Verleysen, M. and Francois, D. (2005): *The Curse of Dimensionality in Data Mining and Time Series Prediction*. Cabestany, J., Prieto, A., and Sandoval, F. (eds.), Computational Intelligence and Bioinspired Systems, Lecture Notes in Computer Science 3512, Springer, pages 758-770.

66. Weiss, G. M. (2004): *Mining with Rarity: A unifying Framework.* ACM SIGKDD Explorations Newsletter, 6(1).

65. Weiss, G. M. and Provost, F. (2001): *The Effect of Class Distribution on Classifier Learning: An Empirical Study.* Technical Report ML-TR-44, Dept. of Computer Science, Rutgers University.

64. Xing, E. P., Jordan, M. I., and Karp, R. M. (2001): *Feature Selection for High-Dimensional Genomic Microarray Data*. Proceedings of the Eighteenth International Conference on Machine Learning, pages 601-608.

67. Xiong, H., Pandey, G., Steinbech, M., and Kumar, V. (2006): *Enhancing Data Analysis with Noise Removal.* IEEE Transactions on Knowledge and Data Engineering, 18(3).

68. Yang, Q., and Wu, X. (2006): *10 Challenging Problems in Data Mining*, International Journal of Information Technology and Decision Making, 5(4):507-604.

69. Ye, L. and Keogh, E. (2009): *Time Series Shapelets: A New Primitive for Data Mining*. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 947-956.

70. Yoon, H., Yang, K., and Shahabi, C. (2005). *Feature Subset Selection and Feature Ranking for Multivariate Time Series*. IEEE Transactions on Knowledge and Data Engineering, 17:1-13.

71. Zhu, X. and Wu, X. (2004): *Class Noise vs. Attribute Noise: A Quantitative Study*. Artificial Intelligence Review,