Advances in Classification in Non-Stationary Environments

By
Hanane Tavasoli

A thesis submitted to
the Faculty of Graduate and Postdoctoral Affairs
in partial fulfilment of
the requirements for the degree of
Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Computer Science
Carleton University
Ottawa, Ontario

October 2015

© Copyright
2015, Hanane Tavasoli

The undersigned hereby recommend to
the Faculty of Graduate and Postdoctoral Affairs
acceptance of the thesis,

Advances in Classification
in Non-Stationary Environments

submitted by

Hanane Tavasoli

Dr. Douglas Howe
(Director, School of Computer Science)

Dr. B. John Oommen
(Thesis Supervisor)

Carleton University
October 2015

ABSTRACT

Classification is a well-known problem in Pattern Recognition that has been extensively studied for decades. The classification process involves assigning a class label to an unlabeled element based on an available training sample. A common assumption in the majority of existing classification algorithms is that the stochastic distribution of the data being classified is stationary and does not change with time. However, in some real-world domains the data distribution can be non-stationary, implying that the distribution or characterizing aspects of the features change over time or the data generation phenomenon itself may change over time, which, in turn, leads to a variation in the data distribution.

In this thesis, we consider a problem of C -class classification and of detecting the source of data in periodic non-stationary environments. Within our model, sequential patterns arrive and are processed in the form of a data stream that was generated from different sources with distinct statistical distributions. Using a family of Stochastic-Learning based Weak Estimators, we adopt a scheme to estimate the vector of the probability distribution of the binomial/multinomial datasets. We also utilize the multiplication-based update algorithm, in order to provide a self-adjusting learning scheme to adapt the model to any abrupt changes occurring in the environment.

In this thesis we consider two different classification scenarios. First we study a scenario in which the stream of data was generated from more than two sources, each with their own fixed stochastic properties. We then proposed a novel *online* classifier for more complex data streams which are generated from non-stationary stochastic properties. An empirical analysis on synthetic datasets demonstrates the advantages of the introduced scheme for both the binomial and multinomial non-stationary distributions.

ACKNOWLEDGEMENTS

I am extremely grateful to have been supervised by Prof. B. John Oommen and it has been a pleasure working with him. I admire him deeply for his useful comments, remarks and engagement through the learning process of this Masters thesis. I would like to thank my husband, who has supported me throughout entire process, both by supporting me psychologically and for helping me in putting pieces together. I will, forever, be grateful for his help. Most of all, I am grateful to my family.

Contents

1	Introduction	2
1.1	Motivation for the Thesis	2
1.2	Objectives of the Thesis	3
1.2.1	Applications	4
1.3	Contributions of the Thesis	5
1.4	Organization of the Thesis	7
2	Literature Review	8
2.1	Introduction	8
2.1.1	Training <i>versus</i> Testing	9
2.1.2	Parametric <i>versus</i> Non-Parametric	9
2.1.3	Supervised <i>versus</i> Unsupervised	10
2.1.4	Known Data <i>versus</i> Stream-based Data	11
2.1.5	Stationary <i>versus</i> Non-Stationary	12
2.2	Foundational Strategies for Training/Estimation	13
2.2.1	Maximum Likelihood Estimation (MLE)	13
2.2.2	Bayesian Estimation	14
2.3	Training/Estimation for NSE	15
2.3.1	Autoregressive(AR) Model	15
2.3.2	Kalman Filter	16
2.4	Learning from Data Streams in NSE	18
2.4.1	FLORA	21
2.4.2	Statistical Process Control (SPC)	21

2.4.3	ADWIN	22
2.5	Stochastic Learning Weak Estimator (SLWE)	23
2.5.1	Learning Automata	24
2.5.2	Model for SLWE	25
2.5.3	Weak estimators of Binomial Distributions	25
2.5.4	Weak estimators of Multinomial Distributions	28
2.6	Applications for Non-stationary Environments	31
2.7	Limitations of the Previous	32
2.8	Model for NSE (Unknown to the PR system)	32
2.8.1	Periodic Switching Environment (PSE)	33
2.8.2	Markovian Switching Environment (MSE)	34
2.9	Conclusions	36
3	C-Class PR using SLWE	38
3.1	The PR Problem	38
3.2	New Problem and The Studied Model	39
3.3	Binomial Vectors: SE and NSE	40
3.3.1	Binomial Vectors: $d=2-6$, 2-class	42
3.3.2	Binomial Vectors: $d=2-6$, C-class	46
3.4	Multinomial Vectors: SE and NSE	58
3.4.1	Multinomial Vectors: $d=2-6$, $r=4$, 2-class	61
3.4.2	Multinomial Vectors: $d=2-6$, $r=4$, C-class	65
3.5	Conclusions	73
4	Online Classification Using SLWE	81
4.1	Introduction	81
4.2	New Problem and the Online Model	82
4.3	Binomial Data Stream	84
4.4	Multinomial Data Stream	91
4.5	Conclusion	94

5	Summary and Conclusion	103
5.1	Contributions	103
5.2	Future Work	105
	Bibliography	107

List of Figures

2.1	Plot of the expected value of $p_1(n)$, at time n , which was estimated by using the SLWE and the MLEW, where $\lambda = 0.817318$ and the window size was 32 (duplicated from [27]).	27
2.2	Plot of the Euclidean norm of $P - S$ (or Euclidean distance between P and S), for both the SLWE and the MLEW, where λ is 0.957609 and the size of the window is 63, respectively (duplicated from [27]). . . .	30
2.3	Plot of the Euclidean distance between P and S , where P was estimated by using both the SLWE and the MLEW. The value of λ is 0.986232 and the size of the window is 43 (duplicated from [27]). . . .	30
2.4	Graphical representation of the PSE model with 3 different states and with $T = 50$	34
2.5	Graphical representation of the PSE model with 3 different states and an unknown value for T	34
2.6	Graphical representation of the MSE model with 4 states and $\alpha = 0.9$. All the transitions between the states occur with the probability of $\frac{0.1}{3}$	35
3.1	An example of the true underlying probability of ‘0’, S_1 , for the first and second dimensions of a test set. The data was generated using two different sources in which the period of switching, T , was 50.	43
3.2	An example of the true underlying probability of ‘0’, S_1 , at time “ n ”, for the first and second dimensions of a test set which was generated with two different sources with a random switching period $T \in [50, 150]$	44

3.3	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 2-dimensional dataset with different switching periods, as described in Table 3.1.	45
3.4	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 3-dimensional dataset with different switching periods, as described in Table 3.2.	48
3.5	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 4-dimensional dataset with different switching periods, as described in Table 3.3.	48
3.6	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 5-dimensional dataset with different switching periods, as described in Table 3.4.	49
3.7	Plot of the accuracies of the SLWE classifier for different binomial datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 3.5.	50
3.8	Plot of the accuracies of the SLWE classifier for different binomial datasets each with a different switching period, T , and a different dimensionality, d . The numerical results of the experiments are shown in Table 3.5.	51
3.9	An example of the true underlying probability of ‘0’, S_1 , at time “ n ”, for the first and second dimensions of a test set which was generated with three different sources with a random switching period $T \in [50, 150]$	52
3.10	An example of the true underlying probability of ‘0’, S_1 , at time “ n ”, for the first and second dimensions of a test set which was generated with three different sources with a random switching period $T \in [50, 150]$	53
3.11	Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 2-dimensional dataset with different switching periods, as described in Table 3.6.	54

3.12	Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 3-dimensional dataset with different switching periods, as described in Table 3.7.	55
3.13	Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 4-dimensional dataset with different switching periods, as described in Table 3.8.	56
3.14	Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 5-dimensional dataset with different switching periods, as described in Table 3.9.	57
3.15	Plot of the accuracies of the SLWE classifier for different datasets with different dimensions d over different values of T . The numerical results of the experiments are shown in Table 3.10.	58
3.16	Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.11.	59
3.17	Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.11.	60
3.18	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 2-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.12.	62
3.19	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 3-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.13.	65
3.20	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 4-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.14.	66
3.21	Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 5-dimensional multinomial dataset with different switching periods, as described in Table 3.15.	68

3.22	Plot of the accuracies of the SLWE classifier for different multinomial datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 3.16.	71
3.23	Plot of the accuracies of the SLWE classifier for different multinomial datasets with different values for the switching period, T , over different values for the dimensionality, d . The numerical results of the experiments are shown in Table 3.16.	72
3.24	Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 2-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.17.	73
3.25	Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 3-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.18.	74
3.26	Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 3-class 4-dimensional dataset with different switching periods, as described in Table 3.19.	75
3.27	Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 5-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.20.	76
3.28	Plot of the accuracies of the SLWE classifier for different datasets with different dimensions d over different values of T . The numerical results of the experiments are shown in Table 3.21.	77
3.29	Plot of the accuracies of the SLWE classifier for different multinomial datasets with different switching period, T , over different dimensionality, d . The numerical results of the experiments are shown in Table 3.21.	78
3.30	Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.22.	79

3.31	Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.22.	80
4.1	Plot of the averages for the estimates of s_{11} , obtained from the SLWE and MLEW at time n , using the available training samples that arrived with the delay of $t_d = 10$. The stochastic properties of each class switched four times at randomly selected times.	85
4.2	An example of the true underlying probability of ‘0’, S_1 , for a one-dimensional binary data stream. The data was generated using two different sources in which the period of switching was 100, and the stochastic properties of the classes switched two times.	86
4.3	Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a one-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.1. . .	87
4.4	Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a 2-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.2.	90
4.5	Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a 3-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.3.	91
4.6	Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a 4-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.4.	92
4.7	Plot of the accuracies of the SLWE classifier for different binomial datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 4.5.	93

4.8	Plot of the accuracies of the SLWE classifier for different binomial datasets involved data from two non-stationary classes. Each dataset was generated with a different switching period, T , and a different dimensionality, d . The numerical results of the experiments are shown in Table 4.5.	94
4.9	Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a one-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.6. . .	97
4.10	Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 2-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.7. . .	98
4.11	Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 3-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.8. . .	99
4.12	Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 4-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.9. . .	100
4.13	Plot of the accuracies of the SLWE classifier for different multinomial($r=4$) datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 4.10.	101
4.14	Plot of the accuracies of the SLWE classifier for different multinomial datasets involved data from two non-stationary classes. Each dataset was generated with a different switching period, T , and a different dimensionality, d . The numerical results of the experiments are shown in Table 4.10.	102

ACRONYMS

AR	Autoregressive model
BE	Bayesian Estimation
KL	Kullback-Leibler
LA	Learning Automata
L_{RI}	Linear Reward-Inaction
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MLEW	MLE that uses a sliding window
MSE	Markovian Switching Environment
PR	Pattern Recognition
PSE	Periodic Switching Environment
SLWE	Stochastic Learning Weak Estimator
SPC	Statistical Process Control

Chapter 1

Introduction

In the past few years, due to the advances in computer hardware technology, large amounts of data have been generated and collected and are stored permanently from different sources. Some the applications that generate data streams are financial tickers, log records or click-streams in web tracking and personalization, data feeds from sensor applications and call detail records in telecommunications. Analyzing these huge amounts of data has been one of the most important challenges in the field of Machine Learning (ML) and Pattern Recognition (PR). Traditionally, ML methods are assumed to deal with static data stored in memory, which can be read several times. On the contrary, streaming data grows at an unlimited rate and arrives continuously in a single-pass manner that can be read only once. Further, there are space and time restrictions in analyzing streaming data. Consequently, one needs methods that are “automatically adapted” to update the training models based on the information gathered over the past observations whenever a change in the data is detected.

1.1 Motivation for the Thesis

Mining streaming data is constrained by limited resources of time and memory. Since the source of data generates a potentially unlimited amount of information, loading all the generated items into the memory and achieving offline mining is no longer

possible. Besides, in non-stationary environments, the source of data may change over time, which leads to a variations in the underlying data distributions. Thus, with respect to this dynamic nature of the data, the previous data model discovered from the past data may become irrelevant or even have a negative impact on the modeling of the new data streams that become available to the system.

A vast body of research has been performed on the mining of data streams to develop techniques for computing fundamental functions with limited time and memory, and it usually involved the sliding-window approaches or incremental methods. In most of cases, these approaches require some *a priori* assumption about the data distribution or need to invoke hypothesis testing strategies to detect the changes in the properties of data.

The motivation for this thesis is to investigate novel methods to tackle this problem.

1.2 Objectives of the Thesis

In this thesis we will study classification problems in non-stationary environments, where sequential patterns are arriving and being processed in the form of a data stream that was potentially generated from different sources with different statistical distributions. The classification of the data streams is closely related to the estimation of the parameters of the time varying distribution, and the associated algorithms must be able to detect the source changes and to estimate the new parameters whenever a *switch* occurs in the incoming data stream.

We will argue that using “strong” estimators that converge with probability of 1 is inefficient for tracking the statistics of the data distribution in non-stationary environments. However, “weak” estimator approaches are able to rapidly unlearn what they have learned and adapt the learning model to new observations. This feature of “weak” estimators make these approaches the most effective methods for estimation in non-stationary environments. In this work, we will employ a family of weak estimators, referred to as Stochastic Learning Weak Estimation (SLWE) methods [27], for classification in non-stationary environments. The SLWE has been

successfully used to solve *two*-class classification problems by Oommen and Rueda [27] by applying it on non-stationary *one*-dimensional datasets. In this thesis we will study the performance of the SLWE with more complex classification schemes, which will be discussed in Chapters 3 and 4.

In this thesis we will consider two different classification scenarios. First we will study a scenario in which the stream of data was generated from more than two binomial and/or multinomial sources, each with their own fixed stochastic properties, and where the source of data might be switched in a periodic non-stationary manner. Subsequently, we will consider a more complex classification problem, where the classes' stochastic properties potentially vary with time as more instances become available.

1.2.1 Applications

The outcome of this work can be used in several real-life applications. We mention two of them here.

First of all, this PR scheme can be applied for the detection of the source of news streams. In this case, an observed stream could be either a live video broadcasted from a TV channel or news released in a textual form, for example, on the internet. This problem has been studied by several researchers by considering shots from the video and using them in the classification to fall into one of a few predefined classes. Since an image processing solution would be very time consuming, its applicability for real-time solutions is impractical. Our method could, however, be used to simplify this problem by considering the news streams that arrive in the form of text blocks extracted from the closed captioning text embedded in the video streams. A similar problem was considered by Oommen and Rueda in [27], in which they analyzed bit streams generated from two sources of news, namely sports and business.

Language detection can be considered as another application for our classification scheme. For example, consider an online conversation in different languages taking place in either a text or a speech format. The conversation can be considered as a stream of symbols, and the aim would be to detect the language of communication

at any given time instant.

Finding suitable non-stationary data streams to be used for testing our method is challenging because all the real-world benchmark data sets provided by the UCI machine learning repository are designed for stationary environments. It should be noted that the available non-stationary news streams utilized in [27] only included *binomial* data, as no multinomial data sets were available for testing. Due to the lack of multinomial non-stationary real-world datasets, we will use synthetic benchmarks in this thesis.

1.3 Contributions of the Thesis

The main contributions of the thesis are the following:

- As a primary contribution, we have studied a problem of classification and detecting the source of data in periodic non-stationary environments using the SLWE family of weak estimators. In Oommen and Rueda’s work [27], the power of the SLWE method was only demonstrated in *two*-class classification problems, and the classification was performed on non-stationary *one*-dimensional datasets, where each source had fixed stochastic properties. In this thesis we have evaluated the performance of the SLWE with more complex classification schemes, where the multinomial/binomial instances arrive sequentially in form of a data stream, and the stochastic properties of the stream could vary as more instances become available.
- Secondly, we have generalized the above SLWE-based scheme for classification of binomial *and* multinomial data streams, which were also multidimensional. Further, the data could have been potentially generated from more than two sources. In our experiments, the SLWE method was used to estimate the *vector* of the probability distribution from binomial and multinomial multidimensional datasets in periodic non-stationary environments, where the periodicity was unknown to the classifier.

- Most of the data stream mining approaches, have involved building an initial model from a sliding window of recently observed instances and thereafter, refining the learning model periodically or whenever its performance degrades based on the current window of observed data. We present a novel framework to deal with concept and distribution drift over data streams in non-stationary environments, which is more efficient and provides more accurate results.
- We have introduced an *online* classification scheme composed of three phases. In the first phase, the model learns from the available labeled samples. In the second phase, the learned model predicts the class label of the unlabeled instances currently observed. In the third phase, after knowing the true class label of the instances, the classification model is adjusted in an *online* manner.
- The *online* classification model that we have adopted for data streams in which the classes' distributions changes *abruptly*, is both interesting and novel. In fact, instead of assuming that each source involved in the generation of the data stream has fixed stochastic probabilities (which makes it possible for the system to train the model in an *offline* manner), we consider the scenario where changes in the distribution of each class occur at unknown random time instants. Furthermore, we suppose that the class distribution changes to a possibly new random distribution after the drift. Indeed, models such as these that include time-varying distributions for the classes, are more realistic than ones that possesses fixed stochastic properties for each class. Clearly, the above described settings represent a more challenging scenario than the previous state-of-the-art model.
- Our classifier scheme provides a real-time self-adjusting learning model, utilizing the multiplication-based update algorithm of the SLWE at each time instance, as new labeled instances arrive. Instead of using a single training model and maintaining counters to keep important data statistics, we have used a technique to replace these frequency counters by data estimators. In this way, the data statistics are updated every time a new element is inserted, without needing to rebuild its model when a change in the distributions is detected.

- Extensive experimental results that we have obtained, for both the binomial and multinomial distributions, demonstrate the efficiency of the proposed classification schemes in achieving a good performance for data streams involving non-stationary distributions under different scenarios of concept drift.

1.4 Organization of the Thesis

The following chapter explains how parameter and distribution estimation play a crucial role in classification and learning. We briefly review the literature available on the families of approaches reported for estimation. We survey the available estimation approaches that have been developed to learn from streams with unknown dynamics in stationary and non-stationary environments. We proceed with discussing the issues and challenges encountered when one learns from data streams and provide a brief explanation about the theoretical properties of the SLWE.

In Chapter 3 we introduce a SLWE-based classifier and study its performance on different data streams. We perform our experiments on synthetic binomial *and* multinomial data streams. These streams are also multidimensional, and could have been potentially generated from more than two sources of data.

Thereafter, in Chapter 4, we present the details of the design and implementation of an online classifier using the general framework presented in the previous chapter. We also show how it can be used to perform online classification, and present a new experimental framework for concept drift.

Chapter 5 concludes the thesis.

Chapter 2

Literature Review

2.1 Introduction

Estimation theory is a fundamental subject that is central to the fields of Pattern Recognition (PR) and data mining. The majority of problems in PR require the estimation of the unknown parameters that characterize the underlying data distributions.

In this chapter, we present a brief survey of how parameter and distribution estimation play a crucial role in classification and learning. This chapter surveys, in some detail, the literature available on the families of approaches for estimation, and proceeds to discuss the issues and challenges encountered when one learns from data streams. In particular, we focus on the special issues to be considered when we work with change detection. Indeed, we rather survey estimation approaches that have been developed to learn from streams with unknown dynamics in non-stationary environments.

In what follows, we shall briefly discuss how estimation plays a fundamental role in the various aspects of PR.

2.1.1 Training *versus* Testing

As we have discussed earlier, classification involves the task of allocating of a set of instances into groups or classes with respect to some common relations or affinities, and is performed in two phases referred to as training (learning) and testing respectively. Both of these phases involve the task of estimation and using the estimates concerned.

Several methods have been proposed which tackle the training problem by defining models for the different groups and categories based on the information given by the training set data. In PR applications, a d -dimensional training set is characterized by a d -dimensional distribution characterizing the corresponding d -dimensional probability vector. Typically, the designer of the system does not possess complete knowledge about this probabilistic structure. The problem involves learning how to design or train the classifier based on this available information. Estimation is the primary and most important task involved in learning the model for the training data, and for the unknown parameters of the underlying distributions using only the training samples.

In the testing phase, the intention is to assign each input vector to one of the finite number of classes. To classify a new point and minimize the probability of misclassification, the testing point should be assigned to the class having the largest posterior probability. Again, in order to determine and maximize the posterior probability, one does not use the true probabilities but the estimates of the unknown probability distributions of each class [10].

2.1.2 Parametric *versus* Non-Parametric

In some learning models, the designer does not assume complete information about the probability structure of the underlying categories. Rather, one assumes the general form of their distributions, which then is central to the estimation. These specific cases are addressed by *parametric estimation* methods, where the parameters of the known distribution are estimated using the observed data set [10]. The binomial/multinomial and Gaussian distributions are specific examples of parametric

distributions that are used for discrete and continuous random variable data domains, respectively. These distributions are governed by a small number of parameters, which, for instance, are the mean and variance used to define a Gaussian distribution.

As opposed to the above scenario, in other cases, assuming a specific functional form for the distribution is inappropriate, as there is no prior parameterized knowledge about the underlying probability structure. For these cases, a *non-parametric* density estimation method is, typically, utilized as an alternative approach that only uses the information contained in the training samples themselves. Such approaches do, indeed, have parameters that control the model's complexity, although they do not involve the form of the distribution. Histogram-based methods are, for example, one of the non-parametric classification approaches that operates using the frequencies of the data samples [6]. Again, estimation is essential to estimate the actual data frequencies in order to approximate the probabilities of the data occurring in the intervals of feature's domains. Briefly stated, kernel-based estimation methods, and nearest-neighbors algorithms are the other well-known methods available for achieving non-parametric estimation.

2.1.3 Supervised *versus* Unsupervised

PR can also be either supervised or unsupervised, and the estimation used in both these settings is also distinct. Learning applications in which the training set consists of labeled samples are known as supervised learning problems, where the information about class labels is crucial for the estimation. The importance of estimation for these kind of problems was discussed earlier. However, in other PR problems the training data consists of a set of input vectors without any corresponding labels. This setting is referred to as unsupervised learning [6, 10].

The task in unsupervised learning is to extract relevant information from the training set that can also assist in assigning labels to the samples.

In situations where one has to build a statistical model from labeled data, a common method consists of estimating the probability density functions associated with

the relevant data within the input space. In this case, density estimation approaches such as histograms, Parzen windows, or kernel-based density estimation, are used to determine the probability density functions.

2.1.4 Known Data *versus* Stream-based Data

Traditionally, in most of the ML and PR applications, such as in speech, and fingerprint recognition, the entire training data is available *a priori* and the data distribution does not change over time.

Learning models in these applications are produced based on the entire training set, where an off-line procedure is applied to the training set to generate a decision model. In this case, all the training samples are available for the estimation phase. Using this set, one can obtain an approximation of the stationary probability distribution that possibly generated the data set.

In more challenging applications, data streams are generated and collected in a one pass manner, where each element can be read only once. As the data samples in these data domains arrive incrementally, loading the entire dataset into memory and processing it off-line is not a feasible option. In these cases, where one has to build a statistical model from massive amounts of data, a common approach is to use a random subset of the training samples. However, in some applications in which data may be generated as per different distributions, more powerful estimation techniques are required to handle concept drift problems and to approximate the form of the distribution generating the data stream.

According to Hulten and Domingos [15], an efficient learning system for mining continuous, high-volume and “infinite” data streams must be able to build a decision model using a single scan over the training set. The model should also be able to handle concept drift problems and function with limited resources of both time and memory. Density estimation is an important component in the classification of data streams where the volume of data is very large, and the data distribution is unknown.

2.1.5 Stationary *versus* Non-Stationary

The majority of ML approaches have been developed to deal with data domains in which the underlying distributions are stationary. Learning in these environments is similar to batch learning. It is pertinent to mention that all of the benchmark data sets that deal with ML and PR fall into this class.

One encounters an additional problem in learning when the data is based on the properties of data streams. The issue at stake is that the data distribution, in these cases, can be non-stationary, implying that the distribution or characterizing aspects of the features change over time. In non-stationary environments, the data generation phenomenon itself may change over time, which, in turn, leads to a variation in the data distribution. The goal of learning approaches in non-stationary environments is to estimate the parameters of the distributions, and to adapt to any abrupt and/or gradual changes occurring in the environment. In other words, the learning and classification models must be updated when significant changes in the underlying data stream are detected.

The important issue here is that the estimation and training must be achieved without a knowledge of when and how the environment has changed, rendering the problem to be far from trivial.

It is very important to understand that in non-stationary environments, old observations become irrelevant to the current state or might even have a negative effect on the learning process. For data domains of these kinds, the estimation mechanisms should be able to incorporate phenomena akin to concept drift. They should be able to forget outdated data and adapt the estimation to the more recently-observed observed data elements.

The body of this thesis deals with training and testing in non-stationary environments.

2.2 Foundational Strategies for Training/Estimation

Apart from the areas of PR and ML, parameter estimation is the classical and central problem encountered in statistics, and it has been solved using several paradigms. In this section, we discuss two well-known and reasonable procedures in estimation theory, which are the Maximum Likelihood Estimation (MLE) and the Bayesian Estimation (BE) paradigms [10, 35]. Since they are well reported in the literature, these reviews will be very brief.

Universally, estimation algorithms learn the required statistics from a collection of the observed dataset. *Linear estimators* are the simplest estimator algorithms that simply returns the expected value of a function of the observed data. The MLE and similar methods view the parameters as being fixed but unknown. The values that maximize the probability of obtaining the observed samples is considered to be the best estimate. In contrast, Bayesian methods view the parameters as random variables themselves, having some known reproducible distribution.

2.2.1 Maximum Likelihood Estimation (MLE)

The MLE approach is a method for estimating the unknown parameters of a statistical model by maximizing the likelihood of the parameters generating the dataset.

In the ML method, it is assumed that for a class ω_j , $p(x|\omega_j)$ has a known parametric form, which is determined uniquely by the value of a parameter, θ . The objective of MLE is to obtain the most likely estimate for the unknown parameter θ , which could have yielded the observed data, D .

Let D be a vector of observed data (of n samples): $D = \{x_1, \dots, x_n\}$, which is assumed to be drawn independently from the probability density $p(x|\theta)$. Then:

$$p(D|\theta) = \prod_{k=1}^n p(x_k|\theta). \quad (2.1)$$

$p(D|\theta)$ is called the likelihood function of p with parameter θ having generated the set of samples. The MLE of θ , $\hat{\theta}$, is the value that maximizes $p(D|\theta)$, and so:

$$\hat{\theta} = \arg_{\theta} \max p(x|\theta). \quad (2.2)$$

In the MLE approach, for all the well-known distributions, $\hat{\theta}$ converges to θ with probability of 1 as the number of training samples increase. In addition, estimation using an MLE approach can be simpler than using alternate methods such as a Bayesian technique (explained below), since the MLE approach rarely requires explicit differential calculus techniques or gradient search for the estimation [10], but they are implicitly used in solving for $\hat{\theta}$ as per Eq. (2.2). The existing literature on the MLE works with the assumption that the data distribution does not change with time.

2.2.2 Bayesian Estimation

Another completely different way of achieving the estimation is by the so-called Bayesian paradigm which uses the Bayesian principle, applicable to almost all areas of probability and statistics and their corresponding application domains. This paradigm invokes the Bayes rule which computes the posterior probabilities/distributions by using the prior probabilities/distributions.

In the Bayesian *estimation* strategy one assumes that the parameter to be estimated is a random variable in its own right. This distribution is somehow dependent on the distribution of the random variable itself. To be more specific, let X be the random variable, characterized by a distribution $p(x|\theta)$, where θ is its unknown parameter. The Bayesian principle when applied to estimation, assumes that θ has a distribution of its own, say, $g(\theta)$. The aim now is to obtain the best value for θ , say, $\hat{\theta}$, which follows $g(\cdot)$ and yet maximizes the probability of generating the dataset.

In order to estimate the value of θ based on the observations, D , the *a priori* probability distribution $g(\theta)$, is used to compute *a posteriori* density $g(\theta|D)$. $\hat{\theta}$ is, typically, the mean of $g(\theta|D)$ or at its maximum. The aim of the Bayesian exercise is to compute $g(\theta|D)$ based on the Bayes formula as follows:

$$g(\theta|D) = \frac{p(D|\theta)g(\theta)}{\int p(D|\theta)g(\theta)d\theta}. \quad (2.3)$$

Generally, one assumes a parametric form for $g(\theta)$ so that the distribution $g(\theta|D)$ is of the same form. Such a distribution is called the “conjugate” prior.

The Bayesian strategy in estimation requires information in the form of the *a priori*

distribution for the unknown parameters. It is, therefore, not an appropriate method for nonparametric problems, where the density function must be estimated either by the Parzen window approach or a direct construction of the decision boundary based on the training data (e.g., by a k -nearest neighbor) [16].

2.3 Training/Estimation for NSE

A common assumption in the majority of estimation algorithms is that the data is stationary and that the parameter, which is being estimated, does not change with time. However, if a target is “moving”, the concept or feature determining the target tends to change with time. For data domains of these types, the estimation mechanisms should be able to incorporate concept drift, forget outdated data and adapt the estimation to the most recent observed data. The Autoregressive and the Kalman filter are two efficient estimation schemes that have the ability to model dependence with time, and we review these below.

2.3.1 Autoregressive(AR) Model

The Autoregressive model (AR) can be used to estimate the unknown parameter of observations that are related to the past observations. An AR model of degree p , which is denoted by $AR(p)$, uses the p recently-observed instances to estimate the unknown parameters at time n which is given by the following equation:

$$x(n) = \sum_{i=1}^p \beta_i x(n-i) + \epsilon(n), \quad (2.4)$$

where β_i is the autoregression coefficient that is associated with the i^{th} measurement, and $\epsilon(n)$ is an uncorrelated innovation process with zero mean [11]. The difference equation in Eq. (2.4) is an expression directly relating the value of x at time n , $x(n)$, to the value of x at a previous p time instances, plus a random variable ϵ dependent on time n , $\epsilon(n)$. The AR coefficients can be derived using different techniques such as a least squares method and the Burg Maximum Entropy method. A common least

squares method is based on the Yule-Walker equations that can be written in matrix form as follows:

$$\begin{bmatrix} 1 & r_1 & r_2 & r_3 & \cdots & r_{p-1} \\ r_1 & 1 & r_1 & r_2 & \cdots & r_{p-2} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{p-1} & r_{p-2} & r_{p-3} & r_{p-4} & \cdots & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_p \end{bmatrix} \quad (2.5)$$

where r_d is the autocorrelation coefficient at delay d [7].

The simplest Autoregressive model of first order is given as follows:

$$AR(1) : x(n) = \beta_0 + \beta_1 x(n-1) + \epsilon(n), \quad (2.6)$$

which is simply a first order linear difference equation. The term “autoregressive” is used to describe this method, since it is actually a linear regression approach based on the past elements.

2.3.2 Kalman Filter

The Kalman Filter [17] is a recursive estimation algorithm that estimates the parameter or the state of a dynamic system from a series of noisy measurements [3]. In this method, the time varying state of x at time n , given the past observed measurements, is estimated by using a linear stochastic difference equation:

$$X(n) = AX(n-1) + Bu(n) + w(n-1), \quad (2.7)$$

where $X(n)$ is the state vector with unknown initialization at time n which is unobservable. On the other hand, the noisy measurements Z are observable and assumed to be:

$$Z(n) = HX(n) + v(n). \quad (2.8)$$

In Eq. (2.7) $u(n)$ is a noise vector, which is characterized by White Gaussian Noise. A in Eq. (2.7) and H in Eq. (2.8) are known matrices that relate the state of the system at time $n-1$ to the current state of the system, and the observed measurement at time n , respectively. $w(n)$ and $v(n)$ are the random vectors, which

represent the process and the measurement noise and are assumed to be independent, normally distributed and centered at 0 with known covariance matrices.

$$p(w) \sim N(0, Q), \quad p(v) \sim N(0, R). \quad (2.9)$$

The aim of the Kalman filter is to estimate the state vector at the current timestep, $\hat{X}(n)$, using the state at the previous timestep and the measurement data corrupted by noise. This estimated state is referred to as the *a priori* state estimate because the Kalman filter uses the estimated state at time $n - 1$ to produce an estimate of the current state at time n without considering the current observations. Subsequently, whenever $Z(n)$ (the current state information) is observed, the *a priori* state is updated using the information about the current observation. In fact, the Kalman filter involves two updating processes, namely the time update equations and the measurement update equations. In the time update process, the filter uses the state estimate from the previous timestep to produce an estimate of the state at the current timestep [11]:

$$\hat{X}(n)^- = A\hat{X}(n-1) + Bu(n). \quad (2.10)$$

The measurement update process uses the obtained feedback and combines the *a priori* estimated state with the new observation in order to obtain an improved *a posteriori* estimate.

The current state is estimated as a linear combination of $\hat{X}(n)^-$ and the difference between the noisy measurement $Z(n)$ and $H\hat{X}(n)^-$:

$$\hat{X}(n) = \hat{X}(n)^- + K(n)(Z(n) - H\hat{X}(n)^-), \quad (2.11)$$

where $(Z(n) - H\hat{X}(n)^-)$ is a difference between the predicted measurement $H\hat{X}(n)^-$ and the observed information $Z(n)$, and the matrix K is referred to as the *gain factor*, which minimizes the covariance matrix of *a posteriori* error [17]. K is defined as follows:

$$K(n) = \frac{P(n)^- H^T}{HP(n)^- H^T + R}. \quad (2.12)$$

$$P(n)^- = AP(n-1)A^T + Q. \quad (2.13)$$

$$P(n) = (I - K(n)H)P(n)^-. \quad (2.14)$$

The Kalman filter's performance depends on the accuracy of the *a priori* assumptions of linearity of the difference stochastic equation. It is also crucial to have normal distributions for $w(n)$ and $v(n)$ with fixed covariances and zero means.

When dealing with data streams that vary over time, both of the mentioned assumptions can cause problems, as the difference equation may not be linear. Also estimating the distribution parameters of $w(n)$ and $v(n)$ is not trivial for data streams [3].

2.4 Learning from Data Streams in NSE

Learning in non-stationary environments is of great importance, and this problem is closely related to that of detecting concept changes and also of estimating the dynamic distribution associated with a set of data. Basseville and Nikiforov [1], Chen and Gupta [8], and Sebastian and Gama [33] have provided fairly good and detailed surveys on the topic of change detection methods. The methods presented in the literature are different with respect to the type of change they are expected to detect, and the underlying assumptions made about the streaming data. In general, most algorithms in the data stream mining literature have one or more of the following modules: a Memory module, an Estimator module, and a Change Detector module [3].

The Memory module is a component that stores summaries of all the sample data and attempts to characterize the *current* data distribution. Data in non-stationary environments can be handled by three different approaches, namely, by using partial memory, by window-based approaches and by instance-based methods. The term “partial memory” refers to the case when only a part of the information pertaining to the training samples are stored and used regularly in the training. In window-based

approaches, data is presented as “chunks”, and finally, in instance-based methods, the data is processed upon its arrival. In fact, the Memory module determines the forgetting strategy used by the mining algorithm operating in the dynamic environments.

The Estimator module uses the information contained in the Memory or only the observed information to estimate the desired statistics of the time varying streaming data. The Change Detector module involves the techniques or mechanisms utilized for detecting explicit drifts and changes, and provides an “alarm” signal whenever a change is detected based on the estimator’s outputs.

Change detection, in and of itself, is a very complex task, as its design is intended to be a trade-off between detecting real changes and avoiding false alarms. A significant amount of work has been performed in the area of concept change detection by both the statistical and machine learning communities. The subject of change detection was first employed in the manufacturing and quality control applications in the 1920-1930’s [1, 30]. By the introduction of sequential analysis, later in the 1950-1960’s, sequential detection procedures were developed, which considered the sequence of observations to detect unusual trends and patterns in the data.

A typical approach for the mining of data streams is based on the use of sliding windows. The algorithm considers a window of size W and divides the data stream into a sequence of data chunks. At each time step, learning is carried out based only on last W samples that are included in the window. Sliding window models are designed based on the assumption that the most recent information is more relevant than the historical data, which is similar to *first in-first out* data structures. At time t_j , when element j arrives, element $j - W$ is forgotten, where W indicates the size of the window [11]. In fact, at every time instant, the learning model of the data stream is generated using only the W samples resident in the window.

Several sliding window models have been presented in the literature [21, 24]. Kuncheva [21] presented a semi-parametric log-likelihood change detector based on Kullback-Leibler statistics. The author applied a log-likelihood framework that accommodates the Kullback-Leibler distance and the Hotelling’s t^2 test for equal means in order to detect changes in streaming multidimensional data. An implementation of

the fixed cumulative windowing scheme was proposed by Kifer *et al.* [19]. The authors here applied two sliding windows in their scheme, the first being a reference window, which was used as a baseline to detect changes, and the second being a “current window” to collect samples. They proposed an algorithm based on a statistical-test that specifies if the observed samples are generated from the same distribution. The high computational cost of maintaining a balanced form of the KS tree, is the main problem associated with this approach.

The main drawback of sliding window approaches is to know how to define the appropriate size for the window. A large window size would perform well on stationary environments but it will not be able to provide quick reactions when changes occur. On the other hand, a small window is suitable for rapid concept change detection algorithms, but it might affect the computational performance.

Apart from the sliding window schemes, many other incremental approaches have been proposed that infer change points during estimation, and use the new data to adapt the learning model trained from historical streaming data. The learning model in incremental approaches is adapted to the most recently received instances of the streaming data. Let $X = \{x_1, x_2, \dots, x_n\}$ be the set of training examples available at time $t = 1 \dots n$. An Incremental approach produces a sequence of hypothesis $\{\dots, H_{i-1}, H_i, \dots\}$ from the training sequence, where each hypothesis, H_i , is derived from the previous hypothesis, H_{i-1} , and the example x_i . In general, in order to detect concept changes in these types of approaches, some characteristics of the data stream (e.g., performance measures, data distribution, properties of data, or an appropriate statistical function) are monitored over time. When the parameters switch during the monitoring process, the algorithm should be able to adapt the model to these changes.

We shall now briefly review some schemes used for learning in non-stationary environments. The review here will not be exhaustive because the methods explained can be considered to be the basis for other modified approaches.

2.4.1 FLORA

Widmer and Kubat [37], presented the FLORA family of algorithms as one of the first supervised incremental learning systems for a data stream. The initial FLORA algorithm uses a fixed-size sliding window scheme. At each time step, the elements in the training window are used to incrementally update the learning model. The updating of the model involves two processes: an incremental *learning* process that updates the concept description based on the new data, and an incremental *forgetting* process in order to discard the out-of-date (or stale) data.

The initial FLORA system does not perform well on large and complex data domains. Thus, FLORA2 was developed to solve the problem of working with a fixed window size, by using a heuristic approach to adjust the window size dynamically. Further improvements of the FLORA were presented to deal with recurring concepts (FLORA3) and noisy data (FLORA4).

2.4.2 Statistical Process Control (SPC)

The Statistical Process Control (SPC) was presented by Gama *et al.* [12] for change detection in the context of data streams. The principle motivating the detection of concept drift using the SPC is to trace the error rate probability for the streamed observations. While monitoring the errors, the SPC provides three possible states, namely, “in control”, “out of control” and “warning” to define a state when a warning has to be given, and when levels of changes appear in the stream. When the error rate is lower than the first (lower) defined threshold, the system is said to be in an “in control” state, and the current model is updated considering the arriving data. When the error exceeds that threshold, the system enters the “warning” state. In the “warning” state, the system stores the corresponding time as the warning time, t_w , and buffers the incoming data that appears subsequent to t_w . In the “warning” mode, if the error rate drops below the lower threshold the “warning” mode is canceled and the warning time is reset. However, in case of an increasing error rate that reaches the second threshold, a concept change is declared and the learning model is retrained from the buffered data that appeared t_w .

2.4.3 ADWIN

Bifet and Gavalda [4, 5] proposed an adaptive sliding window scheme named ADWIN for change detection and for estimating statistics from the data stream. It was shown that the ADWIN algorithm outperforms the SPC approach and that it has the ability to provide rigorous guarantees on false positive and false negative rates. The initial version of ADWIN keeps a variable-length sliding window, W , of the most recent instances by considering the hypothesis that there is no change in the average value inside the window. To achieve this, the distributions of the sub-windows of the W window are compared using the Hoeffding bound, and whenever there is a significant difference, the algorithm removes all instances of the older sub-windows and only keeps the new concepts for the next step. Thus, a change is reliably detected whenever the window shrinks, and the average over the existing window can be considered as an estimate of the current average in the data stream.

Consider a sequence of real values $\{x_1, x_2, \dots, x_t, \dots\}$ that is generated according to the distribution D_t at time t . Let n denote the length of the W window, $\hat{\mu}_t$ be the observed average of the elements in W , and μ_w be the true average value of μ_t for $t \in W$.

Whenever two “large enough” sub-windows of W demonstrate “distinct enough” averages, the system infers that the corresponding expected values are different, and the older fragment of the window should be dropped. The observed average in both sub-windows are “distinct enough” when they differ by more than the threshold ϵ_{cut} :

$$\epsilon_{cut} = \sqrt{\frac{1}{2m} \cdot \ln \frac{4}{\delta'}}, \text{ where} \quad (2.15)$$

$$m = \frac{1}{1/n_0 + 1/n_1}, \quad \text{and} \quad \delta' = \frac{\delta}{n}, \quad (2.16)$$

where n_0 and n_1 denote the lengths of the two sub-windows and δ is a confidence bound.

Using the Hoeffding bound greatly over estimates the probability of large deviations for distributions with a small variance, which degrades the ADWIN’s performance, and it is also computationally demanding [29].

The ADWIN approach is, in fact, a linear estimator enhanced with a change detector. In order to improve the basic ADWIN method's performance, Bifet [3] replaced the linear estimator by an adaptive Kalman filter, where the covariances of $w(n)$ and $v(n)$ in Eqs. (2.7) and (2.8) have been set to $n^2/50$ and $200/n$ respectively, where n is the length of the window maintained by ADWIN.

2.5 Stochastic Learning Weak Estimator (SLWE)

Most of the data stream mining approaches have an estimator module in order to keep the statistics of the data distribution in non-stationary environments updated. However, it can be argued that using “strong” estimators such as the MLE and the Bayesian estimators that converge with probability of 1 are inefficient for dynamic non-stationary environments. In non-stationary environments, it is essential to use estimator schemes which can adopt the model promptly according to the new observations. In other words, the effective methods for estimation in non-stationary environments are the estimators which are able to quickly unlearn what they have learned.

Using the principles of stochastic learning, Oommen and Reuda [27] proposed a strategy to solve the problem of estimating the parameters of a binomial or multinomial distribution efficiently in non-stationary environments. This method is referred to as the Stochastic Learning Weak Estimator (SLWE), where the convergence of the estimate is “weak”, i.e., with respect to the first and second moments. Unlike the traditional MLE and the Bayesian estimators, which demonstrate strong convergence, the SLWE converges fairly quickly to the true value, and it is able to just as quickly “unlearn” the learning model trained from the historical data in order to adapt to the new data.

In particular, the SLWE utilizes the principles of learning which are used in stochastic Learning Automata (LA) algorithms, such as the L_{RI} scheme. Since the SLWE method is central to the work done in this thesis, we will discuss the SLWE, in greater detail, in this section and will proceed to explain how it can be used for classification problems.

2.5.1 Learning Automata

Learning Automata (LA) is an adaptive learning model that operates in random environments. Research in LA began with the remarkable works of Tsetlin [36], and the field has been surveyed by Narendra and Thathachar [22, 23].

A LA learns the optimal action out of a set of possible actions through repeated interactions with the random environment. The environment responds to the chosen action by producing an output, which is probabilistically related to the chosen action. The actions are chosen based on specific action probabilities, which are updated at each time instant, by considering the response received from the environment, in order to improve the learning performance.

The Linear Reward-Inaction (L_{RI}) scheme is one of the LA schemes, which was first introduced by Norman [25]. The basic idea of this method is to refrain from updating the probabilities whenever an unfavorable response is received from the environment. However, when a Reward response is received from the environment for a specific action, $\alpha(n)$, the corresponding probability is increased by the following updating algorithm:

$$p_i(n+1) \leftarrow \lambda p_i(n) \quad \text{if } \alpha(n) = \alpha_{j \neq i} \quad \beta(n) = 0 \quad (2.17)$$

$$\leftarrow 1 - \lambda \sum_{j \neq i} p_j(n) \quad \text{if } \alpha(n) = \alpha_i \quad \beta(n) = 0, \quad (2.18)$$

where $\beta(n)$ corresponds to the output of the environment at time n . Typically, $\beta(n) = 0$ indicates that a favorable result was obtained for the corresponding action $\alpha(n)$, and λ is a user-defined reward parameter, $0 < \lambda < 1$.

There is a close connection between LA schemes and underlying PR problems. For example, actions in the learning machine can be considered to be analogous to the various classes in the PR problems that each given sample can be assigned to. Using the training samples, the LA learns to assign new data to the most appropriate class considering the determined optimal action. The learning scheme can also be related to estimation methods in which the distribution function of a parameter is estimated at each moment based on the observed instances.

2.5.2 Model for SLWE

As mentioned, the SLWE is an estimator method based on the theory of LA and it estimates the parameters of a binomial/multinomial distribution when the underlying distribution is non-stationary. In non-stationary environments, the SLWE updates the estimate of the distribution's probabilities at each time-instant based on the new observations. The updating is achieved by a *multiplicative* rule, similar to the linear action probability updating scheme described in Eqs. (2.17) and (2.18). The estimation model for binomial and multinomial distributions are explained in the following sections.

2.5.3 Weak estimators of Binomial Distributions

The binomial distribution is defined by two parameters, namely, the *number* of Bernoulli trials, and the parameter characterizing *each* Bernoulli trial. The objective of the SLWE is to estimate the *Bernoulli* parameter for each trial based on the stochastic learning methods. Consider X as a random variable of a binomial distribution, which can take the value of either '1' or '2'. We assume that X obeys the distribution S , where $S = [s_1, s_2]^T$, and s_1 and s_2 indicate the probabilities of X taking on the value of either '1' or '2' respectively.

In other words,

$$\begin{aligned} X &= \text{'1' with probability } s_1 \\ &= \text{'2' with probability } s_2, \end{aligned}$$

where, $s_1 + s_2 = 1$.

In order to estimate s_i for $i = 1, 2$, the SLWE maintains a running estimate $P(n) = [p_1(n), p_2(n)]^T$ of S , where $p_i(n)$ is the estimate of s_i at time ' n ', for $i = 1, 2$. The value of $p_i(n)$ is adapted to the receiving data at time ' n ' using the following multiplicative scheme:

$$p_1(n+1) \leftarrow \lambda p_1(n) \quad \text{if } x(n) = 2 \quad (2.19)$$

$$\leftarrow 1 - \lambda p_2(n) \quad \text{if } x(n) = 1, \quad (2.20)$$

where $x(n)$ indicates the observed data at time step ' n ', and λ is a user-defined weak estimation learning constant, $0 < \lambda < 1$, and $p_2(n+1) \leftarrow 1 - p_1(n+1)$. The authors of

[27] provided a formal theory so as to infer that the mean of vector P , which estimates S from Eqs. (2.19) and (2.20), converges exactly to S . This result is presented in Theorem 1 below.

Theorem 1. Let X be a binomially distributed random variable, and $P(n)$ be the estimate of S at time ‘ n ’. Then, if $P(n)$ obeys Eqs. (2.19) and (2.20), $E[P(\infty)] = S$.

The authors of [27] also indicated that the distribution of $E[P(n+1)]$ can be derived from $E[P(n)]$ by means of a stochastic matrix. The mean of limiting distribution of $P(n)$, and its rate of convergence can be determined by examining this relation. It was shown that the mean of the distribution is not dependent on λ , while the rate of convergence is only a function of λ .

Theorem 2. If $P(n)$ obeys Eqs. (2.19) and (2.20), the expectation of the estimated distribution $P(n+1)$ depends on the estimation of distribution at time ‘ n ’ as $E[P(n+1)] = \mathbf{M}^T E[P(n)]$, where $\mathbf{M}$ is an ergodic Markov chain. Therefore, the limiting value of the expectation of $P(\cdot)$ converges to S , and the rate of convergence of P to S is a function of λ .

Theorem 3. Let $P(n)$ be the estimate of S at time ‘ n ’ obtained by Eqs. (2.19) and (2.20). Then, the algebraic expression for the variance of $P(\infty)$ is a function of λ .

The variance tends to zero as $\lambda \rightarrow 1$, which indicates that $P(n)$ obeys mean square convergence. The *maximum* and *minimum* values of the variance are obtained when $\lambda = 0$ and $\lambda = 1$ respectively.

Theoretically, these results are valid only as $n \rightarrow \infty$, but in practice, when λ is chosen from the interval $[0.9, 0.99]$, the convergence occurs after a relatively small value of ‘ n ’. In other words, the SLWE will be able to monitor changes, even if the Bernoulli parameters are switched in a short period of time (e.g. 50 steps). Therefore, there is no need to use the sliding window approach to keep track of the changes.

Experimental results for binomial random variables demonstrate the superiority of the SLWE over the MLE that uses a sliding window (MLEW). In order to demonstrate the superiority of the SLWE, the estimation algorithms were tested for a binomially distributed data stream with random occurrences of the variables for 400

Figure 2.1: Plot of the expected value of $p_1(n)$, at time n , which was estimated by using the SLWE and the MLEW, where $\lambda = 0.817318$ and the window size was 32 (duplicated from [27]).

time instances. The true underlying value of s_1 was obtained randomly for the first step, and was modified after every 50 steps using values drawn from a uniformly distributed random variable in $[0, 1]$. This experiment was repeated 1,000 times, and the ensemble average of estimation at every time step was recorded. In this experiment, the value of λ for the SLWE and the size of the window were randomly generated from the uniform distributions in $[0.55, 0.95]$ and $[20, 80]$, respectively.

Fig. (2.1) shows the plot of the ensemble average estimated probability of 1, p_1 , for the SLWE and the MLEW during this experiment, which demonstrates the SLWE adjusts to the changes much more quickly than the MLEW.

2.5.4 Weak estimators of Multinomial Distributions

Estimation of the parameters of a multinomial distribution using the SLWE scheme is similar to the binomial case, explained earlier. The *Number* of trials, and a probability *vector* specify the multinomial distribution, but in this case, the objective is to estimate the probability *vector* associated with a specific event.

Let X be a random variable of a multinomial distribution, which can take the values from the set $\{‘1’, \dots, ‘r’\}$ with the probability of S , where $S = [s_1, \dots, s_r]^T$ and $\sum_{i=1}^r s_i = 1$. In the other words: $X = ‘i’$ with probability s_i .

Consider $x(n)$ as a concrete realization of X at time ‘ n ’. In order to estimate the vector S , the SLWE maintains a running estimate $P(n) = [p_1(n), p_2(n), \dots, p_r(n)]^T$ of vector S , where $p_i(n)$ is the estimation of s_i at time ‘ n ’, for $i = 1, \dots, r$. The value of $p_i(n)$ is updated with respect to the coming data at each time instance, where Eqs. (2.21) and (2.22) show the updating rules:

$$p_i(n+1) \leftarrow p_i + (1-\lambda) \sum_{j \neq i} p_j \quad \text{when } x(n) = i \quad (2.21)$$

$$\leftarrow \lambda p_i \quad \text{when } x(n) \neq i. \quad (2.22)$$

Similar to the binomial case, the authors of [27] explicitly derived the dependence of $E[P(n+1)]$ on $E[P(n)]$, demonstrating the ergodic nature of the Markov matrix. The paper also derived two explicit results concerning the convergence of the expected vector $P(\cdot)$ to S , and the rate of convergence on the learning parameter, λ .

Theorem 4. Consider $P(n)$, the estimate of the multinomial distribution S at time ‘ n ’, which is obtained by Eqs. (2.21) and (2.22). Then, $E[P(\infty)] = S$.

Theorem 5. Consider $P(n)$, the estimate of the multinomial distribution S at time ‘ n ’, which is obtained by Eqs. (2.21) and (2.22). The expected value of P at time ‘ $n+1$ ’ is related to the expectation of $P(n)$ as $E[P(n+1)] = \mathbf{M}^T E[P(n)]$, where $\mathbf{M}$ is a Markov matrix. Further, every off-diagonal term of the stochastic matrix, $\mathbf{M}$, has the *same* multiplicative factor, $(1-\lambda)$, and the final solution of this vector *difference equation* is independent of λ .

Theorem 6. Consider $P(n)$, the estimate of the multinomial distribution S at time ‘ n ’, which is obtained by Eqs. (2.21) and (2.22). Then, all the non-unity eigenvalues of $\mathbf{M}$ are exactly λ , and therefore the convergence rate of P is fully determined by λ .

Theoretically, since the derived results are asymptotic, they are valid only as $n \rightarrow \infty$. However, in practice, by choosing λ from the interval $[0.9, 0.99]$, the convergence happens after a relatively small value of ‘ n ’. Indeed, if λ is as “small” as 0.9, the variation from the asymptotic value will be in the order of 10^{-50} after 50 iterations. In other words, the SLWE will provide good results even if the distribution parameters change after 50 steps. The reported experimental results in [27], demonstrated a good performance that were achieved by using the SLWE in dynamic environments.

The performance of the SLWE estimator was also investigated for multinomial datasets by performing simulations for multinomial random variables, where the parameters were estimated by both the SLWE and the MLEW. In these experiments a multinomial random variable, X , was considered, which could take any of the four different values, namely 1, 2, 3 or 4, whose probability was obtained randomly for the first step, and was changed after every 50 steps. Similar to the binomial case, the estimation was performed for 400 time instances and the experiment was repeated 1,000 times. The ensemble average of the estimate at every time step was recorded as an estimated value, P , and the Euclidean distance between P and S , $\|P - S\|$, indicated how close the estimated value was to the true value. The plots of the latter distance obtained from the SLWE and the MLEW are depicted in Figs. (2.2) and (2.3). The value of λ and the size of the windows were obtained randomly from a uniform distribution in $[0.9, 0.99]$ and $[20, 80]$, respectively.

From these figures, it can be observed that the MLEW and the SLWE converge to zero relatively quickly in the first epoch. However, this behavior is not present in successive epochs.

The MLEW is capable of tracking the changes of the parameters when the size of the window is small, or at least smaller than the intervals of the constant probabilities, but, it is not able to track the changes properly when the window size is relatively large. Since neither the magnitude nor the frequency of the changes are known *a priori*, this experiment demonstrates the weakness of the MLEW, and its dependence

Figure 2.2: Plot of the Euclidean norm of $P - S$ (or Euclidean distance between P and S), for both the SLWE and the MLEW, where λ is 0.957609 and the size of the window is 63, respectively (duplicated from [27]).

Figure 2.3: Plot of the Euclidean distance between P and S , where P was estimated by using both the SLWE and the MLEW. The value of λ is 0.986232 and the size of the window is 43 (duplicated from [27]).

on the knowledge of the input parameters.

2.6 Applications for Non-stationary Environments

The online data stream mining techniques have been applied in several key areas for the monitoring of streaming data, such as in spam-filtering [2, 39], network intrusion detection [9, 13, 20, 38], and time varying pattern recognition [28, 34]

De Oca *et al.* [9] proposed a nonparametric algorithm for network surveillance applications in non-stationary environments. They adapted the classic CUSUM change detection algorithm that uses a defined time-slot structure in order to handle time varying distributions. Hajji [13] developed a parametric algorithm for real time detection of network anomalies. He used stochastic approximation of the MLE function in order to monitor the non-stationary nature of network traffic and to detect unusual changes in it. Kim *et al.* [20] proposed a multi-chart CUSUM change detection procedure for the detection of DOS attacks in network traffic. Robinson *et al.* [31] proposed a method for monitoring and detecting behavioral changes from an event stream of patient actions.

On the other hand, the SLWE approach has been used successfully in a variety of real-life ML applications specifically those involving estimating binomial/multinomial distribution in non-stationary environments. Rueda and Oommen [32] utilized the SLWE approach for data compression in non-stationary environments, in which the SLWE was applied for an adaptive single-pass encoding process to estimate and update the probabilities of the source symbols. It was also shown in [27] that using the weak estimator for distribution change detection in non-stationary environments surpassed the performance of the MLE method. Oommen and Misra [26] applied the weak-estimation learning scheme to propose a new fault-tolerant routing approach for mobile ad-hoc networks, named the WEFTR algorithm. They utilized the SLWE approach to estimate the probability of the delivery of packets among the available paths at any moment. The SLWE was also used by Stensby *et al.* [34] for language detection and tracking multilingual online documents. Zhan *et al.* [39] applied the SLWE for anomaly detection, specifically for the detection of spam emails, when the

underlying distributions changed with time. They employed the SLWE approach for spam filtering based on a naive Bayes classification scheme. Oommen *et al.* [28] also proposed a strategy for learning and tracking the user's time varying interests to find out the users' preferences in social networks. Most recently, Khatua and Misra [18] developed a controllable reactive jamming detection scheme, referred to as CURD, which applies the CUSUM-test and the weak estimation approach in order to estimate the probability of collisions in packet transmission.

2.7 Limitations of the Previous

In Oommen and Rueda's work [27] the power of the SLWE method was demonstrated in only *two*-class classification problems, and the classification was performed on non-stationary one-dimensional datasets. In their experiments, the SLWE was used to estimate the distribution probabilities of the single-pass source symbols in order to classify or detect the source of the arriving data. They performed *two-class* classification of non-stationary data by estimating the distribution probabilities on synthetic and real-life data sets.

The intention of this thesis is to study the performance of the SLWE for more complex classification schemes, to be discussed in Chapter 3. In this study, contrary to the investigated classification problem by Oommen and Rueda in [27], the classification will be performed on a multidimensional data stream, generated from more than two sources of data. The aim of the classification is to assign a label to each element in the data stream to indicate the source or the class that the element belongs to. In other words, we shall investigate the power of the SLWE method to *C-class* classification by estimating the *vectors* of the probabilities distributions from binomial or multinomial multidimensional data in the respective non-stationary environments.

2.8 Model for NSE (Unknown to the PR system)

The phenomenon of non-stationarity in data stream can occur in many different ways, and considering its behavior, it can be modeled by using different approaches. The

methods for modeling non-stationarity were first developed to analyze economic and financial time series, but in this thesis these concepts will be employed to deal with PR problems. The models we use follow the ones described by Narendra and Thathacher [22] and these are explained below.

2.8.1 Periodic Switching Environment (PSE)

A Periodic Switching Environment (PSE) consists of multiple stationary environments or states, in which after every T time instances, the environment changes state from Q_i to $Q_{(i+1) \bmod k}$, where k indicates the number of states. With respect to the available information, a PSE can be divided into two categories. The simpler case is when the periodic environment has a period, T , known *a priori*, and the other category corresponds to the periodic environment with unknown T , which makes the model more complex. We explain each of these cases below.

- **Period T Known:** In these environments, characterizing the unknown parameter of the data streams can be performed by a deterministic function of time. The changes in the parameter of these kind of environments evolve in a perfectly predictable manner.

Although this type of environment looks simple, it does have real-life applications. A typical example of these models is the weather prediction rule, that may vary significantly with the season. Another example is the pattern of the load of electricity demand during a day, which ascends in the morning as the activities start and is expected to decrease by the end of day. Fig. 2.4 demonstrates a typical PSE with the known period of 50, in which the distribution's probability stays the same for exactly 50 time instances, after which it switches to another probability.

- **Period T , Unknown:** Non-stationarity in these kind of environments takes place with a random period T , which leads to more complex modeling. In order to simplify the problem, it is assumed that the upper bound of the period is known. Fig. 2.5 demonstrate a sample of PSE with unknown value of T .

Figure 2.4: Graphical representation of the PSE model with 3 different states and with $T = 50$.

Figure 2.5: Graphical representation of the PSE model with 3 different states and an unknown value for T .

2.8.2 Markovian Switching Environment (MSE)

A Markovian Switching Environment (MSE) is one of the most popular nonlinear switching models, introduced initially by Hamilton [14]. This model is a composite of several stationary environments that are assumed to be the states of Markov chain. The MSE controls the changes by an unobservable state variable, which follows a first order Markov chain. In the MSE, the states of the environments are also the states of a Markov chain, and switching the states happens in a Markovian manner. In other words, the Markovian property determines if switching the state should take place or

Figure 2.6: Graphical representation of the MSE model with 4 states and $\alpha = 0.9$. All the transitions between the states occur with the probability of $\frac{0.1}{3}$.

not by considering its immediate past state.

In the simplest model for the MSE, the environment stays in the same state with the probability α , and typically switches to another environment's state with probability $(1 - \alpha)/(k - 1)$, where k indicates the number of composite environment's phases. Fig. 2.6 demonstrates an example of the MSE. Consider, for example, an environment with 4 states and $\alpha = 0.9$. The transition matrix for such a Markov chain that characterizes the environment is given below:

$$\begin{bmatrix} 0.9 & \frac{0.1}{3} & \frac{0.1}{3} & \frac{0.1}{3} \\ \frac{0.1}{3} & 0.9 & \frac{0.1}{3} & \frac{0.1}{3} \\ \frac{0.1}{3} & \frac{0.1}{3} & 0.9 & \frac{0.1}{3} \\ \frac{0.1}{3} & \frac{0.1}{3} & \frac{0.1}{3} & 0.9 \end{bmatrix}.$$

2.9 Conclusions

In this chapter we have surveyed, in some detail, the available literature on change detection and estimation methods. In general, there are two major categories of solutions available to deal with change detection:

- Approaches that learn the model from a chunk of data at regular intervals without considering change points. These approaches often use sliding window schemes or weighting methods in order to handle concept drift problems. From this category we reviewed the FLORA and ADWIN systems.
- Incremental approaches that infer change points and use new data to adapt the learning model trained from the historical streaming data. The SPC and SLWE schemes belong to this category.

As the source of the data stream could generate unlimited data, time and memory consumption are important constraints in the associated learning approaches. Since the SPC and SLWE do not require any data structure for change detection, one can conclude that these approaches are less time and memory consuming than the ADWIN and other sliding window approaches. In order to use the SPC and the ADWIN methods, one also has to assume predefined values for the threshold and the confidence bound to evaluate the drifts. However, the SLWE does not need to include any assumptions or invoke a hypothesis testing strategy for change detection.

Most of the data stream mining approaches have an estimator module in order to keep the statistics of the data distribution in non-stationary environments updated. We have argued that using “strong” estimators that converge with probability of 1 is inefficient for dynamic non-stationary environment. On the other hand, “weak” estimator approaches are able to rapidly unlearn what they have learned, in order to adapt to new observations. This feature of “weak” estimators and the linear computational complexity of the SLWE, make these approaches the most effective methods for estimation in non-stationary environments.

The SLWE has been successfully applied in a variety of applications that involve estimating distributions in non-stationary environments. Moreover, using the “weak”

estimators in PR experiments has provided more robust results in comparison with the MLE Methods [27].

The aim of this thesis is to derive SLWE methods to achieve PR for streams involving multi-class and multi-dimensional features.

Chapter 3

C-Class PR using SLWE

3.1 The PR Problem

In this chapter we study a classification problem in non-stationary environments, where sequential patterns are arriving and being processed in the form of a data stream that was generated from different sources with different statistical distributions. The classification of the data streams is closely related to the estimation of the parameters of the time varying distribution, and the associated algorithm must be able to detect the source changes and to estimate the new parameters whenever a *switch* occurs in the incoming data stream. In order to detect underlying changes in the data distributions and to accurately determine the source of each arriving data element, we utilize the SLWE estimation approach during the testing phase.

In Oommen and Rueda's work [27], the SLWE was used to estimate the distribution probabilities of the single-pass source symbols in order to classify or detect the source of the arriving data. They performed *two-class* classification of non-stationary data by estimating the distribution probabilities on synthetic and real-life datasets. In their work, the data was arriving as a stream of bits, which were drawn from two different sources. The intention of that classification problem was to detect the source that generated each symbol of the bit stream. In the training phase, the probability of the symbol '0' for each distribution was learned using an off-line MLE estimation over the training set, and each class was associated with a probability value. The

testing set arrived in the form of blocks and each block included a sequence of bits that was generated randomly from either of the sources with the same probability. However, contiguous blocks might have had different data distributions and belonged to different sources (classes). In the problem that they studied, the order of the blocks and their size were unknown to the classifier. Their solution involved classifying each bit read from the testing stream with respect to the minimum *Euclidean* distance between the estimated distribution probability of ‘0’ and the learned probability of ‘0’ for the two classes obtained during the training phase.

In next sections, we propose and study different types of classification problems in non-stationary environments, which is the main direction of this thesis. In each section, we present the results of the simulation runs and discuss the obtained results. In Sections 3.3 and 3.4 the method will be applied to binomial and multinomial datasets respectively, and the obtained results will be compared with those obtained if we had used the MLE. Finally, the conclusions drawn by reviewing the results are presented in Section 3.5.

3.2 New Problem and The Studied Model

In Oommen and Rueda’s work [27] the power of the SLWE method was demonstrated in only *two*-class classification problems, and the classification was performed on non-stationary *one*-dimensional datasets. The intention of this chapter is to study the performance of the SLWE with more complex classification schemes, which are going to be discussed in the following paragraphs.

Analogous to the classification model explained in Section 3.1, we consider the scenario where the stream of data generated from the different sources, each with their own distinct probability distributions, arrives. In this section, contrary to the classification problem previously investigated by the authors of [27], the data stream is multidimensional, and it can be potentially generated from more than two sources of data. The aim of the classification is to assign a label to each element in the data stream so as to indicate the source or the class that the element belongs to. In other words, in our experiments the SLWE method is utilized for *C*-class classification by

estimating the *vectors* of the probability distributions from binomial or multinomial multidimensional data in the respective non-stationary environments. The learning updating rule in Eqs. (2.19) and (2.20) has a user-defined coefficient, λ . However, as shown in [27] and explained in Section 2.5.4, the convergence of this rule is independent of the value of the learning coefficient, λ . Further, only the variance of the estimate is controlled by λ , and the accuracy of the SLWE-based classifier is also independent of λ . To be consistent with the LA literature, we have set the value of λ to be 0.9 for all our SLWE-based experiments.

To evaluate the performance of the SLWE-based classifier, we investigate this problem for various synthetic binomial and multinomial datasets separately. We have also estimated the probability vector of the distributions by following the traditional MLE with a sliding window scheme (i.e., the MLEW), and have used the results to classify the arriving elements of the data stream in order to show the superiority of the performance of the SLWE over the MLEW. All the experiments were performed on a 2.2 GHz Intel Core i7 machine with 16 Gigabyte main memory, and our classifier algorithms were set up and run in the *MATLAB*® 7.12.0 environment.

3.3 Binomial Vectors: SE and NSE

In the case of synthetic d -dimensional binomial data with C different categories, the classification problem was defined as follows. Given a stream of bit vectors, which are generated from C different periodically switching sources (classes), say, $S_1, S_2, \dots, S_C$, the aim of the classification task is to assign a label to each element in the data stream, which indicates the source or class that the element probably belongs to.

A d -dimensional binomial dataset is characterized by ' d ' binomial distributions and it is exemplified by a stream of elements, and each data element is represented as a binary bit *vector* $X = \{x_1, x_2, \dots, x_d\}$, and where each $x_i \in \{0, 1\}$. Based on this description, a d -dimensional probability vector is assigned to each class, say, $S_{11}, S_{12}, \dots, S_{1C}$, which demonstrates the probability of '0' for the distribution in each dimension.

To train the classifier, a training set was generated using C binomial distributions,

where the probabilities of ‘0’ for the distributions were $S_{11}, S_{12}, \dots, S_{1C}$, respectively. These labeled training set elements were then utilized to achieve the MLE estimation of the probability of ‘0’ for each class in an off-line mode, which are denoted by $\hat{S}_{11}, \hat{S}_{12}, \dots, \hat{S}_{1c}$.

In the testing phase, we are given the stream of unlabeled samples from different sources arriving in the form of a PSE, in which, after every T time instances, the data distribution and the source of the data might change. The aim of the classification is to identify the source of the elements arriving at each time step by using the information in the detected data distribution.

To achieve this class labeling, the SLWE estimates the probabilities of ‘0’ in all the ‘ d ’ dimensions, which we refer to as $P_1(n)$. More explicitly:

$$P_1(n) = [p_{11}(n), \dots, p_{1d}]^T, \text{ where, } p_{1i} = SLWE(s_{1i}).$$

The reader will recall that by virtue of the notation we use, s_{1i} is the probability of ‘0’ in the i^{th} dimension, and s_{2i} is the probability of ‘1’, and the probability vector that has the minimum distance to the estimated probability vector of ‘0’, is chosen as a label of the observed element. The probability distance between the learned SLWE probabilities and the MLEW values estimated during training can be computed using the Kullback-Leibler (KL) divergence measure [33], which qualifies the distance between two probability distributions, and so it can be used to assign the nearest class to the current estimated distribution.

Thus, based on the SLWE classifier, the n^{th} element read from the test set is assigned to class S_j , where

$$j = \arg_i \min KL(\hat{S}_{1i} || P_1(n)), \quad (3.1)$$

where, if $U = [u_1, \dots, u_d]^T$, and $V = [v_1, \dots, v_d]^T$, the KL divergence is:

$$KL(U || V) = \sum_i u_i \log_2 \frac{u_i}{v_i}. \quad (3.2)$$

In this section, the two classifiers, the MLEW and SLWE, are tested for different binomial scenarios in different non-stationary environments. The classification of the binomial dataset has been tested extensively for numerous multidimensional

distributions, but only a subset of the final results are cited here, in the interest of space.

In order to carry out the experiments for this section, various datasets were generated. The generation method used was inspired by Oommen and Rueda [27] who also generated different sources of data with distinct probabilities for the random distribution. In the following section we will investigate datasets that were generated from only *two* different sources, followed by the investigation of the data streams generated based on C different sources.

3.3.1 Binomial Vectors: $d=2-6$, 2-class

In the first set of experiments, the classifiers were tested for different binomial datasets, starting with the simplest scenario involving *two* different classes in a *two*-dimensional (i.e., $d = 2$) space. We tested the classifiers in the periodic environment in which the period of switching from one source of data to the second and vice versa, T , was either fixed or chosen randomly.

First, we performed this experiment on different test sets with various *known* periods, $T = 50, 100, \dots, 500$, and for each value of T , 100 experiments were done. The resulting accuracies were averaged over the experiments to minimize the variance of the estimate. In these problems, the value of T and the switching time were unknown to the SLWE. The window of size w used for the MLEW, was centered around T , and was computed as the nearest integer of a randomly generated value obtained from a uniform distribution $U[\frac{T}{2}, \frac{3T}{2}]$. Fig. 3.1 shows a plot of the data distribution's probability of '0' in two dimensions when $T = 50$, and when it involves two different sources.

Secondly, we repeated the above experiment for the test sets with varying values of T . In these test sets, T was randomly generated from $U[\frac{w}{2}, \frac{3w}{2}]$, where w was the width used by the MLEW. The probability of '0' for two dimensions of the periodic test sets with unknown T and $w = 100$ is shown in Fig. 3.2. In both of these cases the MLE had the additional advantage of having some *a priori* knowledge of the test set's behavior, while the SLWE utilized the same conservative value of $\lambda = 0.9$.

Figure 3.1: An example of the true underlying probability of ‘0’, S_1 , for the first and second dimensions of a test set. The data was generated using two different sources in which the period of switching, T , was 50.

For the results which we report, (other cases leading to similar results are not reported here in the interest of space), the specific values of S_{11} and S_{12} for the 2-dimensional dataset were randomly set to be $S_{11} = [0.5265, 0.8779]^T$ and $S_{12} = [0.1336, 0.6626]^T$, which were assumed to be unknown to the classifiers. The results obtained are provided in Table 3.1, from which we see that classification using the SLWE was uniformly superior to classification using the MLEW. For example, when $T = 200$ the MLE-based classification resulted in the accuracy of 0.7537, while SLWE-based classification performed significantly better with the accuracy of 0.9701. The results of the classification in periodic environment with a varying T chosen randomly from $[50, 150]$ were also similar to the fixed $T = 100$ case, as the classifier achieved the

Figure 3.2: An example of the true underlying probability of ‘0’, S_1 , at time “ n ”, for the first and second dimensions of a test set which was generated with two different sources with a random switching period $T \in [50, 150]$.

accuracy of 0.9505 and 0.9530 in the first and second environments, respectively. We also observe that the accuracy of the classifier increased with the switching period, as is clear from Fig. 3.3.

The experiment described here was repeated on different 2-class datasets with different dimensionalities. These sets were generated randomly based on random vectors with different random distribution probabilities involving 3, 4 and 5 dimensions. The results obtained are shown in Tables 3.2-3.4. The advantage of the SLWE over the MLEW is consistent. For example, when $T = 250$, the MLEW achieved the accuracy of 0.7565, while the SLWE resulted in the accuracy of 0.9729. Similarly for the 3-dimensional data, the MLE-based classifier resulted in the accuracy of 0.7562 and

Figure 3.3: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 2-dimensional dataset with different switching periods, as described in Table 3.1.

the SLWE achieved significantly better results with the accuracy of 0.9859.

In the second set of experiments, we performed a detailed analysis of the SLWE-based classifier relative to the dimensions of the datasets. In order to compare and analyze the performance of the SLWE-based classifier on datasets with different dimensions, the classification procedure explained above was repeated 10 times over different datasets with fixed dimensions, and the ensemble average of the accuracies was obtained over these datasets. In each experiment, the classifiers were tested on a periodic environment with fixed periodicities, $T = 50, 100, \dots, 500$. For each value of T , an ensemble of 100 experiments was performed. The obtained results are shown in Figs. 3.7 and 3.8. Similar to the previous experiments we can see that the accuracy of the classifier increased with the switching period and it is also evident that for the same switching period, when the dimensionality of the dataset is higher the classifiers can process the data more efficiently. For example, in the case of $T = 150$

T	MLEW	SLWE
50	0.7417	0.9179
100	0.7585	0.9530
150	0.7516	0.9647
200	0.7537	0.9701
250	0.7565	0.9729
300	0.7577	0.9763
350	0.7690	0.9772
400	0.7646	0.9783
450	0.7449	0.9793
500	0.7674	0.9804
Random $T \in (50, 150)$	0.7387	0.9505

Table 3.1: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 2-dimensional data streams generated by two different sources.

the SLWE-based classifier resulted to the average accuracy of 0.9704 over several different two-dimensional datasets, while with more useful information in 5-dimensional datasets, it yielded better results with the accuracy of 0.9769.

3.3.2 Binomial Vectors: $d=2-6$, C-class

In this section, we report the results of extending experiments described in Section 3.3.1 for more complex datasets that had more than two distinct classes. In the case of C -class binomial classification, the classification problem was the following: Given a stream of bit vectors, which are generated from C different classes, say, $S_1, S_2, \dots, S_C$, the aim of the classification task is to assign a label to each element in the data stream which indicates the source or class that the element probably belongs to.

The current set of experiments involve testing the classifiers for different binomial datasets consisting of *three* distinct classes in a *two*-dimensional (i.e. $d = 2$) space. The classifiers were tested in periodic environments with either a fixed or an unknown T . For the particular results which we report, the specific values of S_{11} , S_{12} and S_{13} for the 2-dimensional dataset were randomly set to be $S_{11} = [0.0232, 0.3190]^T$, $S_{12} = [0.1711, 0.6482]^T$ and $S_{13} = [0.5080, 0.3823]^T$, which were assumed to be unknown to the classifiers. The experiments were conducted for numerous other data sets and the results obtained were identical.

T	MLEW	SLWE
50	0.7583	0.9337
100	0.7647	0.9660
150	0.7420	0.9769
200	0.7533	0.9818
250	0.7562	0.9859
300	0.7553	0.9877
350	0.7524	0.9894
400	0.7515	0.9902
450	0.7545	0.9911
500	0.7555	0.9918
Random $T \in (50, 150)$	0.7674	0.9666

Table 3.2: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 3-dimensional data streams generated by two different sources.

T	MLEW	SLWE
50	0.7551	0.9327
100	0.7571	0.9631
150	0.7687	0.9745
200	0.7568	0.9798
250	0.7451	0.9823
300	0.7667	0.9850
350	0.7846	0.9871
400	0.7463	0.9876
450	0.7585	0.9886
500	0.7662	0.9896
Random $T \in (50, 150)$	0.7640	0.9640

Table 3.3: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 4-dimensional data streams generated by two different sources.

Figure 3.4: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 3-dimensional dataset with different switching periods, as described in Table 3.2.

Figure 3.5: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 4-dimensional dataset with different switching periods, as described in Table 3.3.

T	MLEW	SLWE
50	0.7493	0.8812
100	0.7604	0.9158
150	0.7588	0.9267
200	0.7575	0.9329
250	0.7580	0.9366
300	0.7595	0.9386
350	0.7527	0.9397
400	0.7427	0.9415
450	0.7496	0.9417
500	0.7567	0.9424
Random $T \in (50, 150)$	0.7697	0.9138

Table 3.4: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 5-dimensional data streams generated by two different sources.

Figure 3.6: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 5-dimensional dataset with different switching periods, as described in Table 3.4.

We present here only a single set of results in the interest of brevity.

Figure 3.7: Plot of the accuracies of the SLWE classifier for different binomial datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 3.5.

d	50	100	150	200	250	300	350	400	450	500
2	0.9258	0.9591	0.9705	0.9764	0.9799	0.9821	0.9837	0.9852	0.9860	0.9863
3	0.9363	0.9681	0.9787	0.9838	0.9870	0.9893	0.9906	0.9919	0.9927	0.9934
4	0.9339	0.9671	0.9779	0.9835	0.9869	0.9888	0.9902	0.9913	0.9921	0.9931
5	0.9331	0.9652	0.9769	0.9824	0.9856	0.9877	0.9897	0.9909	0.9917	0.9925

Table 3.5: The results obtained from testing classifiers which used the SLWE for different binomial datasets with different dimensions, d , which were generated with a fixed switching period of $T = 50, 100, \dots, 500$.

Fig.3.9 shows an example of the test stream for the explained data distribution's probability of '0' in two dimensions when $T = 50$. An example of a test set generated from the same distribution in a periodic environment with random T is shown in Fig. 3.10. The results obtained are shown in Table 3.6 which indicates the superiority of the SLWE-based classifier. For the periodic environment with fixed $T = 100$ the MLEW reached the accuracy of 0.6642, while the SLWE obtained a far superior performance by obtaining an accuracy

Figure 3.8: Plot of the accuracies of the SLWE classifier for different binomial datasets each with a different switching period, T , and a different dimensionality, d . The numerical results of the experiments are shown in Table 3.5.

of 0.9112. In the case of the classification of datasets, which were generated from *three* different sources, similar to the previous experiments including *two* different classes, the accuracy of the classification increased with the switching period, and the classifier resulted in accuracies similar to those obtained for environments with fixed and randomly selected T , as can be seen from Fig. 3.11. For instance, in the environment with fixed $T = 100$ the SLWE-base classifier achieved the accuracy of 0.9112, similar to the resulted accuracy of 0.9141 in the periodic environment with a the varying T chosen randomly from $[50, 150]$.

Analogues to the previous section, the experiment described above was repeated on different 3-classes datasets with different dimensionalities, which were generated randomly based on random vectors with different random distribution probabilities involving 3, 4 and 5 dimensions. The results obtained are shown in Tables 3.7-3.9.

In order to investigate the performance of the SLWE on 3-class datasets with different dimensions, similar to the previous section the procedure was repeated 10 times over distinct datasets with fixed dimensions, and the ensemble average of the accuracies has been reported below. In each experiment, the classifiers were tested on a periodic environment with fixed periodicities, $T = 50, 100, \dots, 500$. Fig. 3.15 displays the obtained results from which

(a)

(b)

Figure 3.9: An example of the true underlying probability of ‘0’, S_1 , at time “ n ”, for the first and second dimensions of a test set which was generated with three different sources with a random switching period $T \in [50, 150]$.

we can see that the classification accuracy increased with the switching period, and for a consistent periodicity, the classifier provided better performance on datasets with a higher dimensionality. For example, when $T = 250$, the average accuracy of classification using SLWE on various two-dimensional datasets was 0.9275, while it yielded better results on 5-dimensional datasets with the accuracy of 0.9645.

In the third experiment of this section, we investigated the performance of the SLWE-based classifier relative to the datasets’ complexity. In this case, we considered the performance of the SLWE-based classifier over datasets that were generated with different number

(a)

(b)

Figure 3.10: An example of the true underlying probability of ‘0’, S_1 , at time “ n ”, for the first and second dimensions of a test set which was generated with three different sources with a random switching period $T \in [50, 150]$.

of classes, involving 2, 3, 4 and 5. The classification procedure was repeated 10 times over distinct datasets generated with fixed number of classes, and the ensemble average of the accuracies was reported as the result. In each experiment, the classifiers were tested on periodic environments with fixed periodicities $T = 50, 100, \dots, 500$. The results are shown in Figs 3.16 and 3.17, and as expected, we can see that the classifier provided superior results for the datasets which had less complexities or in other words for the ones which were generated from lesser number of classes. For example, when the test set was generated from *two* different sources and $T = 250$, the classifier achieved the accuracy of 0.9798, while the average accuracy of classification for the test sets generated from five different sources with

T	MLEW	SLWE
50	0.6632	0.8599
100	0.6642	0.9112
150	0.6791	0.9262
200	0.6463	0.9350
250	0.6694	0.9392
300	0.6696	0.9443
350	0.6622	0.9452
400	0.6604	0.9475
450	0.6892	0.9487
500	0.6703	0.9505
Random $T \in (50, 150)$	0.6941	0.9141

Table 3.6: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 2-dimensional data streams generated by three different sources.

Figure 3.11: Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 2-dimensional dataset with different switching periods, as described in Table 3.6.

T	MLEW	SLWE
50	0.6612	0.8768
100	0.7023	0.9251
150	0.6910	0.9380
200	0.6677	0.9462
250	0.6854	0.9520
300	0.6705	0.9542
350	0.6809	0.9561
400	0.6958	0.9594
450	0.6990	0.9596
500	0.6533	0.9603
Random $T \in (50, 150)$	0.6771	0.9214

Table 3.7: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 3-dimensional data streams generated by three different sources.

Figure 3.12: Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 3-dimensional dataset with different switching periods, as described in Table 3.7.

the same periodicity, was 0.7413. This is, of course, intuitively appealing.

T	MLEW	SLWE
50	0.6734	0.8894
100	0.6681	0.9371
150	0.6821	0.9533
200	0.6679	0.9601
250	0.6557	0.9657
300	0.6682	0.9689
350	0.6981	0.9714
400	0.6733	0.9732
450	0.6954	0.9746
500	0.6686	0.9755
Random $T \in (50, 150)$	0.6819	0.9368

Table 3.8: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 4-dimensional data streams generated by three different sources.

Figure 3.13: Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 4-dimensional dataset with different switching periods, as described in Table 3.8.

T	MLEW	SLWE
50	0.6548	0.8085
100	0.6694	0.8579
150	0.6642	0.8697
200	0.6529	0.8797
250	0.6494	0.8874
300	0.6728	0.8885
350	0.6843	0.8924
400	0.6652	0.8922
450	0.6591	0.8929
500	0.6762	0.8937
Random $T \in (50, 150)$	0.6771	0.8607

Table 3.9: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 5-dimensional data streams generated by three different sources.

Figure 3.14: Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 5-dimensional dataset with different switching periods, as described in Table 3.9.

Figure 3.15: Plot of the accuracies of the SLWE classifier for different datasets with different dimensions d over different values of T . The numerical results of the experiments are shown in Table 3.10.

d	50	100	150	200	250	300	350	400	450	500
2	0.8359	0.8943	0.9113	0.9232	0.9275	0.9303	0.935	0.9363	0.9375	0.9388
3	0.8473	0.8949	0.9129	0.9208	0.9253	0.9295	0.9314	0.934	0.935	0.9356
4	0.8867	0.9353	0.9505	0.9593	0.9638	0.9675	0.97	0.9712	0.9729	0.9738
5	0.8779	0.9348	0.9521	0.9603	0.9645	0.9677	0.9702	0.9716	0.9732	0.9735

Table 3.10: The results obtained from testing classifiers which used the SLWE for different binomial datasets generated from three classes with different dimensions, d , which were generated with a fixed switching period of $T = 50, 100, \dots, 500$.

3.4 Multinomial Vectors: SE and NSE

In this section, we will investigate the performance of the SLWE-based classifier over multinomial datasets, which is a generalization of the binomial case investigated earlier. In the case of synthetic d -dimensional multinomial data with C different categories, the classification problem was defined as follows. Given a stream of vectors, which are generated

Figure 3.16: Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.11.

C	50	100	150	200	250	300	350	400	450	500
2	0.9258	0.9591	0.9704	0.9763	0.9798	0.9821	0.9837	0.9852	0.9860	0.9863
3	0.8358	0.8943	0.9113	0.9231	0.9275	0.9303	0.9350	0.9363	0.9375	0.9387
4	0.7581	0.8147	0.8337	0.8445	0.8483	0.8526	0.8558	0.8577	0.8597	0.8608
5	0.6557	0.7073	0.7264	0.7362	0.7413	0.7432	0.7462	0.7477	0.7489	0.7534

Table 3.11: The results obtained from testing classifiers which used the SLWE for different 2-dimensional binomial datasets generated from different number of classes, C , which were generated with a fixed switching period of $T = 50, 100, \dots, 500$.

from C different periodically switching sources (classes), say, $S_1, S_2, \dots, S_C$, the aim of the classification task is to assign a label to each element in the data stream, which indicates the source or class that the element probably belongs to.

A d -dimensional multinomial dataset is characterized by ' d ' multinomial distributions and it is exemplified by a stream of elements. Each data element is represented as a multinomial *vector* $X = \{x_1, x_2, \dots, x_d\}$, where each x_i is a multinomially distributed random

Figure 3.17: Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.11.

variable, which takes on the values from the set $\{1, \dots, r\}$. Based on this description, ' r ' number of d -dimensional probability vectors are assigned to each class, say, $S_{i1}, S_{i2}, \dots, S_{iC}$, which demonstrates the probability of the value ' i ' for the distribution in each dimension, where $i \in \{1, \dots, r\}$.

To train the classifier, a training set was generated using C multinomial distributions, where the probabilities of the value ' i ' for the distributions were $S_{i1}, S_{i2}, \dots, S_{iC}$, respectively. These labeled training set elements were then utilized to achieve the MLE estimation of the probability of the value ' i ' for each class in an off-line mode, which are denoted by $\hat{S}_{i1}, \hat{S}_{i2}, \dots, \hat{S}_{ic}$, where $i \in \{1, \dots, r\}$.

In the testing phase, similar to the binomial case, we are given the stream of unlabeled samples from different sources arriving in the form of a PSE, in which, after every T time instances, the data distribution and the source of the data might change. The aim of the classification is to identify the source of the elements arriving at each time step by using the information in the detected data distribution.

To achieve this class labeling, the SLWE estimates the probabilities of each possible value, ‘ i ’, in all the ‘ d ’ dimensions, which we refer to as $P_i(n)$. More explicitly:

$$P_i(n) = [p_{i1}(n), \dots, p_{id}]^T, \text{ where, } p_{ij} = SLWE(s_{ij}).$$

The reader will recall that if we use the notation of Section 3.3, s_{ij} is the probability of the value ‘ i ’ in the j^{th} dimension, and the set of probability vectors that have the minimum distance to the estimated probability vectors of all the possible values, is chosen as a label of the observed element. The probability distance between the learned SLWE probabilities and the MLEW values estimated during training, similar to the binomial experiments, is computed using the KL divergence measure, and this measure is used to assign the nearest class to the current estimated distribution.

3.4.1 Multinomial Vectors: $d=2-6$, $r=4$, 2-class

In the first set of multinomial experiments, the classifiers were tested for different multinomial datasets, starting with the simplest scenario involving *two* different classes in a *two*-dimensional (i.e., $d = 2$) space, where in each dimension the data elements could take on four different values (i.e., $r=4$). The classifiers were tested in the periodic environment in which T , was either fixed or chosen randomly.

We performed this experiment on different test sets with various period values, $T = 50, 100, \dots, 500$, and for each value of T , 100 experiments were done. The resulting accuracies were averaged over the experiments. In these problems, the value of T and the switching time were unknown to the SLWE. The window of size w used for the MLEW, was centered around T , and was computed as the nearest integer of a randomly generated value obtained from a uniform distribution $U[\frac{T}{2}, \frac{3T}{2}]$.

Identical to what we did in Section 3.3.1, we repeated the above experiment for the test sets with varying values of T . In these test sets, T was randomly generated from $U[\frac{w}{2}, \frac{3w}{2}]$, where w was the width used by the MLEW. In both of the explained cases the MLE had the additional advantage of having some *a priori* knowledge of the test sets’ behavior, while the SLWE utilized the same conservative value of $\lambda = 0.9$.

For the results which we report, the specific values of S_{i1} and S_{i2} , for the 2-dimensional dataset were randomly set to be as in the following matrices which were assumed to be unknown to the classifiers.

$$S_{i1} = \begin{bmatrix} 0.2951, & 0.4066 \\ 0.3281, & 0.0627 \\ 0.0460, & 0.1791 \\ 0.3308, & 0.3516 \end{bmatrix}^T, \quad S_{i2} = \begin{bmatrix} 0.3139, & 0.4014 \\ 0.3162, & 0.2035 \\ 0.0517, & 0.3356 \\ 0.3182, & 0.0595 \end{bmatrix}^T.$$

The results obtained are provided in Table 3.12, from which it is evident that classification using the SLWE was uniformly superior to classification using the MLEW for multinomial datasets. For example, when $T = 200$ the MLE-based classification resulted in the accuracy of 0.7271, while SLWE-based classification performed significantly better with the accuracy of 0.9531. The results of the classification in periodic environments with a varying T chosen randomly from $[50, 150]$ were also similar to the fixed $T = 100$ case, as the classifier achieved the accuracy of 0.9333 and 0.9357 in the first and second environments, respectively. We also observe that the accuracy of the classifier increased with the switching period, as is clear from Fig. 3.18.

Figure 3.18: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 2-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.12.

T	MLEW	SLWE
50	0.7599	0.8963
100	0.7513	0.9357
150	0.7377	0.9476
200	0.7271	0.9531
250	0.7683	0.9581
300	0.7513	0.9607
350	0.7604	0.9622
400	0.7539	0.9638
450	0.7525	0.9644
500	0.7659	0.9653
Random $T \in (50, 150)$	0.7600	0.9333

Table 3.12: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 2-dimensional data streams generated by two different sources.

The experiment described here was repeated on different 2-class multinomial datasets with different dimensionalities, where each data element could take on four different values (i.e., $r=4$) in all dimensions. These sets were generated randomly based on random vectors with different random probabilities involving 3, 4 and 5 dimensions. The results obtained are shown in Tables 3.13-3.15, and as can be seen the advantage of the SLWE over the MLEW is consistent. For example, when $T = 250$, the MLEW achieved the accuracy of 0.7683, while the SLWE resulted in the accuracy of 0.9581. Similarly for the 3-dimensional data, the MLE-based classifier resulted in the accuracy of 0.7545, while the SLWE achieved significantly better results with a remarkable accuracy of 0.9842.

In the second set of experiments in the multinomial case, we performed a detailed analysis of the SLWE-based classifier relative to the dimensions of the datasets. In order to compare and analyze the performance of the SLWE-based classifier on multinomial datasets (i.e., $r=4$) with different dimensions, the classification procedure explained above was repeated 10 times over different datasets with fixed dimensions, and the ensemble average of the accuracies was obtained over these datasets. In each experiment, the classifiers were tested on a periodic environment with fixed periodicities, $T = 50, 100, \dots, 500$. For each value of T , an ensemble of 100 experiments was performed. The obtained results are shown in Figs. 3.22 and 3.23. Similar to the binomial experiments, we can see that the accuracy of the classifier increased with the switching period and it is also evident that for the same

T	MLEW	SLWE
50	0.7647	0.9295
100	0.7603	0.9657
150	0.7495	0.9751
200	0.7542	0.9809
250	0.7545	0.9842
300	0.7672	0.9867
350	0.7557	0.9884
400	0.7595	0.9896
450	0.7596	0.9905
500	0.7660	0.9914
Random $T \in (50, 150)$	0.7687	0.9646

Table 3.13: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 3-dimensional data streams generated by two different sources.

T	MLEW	SLWE
50	0.7624	0.9312
100	0.7566	0.9654
150	0.75	0.9777
200	0.7549	0.9827
250	0.7567	0.9859
300	0.7554	0.9886
350	0.7512	0.9900
400	0.7458	0.9911
450	0.7765	0.9924
500	0.7684	0.9930
Random $T \in (50, 150)$	0.7590	0.9650

Table 3.14: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 4-dimensional data streams generated by two different sources.

Figure 3.19: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 3-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.13.

switching period, when the dimensionality of the dataset is higher the classifiers can process the data more efficiently. For example, in the case of $T = 150$ the SLWE-based classifier resulted to the average accuracy of 0.9737 over several different two-dimensional datasets, while with more useful information in 5-dimensional datasets, it yielded better results with the accuracy of 0.9654.

3.4.2 Multinomial Vectors: $d=2-6$, $r=4$, C-class

In this section, the experiments described in Section 3.4.1 were repeated for more complex datasets that had more than two distinct classes. The first set of experiments in this section involve testing the classifiers for different multinomial datasets including three distinct classes in the periodic environments with fixed and unknown T . For the particular results which we report, the specific values of S_{i1} , S_{i2} and S_{i3} for the 2-dimensional dataset were randomly set to be as shown in the following matrices which were assumed to be unknown

Figure 3.20: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 4-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.14.

to the classifiers.

$$S_{i1} = \begin{bmatrix} 0.4642, & 0.2375 \\ 0.1472, & 0.2158 \\ 0.1696, & 0.2713 \\ 0.2190, & 0.2754 \end{bmatrix}^T, \quad S_{i2} = \begin{bmatrix} 0.3082, & 0.1038 \\ 0.3013, & 0.3826 \\ 0.1538, & 0.2365 \\ 0.2367, & 0.2771 \end{bmatrix}^T,$$

$$S_{i3} = \begin{bmatrix} 0.4673, & 0.1927 \\ 0.0120, & 0.2857 \\ 0.4344, & 0.2989 \\ 0.0863, & 0.2227 \end{bmatrix}^T.$$

The results obtained are provided in Table 3.17, from which, similar to the binomial case, we see that classification using the SLWE was uniformly superior to classification using the MLEW for multinomial dataset. For instance, when $T = 200$ the MLE-based classification resulted in the accuracy of 0.6654, while SLWE-based classification performed

T	MLEW	SLWE
50	0.7461	0.9342
100	0.7596	0.9672
150	0.7457	0.9774
200	0.7669	0.9836
250	0.7620	0.9867
300	0.7617	0.9892
350	0.7626	0.9906
400	0.7562	0.9916
450	0.7612	0.9927
500	0.7551	0.9934
Random $T \in (50, 150)$	0.7596	0.9671

Table 3.15: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 5-dimensional data streams generated by two different sources.

d	50	100	150	200	250	300	350	400	450	500
2	0.9255	0.9627	0.9737	0.9800	0.9834	0.9861	0.9877	0.9888	0.9897	0.9907
3	0.9263	0.9604	0.9731	0.9789	0.9830	0.9851	0.9870	0.9883	0.9894	0.9902
4	0.9274	0.9618	0.9748	0.9807	0.9840	0.9868	0.9868	0.9883	0.9898	0.9917
5	0.9310	0.9654	0.9769	0.9824	0.9862	0.9883	0.9899	0.9912	0.9922	0.9930

Table 3.16: The results obtained from testing classifiers which used the SLWE for different multinomial datasets generated from two classes with different dimensions, d , and three distinct possible values at each dimension, r . The test sets were generated with a fixed switching period of $T = 50, 100, \dots, 500$.

significantly better with the accuracy of 0.8792. The outcomes of the classification in periodic environment with a varying T chosen randomly from $[50, 150]$ were also similar to the fixed $T = 100$ case, with the accuracy of 0.8554 and 0.8572 in the first and second environments, respectively. We also observe that the accuracy of the classifier increased with the switching period, as is clear from Fig. 3.24.

The described experiment was repeated on different three-class multinomial datasets with different dimensionalities. The datasets were generated randomly based on random vectors with distinct random distribution probabilities including 3, 4 and 5 dimensions. The results are shown in Tables 3.18-3.20, and as can be seen the advantage of the SLWE over

Figure 3.21: Plot of the accuracies of the MLEW and the SLWE classifiers on a 2-class 5-dimensional multinomial dataset with different switching periods, as described in Table 3.15.

the MLEW is consistent. For example, when $T = 250$, the MLEW achieved the accuracy of 0.6739, while the SLWE resulted in the accuracy of 0.8851. Similarly, for the 3-dimensional data, the MLE-based classifier resulted in the accuracy of 0.6658 and the SLWE achieved notably better results with the accuracy of 0.8950.

In order to investigate the performance of the SLWE on three-class multinomial datasets with different dimensions, similar to the previous section the procedure was repeated 10 times over distinct datasets with fixed dimensions, and the ensemble average of the accuracies was reported as the result. In each experiment, the classifiers were tested on a periodic environment with fixed periodicities, $T = 50, 100, \dots, 500$. Figs. 3.28 and 3.29 demonstrate the obtained results from which we see that the classification accuracy increased with the switching period and for the consistent periodicity the classifier provides better performance on datasets with higher dimensionality. For example, when $T = 250$, the average accuracy of classification using SLWE on various two-dimensional datasets was 0.8851, while it yielded better results on 5-dimensional datasets with the accuracy of 0.9803.

T	MLEW	SLWE
50	0.6370	0.8066
100	0.6635	0.8554
150	0.6658	0.8695
200	0.6654	0.8792
250	0.6739	0.8851
300	0.6564	0.8863
350	0.6481	0.8893
400	0.6781	0.8919
450	0.6638	0.8925
500	0.6922	0.8921
Random $T \in (50, 150)$	0.6691	0.8572

Table 3.17: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 2-dimensional data streams generated by three different sources.

T	MLEW	SLWE
50	0.6564	0.8180
100	0.6476	0.8618
150	0.6782	0.8807
200	0.6783	0.8916
250	0.6658	0.8950
300	0.6698	0.9007
350	0.6702	0.9022
400	0.6855	0.9066
450	0.6726	0.9072
500	0.6719	0.9089
Random $T \in (50, 150)$	0.6709	0.8646

Table 3.18: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 3-dimensional data streams generated by three different sources.

T	MLEW	SLWE
50	0.6637	0.8522
100	0.6573	0.9040
150	0.6720	0.9231
200	0.6807	0.9316
250	0.6741	0.9378
300	0.6689	0.9438
350	0.6925	0.9459
400	0.6857	0.9471
450	0.6673	0.9484
500	0.6804	0.9501
Random $T \in (50, 150)$	0.6884	0.9088

Table 3.19: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 4-dimensional data streams generated by three different sources.

T	MLEW	SLWE
50	0.6951	0.9065
100	0.6741	0.9514
150	0.6748	0.9672
200	0.6819	0.9761
250	0.6678	0.9803
300	0.6863	0.9839
350	0.6563	0.9859
400	0.6655	0.9876
450	0.6678	0.9889
500	0.6776	0.9899
Random $T \in (50, 150)$	0.6807	0.9520

Table 3.20: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 5-dimensional data streams generated by three different sources.

Figure 3.22: Plot of the accuracies of the SLWE classifier for different multinomial datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 3.16.

d	50	100	150	200	250	300	350	400	450	500
2	0.8771	0.9295	0.9552	0.9608	0.9647	0.9667	0.9692	0.9701	0.9710	0.9711
3	0.8899	0.9439	0.9594	0.9691	0.9740	0.9771	0.9795	0.9816	0.9828	0.9844
4	0.9004	0.9487	0.9647	0.9728	0.9773	0.9803	0.9829	0.9844	0.9862	0.9873
5	0.9014	0.9511	0.9667	0.9746	0.9797	0.9831	0.9853	0.9872	0.9886	0.9897

Table 3.21: The results obtained from testing classifiers which used the SLWE for different multinomial datasets (i.e., $r=4$) generated from three classes with different dimensions, d , in an environment with a fixed switching period of $T = 50, 100, \dots, 500$.

In the third experiment of multinomial datasets, we investigated the performance of the SLWE-based classifier relative to the datasets' complexity. In this case, we investigated the performance of the SLWE-based classifier over multinomial datasets that were generated with different number of classes, involving 2, 3, 4 and 5 where each element could take on four different values. The classification procedure was repeated 10 times over distinct datasets generated with fixed number of classes, and the ensemble average of the accuracies was

Figure 3.23: Plot of the accuracies of the SLWE classifier for different multinomial datasets with different values for the switching period, T , over different values for the dimensionality, d . The numerical results of the experiments are shown in Table 3.16.

reported as the result. Periodic environment with fixed periodicities $T = 50, 100, \dots, 500$ were used to test the classifier in each experiment. The results are shown in Figs 3.30 and 3.31, in which it is evident that the classifier provides better results for less complex datasets, which were generated from fewer number of classes.

C	50	100	150	200	250	300	350	400	450	500
2	0.9201	0.9554	0.9679	0.9749	0.9786	0.9809	0.9826	0.9843	0.9846	0.9860
3	0.8771	0.9295	0.9469	0.9552	0.9508	0.9647	0.9667	0.9691	0.9701	0.9710
4	0.8119	0.8725	0.8934	0.9022	0.9089	0.9137	0.9161	0.9181	0.9195	0.9210
5	0.7513	0.8142	0.8375	0.847	0.8542	0.8592	0.8625	0.8639	0.8659	0.8670

Table 3.22: The results obtained from testing classifiers which used the SLWE for different 2-dimensional multinomial datasets (i.e., $r=4$) generated from different number of classes, C . The test sets were generated with a fixed switching period of $T = 50, 100, \dots, 500$.

Figure 3.24: Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 2-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.17.

3.5 Conclusions

In this chapter, we considered the problem of classification and detecting the source of data in periodic non-stationary environments. In Oommen and Rueda's work [27] the power of the SLWE method was demonstrated in only *two*-class classification problems, and the classification was performed on non-stationary *one*-dimensional datasets. In this chapter we studied the performance of the SLWE with more complex classification schemes. We performed our experiments on synthetic binomial *and* multinomial data streams, which were also multidimensional, and could have been potentially generated from more than two sources of data. In our experiments we used the SLWE method to estimate the *vector* of the probability distribution from binomial and multinomial multidimensional datasets in periodic non-stationary environments, where the periodicity was unknown to the classifier.

Experimental results for both binomial and multinomial random variables demonstrated the superiority of the SLWE-based *C*-class classification scheme over the classification

Figure 3.25: Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 3-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.18.

method which used the MLE. The results also suggested the classifier's performance improved with the switching period and it was evident that the accuracy of classification in a periodic environment with random switching period was very close to the case involving the data stream with a fixed switching period.

By investigating the outcome of the both binomial and multinomial experiments, we can see that when the dimensionality of datasets was higher the SLWE-based classifier achieved better accuracy. On the other hand, a larger number of classes degraded the performance of the classifier.

In the following chapter we will investigate the power of the SLWE for classification of data streams which were generated from two different classes, where unlike the work done in this chapter, the probability distribution of each class could possibly change with time as the data stream continues to appear.

Figure 3.26: Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 3-class 4-dimensional dataset with different switching periods, as described in Table 3.19.

Figure 3.27: Plot of the accuracies of the MLEW and the SLWE classifiers on a 3-class 5-dimensional multinomial (i.e. $r=4$) dataset with different switching periods, as described in Table 3.20.

Figure 3.28: Plot of the accuracies of the SLWE classifier for different datasets with different dimensions d over different values of T . The numerical results of the experiments are shown in Table 3.21.

Figure 3.29: Plot of the accuracies of the SLWE classifier for different multinomial datasets with different switching period, T , over different dimensionality, d . The numerical results of the experiments are shown in Table 3.21.

Figure 3.30: Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.22.

Figure 3.31: Plot of the accuracies of the SLWE classifier for different datasets with different complexity C over different values of T . The numerical results of the experiments are shown in Table 3.22.

Chapter 4

Online Classification Using SLWE

4.1 Introduction

In this chapter we shall study an *online* classification problem in non-stationary environments, in which the new instances arrive sequentially in the form of a data stream that was generated from various sources with potentially different statistical distributions. In the previous chapter we studied the classification problem involving sources with fixed stochastic properties. However, in the case that will be studied in this chapter, the classes' stochastic properties potentially vary with time as more instances become available.

In the model studied in Chapter 3, the training phase was performed in an *offline* manner, i.e., the training set was used to learn the stochastic properties of each class. Subsequently, the learned model was deployed and used to classify unlabeled data instances that appeared in the form of data streams. However, in many real life applications, it is not possible to analyze the stochastic model of the classes in an *offline* manner because of their dynamic natures. In fact, *offline* classifiers assume that the entire set of training samples can be accessed, as assumed in the previous chapter. However, as explained earlier in Section 2.1.4, in many real life applications, the entire training set is not available either because it arrives gradually or because it is not feasible to store it so as to infer the model of each class. Consequently, one is forced to make the classifier update the learning model using the newly-arriving training samples at any given time instance.

In this chapter, we present a novel *online* classifier scheme, that is able to update the learned model using a single instance at a time. Our goal is to predict the source of the

arriving instances as accurately as possible. In the following sections, we first define the general structure of the *Online* classifier, and then provide some experimental results on synthetic two-class binomial and multinomial datasets.

4.2 New Problem and the Online Model

In Section 3.1, we considered the scenario in which the stream of test data was generated from different sources, each with its own distinct probability distribution. The aim of the previous model was to determine the source that the arriving element belonged to. In this section, contrary to the classification problem previously investigated, the probability distribution of each class could possibly change with time as more instances become available. The aim of the present classification task in this model is to predict the source of each element, and to thereafter update the learning model using the SLWE estimator method from the newly available information arriving at each time instance.

Devising a classifier that deals with the data streams generated from non-stationary sources poses new challenges when compared to the previous model studied in Section 3.2, since the probability distribution of each class might change even as new instances arrive. An important characteristic of *online* learning is that the actual source of the data is discovered shortly after the prediction is made, which can then be used to update the learned model. In other words, an *online* algorithm includes three steps, which is described in Algorithm 1. First, the algorithm receives a data element. Using it and the currently learned model, the classifier predicts the source of that element. Finally, the algorithm receives the true class of the data, which is then used to update and refine the classification model. *Online* classifiers deal with data streams, in which the labeled and unlabeled samples are mixed. Therefore, the training, testing and deploying phases of the *online* classifiers are interleaved as they are applied to these types of data streams. This fascinating avenue is the domain of this chapter, and we have investigated the performance of SLWE-based classifiers to this new scheme.

In order to perform the *online* classification of the instances, we need to obtain the *a posteriori* probability of each class. Analogous to the previous classification model, we assign a label to the new unlabeled data element by comparing the obtained *a posteriori* probabilities and the estimated probability from the unlabeled test stream. Finally, after receiving the true label of the instance, the *a posteriori* probabilities are updated using the

Algorithm 1 Online Classification Algorithm

```

1:  $X \leftarrow$  data stream for classification
2:  $\hat{S} \leftarrow$  initialize posterior probabilities for each class
3: while there exists an instance  $x \in X$  do
    Step 1. Reciveing data:
4:   The model receives the unlabeled sample
5:   for all dimensions  $d$  of  $x$  do
6:      $p_i(n) \leftarrow$  Estimate the probability  $p_i$  using the SLWE
7:   end for
    Step 2. Prediction:
8:    $P(n) \leftarrow \{p_1(n), p_2(n), \dots, p_d(n)\}$ 
9:    $\hat{\omega} \leftarrow \arg_i \min KL(\hat{S}_i || P(n))$ 
    Step 3. Updating the model:
10:  After some delay,  $t_d$ , true category of the instance  $x$  is received
11:   $\omega \leftarrow$  true class of  $x$ 
12:  Update posterior probabilities  $\hat{S}$  using  $\omega$  and the SLWE
13: end while

```

algorithm explained in Eqs. (2.17) and (2.18).

In this classification model the training phase and the testing phase were performed simultaneously, and so the problem can be described as follows. We are given the stream of unlabeled samples generated from different sources arriving in the form of a PSE, in which, after every T time instances, the data distribution and the source of the data might change. In this case, in addition to the switching of the source of the data elements, the probability distribution of each source also possibly changes at random time instances. The aim of the classification is to predict the source of the elements arriving at each time step by using the information in the detected data distribution, and also the information of currant model of each class. In the *online* classification model, shortly after the prediction is made, the actual class label of the instance is discovered, which can be utilized to update the classification model to be used by the SLWE updating algorithm.

In the following sections, we present the results of this classifier on synthetic data. To assess the efficiency of the SLWE-based *online* classifier, we applied it for binomial and multinomial randomly generated data streams. We also classified the data streams' elements by following the traditional MLE with a sliding window, whose size is also selected randomly.

4.3 Binomial Data Stream

In the case of synthetic d -dimensional binomial data with *two* different non-stationary categories, the classification problem was defined as follows. We are given a stream of bit vectors, that were drawn from *two* different periodically switching sources (classes), say, S_1 and S_2 . Unlike the experiments done in previous chapter, their respective distributions could possibly change with time as the data stream continued to appear.

In order to perform the *online* classification, we assumed that we were provided with a small amount of labeled instances before the arrival of the data stream, which was used to obtain the *a posteriori* probability vector of ‘0’ for each class, say, $\hat{S}_{11}$ and $\hat{S}_{12}$. To perform the class labeling of the newly-arriving unlabeled element, the SLWE estimated the probability of ‘0’, which we refer to as $P_1(n)$. This probability allowed us to predict the source that the new instance belonged to. The probability vector that had the minimum distance to the estimated probability vector of ‘0’, was chosen as the label of the observed element. The probability distances between the learned SLWE probabilities, $P_1(n)$, and the SLWE estimation during training, $\hat{S}_{11}$ and $\hat{S}_{12}$, were computed using the KL divergence measure, using Eq. (3.2). Thus, based on the SLWE classifier, the n^{th} element read from the test set was assigned to the class, which had the minimum distance to the estimated probability, $P_1(n)$.

After some delay, t_d , at time $n + t_d$ we received the true category of the n^{th} instance, which was then used to update the corresponding probability learned in the training phase. The true class label for the n^{th} instance was read and added to the previously-trained model by updating the probabilities of the corresponding class, based on the updating algorithm in Eqs. (2.17) and (2.18). For example, Fig. 4.1 demonstrates a sample of an individual one-dimensional non-stationary class with four concept drift points. As can be seen, the probability of ‘0’, $\hat{S}_{11}$, was estimated using training samples arriving with delay of $t_d = 10$, by both the SLWE and the MLEW methods. For the SLWE, the value of λ was set to be 0.9, and the size of the window for MLEW was 80. It is evident that the SLWE was superior in tracking the probability to the MLEW method, and that it adjusted the corresponding probability at the concept drift points more quickly, which led to a better classification performance.

Our aim here is to confirm the efficiency of the SLWE-based classifier for binomial datasets, which were generated from two non-stationary sources. This classification problem

Figure 4.1: Plot of the averages for the estimates of s_{11} , obtained from the SLWE and MLEW at time n , using the available training samples that arrived with the delay of $t_d = 10$. The stochastic properties of each class switched four times at randomly selected times.

has been tested extensively for numerous distributions, but as before, only a subset of the final results are cited here. To carry out the experiments, various datasets were generated, where the generation method was similar to the one explained in Section 3.3. However, in order to generate non-stationary classes, the probabilities were changed several times randomly. It should be noted that in each period of length T , the probability of each class was consistent, and the concept drift could only occur at the end of the specified periods. For the experiments we report, we investigated datasets that were generated for 40 periods with different periodicities from only *two* different binomial sources, in which the probabilities of the distributions of each source changed at several randomly-selected switching points. Fig. 4.2 displays an example of the probability of ‘0’ generated from the above mentioned sources, where, $T = 100$, and the concept drift occurred at time instances of 600 and 2200.

In the first set of experiments we considered a scenario, in which a one-dimensional dataset was generated from *two* different binomial sources and the probabilities of the underlying distributions of the classes changed four times. We tested and trained the classifiers in the periodic environment in which the period of switching, T , was either fixed or chosen randomly. We performed the experiment on different test sets with various *known* periods (although this was unknown to the classifiers), $T = 50, 100, \dots, 500$, and for each

Figure 4.2: An example of the true underlying probability of ‘0’, S_1 , for a one-dimensional binary data stream. The data was generated using two different sources in which the period of switching was 100, and the stochastic properties of the classes switched two times.

value of T , 100 experiments were done and the average value of the accuracies was reported as the result. For the MLEW, the window size w was computed as the nearest integer of a randomly generated value obtained from a uniform distribution $U[\frac{T}{2}, \frac{3T}{2}]$, and the last observed w labeled elements were used to train the MLE model at each time instance.

For the particular results which we report here, the specific values of S_{11} and S_{12} for the one-dimensional dataset were randomly set to be 0.4735 and 0.1221 respectively, which switched five times to the random values of $\{0.8664, 0.4981, 0.6635, 0.3153\}$ and $\{0.3374, 0.1397, 0.8494, 0.4523\}$, respectively. The concept drift times were set randomly, and were assumed to be unknown to the classifiers. The results obtained are provided in Table 4.1, from which we see that the SLWE-based classifier was able to detect the concept drift and adapt the learning model to new elements uniformly better than the MLEW. For example, when $T = 200$, the MLE-based classification resulted in the accuracy of 0.7667, while the SLWE-based classification performed significantly better with the accuracy of 0.8695. From the last row in the table, we observe that the results of the classification in periodic environments with a varying T chosen randomly from $[50, 150]$ were also similar to the fixed $T = 100$ case, as the classifier achieved the accuracy of 0.8551 and 0.8532 in the first and second environments, respectively, while the corresponding accuracies of the MLEW were only 0.7633 and 0.7648 respectively.

T	MLEW	SLWE
50	0.7231	0.8125
100	0.7648	0.8532
150	0.7624	0.8556
200	0.7667	0.8695
250	0.7616	0.8834
300	0.7646	0.8731
350	0.7589	0.8667
400	0.7532	0.8810
450	0.7689	0.8765
500	0.7691	0.8673
Random $T \in (50, 150)$	0.7633	0.8551

Table 4.1: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying one-dimensional data streams generated by two non-stationary different sources.

Figure 4.3: Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a one-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.1.

T	MLEW	SLWE
50	0.7314	0.9083
100	0.7657	0.9473
150	0.7475	0.9606
200	0.7676	0.9666
250	0.7544	0.9705
300	0.7526	0.9729
350	0.7620	0.9744
400	0.7595	0.9753
450	0.7547	0.9780
500	0.7551	0.9788
Random $T \in (50, 150)$	0.7534	0.9523

Table 4.2: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 2-dimensional data streams generated by two non-stationary different sources.

We repeated the above procedure on different 2-class binomial datasets with different dimensionalities. These sets were generated randomly based on random vectors with different random distribution probabilities involving 2, 3 and 4 dimensions. The results obtained are shown in Tables 4.2-4.4, and as can be seen, the advantage of the SLWE over the MLEW is consistent. For example, when $T = 250$, the MLEW achieved the accuracy of 0.7616, while the SLWE resulted in the accuracy of 0.8834. Similarly for the 2-dimensional data, the MLE-based classifier resulted in the accuracy of 0.7544 and the SLWE achieved significantly better results with the accuracy of 0.9705.

In the final set of experiments, we performed a detailed analysis of the SLWE-based *online* classifier relative to the dimension of the datasets, and analyzed the performance of the SLWE-based classifier on datasets with different dimensions. The classification procedure explained above was repeated 10 times over different datasets with fixed dimensions, and the ensemble average of the accuracies was obtained over these datasets. In each experiment, the classifiers were tested on a periodic environment and stochastic property of each class was changed four times. For each value of T , an ensemble of 100 experiments was performed. The obtained results are shown in Figs. 4.7 and 4.8. It is evident that for the same switching period, when the dimensionality of the dataset is higher the classifiers can process the data more efficiently. For example, in the case of $T = 150$ the SLWE-based classifier resulted in the average accuracy of 0.8094 over several different one-dimensional

T	MLEW	SLWE
50	0.7182	0.8798
100	0.7469	0.9237
150	0.7416	0.9400
200	0.7652	0.9495
250	0.7512	0.9494
300	0.7450	0.9537
350	0.7476	0.9580
400	0.7597	0.9581
450	0.7555	0.9611
500	0.7551	0.9612
Random $T \in (50, 150)$	0.7504	0.9267

Table 4.3: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 3-dimensional data streams generated by two non-stationary different sources.

T	MLEW	SLWE
50	0.7145	0.8667
100	0.7496	0.9236
150	0.7491	0.9426
200	0.7671	0.9522
250	0.7544	0.9589
300	0.7574	0.9597
350	0.7515	0.9629
400	0.7506	0.9651
450	0.7557	0.9670
500	0.7498	0.9705
Random $T \in (50, 150)$	0.7512	0.9357

Table 4.4: The ensemble results for 100 simulations obtained from testing binomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 4-dimensional data streams generated by two non-stationary different sources.

Figure 4.4: Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a 2-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.2.

datasets, while with more useful information in 4-dimensional datasets, it yielded better results with the accuracy of 0.9519.

d	50	100	150	200	250	300	350	400	450	500
1	0.7673	0.7972	0.8094	0.8152	0.8190	0.8235	0.8217	0.8265	0.8246	0.8233
2	0.8290	0.8723	0.8851	0.8951	0.9009	0.9031	0.9055	0.9055	0.9051	0.9058
3	0.8433	0.8883	0.9006	0.9087	0.9121	0.9176	0.9197	0.9216	0.9198	0.9205
4	0.8871	0.9343	0.9519	0.9596	0.9643	0.9671	0.9700	0.9716	0.9733	0.9742

Table 4.5: The results obtained from testing classifiers which used the SLWE for different binomial datasets with different dimensions, d , which were generated with a fixed switching period of $T = 50, 100, \dots, 500$, and the stochastic properties of each class switched to different values at four random time instances.

Figure 4.5: Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a 3-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.3.

4.4 Multinomial Data Stream

In this section, we report the results for simulations performed for *multinomial* data streams with *two* different non-stationary categories. The multinomial classification problem is a generalization of the binomial case introduced earlier. Here, the classification problem was defined as follows. We are given a stream of unlabeled multinomially distributed random d -dimensional vectors, which take on the values from the set $\{1, \dots, r\}$, and which are generated from *two* different periodically switching sources (classes), say, S_1 and S_2 . Each class was characterized with probability values, S_{i1} , S_{i2} , which demonstrate the probability of the value ‘ i ’, where $i \in \{1, \dots, r\}$. In this case, similar to the binomial case, the probabilities of the distributions of each class could possibly change with time as the data stream continued to appear.

Analogues to the binomial case, the multinomial data stream classification started with the estimation of the *a priori* probability of each possible value of ‘ i ’, in all the ‘ d ’ dimensions, for each class ‘ j ’ from the available labeled instances, which we refer to as $\hat{S}_{ij}$. To assign a

Figure 4.6: Plot of the accuracies of the MLEW and the SLWE binomial classifiers on a 4-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.4.

label to the newly arriving unlabeled element, the SLWE estimated the probabilities of each possible value ‘ i ’, in all the ‘ d ’ dimensions, from the unlabeled instances, which we refer to as $P_i(n)$. Thereafter, these probabilities were used to predict the class label that a new instance belonged to class ‘ j ’, with the probability vector of the $\hat{S}_j = \{\hat{S}_{1j}, \hat{S}_{2j}, \dots, \hat{S}_r\}$, that had the minimum distance to the estimated probability of $P = \{P_1(n), P_2(n), \dots, P_r(n)\}$, was chosen as the label of the observed element. The distances between the learned SLWE probabilities, $P_i(n)$, and the SLWE estimation during training, $\hat{S}_{ij}$ were computed using the KL divergence measure, using Eq. (3.2). Thereafter, after some delay, t_d , at time $n + t_d$ the algorithm received the true class of the n^{th} instance and used it to refine and update the true class probabilities. The true value of the category for the n^{th} instance was read and added to the previously trained model by updating the probability of the corresponding class based on the updating algorithm in Eqs. (2.17) and (2.18).

The classification procedure explained above was performed on multinomial data streams generated from two different classes where the probability of the distributions of each class

Figure 4.7: Plot of the accuracies of the SLWE classifier for different binomial datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 4.5.

switched four times. For the results which we report, each element of the data stream could take any of the four different values, namely 1, 2, 3 or 4. The specific values of S_{i1} and S_{i2} , were changed and set to random values four times at random time instances, which were assumed to be unknown to the classifiers. The results are shown in Table 4.6, and again, the uniform superiority of the SLWE over the MLEW is noticeable. For example, when $T=100$, the MLEW-based classifier yielded an accuracy of only 0.7443, but the corresponding accuracy of the SLWE-based classifier was 0.8012. We also notice that the results of the classification in periodic environments with a varying T chosen randomly from $[50, 150]$ were also similar to the fixed $T = 100$ case, as the classifier achieved the accuracy of 0.8092 and 0.8012 in the first and second environments, respectively. The results also show that the SLWE-based algorithm handles the concept drift and provides satisfactory performance.

The experiment explained above was repeated on different 2-class multinomial datasets with different dimensionalities. These sets were generated randomly based on random vectors with different random distribution probabilities involving 2, 3 and 4 dimensions and each element could take on four different values. The results obtained are shown in Tables

Figure 4.8: Plot of the accuracies of the SLWE classifier for different binomial datasets involved data from two non-stationary classes. Each dataset was generated with a different switching period, T , and a different dimensionality, d . The numerical results of the experiments are shown in Table 4.5.

4.7-4.9, from which we see that classification using the SLWE was uniformly superior to classification using the MLEW. For example, for the 2-dimensional data, when $T = 250$, the MLE-based classifier resulted in the accuracy of 0.7544 and the SLWE achieved significantly better results with the accuracy of 0.9706. Here the accuracy of the classifier, similar to the binomial case, increased with the dimensionality of the datasets as the classifiers could process the data more efficiently. For example, in the case of $T = 150$ the SLWE-based *online* classifier resulted in the average accuracy of 0.9181 over several different two-dimensional datasets, while with more useful information in 4-dimensional datasets, it yielded better results with the accuracy of 0.9569.

4.5 Conclusion

In this chapter we tackled the problem of classification in periodic non-stationary environments, where instances arrived sequentially in the form of a data stream with potentially

T	MLEW	SLWE
50	0.6786	0.7582
100	0.7443	0.8012
150	0.7476	0.8153
200	0.7595	0.8197
250	0.7514	0.8282
300	0.7509	0.8367
350	0.7587	0.8322
400	0.7598	0.8354
450	0.7635	0.8344
500	0.7574	0.8387
Random $T \in (50, 150)$	0.7523	0.8092

Table 4.6: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying one-dimensional data streams generated by two non-stationary different sources.

T	MLEW	SLWE
50	0.6913	0.8560
100	0.7402	0.9082
150	0.7463	0.9333
200	0.7480	0.9393
250	0.7478	0.9430
300	0.7397	0.9474
350	0.7486	0.9463
400	0.7525	0.9535
450	0.7554	0.9505
500	0.7518	0.9559
Random $T \in (50, 150)$	0.7436	0.9241

Table 4.7: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 2-dimensional data streams generated by two non-stationary different sources.

T	MLEW	SLWE
50	0.6912	0.8798
100	0.7422	0.9384
150	0.7596	0.9576
200	0.7490	0.9657
250	0.7630	0.9709
300	0.7609	0.9743
350	0.7515	0.9772
400	0.7485	0.9797
450	0.7663	0.9814
500	0.7570	0.9821
Random $T \in (50, 150)$	0.7504	0.9328

Table 4.8: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 3-dimensional data streams generated by two non-stationary different sources.

T	MLEW	SLWE
50	0.6906	0.8847
100	0.7535	0.9402
150	0.7557	0.9592
200	0.7532	0.9669
250	0.7550	0.9730
300	0.7531	0.9760
350	0.7522	0.9785
400	0.7460	0.9818
450	0.7572	0.9817
500	0.7562	0.9839
Random $T \in (50, 150)$	0.7526	0.9512

Table 4.9: The ensemble results for 100 simulations obtained from testing multinomial classifiers which used the SLWE (with $\lambda = 0.9$) and the MLEW for classifying 4-dimensional data streams generated by two non-stationary different sources.

Figure 4.9: Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a one-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.6.

d	50	100	150	200	250	300	350	400	450	500
1	0.8016	0.8432	0.8604	0.8653	0.8699	0.8730	0.8748	0.8778	0.8790	0.8798
2	0.8479	0.8999	0.9181	0.9273	0.9318	0.9357	0.9359	0.9387	0.9397	0.9430
3	0.8770	0.9316	0.9496	0.9574	0.9629	0.9660	0.9692	0.9718	0.9726	0.9737
4	0.8844	0.9387	0.9569	0.9656	0.9707	0.9741	0.9772	0.9788	0.9801	0.9817

Table 4.10: The results obtained from testing classifiers which used the SLWE for different multinomial datasets with different dimensions, d , which were generated with a fixed switching period of $T = 50, 100, \dots, 500$, and the stochastic properties of each class switched to different values at four random time instances.

time-varying probabilities that changed over time for each class. Contrary to the classification problem previously investigated in Chapter 3, in which a single learning model was used for the entire stream, in this chapter, we proposed an *online* classification approach that used a single training sample at any given time instance to learn the stochastic properties of each class.

Figure 4.10: Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 2-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.7.

The proposed *online* classification algorithm was used to perform the training and the testing simultaneously in three phases. In the first phase, the algorithm received a new unlabeled instance. After this, the scheme assigned a label to it based on the distributions' estimated probabilities using the SLWE. Finally, after a few time instances, the algorithm received the actual class of the instance and used it to update the training model by invoking the SLWE updating algorithm. Thereafter, the classification model was adjusted to the new available instances in an *online* manner.

We performed our proposed *online* classification on synthetic binomial *and* multinomial data streams, which were generated from two non-stationary sources of data. In our experiments we used the SLWE method to estimate the probability distribution from binomial and multinomial data streams in periodic non-stationary environments, where the statistical distribution of each class could possibly change with time as more instances became available. Experimental results for both binomial and multinomial random variables

Figure 4.11: Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 3-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.8.

demonstrated the efficiency of the SLWE-based *online* classifier as it demonstrated superior classification performance on data streams and it also handled the concept drifts in non-stationary environments. These results also indicated the uniform superiority of the SLWE-based classifier over the the classification scheme using a sliding window and the MLE method. As expected, from the experiment results we can see that when the dimensionality of datasets was higher the SLWE-based classifier could process the data more efficiently and achieved better a accuracy.

Figure 4.12: Plot of the accuracies of the MLEW and the SLWE multinomial classifiers on a 4-dimensional dataset generated from two non-stationary sources with different switching periods, as described in Table 4.9.

Figure 4.13: Plot of the accuracies of the SLWE classifier for different multinomial($r=4$) datasets with different dimensions, d , over different values of the switching periodicity, T . The numerical results of the experiments are shown in Table 4.10.

Figure 4.14: Plot of the accuracies of the SLWE classifier for different multinomial datasets involved data from two non-stationary classes. Each dataset was generated with a different switching period, T , and a different dimensionality, d . The numerical results of the experiments are shown in Table 4.10.

Chapter 5

Summary and Conclusion

This thesis introduced a general framework for C -class classification of binomial and/or multinomial data streams with concept drift. It also presented a new *online* classification method for data streams that were generated from non-stationary sources. We expect that the contributions presented can provide better insights into the understanding of C -class classification problems in non-stationary environments. This final chapter involves an overview of the obtained results followed by a few possible directions for future research.

5.1 Contributions

In this thesis we studied the problem of classification in non-stationary environments, where the data appears with fixed or random periodicities. Using the SLWE family of weak estimators, we adopted a scheme for classification of binomial *and* multinomial data streams, which were also multidimensional, and could have been potentially generated from more than two sources of data. In the schemes presented, the SLWE method was utilized to estimate the *vector* of the probability distribution from binomial and multinomial multidimensional datasets in periodic non-stationary environments, where the periodicity was unknown to the classifier.

Two different classification scenarios were considered in this thesis. Firstly, a scenario was studied in which the stream of data was generated from more than two binomial and multinomial sources, each with its own fixed stochastic properties, and where the source of data switched in a periodic non-stationary manner. Thereafter, we investigated a more

complex classification problem, where the classes' stochastic properties potentially varied with time as more instances became available.

In the first problem, we investigated the power of the SLWE-based classifier for classification and detecting the source of data in periodic non-stationary environments. A similar method was previously used for *one*-dimensional *two*-class classification problems. However, in Chapter 3 we studied the performance of the SLWE-based classifier with more complex classification schemes. We performed our experiments on synthetic binomial *and* multinomial data streams, which were also multidimensional, and which could have been potentially generated from more than two sources of data.

For the second problem, in Chapter 4, we proposed and tested an *online* classification scheme that can adaptively learn from data streams that change over time. Our method is based on using SLWE estimator modules, and it was used to perform the training and the testing simultaneously in three phases. In the first phase, the model learned from the available labeled samples, and also received a new unlabeled instance. Thereafter, the learned model predicted the class label of the observed unlabeled instance. In the third phase, after being informed of the true class label of the instance, the training model was adjusted to the new available instances by invoking the SLWE updating algorithm. Instead of using a single training model and counters to keep important data statistics, the introduced online classifier scheme provided a real-time self-adjusting learning model. The learning model utilized the multiplication-based update algorithm of the SLWE at each time instance as a new labeled instance arrived. In this way, the data statistics were updated every time a new element was inserted, without needing to rebuild its model when changes occurred in the data distributions.

The effectiveness of both algorithms and the superiority of the SLWE in both of these cases were demonstrated through extensive experimentation on a variety of datasets. In summary, we list here the conclusions drawn from the experimental results that we obtained in this Thesis:

- A main advantage of the incremental SLWE-based classifier is that it does not require any assumption about how fast or how often the stream changes. As opposed to this, the sliding window MLWE approach needs some *a priori* knowledge of the systems behavior, in order to consider the window size.

- The experimental results for both binomial and multinomial random variables demonstrated that the performance of the SLWE-based C -class incremental classifier was far superior than the obtained results from the sliding window classification approach, which used the MLE.
- For the case of online classification, where the probability of each class could switch in a periodic form, experimental results demonstrated the efficiency of the SLWE-based *online* classifier. These results also indicated that the SLWE-based incremental classifier was still superior to the classification scheme that used a sliding window and the MLE method.
- The results also suggested that the classifier's performance improved with the switching period. Further, it was evident that the accuracy of classification in a periodic environment with random switching period was very close to the case involving the data stream with a fixed switching period, which indicates that the algorithm is efficient no matter how fast or how often the stream would change.
- By examining the results obtained, we can see that when the dimensionality of datasets was higher the SLWE-based classifier achieved a superior accuracy. On the other hand, datasets generated from larger number of classes led to a more complex classification problem, and this degraded the performance of the classifier. Both of these are intuitively appealing conclusions.

5.2 Future Work

For future work, it would be interesting to see how the classification algorithms would perform in a real-life setting, by using real-life non-stationary data streams, instead of synthetic models. While it was demonstrated that the proposed algorithms provided good performance for synthetic datasets, it would be beneficial to perform experiments on real-world data streams as well.

In this thesis, for all the experiments conducted, we used the constant value of $\lambda = 0.9$ for the updating parameter of the SLWE. One direction for the future work would be that of adjusting this parameter depending on the performance of the classifier at each time instance. In fact, when the performance of classification drops significantly, it can

be inferred that a change in the data distribution has occurred. Thereafter, in order to “unlearn” the model, the updating parameter of the SLWE could possibly be increased.

Finally, another avenue for future work could be the development of a similar online classifier for other distributions, such as the Gaussian, exponential, gamma, and Poisson. It would be interesting to utilize the SLWE estimator to estimate the properties of other distributions such as their mean and variance, and to use the analogous classifiers for outlier detection and one-class-classification problems.

Bibliography

- [1] M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes: Theory and Application*. Prentice-Hall, Inc., Englewood Cliff, 1993.
- [2] S. Bickel and T. Scheffer. Dirichlet-enhanced Spam Filtering Based on Biased Samples. *Advances in Neural Information Processing Systems*, 19:161168, 2007.
- [3] A. Bifet. *Adaptive Learning and Mining for Data Streams and Frequent Patterns*. PhD thesis, Departament de Llenguatges i Sistemes Informatics, Universitat Politcnica de Catalunya, Barcelona Area, Spain.
- [4] A. Bifet and R. Gavalda. Kalman Filters and Adaptive Windows for Learning in Data Streams. In L. Todorovski and N. Lavrac, editors, *Proceedings of the 9th Discovery Science*.
- [5] A. Bifet and R. Gavalda. Learning From Time-changing Data With Adaptive Windowing. In *Proceedings SIAM International Conference on Data Mining*, volume 8, pages 443–448, 2007.
- [6] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [7] G. E. P. Box and G. M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day series in time series analysis and digital processing. Holden-Day, 1976.
- [8] J. Chen and A.K. Gupta. On Change-Point Detection and Estimation. *Communication in Statistics Simulation and Computation*, 30(3):665–697, 2001.

- [9] C. M. De Oca, D. R. Jeske, Q. Zhang, C. Rendon, and M. Marvasti. A CUSUM Change-Point Detection Algorithm for Non-stationary Sequences with Application to Data Network Surveillance. *The Journal of Systems and Software*, 83:12881297, 2010.
- [10] R. Duda, P. Hart, and D. Stork. *Pattern Recognition*. Wiley, second ed. edition, 2000.
- [11] J. Gama. *Knowledge Discovery From Data Streams*. Chapman & Hall/CRC, 2010.
- [12] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with Drift Detection. In AnaL.C. Bazzan and Sofiane Labidi, editors, *Advances in Artificial Intelligence SBIA 2004*, volume 3171 of *Lecture Notes in Computer Science*, pages 286–295. Springer Berlin Heidelberg, 2004.
- [13] H. Hajji. Statistical Analysis of Network Traffic for Adaptive Faults Detection. *IEEE Transactions on Neural Networks*, 16(5), 2005.
- [14] J. D. Hamilton. New Approach to the Economic Analysis of Non-stationary Time Series and the Business Cycle. *Econometrica*, 57:357–384, 1989.
- [15] G. Hulten and P. Domingos. Catching Up with the Data: Research Issues in Mining Data Streams. In *In Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2001.
- [16] A. K. Jain, R. Duin, and J. Mao. Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 2000.
- [17] R. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.
- [18] M. Khatua and S. Misra. CURD: Controllable Reactive Jamming Detection in Underwater Sensor Networks. *Pervasive and Mobile Computing*, 13:203–220, August 2014.
- [19] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proceedings of the International Conference on Very Large Data Bases*, page 180191. Morgan Kaufmann, 2004.
- [20] H. Kim, B. L. Rozovskii, and A. G. Tartakovsky. A Nonparametric Multichart CUSUM Test for Rapid Detection of Dos Attacks in Computer Networks. *International Journal of Computing & Information Science*, 2(3), 2004.

- [21] L. I. Kuncheva. Change Detection in Streaming Multivariate Data Using Likelihood Detectors. *IEEE Transactions on Knowledge and Data Engineering*, 25(5):1175–1180, 2013.
- [22] K. Narendra and M. Thathachar. *Learning Automata. An Introduction*. Prentice-Hall, 1989.
- [23] K. S. Narendra and M. Thathachar. Learning Automata: A Survey. *IEEE Transaction on System, Man, and Cybernetics*, SMC-4(4):323–334, 1974.
- [24] K. Nishida and K. Yamauchi. Detecting Concept Drift Using Statistical Testing. In *Proceedings of the 10th International Conference on Discovery Science*, page 264–269. Springer, 2007.
- [25] M. F. Norman. On the Linear Models with Two Absorbing Barriers. *Journal of Mathematical Psychology*, 5:225–241, 1968.
- [26] B. J. Oommen and S. Misra. A Fault-tolerant Routing Algorithm for Mobile Ad Hoc Networks Using a Stochastic Learning-based Weak Estimation Procedure. In *Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications.*, pages 31–37. IEEE, 2006.
- [27] B. J. Oommen and L. Rueda. Stochastic Learning-based Weak Estimation of Multinomial Random Variables and Its Applications to Pattern Recognition in Non-stationary Environments. *Pattern Recognition*, 39(3):328–341, 2006.
- [28] B. J. Oommen, A. Yazidi, and O. C. Granmo. An Adaptive Approach to Learning the Preferences of Users in a Social Network Using Weak Estimators. *Journal of Information Processing Systems*, 8(2), 2012.
- [29] R. Pears, S. Sakthithasan, and Yun S. Koh. Detecting Concept Change in Dynamic Data Streams. *Machine Learning*, 97(3):259–293, 2014.
- [30] A. S. Polunchenko and A. G. Tartakovsky. State-of-the-Art in Sequential Change-Point Detection. *Methodology and Computing in Applied Probability*, 14(3):649–648, September 2012.

- [31] W. N. Robinson, A. Akhlaghi, T. Deng, and A.R. Syed. Discovery and Diagnosis of Behavioral Transitions in Patient Event Streams. *ACM Transactions on Management Information Systems*, 3(1), 2012.
- [32] L. Rueda and B. J. Oommen. Stochastic Automata-based Estimators for Adaptively Compressing Files with Nostationary Distributions. *IEEE Transaction on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(5), 2006.
- [33] R. Sebastiao and J. Gama. A Study on Change Detection Method. In *Proceedings the 14th Portuguese Conference on Artificial Intelligence*, pages 353–364, Berlin, Heidelberg, 2009. Springer.
- [34] A. Stensby, B. J. Oommen, and O. C. Granmo. Language Detection and Tracking in Multilingua Documents Using Weak Estimators. In *Proceedings of the 2010 Joint IAPR International Conference on Structural, Syntactic, and Statistical Pattern Recognition, ser. SSPR&SPR'10*, pages 600–609, 2010.
- [35] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Elsevier Academic Press, second edition, 2003.
- [36] M. L. Tsetlin. *Automaton Theory and the Modelling of Biological Systems*. Academic Press, 1973.
- [37] G. Widmer and M. Kubat. Learning in the Presence of Concept Drift and Hidden Contexts. In *Machine Learning*, volume 23, pages 69–101, 1996.
- [38] N. Ye, S. Vilbert, and Q. Chen. Computer Intrusion Detection Through EWMA for Autocorrelated and Uncorrelated Data. *IEEE Transactions on Reliability*, 52(1):75–82, 2003.
- [39] J. Zhan, B. J. Oommen, and J. Crisostomo. Anomaly Detection in Dynamic Systems Using Weak Estimators. *ACM Transactions on Internet Technology*, 11(1), July 2011.