Eindhoven University of Technology

MASTER

Concept drift investigation

Zhang, M.

Award date:
2018

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Data Mining Group

Concept drift investigation

Master Thesis

Mingpeiyu Zhang

Supervisors:
prof.dr. M. Pechenizkiy (TU/e)
W. van Ipenburg MSc (Rabobank)

Tutor:
ir. S.B. van der Zon (TU/e)

Final version

Eindhoven, February 2018

Abstract

In this thesis, we developed concept drift investigation tool for high dimensional streams including multivariate numeric data (use case with financial transactions stream analytics) and text data with temporal information (use case with email data). The research problem is formulated to develop a concept drift investigator with specific requirements (i.e., high dimensional, labels are not instantly available), and to develop a visualization tool which can help domain expert to localize the drift and is easy to interact with. We applied different methods to detect drifts in multivariate numeric data and text data.

For numeric data, we proposed a novel concept drift detection method, drift detection ensemble based on alternative label (DDEAL), to meet the requirements. There are three steps in the construction of DDEAL: 1) in the near real-time setting, labels for the data are not instantly available, we perform feature selection on the historical data to select the best feature as alternative label, 2) find a suitable incremental learner as the inner learner of DDEAL, 3) choose detectors in the ensemble and generate weights for each of them. When detecting drifts in the data, DDEAL processes one instance at a time, it first use the inner learner to make prediction for the alternative label on the feature set except for the selected feature, then the ensemble of detectors will make decision on whether a drift or a warning has been triggered by the error rate of the inner learner. The learner will also update itself with the new instance after making the prediction. If a drift has been triggered, the drift will be reported, and all the detectors and the inner learner will be reset. We evaluate the approach mentioned above by comparing it with single detectors, different decision strategies and a competitor approach on multiple real world and synthetic datasets. The results show that it can get at least the same performance with the single detectors which require the class label for drift detections and it is the best ensemble strategy. Besides, it is inline with the competitor approach.

For text data, we applied Dynamic Topic Model to analyze the time evolution of topics in the document collection. We validated its function of showing changes in the topics as well as its stability and reliability on the manually created corpus from the real-world dataset. We also applied DDEAL on the largest component from LSI on text data and showed that it can capture some changes.

For both data, we developed a visual investigation application which not only can automate localization after the detection has happened but also it can create visualizations of machine learning models for domain experts to localize the drift points further and to aid in understanding concept drift. We showed some use cases by using Rabobank transaction dataset and Ling-Spam email dataset on our concept investigation framework.

Preface

This thesis is a result of the master graduation project for Data Science Engineering at Eindhoven University of Technology. The research of this project is done within the Data Mining Group of the TU/e in collaboration with the Financial Economic Crime (FEC) department of Rabobank located in the Netherlands.

It is a great opportunity and valuable experience for me to research the cutting edge techniques for handling concept drifts. From TU/e, I would like to thank my supervisor Prof. Mykola Pechenizkiy who provided me timely support and very valuable feedbacks whenever it is needed. I also would like to thank my tutor Simon and the PhDs, Oren and Tita in the research group for their help during the project. The discussions with Simon inspired me many ideas of how to build up my project. Oren's suggestion initially inspired the novel approach in this thesis. Tita helped me a lot for setting up the NLP part of my project. From Rabobank, I would like to thank my supervisor Werner van Ipenburg for his suggestions and support throughout the whole project. I also would like to thank Jan W. Veldsink and all the data scientists for their help during my internship and my master project at Rabobank. I also appreciate the weekly meeting at Rabobank, each time I can get tons of new ideas.

I would like to thank all of my friends for their help and support during my master study. Finally, I would like to thank my family and my girlfriend for always supporting me throughout my study at TU/e and their unconditional support for all of my decisions.

Mingpeiyu Zhang
Eindhoven, the Netherlands
February 2018

Contents

List of Figures

List of Tables

Concepts and terms

- **Transaction** One transaction is used to describe one money transfer.
- **Fraud transaction** A transaction where a fraudster commits the crime such as stealing money.
- **Phishing email** Emails that are designed to steal money.
- **Concept drift** The statistical properties of the target variable, which the model is trying to predict, change over time in unforeseen ways.
- **Domain expert** A person who is an authority in a particular area or topic.
- **Concept drift localization** Locate the time moment or interval and subgroup of the instances or features where drift occurred.
- **Stop words** The words which have no real meaning, they are usually the most common words in a language.
- **Tokenization** Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation.
- **Ensemble** A type of methods which use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone
- **Error rate** The ratio between the number of a model has wrong predictions and the number of all predictions.
- **False negative** A false negative is when we receive a negative result but we should have received a positive one.
- **False positive** A false positive is where we receive a positive result for a test, when we should have received a negative result. Also called “false alarm”.
- **True positive** A true positive is where we receive a positive result for a test, when we should have received a positive result.

Acronyms

- **DDEAL** Drift Detection Ensemble based on Alternative Label
- **ADWIN** ADaptive sliding WINdow
- **HDDM** Hoeffding Drift Detection Method
- **DDM** Drift Detection Method
- **EDDM** Early Drift Detection Method
- **FHDDM** Fast Hoeffding Drift Detection Method
- **FHDDMS** Stacking Fast Hoeffding Drift Detection Method
- **CUSUM** CUmulative SUM
- **PH** Page Hinkley
- **PCA** Principal Component Analysis
- **LDA** Latent Dirichlet Allocation
- **LSI** Latent Semantic Indexing
- **DTM** Dynamic Topic Model
- **NLTK** Natural Language Toolkit
- **HF** Hoeffding Tree
- **NB** Naïve Bayes
- **KNN** K Nearest Neighbors
- **ALO** At Least One
- **ALHD** At Least Half Detectors
- **AD** All Detectors
- **FN** False Negative
- **FP** False Positive
- **TP** True Positive
- **STS** Standford Twitter Sentiment

Chapter 1

Introduction

With the popularity of information technology, a considerable amount of data is generated every day in all the fields. Companies tend to use machine learning tools to make use of the data to make better decisions in their business. However, the machine learning models are not once and for all since the concept that we try to model would change as time goes on due to the influence of factors such as policy, environment, etc. Besides, the concepts can also evolve, some of these changes can reflect in the data describing them. These changes in hidden contexts, i.e., not directly observed by a model, are called concept drift. Notably, the financial industry is one of the industries that has been experiencing concept drift frequently. Banks and insurance companies are using both machine learning techniques and rule-based systems to detect fraudulent transactions. However, with the introduction of new policies, some field of the data can be changed entirely. If a model then trained on the data with both of the concepts, the model would not get the expected performance. Thus, these learning models need to be able to adapt to changes in data over time.

As one of the major banks in the Netherlands, Rabobank generates millions of transactions every day where potential fraud transactions are hiding in. However, due to the existence of concept drift, the machine learning algorithms used to detect frauds would encounter many bottlenecks in their performance. Apart from the transaction dataset, Rabobank also generates a massive amount of emails from its staffs in the Rabobank email system. To facilitate better use of emails and explore business potentials in emailing, the bank also wants to have a visual tool to help to investigate how the content of the phishing emails changes over time in the dataset.

1.1 Goals and Challenges

The overall goal of this project is to develop and apply suitable methods for concept drift investigation on high dimensional streams, and we can facilitate it for the analysis of the transaction dataset and phishing email dataset provided by Rabobank. These two types of high dimensional datasets are in different format and thus have different challenges.

Transaction dataset (multivariate numeric data)

The transaction data we are using is high dimensional without missing values in it. Table 1.1 shows some examples of the transaction data. For each bank transaction, there are hundreds of features used to describe it. More details of the data are described in Chapter 5.

The data has the following properties which make it different from the drift detection of data from other domains. First of all, the transaction data is pretty high dimensional, and there are hundreds of features in the data to describe one transaction. The high dimensionality makes it difficult to detect the changes in the data since there could be cases when a single feature does

Table 1.1: Examples of the transaction dataset

Timestamp	META_HASH	label	fraud_class	C_000	···	N_000	···	V_000	···
2016-10-05 14:40:42	wE1CtE	1	Mo1	0.1	···	0.142	···	0.1	···
2016-10-11 11:44:26	+ERQsO	1	Mo1	0.102	···	0.186	···	0.105	···
···	···	···	···	···	···	···	···	···	···
2017-09-30 18:01:07	/VLTAH	0	Mo0	0.1	···	0.166	···	0.1	···

not change but the combinations of multiple features may cause a change together. Secondly, the volume of the data is vast as there are millions of transactions being generated every day. These properties may cause three challenges in our project. A difficult challenge in handling concept drift is distinguishing between an actual concept drift and noise, and it is no doubt that there are some noise and outliers in the data. Another challenge is due to the high dimensional and massive volume of the data, and it is challenging to visualize them all at a time. Lastly, the labels of the transactions usually become available a long time after the data is generated. A transaction can only be labeled as fraud when someone reported it to the bank and then confirmed by the staff at the bank. This means that the drift detection can only be conducted on the raw features as there is no direct way to monitor the changes in the joint distribution of predictive features and the target attribute (class label).

To achieve the goal, we first need to find a suitable method to detect the drifts based on the properties stated above. After that, we have to develop an application which can not only visualize the data with the detection results but also provide some means for further investigation in the dataset.

Email dataset (text data with temporal information)

The email data is mainly composed of textual data. Thus it is difficult to find the features to be used for drift detection. Besides, since any two emails almost always contain different contents, it is also hard to define a "drift" in the dataset. If we use the topic model or other techniques to create a higher level representation of the data, it is also a challenge to validate the results.

To overcome the challenges and accomplish the goal, we first need to extract some features from the text for drift detection. Then find an approach to capture the changes of the text over time and design some experiments to validate the output of the approach. Finally, we have to design an application to visualize the dataset and the results.

1.2 Approaches and Results

Summary of approaches

Here we summarize some chosen state-of-art approaches and our novel approach that are used to detect the changes in high dimensional datasets without labels and to create visualization for the detection results for visually localize the drifts in order to address the above challenges of the two datasets (i.e., large volumes of data, high dimensional and unavailability of the labels for the transaction dataset). All the approaches are summarized in Table 1.2.

To handle large volumes of data, we perform pre-processing on the data to remove all the redundant data such that the volume of the data reduced significantly (Section 3.2.1). We use our novel approach "Drift detection ensemble based on alternative label" (DDEAL) to detect drifts in the high dimensional dataset from the online setting where the labels are not instantly available. In DDEAL, we first use feature selection on all the features and select the best one to be the alternative feature. When detecting drifts in the data, it maintains an inner incremental learner,

Table 1.2: Summary of chosen approaches corresponding to each challenge and requirement

Dataset	**Challenges and Requirements**	**Approaches**
Transaction dataset	Large volumes of data	Data pre-processing to remove the redundant data.
	High dimensional In the online setting, labels of the data are not instantly available	1. Incremental learner trained for alternative label. 2. Ensemble of detectors by weighted voting to detect drifts on the error rate of the learner.
	In offline setting, concept drift localization	1. Interactive visualization with highlights on the drift points (results generated by ADWIN). 2. Model visualization.
email dataset	Lack of numeric feature to detect changes	Use Latent Semantic Indexing to extract features out of the text.
	Change detection on the text	Extract topic changes through Dynamic Topic Model.
	Visual investigation	1. Raw features visualization with highlights on the drift points. 2. Interactive visualization for the results of Dynamic Topic Model.

which takes the selected feature as the target attribute and the rest as input features. We then use a weighted majority ensemble of detectors to detect the change in the error rate of the learner (Chapter 4).

To detect concept drifts on the email datasets, firstly we apply Latent Semantic Indexing to group the words that have similar latent meanings together. We can then perform change detection on the largest component in the result. Besides, we use Dynamic Topic Model, a state of art approach to analyzing the time evolution of topics in document collection (Section 3.2.1).

Apart from the detection methods, we also created an interactive application to provide a visual investigation of the drift detection results. There is usually a drift detection delay, i.e., the time between real drift appearance and its detection. Thus the drift time reported by the detector usually is not the actual drift point. To address this problem, we visualize the detection results from ADWIN for each feature for domain experts to dive into details of the data to identify and understand concept drift in the data. Besides, we provide visualizations of machine learning models on both numeric data and text data which can provide further details of the data (Chapter 3).

Summary of results

On a real transaction dataset provided by Rabobank and a real-world electricity consumption dataset, our experiments show that our method can get similar performance as the single detectors which require the class label of the incoming stream. Besides, compared with other ensemble strategies, our weighted majority strategy can get the best performance, and it is in line with state of the art change detection method on high dimensional data. Besides, the experiments also show that the main limitation of the proposed method is when the drift is only on the class label without the change in the features. In this case, we can not find an alternative label which can represent the actual class label well and our method can perform worse than others (Chapter 5).

For the text data, the experiments show that DDEAL together with Dynamic Topic Model can capture the change of topics on the manually created corpus from real-world datasets. Besides,

we also manage to show the results from Dynamic Topic Model is stable and reliable rather than random (Chapter 6).

Finally, the visualization application we built is easy to interact with and can help domain experts to localize the drifts in the data. The visualization of results from Dynamic Topic Model and other machine learning models trained on numeric data can also provide a deep insight into the data. We show the investigation results of the transaction dataset in Section 3.4 and the results of the email dataset in Section 6.5.

1.3 Thesis outline

This thesis is organized as follows:

- In Chapter 2 (Problem Statement), firstly we describe the business problem regarding the two datasets and the application aspect, we then summarize the business problem by addressing the four different aspects (e.g., conditions, objectives, challenges and requirements). Secondly, we introduce the definition of concept drift, explain different types of concept drifts, provide an overview of the existing drift detection methods and formulate our research problem. Finally, we briefly review the existing drift detection methods regarding the challenges and objectives, and we address the need for novel approaches.

- In Chapter 3 (Concept drift investigation framework), we describe the framework of the application in detail. Firstly we provide implementation details of the whole framework. Secondly, we show the details of the drift detection part (data pre-processing, drift detection methods and the generated log files). Thirdly we show details of our data visualization application with the Rabobank transaction data and Ling-spam email data as the use case. Finally, we use this framework to investigate concept drift of the Rabobank transaction data.

- In Chapter 4 (Drift detection ensemble based on alternative label), we first illustrate the intuitions and the main idea of this approach, and then we describe each component of this approach, i.e., the inner classifier and the construction of the ensemble. After that, we discuss the advantages and drawbacks of our novel approach. Finally, we review related work that also use an ensemble of drift detectors to detect concept drift.

- In Chapter 5 (Experiments on the novel approach), we discuss the experiments on the novel ensemble detection method in detail. We first show the setups for the experiments, e.g., dataset description, construction of our ensemble, baseline methods as well as the experimental procedures. After that, we show the experiment results of the performance comparison between our novel ensemble method and individual detectors as well as a competitor detection method. Finally, we give conclusions on all the experiments.

- In Chapter 6 (Experiments on Dynamic Topic Model), we show the experiments of DDEAL and Dynamic Topic Model on text data. We first provide experimental settings including the dataset description, experiment parameters as well as the data sampling strategies for manually creating drifts in the text data. After that, we show the procedure and results of different experiments. Finally, we give conclusions on all the experiments.

- In Chapter 7 (Conclusions), we conclude our contributions in terms of academic and business values. Moreover, we discuss the limitations and future work.

Chapter 2

Problem Statement

In this chapter, firstly we describe the business problem regarding the two datasets and the application aspect, we then summarize the business problem by addressing the four different aspects (e.g., conditions, objectives, challenges and requirements). Secondly, we introduce the definition of concept drift, explain different types of concept drifts, provide an overview of the existing drift detection methods and formulate our research problem. Finally, we briefly review the existing drift detection methods regarding the challenges and objectives.

2.1 Business problem

In this section, we first describe the business problem regarding different datasets and then provide a summary of different aspects of the application. Finally, we address the four different aspects of the business problem: conditions, objectives, challenges, and requirements.

2.1.1 Concept drift on transaction data

A bank transaction usually consists of hundreds of features and the value of each feature may change based on a weekly or daily pattern, i.e., people may spend more money during the weekends and less money during midnights. This may lead to a change of some feature's value on a weekly or daily basis. These values can also drift as a result of the bank policy change which could result in a sudden change, i.e., the value of a field suddenly changes from one value to another at some point and never change back again. Abnormal behaviors such as fraud can also create abnormal values of some fields in the transaction. The fraud detection framework nowadays used in Rabobank takes these transaction data of some particular period as the training set. However, if the training set contains any concept drifts, it will limit the maximum performance of the fraud detection model. For example, if multiple features begin to have values that are not observed in the training set of the model, the model would not be able to make correct predictions for the data. Therefore, it is essential for the machine learning model in the fraud detection framework to be able to adapt to concept drifts in the data. In this context, Rabobank is eager to find a way to detect the drifts in its transaction data to find out whether the data contains concept drift before sending them to the fraud detection model. Besides, it is not enough to only know if there is a drift in the dataset, we also need to localize and confirm the drift which requires visual aids for investigating the dataset.

2.1.2 Concept drift on email data

As one of the most popular communication approaches nowadays, emails are broadly used in our daily lives. For example, co-workers discuss work through emails, friends share social activities and experiences via emails, business companies distribute advertisements by emails. As one of the major banks in the Netherlands, Rabobank has its email system which stores millions of phishing

emails claimed by the customers throughout the years. To facilitate better usages of emails and explore the contents of these emails, it is worth mining and analyzing the information in these emails. An email is composed of date, email address, subject, the body of the email, and so on. It is possible for the body to include pictures, sounds, and programs, but the body is mainly composed of textual data. Thus, it is possible to use text mining techniques to analyze emails and to investigate what are the emails talking about and how the emails change over time. The topics of the emails or any other feature which can describe an email can be regarded as concepts, and the change of them will lead to concept drift. Notably, some abnormal behaviors such as spam emails can also trigger a drift in the topics or other features. The business problem for the email data is to find out how the emails change over time and to investigate the meaning of the changes in the data.

2.1.3 Business problem summary

According to [51], there are three aspects to consider when building a concept drift application: type of the learning task, the environment from which data comes and online operational settings.

Data and task. The task is to do concept drift investigation on high dimensional dataset exemplified by the financial transactions and reported phishing emails datasets. Typically, the transaction dataset is used for fraud detection, and the phishing emails can be used for analytical tasks such as spam email filtering, both in the online setting. Thus, the concept drift detector should be able to conduct in an online setting. Moreover, to visualize the drift detection results which can help to localize the drifts, the application should also provide offline investigation tools.

Characteristics of changes. For both of the two datasets, the change can be sudden, gradual and recurring. For example:

- Sudden change: When the bank introduces a new policy, some fields in the transactions may begin to have values that never appeared before. When a festival is coming, a significant proportion of the emails could suddenly begin to talk about the festival.
- Gradual change: The money amount in the transactions can gradually become higher as living standards improve. The topics in the emails can change gradually from "work" to "daily life" during a day.
- Recurring change: Seasonal shopping behavior such as Christmas shopping can lead to a recurring peak in the money amount in the transactions. The topics of the emails can also change based on seasonal festivals and events.

Operational settings. In the transaction setting, the label of the data arrives at a delay of weeks or even months. The suspicious transactions need to be confirmed by the bank staffs whether it is fraud or not. We can have the labels for historical transaction data, but for the email data, the true labels may are available at all. As we want to detect the drifts in near real-time (as soon as possible), the detector should not be allowed to use the true labels of transaction data.

Besides, domain experts play an essential role in the acceptance of big data solutions. They often want to go away from non-interpretable black-box models and to develop trust in underlying techniques, e.g., to be sure that a control system is really going to react to changes when they happen and to understand how these changes are detected and what adaptation would happen. Therefore it is important that our detection result can be interpretable and help domain experts easily understand how and why a drift happened.

With these settings, we then can conclude the concept drift detection task of this thesis with the following specific objectives and challenges:

- **Conditions**

1. We assume that we have the class labels from historical data, but in the real world setting, we are not allowed to use the labels for change detection.

- **Objectives**

1. Develop a method to detect concept drifts in the transaction dataset and the email dataset.
2. Develop a visualization tool for investigating the datasets to localize and confirm the drifts.
3. Provide the features that are suspected to have drifted by the detection method.

- **Challenges**

1. High dimensional of the transaction dataset.
2. The labels of the data are not instantly available in the real world setting.

- **Requirements**

1. The detection method should be able to detect changes in online settings.
2. The detection result can help domain expert to localize the drift in the data.

2.2 Research problem

In this section, firstly we introduce the definition of concept drift and explain different types of concept drifts. Secondly, we provide an overview of existing methods for drift detection. Finally, we formulate the research problem of this thesis.

2.2.1 Concept drift

In real-world situations, machine learning algorithms often need to act in dynamic environments where the data may change unexpectedly. Each learning algorithm, essentially, is trying to simulate the data generating process to make predictions. If the underlying data generating process or the target that we are predicting has changed, it is likely to lead to the decreasing of the accurate of a predictive model as time passes. We define the data generating process together with the prediction targets to be the *concept*, and we call the change of such concepts as *concept drift*.

Specifically, let $\mathcal{P}$ be the data generating process which provides a sequence of tuples $(\mathbf{X}_t, y_t)$ sampled from an unknown probability distribution $p_t(\mathbf{X}, y)$ at some arbitrary time t. The goal of a learning algorithm is to predict the target variable $y_t \in \Lambda$ given a set of input features $\mathbf{X}_t \in \mathcal{R}^d$ at each time t. Let $p_t(y|\mathbf{X})$ be the posterior distribution and $p_t(\mathbf{X})$ be the evidence distribution. The distributions are deliberately subscripted with time t to explicitly emphasize their time-varying nature. The data may arrive in an online manner, i.e., one single instance $\mathcal{I}_t = \{(\mathbf{X}_t, y_t)\}$ at a time, or in a batch setting which provides a set of tuples $\mathcal{I}_t = \{(\mathbf{X}_t^1, y_t^1), \ldots, (\mathbf{X}_t^N, y_t^N)\}$ at each time.

Based on the cause and effect of the changes, two types of drift are distinguished:

- *Real Drift*: the posterior probability $p(y|\mathbf{X})$ changes over time. Such changes can occur either with or without changes in $p(\mathbf{X})$.

- *Virtual Drift*: the distribution of $p(\mathbf{X})$ changes without affecting the posterior probability of classes $p(y|\mathbf{X})$.

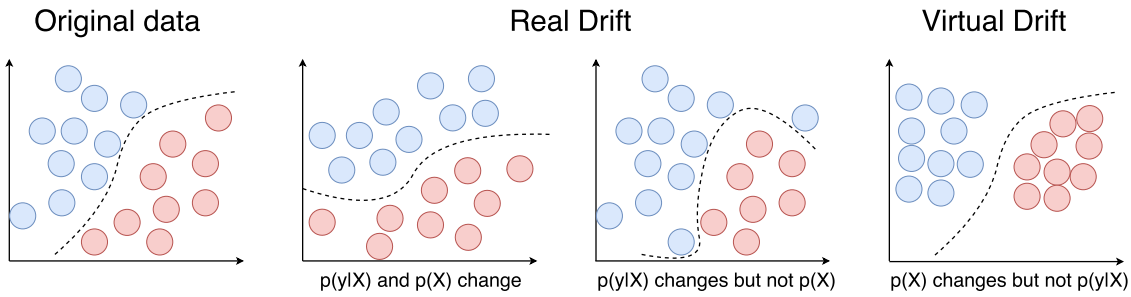

Figure 2.1: Types of drifts.

As illustrated in Figure 2.1, the real concept drifts are the changes of the decision boundaries. Such changes can happen either with or without changes in $p(\mathbf{X})$. Therefore, these changes may or may not be visible from the feature distribution without access to the actual class labels. The real drifts without changes in $p(\mathbf{X})$ cannot be detected solely based on the input feature values. The virtual drifts, on the other hand, do not change the decision boundaries. Such drifts can be detected from the distribution of the input features.

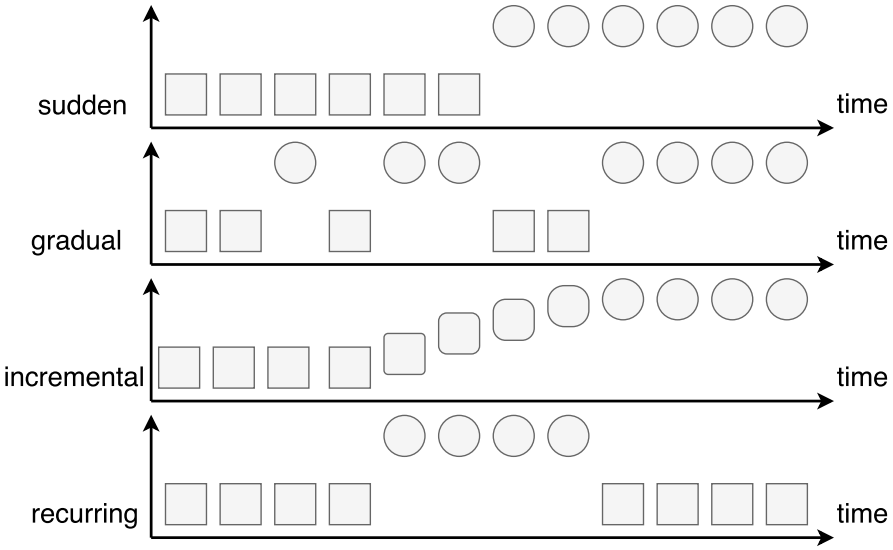

Figure 2.2: Illustration of the four structural types of the drift.

Apart from the cause and effect of the changes, many researchers also divide the drifts based on how the drifts happen. As shown in Figure 2.2, concept drifts can then be further categorized into 4 types: sudden, incremental, gradual and recurring. A sudden drift is used to describe when the changing rate of the feature is extremely high, e.g., when some new policy was announced. Similarly, gradual drift is used to describe the drift with a slow changing rate, e.g., a slowly wearing piece of factory equipment might cause a gradual change in the quality of output parts. In many domains, hidden contexts may be expected to recur. Recurring contexts may be due to cyclic phenomena, such as seasons of the year, this would lead to a reoccurring drift.

Recently, Webb et al. [48] first attempted to provide the more formal framework for comparing different types of drifts and their main properties. The authors also proposed a new, quite comprehensive taxonomy of concept drift types by providing more formal definitions of drift characteristics. Their new taxonomy of drift categories is based on drift subject, drift frequency, drift

transition, drift recurrence and drift magnitude.

2.2.2 Detection methods

Concept drift detection methods are some detectors which can signal the change of data stream distributions based on learning algorithm's performance or the statistics of the input data. The authors of [20] categorized the drift detection methods into four main types:

1. **Methods monitoring distributions of two different time windows.**
   Test whether two fixed-length sequences are from the same distribution according to predetermined confidence. These methods usually use a fixed reference window which represents the summary of the past concept, and a sliding detection window that contains the instances to be tested on. Then some statistical tests will be performed on the two windows with the null hypothesis stating that the distributions are equal. If the null hypothesis is rejected, a change will then be triggered on the test window.

2. **Detectors based on sequential analysis.**
   The sequential probability ratio tests such as the Wald test [47], are the basis for change detection algorithms of this category. These methods detect changes in the following way: let $x_1, \ldots, x_n$ be a sequence of instances. Assume the subset of instances $x_1 \ldots, x_w$ are generated from an unknown distribution P_0 and $x_w \ldots, x_n$ are from P_1. If the probability of observing subsequences under P_1 is significantly higher (above some threshold) than that under P_0, then a drift is triggered at the point w.

3. **Detectors based on Statistical Process Control**
   Statistical Process Control considers learning as a process, and monitors the evolution of this process. For each example from the data stream, the prediction result of the model can be either true or false. For a set of examples, the error is a random variable from Bernoulli trials. The Binomial distribution gives a general form of the probability for the random variable that represents the number of errors in a set of n examples. For each point i in the sequence, the error-rate is the probability p_i of observing false with the standard deviation $\sigma_i = \sqrt{p_i(1-p_i)/i}$. The drift detector manages two registers during the model operation, p_{min} and σ_{min}. At time i after casting the prediction for the current example and verifying the prediction error, if $p_i + \sigma_i$ is lower than $p_{min} + \sigma_{min}$, then $p_{min} = p_i$ and $\sigma_{min} = \sigma_i$.

   For a sufficiently large number of observations, the Binomial distribution is closely approximated by the Normal distribution with the same mean and variance. Considering that the probability distribution should not change unless a concept drift happens, the $1 - \delta/2$ confidence interval for p_i with $n > 30$ examples is approximately $p_i \pm \alpha \times \sigma_i$. The parameter α depends on the desired confidence level. A commonly used confidence level for warning is 95% with the threshold $p_i + \sigma_i \geq p_{min} + 2\sigma_{min}$, and for drift is 99% with the threshold $p_i + \sigma_i \geq p_{min} + 3\sigma_{min}$.

4. **Contextual approaches**
   These methods use context-sensitive learning approaches to detect drifts. The three mechanisms, called staleness, penalization and overall accuracy, which are associated with the individual prototypes that form the classifier, are combined. The first two measures intend to tackle the problem of model complexity, but also gradual drift. If one of the two measured values falls below a very small threshold, called removal threshold, the prototype is removed. The last one is intended for handling gradual and abrupt drift. Staleness tracks the activity of the prototype in making decisions about the new input, and such activity indicates that the recent incoming new input emanates from the input space covered by the prototype. Penalization is about tracking the accuracy of the decisions made and thus observing the evolution of the model in terms of consistency with the recent input patterns. Overall accuracy is to ensure that the accuracy does not deteriorate (at least significantly). The third mechanism is the Statistical Process Control (SPC) criterion which is explained above and

which aims at handling gradual and abrupt changes based on the number of errors produced by the learning model during prediction.

Here we briefly describe a few drift detection methods that we use in our approach and the experiments.

ADaptive sliding WINdow (ADWIN) [6] is the best-known representative of methods from the first type. It takes a (possibly infinite) sequence of real values $x_1, x_2, x_3, \ldots, x_t, \ldots$ and a confidence parameter $\delta \in (0, 1)$, minimum number of items n to start searching changes. ADWIN keeps a variable-length window W of recently seen items, which has the maximal length statistically consistent with the hypothesis "there has been no change in the average value inside the window". Each time the number of items in the window exceeds n, it loop over all the partitions of $W = W_0 W_1$ and it will trigger a change if the average value in one sub-window is significantly different from the other with confidence level δ. The split point of the two windows is the indication of concept drift.

$\text{HDDM}_{\text{A-test}}$ and $\text{HDDM}_{\text{W-test}}$ [17] are also detect algorithms that compare two windows. The former compares the moving averages in different windows to detect drifts. The latter uses the EMWA forgetting scheme [42] to weight the moving averages. After that, weighted moving averages are compared to detect concept drifts. For both cases, the Hoeffding's inequality is used to set an upper bound to the level of difference between averages. The authors noted that the first and the second methods are ideal for detecting abrupt and gradual drifts, respectively.

SeqDrift2 [37] uses a random sampling mechanism to store samples in the detection window. SeqDrift2 stores entries into two repositories called left and right. As entries are processed over time, the left repository contains a combination of older and new entries by applying the reservoir sampling strategy. The right repository collects the new arriving entries. An alarm is triggered when the mean of the two windows are significantly different from each other.

Drift Detection Method (DDM) [18] is the most representative detector from the third type. It assumes that examples arrive one at a time and the the error rate of the learning algorithm p_i and its standard deviation $s_i = sqrt(p_i(1 - p_i)/i)$ should decrease as more training examples are seen. If the classification error has significantly increased, then a drift is triggered. More specifically, if $p_i + s_i \geq p_{min} + 2s_{min}$ then warning level is reached. If $p_i + s_i \geq p_{min} + 3s_{min}$ then drift level is reached. If the model reached warning level at instance x_w , then it assumes a new context start from x_w.

DDM does not perform well with gradual drifts. Instead, Early Drift Detection Method (EDDM) [5] improved the detection of gradual concept drift by considering the distance between two errors: while the learning method is learning, it will improve the predictions and the distance between two errors will increase. However, EDDM is sensitive to errors and noise, and has many false alarms.

Fast Hoeffding Drift Detection Method (FHDDM) [39] detects the drift points by using a sliding window and Hoeffding's inequality. It detects a drift when a significant difference between the maximum probability of correct predictions and the most recent probability of correct predictions is observed. Stacking Fast Hoeffding Drift Detection Method (FHDDMS) [38] improved FHDDM by sliding a long and a short window, that are stacked on each other, to detect concept drifts. The longer window reduces the number of false negatives, while the shorter one detects drifts faster.

CUmulative SUM (CUSUM) [36] is a sequential analysis technique. It triggers an alarm whenever the mean of the input data significantly deviates from zero. Its input is a sequence of real values $x_1, x_2, x_3, \ldots, x_t, \ldots$, and it defines $S_{n+1} = \max\{0, S_n + x_n - \delta\}$ at time $t = n + 1$. A drift is triggered when $S_n > \lambda$. λ and δ should be set wisely as the accuracy of CUSUM is pretty sensitive to these two values. Page-Hinkley is a variant of CUSUM, where the cumulative difference

between observed classifier error and its average are taken into consideration.

DDM, EDDM, and ADWIN have frequently been considered as benchmarks in many works of literature. When a drift has been detected by the detector, the learning algorithm should be updated or retrained by using a new training set which can represent the new concept. However, concept drift detectors may also trigger false alarms which are false positive drift detections. The optimization criteria for change detection is to minimize the detection delay (from the actual change point to detection) and minimize the number of false alarms [51].

2.2.3 Problem formulation

Our research question can be divided into two parts: data change detection and change localization. As discussed above in Section 2.1, in the setting of our problem, the labels of the data are not directly available. This requires us to detect the concept drifts directly through the changes in the input data. Some drift detectors can provide not only drift signals but also warning signals. These warning alarms are usually regarded as the time when a change has suspected to happen in the data. However, due to the existence of noise and the limitation of the detector itself, these detection results sometimes turned out to be false alarms. We want to find a suitable way to visually investigate the detection results in order to visually localize the drifts. In our problem, the localization of the drifts refers to the time moment or interval and subgroup of the instances or features where drift occurred.

Hence, our research problem consists of two parts:

1. Develop a concept drift detector f such that:
    - $f : \mathbf{X} \rightarrow \{0, 1\}$, whenever a instance or a batch of instances arrived, f should be able to give result which indicates whether a drift has triggered,
    - f is capable of detect on high dimensional data.
2. Develop a visualization tool such that:
    - it can help domain expert to localize the drift in the data,
    - it is easy to interact with.

2.3 Related work on the research problem

In this section, we briefly review some state-of-art solutions of our problem with similar challenges (i.e., high dimensional data) and requirements (i.e., detect drift without class labels and can localize the drift through visualization). Besides, we conclude the related works and propose our choice of approaches.

2.3.1 Detect drift on multivariate data

Multivariate distribution monitoring approaches directly monitor the per-feature distribution of the unlabeled data. These approaches are primarily chunk based and store summarized information of the training data chunk (as histograms of binned values), as the reference distribution, to monitor changes in the current data chunk. Hellinger distance and KL-divergence are commonly used to measure differences between the two chunk distributions [12] and to signal drift in the event of a significant change.

In the recent years, principal component analysis (PCA) based methods have been increasingly popular in change detection and anomaly detection. Kuncheva and Faithfull [29] are the first to apply PCA for feature extraction prior to the change detection. They proposed to use eigenvectors

with small eigenvalues for transformation. It was proposed that monitoring the principal components with the lowest 10% of Eigenvalues is sufficient for detecting effective drifts. Principal components with small eigenvalues were selected for reducing data dimensionally because they are analyzed to be more likely influenced by a change. However, Qahtan et al. [41] proved theoretically and empirically that eigenvectors corresponding to large eigenvalues instead are more relevant, as data characteristics in the original feature space are best summarized by the top component vectors, which retain the maximum variance after the reduction. Recently the authors of [33] proposed a linear-time algorithm for robustly detecting non-linear changes in massively high dimensional time series. In this algorithm, they first use scalable PCA to project the high dimensional input data into lower dimension space, and then they perform scalable factorization of the joint distribution. Finally, they compute divergences between these lower dimensional distributions to see if they are different from each other.

However, the created dimensions from PCA usually do not have any meaning in the real world. Thus, by comparing distribution on the lower dimension space, it can not capture the correlations between the original dimensions.

2.3.2 Drift detection on text data

Most of the previous works that detect changes based on capturing the changes of topics of the text. The Latent Dirichlet Allocation topic model of Blei et al. [10] is well-established as an effective approach to extracting topics from a set of documents. Based on LDA, Knights et al. [27] employed a compound topic model (CTM) to track topics across two distinct data sets (i.e., past and present) and to visualize trends in topics over time. Blei et al. [9] developed Dynamic Topic Model to capture the changes in the topics over time. Besides, Ahmed and Xing [2] also presented a topic model, iDTM, that can adapt the number of topics, the word distributions of topics, and the topics trend over time. The model can also capture the birth and death of topics.

Some researchers also proposed to trace the change of text with Latent Semantic Indexing [43], which can group words with similar latent meaning together. Other changes such as sentiment [8], number of words and so forth can be detected directly using the change detection algorithms we showed in Section 2.2.2. As for the visualization of the changes in text data, Havre, Susan, et al. [24] developed "Themeriver", which uses the metaphor of a thematic river to depict temporal changes of word frequencies.

2.3.3 Detect drift from unlabeled data stream

PCA and LDA based methods described above all do not use labels. Besides, other previous works that focus on change detection without class labels are trying to find a way to compare the distribution of the data from different time windows. A representative example of this type of methods is [26]. They exploit order statistics of the data and define generalizations of the Wilcoxon and KolmogorovSmirnoff test in order to define the distance between two distributions. Another representative algorithm is VFDTc [19] which also performs continuous monitoring of the differences in the class-distributions of the examples from two time-windows. The change detection method in VFDTc evaluates past splitting decisions in an incrementally built classification tree and detects splits that have become invalid due to concept drift. Some recent works such as [15] also focused on statistical tests of two time windows. Žliobaitė showed that some types of concept drifts become undetectable when no true labels are provided [50]. She proposed a method to identify detectable drifts without label information.

These methods usually need a pre-defined window's size to test the changes. When testing on multivariate time series data, it is difficult to define a window size which is suitable for every dimension. Besides, without access to class labels, some real drift could not be detected if they are not related a change in $p(\mathbf{X})$ [45].

2.3.4 Visualization of concept drift

Pratt et al. described a visualization technique that uses brushed, parallel histograms to aid in understanding concept drift in multidimensional problem spaces [40]. They used parallel histogram graph to show the distribution of the features and combination of the features. However, the parallel histogram chart is not easy to interact when using high dimensional data. Finding patterns without pre-providing highlights from the graph is difficult.

2.3.5 Conclusion for related work

To conclude, the related works that are focusing on drift detection on multivariate data are not suitable for our project. Because they can only provide the change information over the distribution of the whole feature space, it is hard to localize the drift in the dataset, i.e., on each feature. Moreover, these methods can only detect the virtual drifts, and they cannot detect any real drifts. In our case, we can use a drift detector such as ADWIN for individual features to detect the virtual drifts, and we still need to develop a new method for the detection of real drifts. For text data, we can first apply Latent Semantic Indexing to group the words with similar latent meanings, then apply change detection on the largest component from the result. Moreover, we can use the LDA based methods, i.e., Dynamic Topic Model, to monitor the change of the topics. It is the best-suited approach in our case because we want to investigate the contents of the emails, the generated topics of Dynamic Topic Model can provide many insights on the content. Not too many works focused on visualization of the concept drifts, we also have to develop a new framework for the drift visualization and localization.

Chapter 3

Concept drift investigation framework

Our concept drift investigation framework consists of two major components, as illustrated in Figure 3.1, the drift detection part, and the data visualization application part. The input data is either multivariate numeric data or text data with temporal information, in batch or stream format. First of all, the detection part processes the input data and perform drift detection on the data. After that, some processed file will be generated including the processed data file and the log files which are used to describe the detected drifts. Finally, the visualization application will load the generated files and create corresponding interactive views for the user. The visualizations can help to localize of the drifts in terms of when (time moment or interval) and where (a subgroup of features) the drift occurred (as defined in Section 2.2.3).

Figure 3.1: Framework overview.

In this chapter, we describe this framework in detail. Firstly we provide implementation details of the whole framework. Secondly, we show the details of the drift detection part (data pre-processing, drift detection methods and the generated log files). Thirdly we show details of our data visualization application with the Rabobank transaction data and Ling-spam email data as the use case. Finally, we use this framework to investigate concept drift of the Rabobank transaction data.

3.1 Implementation details

The data flow of our drift investigation framework is shown in Figure 3.2. The left part in the black dashed frame is the detection part and the right in the red dashed frame is the interactive application part. As we can see in the detection part, it starts from the raw dataset from Rabobank (or any other source), and then we pre-process the raw dataset in order to remove the noise and outliers in the dataset. The drift detection algorithm will then iterate over all the data and generate a log file which describes all the drifts the algorithm detected. The processed dataset together with the detection logs then can be used by the application to generate interactive visualization views of the detection results. In the application part, we built it as a web application where the backend is based on Python Flask and the frontend is based on d3.js. The user can interact with the frontend visualization which would send a request to Python Flask to prepare the data to be

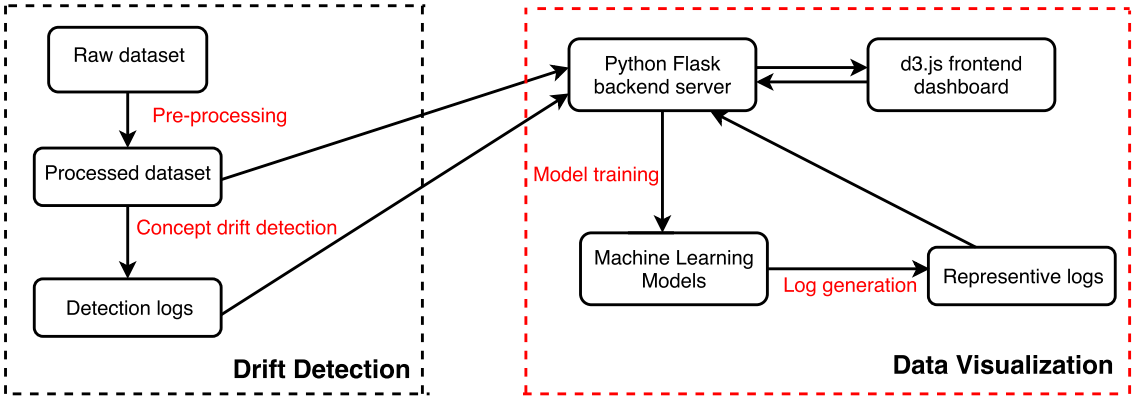

Figure 3.2: Data flow of the framework.

used to visualize. The data used to visualize is directly from the processed dataset. The user can also choose to visualize the data through a model which may provide further details into the data. The models are represented in log files.The backend server can then read the log files and then send the corresponding contents to the frontend to create visualization views. Some models and log files are pre-trained such as Dynamic Topic Model because they are usually extremely time-consuming.

This framework can be used on any multivariate time series dataset and text dataset. Moreover, the drift detection part can be combined with any detection algorithm which meets the following properties: 1) it detects concept drift from raw features, 2) it can take one instance at a time and then make a decision whether a drift has been detected. Besides, as the detection algorithm processes the data one instance at a time, the framework can either be used in an online context where the domain experts can monitor the detection results in real time or be used for batches data.

3.2 Drift detection part

3.2.1 Data pre-processing

In this section, we show the pre-processing techniques that we perform on the datasets. These techniques are used to reduce the noises in datasets. We describe more details of the datasets in Chapter 5 and 6.

Transaction dataset

In Rabobank context, the newly generated transactions from the transaction stream are stored in the buffer database for a predefined period of time. If the buffered transactions are identified as abnormal within this period, they will be labeled as abnormal and immediately moved to the confirmed database. If the buffered transactions have not been identified as abnormal in the predefined period, they will be labeled as normal and moved to the confirmed database at the end of the time period. If a normal transaction in the confirmed database gets identified as abnormal subsequently, its label in the confirmed database will be changed to abnormal immediately. In this setting, the transactions can be disordered in time from the raw dataset we get from Rabobank. The input of detection algorithms should be ordered by time, so we first sort all the transactions in the dataset by the timestamp field.

There are millions of transactions being generated every day in Rabobank. Hence it is no doubt that there are some noise and outliers in the data. The dataset we use contains 1,796,979 trans-

action records from October 2016 to September 2017. As all the features are discretized into a categorical value between 0 and 1 (more details in Section 5.1), in order to remove the noises, we can choose to filter out the values of each feature which occurred less than a threshold. The threshold should be neither too big that we may filter out too much information, nor too small that we can not filter out anything.

Table 3.1: Filtering result for different thresholds.

Remove values appeared	less than 10 times	less than 0.1%	less than 0.01%
Transactions deleted	0.15%	30.05%	4.28%

We tried several options for this threshold and the results are shown in Table 3.1. In the first experiment, we remove the transaction which has any feature value that appeared less than 10 times throughout the dataset. And in the second and the third experiments, we remove the transaction which has any feature value that appeared less than 0.1% and 0.01% respectively. We can see from Table 3.1 that the first and the third actually removed the reasonable number of transactions. In order to investigate how good are the first and the third one, we then use ADWIN to detect the drift on each pre-processed feature quickly and visualized the features that are detected to have drifted. In the visualization of the data from experiment 1, we can visually see drifts from the features N_142, N_135, C_196, N_152, C_046, N_114, C_337, C_054, N_000, C_269, C_082, N_147, and N_062. Moreover, in the visualization of the data from experiment 1, we can visually see drifts from the features N_135, C_196, N_152, C_046, N_114, C_337, C_054, C_269, C_082, N_147. The visualizations are in Appendix A.1. Furthermore, we can see that the latter is a subset of the former, so we can conclude that the third one may have removed some details of the dataset. We use the threshold of the first one in all our other experiments.

Besides, since there are too many records in the dataset, it is impossible to visualize them all at a time. Moreover, we get the information from Rabobank that the values within an hour may not change, so we then reduce the dataset by extracting a median value from every 5-minute time window. Because all the features are discretized to categorical values, it is more meaningful to take the median than average. With this operation, we can significantly reduce the size of the dataset which can then be used for visualization.

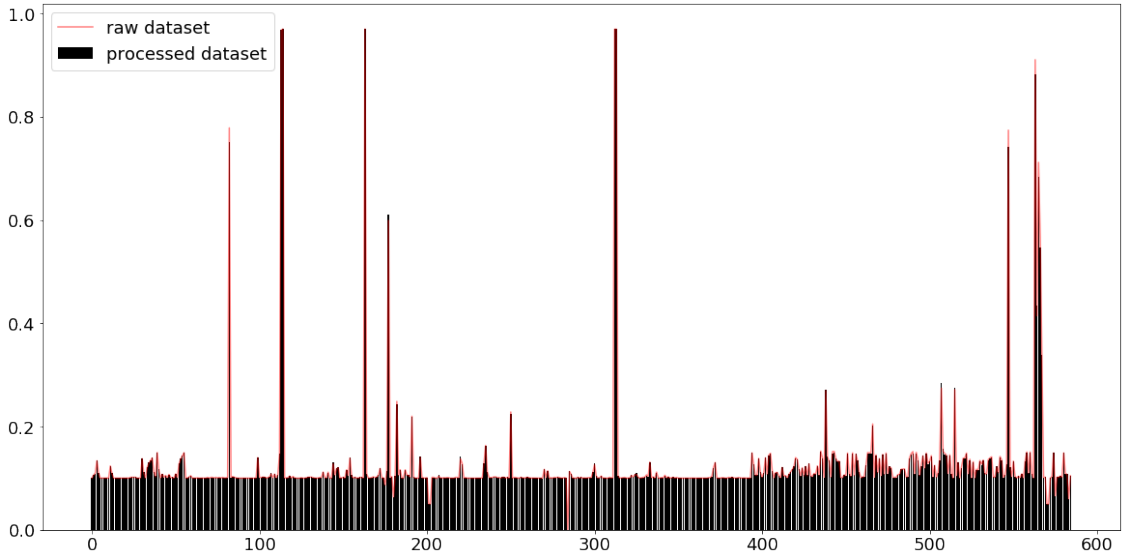

Figure 3.3: Mean values of the features before and after pre-processing.

After the pre-processing step, there are 16,789 rows remain in the dataset. The bar chart of the

average values of each feature before and after pre-processing are shown in Figure 3.3. We can see that for most of the features, the average values before and after the pre-processing are about the same.

Email dataset

The email datasets that we are using in this thesis is the email data from Rabobank and Ling-Spam[1], a publicly available collection of emails. More details of the datasets are provided in Section 6.1. Both of them only contain text information: the subject of the email and the content of the email. The text data can not be used directly by the detection algorithm, so we generate some numeric features by using Latent Semantic Indexing (LSI). What LSI can do is to group the words that have similar latent semantics together. e.g., if we have a document which has a vocabulary of three words: {(car), (truck), (flower)}, the result from LSI could be {(1.3452 * car + 0.2828 * truck), (flower)}. The words "car" and "truck" are grouped, and the value before each of them are used to reflect their relative importance. By applying LSI on each timestamp of the emails, we can get multivariate time series data out of the emails. Then, we can perform change detection on the largest or the smallest component and see how they change over time.

In order to present some meaningful investigation, we remove the useless information such as stop words and emojis from the text. Therefore, after getting the numeric features, we then do the general pre-processing on the content of the emails:

1. Remove URLs and emojis.
2. Tokenize the text.
3. Transform the words into their roots.
4. Remove stop words.
5. Remove words that appear less frequently.

The primary usage of the email content is to extract the topics and latent semantic components out of them and then provide domain experts with how the topics change over time. The first steps in any natural language processing workflow are to remove stop words and lemmatization. Removing stop words involves filtering very common words such as the, this and so on. Lemmatization involves replacing different forms of the same word with a canonical form: both colors and color would be mapped to color, and organize, organizing and organizes would be mapped to organize. Removing stop words and lemmatization is very challenging, and beyond the scope of this project. We use existing stop word list and existing tools to perform the stop words removal and lemmatization.

The tool we use to help us do the pre-processing is NLTK, which is a Python package for natural language processing. The open dataset lingspam only contains emails in English, we use the Onix stopword list because it is probably the most widely used stopword list. The Onix stopword list contains 429 words. The email dataset from Rabobank primarily contains emails written in Dutch, so we use the Dutch stopword from NLTK's corpora which contains 101 words. Apart from the stop words, we also filter out the URLs and emojis in the text as these contents usually do not have any meaning in terms of the topics of the emails. Sometimes the content of the email itself is in an HTML text format, to remove all the HTML tags and get the real content of the HTML, we use "email" package in Python.

[1] https://storage.googleapis.com/trl_data/maildata/lingspam.zip

3.2.2 Concept drift detection

There is usually a long time delay between the time we have the transaction data and when we have the class label for it, and the labels for the email dataset are not available at all. Hence, the concept drift detection algorithm we use here should be able to detect changes in the raw features. In this section, we briefly describe our drift detection methods for multivariate data and for text data.

Detection on multivariate data

Here we provide two directions for detecting the drift. The first is to detect on the whole feature space and give a time when the whole feature space has drifted. In the real world setting, there could be some scenarios where some features shift as a whole rather than as individuals. i.e., assume we have two features feature A and feature B in the dataset, and when we look at them individually, they did not shift. However, when we combine these two features by for example adding them together, the combination product could have drifted. Methods reviewed in Section 2.3.1 can be used here. However these methods usually need to define the size of test window which is hard, and it is difficult to localize the drifts with these methods. Thus, in order to capture such behaviors and to try to capture some of the undetected real drifts, we propose a novel method called drift detection ensemble based on alternative label (DDEAL) which is described in Chapter 4.

The second direction is to detect virtual drifts on each individual feature. There are hundreds of features in the dataset, and it is not possible to visualize them all at a time. In this case, we want to detect changes on each feature and highlight the detected drifts in the visualization in order to further localize the drifts. In this case, we use ADWIN [6] to detect for each feature of the data.

Detection on text data

Most of the existing change detection methods are focusing on numeric features, thus we need to extract some such features from the emails. As shown in Section 3.2.1, we generate 4 numeric features from the raw email data, the number of words, stop words, punctuations, and sentences in the email content. These features can be used directly by our weighted drift detection ensemble to detect them individually. Besides, there is no label in the email dataset, so we can not use DDEAL on this set of features. Instead, we use Dynamic Topic Model [9] in order to capture the changes of the combination of words in text.

Dynamic Topic Model is a probabilistic time series model which is used to analyze the time evolution of topics in a large document collection. The approach is to use state space models on the natural parameters of the multinomial distributions that represent the topics. The model takes a sequence of collections of documents as input, where each collection of documents is regarded to come from the same timestamp. The number of topics to generate is also an input parameter of the model. As a result, a topic generated from the model is a sequence of distributions over words. We can then find an underlying theme of the collection and track how it has changed over time.

3.3 Visualization application with use case

In this section, we show details of our data visualization application with the Rabobank transaction data and Ling-spam email data as the use case. From the application point of view, the application can be divided into frontend and backend, and from the functionality point of view, it can be separated into data visualization and model visualization. In this section, we first give an overview of the structure of the whole application. After that, we show details of the data visualization page and the model visualization page.

3.3.1 Application overview

The application consists of two parts, the Python Flask backend server and the d3.js frontend dashboard as shown in Figure 3.2. We use d3, which is a JavaScript library for visualizing data using web standards, to create visualization charts of the datasets. d3 itself can be used to create visualization applications, and we extend it by using a backend server based on Python Flask in order to make it more easier to train a model or to process data because there are tons of available packages in Python to do such things.

The application has two pages, the main dashboard page, and the model visualization page. The main dashboard is used to visualize the drift detection results from the detectors, and it provides an interactive way to dive into the data and to compare different features or different time. The model page provides visualization of machine learning models which are trained on the dataset which can be used to further investigate the datasets. For the datasets with multivariate numeric features, it provides models such as decision tree, Naïve Bayes and some clustering models. As for the text datasets, we visualize the results from Dynamic Topic Model to show the changes of topics over time. We will show more details in the following subsections.

3.3.2 Dashboard page

The dashboard page is the main interactive part of our application, as shown in Figure 3.4, it mainly consists of five parts: filter, drift detection result from DDEAL, drift visualization, feature comparison and feature details. We will first show the data that we use for the visualization, then show the page in detail from top to bottom.

The log file for the main dashboard is a JSON file, which contains all the drifted features and the corresponding drift time. Listing A.1 in the appendix shows an example of such log file. This log file is generated from the detection results of multivariate numeric data by using ADWIN on every feature, and it helps to highlight the drifted features and the location of the drifts. Apart from the log file, the page loads the data of each feature directly from the processed data file after the pre-processing procedure in Section 3.2.1.

When looking at the structure of the dashboard page, to start with, the filter is used to choose what the user wants to show in the drift visualization part. It has three option bars. The first is used to choose the feature to be visualized, the features to choose from are the features which have been detected to have drifted by our drift detector. The rest two option bars are used to filter the time range. They are lists of drift time of the selected feature. i.e. in our drift detector the feature x_i has shifted at time $[t_1, t_2, \ldots, t_n]$, then if the user wants to visualize x_i, he can choose start and end time from $[t_1, t_2, \ldots, t_n]$ for the time range to be visualized in the drift visualization part. It also provides the option to visualize from the start of the dataset and to visualize till the end of the dataset. After setting all the filter options, the user can then click on the "Show feature" button to show the filtered visualization. All the features are sorted by the number of times it has detected to drift. Besides, one can also click on the "Next view" button to quickly show the next feature in the list from start till end.

The detection result from DDEAL part shows a table generated from the result of our novel method. More details of DDEAL will be described in Chapter 4. In the table, each row describes a detected drift where the first column is the time of the drift, and the second column contains the potentials that are corresponding to this drift. These features are generated from the results of ADWIN on each separate feature. The results of ADWIN are presented in the feature details part which will be explained later. Whenever a feature has detected by ADWIN (discussed in Section 2.2.2) to have drifted within 5 days of the drift time in DDEAL, the feature name would be added to the corresponding column. e.g., the drift time of the first row is "2016-12-23 21:33:53", when looking at the feature details table, the feature "N_072" has drifted at "2016-12-28 19:25:19", we

think this feature could potentially cause the drift in the first row, so we added it to the first row.

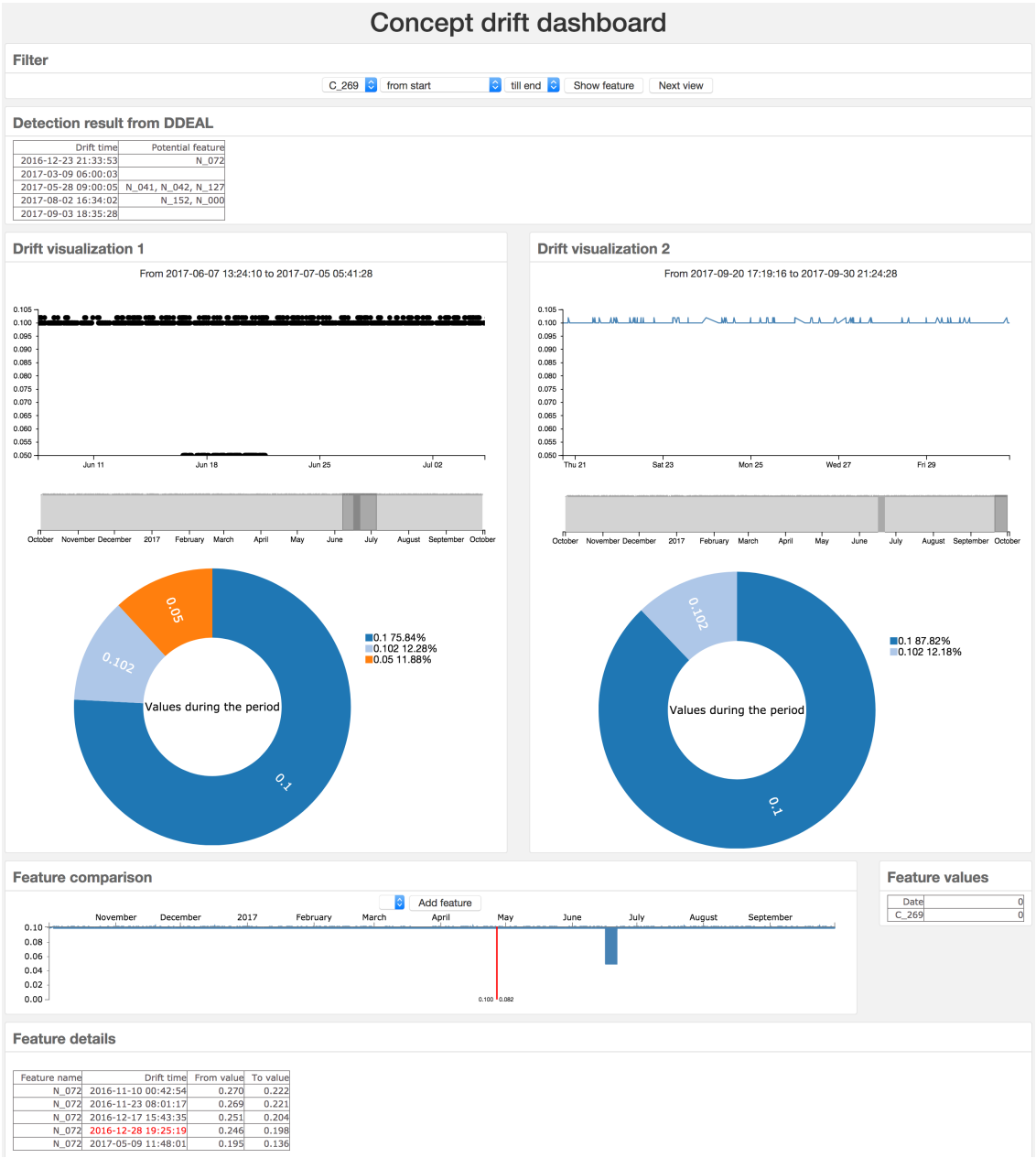

Figure 3.4: An overview of the dashboard page.

The drift visualization part contains two main parts, each of which visualizes the same data filtered by the top options. The difference is that the left one shows a scatter chart of the data and the right one shows a line chart. Here we want to provide the user with different views of the data, and they may provide more details than a single one. We call these two charts the main charts, as they provide the primary visualization of the data. These two charts visualize the detection result from ADWIN. We use red lines in the main charts to indicate the detected drift in the feature, and we add text above the main charts to show the time range of the current selection. Each of the pie charts under the main charts shows the partition of the feature values of the current

visualization.

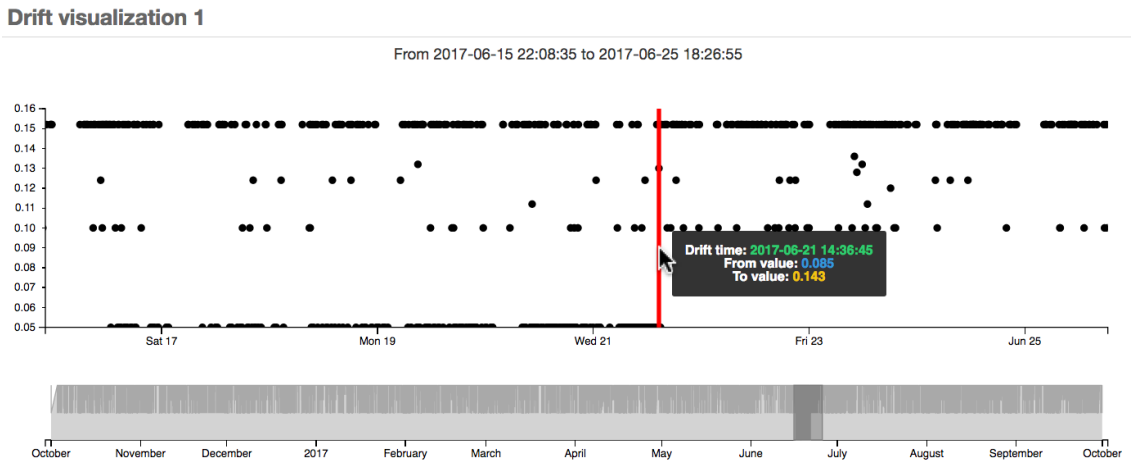

Figure 3.5: An example of the tooltip box.

As for the interaction of the drift visualization part, firstly, the main charts are zoomable, the user can zoom the charts either by moving the cursor into the chart and scrolling the mouse wheel or by brush and select a time window in the grey navigation charts under the main charts. The user can then compare between different time window by zooming in a different level and selecting a location in the main charts. Secondly, if the user moves the mouse over the red lines, a tooltip box will show up to provide details of the corresponding drift. As shown in Figure 3.5, the tooltip shows the drift happening time, the mean value of the feature before and after the drift. Finally, when the time window in the main chart changes, the pie chart under it as well as the time range text will update according to the selected time window.

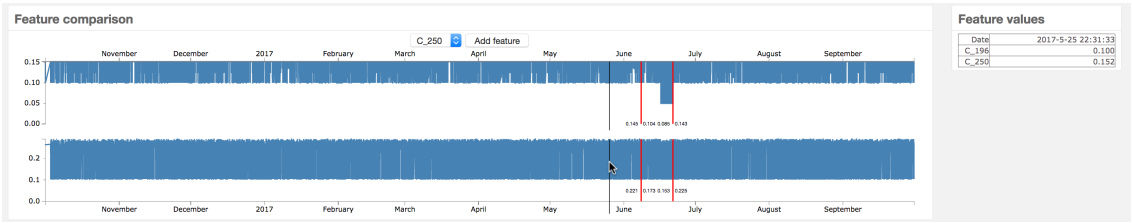

Figure 3.6: Feature comparison part of the dashboard page.

Whenever a filter has applied on the drift visualization part, the feature comparison part will also load the same filtered data. Besides, the user can choose to add another feature beneath the chart of this part to compare between different features. There are a selection bar and a button in this part for this function. The selection bar contains our suggestion features, each of which has at least one same drift time as the main feature of the filter part. We use a line chart to visualize the features. Besides, red lines are also used in the feature comparison charts to indicate the detected drifts. When the user is moving the cursor in the charts, a black line will show up to indicate the cursor location, and the right table will show the values of the features at the selected location. The values in the table will change as the cursor moves in the chart.

At the bottom of the dashboard page, we provide a feature details part which contains a list of tables to show the drift details of all the drifted features detected by ADWIN. For each feature which has been detected to have drifted, we create a table where each row contains a drift. More

specifically, each row shows the drift time and mean values before and after the drift, as shown in the bottom of Figure 3.4.

3.3.3 Model visualization page

We create two views of the model visualization page for multivariate numeric data and text data as the different type of models can be trained on the different type of data. We will show these two views individually in this subsection. We do not need a log file between the backend and the model visualization of multivariate data because these data can be sent in real time after a model has been quickly trained. A Dynamic Topic Model (DTM) for the text datasets usually takes hours to train, so we train some such model in advance and store the results in log files for the visualization.

Model visualization for multivariate data

Figure 3.7: An overview of the model visualization page of multivariate data.

Figure 3.7 shows an overview of the model visualization page of multivariate data which contains two primary parts: the configuration part and the model visualization part. To begin with, the configuration part is a panel where the user can choose the data they want to train a model on. The user can first choose the type of the model, then choose which features to be the features in the training set as well as a feature to be the class label. After that, the user can choose a start time and an end time as the filter to limit the time range on the training set. Finally, we can choose to visualize the result in the panel above (Model visualization 1) or at the bottom (Model visualization 2).

This page provides two types of models to visualize, the k-means clustering and the decision boundaries of decision trees. For the k-means clustering option, the user can choose the parameter k to specify the number of clusters to be used. The user can also choose multiple features from the dataset to set as features and then select the time range to filter the data, and finally choose to visualize it at the top or at the bottom. Figure 3.8 shows an example visualization of the clusters with two features selected and $k = 3$. We use scatter plot matrix to visualize all the combination of the features and give different colors to different clusters. As we can see in the figure, the small dots represent the points from the dataset and the stars represent the centroid points. The user can select different features, different parameter settings and time window for the two visualizations and compare them to get a deeper insight into the data and the concept drifts.

For the decision boundary option, the user can only choose two features to train on. However, all the features together with the original class label can be set as the label. After selecting the time window, the user can choose to visualize it at the top or at the bottom. Figure 3.9 shows an example of the decision boundary result. In this case, there are two classes in the label which are colored by blue and red. The background color indicates which class the region belongs to. Besides, it provides four types of tree models, the Decision Tree, Random Forest, Extra Trees and AdaBoost

which are all visualized in the figure and they can potentially provide different insights of the data.

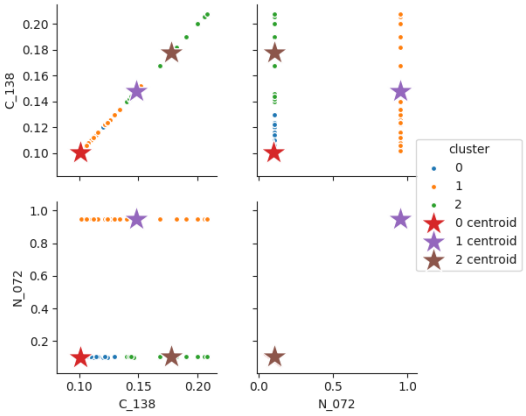

Figure 3.8: An example of the clustering visualization result. (Different color represent different clusters)

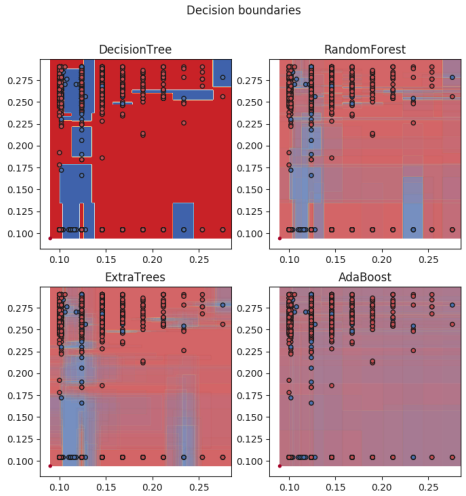

Figure 3.9: An example of the decision boundary result. (Different background colors represent different decision regions of the model)

Model visualization for text data

Assume we have a dataset which contains m emails, and the emails can be divided into k collection of emails $E_1, \ldots, E_k$, i.e. by day or by week, and each collection of emails represents a timestamp. With these data and parameters we can then train a DTM on these emails to get n topics. Each topic t_i is a sequence of distributions over words, i.e. $t_{ij} = \{p_{w1}, p_{w2}, \ldots\}$ is the word distribution of i^{th} topic at timestamp j, where $1 \leq i \leq n$, $1 \leq j \leq k$, p_{w1} is the probability of the topic to generate word $w1$ at this time. By using this result, we can rank the words in the topics at each timestamp by the corresponding probabilities. The higher the ranking is, the more important the word is for the topic. The top 10 words of a topic t_i can be represented as $top_{t_i} = [\{w_{1,1}, w_{2,1}, \ldots, w_{10,1}\} \ldots \{w_{1,k}, w_{2,k}, \ldots, w_{10,k}\}]$, which is a sequence of 10 words. For each topic from the result, we create a file called top word file to store its top 10 words. Each line in the top word file is a set of 10 words from top_{t_i}.

For each email e, DTM can also generate $p(t_i|e)$, which is the probability of email e to belong to topic t_i. The higher the probability is, the more related the email is to the topic. With these probabilities, for each topic t_i at time j, we can sort the values of $p(t_i|e_l)$ such that $e_l \in E_j$, and then get the emails that have highest probabilities. By using this strategy, for each topic t_i we can get a sequence of most related emails $e_1, e_2, \ldots, e_k$. For each topic from the result, we create a file called subject file to store the subject of such emails. Similar to the top word file, each line of the example subject file comes from a timestamp that is the same as the line number.

Assume a word w appears in topic t_i during all the timestamps, then we can have a sequence of probabilities $[p_{w_1}, \ldots, p_{w_j}]$ each of which represents the probability of t_i to generate w at some timestamp. We can then use the sequences of such words to create clusters. Each cluster contains words that change in a similar way in the topic. For each topic, we create a file called cluster file to save the clusters of it.

Apart from the top word file and the subject file, we also use pyLDAvis[2] to generate LDAvis[3] log files for each timestamp. These files can then be directly used by the frontend d3 to create a visualization for DTM at each timestamp. More details of LDAvis can be found in the paper [44].

Figure 3.10 shows a use case of the Dynamic Topic Model visualization on the Ling-spam email dataset. It contains four parts: configuration, top words, word clusters and LDA model visualization.

The first three parts are connected to each other, the contents of them are controlled by the selection bar in configuration. The idea of these three parts comes from Figure 4 in [9]. User can choose different topics in the model to visualize in these three parts.

When a topic is chosen, the top words part will show the top 10 words of that selected topic over time. Each green box in it contains top 10 words of that topic at some particular timestamp generated from the model. With this view, we can see how the words in a particular topic change over time. Furthermore, we select three most frequent words in the topic as the description of the topic, and these words are highlighted with different colors to make it easier for the user to track the change behaviors of them.

Besides, in the configuration part, we also show the example subjects from the selected topic. The subjects are from different timestamps and each of them is from an email which is the most related to that topic at the corresponding timestamp.

In the word clusters part we also visualize the cluster of words within a topic that have the similar changing behaviors. The y axis of the chart is the average probability of the words in the cluster. The construction procedure of the clusters are described above.

The LDAvis at the bottom of the page provides a global view of the topics (and how they differ from each other), while at the same time allowing for in-depth inspection of the terms most highly associated with each topic. The user can interact with it by clicking on different topics and it allows users to explore topic-term relationships flexibly. We also add a selection bar at the top of it for the user to select different timestamp to visualize.

[2]Python library for interactive topic model visualization, available from https://pypi.python.org/pypi/pyLDAvis

[3]A visualization tool built with d3.js.

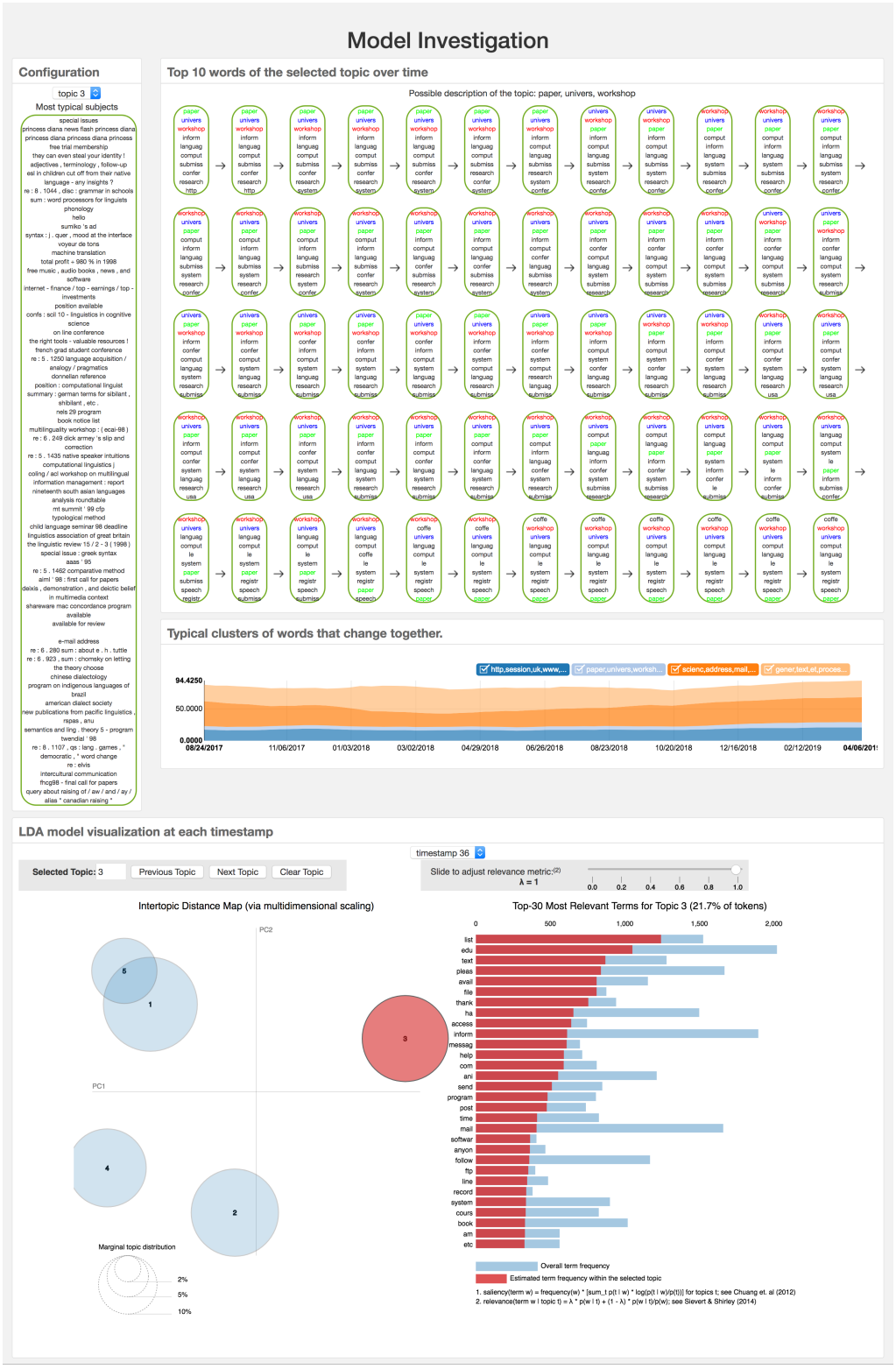

Figure 3.10: An overview of the model visualization page of text data.

3.4 Exploration on Rabobank transaction dataset

In previous sections, we already showed how we could use this framework to investigate the drifts in multivariate numeric data and email data and we used part of the results of the Rabobank transaction data as the use case to explain the function of each component in the application. In this section, we show the summary of the detection results and the investigation results on Rabobank transaction dataset.

Table 3.2: Detection result from DDEAL

Drift time	Potential feature
2016-12-23 21:33:53	N_072
2017-03-09 06:00:03	
2017-05-28 09:00:05	N_041, N_042, N_127
2017-08-02 16:34:02	N_152, N_000
2017-09-03 18:35:28	

Table 3.2 shows the detection result from DDEAL (our novel method). We can see that it detected 5 drifts in the transaction dataset. The first, third and the fourth are potentially related to some features. We then visually investigate the features in the list and see whether we can visually localize the corresponding drift. For the features N_072, N_041, N_042, and N_127 we did not manage to localize the drift visually. However for N_152 and N_000, as shown in Figure 3.11 and 3.12, they both begin to have some high values since "2017-07-31". These two features could be the reason for the fourth drift detected by DDEAL. Here we manage to localize one drift from the results of DDEAL.

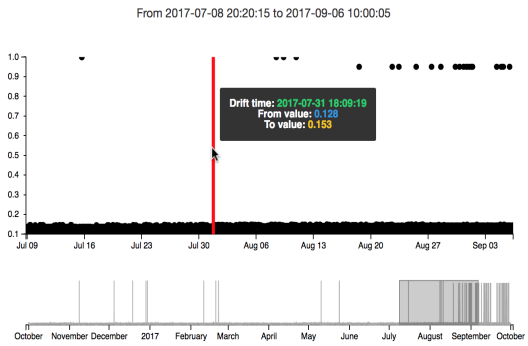

Figure 3.11: Visual investigation of N_152.

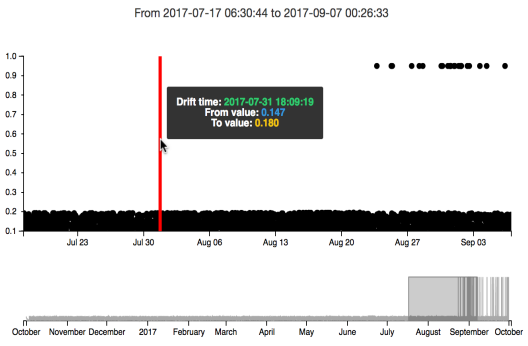

Figure 3.12: Visual investigation of N_000.

Besides, as already showed in the pre-processing step, we found the features N_142, N_135, C_196, N_152, C_046, N_114, C_337, C_054, N_000, C_269, C_082, N_147, and N_062 have some visual drifts as shown in Appendix A.1.1. We can precisely localize the drifts in these features, and they can be confirmed to have drifted by using our investigation framework.

Apart from the raw feature exploration, we also explore the features by the model visualization page. For the two features N_152 and N_000, we visually investigate two time windows: from "30-06-2017" to "30-07-2017" and from "30-07-2017" to "30-08-2017". We use the fraud label as prediction label for the decision boundary visualization. Moreover, we choose to build 3 clusters by using k-means clustering.

Figure 3.13 and 3.15 show the result of the first time window and the results of second time window are shown in Figure 3.14 and 3.16. As we can see in the boundary results, in the first

time window, there is a blue horizontal rectangle at the right bottom of the decision tree decision boundary, whereas in the second time window this rectangle becomes a vertical rectangle boundary. In the results from clustering, from the first window we can see that the cluster with a brown star as centroid is between 0.16 and 0.18 of feature N_000, whereas, in the second time window, this cluster shifted to centroid between 0.8 and 1.0 of feature N_000. With these results, we can further understand how does the drift happen.

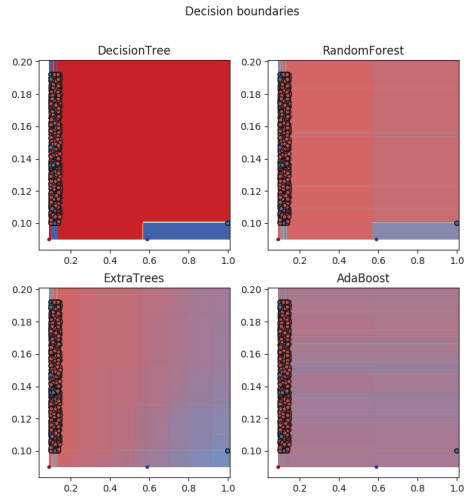

Figure 3.13: Decision boundary visualizaion of the first time window.

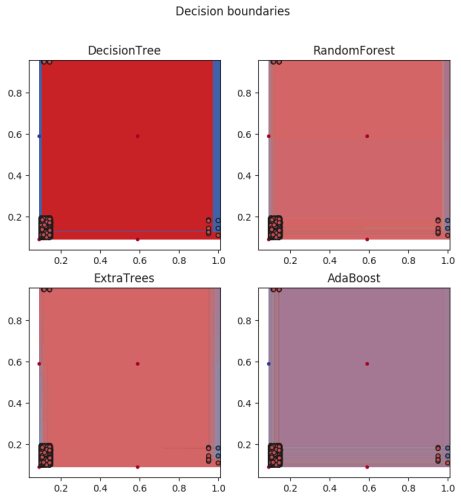

Figure 3.14: Decision boundary visualizaion of the second time window.

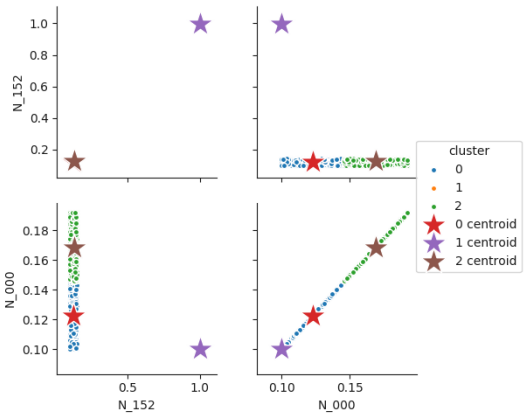

Figure 3.15: Clustering visualizaion of the first time window.

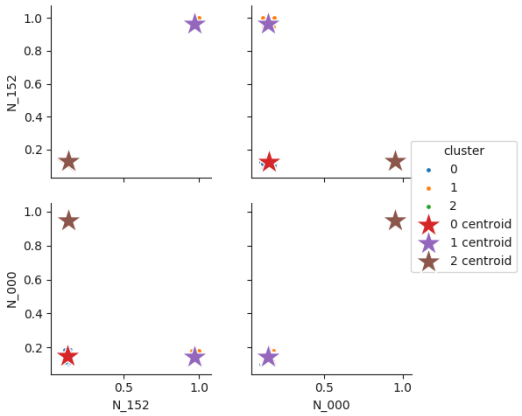

Figure 3.16: Clustering visualizaion of the second time window.

From the exploration of Rabobank transaction dataset, we showed that by using this concept drift investigation framework, we could detect the drifts in the datasets and localize the detected drifts with the help of the visualization application. Meanwhile, the visual investigation of the machine learning models can help to localize further and understand how does the drift happen.

Chapter 4

Drift detection ensemble based on alternative label

In this chapter, we describe a novel approach for concept drift detection when the class labels of the dataset are not directly available. The novel approach uses an ensemble of multiple drift detectors based on weighted majority vote. The ensemble detects drift on the error rate of an inner online classifier which is trained on the data and a selected feature as the prediction attribute. In short, our method is the first to train a model from the data by selecting an alternative feature as the label and then detect drift based on the performance of the model. Besides, we use a novel way to build the ensemble.

In the following sections, we first illustrate the intuitions and the main idea of this approach, and then we describe each component of this approach, i.e., the inner classifier and the construction of the ensemble. After that, we discuss the advantages and drawbacks of our novel approach. Finally, we review related work that also use an ensemble of drift detectors to detect concept drift.

4.1 Intuitions

Recall the concept drift formulation, let $\mathcal{P}$ be the data generating process which provides a sequence of tuples $(\mathbf{X}_t, y_t)$ sampled from an unknown probability distribution $p_t(\mathbf{X}, y)$ at some arbitrary time t. The goal of a learning algorithm is to predict the target variable $y_t \in \Lambda$ given a set of input features $X_t \in \mathcal{R}^d$ at each time t. Let $p_t(y|\mathbf{X})$ be the posterior distribution and $p_t(\mathbf{X})$ be the evidence distribution. The distributions are deliberately subscripted with time t to explicitly emphasize their time-varying nature. The data may arrive in an online manner, i.e., one single instance $\mathcal{I}_t = \{(\mathbf{X}_t, y_t)\}$ at a time, or in a batch setting which provides a set of tuples $\mathcal{I}_t = \{(\mathbf{X}_t^1, y_t^1), \ldots, (\mathbf{X}_t^N, y_t^N)\}$ at each time.

Based on the cause and effect of the changes, two types of drift are distinguished:

- *Real Drift*: the posterior probability $p(y|\mathbf{X})$ changes over time. Such changes can occur either with or without changes in $p(\mathbf{X})$.
- *Virtual Drift*: the distribution of $p(\mathbf{X})$ changes without affecting the posterior probability of classes $p(y|\mathbf{X})$.

For most of the cases in the real world, the class labels $\mathbf{Y}$ of the incoming instances are not directly available, and it may take days or even years to arrive. When the posterior probability $p(y|\mathbf{X})$ changes over time and $p(\mathbf{X})$ does not change, in this case, the drifts are not detectable without knowing the class labels. Besides, the feature vector $\mathbf{X}$ is usually high dimensional, and it is time and space consuming to monitor all the features at the same time. Thus, if we can find

an alternative label x_i from feature vector $\mathbf{X}$ such that the change of $p(y|\mathbf{X})$ leads to a change of $p(x_i|\mathbf{X}\backslash\{x_i\})$. Let $f_1 : \mathbf{X} \rightarrow \mathbf{Y}$ and $f_2 : \mathbf{X}\backslash\{x_i\} \rightarrow x_i$ be two classifiers. Since the change of $p(y|\mathbf{X})$ will reduce the accuracy of f_1, meanwhile $p(x_i|\mathbf{X}\backslash\{x_i\})$ will change which will lead to a decrease on the accuracy of f_2. Hence we can train such a classifier f_2 to simulate f_1, trigger an alarm whenever the error rate of f_2 has significantly increased. Note that the change of $p(x_i|\mathbf{X}\backslash\{x_i\})$ may not always lead to a change of $p(y|\mathbf{X})$, such cases can be regarded as false alarm. And we can use a change detection algorithm f_d to monitor the error rate of f_2.

With this method we can detect some of the "undetectable" real drifts (where the posterior probability $p(y|\mathbf{X})$ changes over time and $p(\mathbf{X})$ does not change). The assumption is that if $\mathbf{X}\backslash\{x_i\}$ and $\{x_i\}$ are generated from the same concept, the prediction of $\mathbf{X}\backslash\{x_i\}$ for $\{x_i\}$ should be stable. If the prediction error suddenly or gradually gets higher, we can monitor $\backslash\{x_i\}$ and $\{x_i\}$ separately and conclude where the change is observed - in one or in both. Meanwhile, if x_i is highly related to the class label, then a real drift detected here could correspond to an "undetectable" real drift in the whole dataset.

4.2 Main idea

The main idea of our novel approach can be decomposed into several steps and components. The workflow and the components of the approach are shown in Figure 1. As we can see in the workflow, the first step is to get the alternative label based on the historical data. Here we assume that we have access to some historical data where the class labels are available, we identify the best alternative label through feature selection as shown in Section 4.3, and it will be used as prediction attribute in our detection method.

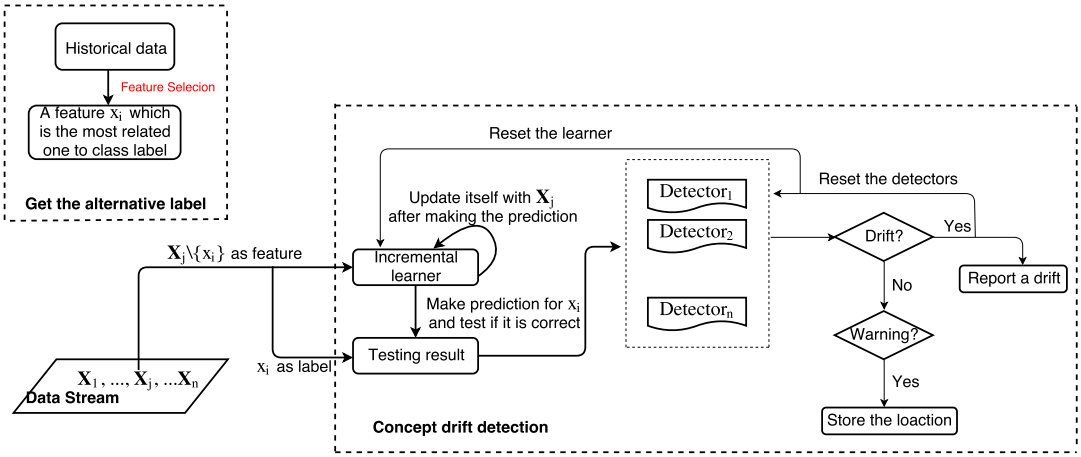

Figure 4.1: Components and workflow of the novel approach.

In the detection phase, we take the input dataset as a stream, where the detector processes each instance of it at a time. The inner incremental learner follows prequential evaluation: for each instance $\mathbf{X}_j$, it first makes the prediction on x_i by using $\mathbf{X}_j\backslash\{x_i\}$ as features, then it uses $\mathbf{X}_j$ to update itself. The test result is either 0 (incorrect) or 1 (correct) which will then be sent to the detector ensemble. The ensemble makes the decision on whether a drift has been detected based on the results of all its detectors by using weighted majority voting. We choose to use weighted voting because different detectors may good at identifying different types of drifts, from practical experiments we can define the weights for each of them. If a drift has been triggered, then it will report the result and reset all the detectors as well as the incremental learner, otherwise, if either of the detectors has reached warning level, we will store the location. We define the drift location to be the nearest warning location if any warning level has been triggered when it triggers a drift.

Otherwise, we take the location of the drift level as the drift location. Since there is usually a delay between the reported warning location and the real drift location, we still need the visualization tool to help us to locate the drifts further.

Figure 4.2 illustrates the idea of drift detection in our ensemble as well as in many existing detectors, which on the basis of incoming information about new examples and their correct classification or a classifier's performance (as accuracy) can return information that data stream distributions are changing. The currently used detectors return only signal about drift detection, which usually requires a quick classifier's model updating or that warning level is achieved, which is usually treated as a moment that new learning set should be gathered (i.e., all new incoming examples should be labeled). The new learning set is used to update the model is drift is detected.

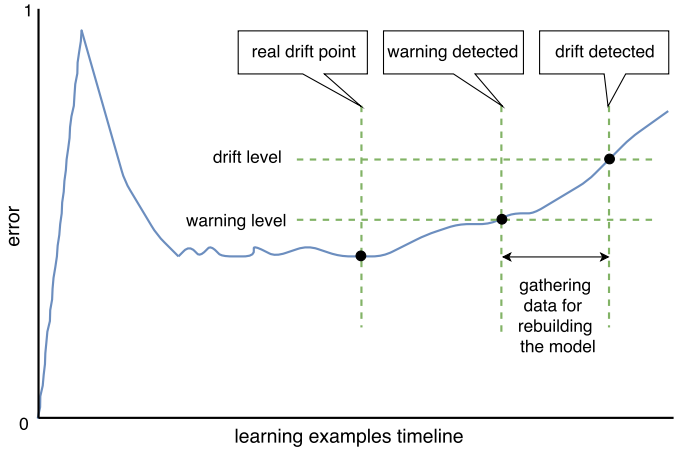

Figure 4.2: Idea of drift detection.

In the following sections, we describe each step in more details (i.e., feature selection, incremental learner and the ensemble strategy). Then give an example of constructing such ensemble in our approach. Finally, we review and discuss related works that are similar to this ensemble approach.

4.3 Feature selection

The problem remains how to find such x_i from the feature vector $\mathbf{X}$ and the choice of classifier f_2. From the description of our approach above we know that x_i should have a strong correlation with the class label y and this relation does not change over time. To meet the first requirement, we can perform a feature selection on the dataset to find the best feature. As for the second requirement, it is rather a strict one since the distribution of class label y could change as time passes, we can not make sure the feature we find would also change in the same manner. Instead, we can choose a feature which has lower standard deviation than the other features, which means that it is stable over time. Hence, assume that we have a collection of data $\mathcal{X}$ with class label $\mathcal{Y}$, we can find such x_i by performing a feature selection on $\mathcal{X}$ and $\mathcal{Y}$ to get best feature x_i which has similar behavior as the class label y in terms of $p(\mathbf{X}, x_i)$ and $p(\mathbf{X}, y)$. By using this feature selection approach, we can find a feature that has similar behavior to the class label. Moreover, when using this feature as prediction attribute, we can detect some changes in the distribution of y by monitoring the changes of this feature.

Now we need a suitable method for the feature selection. The feature selection methods can be divided into 3 main categories: filter methods, wrapper methods and embedded methods:

- **Filter Methods**: this type of methods are generally used as a preprocessing step. The selection of features is independent of any machine learning algorithms. Instead, features

are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable. The correlation is a subjective term here.

- **Wrapper Methods**: in wrapper methods, we try to use a subset of features and train a model using them. Based on the inferences that we draw from the previous model, we decide to add or remove features from the subset. The problem is essentially reduced to a search problem. These methods are usually computationally costly.
- **Embedded Methods**: this type of methods combine the qualities of filter and wrapper methods. It is implemented by algorithms that have their built-in feature selection methods. Some of the most popular examples of these methods are LASSO and RIDGE regression which have inbuilt penalization functions to reduce overfitting. Other examples of embedded methods are Regularized trees, Memetic algorithm.

We want to find the best feature x_i which has similar behavior as the class label y in terms of $p(\mathbf{X}, x_i)$ and $p(\mathbf{X}, y)$. So finding a feature with the best correlation with the label (filter method) is the best option to meet our requirement. In the filter feature selection method, we use χ^2 test to score all the features and then get the one with the highest score, since χ^2 test is the most common one for classification problem on groups of categorical features. Other statistical tests such as ANOVA can also be applied here.

4.4 Inner incremental learner

As for the learner f_2, the first requirement is that it needs to be incremental since we have to monitor the change of the error rate in real time with the incoming data stream. Besides, it should be able to handle high dimensional data. Moreover, it needs to be easy to implement due to the limitation of time and we want to do some experiments to validate the proposed method. The following models are

1. Naïve Bayes
2. Hoeffding Tree[14]
3. K Nearest Neighbours (KNN)

These three models are either based on decision tree or based on pure statistical theories. Naïve Bayes and KNN are two easy to implement classifiers. However, when the data is high dimensional, the probabilities in Naïve Bayes will be rounded to 0 due to the fact that computer stores numbers in a fixed amount of bits. Normally the distance metric uses in KNN is Euclidean distance. This distance measure becomes meaningless when the dimension of the data increases significantly because all vectors are almost equidistant to the search query vector (imagine multiple points lying more or less on a circle with the query point at the centre, the distance from the query to all data points in the search space is almost the same). Thus, these two models are not suitable in our case. The Hoeffding Tree, on the other hand, is capable of learning from massive data streams and it can handle high dimensional data. Although there are some existing practical implementations of Naïve Bayes on high dimensional data, we choose to use Hoeffding Tree for high dimensional datasets because it naturally can handle this kind of data. Moreover, we use Naïve Bayes to test on low dimensional datasets.

4.5 Construction of the ensemble

To build up our approach, the last thing to do is to find a change detection algorithm f_d to detect the drift on the error rate of the online learner. The requirement for such detection algorithm is that the false alarm of it should be as low as possible. Because a false alarm here will eventually lead to a false alarm on $p(y|\mathbf{X})$. There are many existing drift detection methods. We choose to

design an ensemble of some of the methods to be f_d since different detectors are good at detecting the different type of drifts. In this section, we first show the decision method of our ensemble, then show how to build the ensemble.

4.5.1 Decision method of the ensemble

Ensemble methods are techniques that create multiple models and then combine them to produce improved results. Ensemble methods usually produce more accurate solutions than a single model would. For ensemble methods in machine learning, there are several ways to combine the prediction results of the classifiers within an ensemble: voting, stacking, bagging and boosting. Stacking, bagging and boosting are usually used when there is a training phase in the procedure. While in our case, the drift detectors all make a decision based on some statistical property of the data, so voting is the one suits our scenario. Besides, there are several ways to perform the voting method, assume we have four models, M_a, M_b, M_c, and M_d:

- **Majority Voting**: Every model makes a prediction (votes) for each test instance, and the final output prediction is the one that receives more than half of the votes. If none of the predictions get more than half of the votes, we may say that the ensemble method could not make a stable prediction for this instance. e.g., if the prediction of M_a, M_b, M_c are the same, we use this prediction as final result, if only M_a and M_b have same results, we can not make a stable prediction for the input instance because none of the none of the predictions get more than half of the votes. Although this is a widely used technique, the most voted prediction (even if that is less than half of the votes) can also be selected as the final prediction, also called "plurality voting".

- **Weighted Voting**: Unlike majority voting, where each model has the same rights, we can increase the importance of one or more models. In weighted voting, we count the prediction of the better models multiple times. e.g., if we give M_a weight of 4 we will count M_a 4 times for its prediction.

- **Simple Averaging**: In the simple averaging method, for every instance of the test dataset, the average predictions are calculated. This method often reduces overfit and creates a smoother regression model. e.g., assume M_a, M_b, M_c, and M_d are regression models, then the result of their ensemble is the average of each individual results.

- **Weighted Averaging**: weighted averaging is a slightly modified version of simple averaging, where the prediction of each model is multiplied by the weight, and then their average is calculated. e.g., we give different weights to M_a, M_b, M_c, and M_d, assume M_a, M_b, M_c, and M_d are regression models, then the result of their ensemble is the weighted average of each individual results.

In drift detection scenarios, authors of [49] performed experiments on ALO (At Least One Detects Drift), ALHD (At Least Half of the Detectors Detect Drift) and AD (All Detectors Detect Drift). Du et al. in [16] proposed to use ALO because it can locate a concept drift as early as possible, however, false alarms will increase in this case. There is always a trade-off between the number of false alarms (false positive rate) and the recall rate (true positive rate).

Each base detector is good at different scenarios, hence, we choose to use weighted majority vote, where we combine the results of different detectors by giving different weights to each of them. Assume we have k drift detectors $F = \{(f_{d1}, w_2), \ldots, (f_{d_k}, w_k)\}$. Each f_{di} is a change detector which has weight w_i in the ensemble. Use p_t to denote the prediction result of f_2 at time t, $p_t = True$ means the prediction of f_2 is correct and vice versa. Each detector f_{di} takes p_t as input and outputs another binary value r_t to indicate wether a drift has happened. Algorithm 1 shows the pseudocode for the decision procedure of the ensemble.

Algorithm 1 DECISION PROCEDURE OF THE PROPOSED ENSEMBLE METHOD

procedure ENSEMBLE(X_t, x_{it})
  $X_{new} = X_t \backslash \{x_{it}\}$
  $p_t = False$, $result = 0$
  if $f_2(X_{new}) == x_{it}$ **then** $p_t = True$
  for (f_{di}, w_i) **in** F **do**
    if $f_{di}(p_t)$ **then**
      $result += w_i$
  return $(result > 0.5)$

4.5.2 Construction procedure of the ensemble

In this Subsection, we propose a way to select base detectors and to generate the weights of the detectors in ensemble and give an example for the whole procedure.

Although an ensemble which contains as many detectors as possible can help locate more concept drifts, indiscriminate usage of detectors is not the best choice for time-changing data streams. For example, suppose detector f_{d1} outperforms f_{d2} on abrupt drifts, while f_{d2} is better than f_{d1} on gradual drifts; then the ensemble of them will be more powerful than each of them since the ensemble performs well for detecting both abrupt and gradual drifts. However, if f_{d1} is better than f_{d2} on both abrupt and gradual drifts, the ensemble of them will be less effective than f_{d1}. In this case, we can use selective ensemble paradigm, which only chooses the most effective detectors, to select the most effective detectors. This kind of method is used in [16]. Another way to solve the problem is by giving different weights to the detectors. i.e., assume f_{d1} is better than f_{d2} on both abrupt and gradual drifts, if we can find a way to score these two detectors, and the one with higher score will get a higher weight in our ensemble, the ensemble of f_{d1} and f_{d2} can still be as well as f_{d1}.

When comparing the performance of drift detectors, there are some metrics we can use.

- **False negative rate of a detector (FN_{f_d})**: a false negative is when we receive a negative result but we should have received a positive one, which is due to the detector did not manage to detect the drift. The false negative rate of a detector should be as low as possible since we do not want to miss any potential drifts.
- **False positive rate of a detector (FP_{f_d})**: a false positive is where we receive a positive result for a test, when we should have received a negative result, it is often referred to as "false alarm", This measure characterizes the resilience to false alarms when there is no drift, that is detecting drift when there is no change in the target concept. The false positive rate of a detector should also below, under the condition that the detector has found all the "true drifts".
- **Drift detection delay (D_{f_d})**: these measures give the estimate of how many new instances are required to detect a change after the actual occurrence of a change (or how much time would elapse before the change is detected).
- **Memory usage (M_{f_d}) and Runtime (R_{f_d}) of detector**: these two metrics can be used in an online learning procedure where there is a high demand for detectors which are both quick and have low memory usage.
- **Error rate of the classifier (E_f)**: in most cases, the main goal for concept drift detection is to get the best performance of a learner f which is applied on a time-changing data stream. We can make some changes on the learner whenever the detector has triggered a drift. Thus, the performance of the learner, i.e., the accuracy of it, can also be an evaluation metric of the drift detector. Other evaluation measures of the learner such as Kappa-Statistic, Partial AUC, mean square error and so forth can also be applied here.

The first three evaluation metrics measure the performance of the detector whereas the Memory usage and Runtime measure the resource usage of the detector. The last one measures the adaptation of the learner to the drift, which can also be regarded to be a metrics for the detector because the adaptation depends extremely on the detection result. All the above-mentioned evaluation metrics are the lower the better. We then use the CAR measure proposed by [38]:

$$\begin{aligned} \mathrm{CAR}_{(f,f_d)} &:= \mathrm{Classification}_f \diamond \mathrm{Adaptation}_{f_d} \diamond \mathrm{Resourse\ Use}_{f_d} && (4.1)\\ &:= E_f \diamond (D_{f_d} \oplus FP_{f_d} \oplus FN_{f_d}) \diamond (M_{f_d} \oplus R_{f_d}) && (4.2)\\ \mathrm{Score}_{f_d} &= 1 - \mathrm{CAR}_{(f,f_d)} && (4.3) \end{aligned}$$

Note that in Equation 4.1, the $\diamond$ and $\oplus$ symbols only represent the combination of three components. All the values are normalized using the 'min-max' scaling approach. We then can have the score generating procedure as shown in Algorithm 2. Here we assume that these six factors are of equal importance, of cause we can also give each factor a weight to get a weighted score.

Algorithm 2 GENERATING SCORES

```
procedure SCORE(f, f_d)
    FN_{f_d}, FP_{f_d}, D_{f_d}, M_{f_d}, R_{f_d}, E_f = compute_performance(f, f_d)
    return 1 − CAR_{(f,f_d)}
```

The scores are calculated in an online setting. It can be used to evaluate the detectors at each timestamp t. We want to give a higher weight to the detectors which have a higher score in general on a data stream with concept drift. However, in order to get the score value, it requires the ground truth for the concept drift, while in real-world data it is rather difficult to get them. Alternatively, we can generate a synthetic dataset, and put some desirable type of drift in it. i.e., we want to detect gradual drift, so we can some generate synthetic datasets which contain gradual drifts and then score the candidate detectors on, finally assign weights based on the scores.

Algorithm 3 GENERATING WEIGHTS

```
procedure GW(datasets)
    for i = 0 to n do                                   ▷ Initialize the weights
        W_i ← {}
        w_i = 0
    for dataset in datasets do
        F_d = {(f_1, f_d1), ..., (f_n, f_dn)}           ▷ A set of classifier f_i and drift detector f_di
        for X_t, y_t in dataset do
            for (f_i, f_di) in F_d do
                y_pti = f_i.test(X_t)
                p_ti = (y_pti == y_t)
                drift = f_di(p_ti)
                if drift then
                    f_i.reset()                         ▷ Initialize f_i
                else
                    f_i.train(X_t, y_t)
                score_it = Score(f_i, f_di)
        for i = 0 to n do
            count_i ← number of times f_di has the highest score
            W_i = W_i ∪ {count_i/dataset.length}
    for i = 0 to n do
        w_i = W_i.mean
    return w_1, ..., w_n
```

Assume we have a set of datasets and a collection of candidate drift detectors $\{f_{d1} \ldots, f_{dn}\}$. We generate the weight of each detector in the following way. We first construct a set of tuples in the form of $\{(f_1, f_{d1}), \ldots, (f_n, f_{dn})\}$ by distributing a same classifier to each detector. In each $(f_i, f_{di}$, f_i is an online classifier and f_{di} is a drift detector. We then load each of the datasets as an online stream. For each instance in the stream, we use prequential evaluation on the online classifier of each tuple: test on the instance then train based on the instance. The detector of each tuple is then be used to detect drift on the error rate of the online classifier. Whenever a drift has been detected in a tuple, we reset the learner and the detector in that tuple. Besides, after processing an instance in all the tuples, we can then compute the score for each detector. For each dataset, we count the number of times that a detector scored the best and divide this count by the number of instances in the dataset to be the overall score of the detector on this dataset. After processing all the datasets, we compute the average of the overall scores for each detector as its weight in our ensemble. In short, the weight of a detector is the average percentage of it to be the best over all the datasets. Algorithm 3 shows the pseudocode for this weight generating procedure.

4.6 Discussion of the novel approach

We propose this novel approach by addressing the challenge that the class labels of the data arrive at a delay, and we would like to detect some of the "undetectable" real drifts without the true labels. By choosing an alternative feature which is pretty similar to the true label, hypothetically we can achieve this goal.

There are several advantages of our novel method. To begin with, it does not need the class label during the detection, and it provides flexible choice for the domain expert to set the label in it. With the knowledge from domain experts, we can easily find some good features to be the alternative label. Besides, it provides flexibility for choosing the inner learner and the drift detectors, any learner and detector meet the requirements stated in Section 4.4 and 4.5 can be used here. Finally, this method can be applied to both multivariate time series data and text data with temporal information. For text data, we can extract the tf-idf matrix of the words at each timestamp as feature space, and choose one word or one LSI or any other embedding as the alternative label in our approach. In this thesis, we do not have the true labels for Rabobank email data, so we are not going to apply it to the email data.

However, the drawback of it is obvious. Consider the real drift scenario where only $p(y|\mathbf{X})$ changes without changes in $p(\mathbf{X})$. In this case, none of the features from feature space $\mathbf{X}$ can represent the similar concept as that of y so that our method may fail. Meanwhile, if the chosen label itself either fluctuates a lot or is stable all the time, our method will either trigger plenty of drifts or output nothing. We will simulate the scenario where our method performs badly in the experiences of Chapter 5 and further confirm this.

4.7 Related work on ensemble of drift detectors

So far not too many research works focused on the combination of drift detectors. Most of the previous studies focused on using an ensemble of classifiers and detecting concept drifts based on the performance of the classifiers in the ensemble, such as the enhanced adaptive classifiers-ensemble (ACE) system in [34], Batch Weighted Ensemble (BWE) [13], Diversity for Dealing with Drifts (DDD) [32] and etc. Here we only discuss the previous works which studied the ensemble of drift detectors as they are more related to our novel method.

Drift Detection Ensemble (DDE) [30] uses a small ensemble of detectors to make decision about the drift. It has 3 default configurations of concept drift detectors, which depend on the chosen level of sensitivity. Each configuration contains 3 detectors, and they perform drift detection on

the error rate of the base classifier. DDE uses the sensibility as a parameter, different sensibility has the different combination of the detectors, and it also affects the drift and waning level decision. i.e., if the sensibility is 2, then no less than 2 detectors in the ensemble should be at drift level in order to trigger a drift, and if the sensibility is 1, the drift can only be triggered when no less than 1 detectors in drift level, etc.

In [16], Du et al. proposed a method which is similar to our novel method. Their detection method has one incremental classifier, and it keeps an ensemble of base detectors called e-Detector. They proposed a selection method for the detectors in the ensemble, and it can detect both abrupt and gradual drifts. The ensemble follows the rule that if any base detector finds a concept drift, e-Detector declares it. However, this ensemble method may lead to a number of false alarms in the detection results.

More recently, the authors of [49] did some experimental studies by comparing different ensemble strategies with simple methods. The experiment results showed that the ensembles did not perform better than simple drift detection methods.

Apart from the ensemble of drift detectors, concept drift detection on high dimensional data without class labels is also related to our work. In process mining, the event logs used for mining patterns are usually very high dimensional, and there do not exist labels in the event logs. In [11] the authors proposed features and techniques to detect changes (drifts), change points, and change localization in event logs from a control-flow perspective.

Besides, concept drift detection by using clustering techniques can also handle high dimensional data in an unsupervised learning context. [46] uses k-means clustering and additionally maintains a list of objects that do not fit the current clustering w.r.t. a "global boundary". With this method, the technique can detect novelty and concept drift by a single strategy. More recently, Abdallah et al. [1] proposed an adaptive ensemble approach for multi-class novelty detection. The proposed method was based on a two-step cluster formation. Firstly a supervised learning method was applied to divide the initial data into class-based clusters. Then, unsupervised learning was applied to detect sub-concepts within each cluster and thus to create more local models.

Another related direction is concept drift detection based on clustering ensembles. Al-Khateeb et al. [3] proposed a novel ensemble technique which is a class-based ensemble called Class-Based Micro Classifier Ensemble (CLAM). It creates an ensemble of models for each class found by k-means clustering. CLAM has been very successful in detecting novel classes and preventing recurring classes from being detected as a novel class, thereby increasing the classifier accuracy.

Some other previous works focused on semi-supervised clustering ensembles which require a few labels in non-stationary data streams. Hosseini et al. [25] proposed an ensemble of semi-supervised clustering algorithms, where each class is described by a single model. Each new incoming chunk obtains a pre-defined number of labeled instances, which are used to update classifiers in the ensemble. Chunks are assigned based on a semi-supervised Bayesian approach. Authors claim that their approach is able to recognize recurrent concepts within the data stream automatically. Masud et al. [31] proposed an approach where micro-clusters are generated using semi-supervised clustering and a combination of these models are used to handle unlabeled data incoming from the stream. A label propagation technique is used to assign each micro-cluster to a class. Then, inductive label propagation is used to classify a new instance. New micro-clusters can be added to an ensemble with the progress and changes in the stream.

Thus, none of the approaches uses an alternative label idea that we employ. Our approach is expected to perform better when we can find a feature that has almost the same behavior as the actual class label. In that case, our approach can detect almost all the real drifts. However, our approach would get bad performances when all the features are not similar to the class label.

Chapter 5

Experiments on the novel approach

In this chapter, we report the results of different experiments on the real transaction dataset provided by Rabobank, open dataset and synthetic data. More specifically, we first compare the novel ensemble method with individual detectors which require class labels and see whether our method performs better. We then compare our approach with a drift detection competitor aiming at high dimensional data. For performance evaluation metrics, since we do not have the ground truth for the drifts of real-world datasets, we can only use the error rate of a classifier as described in Section 4.5.2. We assign each candidate detector a same online learner, whenever the detector has triggered a drift, we reset the learner and the detector. Moreover, we evaluate the performance of the detectors based on the error rate of the learner. For synthetic datasets, we can also evaluate the detectors by using true positive and false positive from the detection result.

In the following sections, we discuss the experiments on the novel ensemble detection method in detail. We first show the setups for the experiments, e.g., goals of the experiments, dataset description, construction of our ensemble, baseline methods as well as the experimental procedures. After that, we show the experiment results of the performance comparison between our novel ensemble method and individual detectors as well as a competitor detection method. Finally, we give conclusions on all the experiments.

5.1 Experiment setup

In this section, we show the experimental settings including the goals of the experiments, the dataset description, the construction of our ensemble method and provide a brief recap of the baseline methods. The experiments are conducted with the Tornado framework described in [38]. All the experiments reported in this Chapter are performed on a Mac OSX laptop with an Intel Core i7 3.1 GHz processor, 16 GB 1867 MHz DDR3 memory.

5.1.1 Experiment goals

The goal of the experiments is to test the proposed novel approach with different datasets and compare it with baseline methods and competitor methods. Firstly, in order to study whether the proposed ensemble method is better than individual detectors, we use individual detectors baselines and compare them on different datasets. Secondly, as we use weighted voting for the ensemble strategy, it is interesting to compare it with other alternative strategies as see whether it outperforms others. Finally, our approach aims at detecting drifts on multivariate data without access to class labels. We would like to compare it with existing competitors which also have the same focus and see whether our method can get similar performance as the competitors.

5.1.2 Dataset description

Rabobank Transaction dataset

In the raw data, each row of it represents one transaction, and each transaction contains a label indicating whether the transaction has been confirmed as abnormal or normal, a fraud class to show the class of the fraud, a timestamp indicating the transaction time, and a unique hash that can be used to identify the transaction. Besides the label, timestamp, and hash, each transaction also has thousands of features. These features can be numerical, categorical or textual. In order to use the data for machine learning tasks, the raw transaction data has been discretized and anonymized in the following way (by Rabobank):

- **C (categorical)** - one to one encoding: the unique values are first sorted according to ASCII character order, then the feature values are mapped to the bin values between 0.1 and 0.9 according to that order. Each category corresponds to one bin value, and all the values between two consecutive categories also correspond to one bin value. For example, if there are 2 unique categories A and B, then $x = A$ corresponds to bin value 0.3, $A < x < B$ corresponds to bin value 0.5, and $x = B$ corresponds to bin value 0.7, so all the categories and their possible values in between are evenly spread in the range of (0.1, 0.9). Since categorical variable has at most 200 unique values, there are at most 400 bin values for each categorical variable.
- **N (numerical)** - equal frequency based: the unique values are sorted according to numeric order, then the bins with different widths but approximately equal frequencies are generated. There are at most 800 bins (bin values ranging from 0.1 to 0.9, in unit of 0.001). Hence, except some very frequent and identical values, each bin is expected to contain value range that has total frequency around $1/800 = 0.125\%$. The bin id value is taken as the middle point of the bin, for example, if the smallest value 0 has frequency 87.5% (it takes 700 bins), the second smallest value 1 has frequency 0.25% (2 bins), and the third smallest value 2 has frequency 0.125% (1 bin), then the first bin id is $(0.1+(0.1+7000.001))/2 = 0.45$ (original values ranging from $0 \leq x < 1$), and the second bin id is $((0.1 + 7000.001) + (0.1 + 7020.001))/2 = 0.801$ (original values ranging from $1 \leq x < 2$), and so on.
- **T (textual)** - equal frequency based: the unique values are sorted according to ASCII order, then applies the same frequency based binning process as N variables.
- **Global binning setting**: besides the above binning process, the values that are smaller than the minimum value of the binning sample are mapped to the bin id 0.05, values that are larger than the maximum value of the binning sample are mapped to the bin id 0.95, empty value is mapped to 0, and the error value is mapped to 0.999. Hence, for all the variables, the binned values float numbers in a unit of 0.001, ranging from 0 to 1.
- **Removal of artifacts and useless features**: remove variables that are exactly equivalent to the timestamp but in a different encoding (e.g., encoded in numbers), since such variables may give perfect but useless predictions when the time periods of the abnormal and normal transactions in the training set are very different.
- **Anonymization**: anonymize the original feature names by mapping them to a new set of anonymized feature names. Moreover, any information related to the customers have been removed in the binning stage (since all the features are binned to float numbers between 0 and 1).

After this binning step, the dataset which we are using contains 585 features except for the timestamp, hash, label and the fraud class. In this thesis, we only use all the 585 features (which includes 394 C features, 174 N features, and 17 V features), timestamp and the label in it, and we do not take fraud class into account. Table 5.1 shows an example of the transformed dataset, which we use directly for the pre-processing step as described in Section 3.2.1.

Table 5.1: Example of the transformed dataset

Timestamp	META_HASH	label	fraud_class	C_000	···	N_000	···	V_000	···
2016-10-05 14:40:42	wE1CtE	1	Mo1	0.1	···	0.142	···	0.1	···
2016-10-11 11:44:26	+ERQsO	1	Mo1	0.102	···	0.186	···	0.105	···
···	···	···	···	···	···	···	···	···	···
2017-09-30 18:01:07	/VLTAH	0	Mo0	0.1	···	0.166	···	0.1	···

Elec2 (Electricity Market data) [23]

This dataset comes from Australian New South Wales Electricity Market. It is about the electricity price being adjusted according to the demand and supply in the market. The price is set every 5 min. This dataset has 45312 examples dated from May 1996 to December 1998, and each example has six attributes: day of the week, the timestamp, NSW electricity demand, Vic electricity demand, scheduled electricity transfer between states, class label, which spans a period of 30 min. Thus there are 48 examples in a day. The class label stands for the change of the price of a moving average over the past day.

Synthetic Data Streams

MIXED · *with gradual and sudden drift.* Since in the Rabobank transaction dataset there are both numeric attributes and categorical features, we choose to create MIXED data streams which have been used in many previous papers i.e.[18] and [35]. The data stream is created by the following strategy: the dataset has two numeric attributes x and y distributed in the interval $[0, 1]$ with two boolean attributes v and w. The instances are classified as positive if at least two of the three following conditions are satisfied: $v, w, y < 0.5 + 0.3sin(3\pi x)$. After each context change, the classification is reversed. We have placed drift points at every 20,000 instances in it with a transition length of 50 to simulate abrupt concept drifts and 500 to simulate gradual concept drifts. Following Bifet et al. [7], we use the sigmoid function to simulate abrupt and gradual concept drifts. The function determines the probability of belonging to the new context during the transition between two contexts. Each stream contains 100,000 instances with 2 gradual drifts and 2 sudden drifts.

LED · *with gradual and sudden drift.* The objective of this dataset is to predict the digit on a seven-segment display, where each digit has a 10% chance of being displayed. The original generating process of the dataset has 7 attributes related to the class and 17 irrelevant ones. Concept drift is simulated by interchanging relevant attributes [17]. To generate high dimensional data, we create 100 irrelevant features instead of 17. Same as MIXED dataset, each stream contains 100,000 instances with 2 gradual drifts and 2 sudden drifts. The abrupt concept drifts have transition length of 50, and 500 for gradual concept drifts.

5.1.3 Construction of the ensemble

Here we give a practical example of how we generated the weight for our ensemble in the experiments in Chapter 5. The candidate drift detection methods are ADWIN, CUSCUM, DDM, EDDM, FHDDM, FHDDMS, PAGE HINKLEY, SeqDrift2, HDDM.A.test, and HDDM.W.test. The classifier f_i of each pair is Naïve Bayes for all the pairs. We use 10 generated MIXED datasets, as described above, each contains 100,000 instances with 4 drifting points: 2 gradual drifts and 2 abrupt drifts.

As shown in Table 5.2, ADWIN, FHDDM, FHDDMS, PAGE HINKLEY, HDDM.A.TEST and SeqDrift2 performed worse compared to the other 4 detectors. We then removed these 4 and recomputed the weights for the others. Table 5.3 shows the actual weight of the 4 selected detectors.

We then can use these weights to build the ensemble detector and perform some experiments to compare with others.

Table 5.2: Weight generating results.

dataset	ADWIN	CUSUM	DDM	EDDM	FHDDM	FHDDMS	PH	HDDM.W	HDDM.A	SeqDrift2
1	25	19203	758	18721	46841	1178	1200	11976	98	0
2	24	17865	664	19839	46896	1085	85	13445	97	0
3	24	11564	676	19293	53230	1073	629	13414	97	0
4	25	18670	797	18924	45987	1056	999	13445	97	0
5	24	16636	698	19681	47856	1066	242	13700	97	0
6	24	9299	616	19858	55503	1052	64	13487	97	0
7	24	17662	695	19619	47166	1070	297	13369	98	0
8	24	18065	744	19203	46717	987	719	13444	97	0
9	25	7279	683	19774	57361	1092	149	13540	97	0
10	25	16423	726	19601	48321	1062	322	13422	98	0
average	24.4	15266.6	705.7	19451.3	49587.8	1072.1	470.6	13324.2	97.3	0

Table 5.3: Weight generating results after removing the 6 detectors.

	1	2	3	4	5	6	7	8	9	10	weight
CUSUM	19203	17865	11564	18670	16636	9299	17662	18065	7279	16423	0.16
EDDM	18721	19839	19293	18924	19681	19858	19619	19203	19774	19601	0.2
FHDDM	46841	46896	53230	45987	47856	55503	47166	46717	57361	48321	0.5
HDDM.W	11976	13445	13414	13445	13700	13487	13369	13444	13540	13422	0.14

5.1.4 Baseline methods

In order to test the performance of the proposed method, we compare it with 3 sets of baseline methods: the individual detectors, the ensembles with different strategies and state of the art drift detectors.

- **Individual detectors**: these are the 4 methods that we use in our ensemble: CUSCUM, EDDM, FHDDM, and HDDM.W.test. Besides, we also use a no-detector which doesn't detect anything and return the data is not drifting for all the time.
- **Other decision strategies**: we compare our weighted majority strategy with ALO (At Least One Detects Drift), ALHD (At Least Half of the Detectors Detect Drift) and AD (All Detectors Detect Drift), which are the 3 combination rules tested by [49].
- **Competitor method**: the algorithm LIGHT described in [33] is the main competitor we would like to compare with because it also aims at detecting drifts on high dimensional data without requiring class labels.

We use the default parameters for these detectors. In CUSUM we have $min\_instance = 30$, $\delta = 0.005$ and $\lambda = 50$. In FHDDM $n = 100$ and in HDDM.W.test $\lambda = 0.05$. In LIGHT we set the window size to be 576 for Rabobank dataset because there are 288 five minutes in 24 hours, we would like to compare the data between two days. For LED dataset, we use its default value which is 1000. As for our method, we first need to perform feature selection to get the best feature that correlated to the class label. For Rabobank dataset, the feature "C_128" is the selected feature. Feature "day" is selected in elec2, feature "v" is selected in MIXED dataset feature "a1" is selected in LED dataset.

5.1.5 Experiment procedure

We test all the detect methods each with an online classifier. The classifier will be reset whenever a detector has triggered a change in the error rate of the classifier. Each of the classifiers in the reservoir incrementally accepts the incoming instances, one at a time, and proceeds to build a model. Each classifier is combined with each one of the drift detectors. That is, classifier f_{c1} is combined with drift detectors $f_{d1}, f_{d2} \ldots, f_{dm}$ to form the pairs $(f_{c1}, f_{d1}), (f_{c1}, f_{d2}), \ldots, (f_{c1}, f_{dm})$, and so forth. The incremental classifier f_c follows prequential evaluation: for each instance X_j, it first makes prediction on y_j by using X_j as features, then it uses (X_j, y_j) to update itself. The test result is either 0 (incorrect) or 1 (correct) which will then be sent to the detector. The detector makes a decision on whether a drift has been detected. If a drift has been triggered, then we will reset all the detectors as well as the incremental learner in the same pair. As for the learner, we only apply Naïve Bayes on elec2 and MIXED, as it is not able to make predictions on high dimensional data.

Each of the individual detectors performs drift detection directly on the real-time error rate of its learner. Our proposed method, as described above, first select a feature x_i from the feature space as the alternative label. After that, whenever a new instance comes in, the inner incremental classifier will take all the other features except x_i as the new feature, and x_i as the label to build a model. All the ensemble methods will perform drift detection on the error rate of its inner classifier. Besides, the algorithm LIGHT also only requires the feature from the incoming data stream.

We ran each classifier-detector pair 10 times on the two real-world datasets and 100 MIXED datasets and 10 LED datasets. At last, we take the average error rate of the classifier of each pair in all the runs as result to compare.

Besides, since LED is the only high dimensional dataset where we have the ground truth of the concept drifts, we also compare the TP and FP of the detection result between LIGHT and DDEAL on the LED dataset. When calculating TP and FP, we increment the number of true positives when the drift detector alarm is within the acceptable delay range. Otherwise, we increment the number of false negatives, since the alarm has occurred too late. We set the acceptable delay range to be 250 for the LED dataset.

5.2 Experiment results

In this section we show the experiment results. We highlight the result of our method with different colors: blue means it performs as good as other methods, red means it performs worse than others.

Table 5.4: [Lower is better] Average error rate of Hoeffding Tree combined with different detectors.

Detector	Rabobank	elec2	MIXED	LED
HT + FHDDM	0.100	0.181	0.181	0.294
HT + CUSUM	0.096	0.170	0.171	0.297
HT + EDDM	0.091	0.168	0.166	0.295
HT + HDDM.W.test	0.104	0.185	0.183	0.325
HT + DDEAL	0.088	0.197	0.193	0.44

Table 5.4 and 5.5 show the experiment results of the proposed drift detection ensemble based on alternative label (DDEAL) compared with individual detectors on different datasets. We can see that by using our method, the online learner (Hoeffding Tree or Naïve Bayes) can actually get the similar performance or at least as good as the pairs with other detectors on the two real world datasets. However in the two synthetic datasets, we can see that our method performs the worst

among all the detectors, this is because that in MIXED, the drifts are created only by flipping the labels, which are the real drifts without changes in the feature vector (the third illustration shown in Figure 2.1). In LED, the drifts are created by changing the relevant feature over time, meaning that our selected feature can become irrelevant. In these two cases our method would have poor performance since we assume that we can find one feature from the feature space that have the similar changing behavior with the true label. This result further confirms our assumption on the limitation of our method in Section 4.6.

Table 5.5: [Lower is better] Average error rate of Naïve Bayes combined with different detectors.

Detector	elec2	MIXED
NB + FHDDM	0.257	0.155
NB + CUSUM	0.274	0.156
NB + EDDM	0.231	0.149
NB + HDDM.W.test	0.241	0.159
NB + DDEAL	0.257	0.201

When comparing the weighted majority strategy with other strategies, Table 5.6 and 5.7 show the experiment results. We can see that our weighted ensemble has the best performance for Rabobank dataset, elec2 dataset and MIXED dataset among these 4 strategies. For the LED dataset, the error rate of Hoeffding Tree by using weighted ensemble detector are slightly higher than the others. Overall, the weighted ensemble is the best among these 4 strategies.

Table 5.6: [Lower is better] Average error rate of Hoeffding Tree combined with different ensemble strategies.

Detector	Rabobank	elec2	MIXED	LED
HT + ENSEMBLE AD	0.081	0.314	0.307	0.31
HT + ENSEMBLE ALHD	0.098	0.253	0.205	0.347
HT + ENSEMBLE ALO	0.090	0.218	0.262	0.742
HT + DDEAL	0.088	0.197	0.193	0.44

Table 5.7: [Lower is better] Average error rate of Naïve Bayes combined with different ensemble strategies.

Detector	elec2	MIXED
NB + ENSEMBLE AD	0.330	0.367
NB + ENSEMBLE ALHD	0.247	0.277
NB + ENSEMBLE ALO	0.222	0.271
NB + DDEAL	0.255	0.201

Table 5.8 shows the experiment result of our method compared with a competitor method, LIGHT, on the Rabobank transaction dataset and LED dataset. The error rate of Hoeffding Tree together with DDEAL slightly lower than with LIGHT on the Rabobank dataset and the result of DDEAL is much lower than LIGHT on the LED dataset. These two methods both can be used for detecting changes in high dimensional data.

Besides, Table 5.9 shows the number of true positives (TP) and false positives (FP) of DDEAL and LIGHT on the LED dataset. TP is the higher the better whereas lower FP is better. We can see that our method gets better value for FP and they get the same value for TP. Thus, we can

conclude that our method is in line with the competitor approach.

Table 5.8: [Lower is better] Average error rate of Hoeffding Tree combined with DDEAL and LIGHT.

Detector	Rabobank	LED
HT + LIGHT	0.091	0.676
HT + DDEAL	0.088	0.44

Table 5.9: TP and FP of DDEAL and LIGHT on LED dataset.

Detector	TP	FP
HT + LIGHT	1	35
HT + DDEAL	1	15.6

5.3 Conclusion of experiments

From the experiments, we can see that our method can get similar performance as the single detectors which require the class label of the incoming stream. Besides, compared with other ensemble strategies, our weighted majority strategy can get the best performance, and it is actually in line with the competitor change detection method on high dimensional data.

Our method does not require the class label during the detection, and it provides flexible choice for the domain expert to set the label in it. It can be used when labels are not immediately available, i.e., the labels are available after a fixed period of time. However, the performance of it can be bad when there is no change on $p(\mathbf{X})$ so that we can not actually find a feature x_i which has a strong correlation with the real label y. The experiment results in the MIXED datasets confirmed this assumption. Besides, the selected feature can also become irrelevant over time, the experiment results in the LED datasets confirmed this limitation.

Chapter 6

Experiments on text data

In this chapter, we report the results of experiments of DDEAL and Dynamic Topic Model (DTM) with the data provided by Rabobank, open dataset and synthetic data. We apply DDEAL to detect changes on the result from Latent Semantic Indexing (LSI). Moreover, since we want to use DTM to capture the changes of emails over time by generating the topics out of them, we have to make sure that DTM can genuinely show such changes. Besides, the results should be stable and reliable. Thus, in these experiments, we first apply DTM on some corpus where we manually add some changes, i.e., gradual and sudden changes of the documents. We then perform some experiments to show the stability and reliability of DTM. After testing all these properties, we finally test DTM on the Rabobank email dataset to see how it performs on complex real-world data.

In the following sections, we show the experiments of DDEAL and Dynamic Topic Model on text data. We first provide experimental settings including the dataset description, experiment parameters as well as the data sampling strategies for manually creating drifts in the text data. After that, we show the procedure and results of different experiments. Finally, we give conclusions on all the experiments.

6.1 Experiment setup

In this section, we show the experimental settings including the goals of the experiments, the dataset description, experiment parameters and the data sampling strategies for manually creating gradual drifts and sudden drifts.

6.1.1 Experiment goals

The goal of the first two experiments is to explore DDEAL and DTM on text data, and to show that we can use DDEAL and DTM to detect and localize changes on text data. The third experiment aims at making sure DTM can actually extract the topics from documents both stably and reliably. In the first two experiments, we manually create corpus which contains topic shift by sampling documents from the different corpus. We perform such experiment both within one dataset (Ling-Spam email dataset) and between totally different datasets (email and twitter datasets). Besides, to test the stability and reliability of DTM, we randomly select small samples from the corpus for multiples and see whether the results generated from DTM are totally different from each other. Finally, after doing all these experiments, we would like to apply DTM to investigate the Rabobank email dataset and see if it can produce some nice insights of the emails.

6.1.2 Dataset description

Rabobank email dataset

The email dataset provided by Rabobank is a huge one which contains the emails from October of 2014 until March of 2017. Rabobank solicits phishing emails concerning Rabobank products and services to be sent to 'valse-email@'-mailbox. Such emails can be sent by customers as well as any other person. Being publicly known the mailbox also attracts other (unsolicited) ads, which are not relevant for this research. There are about one million emails in total. We are going to use a subset of it for the experiments, as it would take a long time to process them all. As for the content of the emails, for the purpose of our experiments, we only need the timestamp, subject and the main body of the emails. It is worth noting that most of the emails contain HTML data, which should also be taken into consideration when doing pre-processing on the dataset since all the words in HTML tags are meaningless.

Ling-Spam email dataset

The archive of Ling-Spam email dataset [4] contains two directories, spam/ and ham/, containing the spam and legitimate emails, respectively. The dataset contains 2412 ham emails and 481 spam emails, all of which were received by a mailing list on linguistics. Although it does not contain temporal information, we can define it manually. Each file in both directories is assigned a number as the id of the email which is indicated in the file name. i.e. the files in the spam/ directory contains *spam_1.txt* $\cdots$ *spam_481.txt*. We assume these files are named in time order in our experiments.

Stanford Twitter Sentiment (STS)

The Stanford Twitter sentiment corpus[1], introduced by Go et al. [21] consists of two different sets, training and test. The training set contains 1.6 million tweets automatically labeled as positive or negative based on emotions. The test set (STS-Test), on the other hand, is manually annotated and contains 177 negative, 182 positive and 139 neutral tweets. These tweets were collected by searching Twitter API with specific queries including names of products, companies, and people. Although the STS-Test dataset is relatively small, it has been widely used in the literature in different evaluation tasks. In our task we are not going to use sentiment label of this dataset, we use the training set as a corpus which contains time series text data. We use this dataset primarily because it is well-structured text dataset with temporal information.

6.1.3 Experiment parameters

The main pre-processing procedure is shown in Section 3.2.1, here we provide some parameters we set for the experiments. Firstly, in all the experiments, the words which have the frequency less than 25 after the pre-processing step have been removed from the corpus before applying DTM. Secondly, the parameter "number of topics" of DTM we choose for Rabobank data is 20 and for the rest is 5. Due to the limitation of time, we only use 20 for Rabobank data experiments. For the other two experiments, we tried several times and found the result of extracting 5 topics is the most representative one. Finally, the input of DTM is a sequence of collections of documents, so we need to define the time sequences, which contains the time slice definition. For the Rabobank data, we define each time sequence to be one week, so the input of DTM, in this case, would be a sequence of collections of emails, where each collection contains emails within a week. For the other experiments, since there is no temporal information in the Ling-Spam dataset, we define the time slices to be collections which contain the same number of documents.

6.1.4 Data sampling strategies for manually creating drifts

There are primarily two types of changes of the email topics that we care about: gradual change and sudden change. i.e., gradual changes in the topic would be scenarios when some words

[1] Available at http://help.sentiment140.com/

gradually become more (or less) important in the topic, whereas a sudden change would be when the importance of some words in the topic suddenly changed. In order to manually create such two types of changes in the corpus, assume we have two set of documents D_a and D_b which contain completely different topics, we have the following data sampling strategies:

1. **Gradual drift construction**: sample documents from D_a by using a periodic function $f(t)$ as shown in Equation 6.1 and sample the rest from D_b. We use W to denote the number of documents in each timestamp, and t to represent the timestamp. i.e., when $t = 1$ we sample W documents from D_a and 0 from D_b, when $t = 2$ we sample $2W/3$ from D_a and $W/3$ from D_b, etc. If the number of documents in one set is fewer than another, say D_a is fewer than D_b, we append the rest of D_b into the corpus when we run out of documents from D_a.

$$f(t) = W \cdot (-1)^{\lfloor \frac{t}{3} \rfloor} \cdot \left[1 - \frac{1}{3} \cdot \mod (t, 6)\right] \tag{6.1}$$

2. **Sudden drift construction**: append the documents directly one after another, i.e., append D_a to the end of D_b.

6.2 Manually create topic drifts within Ling-Spam

As described above, the Ling-Spam email dataset contains two types of emails, the spam emails, and the legitimate emails. The topics of these two sets of emails can be different from each other. In this experiment, we want to create a corpus which contains topic shift. So the basic idea is to sample emails from the Ling-Spam email dataset and shift from one set gradually or suddenly to another back and forth in order to create gradual drifts and sudden drifts.

There are 2412 ham emails and 481 spam emails in the dataset, for the sake of convenience, we take the first 2400 ham emails and the first 480 spam emails to create the corpus, so 2880 emails in total. The first line of email file is the subject of the email which we are not going to use for building DTM, and we only use the content of the emails. We separate the 2880 emails equally into 60 timestamps, so each timestamp contains 48 emails. We then use the strategies as shown in Section 6.1.4.

The first sampling strategy creates a corpus which contains gradual drifts: the corpus shift from ham to spam and then from spam to ham a couple of times. The second method creates a corpus with sudden drifts: the corpus changes suddenly from ham to spam. We try them both to test on DTM and analyze how the topics change accordingly. For both of the scenarios, we get 43977 unique terms out of the 2880 emails after the pre-processing step. Since it is a rather small corpus, we choose to filter out the words that have a frequency less than 3, and we end up with 16797 unique terms. We then build LSI with 5 components and DTM with 5 topics in this setting. We use the most frequent word in the largest component from LSI as the alternative label in DDEAL.

In the first scenario, DDEAL managed to trigger a drift at timestamp 55 for the largest component of LSI. In this case, the result is a little bit far from the actual drift timestamp 22. We then use can DTM to investigate the topics. Table 6.1 shows part of a topic from the result of DTM, in which the words that have interesting changing behaviors have been highlighted with different colors. In the first setting, the periodic function shown in Equation 6.1 has period of 6, and each period needs $0 + 16 + 32 + 48 + 32 + 16 = 144$ spam emails. We run out of spam emails at around timestamp 21, since we can separate 480 into the form $480 = 144 * 3 + 0 + 16 + 32$, where the number of timestamps is $6 * 3 + 1 + 1 + 1 = 21$.

As we can see in Table 6.1, the word “money” obviously has some periodical pattern, as the ranking of it gets back and forth over time, which is basically in the same pattern of the number of spam emails in each timestamp. Similarly, the ranking of the word “money” also changes back

and forth. Furthermore, when we run out of spam emails at timestamp 20, these two words disappear in our top 10 word list in the rest timestamps. In contrast, the word "peopl" becomes increasingly important in the topic since we ran out of the spam emails. Hence, from the result of this experiment setting, we can conclude that from the result of DTM, we can monitor the gradual changes of the topic over time.

Table 6.1: A topic from the result of gradual change.

time	word 1	word 2	word 3	word 4	word 5	word 6	word 7	word 8	word 9	word 10
1	mail	address	email	list	send	free	program	money	name	time
2	mail	address	email	list	send	free	program	money	report	name
3	mail	address	email	list	send	report	free	program	money	name
4	mail	address	email	report	list	program	send	free	name	money
5	mail	address	report	list	email	program	name	free	send	money
6	mail	report	address	program	list	name	email	send	money	free
7	mail	report	address	program	name	list	email	money	send	receiv
8	mail	report	address	program	email	name	list	money	send	receiv
9	mail	report	address	email	program	list	name	money	send	receiv
10	mail	report	3d	address	email	program	list	money	name	send
11	3d	mail	report	email	address	program	money	list	name	free
12	mail	3d	report	email	address	program	money	list	free	name
13	mail	report	3d	address	email	program	free	money	list	receiv
14	mail	report	address	email	program	free	3d	money	list	receiv
15	mail	report	address	email	free	program	money	receiv	link	list
16	mail	report	address	link	email	free	program	receiv	list	money
17	mail	report	address	email	program	free	receiv	link	list	inform
18	mail	report	address	email	program	list	receiv	free	send	inform
19	mail	report	address	email	program	list	send	receiv	peopl	time
20	mail	report	address	program	email	list	send	receiv	peopl	time
21	mail	report	address	program	email	list	send	peopl	receiv	time
22	mail	report	address	program	list	email	send	peopl	ani	time
23	mail	list	address	program	report	send	peopl	ani	email	time
24	mail	list	address	program	ani	peopl	send	report	time	email
25	mail	list	ani	program	peopl	address	send	time	report	pleas
26	list	mail	ani	peopl	program	address	send	pleas	time	inform
27	list	ani	mail	peopl	program	address	send	pleas	time	inform
28	list	ani	peopl	mail	program	pleas	send	address	time	inform
29	list	ani	peopl	mail	pleas	program	send	address	time	inform
30	list	ani	peopl	pleas	program	mail	send	inform	time	address

In the sudden change scenario, we run out of ham emails at around timestamp 50, since $2400 = 48 * 50$. The resulting topics in the first half part do not have too many fluctuations, as they are all from the ham email folder. In this case, DDEAL managed to trigger a drift at timestamp 49 for the largest component of LSI, which is pretty close to the actual change timestamp 51. In Table 6.2 we show the second half of a topic in the second scenario stating from timestamp 31. In the table, the words that have unusual changing behaviors have been highlighted with different colors.

As we can see in Table 6.2, the word "list" was the most important word in the topic until the spam emails were added into the corpus. The importance of it decreased to around the sixth in the topic. The word "email" and "mail", on the other hand, have suddenly become more and more important since the spam emails were added. These two words were not even in the top 10

before timestamp 38. Similarly, the word "free" and "money" are also new in the list from where the spam emails were added. From the result of this experiment setting, we can observe some sudden changes of the words in a topic over time.

Table 6.2: A topic from the result of sudden change.

time	word 1	word 2	word 3	word 4	word 5	word 6	word 7	word 8	word 9	word 10
31	list	inform	help	send	avail	pleas	ani	program	ha	messag
32	list	inform	send	help	avail	pleas	ani	program	ha	messag
33	list	inform	send	avail	help	pleas	program	ani	ha	messag
34	list	inform	send	pleas	avail	program	ani	help	ha	softwar
35	list	inform	send	program	pleas	avail	ani	ha	help	softwar
36	list	inform	send	program	pleas	ani	ha	avail	softwar	address
37	list	inform	program	send	pleas	ani	softwar	ha	address	avail
38	list	inform	program	send	softwar	ani	pleas	address	mail	ha
39	list	inform	program	send	softwar	address	ani	pleas	mail	ha
40	list	program	inform	address	send	softwar	mail	ani	pleas	ha
41	list	program	inform	address	mail	softwar	send	ani	pleas	ha
42	list	program	address	mail	inform	send	softwar	ani	pleas	file
43	list	program	address	mail	inform	send	ani	softwar	pleas	email
44	list	address	mail	program	inform	send	ani	email	softwar	pleas
45	list	mail	address	program	send	inform	email	ani	peopl	pleas
46	mail	list	address	program	email	send	inform	ani	peopl	pleas
47	mail	address	list	program	email	send	peopl	inform	ani	3d
48	mail	address	list	email	program	send	peopl	3d	inform	ani
49	mail	address	email	program	list	3d	send	peopl	free	inform
50	mail	address	email	program	3d	list	send	peopl	free	time
51	mail	email	address	program	3d	list	send	peopl	report	free
52	mail	email	address	program	report	list	send	peopl	3d	free
53	mail	address	email	report	program	list	send	free	3d	peopl
54	mail	address	report	email	list	free	program	send	receiv	money
55	mail	address	report	email	free	list	program	receiv	money	send
56	mail	report	address	email	free	list	program	money	receiv	name
57	mail	report	address	program	email	list	money	free	receiv	name
58	mail	report	address	program	money	email	list	receiv	name	send
59	mail	report	address	program	email	list	money	name	receiv	send
60	mail	report	address	program	email	name	list	money	receiv	send

To conclude, from the observations in the results of above two scenarios, we can see that the topic generated from DTM can actually change as a result of the changes in the corpus itself, both gradually and abruptly. And our novel approach DDEAL managed to trigger a drift in both two scenarios. However, these two experiments have the limitation that the two corpus sets are from the same dataset, both the spam and the ham emails were collected from the mailing list on linguistics. In next Section, we show the experiments on the corpus which combines document form different datasets.

6.3 Manually create topic drifts between Ling-Spam and STS

In this experiment, we perform the same procedure as described in the previous experiments, except we sample documents from different datasets. We use the first 2400 ham emails from Ling-Spam dataset and first 12000 tweets from the STS dataset. The STS dataset is not ordered in time. We first sort all the tweets in it by the creation time before doing any pre-processing. There are in total 14400 documents among these two datasets, we put 240 documents in every timestamp, and we then get 60 timestamps. With these settings, we can repeat the two experiments described above. In both of these two scenarios, DDEAL did not manage to detect any changes. However, we can still use DTM to investigate the topics over time.

In the first scenario, we use the same procedure as before with Equation 6.1, except the window size W is 240 instead of 48. i.e., when $t = 1$ we sample 240 tweets and 0 emails, when $t = 2$ we sample 160 tweets emails, etc. Since the number of tweets more than the number of emails, we append the result of ham email into the corpus when we run out of emails. Similar to the first setting in the previous experiment, we run out f emails at timestamp 21. Table 6.3 shows the first 50 timestamps of a topic in this setting, and we give the words that have interesting behaviors different colors.

As we can see in the table, the word "studi" and "book" have some periodical patterns during the first 21 timestamps and from 22, the ranking of them suddenly becomes higher and result in the top two words during about 15 timestamps from 29. "languag" and "linguist" on the other hand, suddenly become less important from 22 and 29 respectively. The behaviors of "studi" and "book" suggest a gradual periodical pattern in the dataset, and 'languag" and "linguist" show a sudden change in the dataset. Moreover, these patterns are exactly how we created this corpus.

In the second scenario, we simply append the twitter data to the end of the email data, and we run out of ham emails at around timestamp 10, since $2400 = 240 * 10$. In Table 6.4 we show the first 50 timestamps of a topic in the second setting. In the table, the words that have interesting changing behaviors have been highlighted with different colors.

As we can see from Table 6.4, the word "linguist" suddenly becomes less important after timestamp 12, and disappear from the top 10 words from 18. On the other hand, the words "page", "learn" and "write" suddenly become more and more important from timestamp 16. These behaviors all suggest a sudden change in the corpus.

In this experiment, we sample documents from totally different text datasets, and the results of DTM show the same pattern as of how we create the corpus (gradual change and sudden change). Although DDEAL did not manage to capture any changes, we can still localize the drift from the result of DTM. Based on this and the previous experiment, we can conclude that DDEAL together with DTM can be used to monitor the gradual change and sudden change of the emails.

Table 6.3: A topic from the result of the gradual change.

time	word 1	word 2	word 3	word 4	word 5	word 6	word 7	word 8	word 9	word 10
1	languag	linguist	univers	book	theori	studi	semant	syntax	structur	phonolog
2	languag	linguist	univers	book	theori	studi	semant	syntax	structur	phonolog
3	languag	linguist	univers	book	theori	semant	studi	syntax	structur	phonolog
4	languag	linguist	univers	book	theori	semant	structur	studi	syntax	phonolog
5	languag	linguist	univers	book	theori	semant	structur	studi	syntax	phonolog
6	languag	linguist	univers	book	theori	structur	semant	studi	phonolog	syntax
7	languag	linguist	univers	book	theori	structur	semant	studi	phonolog	analysi
8	languag	linguist	univers	theori	book	structur	semant	studi	phonolog	analysi
9	languag	linguist	univers	theori	book	structur	studi	semant	phonolog	pp
10	languag	linguist	univers	theori	book	structur	studi	pp	semant	phonolog
11	languag	linguist	univers	theori	book	pp	studi	structur	semant	phonolog
12	languag	linguist	univers	theori	pp	book	studi	structur	semant	phonolog
13	languag	linguist	univers	theori	pp	studi	book	structur	semant	phonolog
14	languag	linguist	univers	theori	studi	book	pp	structur	semant	phonolog
15	languag	linguist	univers	theori	studi	book	structur	semant	pp	phonolog
16	languag	linguist	univers	theori	book	studi	structur	semant	grammar	phonolog
17	languag	linguist	univers	theori	book	studi	structur	grammar	semant	inform
18	languag	linguist	univers	theori	book	studi	structur	inform	grammar	discours
19	languag	linguist	univers	theori	book	studi	structur	discours	inform	semant
20	languag	linguist	univers	theori	book	discours	studi	structur	inform	semant
21	languag	linguist	univers	discours	book	theori	studi	structur	inform	semant
22	linguist	languag	univers	discours	book	studi	theori	structur	inform	semant
23	linguist	languag	univers	studi	book	discours	theori	structur	inform	semant
24	linguist	languag	studi	book	discours	univers	theori	structur	inform	semant
25	linguist	languag	studi	book	discours	univers	theori	structur	inform	paper
26	linguist	languag	studi	book	discours	theori	univers	structur	inform	paper
27	linguist	studi	book	languag	discours	theori	univers	structur	paper	inform
28	linguist	studi	book	languag	discours	theori	univers	structur	paper	inform
29	studi	book	linguist	discours	languag	theori	paper	univers	structur	inform
30	studi	book	linguist	discours	theori	languag	paper	structur	univers	inform
31	studi	book	linguist	discours	theori	paper	structur	languag	text	grammar
32	studi	book	linguist	discours	paper	theori	text	structur	page	grammar
33	studi	book	linguist	discours	paper	theori	text	page	grammar	structur
34	studi	book	linguist	discours	paper	text	page	theori	edit	grammar
35	studi	book	discours	paper	linguist	page	text	edit	theori	featur
36	studi	book	paper	page	text	discours	linguist	edit	featur	theori
37	studi	book	page	paper	text	discours	edit	linguist	featur	grammar
38	studi	book	page	text	paper	edit	discours	featur	linguist	ha
39	studi	book	page	text	paper	edit	discours	featur	ha	grammar
40	studi	book	page	text	paper	edit	discours	featur	ha	chang
41	studi	book	page	text	paper	edit	discours	featur	ha	chang
42	studi	book	page	text	edit	paper	discours	ha	featur	chang
43	studi	book	page	text	edit	paper	ha	featur	discours	chang
44	studi	book	page	text	edit	paper	ha	featur	discours	chang
45	studi	book	page	text	edit	paper	ha	featur	chang	discours
46	studi	book	page	text	edit	paper	ha	featur	chang	discours
47	studi	page	book	text	edit	paper	ha	featur	chang	www
48	studi	page	book	text	edit	paper	ha	featur	chang	www
49	studi	page	book	text	edit	paper	ha	featur	chang	www
50	studi	page	book	text	edit	paper	ha	featur	chang	www

Table 6.4: A topic from the result of the sudden change.

time	word 1	word 2	word 3	word 4	word 5	word 6	word 7	word 8	word 9	word 10
1	languag	linguist	univers	studi	book	theori	english	inform	develop	research
2	languag	linguist	univers	studi	book	theori	english	inform	develop	research
3	languag	linguist	univers	studi	book	english	theori	inform	research	develop
4	languag	linguist	univers	studi	book	english	theori	inform	research	discours
5	languag	linguist	univers	studi	book	english	theori	research	inform	discours
6	languag	linguist	univers	studi	book	english	theori	pp	discours	research
7	languag	linguist	univers	studi	book	theori	english	pp	discours	research
8	languag	linguist	univers	book	studi	theori	english	pp	discours	paper
9	languag	linguist	univers	theori	studi	book	english	discours	paper	research
10	languag	linguist	univers	theori	book	studi	english	paper	discours	research
11	languag	linguist	univers	book	theori	studi	english	paper	research	discours
12	languag	linguist	univers	book	theori	studi	english	research	paper	discours
13	languag	univers	linguist	book	studi	theori	english	research	paper	phonolog
14	languag	univers	book	linguist	studi	theori	english	research	paper	develop
15	languag	book	univers	studi	linguist	theori	english	research	paper	develop
16	languag	book	univers	studi	theori	english	linguist	research	paper	page
17	languag	book	univers	studi	theori	english	research	linguist	page	paper
18	languag	book	univers	studi	theori	english	page	research	paper	develop
19	languag	book	univers	studi	theori	english	page	research	paper	learn
20	languag	book	univers	studi	theori	page	english	research	learn	paper
21	book	languag	univers	studi	theori	page	english	research	learn	write
22	book	languag	univers	studi	page	theori	english	research	learn	write
23	book	languag	univers	studi	page	theori	english	learn	write	research
24	book	languag	univers	studi	page	theori	english	learn	write	research
25	book	languag	univers	studi	page	theori	english	write	learn	research
26	book	languag	univers	studi	page	write	learn	theori	english	research
27	book	languag	univers	page	studi	write	learn	english	theori	research
28	book	languag	univers	page	studi	write	learn	english	theori	research
29	book	languag	page	univers	studi	write	learn	english	theori	research
30	book	languag	page	univers	studi	write	learn	english	theori	research
31	book	languag	page	univers	studi	write	learn	english	text	research
32	book	page	languag	univers	write	studi	learn	text	english	research
33	book	page	languag	write	univers	studi	learn	text	english	research
34	book	page	write	languag	univers	studi	learn	text	english	research
35	book	page	write	languag	studi	univers	learn	text	research	english
36	book	page	write	studi	languag	univers	learn	text	current	research
37	book	page	write	studi	univers	languag	learn	text	current	research
38	book	page	write	studi	learn	univers	languag	text	current	research
39	book	page	write	studi	learn	univers	languag	text	current	research
40	book	page	write	studi	learn	univers	languag	text	current	research
41	book	page	write	studi	learn	univers	languag	text	current	research
42	book	page	write	studi	learn	univers	languag	text	current	research
43	book	page	write	studi	learn	univers	languag	text	current	research
44	book	page	write	studi	learn	univers	text	languag	current	pleas
45	book	page	write	studi	learn	univers	text	languag	current	pleas
46	book	page	write	studi	learn	univers	text	languag	current	pleas
47	book	page	write	learn	studi	univers	text	languag	current	pleas
48	book	page	write	learn	studi	univers	text	languag	current	pleas
49	book	page	write	learn	studi	univers	text	languag	current	pleas
50	book	page	write	learn	studi	univers	text	languag	current	pleas

6.4 Experiment for stability and reliability of DTM

We have shown in previous experiments that DTM can capture different kinds of changes on the dataset. However, there is still another question to ask, is the result of DTM reliable? We want the results of it to be stable, and the generated topics are actually the main subject that the emails were talking about. So we need to test the stability and reliability of DTM with real-world data.

In many works of literature, i.e., [22] and [28], researchers used different measures to compute how different are the results from different runs by using one topic model. However, in our case, we want to show the topics from DTM are actually representative words within the documents rather than some random words. Thus, we design to run the model multiple times with different small samples from the same corpus and see whether most of the words in the results remain the same and whether the representative words are always in the topics.

Table 6.5: The union set of the 5 topics from each run.

run	common words of the 5 topics
1	abstract, address, ani, au, benjamin, book, comput, confer, der, discours, doe, du, edu, en, english, et, exampl, fax, featur, grammar, gruyter, ha, hi, http, human, includ, inform, languag, le, linguist, mail, mean, mouton, onli, paper, pari, peopl, phonolog, pleas, pp, refer, research, semant, session, speech, structur, studi, submiss, syntax, system, text, theori, und, univers, universit, verb, word, workshop
2	abstract, address, am, ani, author, benjamin, book, comput, confer, depart, der, discours, doe, edu, english, et, exampl, fax, featur, ha, hi, http, includ, inform, languag, le, lexic, linguist, logic, mail, mean, onli, paper, peopl, phonolog, pleas, pp, press, publish, question, research, resourc, scienc, semant, session, speech, studi, submiss, system, theori, univers, usa, verb, volum, word, workshop, www
3	abstract, address, am, ani, benjamin, book, canada, comput, confer, del, depart, discours, doe, du, edu, en, english, et, evalu, exampl, fax, featur, franc, grammar, ha, hi, http, human, includ, inform, languag, le, linguist, mail, mean, onli, paper, pari, peopl, phonolog, pleas, pp, registr, research, resourc, semant, session, spanish, speech, structur, studi, submiss, syntax, system, text, theori, translat, univers, universit, verb, word, workshop
4	abstract, address, ani, applic, benjamin, book, comput, confer, cours, depart, der, develop, discours, doe, dr, du, edu, en, english, et, exampl, featur, gener, ha, hi, http, inform, languag, le, linguist, mail, mean, onli, paper, peopl, pleas, posit, pp, press, program, public, publish, question, refer, registr, research, session, speech, studi, submiss, system, teach, theori, und, univers, universit, verb, volum, word, workshop
5	abstract, address, ani, book, break, coffe, comput, confer, cours, depart, discours, edu, educ, en, english, et, exampl, fax, featur, gener, georgetown, grammar, ha, hi, http, includ, inform, korean, languag, le, linguist, list, lunch, mail, mean, onli, paper, phonolog, pleas, pp, question, refer, registr, research, session, spanish, student, studi, submiss, syntax, system, teach, theori, univers, verb, word, workshop
intersection	abstract, address, ani, book, comput, confer, discours, edu, english, et, exampl, featur, ha, hi, http, inform, languag, le, linguist, mail, mean, onli, paper, pleas, pp, research, session, studi, submiss, system, theori, univers, verb, word, workshop

We use the first 2400 ham emails from Ling-Spam dataset, each time randomly sample 40% emails and separate them into 40 equal timestamps, each of which contains 24 emails. We build a DTM with 5 topics in this setting and run it for 5 times, then compare the results. After the pre-processing procedure, there are in total 40866 unique words in it, and we then remove the words that appear less than 3 times, which is same as the previous experiments. The data ends up with 15586 unique words.

We test the stability of DTM with respect to small samples from the data and see whether the common words that appeared in all the 5 topics change in each run. As for the reliability, we want DTM to create some meaningful topics rather than randomly picking up words form the documents. Since we already know these emails are collected from a linguistics mailing list, then words like "linguistic", "language" and "research" should definitely show up in one or more topics. We traverse over the words in the 5 topics and extract the words that appeared in all the 5 topics.

In each run we have a set of 5 topics $\{topic_1, \ldots, topic_5\}$ and we compute the union of the top 10 words appeared in each topic, since these words are the most common words that make up these 5 topics and the top 10 words do not change too much compared to top 20 or top 30. As a result, the number of words in the union set of each run is 58, 57, 62, 60 and 57. Although most of them are different from each other, the number does not fluctuate too much. Thus, the average number of common words in the topics is 58.8. After that, we compute the intersection set among these 5 union sets.

Table 6.5 shows the resulting union set of each run and the intersection of these 5 sets. There are 35 words in the intersection set, which means that these 5 runs have 35 words in common which is 60% of the average value 58.8. We may conclude that the DTM is not generating words randomly. When looking at the words in the intersection set, words like "languag", "linguist" and "research" are in there, which confirms our assumption. Besides, most of the other words in the set are quite related to the overall topic "linguistics research". Hence, DTM meets our requirement for stability and reliability.

6.5 Exploration on Rabobank email dataset

The Rabobank email dataset contains a huge amount of emails, and we cannot process them all at a time. We choose to analyze emails from first six months of 2015. There are in total 265,526 emails. We choose one week as one timestamp in the topic model, so there are in total 26 timestamps. After pre-processing step, we get 238,643 unique words, we then remove all the words that appear less than 25 times, and we end up with a vocabulary of 12,139 words. We then use DTM to build 20 topics over this corpus on a 2.67GHZ Intel(R) Xeon(R) CPU Windows 10 server. Posterior inference took approximately 9 hours.

Table 6.6 shows a topic from the results. All the words are anonymized by using the initial letter plus a number. The words that have interesting behaviors are highlighted by different colors. The word "t10" at the beginning is the most important word in this topic, but as time goes by, it becomes less important and finally gets important again. "m8" on the other hand, is the 8th important word in the topic at the beginning and the ranking of it increased to the 3rd and then dropped continuously to the 10th. "r11" continuously becomes more important at the beginning and dropped to the 8th at the end. These interesting behaviors can reveal the change and the evolvement of the topics over time.

The results were reported to Rabobank, and they found it interesting to see the changing behaviors of the topics over time. The results together are shown with the visualization tool in Chapter 3 provide more insights into the Rabobank email dataset and help Rabobank to understand more about the phishing emails.

Table 6.6: A topic from the result of the Rabobank email data.

time	word 1	word 2	word 3	word 4	word 5	word 6	word 7	word 8	word 9	word 10
1	t10	a7	o3	o1	g1	r11	v27	m8	b1	b7
2	t10	a7	o3	o1	g1	r11	v27	m8	b1	b7
3	t10	a7	o3	o1	g1	r11	v27	m8	b1	v26
4	t10	a7	o1	o3	g1	r11	b1	v27	m8	v26
5	t10	a7	o1	o3	g1	r11	b1	v27	l2	m8
6	t10	a7	o3	o1	g1	r11	b1	m8	l2	v27
7	t10	o3	a7	g1	r11	o1	m8	b1	m20	g10
8	t10	o3	r11	g1	a7	m8	m20	g10	b1	b7
9	o3	r11	t10	g1	m8	w1	a7	m20	b7	g10
10	o3	r11	g1	m8	t10	w1	b7	b1	g10	m20
11	r11	o3	m8	w1	g1	t10	b7	b1	o10	w4
12	r11	o3	m8	w1	g1	o10	b7	t10	w4	b1
13	r11	o3	w1	m8	g1	o10	w4	b7	b1	m1
14	r11	o3	w1	m8	w4	m1	o10	g1	b7	v12
15	r11	w1	o3	m8	w4	m1	v12	b7	o10	p5
16	r11	o3	w1	m1	m8	w4	v12	b7	o10	g1
17	r11	o3	b7	w1	m8	g1	m1	o10	t10	g10
18	r11	o3	b7	g1	t10	g10	w1	m8	k7	o10
19	r11	t10	o3	b7	g1	a7	g10	k7	b35	m8
20	a7	t10	r11	g1	o3	b7	g10	k7	h9	b35
21	a7	t10	o3	g1	r11	b7	g10	k7	b35	h9
22	a7	t10	o3	g1	g10	r11	k7	b7	b35	o1
23	a7	t10	o3	g1	o1	g10	k7	r11	b35	b7
24	a7	t10	o3	o1	g1	g10	k7	r11	b35	d9
25	a7	t10	o3	o1	g1	n14	g10	n15	k7	d9
26	a7	n14	t10	n15	o1	o3	g1	g10	k7	d9

6.6 Conclusion of experiments on DTM

From the above experiment results, we can see that the topic generated from DTM can actually change as a result of the changes in the corpus itself, both in a gradual manner and in an abrupt manner. We showed this by sampling documents both from the same dataset and from totally different datasets. We can conclude that DTM can be used to monitor the gradual change and sudden change of the emails. Besides, the experiments also show that the results from DTM are stable with respect to small samples from the corpus and it can generate reliable results. So the conclusion is we can use DTM to monitor the topic changes over time in Rabobank emails.

Chapter 7

Conclusions

In this thesis, we developed concept drift investigation tool for high dimensional streams including multivariate numeric data (i.e., transactional data) and text data with temporal information (i.e., email data). The research problem is formulated to develop a concept drift investigator with specific requirements (i.e., high dimensional, labels are not instantly available), and to develop a visualization tool which can help domain expert to localize the drift and is easy to interact with. We use different methods to detect drifts in numeric data and text data.

In the following sections, we first conclude our contributions in terms of academic and business values. Then we discuss the limitations and future work.

7.1 Contributions

In this section, firstly we conclude our academic contributions. Then we conclude our contributions in terms of business values.

7.1.1 Academic contributions

In this thesis, firstly we formulated the research problem of concept drift and showed different types of drifts based on both effects of the changes and how the drifts happen. We reviewed the existing drift detection methods from the four categories: methods monitoring distributions of two different time windows, detectors based on sequential analysis, detectors based on Statistical Process Control and contextual approaches. Due to the fact that without access to the class labels, some of the real drifts cannot be detected. By addressing this fact, we then proposed a novel drift detection ensemble based on alternative label (DDEAL) using weighted majority vote.

The novel approach can be divided into three parts (procedures): the inner incremental learner, the ensemble of detectors and the feature selection procedure. We first proposed different ways to get the alternative label from the dataset, then we discussed the requirements of the inner incremental learner and gave some existing approaches. Finally, we showed different decision strategies for the ensemble and designed a new way to construct the ensemble.

We also validated the performance of DDEAL in some experiments by comparing with individual detectors, different ensemble strategies, and a competitor detection approach. Furthermore, we tested the limitation of this approach and we concluded the main limitation of the proposed method is when the drift is only on the class label without the change in the features. In this case, we can not find an alternative label which can represent the true class label well and our method can perform worse than others.

For text dataset, we applied DDEAL on the result from Latent Semantic Indexing and we used Dynamic Topic Model to analyze the time evolution of topics in a document collection. We showed that DDEAL can be used to detect changes in text data. Moreover, we validated DTM's function of showing changes in the topics as well as its stability and reliability on the manually created corpus from real-world dataset.

Besides, we built an interactive application to visualize the drift detection results which can be used to localize the drift points in the data further and to aid in understanding concept drift. We used line chart and scatter plot to visualize the features and highlighted the drift locations in them. We also used the pie chart to show the distribution of the values within a selected time window. Apart from the data visualization, we also provided the visualization of machine learning models. It provides flexibility for users to choose different features from different time window to compare the models. Besides, for text data, we created an interactive visualization of the results from Dynamic Topic Model.

7.1.2 Business contributions

We made the following contributions in terms of business values for Rabobank who provided the real and anonymized transaction dataset and the email dataset:

- We built a visualization application to provide the visual investigation of the drift detection results. The visualization be used directly in Rabobank to help domain experts to localize the drift points and provide more insights of the data by the model visualizations. The bank can then take further actions to adapt their fraud detection models to the detected drifts.
- We applied our novel approach as well as some existing drift detection approaches on the Rabobank transaction dataset and used our application to visualize the detection results. Besides, DDEAL also provides the flexibility for domain experts to choose the alternative label with their experiences. The detection results helped Rabobank to find the features that have drifted in the dataset.
- We applied Dynamic Topic Model on the email dataset provided by Rabobank and visualized the results by our application. The results helped Rabobank to investigate into their huge phishing email dataset and provided a clear view of what were people talking about on in the emails.

7.2 Limitations and Future Work

The limitations and future work of this thesis are as follows:

- We compared our novel method with only one drift detection competitor and only on one real-world high dimensional dataset (Rabobank transaction data). Although there are a few methods from related work, due to the limitation of time we did not compare with all of them. For future work, we can perform more experiments to compare our novel approach with more other related works.
- In the concept drift investigation framework, we only applied a few machine learning algorithms to investigate further into the data, and we only applied ADWIN to detect changes in each feature. For future work, we can apply more alternative methods to add more functions to the application.
- We only used Dynamic Topic Model to capture the change of topics over time, since it is the most widely used one for analyzing the time evolution of topics in a document collection. However, there are a few alternative methods. For future work, we can add more alternative models to extract topics and then visualize the results for the text data in the application framework.

- We only tried one feature selection method to get the alternative label in our novel approach. We proposed several ways to select the feature to be the alternative label, however, we did not do experiments to evaluate each of them. Besides, the feature can also be selected from domain expert's experiences. For future work, we can design some experiments with the help of domain experts to evaluate each of them.

Bibliography

[1] Zahraa S Abdallah, Mohamed Medhat Gaber, Bala Srinivasan, and Shonali Krishnaswamy. Anynovel: detection of novel concepts in evolving data streams. *Evolving Systems*, 7(2):73–93, 2016. 36

[2] Amr Ahmed and Eric P Xing. Timeline: A dynamic hierarchical dirichlet process model for recovering birth/death and evolution of topics in text stream. *arXiv preprint arXiv:1203.3463*, 2012. 12

[3] Tahseen Al-Khateeb, Mohammad M Masud, Khaled M Al-Naami, Sadi Evren Seker, Ahmad M Mustafa, Latifur Khan, Zouheir Trabelsi, Charu Aggarwal, and Jiawei Han. Recurring and novel class detection using class-based ensemble for evolving data stream. *IEEE Transactions on Knowledge and Data Engineering*, 28(10):2752–2764, 2016. 36

[4] Ion Androutsopoulos, John Koutsias, Konstantinos V Chandrinos, George Paliouras, and Constantine D Spyropoulos. An evaluation of naive bayesian anti-spam filtering. *arXiv preprint cs/0006013*, 2000. 45

[5] Manuel Baena-García, José del Campo-Ávila, Raúl Fidalgo, Albert Bifet, Ricard Gavaldà, and Rafael Morales-Bueno. Early drift detection method. 2006. 10

[6] Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM international conference on data mining*, pages 443–448. SIAM, 2007. 10, 18

[7] Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Richard Kirkby, and Ricard Gavaldà. New ensemble methods for evolving data streams. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 139–148. ACM, 2009. 39

[8] Albert Bifet, Geoffrey Holmes, Bernhard Pfahringer, and Ricard Gavalda. Detecting sentiment change in twitter streaming data. 2011. 12

[9] David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120. ACM, 2006. 12, 18, 24

[10] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003. 12

[11] RP Jagadeesh Chandra Bose, Wil MP van der Aalst, Indrė Žliobaitė, and Mykola Pechenizkiy. Handling concept drift in process mining. In *International Conference on Advanced Information Systems Engineering*, pages 391–405. Springer, 2011. 36

[12] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007. 11

[13] Magdalena Deckert. Batch weighted ensemble for mining data streams with concept drift. In *International Symposium on Methodologies for Intelligent Systems*, pages 290–299. Springer, 2011. 35

[14] Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 71–80. ACM, 2000. 31

[15] Denis Moreira dos Reis, Peter Flach, Stan Matwin, and Gustavo Batista. Fast unsupervised online drift detection using incremental kolmogorov-smirnov test. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1545–1554. ACM, 2016. 12

[16] Lei Du, Qinbao Song, Lei Zhu, and Xiaoyan Zhu. A selective detector ensemble for concept drift detection. *The Computer Journal*, 58(3):457–471, 2014. 32, 33, 36

[17] Isvani Frías-Blanco, José del Campo-Ávila, Gonzalo Ramos-Jiménez, Rafael Morales-Bueno, Agustín Ortiz-Díaz, and Yailé Caballero-Mota. Online and non-parametric drift detection methods based on hoeffdings bounds. *IEEE Transactions on Knowledge and Data Engineering*, 27(3):810–823, 2015. 10, 39

[18] Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In *Brazilian symposium on artificial intelligence*, pages 286–295. Springer, 2004. 10, 39

[19] João Gama, Ricardo Rocha, and Pedro Medas. Accurate decision trees for mining high-speed data streams. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 523–528. ACM, 2003. 12

[20] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):44, 2014. 9

[21] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12), 2009. 45

[22] Derek Greene, Derek OCallaghan, and Pádraig Cunningham. How many topics? stability analysis for topic models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 498–513. Springer, 2014. 52

[23] Michael Harries and New South Wales. Splice-2 comparative evaluation: Electricity pricing. 1999. 39

[24] Susan Havre, Elizabeth Hetzler, Paul Whitney, and Lucy Nowell. Themeriver: Visualizing thematic changes in large document collections. *IEEE transactions on visualization and computer graphics*, 8(1):9–20, 2002. 12

[25] Mohammad Javad Hosseini, Ameneh Gholipour, and Hamid Beigy. An ensemble of cluster-based classifiers for semi-supervised classification of non-stationary data streams. *Knowledge and Information Systems*, 46(3):567–597, 2016. 36

[26] Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 180–191. VLDB Endowment, 2004. 12

[27] Dan Knights, Michael C Mozer, and Nicolas Nicolov. Detecting topic drift with compound topic models. In *ICWSM*, 2009. 12

[28] Sergei Koltcov, Sergey I Nikolenko, Olessia Koltsova, Vladimir Filippov, and Svetlana Bodrunova. Stable topic modeling with local density regularization. In *International Conference on Internet Science*, pages 176–188. Springer, 2016. 52

[29] Ludmila I Kuncheva and William J Faithfull. Pca feature extraction for change detection in multidimensional unlabeled data. *IEEE transactions on neural networks and learning systems*, 25(1):69–80, 2014. 11

[30] Bruno Iran Ferreira Maciel, Silas Garrido Teixeira Carvalho Santos, and Roberto Souto Maior Barros. A lightweight concept drift detection ensemble. In *Tools with Artificial Intelligence (ICTAI), 2015 IEEE 27th International Conference on*, pages 1061–1068. IEEE, 2015. 35

[31] Mohammad M Masud, Clay Woolam, Jing Gao, Latifur Khan, Jiawei Han, Kevin W Hamlen, and Nikunj C Oza. Facing the reality of data stream classification: coping with scarcity of labeled data. *Knowledge and information systems*, 33(1):213–244, 2012. 36

[32] Leandro L Minku and Xin Yao. Ddd: A new ensemble approach for dealing with concept drift. *IEEE transactions on knowledge and data engineering*, 24(4):619–633, 2012. 35

[33] Hoang-Vu Nguyen and Jilles Vreeken. Linear-time detection of non-linear changes in massively high dimensional time series. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 828–836. SIAM, 2016. 12, 40

[34] Kyosuke Nishida. Learning and detecting concept drift. *Information Science and Technology*, 2008. 35

[35] Kyosuke Nishida and Koichiro Yamauchi. Detecting concept drift using statistical testing. In *International conference on discovery science*, pages 264–269. Springer, 2007. 39

[36] Ewan S Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954. 10

[37] Russel Pears, Sripirakas Sakthithasan, and Yun Sing Koh. Detecting concept change in dynamic data streams. *Machine Learning*, 97(3):259–293, 2014. 10

[38] Ali Pesaranghader, Herna Viktor, and Eric Paquet. Reservoir of diverse adaptive learners and stacking fast hoeffding drift detection methods for evolving data streams. *arXiv preprint arXiv:1709.02457*, 2017. 10, 34, 37

[39] Ali Pesaranghader and Herna L Viktor. Fast hoeffding drift detection method for evolving data streams. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 96–111. Springer, 2016. 10

[40] Kevin B Pratt and Gleb Tschapek. Visualizing concept drift. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 735–740. ACM, 2003. 13

[41] Abdulhakim A Qahtan, Basma Alharbi, Suojin Wang, and Xiangliang Zhang. A pca-based change detection framework for multidimensional data streams: Change detection in multidimensional data streams. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944. ACM, 2015. 12

[42] Gordon J Ross, Niall M Adams, Dimitris K Tasoulis, and David J Hand. Exponentially weighted moving average charts for detecting concept drift. *Pattern recognition letters*, 33(2):191–198, 2012. 10

[43] Eyal Sagi, Stefan Kaufmann, and Brady Clark. Tracing semantic change with latent semantic analysis. *Current methods in historical semantics*, pages 161–183, 2011. 12

[44] Carson Sievert and Kenneth Shirley. Ldavis: A method for visualizing and interpreting topics. In *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, pages 63–70, 2014. 24

[45] Piotr Sobolewski and Michał Woźniak. Comparable study of statistical tests for virtual concept drift detection. In *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013*, pages 329–337. Springer, 2013. 12

[46] Eduardo J Spinosa, André Ponce de Leon F de Carvalho, and João Gama. Olindda: A cluster-based approach for detecting novelty and concept drift in data streams. In *Proceedings of the 2007 ACM symposium on Applied computing*, pages 448–452. ACM, 2007. 36

[47] Abraham Wald. *Sequential analysis*. Courier Corporation, 1973. 9

[48] Geoffrey I Webb, Roy Hyde, Hong Cao, Hai Long Nguyen, and Francois Petitjean. Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994, 2016. 8

[49] Michał Woźniak, Paweł Ksieniewicz, Bogusław Cyganek, and Krzysztof Walkowiak. Ensembles of heterogeneous concept drift detectors-experimental study. In *IFIP International Conference on Computer Information Systems and Industrial Management*, pages 538–549. Springer, 2016. 32, 36, 40

[50] Indre Žliobaite. Change with delayed labeling: When is it detectable? In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 843–850. IEEE, 2010. 12

[51] Indrė Žliobaitė, Mykola Pechenizkiy, and Joao Gama. An overview of concept drift applications. In *Big Data Analysis: New Algorithms for a New Society*, pages 91–114. Springer, 2016. 6, 11

Appendix A

Appendix

A.1 Visualizations of detection results on different pre-processing data.

This section shows the visualization results we found by using our drift detection investigation framework on the Rabobank transaction dataset. The listed results are the ones that can be visually confirmed to have drifted. For the results of third pre-processing strategy, we only list three of the features to give an idea what they look like. In every figure of this section, the left part and the right part are showing the same data. The left chart shows the data with scatter plot whereas the right one shows a line chart. The red line in each chart is the time when ADWIN detects the feature to have drifted (the drift location where ADWIN reaches warning level).

A.1.1 Features that have visual drifts from pre-processing 1

Here we show the visualization of detection results of features N_142, N_135, C_196, N_152, C_046, N_114, C_337, C_054, N_000, C_269, C_082, N_147 and N_062 which have some visual drifts by using the first pre-processing strategy. In all the visualizations, we can see some clear changes with or without the highlight generated by ADWIN detection results.

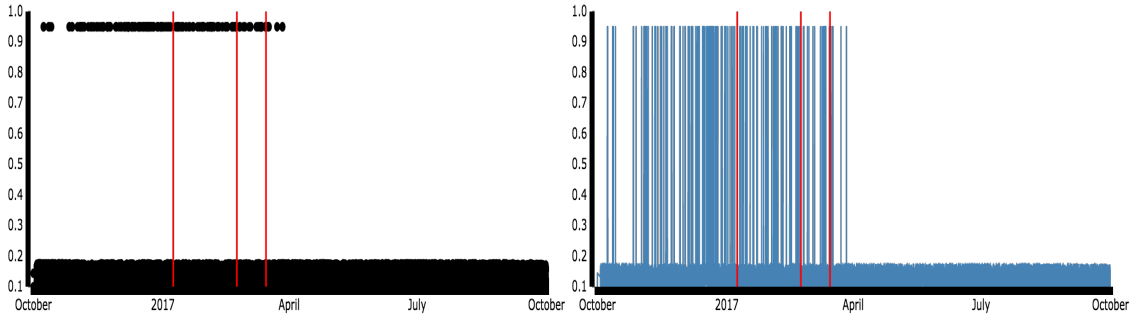

Figure A.1: N_142

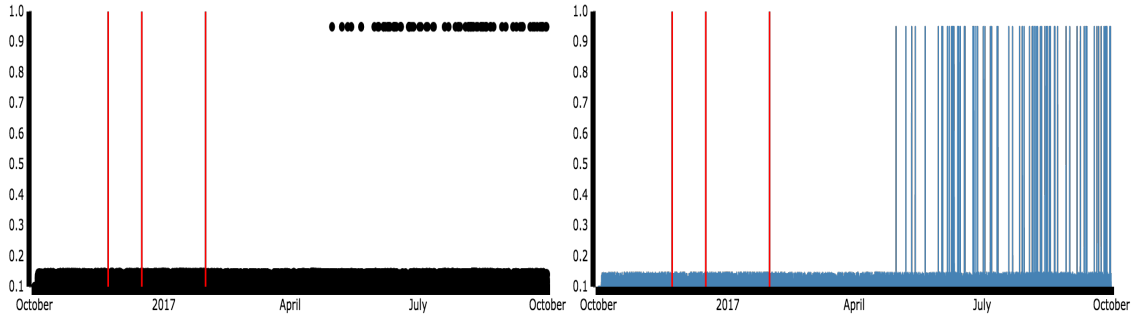

Figure A.2: N_135

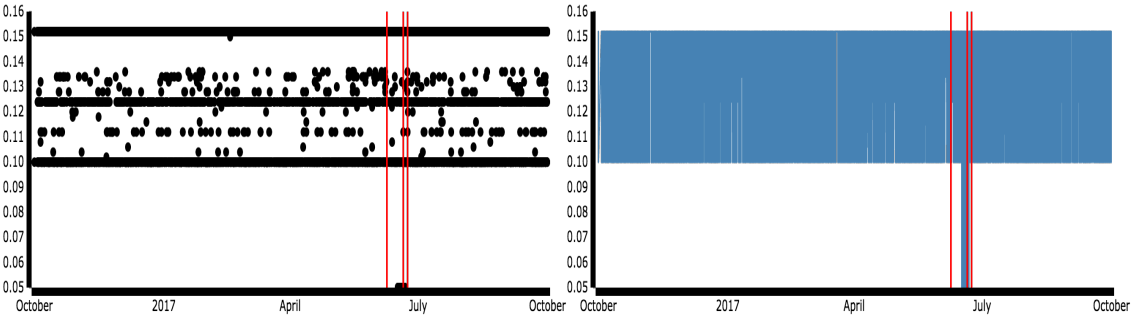

Figure A.3: C_196

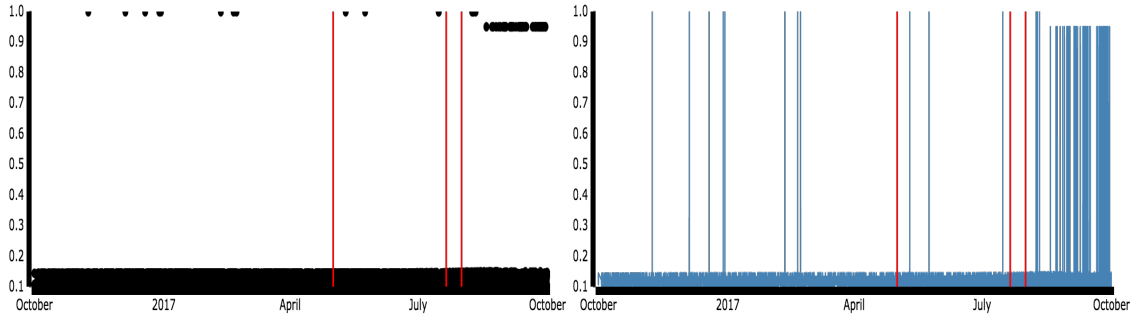

Figure A.4: N_152

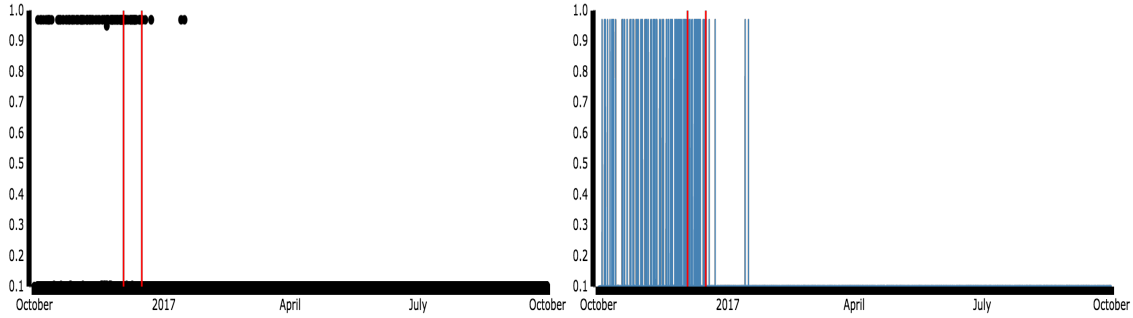

Figure A.5: C_046

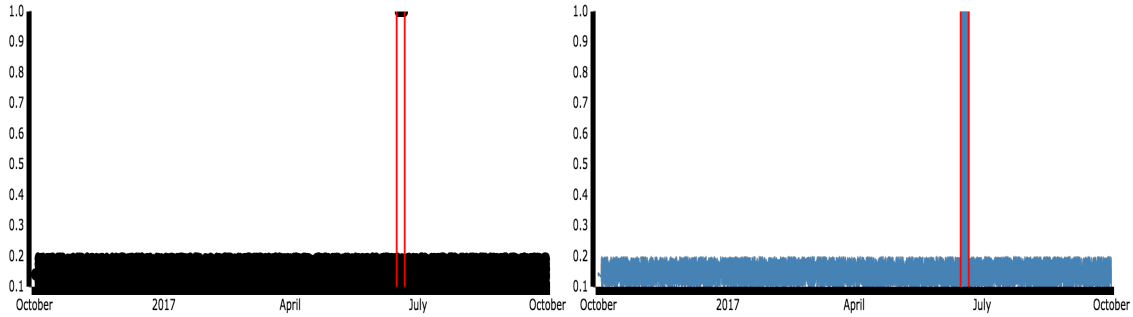

Figure A.6: N_114

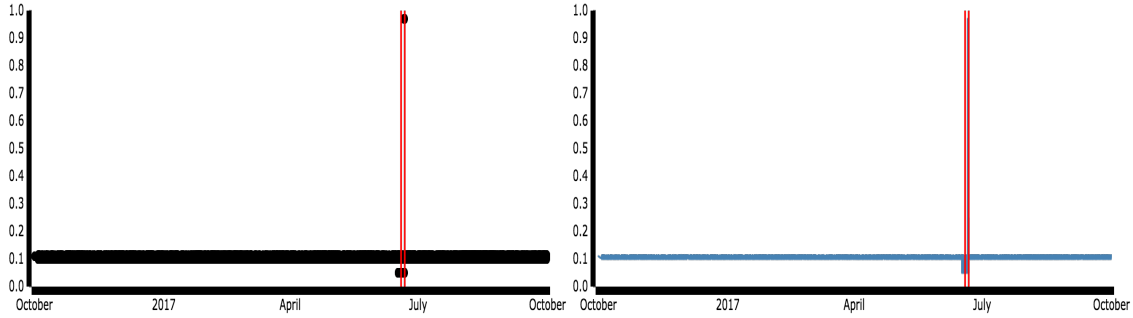

Figure A.7: C_337

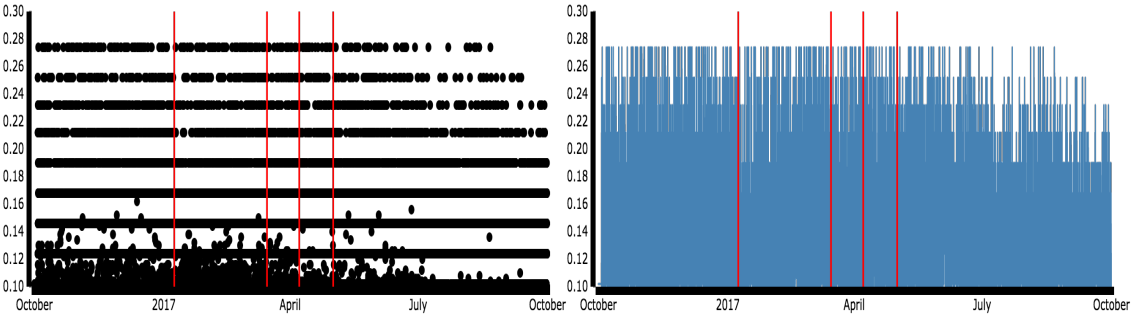

Figure A.8: C_054

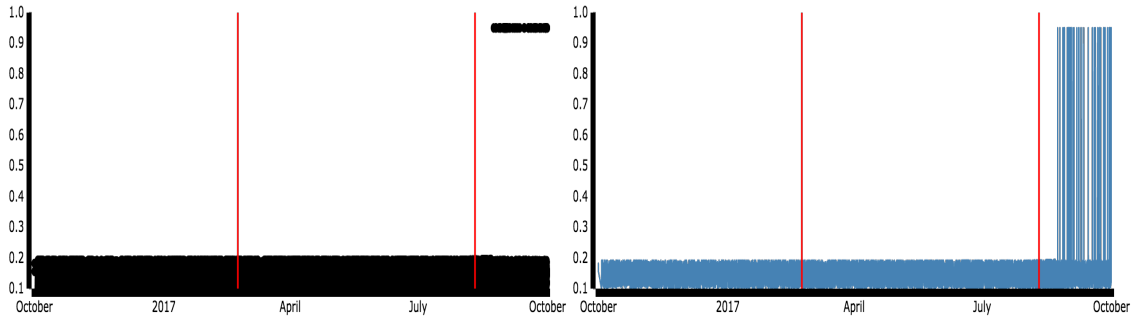

Figure A.9: N_000

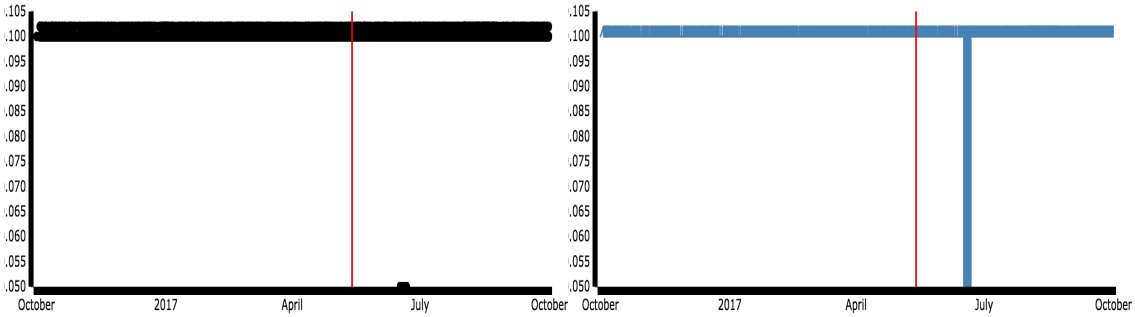

Figure A.10: C_269

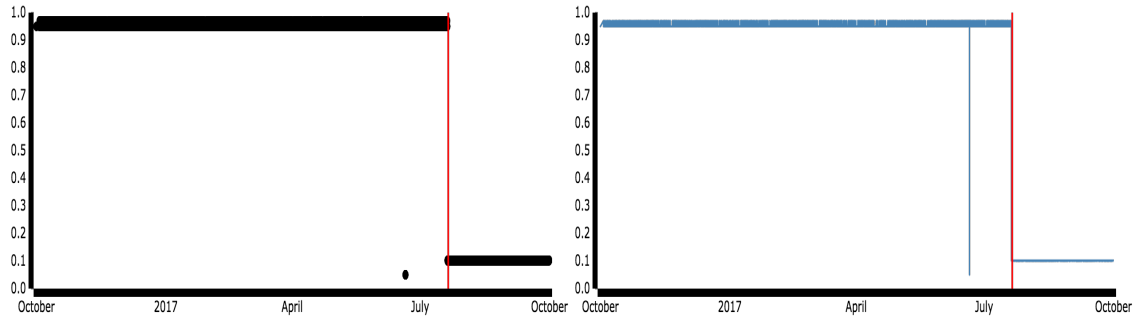

Figure A.11: C_082

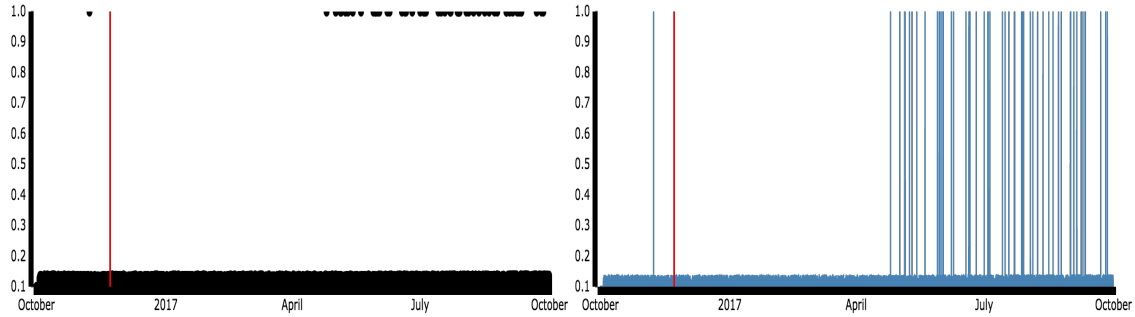

Figure A.12: N_147

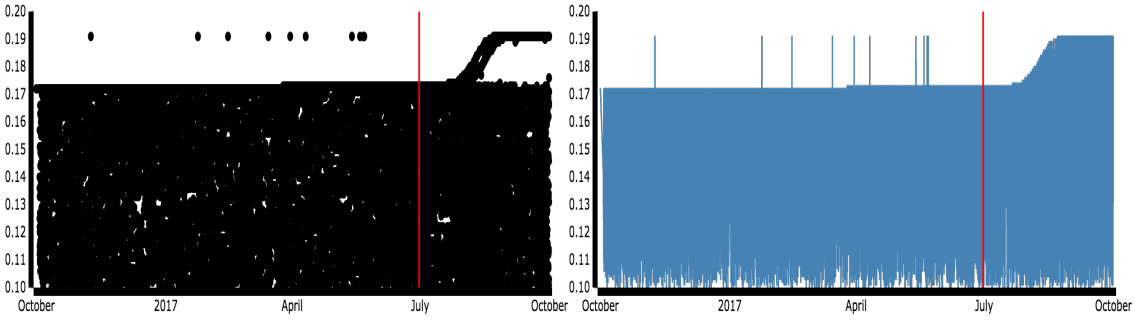

Figure A.13: N_062

A.1.2 The features that have visual drifts from pre-processing 3

Here we show the visualization of detection results of part of the features which have some visual drifts by using the third pre-processing strategy. They look almost the same as the result from the first strategy. However, ADWIN did not manage to detect the drifts on C_054, N_000, N_036, N_062 and N_142 by using data from this strategy. Hence, we choose to use the first strategy in all the experiments for Rabobank transaction data.

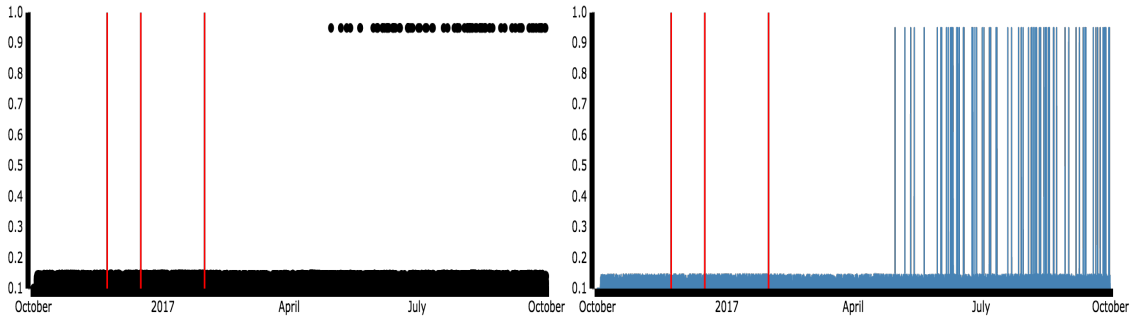

Figure A.14: N_135

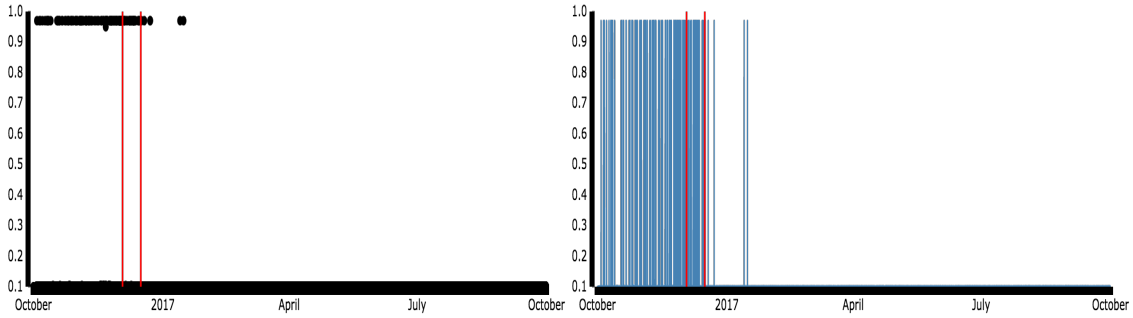

Figure A.15: C_046

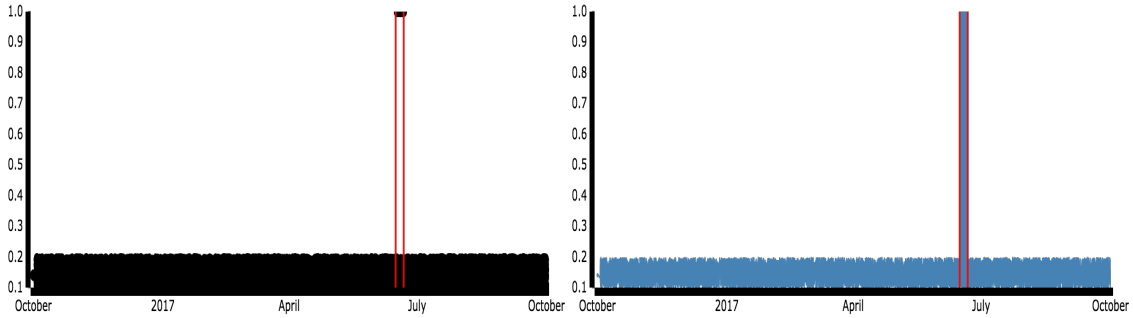

Figure A.16: N_114

A.2 Detection log file


```
{
  ``num_words": [
    [
      ``2017-10-26 12:25:38",
      ``323.3532608695652",
      ``561.4125"
    ],
    [
      ``2017-11-06 04:25:38",
      ``400.9234375",
      ``503.6421875"
    ],
    [
      ``2017-11-16 20:25:38",
      ``408.443359375",
      ``509.90393518518516"
    ]
  ],
  ``num_sentences": [
    [
      ``2017-10-29 04:25:38",
      ``31.142473118279568",
      ``128.16666666666666"
    ]
  ]
}
```


Listing A.1: Example dashboard log file.

List A.1 shows an example log file (in JSON format) generated from the detection part of the framework (result from ADWIN), where the feature names are the keys. Each of the keys has an array which is used to describe a detected drift. This description array has 3 items in it. The first is the time t that the drift happened, the second is the mean value μ_1 of the feature before the drift, and the last value is the mean value μ_2 after the drift. More specifically, we take the first description array as an example, the feature "num_words" has detected to have drifted from the detector at time t ="2017-10-26 12:25:38". In fact, t is the time that a warning level has triggered by the detector, and it will be followed with another time t_d where a drift level has triggered otherwise no drift will be alarmed. The μ_1 is the mean value of the feature between the last drift level point and this warning level point t, and μ_2 is the mean value of the feature between the time t and t_d. A significant increase or decrease of the mean value indicates a drift in the feature.