Title

Bayesian Nonparametric Methods for Cyber Security with Applications to Malware Detection and Classification

Abstract

The statistical approach to cyber security has become an active and important area of research due to the growth in number and threat of cyber attacks perpetated nowadays. In this thesis, we centre our attention on the Bayesian approach to cyber security, which provides several modelling advantages such as the flexibility achieved through the probabilistic quantification of uncertainty. In particular, we have found that Bayesian models have been mainly used to detect volume-traffic anomalies, network anomalies and malicious software. To provide a unifying view of these ideas, we first present a thorough review on Bayesian methods applied to cyber security.

Bayesian models applied to detecting malware and classifying them into known malicious classes is one of the cyber security areas discussed in our review. However, and contrary to detecting traffic and network anomalies, this area has not been widely developed from a Bayesian perspective. That is why we have centred our attention on developing novel supervised learning Bayesian nonparametric models to detect and classify malware using binary features built directly from the executables' binary code. For these methods, important theoretical properties and simulation techniques are fully developed and for real malware data, we have compared their performance against well-known machine learning models which have been widely applied in this area.

With respect to our methodologies, we first present a new discrete nonparametric prior specifically designed for binary data that builds on an elegant nonparametric hierarchical structure, which allows us to study the importance of each individual feature across the groups found in the data. Moreover, and due to the large, and possibly redundant, number of features, we have developed a generalised version of the model that allows the introduction of a feature selection step within the inferential learning. Finally, for a more complex modelling where there is a need to introduce dependence across the features, we have extended the capabilities of this new class of nonparametric priors by using it as the building block of a latent feature model.

Word Count / Number of Pages

35174 / 206

Year of Submission

2022

Academic School

School of Mathematics, Statistics and Actuarial Science

Bayesian Nonparametric Methods for Cyber Security with Applications to Malware Detection and Classification

JOSE ANTONIO PERUSQUIA CORTES

School of Mathematics, Statistics and Actuarial Science

University of Kent

Thesis submitted for the degree of

Doctor of Philosophy in Statistics

PhD. Supervisors:

DR CRISTIANO VILLA
PROF JIM GRIFFIN

September, 2021

This page intentionally left blank

Abstract

The statistical approach to cyber security has become an active and important area of research due to the growth in number and threat of cyber attacks perpetated nowadays. In this thesis, we centre our attention on the Bayesian approach to cyber security, which provides several modelling advantages such as the flexibility achieved through the probabilistic quantification of uncertainty. In particular, we have found that Bayesian models have been mainly used to detect volume-traffic anomalies, network anomalies and malicious software. To provide a unifying view of these ideas, we first present a thorough review on Bayesian methods applied to cyber security.

Bayesian models applied to detecting malware and classifying them into known malicious classes is one of the cyber security areas discussed in our review. However, and contrary to detecting traffic and network anomalies, this area has not been widely developed from a Bayesian perspective. That is why we have centred our attention on developing novel supervised learning Bayesian nonparametric models to detect and classify malware using binary features built directly from the executables' binary code. For these methods, important theoretical properties and simulation techniques are fully developed and for real malware data, we have compared their performance against well-known machine learning models which have been widely applied in this area.

With respect to our methodologies, we first present a new discrete nonparametric prior specifically designed for binary data that builds on an elegant nonparametric hierarchical structure, which allows us to study the importance of each individual feature across the groups found in the data. Moreover, and due

to the large, and possibly redundant, number of features, we have developed a generalised version of the model that allows the introduction of a feature selection step within the inferential learning. Finally, for a more complex modelling where there is a need to introduce dependence across the features, we have extended the capabilities of this new class of nonparametric priors by using it as the building block of a latent feature model.

To my family,

For your everlasting love and support.

Acknowledgments

It has been almost four years from the beginning of this incredible, and sometimes difficult journey. All the personal and professional growth that I have achieved would not have been possible without the patience, support and continuous encouragement of my supervisors, Cristiano Villa and Jim Griffin. Thank you so much for believing in me, giving me the opportunity to work by your side and learn from you, it has been an absolute honour. I would also like to acknowledge the University of Kent, SMSAS, and its people, responsible of creating such a vibrant and friendly place to be. I would like to use this space to particularly thank Maria Kalli, Fabrizio Leisen, Claire Carter and Derek Baldwin, for your friendship and for all the technical and non-technical support and advice that you gave me throughout these years. I wish nothing but the best for you and your loved ones.

On a more personal note, I still remember how difficult it was to leave everything and everyone I love to start this journey. Few days before my arrival, my beloved Mexico was struck by tragedy and saying goodbye became even more difficult. However, I quickly found myself surrounded by amazing people which would later become some of my closest friends ever. Alan, Alex, Aniketh, Larry and Michele thank you so much for all the laughs, the games, the late dinners, lunch chats and discussions, and drinks we had in these years, I love you guys. Of course, as years went by, this journey also allowed me to make some great new friends that there was no way I would not mention. Fabian, Fenia, Ian, Joe, Mariza, Peter and Sarah, thank you so much for getting to know me and more importantly, for letting me know you.

And last, but not least, there are several people that I want to acknowledge before directing our attention to the contents of this thesis. In first place, I would like to thank Ramsés Mena for his continuous advice and guidance before, during and after the PhD. I would also like to thank my close friends and family that I left in Mexico, for always being there for me despite the distance. In particular, there are three wonderful individuals that have been there with me since the start and that without their patience, love and caring words I would not be where I am. That is why this work is entirely dedicated to: my mom, thank you for literally giving me everything in this life; to my brother, thank you for your always sincere words and camaraderie; and of course to Yazmín, thank you for being the light in my life that never goes out, I love you.

Contents

Introduction

Nowadays, cyber security is an important concern for all individuals, organisations and governments globally. Cyber attacks have become more sophisticated, more frequent and more dangerous than ever, and traditional anomaly detection methods have been proved to be less effective when dealing with these new classes of cyber attacks. In order to address this, both classical and Bayesian statistical models offer a valid and innovative alternative to the traditional signature-based methods, motivating the increasing interest in statistical research that it has been observed in recent years. In particular, Bayesian methods yield interesting insights and answers to the uncertainty around large and complex systems like computer networks and hence, for cyber security data.

Statistical models applied to cyber security anomaly detection have been mainly centred on three different types of cyber threats, which are: volume-traffic anomalies, network anomalies and malicious software (malware). From a Bayesian perspective, both parametric and nonparametric models have been effectively designed and applied to detect such anomalies. However, and despite the increase in cyber security research from a Bayesian point of view, there are still some modelling challenges that need to be fully addressed and some areas that have not been widely explored as others and that need to be more deeply developed. One of such areas is the detection and classification of malware.

A malware is any program designed to damage, disrupt or gain unauthorised access to computer systems. Understandably, the accurate detection of malware has become an important task in the field of cyber security. However, when dealing with a malware it is also important to know its type and its family to speed-up the analysis and hence, to have a deeper understanding of the threat and the damage caused. In order to detect and correctly classify malware one can either perform a *static* analysis or a *dynamic* one. From a statistical point of view both approaches represent interesting lines of work that require different assumptions and types of models due to the nature of the data considered.

To understand both the *static* and the *dynamic* approach and how they can be used to detect and classify malware, it is important to remark that all benign and malicious software, without regard of the type and family, are composed of a list of instructions that are executed during runtime and hence, they can be used to determine the true nature of a program. In the *dynamic* approach the idea is to run the malware in a safe environment and analyse the sequence of instructions (e.g. *mov, jmp, push, call*) as they occur. Whereas in the *static* approach, the objective is to analyse the binary content of the malware, that is, the sequence of bytes expressed in hexadecimal notation (e.g. *31 2e 20 ...*) representing all the instructions, strings, headers, and other relevant metadata, without the malware been executed at all.

From a Bayesian perspective, it is compelling to notice that for the *dynamic* approach there are already some methodologies that have been developed and applied; whereas, the *static* approach has been mainly addressed using data mining and machine learning models. In this direction, one way to differentiate benign from malicious executables is to leverage on their hexadecimal representation by creating a set of binary features that completely characterise each executable. These binary features are usually created by considering a contiguous sequence

of n bytes that can represent one or more instructions, a string, a call or another piece of information regarding the malware. Therefore, we believe this is an interesting scenario that has not yet been considered from a Bayesian perspective, and that is why in this thesis we centre our attention to developing novel supervised learning Bayesian models for binary data that provide an effective probabilistic approach to discrimination tasks.

The models developed and presented in this thesis are built under nonparametric assumptions. It can be broadly argued that Bayesian nonparametrics deals with the construction of probability measures on large parameter spaces, yielding more flexible models than their parametric counterpart. For obvious reasons, the mathematical and computational challenges required for nonparametric models are more complex and more demanding. However, due to the continuous advancements of computational software and the increasing research on efficient simulation methods, Bayesian nonparametric models are no longer intractable and have become widely popular in many research areas and computer science and cyber security are not the exception.

With respect to cyber security research, Bayesian nonparametric methods have been effectively used in detecting network anomalies and, when modelling the set of instructions executed by the malicious software, for malware detection and classification. As we further explore in Chapter 1, these methodologies have at their core one of the most important Bayesian nonparametric priors: the Dirichlet process (Ferguson, 1973). One of the most appealing characteristics of this stochastic process is its almost sure discreteness, which provides a natural framework for clustering applications and density estimation. This is certainly attractive and useful when trying to detect anomalies by comparing them to clusters of normal behaviour.

In this thesis, however, we centre our attention on developing Bayesian nonparametric methods that have another vital nonparametric prior at their core: the beta process (Hjort, 1990). This stochastic process was originally introduced for modelling random hazard functions in the real line. However, more general spaces have also been considered. In particular, when combined with the Bernoulli process it provides a natural framework for modelling binary data like in document classification (see *e.g.* Thibaux and Jordan, 2007). Hence, it also provides an interesting approach to the static analysis of malware where for each executable we have a set of binary variables that characterise them. Moreover, the beta and Bernoulli processes can also be used for factorial models like in the Indian Buffet Process (IBP) (see *e.g.* Griffiths and Ghahramani, 2005, 2011), where the interest relies on characterising the data through a set of K latent binary traits. It is interesting to notice, that this featural representation of the data can also be thought as a clustering one with 2^K possible configurations.

Although the core chapters of this thesis are centred on the malware detection and classification tasks through Bayesian nonparametric supervised learning models, there is another important contribution of this work that we believe and hope researchers would find useful for future statistical cyber security research. This contribution takes the form of a review of Bayesian methods that have been effectively applied to cyber security anomaly detection problems, and that we use as the basis for Chapter 1. In this chapter, we also discuss some of the inherent challenges that the statistical community faces when dealing with the detection of cyber threats, including a quick discussion on alternative methods. Finally, we would like to remark that this review has been accepted for publication at the *International Statistical Review*.

As for the novel Bayesian nonparametric models, in Chapter 2 we present the first version of the model that we have named *beta-CoRM*. This nonparametric model builds on the idea of vectors of dependent completely random measures as defined in Kingman (1967), which allows the construction of a suitable Bayesian hierarchical model for supervised learning. A similar Bayesian nonparametric approach is the hierarchical beta process (Thibaux and Jordan, 2007), where d beta processes have the same directing beta process as their base measure, allowing the sharing of information across groups. The beta-CoRM model, however, modifies the jumps of the directing beta process at group level providing more flexibility at the moment of detecting influential features. Results and a comparison against machine learning supervised algorithms on malware detection and classification data sets are also presented and discussed.

In Chapter 3 we present a generalisation of the beta-CoRM model which did not only yield a better classification accuracy, but it also allowed us to introduce a feature selection step within the inferential learning. This generalisation arose from an elegant *spike-and-slab* interpretation given to the score parameter of the original model. With this interpretation in mind, the basic idea is for each feature to have a unique score parameter whose posterior values would allow us to detect the features with the best discriminative power. The theory and results of Chapters 2 and 3 are the basis of the paper submitted to *Bayesian Analysis* which is currently undergoing a major revision.

The models developed and described in Chapters 2 and 3 are built on the assumption that the binary features are conditionally independent. However, in practice we would like to introduce some level of dependence that might allow a more complete generative process. To this end, in Chapter 4, we introduce a binary matrix factorisation procedure that uses the *beta-CoRM* as the prior on a set of latent traits. The structure of this model allows us to introduce dependence

not only across features belonging to the same observations, but also a level of dependence between features within and across groups.

Lastly, in Chapter 5 and Chapter 6 we present respectively two future lines of research and the final remarks. For the future work, we first centre our attention on the modelling of binary matrices using a Bayesian nonparametric logistic regression model. To this end, the IBP is considered as the nonparametric prior since it allows us to use a Polya-gamma augmentation scheme for the inferential procedure. Finally, the second line of research is to consider the matrix of counts rather than just the binary features. Prior evidence on the data shows that by doing so, the differences among the groups become clearer and hence, a better classification could be achieved. For both approaches, interesting details and theoretical results are presented and discussed.

Chapter 1

Bayesian models applied to cyber security anomaly detection problems

Cyber security can be broadly defined as the set of tasks and procedures required to defend computers and individuals from malicious attacks. Its origin can be traced back to 1971, a period where the Internet, as we know it today, was not even born. Among the computer science community it is widely accepted that it all started with Bob Thomas and his harmless experimental computer program known as the Creeper. This program was designed to move through the ARPANET[1] leaving the following message: "I'm the creeper: catch me if you can". Inspired by Bob Thomas' Creeper, Roy Tomlinson created an enhanced version, allowing the Creeper to self-replicate; therefore, coding the first computer worm. Later on, he would also design the Reaper which can be considered the

[1]The Advanced Research Projects Agency Network (ARPANET) was a packet switching network developed in the late 1960s that is widely considered to be the predecessor of the Internet (Oppliger, 2001).

first antivirus program, since it was designed to move across the ARPANET and delete the Creeper.

Despite a harmless origin, some years later the world would find out that network breaches and malicious activity were more dangerous than expected and cyber threats became a serious matter. Nowadays, cyber security is considered a major concern that affects people, organisations and governments equally, due not only to the growth of computer networks and Internet usage but also to the fact that cyber attacks are more sophisticated and frequent than ever. These attacks represent a complex new challenge that demands more innovative solutions, hence, it requires a multi-disciplinary effort in order to be well-prepared and protected against such threats. Some of the disciplines involved in this task include computer science, computer and network architecture and statistics (Adams and Heard, 2014).

In this introductory chapter we are mainly interested in the Bayesian approaches to cyber security problems and we centre our attention on how the discovery of cyber threats has been tackled as an anomaly detection problem. In particular, we discuss three different classes of anomaly detection problems within cyber security research and the representative types of data used in each one of them. The first of these problems is about volume-traffic anomaly detection, the second one about network anomaly detection and, finally, the third one is about malware detection and classification.

We of course, acknowledge that methods other than the Bayesian ones are suitable to deal with cyber threats (see *e.g.* Buczak and Guven, 2016; Chandola et al., 2009; Gupta et al., 2014; Adams and Heard, 2014, for reviews on classical statistics, machine learning and data mining approaches). The intent of this chapter is to present the reader with a Bayesian perspective, discussing the avail-

able options, their advantages and limitations in order to have a comprehensive understanding of the methodologies that the Bayesian framework provides. In Section 1.1, we provide some of the reasons on why we believe Bayesian methods are an interesting and appropriate approach to cyber security anomaly detection.

Traditionally, cyber security threat detection systems have been built around signature-based methods; in this approach, large data sets of signatures of known malicious threats are developed and the network is constantly monitored to find appearances of such signatures. These systems have been proved effective for known threats but can be slow or ineffective when dealing with new ones, with mutations of known ones or with time-evolving threats. Dealing with these threats is one of the reasons why we need to consider alternatives to signature-based methods. In order to do so, statistics offers a wide range of options for cyber security problems; these include both classical and Bayesian approaches that, in general, can be built on either parametric or nonparametric assumptions. However, in essence, statistical anomaly detection methods usually build a model of normal behaviour to be considered as a benchmark, so that departures from this behaviour might be an indication that an anomaly has occurred.

Cyber security research from a mathematical and statistical point of view has proved to be an interesting and complex challenge that has led to an increasing interest in recent years. There are various reviews and reports (see *e.g.* Willinger and Paxson, 1998; Catlett, 2008; Meza et al., 2009; Dunlavy et al., 2009) that outline some of the key areas, problems and challenges the mathematical community faces. It was early remarked (Willinger and Paxson, 1998) how the constant changes in time and sites made the Internet such a difficult object to understand. Since then, all authors have agreed that, due to the continuously exponential growth of the Internet and computer networks, there is a need for statistical models able to scale well to high-volume data sets of real time het-

eroscedastic and non-stationary data, which represents (among other things) a significant computational challenge. Moreover, as pointed out by Catlett (2008), the mathematical models used should be able to effectively distinguish between harmless anomalies and malicious threats.

The need to design on-line detection methods able to handle high-volumes of data is not the only challenge discussed in these reviews. For example, Catlett (2008) also discussed the role mathematics play, by allowing us to understand computer networks, the Internet and malware behaviour in providing predictive awareness for secure systems. On a separate note, the author also remarked the need to advance the state of the art in graph theory and large-scale simulation to understand the spreading process of malicious code. Meza et al. (2009) further emphasised the importance of having access to reliable data.

The lack of reliable data has been mainly due to privacy and confidentiality reasons and has made researchers study the best way to anonymise or sanitise the data. Some methods, as discussed in Bishop et al. (2006), include using synthetic data, extracting the data from sources with no privacy constraints or a proper sanitising process. For example, in user-systems any characteristic that can be associated to an individual should be suitably changed (e.g. IP address or user name). More complex anonymisation processes have also been developed; for example, in Tang et al. (2010) the authors described a process based on subnet clustering where three parts of a whole IP address are anonymised by different methods. Fortunately, as we see in the following sections, there are some publicly available data sets that can be used for research purposes.

All of the other challenges just described can be well-grouped into three general cyber security research areas (Dunlavy et al., 2009). The first area deals with the modelling of large-scale dynamic networks, like the modern Internet or

any current computer network. Their mathematical representation (just as with any other network) is through a graph theory formulation (Newman, 2010) and, historically, the Erdös-Rényi formulation of random graphs provided a mathematical model that could handle small-scale networks like the infant Internet (Chen et al., 2015). However, nowadays computer networks are examples of large-scale dynamic networks, so they have a large amount of nodes and edges (representing the computers/users and the connections among them) that are constantly evolving and changing over time and that are not completely random. Hence, there is a need to develop more sophisticated network mathematical formulations and new statistical techniques for comparing them. The reader could refer to Olding and Wolfe (2014) for a review on classical graph theory methods applied to modern network data.

Discovering cyber threats is the second cyber security research area. As already established, cyber attacks are more sophisticated and frequent than ever, hence, the need for models capable of detecting malicious activity along with their variations, complicated multi-stage attacks and, if possible, the source of the cyber attack (Dunlavy et al., 2009). Moreover, the detection methods should ideally be designed for on-line detection and able to handle time-evolving data as well. This is the area we explore in more depth in this introductory chapter.

Finally, and due to the fact that almost all cyber attacks work by spreading malicious code through a vast number of the computer network's nodes, the last of the mathematical challenges found in cyber security is related to network dynamics and cyber attacks. This area is mainly dedicated to understanding the spreading characteristics of the malicious code through a computer network, before and after it has been detected and protections have been released. Particularly interesting problems consist in determining the potential limit of the infection and the interplay of the malicious spreading and the protection pro-

cesses. More details and challenges related to this area can be found in Dunlavy et al. (2009).

The remainder of the chapter is organised as follows: in Section 1.1 we provide a gentle discussion on why Bayesian statistics yield an interesting approach to complex systems and data sets such as the ones found in cyber security. In Sections 1.2, 1.3 and 1.4 we describe and explore respectively, volume-traffic anomaly detection, network anomaly detection and malware detection and classification. In each of these sections we provide a gentle description of the kind of data used, we present some of the Bayesian models used to address these problems and we go through particularly interesting case studies found in the literature. In Section 1.5 we provide an insight into alternative cyber threat anomaly detection procedures. In Section 1.6 we describe some emerging challenges. Lastly, Section 1.7 presents final points of discussion.

1.1 Why focus on Bayesian models?

Statistical anomaly detection models have become increasingly popular in cyber security research. From the classical statistics and the machine learning points of view, it is possible to find in the literature comprehensive reviews for the above problems (see *e.g.* Buczak and Guven, 2016; Chandola et al., 2009; Gupta et al., 2014; Adams and Heard, 2014). That is why, we have then deemed as appropriate to provide the reader with a review on the Bayesian perspective, and in this section, we will highlight some of the reasons why a Bayesian approach might be considered. In particular, we provide the reader with motivations why Bayesian statistics yield interesting approaches to the modelling of large and complex systems, such as computer networks. However, it is important to keep an

open-minded approach in considering the methodologies discussed in this review, as neither classical nor Bayesian statistics (or machine learning) provide obliquitous solutions, and it is always fundamental to consider the problem at hand and its context in order to identify the most suitable approach. For a general introduction to Bayesian statistics and its governing ideas, the reader could refer to Bernardo (2003) Goldstein (2013) and Gelman et al. (2013), to mention a few.

Centring on cyber security, we can find Bayesian models in machine learning that have been successfully developed and used to provide solutions to several anomaly detection problems such as the latent Dirichlet allocation (Section 1.3.1.2), Bayesian clustering (Section 1.3.2.1), Poisson factorisation (Section 1.3.2.3) and more general Bayesian nonparametric methods (Section 1.3.2.4). These models are linked by an attempt to fit large latent variable models for which Bayesian inference is particularly attractive, and also allows us to find unobserved structure in the data.

A second important remark about Bayesian methods, is about their inherent probabilistic representation of uncertainty. Having probabilistic statements associated to unknown quantities, such as parameters or predicted values, leads to an understanding of such statements that is clearer than other methods (e.g., classical statistics). The above fundamental property of Bayesian methods is essentially appealing in an anomaly detection framework, because uncertainty can be propagated to predictions making them, often, more stable.

Finally, Bayesian methods also allows us to combine different types of information in a single inferential framework, and more general forms of Bayesian reasoning. In this direction, Bayesian networks 1.3.1.1 deserve special mention, since as explained in Chockalingam et al. (2017), using them would not only allow us to combine different sources of knowledge, but also handle and overcome the

scarcity of data related to cyber attacks, which sometimes represent a big issue for their modelling.

1.2 Volume-traffic anomaly detection

To begin developing statistical methods for computer network data, it is useful to have a high-level description of a computer network. The Open Systems Interconnect (OSI) is a widely-used conceptual set of rules for computer systems to be able to communicate with one another. The correct and reliable transmission of information is achieved through the joint work of seven sequentially connected layers, each one with its own purpose. These layers are: the physical layer, the data link layer, the network layer, the transport layer, the session layer, the presentation layer and the application layer. For a thorough understanding of these layers and their role in the communication process the reader can refer to Hall (2000) and Myhre (2001).

Since these layers work in conjunction with one another, malicious activity could be targeted to any of the layers in order to destabilise the communication process between computer systems. For the purposes of this section we restrict our attention to the third layer of the OSI-model: the network layer. This layer is in charge of structuring and managing a multi-edge network including addressing, routing and traffic control (Hall, 2000). The data is transmitted by breaking it down into pieces called packets that contain the user data (or payload) and the control information which provides data for delivering the payload, e.g. source and destination network addresses, error detection codes and segment information.

The packet rate, which is defined as the number of packets per time unit moving across the network, is one of the most common volume-traffic characteristics used for analysing a network's traffic. Its constant surveillance is useful for the detection of certain cyber threats that create changes in the network's normal traffic behaviour, such as distributed denial of service (DDoS) attacks which are intended to saturate the victim's network with traffic. Volume-traffic data sets can be obtained upon request from Los Angeles Network Data Exchange and Repository (LANDER) project. Another free network flow data set is described in Kent (2015b). The downloadable file "flows.txt.gz" presents network flow events from 58 consecutive days within Los Alamos National Laboratory's corporate internal computer network; each event is characterised by 9 variables: time (t), duration (dur.), source computer (src. comp.), source port (src. port), destination computer (des. comp.), destination port (des. port), protocol (prot.), packet count (packets) and byte count (bytes). The first three events included in the file are reported, as an illustration, in Table 1.1.

t	dur.	src. comp.	src. port	des. comp.	des. port	prot.	packets	bytes
1	0	C1065	389	C3799	N10451	6	10	5323
1	0	C1423	N1136	C1707	N1	6	5	847
1	0	C1423	N1142	C1707	N1	6	5	847

Table 1.1: Extract from the network flow events (Los Alamos National Library).

In this data set we can identify two different kinds of variables. First, we have access to volume-traffic characteristics such as the packet or byte count which can be used for volume-traffic anomaly detection purposes. The second set of variables characterise each event by providing the source and destination computer, the ports and the protocol used which allow us to perform a more refined analysis by developing multi-channel detectors by splitting the traffic into separate bins represented by the source or destination. Furthermore, as we discuss in Section 1.3, these variables will be useful for a different kind of network anomaly

detection models.

1.2.1 Bayesian approaches to volume-traffic anomaly detection

Volume-traffic anomaly detection is concerned with detecting cyber attacks that produce changes in traffic measures, such as the packet rate. The main goal is to detect as fast as possible changes in the normal behaviour. Once a change has been detected, an alarm needs to be sent off so that the system can be checked and then decide whether there has been an attack or not (false alarm). It is important to remark that false alarms could yield important interruptions in the computer network, so there is a need to find the true change by seeking a low false positive rate as well. This yields a tradeoff between the detection delay and the false alarm rate that needs to be considered for the detection procedure. The methods used to analyse these kind of data are mainly based on the statistical theory of change-point analysis.

1.2.1.1 Change-point analysis

The main objective of change-point analysis is the accurate detection of changes in a process or system that occur at unknown moments in time. In a single change-point setting it is assumed that there is a sequence of random variables $\{X_n\}_{n\geq 1}$ with a common probability density function (pdf) f, known as the pre-change density, that is, $X_n \sim f(X_n|X^{(n-1)})$, where $X^{(n-1)} = (X_1, ..., X_{n-1})$. Then, at an unknown time ν, something unusual occurs and from the time $\nu+1$ onwards $X_n \sim g(X_n|X^{(n-1)})$. In this setting ν is known as the change-point and the pdf $g \neq f$

is called the post-change density. It is important to remark that theoretically, the densities f and g might depend on n and ν as well, in fact, allowing these densities to depend on n and ν might help us to more realistically explain time-evolving data found while doing cyber security research. In practice, g might only be known up to some unknown parameters θ, hence, in some applications the problem can be reduced to detecting changes in mean, changes in variance or changes in both.

In order to deal with change-point detection problems, one could either follow a non-sequential approach, where the objective is to detect the changes in a fixed set of observations, or a sequential approach, where the goal is to detect changes as new data arrives. Since both of these approaches have been tackled from a classical and a Bayesian perspective, the choice will certainly depend on the type of problem at hand and the objective of the analysis. From a cyber security point of view there is a need for constant surveillance of the computer network, therefore it is important to have fast on-line detection procedures. That is why in this chapter we only provide an insight into the sequential change-point analysis theory (for a complete review the reader can refer to Polunchenko and Tartakovsky (2011)) and how it has been applied to volume-traffic anomaly detection problems.

1.2.1.2 Sequential change-point analysis

As established in the previous section, the objective of the sequential approach to change-point analysis is to decide after each new observation if the common pdf is still f or if it has changed. One of the main challenges of this approach, is the fact that the detection should be done with as few observations as possible while raising a low number of false alarms. In other words, a compromise must

be reached between the losses associated to the detection delay and to the false alarms. Therefore, as explained in Polunchenko et al. (2012), an ideal sequential procedure should minimise the average detection delay (ADD) subject to a constraint on the false alarm rate (FAR). In the literature, several approaches to analyse the tradeoff and hence, several detection procedures, have been considered. However, for the purposes of this chapter, we centre our attention on the Bayesian formulation, where the change-point ν is a random variable.

From a statistical perspective, at each step there is a need to test the hypothesis $H_k : \nu = k \geq 0$ vs $H_\infty : \nu = \infty$. In the scenario where both f and g are known, a detection statistic based on the likelihood ratio (LR), $\Lambda_n^k = \frac{p(X^{(n)}|H_k)}{p(X^{(n)}|H_\infty)}$, is chosen and supplied to an appropriate detection procedure defined as a stopping time T with respect to the natural filtration $\mathscr{F}_n = \sigma(X^{(n)})$. For example, in the Bayesian framework, the Shiryaev-Roberts (SR) procedure (Shiryaev, 1963; Roberts, 1966) and several modifications of it, have been widely used and analysed. Now, in the scenario where f and g are not known, the LR can not longer be used. In order to address this, in Tartakovsky (2014) it is suggested replacing the LR for a score sensitive function, yielding a suitable modification of the detection statistic. The choice of the score function will depend on the type of change one is trying to detect, for example, for a change in mean, in Tartakovsky et al. (2006a,b) the authors used a linear memoryless score function.

Once a realisation of the detection procedure takes place at, say, time T, we have a false alarm if $T \leq \nu$ otherwise, the detection delay is given by the random variable $T - \nu$. For example, in the Bayesian framework, the SR procedure is defined as the stopping time $S_A = \inf\{n \geq 1 : R_n \geq A\}$ where A is the detection threshold, and R_n is the SR statistic which can be recursively computed as $R_n = (1 + R_{n-1})\Lambda_n$ with initial value $R_0 = 0$. Other detection procedures are defined similarly, and hence, the idea is to find the optimal stopping time T_{opt}

with an average run length to false alarm (ARL) above a desired level $\gamma > 1$ such that the ADD is minimised.

Interesting optimality results have been obtained within the original formulation, where the detection procedure is only applied once. However, for surveillance applications, such as cyber security, where the detection procedure is repeatedly applied, a renewal mechanism needs to be specified. For example, assuming an homogeneous process, the monitoring starts from scratch after every alarm, yielding a multi-cyclic model (Tartakovsky, 2014), where a sequence $T_1, T_2, ...$ of independent detection times are recorded. In this multi-cyclic setting, the objective is to find the optimum stopping time in the set $\Delta(\gamma) = \{T : ARL(T) \geq \gamma\}$ such that the stationary ADD (SADD) is minimised, where the SADD can be thought as the limiting case of the ADD and $\gamma > 1$. From a Bayesian perspective it was proved in Pollak and Tartakovsky (2009) that the SR procedure is optimal in this multi-cyclic with respect the SADD.

From a practical point of view, one way to find the threshold detection A and hence, the optimal detection procedure is by considering the asymptotic case as $\gamma \to \infty$. In this chapter we do not cover the mathematical reasoning behind this, the reader could refer to Polunchenko et al. (2012) for its thorough understanding. However, it is important to mention that under this asymptotic approach, for the SR procedure one can use the approximation $ARL(S_A) = \gamma \approx A/\zeta$ to find A, where $\zeta = \frac{1}{\mathbb{E}_0(Z_1)} \exp\left[-\sum_{k=1}^{\infty} \frac{1}{k}(\mathbb{P}_\infty(S_k > 0) + \mathbb{P}_0(S_k \leq 0))\right]$, $Z_n = \log(\Lambda_n)$ and $S_n = \sum_{i=1}^{n} Z_i$. Therefore, for a large and fixed value γ the detection threshold is equal to $A = \gamma\zeta$, where ζ can be computed directly or approximated using a Monte Carlo scheme.

In network security, change-point detection theory provides a natural framework for volume-traffic anomaly detection. Still, there are some important con-

siderations we need to contemplate. In Tartakovsky (2014) it is argued that the behaviour of both pre- and post-attack traffic is poorly understood, as result, neither the pre- nor post-change distributions are known and as already explained the LR can not longer be used. Another important observation stated in Tartakovsky et al. (2006a,b) is that, in certain conditions, splitting packets in bins and considering multichannel detectors helps localise and detect attacks more quickly. This multichannel setting can be thought as a generalisation of the classical change-point detection problem, where an n-dimensional stochastic process is observed simultaneously and at a random time only one of the entries changes its behaviour. This setting might be useful when dealing for example with certain Denial of Service (DoS) or DDoS attacks where it has been observed that an increase number of packets of certain size occurs during the attack.

1.2.2 Case study: ICMP reflector attack

The Internet Control Message Protocol (ICMP) reflector attack was a distributed denial of service (DDoS) attack that sent echo reply packets to a victim within Los Nettos Internet Service Provider network that lasted 240 units of time. Due to the nature of the attack, a change-point model is a sensible approach to analyse it, and in this section we discuss how the multi-cyclic SR procedure based on the $N(\mu, a\mu)-\text{to}-N(\theta, a\theta)$ change-point model proposed in Polunchenko et al. (2012) was used to detect it.

In this change-point model, it is assumed that the normal behaviour of the system follows a $N(\mu, a\mu)$ distribution and after the change-point, ν, the system follows a $N(\theta, a\theta)$ distribution. This is an assumption that as the authors mentioned, is interesting for a wide variety of applications including of course, cyber security. In particular, for this ICMP attack, we centre our attention on the

packet rate, which historically, has been modelled using a Poisson process (see *e.g.* Cao et al., 2003; Karagiannis et al., 2004) with an arrival rate equal to the average packet rate, which increases when an anomaly occurs. However, and despite the discrete nature of the time series, the authors found in their exploratory analysis that a Gaussian assumption was more realistic for these observations. As a result, point estimates of the mean and variance for the normal traffic and the traffic during the attack were obtained and are given by $\widehat{\mu}_{\text{nor}} = 13329.764$, $\widehat{\sigma}^2_{\text{nor}} = 266972.736$ and $\widehat{\mu}_{\text{atk}} = 17723.833$, $\widehat{\sigma}^2_{\text{atk}} = 407968.14$ respectively. A reproduction of the real trace using these estimates is displayed in Figure 1.1.

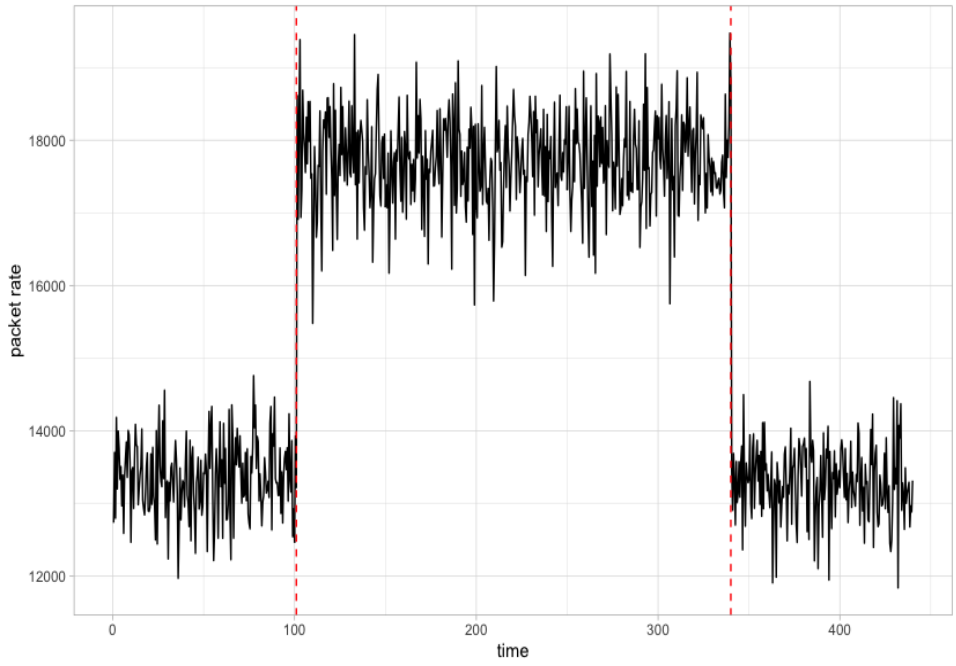


Figure 1.1: Reproduction of the original trace.

Due to the abrupt change in the trace, any good change-point detection procedure should be able to detect it. However, it is clear that in practice, this ideal situation will not always occur. Hence, and in order to test the model and its detection performance on a more challenging and realistic set, the authors manually

lowered the intensity of the attack by applying the transformation

$$X_i^* = \sqrt{13600 * 20.028} \times \frac{X_i - 17723.833}{\sqrt{407968.14}} + 13600$$

on the recorded observations during the attack. By doing so, the authors changed the real trace for it to behave as a $N(\mu, a\mu)-\text{to}-N(\theta, a\theta)$ model with parameters $\mu = 13329.764$, $\theta = 13600$ and $a = 20.028$. Figure 1.2 shows the simulated trace after applying the same transformation.

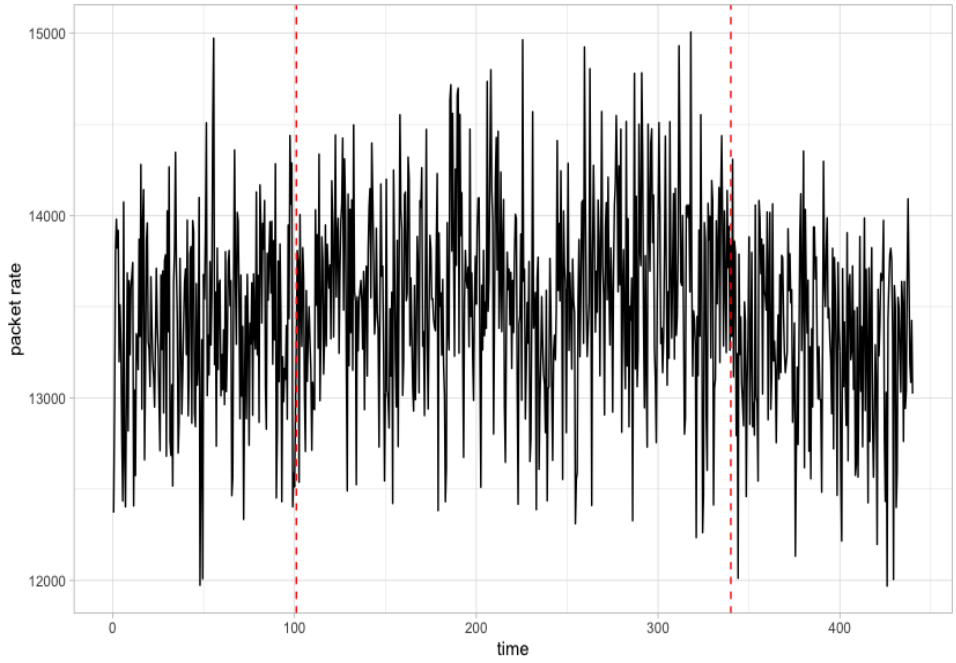


Figure 1.2: Reproduction of the transformed trace. The red lines indicate the beginning and the end of the attack.

In order to find the detection threshold for the SR procedure, γ is fixed to be 1000 so that the asymptotic results described in Section 1.2.1.2 can be used, yielding a value $A = 731.3$. Then the multi-cyclic procedure can be applied, so that each time an alarm is raised the process starts afresh. The results of this procedure are illustrated in Figure 1.3. It can be appreciated that the true attack

is detected 22 seconds after it started and that the procedure raised two false alarms. It is clear that this model works reasonably well since the attack was detected (even though at first sight it was not visible) not long after the attack started. However, it is important to notice that it also raised two false alarms close to one another.

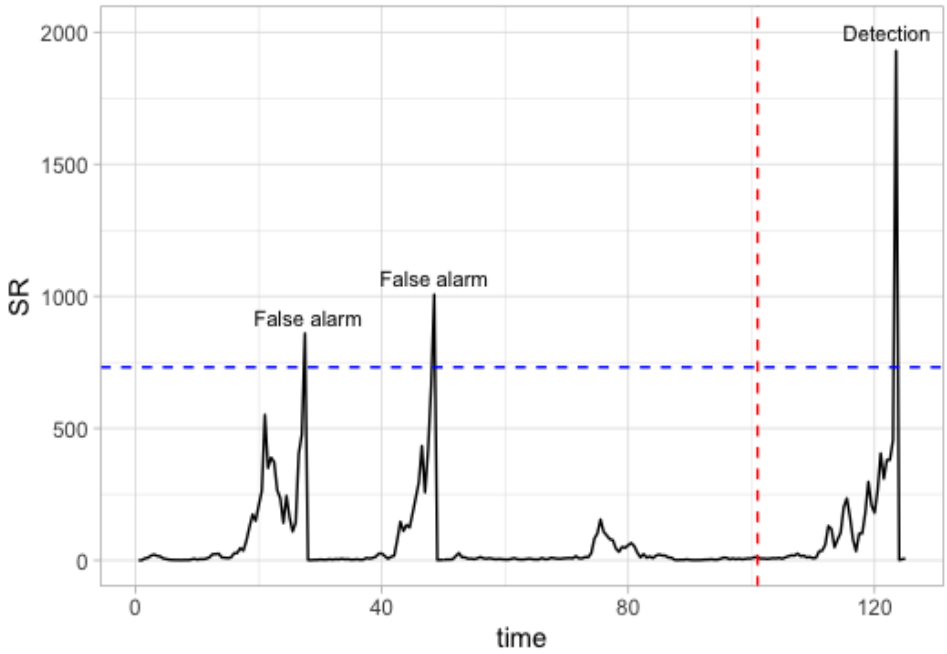


Figure 1.3: Multi-cyclic SR procedure on the diminished trace illustrated in Figure 1.2. The blue line represents the detection threshold A and the red line the attack's start time.

1.3 Network anomaly detection

Monitoring volume-traffic data is one important method of cyber security anomaly detection. However, there are other variables we can consider for network analysis. For example, monitoring the features that characterise each packet such as

its length, the version, the header length, the priority, or the characteristics of the connections between computers such as the source and destination computer or protocol can be used and need different anomaly detection methods. From a statistical perspective there is an interest in characterising the normal pattern connections within a computer network and this is usually done by creating clusters of normal behaviour, so that any new activity that cannot be grouped into these clusters will be flagged as an anomaly.

As we discuss in the following sections, the network anomaly detection models considered in this chapter have been mainly targeted to detect cyber security attacks such as intrusion detection, misuse of credentials, rogue users, etc. This research area has been tackled with a wide range of Bayesian models and for a clearer understanding of the chapter, we split them into two subsections depending on the kind of data used. That is, some of the methods have been used on the connections occurring within a computer network that can be viewed as a bipartite graph. The second approach is to model a sequence of multivariate discrete variables that characterise these connections such as the ones found in Table 1.1. However, it is important to keep in mind that the objective, no matter the approach, is to provide a probabilistic characterisation of the connections in a computer network.

1.3.1 Network Flows

Computer networks are complex systems that are able to provide a vast amount of information. In particular for each connection occurring within the network there is the possibility to record a multivariate sequence of events that characterise each connection. For example, through some monitoring software such as *tcpdump* or *Wireshark*, we can capture information regarding the IP source,

the IP destination, the network protocol (e.g. TCP/IP, HTTPS), the length of the packet, the flags, among other variables, that can help us in order to detect anomalies. As an example, in Table 1.2 we display 3 TCP connections of a user within a small computer network. Other data sets such the one described in Section 1.2, (Table 1.1), also contain information about each connection such as the protocol and the packet or byte count.

time (s)	IP src	IP destn	IPv	Flags	seq	ack	win	length
.000160	xxx.xxx.x.72	xxx.xxx.x.67	4	-	-	1	2048	0
.052568	xxx.xxx.x.72	xx.xx.xxx.210	4	-	-	35	4096	0
10.842233	xxx.xxx.x.72	xxx.xxx.xx.130	4	P	2945:3891	1	2048	946

Table 1.2: 3 TCP connections of a user with sanitised IP address within a small network captured using *tcpdump*.

1.3.1.1 Bayesian networks

A Bayesian network (BN) is a directed acyclic graph in which the nodes are the variables and the vertices represent the direct influences among the variables and their parent nodes (Pearl, 1985). These influences are measured through the conditional probabilities and hence the model is completely characterised by them. For cyber security research purposes, BN's have been mainly used for intrusion detection. It is commonly argued that BN's yield robust models able to capture more realistic scenarios since they can directly model the combined effects of the vulnerabilities, contrary to other models where individual vulnerabilities are measured and then aggregated (Frigault and Wang, 2008). In Kruegel et al. (2003) it was also discussed that using BN's might reduce the number of false positives that other anomaly detection models usually face.

One of the first approaches to intrusion detection through BN's can be found in Valdes and Skinner (2000), where the authors developed the eBayes TCP model

in order to analyse temporally contiguous bursts of traffic at periodic intervals. In this model, the root node is the (unobserved) session class and the child nodes different (observed) variables, such as number of unique ports, service distribution and event intensity. At each interval the idea is to know if an attack is taking place and the session class is assumed to propagate as a discrete Markov chain through the intervals. A similar approach for network packet traces can be found in Jing and Shelton (2010), where a time continuous Markov chain is used instead. Other approaches for intrusion detection can be found in Kruegel et al. (2003), where BN's are used to classify events as *normal* or *anomalous*, and in Pauwels and Calders (2018), where the authors made an extension of dynamic Bayesian networks in order to model and detect anomalies on log files within the context of Business Processes, which are series of structured activities in order to perform a task.

Finally, another interesting use that researchers have given to BN's is the modelling of *attack graphs* that represent how different network vulnerabilities could be combined in order to breach the network's security. This has been especially useful in order to measure and assess the risk associated to these vulnerabilities and therefore, for risk management (see *e.g.* Dantu and Loper, 2004; Frigault and Wang, 2008; Poolsappasit et al., 2012).

1.3.1.2 Topic models

Topic models are probabilistic models that have their origins in latent semantic indexing (LSI) modelling (Deerwester et al., 1990). LSI uses the singular value decomposition of a term-document association matrix in order to create a space where related terms and documents are placed near one another. Hofmann (2001), using this idea, developed the probabilistic latent semantic analysis (PLSA) which

can be considered the first topic model. His approach uses a generative latent class model to perform a probabilistic mixture decomposition. Other models, such as the latent Dirichlet allocation (LDA) model (Blei et al., 2003), have also been introduced with the task of discovering the topics that occur in a set of documents. However, they have been widely used in other fields where there is a need of unsupervised clustering.

The topics produced by these models are clusters of similar words which allow examining a set of documents. As discussed in Blei et al. (2003) two of the usual assumptions made are that the words in a document are exchangeable (also known as the "bag-of-words" assumption) and so are the documents. These assumptions allows us to exploit de Finetti's representation theorem for exchangeable random variables[2]. In the LDA model, the basic idea is that every document in the corpus can be represented as a random mixture over a known and fixed number k of latent topics, which are characterised by a distribution over words. It is further assumed that the word probabilities are characterised by an unknown but fixed matrix β of dimensions $k \times M$ that needs to be estimated, where M is the size of the vocabulary. In practice M is usually large, thereby creating issues related to sparsity and with the prediction of new documents. In order to address this, Blei et al. (2003) also developed a fuller Bayesian approach, usually called the smoothed LDA, by allowing β to be random and a Dirichlet prior distribution assigned to each row β_i.

The LDA model's original setup includes a corpus X with N documents $\mathbf{w}_1$,...,$\mathbf{w}_N$, each one having P_n words, $w_{1,1}, ..., w_{n,P_n}$. The length of each document can be sampled from a Poisson distribution or from a more realistic document-length distribution. For each of the words in the n-th document we first select a

[2]de Finetti's representation theorem is due to Hewitt and Savage (1955) who generalised de Finetti's theorem for exchangeable 0-1 random variables (de Finetti, 1930)

random topic z_i from which a random word will be assigned. The mathematical representation of the generative process is:

$$\begin{aligned} \theta_n &\sim \text{Dirichlet}(\alpha) && n \in \{1, ..., N\} \\ z_i &\sim \text{Multinomial}(\theta_n) && i \in \{1, ..., P_n\} \\ w_i &\sim \text{Multinomial}(\beta_{z_i}). \end{aligned}$$

In matters of anomaly detection, Cao et al. (2016) used the LDA model to analyse features obtained from the packet headers captured using the *tcpdump* software. In their approach, the documents are represented by the tcpdump traffic obtained within a time slot and the words are the unique packet's network features. The LDA model is used on free attack traffic data in order to learn its feature patterns and new traffic data is then compared against it. The authors proposed using the likelihood of a new document as the detector of anomalous activity. A similar procedure can be found in Cramer and Carin (2011), where the LDA model and the dynamic LDA (dLDA) (Pruteanu-Malinici et al., 2010) are considered to analyse Ethernet packets. In their approach the data is also divided into fixed time intervals and the words are any event of interest observed across 45 well-known ports used for network topic modelling. In their results the dLDA proved to be a better choice for modelling this kind of data due to the dLDA's ability to analyse time-dependent documents by letting the weights over the topics to change in time.

1.3.2 User-computer connections

Now we turn our attention to the Bayesian analysis of the connections and authentications on a computer network. In research some of the most commonly

used data sets for modelling computer network behaviour belong to LANL (see *e.g.* Hagberg et al., 2014; Kent, 2015b; Turcotte et al., 2018). These data sets are mainly comprised of network and computer events collected from LANL enterprise network. For example, the User-Authentication Associations in Time data set (Hagberg et al., 2014) encompasses 9 months of successful event authentications for a total of 708,304,516 connections. As an illustration, the first four events are shown in Table 1.3.

time	user	computer
1	U1	C1
1	U1	C2
2	U2	C3
3	U3	C4

Table 1.3: User-computer authentications associations in time.

This kind of data sets allow us to view the computer network as a bipartite graph with users and computers as nodes and the connections as edges. This approach is particularly important for network anomaly detection since we can analyse and study the normal connection pattern of each individual, group them and even learn their expected behaviour by studying their peers. The models used for each of these tasks will be useful for detecting anomalies such as intrusion detection, misuse of credentials or rogue users which can and will compromise the network if they go undetected.

1.3.2.1 Bayesian clustering

Clustering comprises a set of unsupervised learning models that attempts to create homogeneous groups from heterogenous observations. From a Bayesian perspective, and as explained in Lau and Green (2007), is that the set of clusters

created work as a parameter of the model for the data. Therefore, the inference on the partition is carried out through the posterior distribution that can be done through MCMC procedures. The reader can refer to Lau and Green (2007) and the references therein for an overview on Bayesian clustering procedures.

For cyber security research, in Metelli and Heard (2016), the authors used a 2-step procedure for inferring cluster configurations of users with similar connection patterns and at the same time modelling new connections across the network. The first step uses a Bayesian agglomerative clustering algorithm with the choice of the multiplicative change in the posterior probability as a similarity measure. This algorithm yields an initial cluster configuration of users with similar connection behaviours, which is then used in a Bayesian Cox proportional hazards model (Cox, 1972) with time-dependent covariates for the identification of new edges within the computer network. In this case, the chosen covariates for a connection between the i-th user and the j-th computer are the overall unique number of authentications over time for the computer and the restriction of these authentications to the user's cluster. This 2-step procedure requires a Markov Chain Monte Carlo (MCMC) algorithm for the joint update of the initial clusters and the coefficient parameters.

Working along this line, Metelli and Heard (2019) presented a Bayesian model for new edge prediction and anomaly detection using a Bayesian Cox regression model like the one previously introduced in Metelli and Heard (2016). However, in the most recent approach the authors used a more robust set of covariates and the initial cluster configuration was obtained through the spectral biclustering algorithm (Dhillon, 2001). The covariates used can be grouped into two different classes: the first group is comprised of the unique number of authentication over time for each client (time-varying out-degree), the unique number of authentication over time to each computer (time-varying in-degree) and two indicator

functions, respectively specifying if the last connection and the last two connections made by the client were new. The second set of covariates represent what the authors described as the notion of attraction between clients and servers. For their construction both hard-threshold and soft-threshold clustering models were used in a latent feature space.

1.3.2.2 LDA

The LDA model as explained in Section 1.3.1.2 has also been shown to be a valid and useful technique for network modelling using authentication records. The reader can refer to Heard et al. (2016) for an example on how the LDA model can be used to analyse computer network connection traffic data to determine the number of users present. In their approach, each document is represented by the day's authentication records, different users are the topics and the destination computers play the role of the words. In this scenario each of the entries of θ_n indicates how active was the respective user in the n-th day. As discussed by the authors this procedure could play an important role for detecting misuse of credentials.

1.3.2.3 Poisson factorisation

Topic models are not the only probabilistic models used for cyber security research that have been originally designed for other purposes. Poisson factorisation (PF) models, which are widely used for recommender systems in machine learning (see *e.g.* Gopalan et al., 2014), have also been used for network anomaly detection. PF is a probabilistic model of users and items that was proposed as an alternative to the classical probabilistic matrix factorisation (PMF) (Salakhutdinov and Mnih,

2007).

The main assumption is that the data can be represented in a matrix, in the recommender system the rows are the clients and the columns the number of items. Each entry of this matrix is assumed to be the rating given by a certain user to a particular item, and these are modelled using the dot-product of latent factors for both the users and the items. PMF assumes that each entry is normally distributed and Gaussian priors are assigned on the latent factors for both the users and movies. This will theoretically imply that the ratings could become negative which is something we would not desire to have. In order to address this issue, PF assumes that both the users and the latent factors are non-negative, and so a Poisson distribution for the entries and gamma distributions for the latent factors are used instead. These assumptions make PF more applicable to real data sets like The Netflix Prize data set (Bennett and Lanning, 2007), where we have a set of users and the rankings for each movie in the catalogue.

With respect to cyber security research, Turcotte et al. (2016a) considered PF models for peer-based user analysis which provides a better understanding of the individuals by learning their peer's behaviour. The basic idea is that computer users with similar roles within an organisation will have similar patterns of behaviour. This type of analysis can be particularly important for quickly detecting rogue users. The behaviour of a new user can be compared to their peers and anomalies detected. The data used is the recorded authentication events (Table 1.3) as briefly explained in Section 1.2. The model is completely specified by letting Y_{ui} be the number of times that user u authenticates on machine i and where it is assumed that $Y_{ui} \sim \text{Poisson}(\theta_u \beta_i)$, where, θ_u for $u = 1, \ldots, U$, and β_i, for $i = 1, \ldots, C$, are k-dimensional vectors of positive values. The model is interpreted as having k latent features characterising the users (such as job title, department, etc.) with θ_u representing their scores for the u-th user and k latent

features characterising each computer (such as the number of daily processes, the type of computer, etc.), with β_i representing their scores for the i-th computer.

A certain feature might have a high score for all machines within one department and low scores otherwise. If a user had a high-score on that feature then they are likely to have many authentication events on machines in that department (perhaps, representing that they work in that department). If a user had a low-score on that feature then they are likely to have a very low number of authentication events. In general, the mean number of authentication events for a user on a machine is the sum over products of many features which allows similarities between users and computers to be learnt from the data. The specification of the model is completed by assuming that $\theta_{u,j} \overset{i.i.d.}{\sim} \text{gamma}(a, \zeta_u)$, $\beta_{u,j} \overset{i.i.d.}{\sim} \text{gamma}(b, \eta_i)$, $\zeta_u \sim \text{gamma}(a', b')$ and $\eta_i \sim \text{gamma}(c', d')$. The model is fitted to a training sample and anomalies can be detected by comparing predictions from this model to observed values from a testing sample.

1.3.2.4 Dirichlet process

So far, we have only described Bayesian models working under parametric assumptions. Although these models have been proved effective in detecting network anomalies, there is still a missing methodology that we need to consider: the nonparametric realm. Bayesian nonparametric models have become increasingly popular and appealing in research areas such as finance, biology and machine learning, among others, because they posses a flexibility that can hardly be achieved by parametric models. This flexibility is explained through the fact that these models, in contrast to parametric ones, assume that the parameter space is an infinite-dimensional object, something particularly useful as more data becomes available. For cyber security research, where huge amounts of time evolving

data are available almost instantly, Bayesian nonparametrics models should provide more suitable modelling techniques and interesting insights and solutions to the problem at hand.

Historically, it is widely accepted that Bayesian nonparametrics had its beginnings with the introduction of the Dirichlet Process (DP) by Ferguson (1973) and since then, the DP has played a vital role in Bayesian nonparametrics and its applications (see *e.g.* Hjort et al., 2010). The DP works as a prior on the space of probability distributions and just as the Dirichlet distribution it has a nice conjugacy property. More precisely, if $\{X_i\}_{i=1}^n \overset{\text{i.i.d}}{\sim} P$ and $P \sim DP(\alpha)$ then $P|X_1, ..., X_n \sim \text{DP}\left(\alpha + \sum_{i=1}^n \delta_{X_i}\right)$. In this setting, α is a finite measure defined on the same space as P and it is commonly expressed as $\alpha = \theta P_0$, where θ is the total mass and P_0 a probability measure. Using this structure we obtain a nice expression for the predictive distribution

$$X_{n+1}|X_1, ..., X_n \sim \begin{cases} \delta_{X_j^*} & \text{with probability proportional to } n_j \\ P_0 & \text{with probability proportional to } \theta, \end{cases}$$

where $\{X_j^*\}_j$ is the set of distinct values observed in $X_1, ..., X_n$ and n_j their frequencies. Hence, we will be observing either a previously seen value or a completely new one, a characteristic which is really useful in clustering applications.

Exploiting the structure of the posterior and predictive distribution of the DP, Heard and Rubin-Delanchy (2016) developed a Bayesian nonparametric approach to intrusion detection by assuming a DP-based model for each message recipient on a set of computers and the directed connections among them that represent the node set V and the set of edges E respectively in a directed graph (V, E), which can be thought as a set of objects connected together where the source node and destination node can be identified for each connection (the direction

matters). The first step of their anomaly detection procedure is to obtain the predictive p-value for the event x_{n+1}, defined as $p_{n+1} = \sum_{x \in V: \theta_x^* \leq \theta_{x_{n+1}}^*} \frac{\theta_x^*}{\theta^*}$, where $\theta_x^* = \theta P_0(x) + \sum_{i=1}^n \delta_{X_i}(x)$ and $\theta^* = \theta + n$. These p-values quantify the level of surprise of a new connection. Since the goal is to detect anomalies in each source computer, the m p-values observed in the edge (x, y) are reduced to a single score using Tippett's method (Tippett, 1931). Then a single score for each node x is obtained using Fisher's method (Fisher, 1934). Finally, the computers are ranked through these scores, and compromised ones should have higher ranks.

Working along this line, Sanna Passino and Heard (2019) examined a joint model of a sequence of computer network links $\{(x_i, y_i)\}_{i=1}^n$, with x_i and y_i representing the source and destination computer respectively, based on the Pitman-Yor process (PY) (Perman et al., 1992). The PY process, also known as the two-parameter Poisson-Dirichlet process, requires two parameters which are usually denoted by $\sigma \in [0, 1)$ (the discount parameter) and $\theta > -\sigma$ (the strength parameter). An appealing characteristic of the PY process is the closed form of the predictive distribution given by:

$$X_{n+1}|X_1, ..., X_n \sim \begin{cases} \delta_{X_j^*} & \text{with probability proportional to } (n_j - \sigma) \\ P_0 & \text{with probability proportional to } \theta + k\sigma, \end{cases}$$

where k is the number of different observations and P_0, n_j and X_j^* are defined just as for the DP. We can immediately notice that if $\sigma \to 0$ we recover the DP, thus, the PY process can be thought as a generalisation of the DP. Another interesting remark is that the probability of a new value depends on k, so the more unique observations we have, the more likely it will be to obtain new samples from P_0. This is certainly useful for applications where power-law behaviour is observed, something the DP can not achieve. Finally, as $\sigma \to 1$, unique observations X_j's having small frequencies n_j are unlikely to be sampled.

In Sanna Passino and Heard (2019) the joint modelling of $p(x, y)$ is achieved through the decomposition $p(x|y)p(y)$ by assuming the sequence of destination nodes, $\{y_i\}$, to be exchangeable with a hierarchical PY distribution and conditioned on the destination node the sequence of source nodes connecting to that destination, $\{x_i|y\}$, are also exchangeable with a hierarchical PY distribution with parameters depending on y. The mathematical representation of the model is:

$$
\begin{aligned}
x|y &\sim F_{x|y_i} \\
y_i &\sim G \\
F_{x|y} &\sim \mathrm{PY}(\alpha_y, \theta_y, F_0) \\
G &\sim \mathrm{PY}(\alpha, \theta, G_0)
\end{aligned}
$$

As for the detection procedure, it follows the same reasoning as in Heard and Rubin-Delanchy (2016), that is, for each source computer one needs to obtain the predictive p-values and combine them into a single score. Besides the use of the PY process rather than the DP, there are two other interesting results found in this approach. The first one is that the authors did not restrict their attention to the use of p-values and they further explored the use of mid p-values (the reader can direct the attention to Lancaster (1952) and Rubin-Delanchy et al. (2019) for an insight on mid p-values and why in some problems they might be preferred over p-values). Finally, they also explored Pearson (Pearson, 1933) and Stouffer's (Stouffer, 1949) p-value combiners.

1.3.3 Case study. LANL user-authentication data set

The user-authentication data set by Los Angeles National Laboratory (Kent, 2015a) is a comprehensive summary of 58 days of traffic of the enterprise's network, which also contains a set of 48,079 red team compromise events resulting from a simulated intrusion attack. In total, there are four source computers compromised (C17693, C18025, C19932, and C22409), and the task is to identify the unusual activity occurring and flag these computers. These connections between source and destination nodes can be modelled as a bipartite graph and hence, we could use the model proposed by Heard and Rubin-Delanchy (2016) or Sanna Passino and Heard (2019) in order to identify the compromised computers.

As described in Section 1.3.2.4, these models allow us to quantify the surprise of each new connection through the p-values or mid p-values, which are then aggregated for each source computer to have a unique score. These scores are then used to rank the computers and, in the case of the compromised ones, a high anomaly score should be assigned. In Table 1.4 we present the results that Heard and Rubin-Delanchy (2016) obtained with the DP p-values aggregated using Fisher's method (Fisher, 1934) and the best results of Sanna Passino and Heard (2019) obtained with the PY mid p-values aggregated using Pearson's method (Pearson, 1933).

Src. Comp.	Rank with DP p-values	Rank with PY mid p-values
C17693	5	1
C18025	94	74
C19932	5347	2754
C22409	7172	6984

Table 1.4: Rank of the compromised source computers.

1.4 Malware detection and classification

The malware detection and classification problem is the third group of cyber threat investigation problems that we consider in this chapter. A malware is defined as a software specifically designed to disrupt, damage or gain access to a computer system. Nowadays there are many types of malware, such as spyware, adware, ransomware, among others, including several variations of them. That is why the fast detection of unknown malware is one of the biggest concerns of cyber security. However, accurate detection is not the only task required when dealing with malicious software. Malware have to be classified into families for a better understanding of how they infect computers, their threat level and therefore, how to be protected against them. Correct classification of new malware into known families may also speed-up the process of reverse-engineering to fix computer systems that were infected. In order to have good explanatory and predictive models for the malware detection and classification problem, researchers have mainly used the content of the malware in two ways, either by using the hexadecimal representation of the binary code or the dynamic trace.

1.4.1 Hexadecimal representation and n-grams

In computer science the byte is the basic unit of information for storage and processing and it is most commonly represented by a sequence of 8 binary digits (bits). Every instruction given to a computer (malicious or not) can be broken down into sequences of bytes, which form the instruction's binary code. These sequences can be expressed in a more compact way through their hexadecimal representation, where each byte is written as a combination of two elements of the set $\{0, 1, ..., 9, A, B, ..., F\}$. For malware detection and classification the hexadec-

imal representation of the binary code is used in conjunction to what is known in literature as n-grams.

An n-gram is usually defined as a contiguous sequence of n elements; these elements, depending on the field, can be letters, numbers, words, etc. In computer science and for malware detection and classification purposes, these elements are the hexadecimal representation of each byte. For example, considering $n = 4$, for the code extract given by $00\,00\,1C\,40\,2A\,28$, we take all the possible sequences of 4 contiguous bytes to create the set of 4-grams, that is $\{00001C40, 001C402A, 1C402A28\}$. The elements of the set of all the different n-grams are assumed to completely characterise benign and malicious code through their presence and absence. Hence, the set of n-grams needs to be modelled for the prediction of new observations, contrary to the classical n-gram analysis where these structures are used as a probabilistic model for the prediction of the next item in a sequence. In practice, it has been observed that the set of unique n-grams is huge even for small data sets (easily reaching the order of millions), so it becomes computationally unfeasible to use all this information and a feature selection procedure is required as a first step.

To the best of our knowledge, there is no Bayesian approach to malware detection using the hexadecimal representation of the binary code. However, we believe this line of work should be appealing to the Bayesian community as well, due to assumptions made and the challenge it represents. First of all, the data can be represented by a $N \times M$ binary matrix where the rows are either malicious and benign code. From a Bayesian perspective one could be interested in modelling this binary matrix using a conjugate beta-Bernoulli hierarchical model or by choosing more complex models like a logistic regression for malware detection. It should also be clear that this is a high-dimensional problem in both rows and columns, so Bayesian nonparametric models could be definitely helpful

for analysing this sort of data. Moreover, the assumptions made can be easily generalised for the classification into known families problem.

1.4.2 Dynamic traces as Markovian structures

The second approach to malware detection and classification relies on the dynamic traces of the malware, which are basically the set of instructions executed in order to infect the system. Following the ideas of Storlie et al. (2014), many authors have assumed that these traces have a Markovian structure and in that way the interest relies on modelling and analysing the probability transition matrix. Since there are hundreds of commonly used instructions and thousands of them overall, modelling the one-to-one transition is not feasible. However, there are some instructions that perform the same or similar task so creating groups of similar instructions is a reasonable first step. Storlie et al. (2014) developed four different categorisations with $8, 56, 86$ and 122 groups of similar instructions. In practice the most widely used categorisation is the one with 8 classes, which include among others: math, memory, stack, and other.

The mathematical framework is fully specified by letting c to be the number of instruction categories previously chosen (e.g. 8), then a dynamic trace is defined as a sequence $\{x_1, ..., x_n\}$ with $x_i \in \{1, ..., c\}$ that are modelled as a Markov chain (MC). Hence, we let $\mathbf{Z}_i$ be the transition counts matrix for the i-th program, $\mathbf{P}_i$ to be the probability transition matrix and B_i an indicator if the program is malicious or not. Storlie et al. (2014) proposed that the entries of the estimated $\mathbf{P}_i$, denoted by $\widehat{\mathbf{P}}_i$, should be used as predictors to classify a program as malicious or not through a logistic spline regression model. In practice, the actual predictors used to model B_i are: $\text{logit}(\widehat{P}_{i,1,1})$, $\text{logit}(\widehat{P}_{i,1,2})$, ..., $\text{logit}(\widehat{P}_{i,c,c-1}), \text{logit}(\widehat{P}_{i,c,c})$. Finally, a symmetric prior Dirichlet distribution with

parameter ν is used as a prior for each row of $\mathbf{P}_i$.

Working directly on this approach, Kao et al. (2015) proposed a flexible Bayesian nonparametric approach to model the probability transition matrices $\{\mathbf{P}_i\}_i$ by using a mixture of Dirichlet processes (MDP) (Antoniak, 1974), that is,

$$\begin{aligned} \mathbf{P}_i|\mathbf{Q}_i, \sigma &\overset{\text{i.i.d.}}{\sim} MD(\sigma\mathbf{Q}_i) \\ \mathbf{Q}_i|G &\overset{\text{i.i.d.}}{\sim} G \\ G &\sim DP(\theta, P_0), \end{aligned}$$

where P_0 follows a matrix Dirichlet (MD) distribution centered in some constant matrix $\widehat{P}$, $\sigma > 0$ controls the variance of $\mathbf{P}_i$ and the matrix $\mathbf{Q}_i$ is the shape parameter. This MD distribution implies that each row of $\mathbf{P}_i$ independently follows a c-dimensional Dirichlet distribution with concentration parameter equal to the corresponding row of $\sigma\mathbf{Q}_i$. The model is completely specified by letting B_i be the indicator random variable of maliciousness just like before. A new program i^* is classified as malicious if $\mathbb{P}(B_{i^*} = 1|\mathbf{Z}_{\mathbf{i}^*}, \mathscr{Z})$ exceeds a predefined threshold, with $\mathscr{Z}$ being the collection of all observed counts matrices. Moreover, if the program is malicious it can be further classified into a cluster with existing programs that share common features.

A different approach for the modelling of dynamic traces was developed by Bolton and Heard (2018). Their approach followed the same assumptions, i.e. that dynamic traces, specified by the prior clustering of common instructions, have a Markovian structure. However, they further assumed that this structure changes over time with recurrent regimes of transition patterns. Hence, each dynamic trace can be modelled as a MC with a time varying transition probability matrix $\mathbf{P}(t)$. In order to detect the regime changes three change-point models were described. The basic idea is that there are $k \geq 0$ change-points that partition

the dynamic trace and where within each segment the trace follows a homogeneous MC. The methods vary in the way the probability transition matrices are defined within each segment. The first one changes the whole matrix in each segment, the second one only allows some of the rows to change and finally the regime switching method allows the change in rows not only to be forward but also to go back to a vector of probabilities that governed the Markov chain in earlier segments. Finally, the authors proposed a classification procedure based on a similarity measure of the vectors of change-points and their regimes, that obtains the minimum value for the two samples of the proportions of instructions which occur within regimes shared by both traces. A high level of similarity requires that a large number of observations in both traces are drawn from common regimes.

1.4.3 Case study: Malware dynamic traces

In this section we discuss and present the methodology followed in Bolton and Heard (2018) to classifying 141 malware provided by reverse engineers of Los Alamos National Laboratory into families and subfamilies using their dynamic traces. The first thing we would like to remark, is the fact that in order to perform this kind of analysis, there is a need to safely execute the malware in a controlled sandbox environment to prevent infecting the system and to record the instructions as they are executed. This is certainly a complex task that is time consuming; however, it also provides key information on how the malware was created and their objective, which is really important because it allows to create some sense of similarity among them and therefore, to discover the type of malware we are dealing with.

As explained in Section 1.4.2, in Bolton and Heard (2018) the authors assumed that the traces had a time varying homogeneous Markovian structure. And in order to perform the classification, three methodologies were considered. For the first one and in order to compare their time evolving assumption, an homogeneous MC was assumed and in order to assess the similarity among traces a standard square exponential kernel was used on the empirical transition probability distributions. For this approach, a nearest neighbour algorithm was used for the classification procedure.

The second methodology considered was the Bayesian approach developed by the authors, where a reversible jump MCMC algorithm was used in order to obtain a posterior estimate of the similarity measure of two traces defined as the minimum of the proportion of shared instructions within regimes. With these posterior estimates, both single link and nearest neighbour algorithms were considered for the classification. Finally, and in order to improve the accuracy, the authors also developed a hybrid method combining the previous two approaches. As a first step, the kernel methodology was used as a filter by selecting the malware whose distance was less than 1.05 from the new malware. Then the regime-switching model was applied to this reduced set and in order to perform the classification, the authors proposed Fisher's p-value combiner and hence, assigning the family which yielded the lowest value using Fisher's method. The reported accuracy performance for the three methods is illustrated in Table 1.5.

Methodology	Kernel	Regime-Switching	Hybrid
Family	91.00	91.00	94.00
Subfamily	85.00	84.00	89.00

Table 1.5: Accuracy performance (in %) of the kernel, the regime-switching and the hybrid approaches to dynamic malware classification.

1.5 Alternative cyber threat anomaly detection approaches

It is imperative to stress that the models and the kind of data described in the previous sections are far from being exhaustive. They represent the ones that we have found to be the most frequently used for the general class of cyber threat anomaly detection problems presented here. However, there are other kind of cyber security related problems that have been tackled from a Bayesian perspective and that could be appealing for future research purposes.

For example, Turcotte et al. (2016b) used computer event logs to identify misuse of credentials within a computer network. In their approach these logs are treated as an aggregated multivariate data stream comprised of the client computer x, the server computer y and the type of event e. These features were modelled independently for each user credential and the probability of an observed triplet (x_t, y_t, e_t) at time t is modelled using the conditional probabilities, that is, $\mathbb{P}(x_t, y_t, e_t) = \mathbb{P}(x_t)\mathbb{P}(y_t|x_t)\mathbb{P}(e_t|x_t, y_t)$. For each of these components an appropriate multinomial-Dirichlet based model was considered, and in order to detect anomalies, the predictive distributions were used to obtain the p-value that can be compared against a predefined threshold.

A second example for detecting compromised credentials can be found in Price-Williams et al. (2018), where the authors proposed a users' activity anomaly detection approach by analysing the amount of user activity on a given day and the times where these activities were realised. A seasonal behaviour model was developed by first constructing a model to measure the user's activity in a certain period and then using the events registered, a change-point density estimation model was used to estimate the times at which the events occurred. In this setting,

users working at hours that differ from their normal schedule are considered to be anomalous activities.

Other interesting approaches are aimed to obtain a better understanding of a computer network's behaviour. An example of this can be found in Price-Williams et al. (2017), where the authors' main goal is to detect automated events that can be viewed as polling behaviour from an opening event originated by a user. It was discussed that achieving this should yield an improvement in the statistical model used and enhance its anomaly detection capabilities. In this approach a change-point model was used in each edge of the computer network in order to separate human behaviour from automated events. This methodology works as an alternative to the one presented in Heard et al. (2014), where the discrete Fourier transform was used.

Being able to detect automated events from human activity is not the only approach undertaken for a better understanding of a computer network. Nowadays, there are computer networks that contain a vast number of nodes and thereby, a large number of connections among them. In practice, temporal independence for the nodes is usually assumed to have mathematical tractability; however, this is a strong assumption and a deeper understanding of the dependance among these nodes is required. A recent approach by Price-Williams et al. (2019) aimed to detect and understand the interactions between computer nodes to detect correlated traffic patterns to reduce false positives when performing anomaly detection. A test based on higher criticism (Donoho and Jin, 2004) was used to detect this dependence.

1.6 New and emerging challenges

Some of the challenges the mathematical community face when dealing with cyber security problems have already been described in the introduction of this chapter. These include challenges related to the data itself, like the privacy and ethical issues, and to the models and how there is a need to handle large volumes of non-homogeneuous data. However, we would like to describe two of the main challenges that still need to be fully considered in Bayesian cyber security research.

1.6.1 Robustness

As already established, in cyber security an anomalous activity might be a sign that an attack is occurring, so it is imperative to detect it as fast as possible while keeping a low false positive rate. Clearly, the performance of most of the anomaly detection models heavily rely on the data used in the training step. Ideally, this data should be as reliable as possible and among other things, it should be noise-free. However, in real case scenarios, and especially nowadays with larger and more complex networks, delivering noise-free data might not be an easy task to achieve. Therefore, designing robust models able to deal with noisy data and to capture more complex (and realistic) scenarios is also a crucial task.

From a non-Bayesian perspective, there are already some robust anomaly detection models that have been used in cyber security. For example, Eleazar (2000) described a probabilistic approach to anomaly detection without training on normal data in order to detect intrusions in UNIX system call traces; Paffenroth et al. (2018) developed a robust principal component analysis (RPCA)

assuming noisy and missing data on network packets; Hu et al. (2003) used robust support vector machines (RSVM) to study intrusion detection over noisy data. Finally, it is worth mentioning that deep learning models have also been used in order to provide more robust models, like the deep autoencoders which are a class of unsupervised neural networks (Berman et al., 2019).

From a Bayesian perspective, robust models applied to cyber security data are still little explored. Some of the already known robust models include Bayesian networks which are able to model more realistic scenarios by analysing the combined effect of the vulnerabilities. For volume-type traffic data, there is also a need of more robust models, since this type of data usually contains outliers, deriving from normal activities. In this direction, but not directly applied to cyber data, Knoblauch et al. (2018) developed a robust Bayesian online change-point detection algorithm that achieved promising results when dealing with outliers in well-log data and analysing noisy measurements of nitrogen oxide levels.

1.6.2 Scalability

Bayesian models have become appealing due to their rich theoretical background and ability to model complex data. Bayesian inference relies on the posterior distribution of the parameters which, in most of the cases, will not be known in a closed form. In order to obtain samples from the posterior, one could use approximations to the unknown integral or to the posterior, with the intent of minimising the discrepancy that forms the basis of variational Bayes (see *e.g.* Blei et al., 2017, for a review on variational inference). In this direction, stochastic gradient descent methods can be used to scale variational Bayes methods to very large data sets. Alternatively, in some instances an MCMC scheme could be designed to produce correlated samples from the posterior. Although theo-

retically efficient, MCMC schemes often face several computational issues due to a slow mixing and slow convergence, which usually get worse when dealing with high-dimensional data. Possible solutions include, parallel MCMC, approximate MCMC, C-Bayes and Hybrid algorithms. A thorough review on theoretical and practical aspects of scalable Bayesian models can be found in Angelino et al. (2016).

From a Bayesian perspective, some scalable approaches have been designed for modelling and detecting anomalies in cyber security applications. Examples of these include: Clausen et al. (2018), where a Markov-modulated Poisson process embedded in a fast and scalable Bayesian framework was used in the modelling for network flow data; Chen et al. (2018), where a novel class of Bayesian dynamic models was introduced and applied to Internet traffic and, according to the authors, the sequential analysis is fast, scalable and efficient; Muñoz González et al. (2017) explored two methods for scalable inference on Bayesian attack graphs; and other models like the one described in Heard and Rubin-Delanchy (2016) and Sanna Passino and Heard (2019) are fully parallelisable and suitable for platforms designed for Big Data analysis like Hadoop.

1.7 Concluding remarks

Cyber security research from a mathematical and statistical point of view is challenging due to the inherent complexity of the problems and the nature of the data. We believe that in order to be well-prepared against the current cyber threats, Bayesian statistics offers a wide range of flexible models that might be the key for a deeper understanding of the generative process at the basis of malicious attacks and, at the same time, for us to have predictive models able to

handle large volumes of time-evolving data. That is why in this chapter we have presented the statistical approach to cyber security anomaly detection methods, making particular emphasis on Bayesian models.

However, as remarked in Section 1.5, the methodologies described in this chapter are far from being exhaustive. In a highly connected world with cyber threats being more dangerous than ever there is a need for a thorough understanding on the computer networks' behaviour. That is why as the interest in cyber security keeps increasing we are able to find (in a frequent basis) new models that work directly along the line of some of the ones we have presented here. Moreover, alternative approaches to the ones described in this chapter have been considered and proved useful for both network modelling and anomaly detection.

We would also like to point out that, although there has been an actual increase in cyber security research from a Bayesian point of view, to the best of our knowledge, there are some areas that have not been as widely explored as others. Most of the work we have encountered corresponds to either volume-traffic or network anomaly detection. Malware related problems, like detection and classification, are still open areas of research that need to be deeply developed.

As a final comment, we would like to remark that, although not mentioned directly in each of the sections of this chapter, anomaly detection models for cyber security research require the analysis of high-volumes of data. No matter if it is for volume-traffic analysis, network modelling or malware detection and classification, all of them require handling and learning from data sets that are usually very large. This definitely plays a vital role in cyber security research, since we have always to keep in mind that while developing statistical models for this kind of problems, there is a need for algorithms able to scale well, and (preferably) able to perform in a sequential procedure as new data is observed.

Chapter 2

A Bayesian approach to malware detection and classification through n-gram profiles

Malware is a computational term that is commonly used to describe any software specifically designed to disrupt, damage or gain access to a computer system. Twenty years ago, it was remarked by McGraw and Morrisett (2000) that dealing with malicious code was a rapidly increasing problem affecting individuals, organisations and governments equally. Nowadays, in a highly connected world, the fast and accurate detection of malware is one of the major concerns of cyber security. Traditionally, the use of antivirus software has been essential in order to detect malicious code and to keep the computer systems protected. Antivirus software usually makes use of the blacklisting method, where a new program is scanned in search of signatures of known malware and if found, the program is disabled and a warning is flagged. This approach is effective for detecting known threats; however, antivirus software have been proved to be less effective with

new threats and with slight modifications made to the original code to avoid recognition (McGraw and Morrisett, 2000). Hence, it is imperative to consider and develop more flexible malware detection methods.

In recent years, machine learning and statistical approaches have been used as an alternative to the blacklisting method and several approaches have been proposed in order to detect malicious code. These detection procedures can work by either directly analysing the executable content through n-gram profiles (see *e.g.* Kolter and Maloof, 2004, 2006; Masud et al., 2007; Pektaş et al., 2011) or dynamic traces (see *e.g.* Storlie et al., 2014; Bolton and Heard, 2018; Kao et al., 2015), or by analysing the network traffic or the packet payload content (see *e.g.* Ahmed and Lhee, 2011; Prasse et al., 2017; Vidal et al., 2017). Deciding which methodology to use will certainly depend on the problem at hand. However, it seems that a content-based analysis yields a more general detection procedure, while a behavioural approach might be better for specific situations. For example, Ahmed and Lhee (2011) proposed a model for detecting executable code by analysing the packet payload; this is certainly useful for network servers that do not expect that kind of code such as shopping stores or online streaming services.

To the best of our knowledge, malware detection through n-gram profile analyses has been mainly done using discriminative binary classifiers, like support vector machine, decision trees and boosted versions of them (see *e.g.* Kolter and Maloof, 2006; Masud et al., 2007). These models have proved to be extremely accurate while keeping a low false positive rate, especially when there is a large amount of data available. Moreover, these classifiers usually outperform probabilistic models like naive Bayes, that is why in practice the latter have not been considered at all. However, probabilistic models are extremely useful because they allow us to understand the generative process of the data (and with it the opportunity to replicate it) and they can also be used for tasks such as clustering

(unsupervised learning).

It is important to remark that malware detection is not the only task required when dealing with malicious software. In order to understand their infectious process, their potential threat level and therefore, how to be well-protected against these malicious softwares, there is a need to correctly identify the family to which they belong. The accurate classification may also speed-up the process of reverse-engineering to fix computer systems that were infected as well as for developing security patches to prevent more computers to become infected. This classification task can also be done through an n-gram profile analysis since most of the binary classifiers used for malware detection can be extended to a multi-class setting. That is why we are mainly interested in directly analysing the executables' content through n-gram profiles. In this approach it is assumed that the n-grams, which in this case are contiguous sequence of n bytes, work as binary features that completely characterise each class.

In this chapter we present a novel supervised learning model for binary matrices. Our approach is based on the underlying theory of compound random measures (CoRM's) (Griffin and Leisen, 2017), which allows us to define a hierarchy across groups controlled by a beta global distribution and a beta score distribution at a group level. The beta-CoRM approach proposed here is a flexible probabilistic model for which a slice sampling method for the posterior and predictive inference is also derived. CoRM's are particularly attractive since they allow us to construct correlated measures that characterise and differentiate the groups. This is especially useful in situations where it is desired to find from a set of common features the ones that are most influential in each group. Moreover, these features are conditionally independent and, under certain prior specification, the inference can be performed independently on each feature and hence the computations can be easily parallelised. For malware detection and classification

purposes these characteristics are particularly attractive, since the main assumption is that the n-grams are independent, completely characterise each group and usually the set of n-grams is a high-dimensional object.

2.1 Related work

As detailed in the previous chapter, every instruction given to a computer can be broken down into sequences of bytes, which form the instruction's binary code. These binary sequences can be expressed in a more condensed form using the hexadecimal notation. That is, each byte is represented as the combination of two elements of the set $\{0, 1, ..., 9, A, B, ..., F\}$. For malware detection purposes, these sequences of bytes in hexadecimal notation are used to create a set of binary features that are assumed to characterise both benign and malicious programs. In order to create these features, one commonly used structure are the n-grams, which may represent an instruction, a part of one or more instructions or even a string data inside the code (Masud et al., 2007).

Using the hexadecimal representation of the binary code and n-grams is not the only approach used in order to create the binary features. There is more information from the executables that can be retrieved and used, for example, Schultz et al. (2001) used three different feature extraction processes. As explained in Kolter and Maloof (2004), the first method uses a list of Dynamically Linked Libraries (DLLs), function calls from these DLLs and the number of different system calls from within these DLLs. The second approach uses the UNIX string command, which provided the printable strings in an object or binary file. And the third method uses the hexadecimal representation of the executable content. It was further discussed that we should question the stability of the DLL names,

the function names and string features since they could be easily modified or there could not be information of them at all.

As for the hexadecimal code, a thorough study of n-grams as features is due to Kolter and Maloof (2004); since then it has been proved to be an effective approach and, as explained in Raff et al. (2016), it is particularly attractive since it can also be applied to other file formats like PDF's. However, there is an important consideration that needs to be discussed first. No matter which size of n-grams is used, the feature space is generally very large, and even for a small collection of benign and malicious executables the number of unique n-grams can reach the order of hundreds of millions, making it computationally and statistically unfeasible to consider them all. Therefore, a selection procedure is required as a first step.

Kolter and Maloof (2004) proposed to use the M most important n-grams in terms of the *information gain* (IG), which is a measure that has been used in machine learning for variable selection in text categorisation. In Yang and Pedersen (1997) a complete study on the IG and other common feature selection approaches in text categorisation was developed. As for the IG, it was showed that it favours common terms, it considers the feature absence in its calculation, and most importantly, for scenarios with an extreme dimensionality reduction (up to 98 % of the feature space) the classifiers performed the best with the IG features. This makes the IG particularly attractive for malware detection through n-gram profiles due to the presence of millions of them. The IG for the j-th n-gram is calculated as

$$IG(j) \;=\; \sum_{v_j \in \{0,1\}} \sum_{C_i} P(v_j, C_i) \log\left(\frac{P(v_j, C_i)}{P(v_j)P(C_i)}\right), \tag{2.1}$$

where v_j is the indicator of a program v having or not the j-th n-gram and C_i the i-th class. The reader can appreciate that the IG as defined in (2.1) is actually the Kullback-Leibler divergence between the joint distribution $P(v_j, C)$ and the distribution assuming independence $P(v_j)P(C)$. For its computation, Kolter and Maloof (2004) used the empirical distributions as an estimate for the required probabilities, i.e., $P(v_j, C_i)$ is the proportion of the observations in class C_i that contain (or not) the j-th feature, $P(C_i)$ is the proportion of the data belonging to class C_i and $P(v_j)$ is the proportion of the training data containing (or not) the j-the feature.

In practice, this feature extraction and selection approach requires the prior specification of the size of n-grams and the number of features, M, to be considered. In order to know the best possible combination Kolter and Maloof (2004) performed pilot studies where they considered different values for n and M. These combinations were tested in a subset of the data and compared through the performance of the classifiers used for the detection phase which included naive Bayes, decision trees, support vector machine and boosted versions of them. Furthermore, in these pilot studies the authors also considered different lengths of bytes and it was determined that the best results were obtained by using single bytes and setting $n = 4$ and $M = 500$.

Since its introduction, most of the research on malware detection through n-grams profiles has used this approach as the basis for the feature selection process. For example, some authors like Masud et al. (2007) have proposed hybrid detection models where both n-grams and DLL's are used, while others like Pektaş et al. (2011) have only used n-grams. As an example of the kind of data considered, Figure 2.1 illustrates a set of benign and malicious executables characterised by 503 binary n-grams. This data set can be found at the University of California Irvine Machine Learning repository (Rumao, 2016).

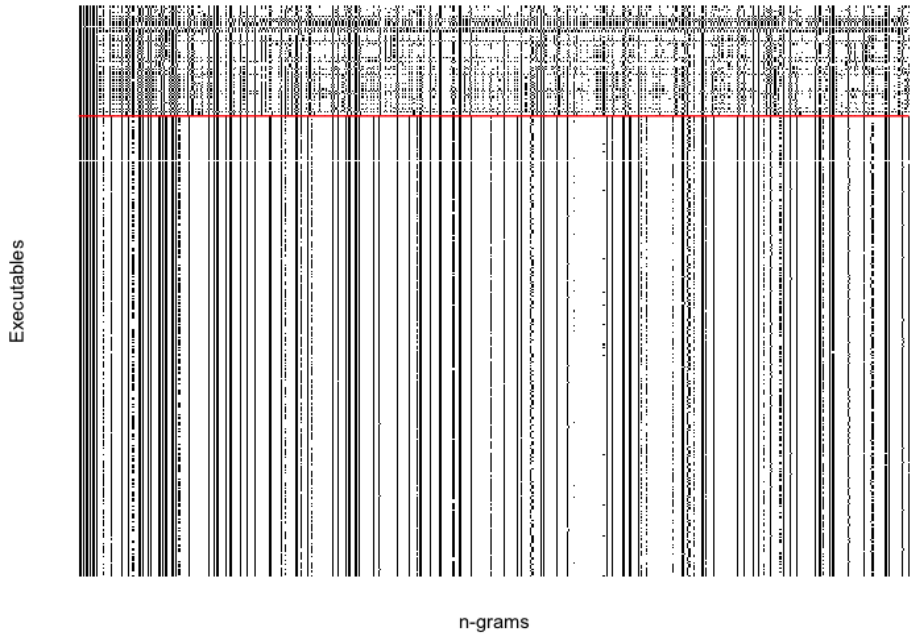


Figure 2.1: Malware detection data set with 72 benign and 301 malicious executables (separated by the red line) each one with 503 binary n-gram features.

Although widely used, the approach described by Kolter and Maloof (2004) might not be the best one overall and further improvements could be made. For example, in the experiments performed by Raff et al. (2016), it was showed that the number of unique n-grams can reach the order of billions, and hence, making it computationally expensive to obtain the IG for all of them. It was also discussed that determining the actual number of features used is also an expensive process that is not well addressed in Kolter and Maloof (2004) since they only used a subset of the data in order to fix its value.

In order to address these issues, Raff et al. (2016) proposed a three step feature selection process. The first step was to consider only the n-grams that appear in at least 1% of the observations. This step is justified by the fact that there is a need to select features that appear frequently enough to observe them in

new data. By doing so the authors achieved to reduce the number of n-grams from 36 billions to 1.6 millions. The second step was to consider a measure like the IG in order to further reduce the number of features since having millions of them is still a computational burden. In their experiments they selected the 200 thousand most important ones. Finally, the third step was to use a learning model able to do feature selection at the same time like the elastic-net regularised logistic regression model that the authors used.

As for this chapter, its main contribution is a new Bayesian supervised learning model specifically designed for binary matrices. This probabilistic approach can be used in data sets with two classes, like the malware detection data set illustrated in Figure 2.1, or in a multi-class setting for malware classification into known families, such as Trojan, backdoor, virus, etc., like the one released as part of the Microsoft Malware Classification Challenge (Ronen et al., 2018) which is thoroughly described in Section 2.4.2. For this multi-class data set, we further explored a different feature selection process which only considers the 4-grams that appear at least once in each class. This selection process drastically reduces the feature space and the computational resources required and promising results are obtained with the beta-CoRM and with other commonly used classifiers.

2.2 Discrete beta compound random measure

In this section, we present a novel Bayesian nonparametric approach to supervised learning that builds on a special type of d-dimensional vectors of completely random measures (CRM's) (Kingman, 1967), known as *compound random measures* (CoRM's) (Griffin and Leisen, 2017). To have a complete theoretical understanding of the methodology we present in this chapter, we first introduce some of the

basic concepts governing the notion of completely random measures.

2.2.1 Preliminaries

Since their introduction in Kingman (1967), completely random measures have become essential for most Bayesian nonparametric models. One of the key properties of these random measures is their almost sure discreteness, which means that their realisations are discrete with probability 1. This characteristic allows us to use CRM's to model data generated by a discrete distribution or to use them as the basic building block in mixture models. A useful representation of a CRM is due to Kingman (1967) where it was showed that if X is a CRM then,

$$X = \sum_{i=1}^{\infty} J_i \delta_{x_i}, \tag{2.2}$$

where both the locations, x_i's, and the jumps, J_i's, are random.

From a probabilistic point of view, CRM's can also be considered as a particular instance of the well-known class of stochastic processes known as Lévy processes. The reader can refer to Sato (2013) for a thorough and complete theoretical description of Lévy processes. However, for the purposes of this preliminary section, we are only interested in the fact that the distribution of any Lévy process is fully characterised by its characteristic exponent through the Lévy-Khintchine formula stated in Theorem 2.1.

Theorem 2.1 (Lévy-Khintchine formula)**.** *If X is a Lévy process, then its characteristic function, $\varphi_X(u)$ is given by*

$$\varphi_X(u) \;=\; iau + \frac{1}{2}\sigma^2 u^2 + \int_{\mathbb{R}} \left(1 - e^{iux} + iux\mathbb{1}_{|x|<1}\right) \nu(dx),$$

where ν is a measure such that $\int_{\mathbb{R}} \min(1, x^2)\nu(dx) < \infty$.

In the literature ν is known as the Lévy measure and contains crucial information of the jumps of the underlying Lévy process, such as the number of jumps, their locations, and their sizes. Moreover, (a, σ, ν) is the generating triplet and it fully characterises the Lévy process. In particular, for completely random measures we are going to be interested on Lévy processes with generating triplet $(0, 0, \nu)$ and with $\nu(-\infty, 0) = 0)$. This restriction on ν yields a Lévy process with almost surely positive jumps and therefore, nondecreasing paths, also known as a *subordinator*, which is a necessary constraint for building probability measures. Therefore, if X is a CRM, and hence a Lévy process, then it is fully characterised by the Lévy measure ν that as we have established, contains all the information about the distribution of the jumps and their locations.

With this in mind we can turn our attention to compound random measures, where the basic idea is that if we consider a CRM as in (2.2) then we can define d correlated measures by perturbing the jumps, that is, if μ_j represents the j-th random measure then,

$$\mu_j \;=\; \sum_{i=1}^{\infty} m_{ji} J_i \delta_{x_i} \qquad\qquad m_{1i}, ..., m_{di} \overset{\text{iid}}{\sim} h,$$

where the m_{ji}'s are the perturbation coefficients that identify specific features on the j-th random measure and h is the score distribution. Therefore, CoRM's are completely characterised through the distribution h and the directing Lévy

measure ν of the CRM. However, as showed in Griffin and Leisen (2017) CoRM's can also be built by specifying the d marginal random measures and by h, a certainly useful property that yields a flexible modelling. For a complete theoretical study on CoRM's and some of their applications the reader can refer to Griffin and Leisen (2017, 2018).

2.2.2 Construction

From the broad description given above, it can be easily argued that compound random measures are particularly attractive for any kind of grouped data and hence, for supervised learning. Since our interest relies on the probabilistic modelling of grouped binary matrices (e.g., Figure 2.1), a natural approach would be to consider a beta-Bernoulli model. In a Bayesian nonparametric framework this can be achieved by choosing a beta process (BP) B, as the directing CRM on a suitable space Ω. One of the main features of this stochastic process is that it concentrates the jumps in $(0, 1)$ and hence they can be used as the parameters for Bernoulli random variables.

As a completely random measure, the beta process is completely characterised by its Lévy measure

$$\nu(d\omega, dp) \quad = \quad c(\omega)p^{-1}(1-p)^{c(\omega)-1}dpB_0(d\omega),$$

where $c(\omega)$ is a concentration function and B_0 is a finite fixed measure on Ω. In practice, it is usual to consider $c(\omega) = c$, so that c is a concentration parameter. As for B_0 this measure can be continuous, discrete or a mix of both types. In the Bayesian nonparametric literature the most common choice for this base measure is to be absolutely continuous. This choice is particularly useful for

factorial models like the Indian buffet process (and related models) where the interest relies on the inference of the possible infinite number of unknown latent factors (see e.g.: Griffiths and Ghahramani (2005, 2011) and Thibaux and Jordan (2007)).

For an n-gram profile analysis for malware detection a factorial model could also be used; however, in this chapter we are interested in modelling the data directly. In order to do so, a discrete base measure on the set of unique n-grams might be a more appropriate choice. That is,

$$B_0 = \sum_{i=1}^{\infty} q_i \delta_{\omega_i}, \tag{2.3}$$

where the set of ω_i's just work as labels in order to distinguish the n-grams, contrary to factorial models where each ω_i usually represents a latent factor that needs to be inferred. This is a particularly interesting approach since the beta process will share the same atoms as B_0 with corresponding jumps p_i sampled from a beta distribution $(cq_i, c(1-q_i))$ and hence, B has the following representation

$$B = \sum_{i=1}^{\infty} p_i \delta_{\omega_i}. \tag{2.4}$$

The random measure B, as defined in (2.4) is going to be the directing CRM used for the discrete beta-CoRM model we develop in this section. With this choice, in the malware detection and classification context, the set of jumps p_i can be thought as the probability that an executable regardless of the class has the corresponding n-gram. Then, the perturbed coefficients $m_{ji}p_i$ represent the

probability of observing each n-gram for each of the d groups; therefore, we need to ensure that they remain in the interval $(0, 1)$, which can be done by selecting a score distribution defined on this interval, e.g., a beta distribution.

Hence, for the j-th group we have a marginal process given by

$$B_j = \sum_{i=1}^{\infty} m_{ji} p_i \delta_{\omega_i} \qquad\qquad m_{ji} \overset{\text{ind}}{\sim} \text{beta}(a, 1). \tag{2.5}$$

The choice of a beta (a,1) distribution for the scores was first proposed in Griffin and Leisen (2017) where a slightly different version of a beta-CoRM was considered. In its original formulation, the marginal processes of the beta-CoRM were fixed to be beta processes and using the beta (a,1) score distribution it was proved that the directing process was the sum of the original beta process and a compound Poisson process with beta-distributed jumps.

Finally, the generative process is fully described by assuming that each observation X_{kj} in group j follows a Bernoulli process with corresponding base measure B_j, so that,

$$X_{kj} = \sum_{i=1}^{\infty} x_{kji} \delta_{\omega_i} \qquad\qquad x_{kji} \sim \text{Ber}(m_{ji} p_i). \tag{2.6}$$

We believe that this Bayesian nonparametric model is an interesting approach to malware detection and classification through an n-gram analysis due to several reasons. First of all, Bayesian nonparametric models have an inherent flexibility

that allows us to control the actual number of features considered. For a beta process with a continuous base measure this is controlled through the directing Lévy measure since $\int \nu(d\omega, dp) = \infty$, which means that there are an infinite number of small jumps. For a discrete base measure this can be done by directly specifying for which locations ω_i their respective jumps q_i's are non-zero. More specifically, a CoRM approach allows us to analyse each feature independently through the perturbation coefficients which is useful in situations where we expect to differentiate groups through differences encountered in some of the features.

2.2.3 Properties

Now that the model has been fully described it is important to analyse its properties in order to fully understand the generative process and the role of the hyperparameters in the learning process. As a first step, we centre our attention on the directing discrete beta process, for which we can obtain through an elegant probabilistic argument, its expectation and its variance as first detailed in Hjort (1990). Having access to these results is vital since they provide a clear insight into the role of the jumps, q_i's, and the concentration parameter c.

Proposition 2.2. *Let B be a beta process with discrete base measure as in* (2.4) *and* (2.3) *respectively. Then*

1. $\mathbb{E}(B) = B_0$ *and*
2. $Var(B) = \dfrac{1}{c+1} \sum_{i=1}^{\infty} q_i(1 - q_i)$.

Proof. Proving these properties only requires to remark that p_i are beta distributed with parameters $(cq_i, c(1 - q_i))$. Therefore $\mathbb{E}(p_i) = q_i$, and using the

monotone convergence theorem, we get

$$\mathbb{E}(B) = \mathbb{E}\left(\sum_{i=1}^{\infty} p_i \delta_{\omega_i}\right) = \sum_{i=1}^{\infty} \mathbb{E}(p_i)\delta_{\omega_i} = \sum_{i=1}^{\infty} q_i \delta_{\omega_i} = B_0.$$

Following the same monotone convergence reasoning and using the fact that $\mathbb{E}(p_i^2) = \frac{q_i(1-q_i)}{c+1} + q_i^2$ it can be shown that

$$\mathbb{E}(B^2) = B_0^2 + \frac{1}{c+1}\sum_{i=1}^{\infty} q_i(1-q_i).$$

From which the variance can be obtained directly. □

This is a certainly useful result since it contains important information about the parameters of the directing beta process and the role they have in the generative process. For instance, q_i represents our prior knowledge on the global probabilities, and the concentration parameter, c, controls the similarity between the p_i's and the q_i's. As $c \to \infty$, $\text{Var}(p_i) \to 0$ and hence, $p_i \overset{\text{a.s.}}{\to} q_i$, therefore, large values of c should be used when there is a strong prior belief that the q_i's are good estimates of the p_i's. On the other hand, for values of c close to 0 then p_i will be either close to 1 or 0 with probabilities q_i and $(1 - q_i)$ respectively.

As for the measures B_j's, that characterise each group in this novel beta-CoRM formulation, useful properties like their expectation and their variance can also be derived following similar probabilistic arguments. Moreover, since the shared jump p_i introduces dependence between the jump heights in each measure, the covariance and the correlation at each location ω_i can also be derived. All this information is grouped in Proposition 2.3.

Proposition 2.3. *Let B a beta process defined as in Proposition 2.2 and B_j and B_k denote the j-th and the k-th measure defined as in* (2.5)*, then*

1. $\mathbb{E}(B_j|B) = \dfrac{a}{a+1}B$ *and hence* $\mathbb{E}(B_j) = \dfrac{a}{a+1}B_0$.

2. *For* $i \neq l$ *then* $Cov(B_j(d\omega_i), B_j(d\omega_l)) = 0$.

3. *For a fixed feature* ω_i

$$Var(B_j(d\omega_i)) = \left(\frac{aq_i}{a+2}\right)\left(\frac{(1-q_i)(a+1)^2 + q_i(c+1)}{(c+1)(a+1)^2}\right)$$

and for $j \neq k$,

$$Cov(B_j(d\omega_i), B_k(d\omega_i)) = \left(\frac{a}{a+1}\right)^2 \frac{q_i(1-q_i)}{c+1}$$

and

$$Corr(B_j(d\omega_i), B_k(d\omega_i)) = \frac{a(a+2)(1-q_i)}{(a+1)^2(1-q_i) + q_i(c+1)}.$$

Proof. The first property is straightforward since it follows the same reasoning as in Proposition 2.2, and the second property is a direct consequence of the scores and jumps being mutually independent. Now, for the variance,

$$\begin{aligned}
\mathrm{Var}(B_j(d\omega_i)) &= \mathbb{E}(B_j^2(d\omega_i)) - \mathbb{E}(B_j d(\omega_i))^2 \\
&= \mathbb{E}(m_{ji}^2 p_i^2) - \mathbb{E}(m_{ji} p_i)^2 \\
&= \left(\frac{a}{(a+1)^2(a+2)} + \frac{a^2}{(a+1)^2}\right)\left(\frac{q_i(1-q_i)}{c+1} + q_i^2\right) \\
&\quad - \left(\frac{a}{a+1}\right)^2 q_i^2 \\
&= \left(\frac{a}{(a+1)^2(a+2)} + \frac{a^2}{(a+1)^2}\right)\left(\frac{q_i(1-q_i)}{c+1}\right) \\
&\quad + \left(\frac{a}{(a+1)^2(a+2)}\right) q_i^2 \\
&= \left(\frac{a}{a+2}\right)\left(\frac{q_i(1-q_i)}{c+1}\right) + \left(\frac{a}{(a+1)^2(a+2)}\right) q_i^2 \\
&= \left(\frac{aq_i}{a+2}\right)\left(\frac{(1-q_i)(a+1)^2 + q_i(c+1)}{(c+1)(a+1)^2}\right).
\end{aligned}$$

From the previous expressions it can be clearly appreciated that, for a fixed feature, the variance is the same across families, that is $\mathrm{Var}(B_j(d\omega_i)) = \mathrm{Var}(B_k(d\omega_i))$, which will be useful in order to obtain the correlation of the jumps in different groups. But first, for the covariance between $B_j(d\omega_i)$ and $B_k(d\omega_i)$ we have from the first property in this Proposition that

$$\mathbb{E}(B_j(d\omega_i)) = \left(\frac{a}{a+1}\right) B_0(d\omega_i) = \left(\frac{a}{a+1}\right) q_i = \mathbb{E}(B_k(d\omega_i)),$$

and

$$\begin{aligned}
\mathbb{E}(B_j(d\omega_i)B_k(d\omega_i)) &= \mathbb{E}(m_{ji}m_{ki}p_i^2) \\
&= \left(\frac{a}{a+1}\right)^2 \mathbb{E}(p_i^2) \\
&= \left(\frac{a}{a+1}\right)^2 \left(\frac{q_i(1-q_i)}{c+1} + q_i^2\right),
\end{aligned}$$

therefore,

$$\begin{aligned}
\mathrm{Cov}(B_j(d\omega_i), B_k(d\omega_i)) &= \mathbb{E}(B_j(d\omega_i)B_k(d\omega_i)) - \mathbb{E}(B_j(d\omega_i))\mathbb{E}(B_k(d\omega_i)) \\
&= \left(\frac{a}{a+1}\right)^2 \left(\frac{q_i(1-q_i)}{c+1} + q_i^2\right) - \left(\frac{a}{a+1}\right)^2 q_i^2 \\
&= \left(\frac{a}{a+1}\right)^2 \left(\frac{q_i(1-q_i)}{c+1}\right).
\end{aligned}$$

Hence, the correlation is given by

$$\begin{aligned}
\mathrm{Corr}(B_j(d\omega_i), B_k(d\omega_i)) &= \frac{\mathrm{Cov}(B_j(d\omega_i), B_k(d\omega_i))}{\mathrm{Var}(B_j(d\omega_i))} \\
&= \left(\frac{a}{a+1}\right)^2 \left(\frac{q_i(1-q_i)}{c+1}\right) \left(\frac{a+2}{aq_i}\right) \\
&\times \left(\frac{(c+1)(a+1)^2}{(1-q_i)(a+1)^2 + q_i(c+1)}\right)
\end{aligned}$$

$$= \frac{a(a+2)(1-q_i)}{(1-q_i)(a+1)^2 + q_i(c+1)}.$$

□

From these properties we can immediately obtain the probability of an observation having the feature ω_i, which does not depend on c and is given by,

$$\mathbb{P}(x_{kji} = 1|a, q_i) = \mathbb{E}(B_j(d\omega_i)) = \frac{a}{a+1} q_i.$$

It is also interesting to notice that the covariance is the difference between the joint probability of two observations in different groups having the feature ω_i and the distribution assuming independence, that is, for the n-th and m-th observation in the j-th and k-th group respectively,

$$\begin{aligned}
\mathrm{Cov}(B_j(d\omega_i), B_k(d\omega_i)) &= \mathbb{P}(x_{nji} = 1, x_{mki} = 1|a, c, q_i) \\
&- \mathbb{P}(x_{nji} = 1|a, q_i)\mathbb{P}(x_{mki} = 1|a, q_i) \\
&= \left(\frac{a}{a+1}\right)^2 \left(\frac{q_i(1-q_i)}{c+1} + q_i^2\right) - \left(\frac{a}{a+1}\right)^2 q_i^2 \\
&= \left(\frac{a}{a+1}\right)^2 \left(\frac{cq_i^2 + q_i}{c+1}\right) - \left(\frac{a}{a+1}\right)^2 q_i^2.
\end{aligned}$$

The joint distribution can be further generalised in order to consider all groups by simply obtaining the d-th moment of a beta distribution with parameters $(cq_i, c(1-q_i))$, yielding

$$\begin{aligned}
\mathbb{P}\left(\prod_{j=1}^{d} x_{kji} = 1 \middle| a, c, q_i\right) &= \mathbb{E}\left(\prod_{j=1}^{d} B_j(d\omega_i)\right) \\
&= \left(\frac{a}{a+1}\right)^d \mathbb{E}(p_i^d)
\end{aligned}$$

$$= \left(\frac{a}{a+1}\right)^d \prod_{j=0}^{d-1} \frac{cq_i + j}{c+j}.$$

A final remark on the expression for the covariance between the jump heights is the fact that a and c have an opposite effect on it. That is, for a fixed a, the covariance decreases as c increases and, for a fixed c, the covariance decreases as a decreases.

2.2.4 Posterior inference

Due to the discrete nature of the model, the joint posterior distribution of the random measures B_j and the directing beta process B is the product of the posterior distribution of the random variables associated to each atom, that is, p_i and the set of scores $m_{1i}, ..., m_{di}$. Therefore, we can analyse the posterior density on each atom. So, if we consider the atom ω_i, the posterior density (up to proportionality) of the associated random variables is given by

$$\begin{aligned} & \left(\prod_{j=1}^{d} (p_i m_{ji})^{x_{\cdot ji}} (1 - p_i m_{ji})^{n_j - x_{\cdot ji}}\right) \left(p_i^{cq_i - 1} (1-p_i)^{c(1-q_i)-1}\right) \prod_{j=1}^{d} m_{ji}^{a-1} \\ = \; & \left(p_i^{cq_i + x_{\cdot\cdot i} - 1} (1-p_i)^{c(1-q_i)-1}\right) \prod_{j=1}^{d} m_{ji}^{x_{\cdot ji} + a - 1} (1 - p_i m_{ji})^{n_j - x_{\cdot ji}}, \end{aligned} \tag{2.7}$$

where $x_{\cdot ji} = \sum_{k=1}^{n_j} x_{kji}$ and $x_{\cdot\cdot i} = \sum_{j=1}^{d} x_{\cdot ji}$.

From (2.7) there are two important remarks for the posterior inference that need to be made. First, we can notice that the parameters in this model are non-identifiable. This can be directly seen from the likelihood for a specific feature ω_i

that is given by

$$\mathscr{L}(\{m_{ji}\}_j, p_i) = \prod_{j=1}^{d} (p_i m_{ji})^{x_{\cdot ji}} (1 - p_i m_{ji})^{n_j - x_{\cdot ji}}. \tag{2.8}$$

The non-identifiability is due to the presence of the product of the p_i and the m_{ji}'s in (2.8), since we can choose two different set of parameters that will yield the same likelihood. That is why for this model the posterior inference heavily relies on the choice of priors and hence, on the choice of c, a and the q_i's. However, it is important to remark that, although the $m'_{ji}s$ and the p_i's are useful for building a structure for modelling the variability across groups and features, we are not interested in their estimation since the classification task only relies on the joint distribution of the X's that is induced.

The second remark that needs to be made from the expression 2.7 is that due to the presence of the d terms $(1 - m_{ji}p_i)$, the joint and the conditional distributions do not have a known form. Therefore, a Gibbs sampling algorithm cannot be applied directly to this distribution. One way to address this issue is with the introduction of a set of latent variables $\{y_{kji}\}$ that allows us to define an artificial measure B_{kj} as the base measure for the k-th observation in the j-th group, that is:

$$\begin{aligned} B_{kj} &= \sum_i y_{kji} p_i \delta_{\omega_i} & y_{kji} &= \mathbb{1}(u_{kji} < m_{ji}) \sim \text{Bernoulli}(m_{ji}) \\ X_{kj} &= \sum_i x_{kji} \delta_{\omega_i} & x_{kji} &\sim \text{Ber}(y_{kji} p_i). \end{aligned} \tag{2.9}$$

This approach is based on the idea of slice sampling (Damien et al., 1999), where a set of latent variables that preserve the marginal distribution are introduced. Slice sampling schemes have become widely used in Bayesian nonparametric models since they yield efficient computational methods for the infinite dimensional objects that are at their core. The reader can refer to Griffin and Holmes (2010) and the references therein for an overview on the computational issues found in some nonparametric models and the approaches used to address them. As for the normalised completely random measures, the slice sampling technique is useful in order to introduce a random truncation point and hence, consider only a random finite number of jumps. In our case, the slice sampling approach that we propose allows us to create these infinite activity measures B_{kj}, that yield a suitable augmented likelihood from which we can recover the original one by integrating out the latent variables as we demonstrate in Lemma 2.3.

Lemma 2.4. *The discrete beta-CoRM defined by equations* (2.4)*,* (2.5) *and* (2.6) *is equivalent to the augmented model in 2.9.*

Proof. Let us consider first the augmented model. In this case it is straightforward to see that conditioned on $y_{kji} = 0$ we have that $x_{kji} \overset{\text{a.s.}}{=} 0$ and conditioned on $y_{kji} = 1$ we have that $x_{kji} \sim \text{Ber}(p_i)$. With this in mind, the augmented likelihood is

$$\prod_{j=1}^{d}\prod_{k=1}^{n_j}\left(\delta_0^{x_{kji}}\right)^{(1-y_{kji})}\left(p_i^{x_{kji}}(1-p_i)^{(1-x_{kji})}\right)^{y_{kji}},$$

and the posterior distribution is proportional with respect to the latent variables to

$$\prod_{j=1}^{d}\prod_{k=1}^{n_j}\left(\delta_0^{x_{kji}}\right)^{(1-y_{kji})}\left(p_i^{x_{kji}}(1-p_i)^{(1-x_{kji})}\right)^{y_{kji}} m_{ji}^{y_{kji}}(1-m_{ji})^{(1-y_{kji})}.$$

Integrating out the latent variables yields,

$$\prod_{j=1}^{d}\prod_{k=1}^{n_j}\delta_0^{x_{kji}}(1-m_{ji})+m_{ji}p_i^{x_{kji}}(1-p_i)^{(1-x_{kji})}. \tag{2.10}$$

Therefore, we can appreciate that the marginal posterior is the product of the mixture of a degenerate distribution and a Bernoulli distribution with corresponding weights $(1-m_{ji})$ and m_{ji}. This expression at first sight does not resemble the posterior distribution for the original beta-CoRM model. However, it is sufficient to notice that from (2.10) we obtain that $\delta_0^{x_{kji}}(1-m_{ji})+m_{ji}p_i^{x_{kji}}(1-p_i)^{(1-x_{kji})}$ is equal to

$$\begin{aligned} 1-m_{ji}+m_{ji}(1-p_i)=1-m_{ji}p_i \qquad & \text{if } x_{kji}=0 \\ m_{ji}p_i \qquad & \text{if } x_{kji}=1. \end{aligned}$$

Hence, we recover the original posterior distribution. □

Now that it has been shown that both approaches are equivalent we can obtain the complete augmented posterior (up to proportionality),

$$\begin{aligned} & \mathbb{P}(\{y_{kji}\}_{j.k},\{m_{ji}\}_j,p_i|\{x_{kji}\}_{j,k}) \\ \propto\ & \left(\prod_{j=1}^{d}\prod_{k=1}^{n_j}\left(\delta_0^{x_{kji}}\right)^{(1-y_{kji})}p_i^{x_{kji}y_{kji}}(1-p_i)^{(1-x_{kji})y_{kji}}\right) \\ \times\ & \left(\prod_{j=1}^{d}\left[m_{ji}^{a-1}\prod_{k=1}^{n_j}m_{ji}^{y_{kji}}(1-m_{ji})^{1-y_{kji}}\right]\right)\left(p_i^{cq_i-1}(1-p_i)^{c(1-q_i)-1}\right). \end{aligned} \tag{2.11}$$

From 2.11 we can immediately notice that for $j \in \{1, ..., d\}$ and $k \in \{1, ..., n_j\}$ the conditional posterior distributions are

$$
\begin{aligned}
p_i|\{x_{kji}\}, \{y_{kji}\} &\sim \text{beta}\left(\sum_{j,k} x_{kji}y_{kji} + cq_i, \sum_{j,k}(1 - x_{kji})y_{kji} + c(1 - q_i)\right) \\
m_{ji}|\{y_{kji}\} &\sim \text{beta}\left(a + \sum_{k=1}^{n_j} y_{kji}, 1 + n_j - \sum_{k=1}^{n_j} y_{kji}\right) \\
y_{kji}|x_{kji}, m_{ji}, p_i &\sim \begin{cases} \delta_1 & \text{if} \quad x_{kji} = 1 \\ \text{Ber}\left(\frac{(1-p_i)m_{ji}}{1-p_i m_{ji}}\right) & \text{if} \quad x_{kji} = 0 \end{cases}.
\end{aligned}
$$

With these conditional distributions now a standard Gibbs sampling procedure can be used in order to obtain samples from the joint posterior. Finally, once a new observation, Y, is available the classification procedure consists on computing the posterior probability of having such observation for each of the possible groups and assigning the observation to the group that has the highest probability. That is, for all j we need to obtain $\mathbb{P}(X_{n_j+1,j} = Y|X)$, which is the product of the posterior probability on each atom ω_i. Therefore, we are interested in obtaining

$$
\begin{aligned}
\mathbb{P}(x_{n_j+1,j,i} = 1|X) &= \int \mathbb{P}(x_{n_j+1,j,i} = 1|\{m_{ji}\}_j, p_i) f(\{m_{ji}\}_j, p_i|X) \\
&= \int (m_{ji}p_i) f(\{m_{ji}\}_j, p_i|X) \\
&= \mathbb{E}_{\{m_{ji}\}_j, p_i|X}(m_{ji}p_i) \\
&\approx \frac{1}{T}\sum_{t=1}^{T} m_{ji}^{(t)} p_i^{(t)}.
\end{aligned}
\tag{2.12}
$$

And by the same reasoning $\mathbb{P}(x_{n_j+1,j,i} = 0|X) \approx 1 - \frac{1}{T}\sum_{t=1}^{T} m_{ji}^{(t)} p_i^{(t)}$. Where $m_{ji}^{(t)}$ and $p_i^{(t)}$ are obtained through the Gibbs sampling procedure on the augmented model.

2.3 Synthetic data

In this section we present and describe some illustrations and results of the proposed model. In order to have a better understanding on its performance, we have divided this section into two parts by considering two different kind of data sets. First we consider a scenario where the data is composed of three non-overlapping groups, which for the purposes of this thesis represent groups characterised by disjoint sets of scores. Finally, for the second scenario we consider a more complex setting where the data is comprised of five overlapping groups. There are two main objectives of these experiments. The first one is to analyse the posterior inference and the effect the hyperparameters have on it. The second one is to analyse the predictive performance of the proposed models and compare it against other commonly used supervised learning classifiers.

2.3.1 Three non-overlapping groups

In this scenario the data is comprised of 100 observations divided into three non-overlapping groups with 33, 34 and 33 observations respectively. In order to have the desired behaviour the global probabilities p_i were sampled from a uniform distribution on the interval $(0.3, 1)$, and for each group the set of scores were sampled from uniform distributions on the intervals $(0.6, 1)$, $(0.4, 0.6)$ and $(0, 0.4)$ respectively. Figure 2.2 is the graphical representation of the binary matrix, from

which it is immediate to appreciate that there is a significant difference in the amount of features present in each group (black dots).

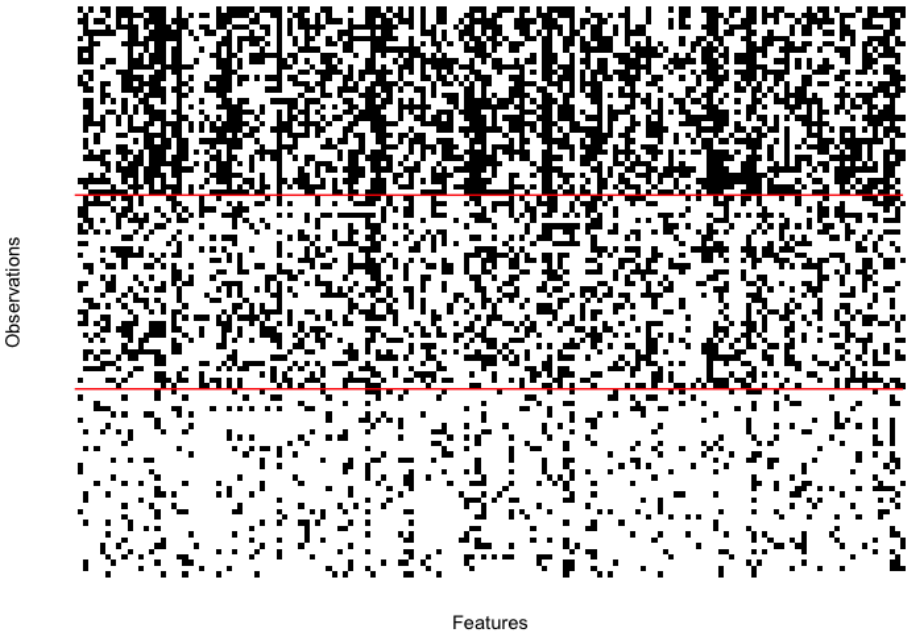


Figure 2.2: Synthetic binary matrix comprised of 150 features (columns) and 100 observations (rows) divided into three well-defined groups separated by the horizontal red lines. The dots representing the features present for each observation.

For the posterior inference, the full posterior conditional distributions obtained in Section 2.2.3 are used for a Gibbs sampling procedure which consisted of 55,000 iterations. The first 10,000 are used as the burning period and a thinning of 15 is used in the remaining 45,000 ones, yielding an effective sample size of 3,000 iterations. As for the hyperparameters, the q_i's are fixed to be the maximum across groups of the proportions of the observations having the corresponding feature ω_i, that is, $q_i = \max_j\{\sum_{k=1}^{n_j} \delta_1^{(x_{kji})}/n_j\}$. Finally, the concentration parameter c, and the parameter of the score distribution a are fixed to be equal to 1. Hence, the scores m_{ji}'s have a prior uniform distribution and the p_i's will have more flexibility in the sense that they will not be that similar to the q_i's.

With these prior specifications we obtained the posterior estimates shown in Figure 2.3, where the upper graph shows the estimates (red dots) of the global probabilities (spikes) and where the first heat map is the graphical representation of the real scores and the second heat map shows the posterior estimates. The reader can appreciate that good estimates of the true jumps, p_i's, and scores, m_{ji}'s, are produced with the prior choices made about the hyperparameters. These results can be directly observed in Figure 2.3 by first noticing in the upper graph that the posterior estimates $\widehat{p}_i$'s, represented by the red dots, are quite close to the spikes that represent the true values. Finally, it is also compelling to notice that the estimates of the scores actually reflect the non-overlapping structure created on the synthetic data.

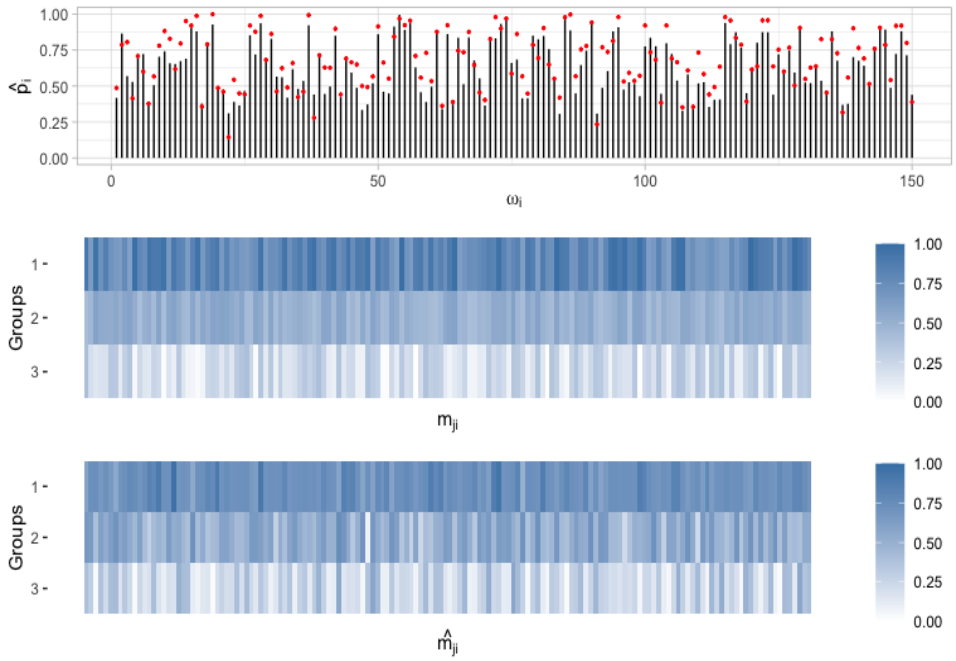
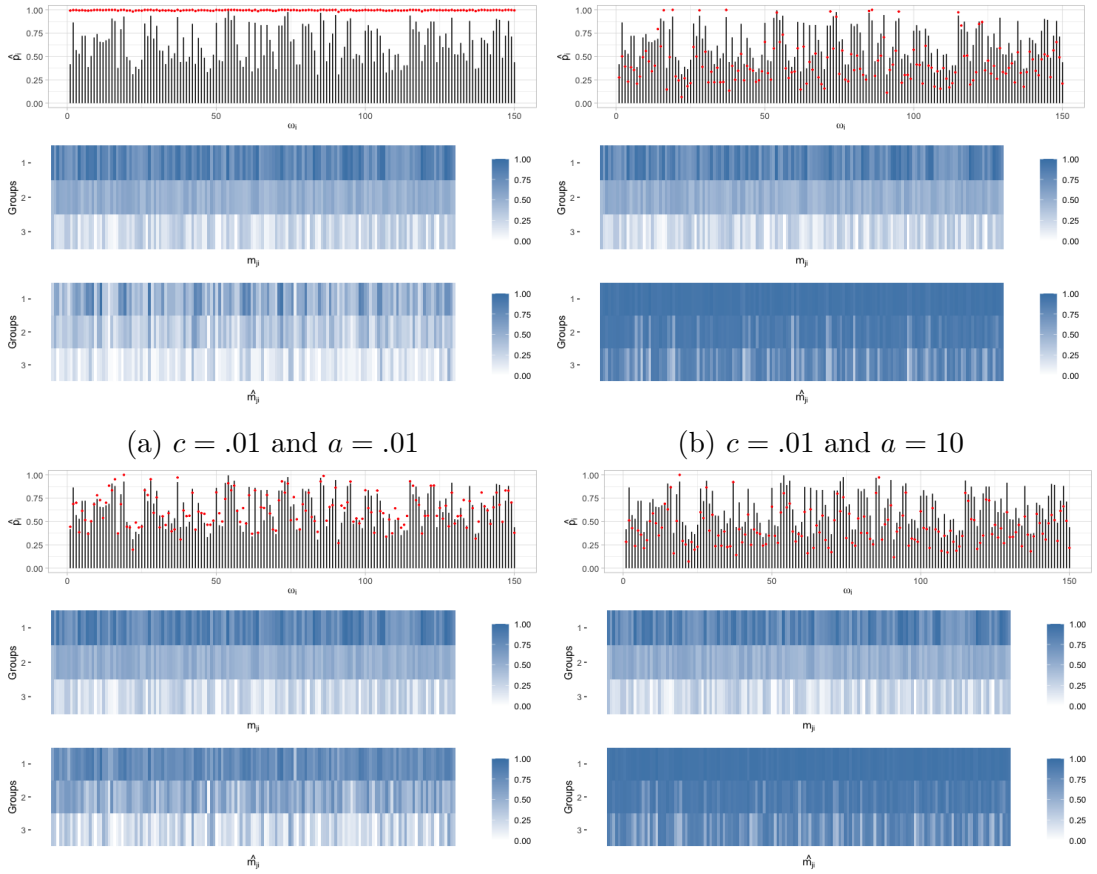


Figure 2.3: Posterior inference results of the beta-CoRM models for the data illustrated in Figure 2.2. Upper graph: real probabilities (spikes) and posterior estimates (red dots). Middle graph: real scores. Bottom graph: posterior estimates of the scores.

Of course, it is important to remember that for the beta-CoRM model the parameters are non-identifiable and hence, different sets of hyperparameters will have an impact on the posterior inference and the classification performance. In Figure 2.4 the reader can appreciate the effects of different choices of a and c, while keeping the q_i's fixed to the maximum proportion across groups. From this figure, interesting observations about the effect of c and a on the posterior inference can be made that can work as a guideline for their prior specification.



(a) $c = .01$ and $a = .01$

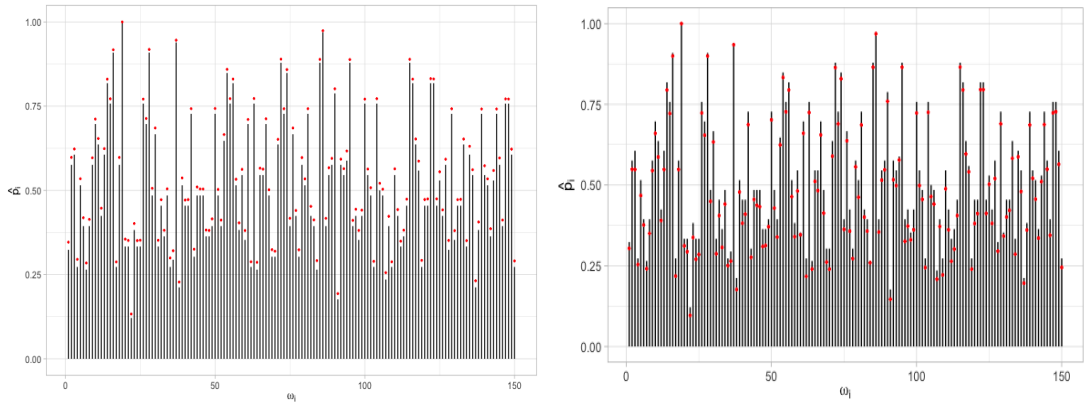
(b) $c = .01$ and $a = 10$

(c) $c = 10$ and $a = .01$

(d) $c = 10$ and $a = 10$

Figure 2.4: Posterior inference results for the beta-CoRM model with different choices of c and a for the data illustrated in Figure 2.2. For each subplot: Upper graph: real probabilities (spikes) and posterior estimates (red dots). Middle graph: real scores. Bottom graph: posterior estimates of the scores.

As a first remark, it is interesting to observe the effect that different values of c cause to the posterior estimates of the jumps of the directing beta process, $\widehat{p}_i$'s. From Figure 2.4a and Figure 2.4c, it can be clearly appreciated that for smaller values of a the impact on the $\widehat{p}_i$'s is more noticeable as c increases. Of course, it is important to remember from Proposition 2.2. that $Var(B) = \frac{1}{c+1}\sum_{i=1}^{\infty} q_i(1-q_i)$, hence as $c \to \infty$ the variance decreases to zero. Therefore, for large values of c and regardless of a the posterior estimates $\widehat{p}_i$'s will be closer to the q_i's. This is something that can be appreciated in Figure 2.5 where it can be seen that for both small a (Figure 2.5a) and large a (Figure 2.5b), the red dots representing the posterior estimates are quite close to the spikes which in this case represent the q_i's.

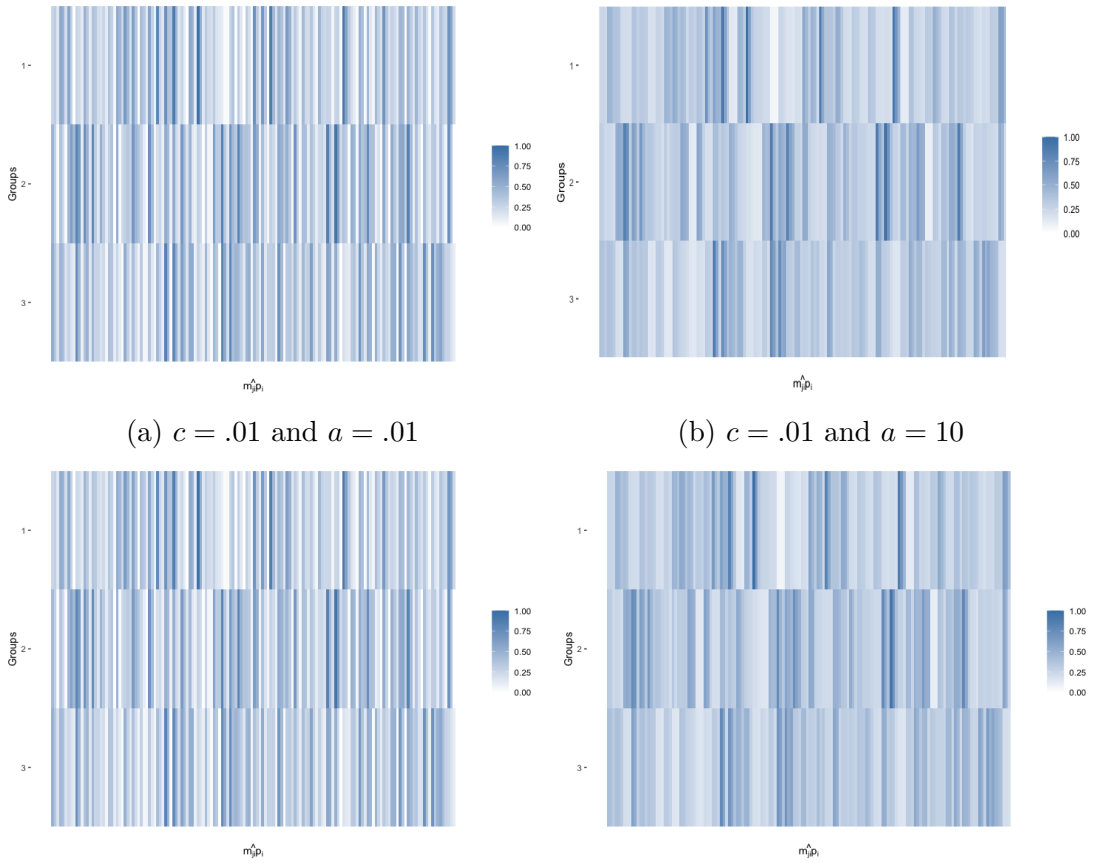


(a) $c = 100$ and $a = .01$ (b) $c = 100$ and $a = 10$

Figure 2.5: Comparison between the posterior estimates of the jumps of the directing beta process (red dots) and the q_i's (spikes) for a large value of c and for a small and a large value of a.

Now we turn our attention to the score parameter. The reader can observe that for small values of a (Figure 2.4a and Figure 2.4c) there is a clear difference in the posterior scores, $\widehat{m}_{ji}$'s, across groups that vanishes as a increases. It can be clearly appreciated that this is an effect that occurs regardless of the value of c and is due to the fact that the score distribution of the beta-CoRM model is

beta(a,1). Hence, as $a \uparrow \infty$ the scores $m_{ji} \uparrow 1$ yielding indistinguishable groups and therefore, making the beta-CoRM model not suitable for supervised learning. That is why, it can be easily argued that smaller values of a should be preferred. Otherwise, for large values of a, it would be like randomly assigning a group to each new observation. To appreciate this effect more carefully, in Figure 2.6, we illustrate the effect of a small and a large value of a for two different values of c on the posterior predictive probabilities.



(a) $c = .01$ and $a = .01$

(b) $c = .01$ and $a = 10$

(c) $c = 10$ and $a = .01$

(d) $c = 10$ and $a = 10$

Figure 2.6: Comparison of the posterior predictive probabilities between the beta-CoRM with a small a (left) and with a large a (right) for the data illustrated in Figure 2.2.

From Figure 2.6 the reader can appreciate how the plots on the left (Figure 2.6a and Figure 2.6c) that correspond to the posterior predictive probabilities for small a are more distinguishable among groups. That is, for each feature ω_i we are able to clearly tell on which of the groups it is more likely to belong. Whereas for large a this is not an easy task for most of the features since there is not a clear difference among groups.

From a practical point of view, one way to find suitable values for c and a would be to run the model for different pairs of values and perform the classification in the training/validation set or the test set (if applicable), and select the ones that have the best predictive performance with respect some classification metric such as the accuracy, which represents the percentage of correctly classified observations. For the data illustrated in Figure 2.2 we followed this procedure by testing values of a and c on a grid of the interval $[.01, 2]$, since we are especially interested in small values for these parameters, and also on the extreme cases considered throughout this section, that is, for large values of c and a. As an example, in Table 2.1 we display the classification accuracy on the training and the test set for the values of a and c that we have found to be more appealing for the purposes of this thesis.

c	a	Accuracy on training data	Accuracy on testing data
.01	.01	100	96
1	1	100	97
2	0.5	100	97
.01	10	92	90
10	.01	99	96
10	10	91	89

Table 2.1: Classification accuracy comparison (in %) of the beta-CoRM model for different values of c and a for the binary matrix illustrated in Figure 2.2.

From the results displayed in Table 2.1, it can be appreciated that the best performance was achieved when both c and a were small. Also, it seems that the size of a (being small) is more important compared to the size of c. In fact, for $c = 10$ and $a = 0.01$, we still had a decent result whereas for $a = 10$ and both $c = 10$ and $c = .01$ we saw the worst performance. However, it is compelling to notice that even for the worst case scenario the beta-CoRM model outperforms the classification performance achieved if the maximum likelihood estimators (MLE) for $\mathbb{P}(X_{kji} = 1)$, given by the cell probabilities $x_{\cdot ji}/n_j$, were used since this approach only achieves an accuracy of 82%.

In this section, we have described an intuitive way of finding optimal values of a and c; however, we acknowledge that other approaches to finding these values exist. For instance, from a Bayesian perspective, a and c can be considered random variables that need to be inferred. Naturally, this is an attractive approach that would allow us to consider different kinds of hyperpriors and with it, provide a wider range of modelling possibilities such as feature selection as developed in Chapter 3. However, it is also important to remember that for complex non-conjugate Bayesian models, adding hyperprios inevitably increases the computational cost of the MCMC, something that needs to be carefully considered depending on the problem at hand.

For the reminder of this chapter, however, we have decided to use a grid search approach to find optimal values of a and c by leveraging on the classification performance of the beta-CoRM model. By keeping fixed a and c then the posterior inference can be easily parallelised due to the discrete nature of the model, yielding a fast supervised learning model. Under a static approach to malware detection and classification, being able to provide fast and parallelisable algorithms is certainly desirable not only on account of the large number of new malicious software detected every day but also on account of the large number of

binary features that can be extracted from the binary content of the malware.

2.3.2 Five overlapping groups

For this section we considered a more complex scenario with five groups with overlapping score distributions, and tested the model's performance in different kinds of circumstances by having a balanced and imbalanced number of observations and by considering more observations than features and more features than observations. The data was generated by sampling the global probabilities from a uniform distribution on $(3, 1)$ and for each of the five groups we sampled the scores from uniform distributions on the intervals $(0.7, 1)$, $(0.5, 0.8)$, $(0.4, 0.6)$, $(0.2, 0.5)$ and $(0, 0.3)$, respectively. For both the balanced and imbalanced scenarios we tested scenarios with 150, 200 and 250 observations and 100, 200 and 300 features.

2.3.2.1 Balanced groups

In this scenario all the classes contain the same number of observations for both the training and the test set. Just as for the 3 classes example presented in Section 2.3.1, for the beta-CoRM model we first analysed the classification results for different choices and combinations of the hyperparameters, a and c. In this case, and just as expected from our previous analysis, the worst performance was obtained for large values of a and c with the best results achieved overall for values close to 1. In Table 2.2 the reader can appreciate some of the classification accuracies for values of $a \in (0, 1)$ and $c \in (0, 2)$.

Obs	Feat	$c = 1.5, a = 0.5$	$c = 0.1, a = 0.1$	$c = .5, a = .25$	$c = 1, a = 1$
150	100	82.67	83.33	84.00	81.33
150	200	90.67	89.33	90.00	92.67
150	300	96.00	95.33	96.00	97.33
200	100	80.50	78.50	79.50	81.00
200	200	93.00	93.00	93.00	92.50
200	300	97.50	97.00	97.50	97.50
250	100	82.40	80.80	81.20	82.40
250	200	93.67	93.60	93.60	94.40
250	300	96.00	95.20	95.60	96.00

Table 2.2: Classification accuracy comparison (in %) of the beta-CoRM model for different values of c and a, and varying number of observations and features.

From Table 2.2 it can be seen that although the performance is similar for the different values of a and c, the best results overall are obtained with $c = 1$ and $a = 1$. That is why these values have been chosen in order to compare the beta-CoRM model against the MLE approach and to some well-known supervised learning models. For the latter it is important to remark that we have chosen binary classifiers that can be easily extended to a multi-class setting and that have already been used for detecting and classifying malware. That is why the algorithms considered are: *naive Bayes* (see *e.g.* Mitchell, 1997); *multinomial logistic regression* (see *e.g.* Hosmer et al., 2013); and *decision trees* (see *e.g.* Mitchell, 1997) with their *adaptive* (Freund and Schapire, 1996), *gradient* (Friedman, 2001) and *extreme gradient* (Chen and Guestrin, 2016) boosted versions, and for their implementation we respectively used the R packages: *e1071* (Meyer et al., 2021), *nnet* (Venables and Ripley, 2002), *tree* (Ripley, 2021), *adabag* (Alfaro et al., 2013), *gbm* (Greenwell et al., 2020) and *xgboost* (Chen et al., 2021).

The accuracy performance of the discrete beta-CoRM (b-CoRM) with parameters $a = 1, c = 1$, MLE ,naive Bayes (nB), multinomial logistic regression (ML) and decision trees (dT) with their adaptive (AB), gradient (GB) and extreme gradient (XGB) boosted versions can be seen and compared in Table 2.3

where it can be immediately appreciated that the proposed model has the best performance for all the considered scenarios.

Obs	Feat	b-CoRM	MLE	nB	dT	AB	GB	XGB	ML
150	100	81.33	74.00	66.00	49.33	63.33	66.00	68.00	51.33
150	200	92.67	78.00	70.00	44.67	70.67	71.33	69.33	46.67
150	300	97.33	71.33	71.33	50.67	74.00	69.33	70.00	50.67
200	100	81.00	72.5	66.50	44.50	68.00	65.00	69.00	50.50
200	200	92.50	79.00	83.00	48.50	76.50	75.50	73.50	50.50
200	300	97.50	84.00	79.00	56.00	75.00	79.00	81.00	55.00
250	100	82.40	75.60	77.60	40.40	66.00	68.00	70.00	48.00
250	200	94.40	90.40	85.60	52.00	81.60	80.80	83.20	61.60
250	300	96.00	85.20	83.20	50.40	76.80	75.60	82.80	58.80

Table 2.3: Classification accuracy comparison (in %) of the beta-CoRM model against other commonly used supervised learning algorithms for the balanced group scenario with varying number of observations and features.

2.3.2.2 Imbalanced groups

For the imbalanced scenario the middle group had twenty observations for all the cases contrary to the other groups that had more as we increased the number of total observations. We started with 150 and increased to 200 and 250 total observations, with respective partitions $(30, 40, 20, 30, 30)$, $(45, 50, 20, 45, 40)$ and $(55, 70, 20, 55, 50)$. In our analysis we first studied the classification performance of the beta-CoRM for varying c and a. Just as for the balanced scenario, the best overall results were achieved for values close to 1. In Table 2.4 the reader can appreciate some of the classification accuracies for values of $a \in (0, 1)$ and $c \in (0, 2)$.

Obs	Feat	$c = 1.5, a = 0.5$	$c = 0.1, a = 0.1$	$c = .5, a = .25$	$c = 1, a = 1$
150	100	85,33	82.00	83.33	86.00
150	200	92.67	92.67	92.67	93.33
150	300	94.00	93.33	94.00	94.00
200	100	91.00	89.00	90.00	90.50
200	200	94.50	94.00	94.50	95.50
200	300	97.00	97.00	97.50	97.50
250	100	88.80	87.20	87.20	89.20
250	200	93.20	93.20	93.60	92.80
250	300	96.40	95.20	95.60	97.20

Table 2.4: Classification accuracy comparison (in %) of the beta-CoRM model for different values of c and a, and varying number of observations and features.

Finally the classification accuracy of all the classifiers, with the beta-CoRM with hyperparameters $a = 1, c = 1$ can be seen and compared in Table 2.5. It can be immediately appreciated that just as for the balanced scenario, the beta-CoRM model had the best performance for all the scenarios considered.

Obs	Feat	b-CoRM	MLE	nB	dT	AB	GB	XGB	ML
150	100	86.00	73.33	64.67	36.67	68.67	61.33	66.67	38.67
150	200	93.33	74.00	79.33	53.33	72.67	72.00	75.33	52.00
150	300	94.00	66.00	74.67	60.00	76.67	78.00	78.67	48.67
200	100	90.50	84.00	75.50	59.00	76.00	76.50	80.00	57.50
200	200	95.50	88.50	83.50	57.00	78.50	76.50	79.00	58.00
200	300	97.50	79.50	78.50	57.50	82.00	79.50	80.50	56.50
250	100	89.20	84.40	69.20	68.40	83.60	81.20	82.00	66.40
250	200	92.80	82.00	87.20	64.40	82.80	77.20	82.80	54.40
250	300	97.20	86.00	82.40	60.80	84.80	82.40	85.20	61.60

Table 2.5: Classification accuracy comparison (in %) of the beta-CoRM model against other commonly used supervised learning algorithms for the imbalanced group scenario with varying number of observations and features.

2.4 Real-data applications

In this section we present results for both the malware detection and classification into known families problems. It is important to remark that we used two different data sets obtained from different sources. However, both the detection and classification could be done on a suitable data set that contained both benign and malicious executables for which we knew their corresponding family.

2.4.1 Malware detection

For the malware detection task we used the data set found in the University of California Irvine Machine Learning repository by Rumao (2016) (Figure 2.1). This data set is originally comprised of 72 benign and 301 malicious executable programs and 531 features comprised of 503 different n-grams and 28 DDL features. For our purposes we only considered the 503 n-grams which were obtained according to the author, following the assumptions and procedures described by Kolter and Maloof (2004), so we believe that $n = 4$ although it is not directly specified.

For this data set we can appreciate a clear difference between the two classes, just as for the first synthetic data considered. In order to perform a discrimination analysis we split the binary matrix into a training and a test set. This procedure was done several times to analyse if there was a negative impact in the performance when the number of observations in the training set decreased at the point that there were more observations in the test set. First, we (randomly) selected 90 percent of the data to be in the training set and continued to decrease this percentage by 10 points each time until we reached the scenario where 30%

of the data was used as the training set and the remaining 70% as the test set. The performance of the beta-CoRM with parameters $a = c = 1$ and the rest of classifiers can be compared in Table 2.6.

Ratio	b-CoRM	MLE	nBayes	Tree	ABoost	Gboost	XGBoost
90-10	100	100	100	100	100	100	100
80-20	98.67	97.33	76.00	98.67	98.67	98.67	98.67
70-30	99.11	98.21	77.68	99.11	99.11	99.11	99.11
60-40	99.33	97.32	79.19	97.99	99.33	99.33	99.33
50-50	98.93	97.37	80.75	98.93	99.47	98.40	99.47
40-60	99.11	96.49	80.36	98.66	99.55	98.21	99.11
30-70	99.23	96.55	80.84	97.32	99.62	99.23	98.47

Table 2.6: Classification accuracy comparison (in %) of the beta-CoRM model against commonly used supervised learning algorithms for the malware data set illustrated in Figure 2.1, with a decreasing number of observations in the training set.

Interesting conclusions can be obtained from these results. First of all, it is clear how the beta-CoRM model has an impressive performance for this data set. Moreover, we can also observe that it also outperforms the other probabilistic classifier, naive Bayes, in all scenarios, which is due to the fact that as soon as we reduced the number of observations in the training set, naive Bayes classified all observations as malware yielding a poor accuracy. Finally, and contrary to the synthetic data used in the previous section, there are no results for the logistic model, since in all the scenarios considered the training data contained several features with no variation which resulted in an error while fitting the generalised linear model function in R.

2.4.2 Malware classification

The data used for the classification task was released as part of the Microsoft Malware Classification Challenge (Ronen et al., 2018) hosted at Kaggle in 2015. The training data set is composed of almost 11,000 malware representing a mix of nine different families. For each of these malicious executables the data provides the label representing the true family, a file with the hexadecimal representation of the binary code and a metadata manifest containing information extracted from the binary. Table 2.7 provides the name of the families considered, the number of total malware in each one of them as well as their type.

Family	No. of train samples	Type
Ramniit	1541	Worm
Lollipop	2478	Adware
Kelihos_ver3	2942	Backdoor
Vundo	475	Trojan
Simda	42	Backdoor
Tracur	751	TrojanDownloader
Kelihos_ver1	398	Backdoor
Obfuscator.ACY	1228	Obfuscated malware
Gatak	1013	Backdoor

Table 2.7: Summary of the Microsoft Malware Classification Challenge data set, containing for each of the nine families the number of observations and their type.

Since one of the families only contains 42 observations we decided to work with a random sample of 842 malware (100 observations per group plus the 42 of family 5). These malware were further split into a training set comprised of 590 observations and a testing set of 292 elements. Following the procedure described in Section 2.1, we obtained the unique 4-grams that appeared at least once in each family, yielding a total of 826 unique features. The graphical representation of the binary matrix can be seen in Figure 2.7. In this case, and contrary to the malware detection application and synthetic data examples, we can observe that

for some of the families there is not a well-defined structure, for example, from top to bottom families one, six and nine. Finally, we can also appreciate that there are families that have

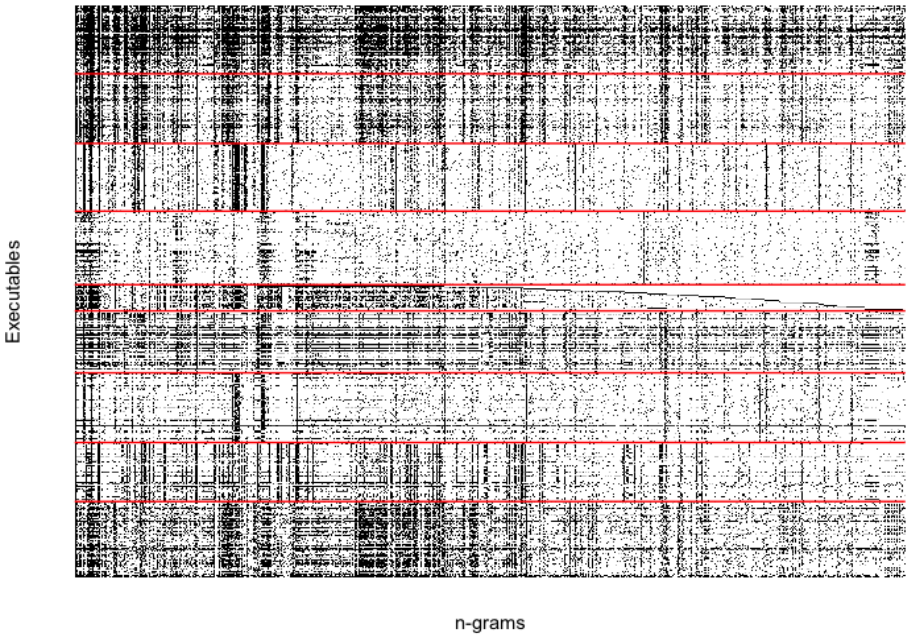


Figure 2.7: Graphical representation of the data set comprised of 590 malware (rows) of 9 families separated with the solid horizontal lines and 826 4-grams (columns). The dots represent the features that appear in each executable.

As for the classification is concerned, we used the beta-CoRM model with parameters $a = c = 1$, the MLE approach and the rest of supervised learning algorithms considered so far. Table 2.8 shows the classification performance for all the models.

b-CoRM	MLE	nBayes	Tree	ABoost	GBoost	XGBoost	ML
80.95	79.76	77.38	80.15	94.05	91.67	91.27	84.92

Table 2.8: Classification accuracy comparison (in %) for the beta-CoRM model against commonly used supervised learning algorithms for the malware data set illustrated in Figure 2.7.

In this case, and contrary to the malware detection task, the boosted algorithms outperformed the beta-CoRM model proposed; nevertheless, promising results were obtained and interesting conclusions can be made about the feature extraction and selection process and the role of the hyperparameters for further work and research.

2.5 Concluding remarks

In this chapter we have presented a novel Bayesian supervised learning model for grouped binary matrices known as beta-CoRM and a slice sampling method that allows an efficient sampling from the posterior distribution. This model is built up on the underlying theory of compound random measures that belongs to a wider class of Bayesian nonparameteric discrete priors. From the synthetic examples and the real-data sets we were able to see that for binary matrices with a well-defined group structure like the synthetic data and the malware detection set, the beta-CoRM achieved a comparable performance with the other methods. However, for the classification task, where there was not a clear difference among groups, the beta-CoRM model was outperformed and with the adaptive algorithms achieving the best performance.

It is also worth mentioning some of the advantages of the whole methodology. First, a drastic feature dimensionality reduction approach was used and outstanding results were obtained in all scenarios and data sets considered. For the beta-CoRM it is also important to point out that not allowing the concentration and score parameters to be random allowed us to have a model easily parallelisable, and hence, the inference could be performed in each atom ω_i separately, which makes it suitable for data with a large number of features involved.

Chapter 3

Beta compound random measure with feature selection

The beta-CoRM model described in the previous chapter is a novel supervised learning model for which promising results were obtained in synthetic and real data. One key advantage of this model is the fact that we kept the hyperparameters not random and hence, a parallelisable inferential procedure could be performed. However, as it is described in this chapter, a straightforward generalisation can be derived by allowing the score parameter a to be random. By doing so, we are not only providing a deeper understanding of the behaviour across groups for each feature, but this also allows us to introduce a feature selection process within the posterior learning. Something that will be particularly useful for situations where the feature space is a high-dimensional object and there is a need to learn which are the best discriminative features.

As described in the previous chapter, the idea of a feature selection step within the inferential process is something that had already been discussed for malware detection and classification through an n-gram analysis (Raff et al., 2016). This is something particularly crucial since even small data sets can yield a large amount of binary n-grams and there is an obvious need of learning and hence, considering, the features with the best predictive capabilities.

3.1 Generalised beta-CoRM

The generalised beta-CoRM approach arises naturally by noting that the density of the beta(a,1) score distribution can be written as

$$\begin{aligned} ax^{a-1} &= x_0^a \frac{ax^{a-1}}{x_0^a}\mathbb{1}_{(0,x_0)}(x) + (1-x_0^a)\frac{ax^{a-1}}{1-x_0^a}\mathbb{1}_{(x_0,1)}(x) \\ &= (1-w)f(x) + wg(x), \end{aligned}$$

where $f(x)$ and $g(x)$ are truncated beta distributions on $(0, x_0)$ and $(x_0, 1)$ respectively for small x_0 and with w the probability of "including" a variable. This representation mimics the form of a spike-and-slab prior (see *e.g.* Mitchell and Beauchamp, 1988; Ishwaran and Rao, 2005) with $g(x)$ the slab distribution with cumulative distribution function (cdf) given by

$$G(x) = \frac{x^a - x_0^a}{1-x_0^a}.$$

Since $a = \log(1-w)/\log(x_0)$ as $w \downarrow 0$, $a \downarrow 0$ and so we are interested in the limit of the above cdf as $a \downarrow 0$, which, using L'Hôpital's rule, is

$$\lim_{a\downarrow 0} G(x) = \frac{\log(x_0) - \log(x)}{\log(x_0)},$$

and the corresponding density $g(x)$ is

$$g(x) \;=\; \frac{1}{\log(1/x_0)}\frac{1}{x}, \qquad x > x_0.$$

Therefore, we can understand the prior distribution as a spike-and-slab prior where a controls the size of the spike and, if a is close to zero, the pdf of the slab is approximately $g(x)$. In the beta-CoRM, the beta$(a, 1)$ random variables moderate p_i, and so, for small a, the prior expects some of the products $m_{ji}p_i$ to be close to zero with w controlling the proportion close to "zero".

With this interpretation in mind, the generalised beta-CoRM is fully described by giving to each feature ω_i an individual score distribution beta(a_i,1), that is,

$$\begin{aligned} p_i &\sim \text{beta}(cq_i, c(1-q_i)) && i \in \{1, ..., J\} \\ m_{ji} &\sim \text{beta}(a_i, 1) && j \in \{1, ..., d\} \\ x_{kji} &\sim \text{Bernoulli}(m_{ji}p_i) && k \in \{1, ..., n_j\}, \end{aligned} \tag{3.1}$$

with all the a_i's having a common distribution. Following the *spike-and-slab* interpretation of the score distribution then these score parameters can be used for a feature selection procedure. For this, we can recall that a_i moderates the corresponding weight p_i and for small a_i the prior expects that for some of the groups the probability of observing the associated feature ω_i to be close to zero and hence, indicating a feature worth retaining in order to discriminate new observations. More intuitively, as a_i gets smaller the spike gets bigger yielding a larger number of m_{ji}'s to be close to zero with only a small number of scores sampled from the slab distribution representing the groups for which the feature ω_i is important.

3.2 Posterior inference

For this model, the slice sampling approach developed in Section 2.2.3 is still valid, therefore, we only need to provide the details for the posterior inference on the score parameters, a_i's. In order to do so, we first notice that

$$f(\{m_{ji}\}_j|a_i) \;=\; \prod_{j=1}^{d} a_i m_{ji}^{a_i-1} \overset{\text{w.r.t. } a_i}{\propto} a_i^d \exp\left(a_i \sum_{j=1}^{d} \log(m_{ji})\right),$$

is the kernel of a gamma distribution with parameters $(d+1, -\sum_j \log(m_{ji}))$. Therefore, a gamma prior could be used on each a_i to have a conjugate model, that is, if $a_i \sim \text{gamma}(\alpha, \beta)$ then the posterior is $\text{gamma}(\alpha + d + 1, \beta - \sum_j \log(m_{ji}))$. Finally, and in order to have a full Bayesian hierarchical model, we assign to the hyperparameters α and β a vague gamma prior, that is, gamma (0.001, 0.001). Hence, the joint density of the M score parameters is

$$f(\{a_i\}_{i=1}^M|\beta,\alpha) \;=\; \prod_{i=1}^{M} \frac{\beta^\alpha a_i^{\alpha-1}}{\Gamma(\alpha)} \exp\left(-a_i\beta\right), \tag{3.2}$$

Expression (3.2) is proportional with respect to β to

$$\beta^{\alpha M} \exp\left(-\beta \sum_{i=1}^{M} a_i\right),$$

which is the kernel of a gamma distribution. Hence, the posterior is also a gamma with updated parameters $(\alpha M + 0.001, 0.001 + \sum_{i=1}^M a_i)$. As for α, the posterior

distribution is (up to proportionality)

$$\begin{aligned} f(\alpha|\{a_i\},\beta) &\propto \left(\frac{1}{\Gamma(\alpha)^M}\prod_{i=1}^{M}(\beta a_i)^{\alpha}\right)[\alpha^{.001-1}\exp(-.001\alpha)] \\ &= \frac{\alpha^{.001-1}\exp(-.001\alpha)}{\Gamma(\alpha)^M}\prod_{i=1}^{M}\exp\left[\alpha\log(\beta a_i)\right] \\ &= \frac{\alpha^{.001-1}}{\Gamma(\alpha)^M}\exp\left[-\alpha\left(.001-\sum_{i=1}^{M}\log(\beta a_i)\right)\right]. \end{aligned}$$

In order to update α, we use a Metropolis-Hastings step since the posterior is not known. This can be done with the adaptive random walk Metropolis Hasting (Atchadé and Rosenthal, 2005) on $\phi = \log\alpha$. So, under this transformation, it is straightforward to see that

$$f(\phi|\{a_i\},\beta) \propto \frac{\exp(\phi)^{.001}}{\Gamma(\exp(\phi))^M}\exp\left[-\exp(\phi)\left(.001-\sum_{i=1}^{M}\log(\beta a_i)\right)\right].$$

Since $\phi \in \mathbb{R}$ we use $q_\sigma(x,y) = N(x,\sigma^2)$ as the proposal distribution and accept the move with probability

$$\gamma(x_n, y_{n+1}) = \min\left\{1, \frac{\pi(y_{n+1})}{\pi(x_n)}\right\},$$

where π is our target distribution for which we do not need the normalising constant due to the ratio $\pi(y)/\pi(x)$. Following the steps and notation proposed by Atchadé and Rosenthal (2005), we need to fix constants ϵ_1 and A_1, such that, $0 < \epsilon_1 < A_1$ in order to define the region $\Theta = \{\sigma : \epsilon_1 \leq \sigma \leq A_1\}$ for which we assume there is an optimum σ_{opt} such that the asymptotic acceptance rate is $\tau(\sigma_{opt}) = \bar{\tau} = .234$. In order to guarantee that at each step $\sigma_n \in \Theta$ there is a

need to define a control function p such that,

$$p(\sigma) = \begin{cases} \sigma & \text{if } \sigma \in \Theta \\ \epsilon_1 & \text{if } \sigma < \epsilon_1 \\ A_1 & \text{if } \sigma > A_1. \end{cases}$$

With this in mind we can use the following adaptive random walk Metropolis-Hastings algorithm

1. Start at some $\phi_0 = \log \alpha_0$ and $\sigma_0 \in \Theta$.
2. At time $n + 1$,
   (a) propose $Y_{n+1} \sim N(X_n, \sigma_n^2)$ and sample $U \sim \text{Unif}(0, 1)$,
   (b) if $U \leq \gamma(X_n, Y_{n+1})$ then $X_{n+1} = Y_{n+1}$. Otherwise, $X_{n+1} = X_n$.
   (c) $\sigma_{n+1} = p(\sigma_n + \psi_n(\gamma(X_n, Y_{n+1}) - \bar{\tau}))$ with $\psi_n = \sigma_0/n$.

Once posterior estimates of the a_i's are obtained, a feature selection procedure can be used. In order to know which features are the "best" ones, we can recall from the *spike-and-slab* interpretation of the score distribution (Section 3.1) that features with small score parameters a_i's should be preferred. From a practical point of view a natural approach to find the optimum threshold T_{opt} would be to define a grid Δ on the interval $(\min\{\widehat{a}_i\}, \max\{\widehat{a}_i\})$ and then for every $T \in \Delta$:

1. Select all the features ω_i such that $a_i \leq T$.
2. Using the associated posterior probabilities of the chosen features, that is, $\widehat{m_{ji} p_i}$, proceed to the classification of the malware in the validation set.

Once the accuracy for each $T \in \Delta$ has been obtained we set the optimal value, T_{opt}, to the threshold T for which the best classification performance was observed. In the examples considered in the following sections, this procedure is realised in the test set since we have access to the real labels. Of course, it is important to acknowledge the advantage of optimising over thresholds when the ground truth is available. That is why in real life situations when the ground truth is not known, finding the optimum threshold can be achieved using a cross-validation procedure which might decrease the classification performance.

3.3 Synthetic data

The synthetic data sets used to test on a first instance the generalised beta-CoRM model are the same data sets used on the previous chapter, that is, we first consider a scenario with three non-overlapping groups and then we move to a more complex scenario with both balanced and imbalanced data sets composed of five overlapping groups (for a thorough description on how the data sets were generated the reader should refer to Section 2.3).

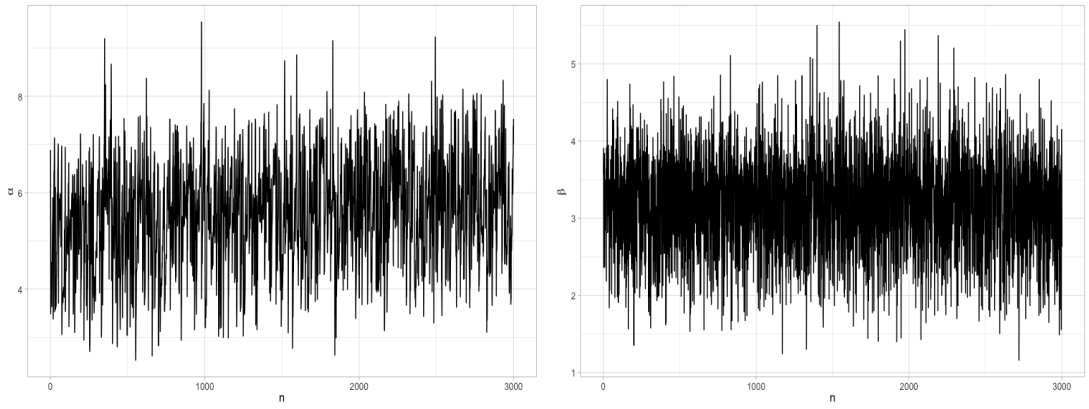
Of course, we acknowledge that different data sets "better suited" to the modelling characteristics of the generalised beta-CoRM model could have been generated. However, for the purposes of these exploratory set of experiments we believe that using the same synthetic data sets represents a more interesting exercise in order to appreciate how the generalised beta-CoRM is able to improve on the performance of the beta-CoRM model even on data sets where in principle, we could expect most of the features to be equally important. In this direction, and as we shall see in the following sections, it is certainly interesting to notice that for all the scenarios considered the generalised beta-CoRM methodology is

able to discover a set of features that should not be included in the classification procedure to achieve the best predictive performance.

3.3.1 Three non-overlapping groups

For the generalised version of the beta-CoRM we fixed $c = 1$ and the q_i's to the maximum proportion across groups of observations having the corresponding feature, just as we did for the beta-CoRM. Now, for the Metropolis-Hasting step, we started with initial values $\alpha = \beta = 1$, which yield an initial prior mean of 1 for the score parameters a_i. We believe this is a reasonable choice since we have seen how the "best" features tend to have a score parameter below 1 and on the other hand, features with a smaller discriminative power have a score parameter above 1. For the adaptive step, we started with $\sigma = 10$ to allow the sampler to explore a large region at the beginning and we fixed $\Theta = (0.0001, 1000)$ which should give σ a reasonably large region to adapt to the changes of the MCMC, which we run using the same burning period (10,000), total number of simulations (55,000) and thinning (15) as for the beta-CoRM.

The reader can observe the thinned dynamics of the hyperparameters in Figure 3.1 and the posterior estimates of the score parameters $\widehat{a_i}$ in Figure 3.2.



(a) Sample from the posterior of α. (b) Sample from the posterior of β.

Figure 3.1: Posterior samples of the hyperparameters α and β corresponding to the distribution of the score parameters.

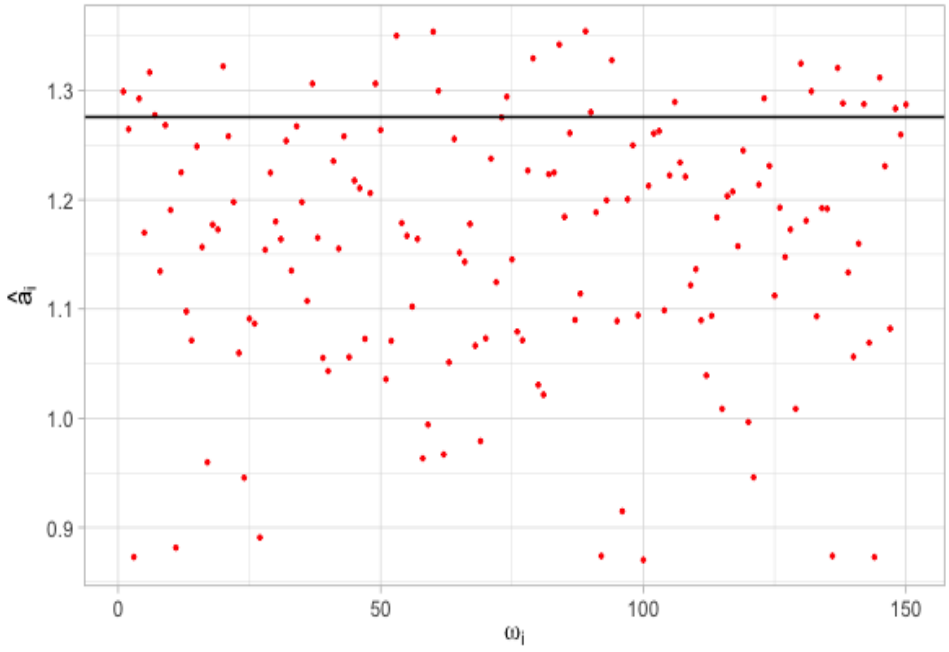


Figure 3.2: Posterior estimates of the respective score parameter a_i for each feature, along with the optimal threshold represented by the black horizontal line.

Just as expected, the reader can observe that there does not seem to be a big difference among the posterior estimates of the score parameters. However, we can still proceed to check if there is an optimum threshold that maximises the classification accuracy. In order to do so, we notice that the $\min\{\widehat{a}_i\} = 0.8702871$ and $\max\{\widehat{a}_i\} = 1.354212$, hence, in the interval $\Delta = (.87, 1.4)$ we create a grid with step .001 and perform the classification procedure as described in Section 3.2. That is, for each threshold T in the grid, we apply the classification procedure only considering the features whose $a_i \leq T$. The results of this procedure are shown in Figure 3.3, where the reader can appreciate that a maximum classification accuracy of 97% is achieved when we consider $T_{opt} = 1.275738$ for a total number of 124 features used.

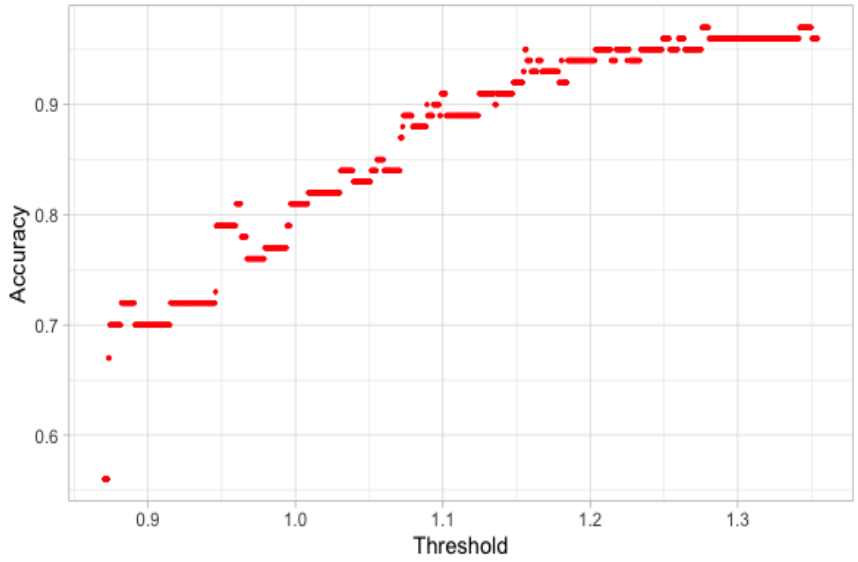


Figure 3.3: Accuracy performance for different threshold levels.

It is certainly compelling to notice that even though the data was generated without any predominant features, with the generalised beta-CoRM we were still able to find a subset of the features that were not worth retaining in order to achieve the best classification performance.

3.3.2 Five overlapping groups

Just as for the three non-overlapping groups, for the following sections we used the same data sets as in the previous chapter, that is, the five balanced and imbalanced overlapping groups. However, for these scenarios we only compare the beta-CoRM against its generalised version, since it was already the best classifier. It is important to mention that hereby, the generalised beta-CoRM will have concentration parameter $c = 1$, the q_i's will be fixed the same as before and the same initial values for the Metropolis-Hastings step will be used.

3.3.2.1 Balanced groups

For the balanced scenario we can clearly appreciate in Table 3.1 how the generalised version of the beta-CoRM outperforms the beta-CoRM model in all scenarios. Furthermore, for all the cases considered, with the generalised beta-CoRM model we were able to find some features not worth retaining without regard of the similar posterior estimates of the score parameters.

Observations	Features	beta-CoRM	generalised beta-CoRM
150	100	80.67	84.67
150	200	92.67	92.67
150	300	97.33	98.00
200	100	81.00	81.00
200	200	92.50	93.00
200	300	97.50	98.00
250	100	82.40	83.20
250	200	94.40	94.80
250	300	96.00	96.40

Table 3.1: Classification accuracy comparison (in %) of the beta-CoRM model against its generalised version for the balanced group scenario with varying number of observations and features.

3.3.2.2 Imbalanced groups

For the imbalanced groups scenario, the generalised beta-CoRM achieved, as expected, also a better accuracy in all cases compared to the beta-CoRM as seen in Table 3.2. And just as the balanced group there was not a clear difference among the posterior estimates of the a_i's; however, we were still able to find an optimal subset of the features yielding the best performance of the model.

Observations	Features	beta-CoRM	generalised beta-CoRM
150	100	86.00	88.67
150	200	93.33	94.00
150	300	94.00	95.33
200	100	90.50	91.00
200	200	95.50	95.50
200	300	97.50	98.00
250	100	89.20	90.40
250	200	92.80	92.80
250	300	97.20	97.20

Table 3.2: Classification accuracy comparison (in %) of the beta-CoRM against its generalised version for the imbalanced group scenario with varying number of observations and features.

3.4 Real-data applications

For this section the data sets used were the same as in Chapter 2. However, in these cases and contrary to the synthetic data, we compared the performance of the generalised beta-CoRM against all the classifiers. This is quite important, since as seen in the previous chapter, the beta-CoRM model was outperformed by some of them.

3.4.1 Malware detection

For the malware detection task, we proceeded as before. That is, we split the data into training and test set several times. As seen in the previous chapter, the naive Bayes classifier had trouble classifying the new observations as we reduced the number of observations in the training set. That is why its classification accuracy is not presented in Table 3.3, where we can clearly appreciate how the generalised beta-CoRM (gen. b-CoRM) outperformed the beta-CoRM (b-CoRM) model, the MLE approach ,the decision tree and its adaptive boosted (AB), gradient boosted (GB) and extreme gradient boosted (XGB) versions.

Ratio	b-CoRM	gen. b-CoRM	MLE	Tree	AB	GB	XGB
90-10	100	100	100	100	100	100	100
80-20	98.67	100	97.33	98.67	98.67	98.67	98.67
70-30	99.11	100	98.21	99.11	99.11	99.11	99.11
60-40	99.33	100	97.32	97.99	99.33	99.33	99.33
50-50	98.93	99.47	97.37	98.93	99.47	98.40	99.47
40-60	99.11	99.55	96.49	98.66	99.55	98.21	99.11
30-70	99.23	99.62	96.55	97.32	99.62	99.23	98.47

Table 3.3: Classification accuracy comparison (in %) of the beta-CoRM models against commonly used supervised learning algorithms for the malware data set illustrated in Figure 2.1, with a decreasing number of observations in the training set.

For this data set and contrary to the synthetic data, interesting results about the feature selection step can be obtained. For illustrative purposes, we focus our attention on the posterior results of the last scenario. In Figure 3.4 the reader can appreciate that there is an actual difference for the posterior estimates of the a_i's, with most of them below the value .5 and some others above it. However, the optimal threshold represented by the black line shows the few features required to obtain a high classification accuracy.

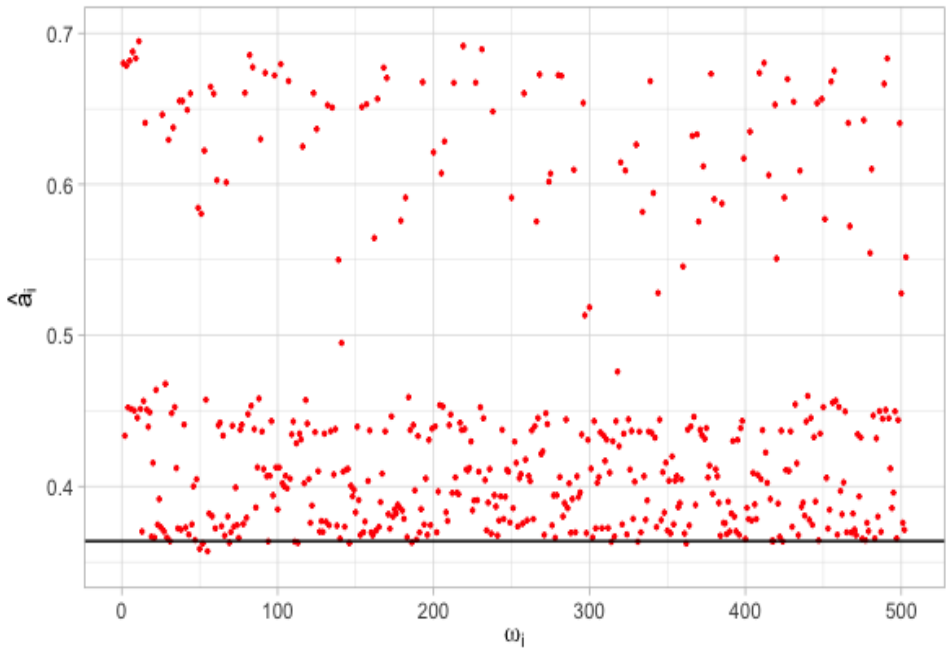


Figure 3.4: Posterior mean of the score parameters represented by the red dots and the optimal threshold as the horizontal black line.

Moreover it can be seen that the best accuracy can be achieved with a low threshold (Figure 3.5). In fact, the optimum threshold can be found at 0.3638 yielding only five total features used. Although it is also clear that after this optimum value, the accuracy performance remains stable and there is not much difference. Something that should not surprise us since there was a clear difference among the malicious and the benign executables.

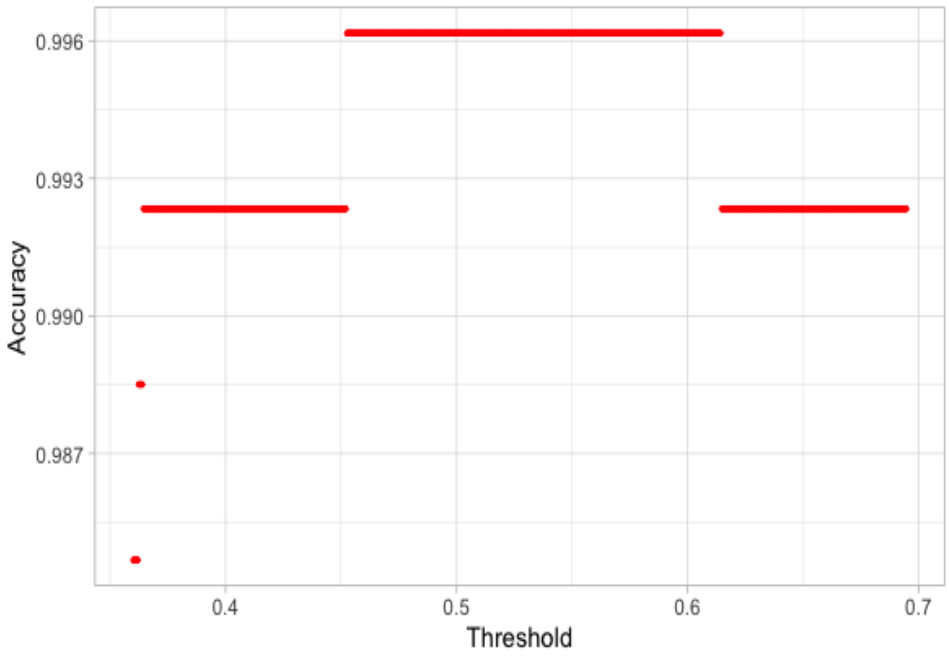


Figure 3.5: Accuracy for different thresholds.

3.4.2 Malware classification

Now, we turn our attention to the most interesting scenario considered so far, the malware classification task. In Chapter 2 it was seen that the beta-CoRM model was clearly outperformed by the boosted algorithms and the multinomial logistic regression model. This led us to think and propose a generalised version of the model for which an actual increase in the classification performance has been seen in the other scenarios considered so far. Hence, in this section we test the model capabilities in this challenging data set and compare them again against the MLE approach and the rest of the supervised learning classifiers, that is, beta-CoRM (b-CoRM), naive Bayes (nBayes), multinomial logistic regression model (ML), decision tree and its adaptive (AB), gradient (GB) and extreme gradient(XGB) boosted version. In Table 3.4 we present the classification performance for all the

models.

b-CoRM	gb-CoRM	MLE	nBayes	tree	AB	GB	XGB	ML
80.95	85.32	79.76	77.38	80.15	94.05	91.67	91.27	84.92

Table 3.4: Classification accuracy comparison (in %) for the generalised beta-CoRM model against commonly used supervised learning algorithms for the malware data set illustrated in Figure 2.7.

From the results presented in Table 3.4 we can immediately appreciate that the generalised beta-CoRM model performance is better than the original model, something that we were definitely expecting. Moreover, we can also appreciate how the generalised beta-CoRM is now only outperformed by the boosted algorithms. These are compelling preliminary results; however, there is still more information and a deeper analysis we can obtain from the feature selection step and its impact on the data and the classification performance.

3.4.2.1 Feature selection analysis

The generalised beta-CoRM model was developed on the elegant spike-and-slab interpretation of the score distribution and one of the most important contributions of this model is the possibility to introduce a feature selection step through the posterior analysis of the score parameters. For the malware classification data this posterior analysis represents an interesting opportunity to analyse the features that were selected as the "best" ones. In Figure 3.6 we present the results of this procedure by displaying the posterior mean of each score (red dots) and the optimal threshold represented by the black horizontal line.

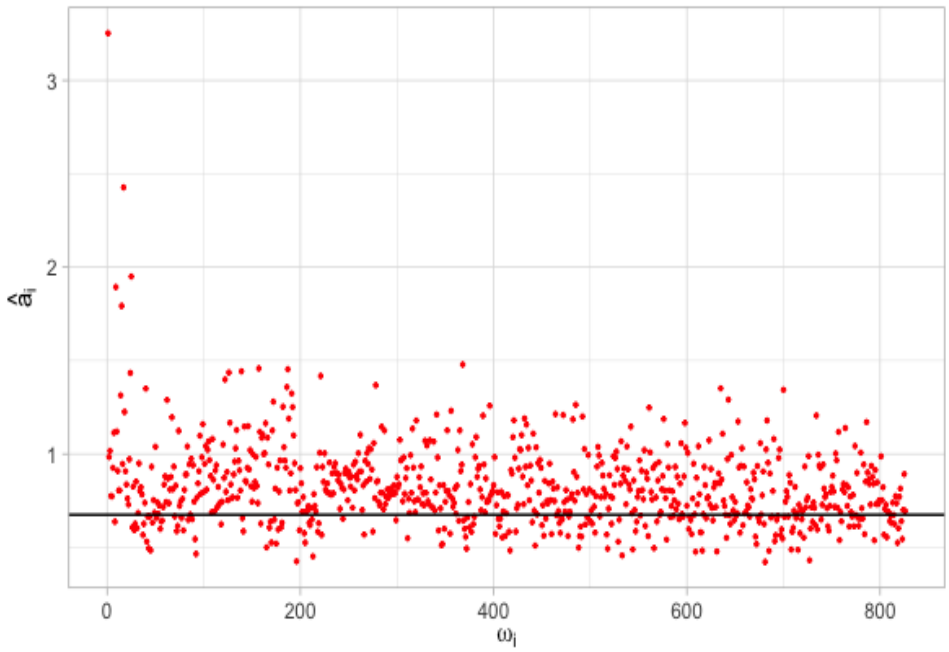


Figure 3.6: Posterior mean of the score parameters represented by the red dots and the optimal threshold as the horizontal black line.

At a first glance we can immediately appreciate that there are five features that have an associated score parameter above 1.5, which is certainly interesting since the rest of the features have an associated score parameter closer to one. Doing a more refined analysis, we were able to find out that these relatively large five posterior score parameters were associated to the most popular features in the data set, representing the features appearing in a large number of observations, as showed in Table 3.5.

Feature	Popularity	$\widehat{a}_i$
1	96.27%	3.252165
17	84.75%	2.427305
25	77.29%	1.950262
9	77.12%	1.893666
15	73.05%	1.792781

Table 3.5: The five most popular features across the data with their respective popularity and posterior score parameter.

Due to their popularity across the observations in the data set it is clear that their discriminative power will be certainly smaller compared to the "best" features and hence, it is not surprising why we should not include them to classify new malware. In a similar way, we can also analyse the least popular features, representing rare features appearing in a small number of observations. In Table 3.6 we present the five least popular features, with their level of popularity and their associated score parameter.

Feature	Popularity	$\widehat{a}_i$
172	4.40%	1.280181
633	4.40%	0.8957767
636	4.40%	0.8641384
642	4.40%	0.9728476
767	4.58%	0.8224271

Table 3.6: The five least popular features across the data with their respective popularity and posterior score parameter.

From the results presented in Table 3.6, it is interesting to notice how these rare features have a posterior score parameter close to one which might be an indication that just as the most common features they should not be included in the classification procedure to get the best performance. Now that we have analysed both rare and common features we turn our attention to the features marked as the "best" ones. As an example, the five "best" features along with their popularity and their score parameter are displayed in Table 3.7.

Feature	Popularity	$\widehat{a}_i$
681	16.95%	0.4223356
196	8.64%	0.4256526
727	11.53%	0.4313577
213	18.30%	0.4520498
533	9.66%	0.4571787

Table 3.7: The five "best" features across the data with their respective popularity and posterior score parameter.

Finally, and in order to have a thorough understanding of the effect of the feature selection step on the classification performance of the generalised beta-CoRM, we obtained the classification accuracy achieved for different thresholds which can be seen in Figure 3.7. We can appreciate how the accuracy increases dramatically as we start introducing the "best" features into the discrimination procedure and how once the rest of the features are introduced the accuracy decreases until it reaches a stable accuracy of 80.16%.

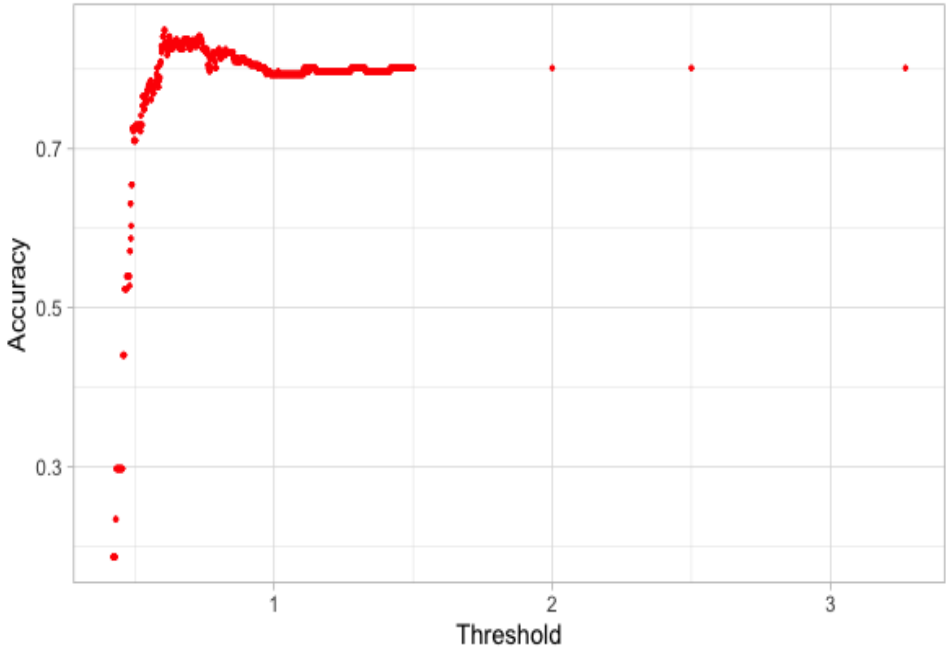


Figure 3.7: Accuracy for different thresholds.

It is also interesting to notice that the best accuracy performance was achieved with a threshold of 0.605, which represented only 96 features used yielding a reduction of almost 90% of the original 826 features considered at the beginning. In this case, it is worth noticing that for this threshold, both the most popular features and the least popular features were not selected. In this direction, it is an interesting exercise to appreciate how does the binary matrix look when we restrict our attention to only these 96 features (Figure 3.8).

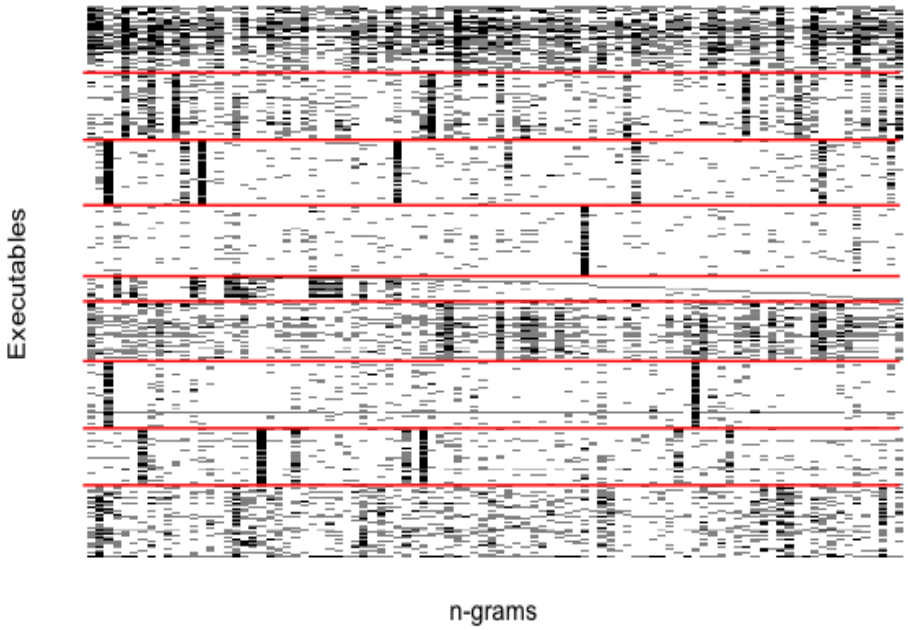


Figure 3.8: Graphical representation of the data set comprised of 590 malware (rows) of nine families separated with the solid horizontal lines and the 96 features selected through the generalised beta-CoRM. The dots represent the features that appear in each executable.

It is quite compelling to see how by restricting the data set to these 96 features we obtain groups with a clearer structure and hence, more distinguishable among each other compared to the original data illustrated in Figure 2.7. This is something that can also be noticed in the test set by restricting it to these 96 features as displayed in Figure 3.9.

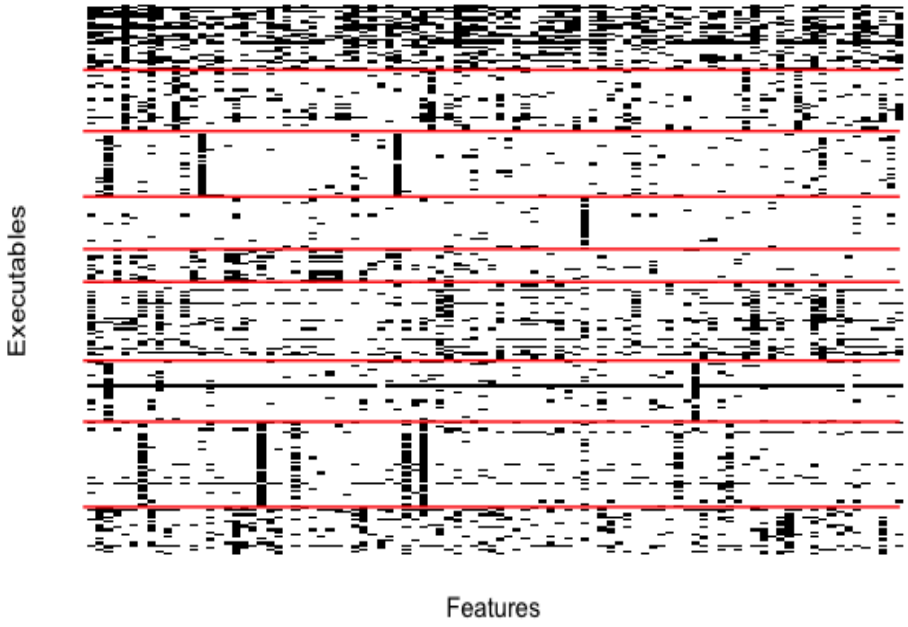


Figure 3.9: Graphical representation of the test set comprised of 252 malware (rows) of nine families separated with the solid horizontal lines and the 96 features selected through the generalised beta-CoRM. The dots represent the features that appear in each executable.

Finally from the results of the posterior inference, it is also interesting to direct our attention to the posterior estimate of the probabilities for each feature to appear in each of the nine families, that is, $\widehat{m_{ij}p_i}$. In Figure 3.10, the reader can find the graphical representation of these probabilities. It is attractive to see how the model is able to detect the differences across groups by highlighting with a deep blue their predominant features.

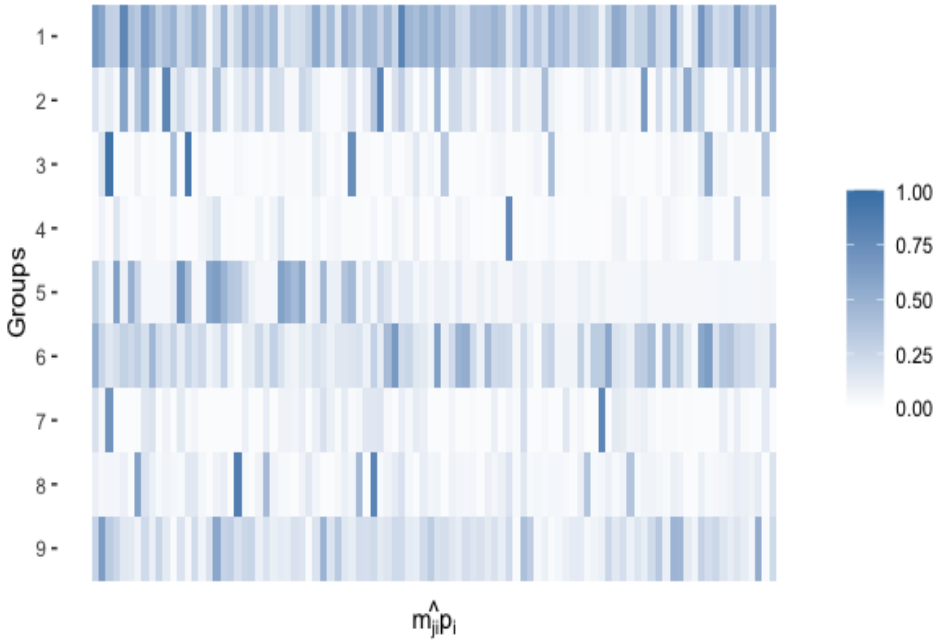


Figure 3.10: Posterior probability of each of the 96 features appearing in each of the 9 families of the malware classification data set.

3.4.2.2 Classification analysis

With the feature selection process and the analysis carried out so far, we believe it is also worth doing a more refined analysis of the classification under the generalised beta-CoRM model. To do so, we can recall that until now we have only used the accuracy to measure the models' classification performance. This is certainly a useful metric for the data sets considered so far because even for the imbalanced scenarios the class proportions are not that distant from one another. For instance, for the malware classification data set only one of the groups has significantly less observations. Hence, through the use of the accuracy we were able to centre directly our attention on the predictive performance of the model.

Nevertheless, there are other considerations that can be taken into account to measure the performance of a particular model. For example, in section 2.3.2.1 we saw that naive Bayes started classifying all the executables as malicious, yielding a large number of false positives. This is something that we have already discussed is not desired in a cyber security context. To understand how the model is performing on this area we can make use of the precision which is defined as the ratio between true positives and the total predicted positives (see *e.g.* Olson and Delen, 2008), that is,

$$\text{Precision} \quad = \quad \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}.$$

On the other hand we can also question ourselves about the false negatives. For the malware detection problem this is quite important since it would mean that some malware were classified as benign executables putting in risk the whole computer network. In this direction we can centre our attention on the recall which is defined as the ratio between true positives and the total actual positives (see *e.g.* Olson and Delen, 2008), that is,

$$\text{Recall} \quad = \quad \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}.$$

Finally, if the goal is to achieve a balance between the false positives and the false negatives, one could use the F_1 score which is defined as the harmonic mean of the precision and recall (see *e.g.* Olson and Delen, 2008), that is,

$$\text{F}_1 \quad = \quad 2\left(\frac{\text{precision*recall}}{\text{precision} + \text{recall}}\right),$$

and that has been widely used to measure the performance of the models for imbalanced data sets. From a modelling perspective, a good model should have

a precision, a recall and an F_1 score close to one. It is important to notice that although these metrics have been defined in a binary setting they can be easily extended to a multi-class framework since for each of the classes we can obtain their individual precision and recall and hence, their F_1 score. Then, these individual metrics can be combined to get a global metric for the classification model by averaging them together (see *e.g.* Yang and Liu, 1999; Grandini et al., 2020)

Now, turning our attention back to the malware classification example, we first recall that a classification accuracy of 84.92% was achieved, which represents that 214 out of 252 malware were correctly classified. For illustrative purposes, in Figure 3.11 we display only the correctly classified observations for each of the groups. From this figure, we can appreciate that the model is able to capture the general structure of the groups. This can be reasserted by putting together the correctly classified observations along with the misclassified ones, this is something that can be seen in Figure 3.12 where the misclassified malware is represented by the blue dots.

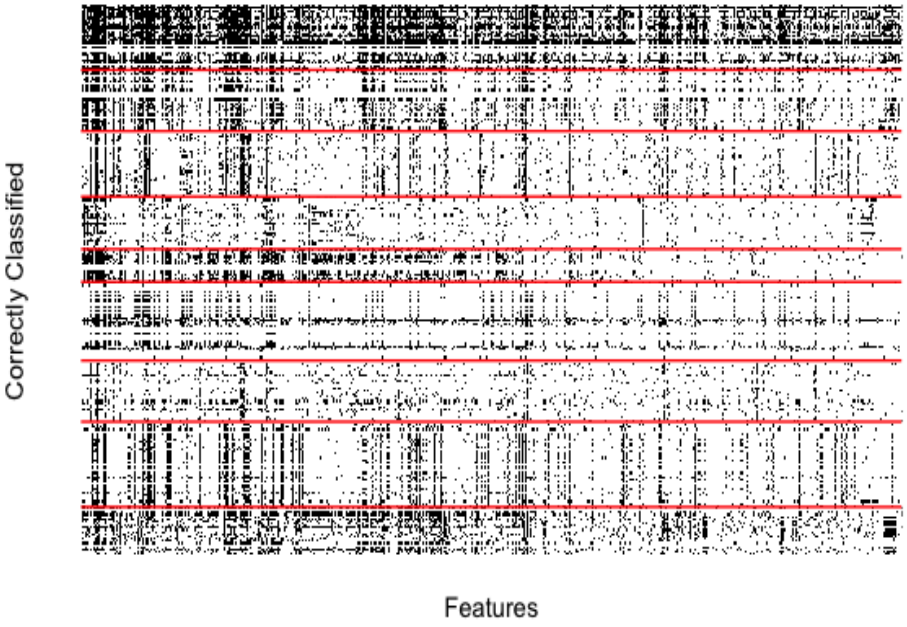


Figure 3.11: Correctly classified malware.

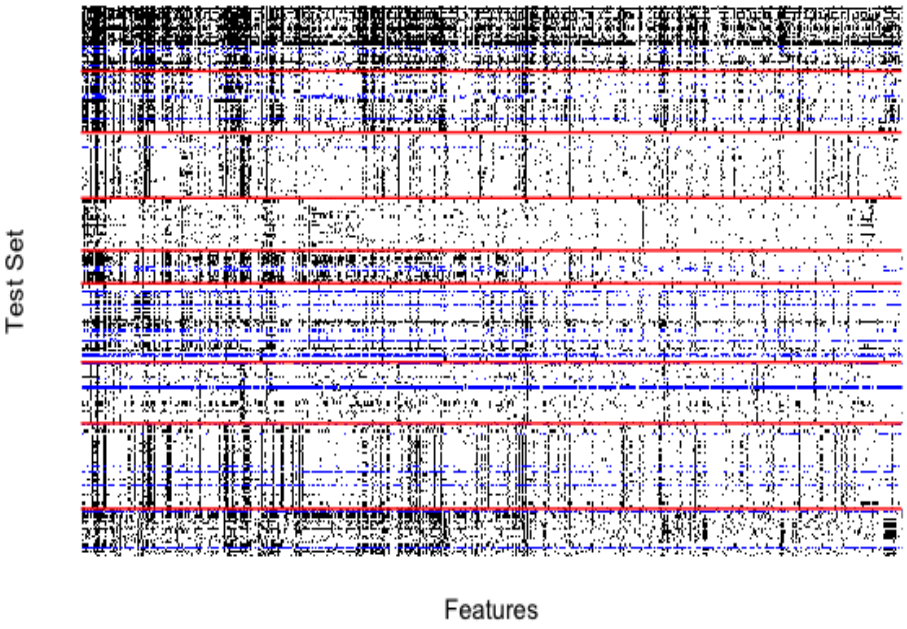


Figure 3.12: Correctly classified (black) and misclassified (blue).

As previously analysed, the model is able to capture the general structure of each group; that is why for the next step of our analysis we centre our attention on the confusion matrix (Table 3.8) which will provide us with a deeper understanding of the misclassified elements of the generalised beta-CoRM.

		Predicted families								
		1	2	3	4	5	6	7	8	9
True families	1	26	1	0	0	0	0	1	1	1
	2	1	22	0	0	1	1	2	0	1
	3	0	0	28	1	1	0	0	0	0
	4	0	0	0	24	0	0	0	0	0
	5	0	0	1	0	12	0	1	0	1
	6	6	0	0	7	0	23	0	0	0
	7	1	0	0	1	0	0	25	0	1
	8	3	0	0	0	0	1	0	34	1
	9	2	0	0	0	0	0	0	0	20

Table 3.8: Confusion matrix of the generalised beta-CoRM for the malware classification data set.

Interesting remarks can be made from the confusion matrix by recalling that families 3, 5, 7 and 9 are all Backdoors and that families 4 and 6 are certainly related since they are respectively Trojan and TrojanDownloader. This is certainly interesting because it can be noted that half of the misclassified observations in families 3, 5, 7 and 9 went to one of these families and that more than half of the misclassified observations in family 6 were classified as malware of the family 4. Finally, other compelling conclusions can be obtained through the precision, the recall and F_1 score for each class which are displayed in Table 3.9.

Family	1	2	3	4	5	6	7	8	9
Precision	66.67	95.65	96.55	72.73	85.71	92.00	86.21	97.14	80.00
Recall	86.67	78.57	93.33	100	80.00	63.89	89.29	87.18	90.91
F_1	75.36	86.27	94.92	84.21	82.76	75.41	87.72	91.89	85.11

Table 3.9: Precision, recall and F_1 score (in %) of the generalised beta-CoRM for each family of the malware classification data set.

From the results displayed in Table 3.9, we can clearly appreciate how family 6 despite having a good precision is being punished by the large number of misclassified elements, yielding a low recall and hence the second lowest F_1 score. On the other hand, it can also be seen that family 4 despite having a perfect recall is being punished by all the false positives, yielding a low precision. This issue can also be appreciated in family 1 where due to the large number of false positives the lowest precision is obtained which negatively impacts its F_1 score which is in fact, the lowest overall. It is clear that families 1 and 6 are the most problematic and on the contrary, it can be easily argued that the best performance is achieved for family 3 due to its high precision, recall and hence, F_1 score.

Finally, using these results we can also provide overall metrics for the generalised beta-CoRM which are displayed in Table 3.10 by averaging the individual metrics. We can appreciate consistent results with a slightly higher F_1 score compared to the accuracy, something that it is always desired since the F_1 score takes into account false positives and false negatives as well.

Accuracy	Precision	Recall	F_1
84.92	84.85	85.85	85.54

Table 3.10: Global precision, recall and F_1 score (in %) of the generalised beta-CoRM for the malware classification data set.

3.5 Concluding remarks

In this chapter we described a generalisation of the the beta-CoRM model introduced in the previous chapter, that allowed us to incorporate a feature selection step in the learning process. This model yielded a better classification accuracy and a deeper understanding of the importance of each of the features. This opens

interesting directions of research. First of all, there are more malware that we could use and from which more n-grams could be obtained. Now that the generalised model allows us to choose an optimum threshold, we could also extract the features with the better discriminative power. With this in mind, a not so restrictive first dimensionality reduction approach could be used on the feature space, since we might be leaving out n-grams that could potentially improve the classification accuracy. Moreover, further research about the optimum threshold also needs to be done.

Chapter 4

Beta compound random measure binary matrix factorisation

The beta-CoRM models described in the previous chapters are without a doubt an interesting approach to supervised learning for binary matrices. Although these models were specifically designed to model the data directly, there are other applications to the beta-CoRM as a prior that are worth considering and developing. In this chapter, we explore a binary matrix factorisation procedure with an underlying Bayesian nonparametric latent feature model that uses the beta-CoRM as prior on the latent causes.

Latent variables have an important role in many statistical models. These variables, often play the role of properties that have not been directly observed or hidden causes that explain the observations. In a parametric setting, the number of latent variables is assumed to be finite, which implies there is a unique representation that correctly characterises the data (Ghahramani et al., 2007). This assumption, although computationally convenient, might not always be ad-

equate, like in topic modelling, where new documents might not be well-modelled using the k latent topics initially considered. To overcome this, Bayesian nonparametric latent feature models work on the assumption that there is an unbounded number of possible latent features and only a small number of them are "active". Examples of Bayesian nonparametric latent models include the mixture of Dirichlet process (MDP) (see *e.g.* Antoniak, 1974; Escobar and West, 1995), the hierarchical Dirichlet process (HDP) (Teh et al., 2006) and the Indian Buffet Process (IBP) (Ghahramani and Griffiths, 2006; Ghahramani et al., 2007).

For the application to malware detection and classification presented throughout this thesis, the infinite number of latent traits can represent pieces of code, functions executed by the malware, or any other binary characteristic associated to executable programmes. This particular characteristic of factorial models, where different kinds of binary variables can be considered, has already been exploited in Thibaux and Jordan (2007), where the hierarchical beta process (HBP) was defined and applied to document classification. In their approach, the primary binary features were the words present in each text and possible new binary latent variables could also be considered like the indentation of the text.

For the approach that we will present in this chapter, we will assume that each observation $X_{kj} = \{x_{kji}\}_{i=1}^{M}$ is generated by a finite number of latent traits, $Z_{kj} = \{z_{kjl}\}_l$, with each one of these traits having an associated probability, a_{li}, of generating the i-th binary feature. Then, as we will discuss in the following section, the factorisation is done through an elegant link function that can be interpreted as if there was a second layer of binary latent indicators modulating the presence or absence of the n-grams.

4.1 Beta-CoRM BMF generative model

In order to provide a constructive specification of the generative process of the beta-CoRM BMF model, we recall that a beta process B is a stochastic process defined on a suitable space Ω. In the original formulation, Ω was considered to be the real line (Hjort, 1990); however, more general spaces can be considered. For the purposes of this chapter, and for the beta-CoRM BMF construction we consider the two-parameter beta process, $BP(c, \gamma)$, where c is the concentration parameter and $\gamma = B_0(\Omega)$ is the total mass of the base measure B_0. Another interesting possibility, that we do not cover in this thesis, would be the use of the three-parameter beta process (Broderick et al., 2012) which, contrary to the original formulation, exhibits a power law behaviour through the new discount parameter considered.

Contrary to Chapter 2 and 3, where the base measure was discrete, for the beta-CoRM BMF we consider a continuous base measure B_0. Hence, the Lévy measure of B is

$$\nu(dp, d\omega) \quad = \quad cp^{-1}(1-p)^{c-1}dpdB_0(d\omega).$$

It is straightforward to see that B is an infinite activity Lévy process, i.e., $\int \nu(dp, d\omega) = \infty$. Therefore, B contains an infinite number of small jumps, which makes it suitable for sparse latent variable models like the IBP and the beta-CoRM BMF. Using Kingman's representation theorem for completely random measures (Kingman, 1967), realisations of the beta process can be seen as

$$B = \sum_l p_l \delta_{\omega_l},$$

where the set of jumps $\{p_l\}_l$ lie in the unit interval, yielding an infinite number of coin-toss probabilities.

Following the generative process of the discrete beta-CoRM of the previous chapters, B will be used as the directing Lévy process whose jumps are going to be perturbed at group level by beta distributed scores $m_{1l}, ..., m_{dl} \sim \text{beta}(a, 1)$, that is, for each group $j \in \{1, ..., d\}$ we will have a base measure B_j such that,

$$B_j = \sum_l m_{jl} p_l \delta_{\omega_l}.$$

The perturbed weights are then used in conjunction with several realisations of Bernoulli processes to create the binary matrix of latent traits $\mathbf{Z}$. That is for observation k in group j we define

$$Z_{kj} = \sum_l z_{kjl} \delta_{\omega_l} \qquad\qquad z_{kjl} \sim \text{Bernoulli}(m_{jl} p_l).$$

Finally, the binary matrix factorisation model is fully specified by introducing a set of beta-distributed loadings $a_{l1}, .., a_{li}, .., a_{lM}$ which as established before, represent the probability of each latent trait generating the i-th binary feature and with M being the total number of features in the data. With the latent traits and their associated loadings, then x_{kji} follows a Bernoulli distribution with parameter given by

$$\mathbb{P}(x_{kji} = 1) \;\; = \;\; g(Z_{kj}, a_{\cdot i}) = 1 - \prod_l (1 - z_{kjl} a_{li}). \tag{4.1}$$

This particular choice of link function can be elegantly thought as if there were a second layer of independent latent indicator variables $v_{kjil} \sim \text{Ber}(z_{kjl}a_{li})$ and with $x_{kji} = \max_l\{v_{kjil}\}$. In this way, the corresponding feature would be present if at least one of the indicator variables, $\{v_{kjil}\}_l$, was active. Interestingly, this max representation is not unique, in fact we could also write $x_{kji} = 1 - \prod_l(1 - v_{kjil})$, yielding the same probabilities and results, hence, it is just a matter of choosing the one the researcher finds more useful. In our case, the *max* representation is especially useful for deriving important theoretical properties of this model, while for the posterior inference we (mostly) use the representation provided in equation (4.1).

Finally, it is also important to remark that this generative process also allows us to introduce dependence between features across observations through the set of latent traits and their respective set of loadings. Furthermore, this dependence will also exist within and across families, and will explicitly rely on the number of shared latent traits.

4.2 Properties

In this section we provide and develop some theoretical properties mainly related to the new dependent structure of the beta-CoRM BMF. These results will be particularly useful for the posterior inference, since they provide important information on the hyperparameters, and as explained in the previous section, for their derivation we use the *max* representation of the data since it certainly simplifies and makes clearer some of the expressions.

First of all, it is important to properly characterise the probability of an observation having (or not) a feature. From the beta-CoRM BMF model, it is easy to appreciate that $x_{kji} = 1$ if and only if $\sum_l v_{kjil} \geq 1$ and $x_{kji} = 0$ if and only if $v_{kjil} = 0 \; \forall l$. Since all the latent variables $\{v_{kjil}\}_l$ are independent, then

$$\mathbb{P}(x_{kji} = 0) \;\; = \;\; \prod_l \mathbb{P}(v_{kjil} = 0),$$

with

$$\begin{aligned}
\mathbb{P}(v_{kjil} = 0) &= \mathbb{P}(v_{kjil} = 0 | z_{kjl} = 0)\mathbb{P}(z_{kjl} = 0) + \mathbb{P}(v_{kjil} = 0 | z_{kjl} = 1)\mathbb{P}(z_{kjl} = 1) \\
&= \mathbb{P}(z_{kjl} = 0) + (1 - a_{li})\mathbb{P}(z_{kjl} = 1) \\
&= (1 - m_{jl}p_l) + (1 - a_{li})m_{jl}p_l \\
&= 1 - a_{li}m_{jl}p_l.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mathbb{P}(x_{kji} = 0) &= \prod_l (1 - a_{li}m_{jl}p_l) \\
\mathbb{P}(x_{kji} = 1) &= 1 - \prod_l (1 - a_{li}m_{jl}p_l).
\end{aligned} \tag{4.2}$$

Equation (4.2) allows us to provide a full characterisation of (4.1) in terms of the underlying probabilities by integrating out the latent traits. Since the probability of an observation having a feature increases as there are more latent traits present, then the weights of the beta process, the scores and the loadings modulate the probability of success found in (4.1). This is particularly useful to know the role of the hyperparameters of each component of the beta-CoRM

BMF model. For this, we notice that the hierarchical model is fully specified by considering

$$\begin{aligned} p_l &\sim \text{beta}(cB_0(d\omega_l), c(1-B_0(d\omega_l))) \\ m_{jl} &\sim \text{beta}(a,1) \\ a_{li} &\sim \text{beta}(\alpha,\beta). \end{aligned}$$

With these prior distributions we can easily integrate out the set of weights $\{p_l\}$, the scores $\{m_{jl}\}$, and the loadings $\{a_{li}\}$ to express everything in terms of the hyperparameters α, β, a. In order to simplify the notation we denote $B_0(d\omega_l) = b_{0,l}$, and hence,

$$\begin{aligned} & \mathbb{P}(x_{kji}=0|\alpha,\beta,a,b_{0,l}) \\ = & \int \mathbb{P}(x_{kji}=0|\{a_{li}\},\{m_{jl}\},p_l) f(\{a_{li}\},\{m_{jl}\},p_l|\alpha,\beta,a,b_{0,l}) \\ = & \int \prod_l (1-a_{li}m_{jl}p_l) f(\{a_{li}\},\{m_{jl}\},p_l|\alpha,\beta,a,b_{0,l}) \\ = & \prod_l \int (1-a_{li}m_{jl}p_l) \frac{a_{li}^{\alpha-1}(1-a_{li})^{\beta-1}}{B(\alpha,\beta)} a m_{jl}^{a-1} \frac{p_l^{cb_{0,l}-1}(1-p_l)^{c(1-b_{0,l})-1}}{B(cb_{0,l},c(1-b_{0,l}))} \\ = & \prod_l \left(1 - \int \frac{a_{li}^{\alpha}(1-a_{li})^{\beta-1}}{B(\alpha,\beta)} a m_{jl}^{a} \frac{p_l^{cb_{0,l}}(1-p_l)^{c(1-b_{0,l})-1}}{B(cb_{0,l},c(1-b_{0,l}))}\right) \\ = & \prod_l \left(1 - \frac{\alpha}{\alpha+\beta}\frac{a}{a+1} b_{0,l}\right), \end{aligned}$$

and similarly,

$$\mathbb{P}(x_{kji}=1|\alpha,\beta,a,b_{0,l}) \;=\; 1 - \prod_l \left(1 - \frac{\alpha}{\alpha+\beta}\frac{a}{a+1} b_{0,l}\right).$$

From the Chapters 2 and 3, we have already realised that smaller values of a should be preferred since larger values yield scores close to one and hence, undistinguishable groups. In fact in the limit $a \to \infty$, then $m_{jl} \to 1$ and it would be like having just N beta-Bernoulli processes. In this case, however, we can analyse the effect of α and β. It is clear that larger values of β will make more likely x_{kji} to be zero and larger values of α will make the loadings close to one and hence more likely x_{kji} to have the feature. Related to this, it is also important to remark that in the limit that $\mathbb{E}(a_{li}) = 1$, then $v_{kjil} = z_{kjl}$ and if $\mathbb{E}(a_{li})$ moves away from 1, it allows more discrepancy between v_{kjil} and z_{kjl}. In other words, an observation might have the trait but not the associated feature.

The results obtained so far, will be useful for the following sections, where the dependence structure is completely analysed. As established before, the main goal of the beta-CoRM BMF model is to introduce dependence across features, something that was not present in the beta-CoRM models presented in the previous chapters where there is zero correlation between the features. This dependency is achieved through the shared latent traits, something that will also be present across and within families.

4.2.1 Dependence across features for the same observation

In order to analyse the dependence structure we start with some notation, by defining the set of active latent traits of observation k in group j as $A_{kj} = \{l : z_{kjl} = 1\}$. As we will see below, this set and its cardinality play an important role in the beta-CoRM BMF structure.

To obtain the covariance, we proceed by first obtaining the joint probability of the same observation having feature i and i'. This procedure is done by noting that, conditioned on A_{kj}, x_{kji} and $x_{kji'}$ are independent, hence,

$$\begin{aligned}\mathbb{P}(x_{kji}=1,x_{kji'}=1) &= \sum_{A_{kj}}\mathbb{P}(x_{kji}=1,x_{kji'}=1|A_{kj})\mathbb{P}(A_{kj}) \\ &= \sum_{A_{kj}}\mathbb{P}(x_{kji}=1|A_{kj})\mathbb{P}(x_{kji'}=1|A_{kj})\mathbb{P}(A_{kj}).\end{aligned}$$

Given the set of active latent traits, the conditional marginal probabilities can be easily obtained from (4.1) and given by,

$$\mathbb{P}(x_{kji}=1|A_{kj}) = 1-\prod_{l\in A_{kj}}(1-a_{li}).$$

By the same reasoning, we have that $\mathbb{P}(x_{kji}=1|A_{kj}) = 1-\prod_{l\in A_{kj}}(1-a_{li'})$, and the joint probability can be finally expressed as

$$\begin{aligned}&\sum_{A_{kj}}\left(1-\prod_{l\in A_{kj}}(1-a_{li})\right)\left(1-\prod_{l\in A_{kj}}(1-a_{li'})\right)\mathbb{P}(A_{kj}) \\ &=\sum_{A_{kj}}\left(1-\prod_{l\in A_{kj}}(1-a_{li})\right)\left(1-\prod_{l\in A_{kj}}(1-a_{li'})\right)\prod_{l\in A_{kj}}m_{jl}p_l\prod_{l\in A_{kj}^c}(1-m_{jl}p_l).\end{aligned} \tag{4.3}$$

To have a well-defined expression for the joint probability it is just a matter of defining the product over the empty set, $A_{kj}=\emptyset$, to be equal to 1. By doing so, the first term of (4.3) is zero which corresponds where there are no active traits. Then, using equation (4.3) and the marginal probabilities given in equation (4.2), we can express the covariance of x_{kji} and $x_{kji'}$ as

$$
\begin{aligned}
&\mathbb{E}(x_{kji}x_{kji'}) - \mathbb{E}(x_{kji})\mathbb{E}(x_{kji'}) \\
&= \mathbb{P}(x_{kji} = 1, x_{kji'} = 1) - \mathbb{P}(x_{kji} = 1)\mathbb{P}(x_{kji'} = 1) \\
&= \sum_{A_{kj}} \left(1 - \prod_{l \in A_{kj}} (1 - a_{li})\right) \left(1 - \prod_{l \in A_{kj}} (1 - a_{li'})\right) \prod_{l \in A_{kj}} m_{jl}p_l \prod_{l \in A_{kj}^c} (1 - m_{jl}p_l) \\
&- \left(1 - \prod_{l} (1 - a_{li}m_{jl}p_l)\right) \left(1 - \prod_{l} (1 - a_{li'}m_{jl}p_l)\right).
\end{aligned}
\tag{4.4}
$$

It is interesting to notice that (4.4) can be simplified in the case when $z_{kjl} = v_{kjil}$ a.s. by letting $a_{li} \to 1$, yielding,

$$
\mathrm{Cov}(x_{kji}, x_{kji'}) = \sum_{A_{kj}} \prod_{l \in A_{kj}} m_{jl}p_l \prod_{l \in A_{kj}^c} (1 - m_{jl}p_l) - \left(1 - \prod_{l} (1 - m_{jl}p_l)\right)^2
$$

and, on the other hand, it is clear that as $a_{li} \to 0$ the covariance tends to 0.

Further interesting insights about the hyperparameters of the loadings, α and β, can be found by taking the expectation with respect to the set of loadings $a_{\cdot i} = \{a_{li}\}_l$ and $a_{\cdot i'} = \{a_{li'}\}_l$ of the joint conditional probability of $x_{kji} = 1$ and $x_{kji'} = 1$ given a fixed set of active latent traits A_{kj}. By doing so, we are able to characterise the joint probability as a function of α, β and $|A_{kj}|$, that is,

$$
\begin{aligned}
&\mathbb{E}_{a_{\cdot i}, a_{\cdot i'}}[\mathbb{E}(x_{kji}x_{kji'}|A_{kj})] \\
= \;& \mathbb{E}_{a_{\cdot i}, a_{\cdot i'}} \left[\left(1 - \prod_{l \in A_{kj}} (1 - a_{li})\right) \left(1 - \prod_{l \in A_{kj}} (1 - a_{li'})\right)\right]
\end{aligned}
$$

$$
\begin{aligned}
&= \mathbb{E}_{a_{\cdot i}, a_{\cdot i'}} \left[1 - \prod_{l \in A_{kj}} (1 - a_{li}) - \prod_{l \in A_{kj}} (1 - a_{li'}) + \prod_{l \in A_{kj}} (1 - a_{li}) \prod_{l \in A_{kj}} (1 - a_{li'}) \right] \\
&= 1 - 2 \left(\frac{\beta}{\alpha + \beta} \right)^{|A_{kj}|} + \left(\frac{\beta}{\alpha + \beta} \right)^{2|A_{kj}|} \\
&= \left(1 - \left(\frac{\beta}{\alpha + \beta} \right)^{|A_{kj}|} \right)^2 .
\end{aligned}
\tag{4.5}
$$

From (4.5) there are two trivial cases that arise immediately. In the limit $\alpha \to 0$ $a_{li} \to 0$ $\forall l$ and $x_{kji} = 0$ $\forall i$. On the contrary, in the limit $\beta \to 0$, and considering there is at least one latent trait active, $x_{kji} = 1$ $\forall i$. Also, for $\beta \neq 0$, if there are no latent traits active $x_{kji} = 0$ $\forall i$ and on the contrary, for a large number of latent traits active this probability will be close to one since $\frac{\beta}{\alpha+\beta} \leq 1$. This is something that we would have expected since it is the effect of $z_{kjl} = v_{kjil}$ as the loadings get closer to one.

4.2.2 Dependence between two observations in the same group

For the dependence between two observations in the same group, there are two scenarios that we could consider. First, the case for the same feature, that is x_{kji} and $x_{k'ji}$, and secondly, the case with different features, that is, x_{kji} and $x_{k'ji'}$. However, for the second case, it is easy to see that the variables are conditionally independent due to the sets of active traits A_{kj} and $A_{k'j}$ being conditionally independent and to the independence of the corresponding set of loadings $\{a_{li}\}_i$ and $\{a_{li'}\}_{i'}$.

Now, for the first case, it is true that they are also conditionally independent due to the conditional independence of the corresponding set of active traits. However, in this case, due to the shared loadings $\{a_{li}\}_i$, there is an interesting analysis that can be done, similar to the one carried out to obtain equation (4.5). By doing so, there is the possibility of further analysing the effect of α and β and the cardinalities of A_{kj} and $A_{k'j}$ in the beta-CoRM BMF model.

4.2.2.1 Dependence between x_{kji} and $x_{k'ji}$

Conditioning on A_{kj} and $A_{k'j}$ and obtaining an expression with respect to the hyperparameters by taking the expectation with respect to the loadings, yields that the joint marginal distribution can be expressed as

$$
\begin{aligned}
&\mathbb{E}_{a_{\cdot i}}[\mathbb{E}(x_{kji}x_{k'ji}|A_{kj}, A_{k'j})] \\
&= \mathbb{E}_{a_{\cdot i}}\left[\left(1 - \prod_{l\in A_{kj}}(1-a_{li})\right)\left(1 - \prod_{l\in A_{k'j}}(1-a_{li})\right)\right] \\
&= \mathbb{E}_{a_{\cdot i}}\left[1 - \prod_{l\in A_{kj}}(1-a_{li}) - \prod_{l\in A_{k'j}}(1-a_{li}) + \prod_{l\in A_{kj}}(1-a_{li})\prod_{l\in A_{k'j}}(1-a_{li})\right] \\
&= 1 - \left(\frac{\beta}{\alpha+\beta}\right)^{|A_{kj}|} - \left(\frac{\beta}{\alpha+\beta}\right)^{|A_{k'j}|} \\
&+ \mathbb{E}_{a_{\cdot i}}\left[\prod_{l\in A_{kj}\setminus A_{k'j}}(1-a_{li})\prod_{l\in A_{k'j}\setminus A_{kj}}(1-a_{li})\prod_{l\in A_{kj}\cap A_{k'j}}(1-a_{li})^2\right] \\
&= 1 - \left(\frac{\beta}{\alpha+\beta}\right)^{|A_{kj}|} - \left(\frac{\beta}{\alpha+\beta}\right)^{|A_{k'j}|} \\
&+ \left(\frac{\beta}{\alpha+\beta}\right)^{|A_{kj}\setminus A_{k'j}|}\left(\frac{\beta}{\alpha+\beta}\right)^{|A_{k'j}\setminus A_{kj}|}\left(\frac{\beta+1}{\alpha+\beta+1}\right)^{|A_{kj}\cap A_{k'j}|}
\end{aligned}
\tag{4.6}
$$

By noticing that $|A_{kj} \backslash A_{k'j}| = |A_{kj}| - |A_{kj} \cap A_{k'j}|$ and similarly, that $|A_{k'j} \backslash A_{kj}| = |A_{k'j}| - |A_{kj} \cap A_{k'j}|$, and by taking the expectation with respect the loadings for x_{kji} and $x_{k'ji}$ individually, we can obtain a simplified equation for the covariance given by

$$\left(\frac{\beta}{\alpha+\beta}\right)^{|A_{kj}|+|A_{k'j}|}\left[\left(\frac{(\alpha+\beta)^2(\beta+1)}{(\alpha+\beta+1)\beta^2}\right)^{|A_{kj}\cap A_{k'j}|}-1\right]. \tag{4.7}$$

Clearly, if for each observation the features do not share any latent traits, that is, $A_{kj} \cap A_{k'j} = \emptyset$ then the covariance is zero. On the other hand, if they share all the latent traits, then the covariance becomes

$$\left(\frac{\beta}{\alpha+\beta}\right)^{2|A_{kj}|}\left[\left(\frac{(\alpha+\beta)^2(\beta+1)}{(\alpha+\beta+1)\beta^2}\right)^{|A_{kj}|}-1\right]=\left(\frac{\beta+1}{\alpha+\beta+1}\right)^{|A_{kj}|}-\left(\frac{\beta}{\alpha+\beta}\right)^{2|A_{kj}|},$$

which works as an upper bound for (4.7). Finally, another trivial case occurs when one of the sets is properly contained in the other, for example, if $A_{k'j} \subset A_{kj}$ then the covariance becomes

$$\left(\frac{\beta}{\alpha+\beta}\right)^{|A_{kj}|+|A_{k'j}|}\left[\left(\frac{(\alpha+\beta)^2(\beta+1)}{(\alpha+\beta+1)\beta^2}\right)^{|A_{k'j}|}-1\right].$$

4.2.3 Further properties

From all the expressions and results derived in the previous sections, it is clear now the importance of the latent traits and the loadings for the dependence structure. With respect the latent traits, it is particularly interesting to know

the number of active traits and the number of shared traits within and across families. In this direction it is worth recalling that

$$Z_{kj} = \sum_l z_{kjl}\delta_{\omega_l} \qquad\qquad z_{kjl} \sim \text{Bernoulli}(m_{jl}p_l),$$

and since $\mathbb{E}(B(\Omega)) = B_0(\Omega)$

$$\mathbb{E}(Z_{kj}) \;=\; \mathbb{E}[\mathbb{E}(Z_{kj}|B_j)] = \sum_l \mathbb{E}(m_{jl}p_l)\delta_{\omega_l} = \frac{a}{a+1}B_0(\Omega) = \frac{a}{a+1}\gamma.$$

Therefore, the expected number of active latent traits is equal to $\frac{a}{a+1}\gamma$. For a fixed mass parameter γ, the score parameter a controls the number of latent traits present for each observation. That is, for small values of a we expect to see fewer active latent traits and for large a we expect to observe a γ number of active latent traits. Finally, for the behaviour on the shared latent traits, we have already derived some useful properties like the probability of two observations in different groups sharing the l-th latent trait,

$$\mathbb{P}(z_{kjl} = 1, z_{k'j'l} = 1) \;=\; \left(\frac{a}{a+1}\right)^2 \left(\frac{cb_{0,l}^2 + b_{0,l}}{c+1}\right)$$

and its covariance,

$$\text{Cov}(z_{kjl}, z_{k'j'l}) \;=\; \left(\frac{a}{a+1}\right)^2 \left(\frac{b_{0,l}(1-b_{0,l})}{c+1}\right).$$

Following the same reasoning, we can also obtain closed expressions for the probability of two observations in the same group sharing the l-th latent trait as

$$\begin{aligned}\mathbb{P}(z_{k'jl} = 1, z_{kjl} = 1) &= \mathbb{E}[\mathbb{E}(z_{k'jl}z_{kjl})|B_j(d\omega_l)] \\ &= \mathbb{E}(m_{jl}^2 p_l^2)\end{aligned}$$

$$= \left(\frac{a}{a+2}\right)\left(\frac{cb_{0,l}^2 + b_{0,l}}{c+1}\right)$$

and its covariance

$$\begin{aligned}\mathrm{Cov}(z_{k'jl}, z_{kjl}) &= \mathrm{Var}(B_j(d\omega_l)) \\ &= \left(\frac{ab_0}{a+2}\right)\left(\frac{(1-b_{0,l})(a+1)^2 + b_{0,l}(c+1)}{(c+1)(a+1)^2}\right).\end{aligned}$$

Finally, the loadings allow us to control the level of dependence. In this direction, the hyperparameters α and β have a key role since they control how far or close will the loadings be from 1, depending on whether we expect the observations within and across groups having a larger correlation or not.

4.3 Inference

In order to perform the posterior inference on the beta-CoRM BMF model, we first notice that complete likelihood can be written as

$$f(\mathbf{X}|\mathbf{Z}, \mathbf{a}) = \prod_{j=1}^{d}\prod_{k=1}^{n_j}\prod_{i=1}^{M}\left[1 - \prod_{l}(1 - z_{kjl}a_{li})\right]^{x_{kji}}\left[\prod_{l}(1 - z_{kjl}a_{li})\right]^{1-x_{kji}}, \tag{4.8}$$

with corresponding priors on the latent traits and their loadings given by

$$f(\mathbf{Z}|\mathbf{m}, \mathbf{p}) = \prod_{j=1}^{d}\prod_{k=1}^{n_j}\prod_{l}(m_{jl}p_l)^{z_{kjl}}(1 - m_{jl}p_l)^{1-z_{kjl}}$$

$$f(\mathbf{a}|\alpha, \beta) = \prod_{l}\prod_{i=1}^{M}\frac{1}{B(\alpha, \beta)}a_{li}^{\alpha-1}(1 - a_{li})^{\beta-1}.$$

Then the full posterior distribution up to proportionality is given by

$$\begin{aligned} f(\mathbf{Z},\mathbf{a},\mathbf{m},\mathbf{p}|\mathbf{X}) &\propto f(\mathbf{X}|\mathbf{Z},\mathbf{a})f(\mathbf{Z}|\mathbf{m},\mathbf{p})f(\mathbf{m}|a)f(\mathbf{p}|c,\gamma)f(\mathbf{a}|\alpha,\beta) \\ &= \prod_{j,k,i}\left[1-\prod_{l}(1-z_{kjl}a_{li})\right]^{x_{kji}}\left[\prod_{l}(1-z_{kjl}a_{li})\right]^{1-x_{kji}} \\ &\times \prod_{jkl}(m_{jl}p_l)^{z_{kjl}}(1-m_{jl}p_l)^{1-z_{kjl}}\prod_{l,i}a_{li}^{\alpha-1}(1-a_{li})^{\beta-1} \\ &\times f(\mathbf{m}|a)f(\mathbf{p}|c,\gamma). \end{aligned} \tag{4.9}$$

Just as for the beta-CoRM models discussed in Sections 2 and 3, we can introduce the slice sampling technique in order to facilitate the posterior sampling of the weights p_l's and the score parameters m_{jl}'s. With this simulation technique, we can augment the posterior distribution to

$$\begin{aligned} f(\mathbf{Z},\mathbf{Y},\mathbf{a},\mathbf{m},\mathbf{p}|\mathbf{X}) &\propto f(\mathbf{X}|\mathbf{Z},\mathbf{a})f(\mathbf{Z}|\mathbf{Y},\mathbf{p})f(\mathbf{Y}|\mathbf{m})f(\mathbf{m}|a)f(\mathbf{p}|c,\gamma)f(\mathbf{a}|\alpha,\beta) \\ &= \prod_{j,k,i}\left[1-\prod_{l}(1-z_{kjl}a_{li})\right]^{x_{kji}}\left[\prod_{l}(1-z_{kjl}a_{li})\right]^{1-x_{kji}} \\ &\times \prod_{jkl}\left(\delta_0^{z_{kjl}}\right)^{1-y_{kjl}}\left(p_l^{z_{kjl}}(1-p_l)^{1-z_{kjl}}\right)^{y_{kjl}}\prod_{jkl}m_{jl}^{y_{kjl}}(1-m_{jl})^{1-y_{kjl}} \\ &\times \prod_{l,i}a_{li}^{\alpha-1}(1-a_{li})^{\beta-1} \\ &\times f(\mathbf{m}|a)f(\mathbf{p}|c,\gamma). \end{aligned} \tag{4.10}$$

4.3.1 Posterior of the directing beta process

Now that the posterior distribution has been obtained, we can then proceed to the marginal distributions which will be required for the Gibbs sampling algorithm.

So, we first turn to the inference on the directing beta process.

The posterior inference on the classical beta-Bernoulli process and related models like the IBP, heavily rely on the conjugacy property of the beta and the Bernoulli process. A quick derivation of this property can be obtained from the hierarchical representation of the beta-Bernoulli model given by,

$$\begin{aligned} Z_i(d\omega)|B(d\omega) &\overset{\text{iid}}{\sim} \text{Bernoulli}(B(d\omega)) \\ B(d\omega) &\sim \text{beta}(cB_0(d\omega), c(1 - B_0(d\omega))). \end{aligned}$$

From which it can be derived that,

$$\begin{aligned} B(d\omega)|Z_{1,\ldots,n} \sim \text{beta}\Big(&cB_0(d\omega) + \sum_{i=1}^{n} Z_i(d\omega), \\ &c(1 - B_0(d\omega)) + \sum_{i=1}^{n} (1 - Z_i(d\omega))\Big), \end{aligned}$$

and using this expression it can be easily derived that

$$B|Z_{1,\ldots,n} \sim \text{BP}\left(c + n, \frac{c}{c+n}B_0 + \frac{1}{c+n}\sum_{i=1}^{n} Z_i\right).$$

This structure has been particularly useful for the Bayesian nonparametric models that use the IBP as a prior, where the beta process is integrated out and a nice predictive distribution can be obtained,

$$Z_{n+1}|Z_{1,\ldots,n} \sim \text{BeP}\left(\frac{c}{c+n}B_0 + \frac{1}{c+n}\sum_{i=1}^{n} Z_i\right).$$

In Chapter 2 we saw that, contrary to the beta process, the beta-CoRM model does not have a conjugacy property with respect to Bernoulli observations. Hence, the posterior inference for the directing beta process cannot be done directly and a slice sampling technique was developed that allowed us to perform a fast and efficient posterior inference. In equation (4.9) the reader can appreciate the effect of this procedure yielding a more tractable expression than the one found in the original model and that provides a conjugate model between the directing beta process and the Bernoulli latent traits, that is,

$$\begin{aligned} B(d\omega)|Z,Y \quad \sim \quad & \text{beta}\Big(cB_0(d\omega) + \sum_{j,k} Z_{kj}(d\omega)Y_{kj}(d\omega), \\ & c(1 - B_0(d\omega)) + \sum_{j,k}(1 - Z_{kj}(d\omega))Y_{kj}(d\omega)\Big). \end{aligned} \tag{4.11}$$

This representation will be extremely useful no matter the approach considered for the posterior inference of the beta process. It is important to remark that, just as for any Bayesian nonparametric prior, it is computationally unfeasible to consider the infinite number of atoms that conform the beta process. In practice, several inferential techniques have been proposed to address this issue. One of such approaches is to define a truncation of the completely random measure which leads to a finite-dimensional posterior. For example, for stick-breaking priors, in Ishwaran and James (2001) it is argued that the number of atoms included should be chosen in such a way that the finite model is nearly undistinguishable from the infinite one, where the difference between them can be measured using the L_1 distance. Since defining this number might be troublesome for other nonparametric priors, more general schemes where the truncation point is random can also be considered, for a review on this the reader can refer to Griffin and Holmes (2010) and the references therein.

For the beta process and the IBP, which have stick-breaking representations, the fixed and random truncation approaches for the inferential process have been widely used in both MCMC and variational schemes (see *e.g.* Doshi et al., 2009; Paisley et al., 2011, 2012; Teh et al., 2007). Another interesting inferential scheme for this nonparametric prior is to split the full conditional into a finite (the observed atoms) and an infinite part (the unobserved ones). In this case the inference exploits the Lévy properties of this stochastic process, so that it can be done separately by independently updating the weights of the observed atoms and approximating the remaining space (see *e.g.* Paisley and Jordan, 2016).

4.3.1.1 Truncation method

For the purposes of this thesis we consider a truncation method on the directing beta process. However, it is compelling to notice that the truncation is done directly on the completely random measure rather than on the stick-breaking construction. For this to be done we consider beta prior sieves which can be defined as:

Definition 4.1 (Beta prior sieves)**.** *Let c be the concentration parameter and B_0 be a diffuse and finite measure defined on Ω. For an integer $R > B_0(\Omega) = \gamma$, we define a finite approximation to the beta process as $B^{(R)} = \sum_{l=1}^{R} p_l \delta_{\omega_l}$, where $p_l \sim beta(c\gamma/R, c(1-\gamma/R))$ and $\omega_l \sim B_0/\gamma$, with all the random variables drawn independently.*

By using beta prior sieves we are also able to exploit the simulation techniques derived in Chapter 2 and 3, where a discrete base measure was considered. Hence, considering the conjugacy property of the augmented model it can be seen that

the posterior for the weights p_l's is given by:

$$p_l|\mathbf{Y},\mathbf{Z} \;\sim\; \text{beta}\left(\sum_{j,k} z_{kjl}y_{kjl} + \frac{c\gamma}{R}, \sum_{j,k}(1-z_{kjl})y_{kjl} + c\left(1-\frac{\gamma}{R}\right)\right)$$

Finally, we could also give prior distributions to γ and c. Since both of them are positive variables an adaptive random walk Metropolis-Hastings scheme could be used on $\log(\gamma)$ and $\log(c)$ since clearly there is no conjugacy and the posterior does not have a closed form due to the beta function involved. Gamma priors with respective parameters ψ_γ, κ_γ and ψ_c, κ_c seems to be a sensible choice, with all of the hyperparmeters having a gamma vague prior distribution.

4.3.2 Posterior of the scores

The prior distribution of the scores associated to p_l is given by

$$\mathbb{P}(\{m_{jl}\}|a) \;=\; \prod_{j=1}^{d} a m_{jl}^{a-1}$$

and the posterior distribution, just as for the discrete beta-CoRM model is (up to proportionality):

$$\mathbb{P}(\{m_{jl}\}|\{y_{kjl}\},a) \;\propto\; \prod_{j=1}^{d} m_{jl}^{a-1} m_{jl}^{\sum_k y_{kjl}} (1-m_{jl})^{n_j - \sum_k y_{kjl}}$$

hence,

$$m_{jl}|\mathbf{Y},a \;\sim\; \text{beta}\left(a + \sum_k y_{kjl}, n_j - \sum_k y_{kjl} + 1\right).$$

The structure of the prior distribution also allows us to provide a conjugate model for the score parameter a by choosing a gamma prior (just as it was done in Chapter 3 for each a_i). By doing so, if we let $\mathbf{M}$ be the matrix of scores, then

$$a|\mathbf{M}, \psi_a, \kappa_a \sim \text{gamma}\left(Rd + \psi_a, \kappa_a - \sum_{l,j} \log(m_{jl})\right),$$

where ψ_a and κ_a can have assigned vague gamma priors. So that, κ_a has a conjugate posterior and ψ_a can be updated using the adaptive random walk Metropolis-Hastings described in Chapter 3.

4.3.3 Posterior of Y

Now, the posterior inference on the auxiliary variables of the slice sampling model can be done by noting that the prior for an observed atom ω_l is given by:

$$\mathbb{P}(\{y_{kjl}\}|\{m_{jl}\}) = \prod_{j=1}^{d}\prod_{k=1}^{n_j} m_{jl}^{y_{kjl}}(1-m_{jl})^{1-y_{kjl}}$$

and the posterior is equal to

$$\prod_{j=1}^{n_j}\prod_{k=1}^{n_j}(\delta_0^{z_{kjl}})^{(1-y_{kjl})}(p_l^{z_{kjl}}(1-p_l)^{(1-z_{kjl})})^{y_{kjl}} m_{jl}^{y_{kjl}}(1-m_{jl})^{1-y_{kjl}} \tag{4.12}$$

In this case, we need to consider two cases, conditioned on $z_{kjl} = 1$ we have that $y_{kjl} = 1$ a.s. and conditioned on $z_{kjl} = 0$ from (4.11) we have that

$$\mathbb{P}(y_{kjl} = 1|\{z_{kjl} = 0\}, m_{jl}, p_l) \propto (1-p_l)m_{jl}$$

$$\mathbb{P}(y_{kjl} = 0 | \{z_{kjl} = 0\}, m_{jl}, p_l) \propto 1 - m_{jl}.$$

Therefore,

$$y_{kjl}|z_{kjl}, m_{jl}, p_l \sim \begin{cases} \delta_1 & \text{if} \quad z_{kjl} = 1 \\ \text{Ber}\left(\frac{(1-p_l)m_{jl}}{1-p_l m_{jl}}\right) & \text{if} \quad z_{kjl} = 0 \end{cases}$$

4.3.4 Posterior of Z

The marginal posterior density is given by

$$\begin{aligned} f(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{p}, \mathbf{a}) \quad &\propto \quad \prod_{kji} \left\{ \left[1 - \prod_l (1 - z_{kjl} a_{li}) \right]^{x_{kji}} \left[\prod_l (1 - z_{kjl} a_{li}) \right]^{1 - x_{kji}} \right\} \\ &\times \quad \prod_{kjl} (\delta_0^{z_{kjl}})^{(1-y_{kjl})} (p_l^{z_{kjl}} (1 - p_l)^{(1-z_{kjl})})^{y_{kjl}}. \end{aligned} \tag{4.13}$$

To update the latent traits, there are two cases two consider. First, if $y_{kjl} = 0$ then $z_{kjl} = 0$ a.s.. On the contrary if $y_{kjl} = 1$ then $z_{kjl} = 1$ with probability proportional to

$$p_l \prod_i \left\{ \left[1 - (1 - a_{li}) \prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{x_{kji}} \left[(1 - a_{li}) \prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{1 - x_{kji}} \right\}.$$

Similarly, given that $y_{kji} = 1$ then $z_{kjl} = 0$ with probability proportional to

$$(1 - p_l) \prod_i \left\{ \left[1 - \prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{x_{kji}} \left[\prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{1 - x_{kji}} \right\}.$$

Under this inferential scheme it is possible that a slow mixing will be observed in the MCMC. One way to address this might be by marginalising over $\mathbf{Y}$ in (4.10) to update $\mathbf{Z}$ which yields that $z_{kjl} = 0$ with probability proportional to

$$(1 - m_{jl} p_l) \prod_i \left\{ \left[1 - \prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{x_{kji}} \left[\prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{1 - x_{kji}} \right\}$$

and $z_{kjl} = 1$ with probability proportional to

$$p_l m_{jl} \prod_i \left\{ \left[1 - (1 - a_{li}) \prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{x_{kji}} \left[(1 - a_{li}) \prod_{l' \neq l} (1 - z_{kjl'} a_{l'i}) \right]^{1 - x_{kji}} \right\}.$$

4.3.5 Posterior of the loadings

Now that we have derived closed expressions for the rest of the variables involved in the beta-CoRM BMF model, the only one remaining are the loadings. From the full posterior distribution (4.9), we can see that the conditional posterior is (up to proportionality) given by

$$\begin{aligned} f(\mathbf{a}|\mathbf{Z}, \mathbf{X}) \;\propto\; & \prod_{jki} \left\{ \left[1 - \prod_l (1 - z_{kjl} a_{li}) \right]^{x_{kji}} \left[\prod_l (1 - z_{kjl} a_{li}) \right]^{1 - x_{kji}} \right\} \\ & \times \prod_{li} a_{li}^{\alpha - 1} (1 - a_{li})^{\beta - 1}. \end{aligned}$$

Hence, for each a_{li} we have the following expression

$$\begin{aligned} & a_{li}^{\alpha-1}(1-a_{li})^{\beta-1}\prod_{j,k}\left\{\left[1-\prod_{l}(1-z_{kjl}a_{li})\right]^{x_{kji}}\left[\prod_{l}(1-z_{kjl}a_{li})\right]^{1-x_{kji}}\right\} \\ \propto \; & a_{li}^{\alpha-1}(1-a_{li})^{\beta-1}\prod_{j,k}\left\{\left[1-(1-z_{kjl}a_{li})\prod_{l'\neq l}(1-z_{kjl'}a_{l'i})\right]^{x_{kji}}\left[(1-z_{kjl}a_{li})\right]^{1-x_{kji}}\right\}. \end{aligned} \tag{4.14}$$

The product in (4.13) can be further simplified by noting that we only need to consider the product over the pairs (j,k) such that $z_{kjl}=1$, so that if denote this set by B_l and similarly if we define $C_i=\{(j,k):x_{kji}=1\}$ then the posterior can be written as

$$\begin{aligned} & \propto \; a_{li}^{\alpha-1}(1-a_{li})^{\beta-1}\prod_{j,k\in B_l}\left\{\left[1-(1-a_{li})\prod_{l'\neq l}(1-z_{kjl'}a_{l'i})\right]^{x_{kji}}\left[(1-a_{li})\right]^{1-x_{kji}}\right\} \\ & = \; a_{li}^{\alpha-1}(1-a_{li})^{\beta+|B_l|-C_l-1}\prod_{j,k\in B_l}\left[1-(1-a_{li})\prod_{l'\neq l}(1-z_{kjl'}a_{l'i})\right]^{x_{kji}} \\ & = \; a_{li}^{\alpha-1}(1-a_{li})^{\beta+|B_l|-C_l-1}\prod_{j,k\in B_l\cap C_i}\left[1-(1-a_{li})\prod_{l'\neq l}(1-z_{kjl'}a_{l'i})\right] \\ & \propto \; a_{li}^{\alpha-1}(1-a_{li})^{\beta+|B_l|-C_l-1}\prod_{j,k\in B_l\cap C_i}\left[1+\frac{\prod_{l'\neq l}(1-z_{kjl'}a_{l'i})}{1-\prod_{l'\neq l}(1-z_{kjl'}a_{l'i})}a_{li}\right]. \end{aligned} \tag{4.15}$$

Equation (4.15) sort of resembles a hypergeometric-Gauss distribution. In order to sample from this density we could use Metropolis-Hastings adaptive step. However, there is another alternative worth exploring by recalling the existence of the second layer of binary latent traits. These latent traits can be used in order

to update the loadings by noting that

$$\begin{aligned} f(\mathbf{V}|\mathbf{Z},\mathbf{a}) &= \prod_{j=1}^{d}\prod_{k=1}^{n_j}\prod_{i=1}^{M}\prod_{l=1}^{R}(z_{kjl}a_{li})^{v_{kjil}}(1-z_{kjl}a_{li})^{1-v_{kjil}} \\ &\overset{\text{w.r.t } a_{li}}{\propto} \prod_{j=1}^{d}\prod_{k=1}^{n_j}(z_{kjl}a_{li})^{v_{kjil}}(1-z_{kjl}a_{li})^{1-v_{kjil}}. \end{aligned} \tag{4.16}$$

Hence, a close expression for the loadings can be obtained by using the set B_l previously defined, and given by a beta distribution proportional to

$$f(a_{li}|\mathbf{V},\mathbf{Z}) \propto a_{li}^{\alpha+\sum_{(j,k)\in B_l} v_{kjil}-1}(1-a_{li})^{\beta+|B_l|-\sum_{(j,k)\in B_l} v_{kjil}-1},$$

with this approach gamma hyperpriors could be assigned to α and β and an adaptive random walk Metropolis Hastings used to update them.

Lastly, we need to update as well the latent traits, v_{kjil}'s. For this, we need to recall that $x_{kji} = 0$ if $v_{kjil} = 0\ \forall l$, hence, conditioned on $x_{kji} = 0$ all the latent traits will be zero. On the contrary, $x_{kji} = 1 \Leftrightarrow \sum_l v_{kjil} \geq 1$, hence, we can update the variables by considering the following steps:

1. For $l = 1$,

$$\begin{aligned} \mathbb{P}(v_{kji1} = 1) &\propto z_{kj1}a_{1i} \\ \mathbb{P}(v_{kji1} = 0) &\propto (1-z_{kj1}a_{1i})\left(1-\prod_{l>1}(1-z_{kjl}a_{li})\right). \end{aligned}$$

2. For $l = m < R$, there are two cases to consider, if $\sum_{l=1}^{m-1} v_{kjil} > 0$, then

$v_{kjim} \sim$ Bernoulli $(z_{kjm}a_{lm})$. On the contrary,

$$\begin{aligned}\mathbb{P}(v_{kjim}=1) &\propto z_{kjm}a_{mi}\\ \mathbb{P}(v_{kjim}=0) &\propto (1-z_{kjm}a_{mi})\left(1-\prod_{l>m}(1-z_{kjl}a_{li})\right).\end{aligned}$$

3. Finally, for $l=R$, if $\sum_{l=1}^{R-1} v_{kjil} > 0$ then $v_{kjiR} \sim$ Bernoulli $(z_{kjR}a_{Ri})$, and on the contrary, $v_{kjiR}=1$ almost surely.

4.4 Predictive distribution

Just as for the beta-CoRM classification, for a new observation Y we obtain the predictive distribution for each of the known families, and select the one with the highest probability. In this case, we can use equation (4.2) to obtain the posterior probabilities given by,

$$\begin{aligned}\mathbb{P}(x_{(k+1)ji}=1|\{x_{kji}\}) &= \int \mathbb{P}(x_{(k+1)ji}=1|\mathbf{a},\mathbf{m},\mathbf{p})f(\mathbf{a},\mathbf{m},\mathbf{p}|\{x_{kji}\})\\ &= \int\left[1-\prod_{l}(1-a_{li}m_{jl}p_l)\right]f(\mathbf{a},\mathbf{m},\mathbf{p}|\{x_{kji}\})\\ &= 1-\mathbb{E}_{a_{\cdot i},mj\cdot p.}\left[\prod_{l}(1-a_{li}m_{jl}p_l)\right],\end{aligned} \tag{4.17}$$

which can be approximated using a Monte Carlo estimator with T samples from the posterior distribution by

$$\mathbb{P}(x_{(k+1)ji}=1|\{x_{kji}\}) \approx 1-\frac{1}{T}\sum_{t=1}^{T}\left[\prod_{l}\left(1-a_{li}^{(t)}m_{jl}^{(t)}p_l^{(t)}\right)\right],$$

and,

$$\mathbb{P}(x_{(k+1)ji} = 0 | \{x_{kji}\}) \approx \frac{1}{T} \sum_{t=1}^{T} \left[\prod_{l} \left(1 - a_{li}^{(t)} m_{jl}^{(t)} p_{l}^{(t)} \right) \right].$$

4.5 Synthetic data

In this section we present and discuss some illustrations and results of the beta-CoRM BMF model on a synthetic data set comprised of three groups with ten observations each, which has been sampled from the generative process described in Section 4.1 using as parameters $(\gamma, c, R, a, \alpha, \beta) = (2, 1, 5, 1, .1, .1)$. This particular choice of parameters will be discussed further. Meanwhile, in Figure 4.8a and Figure 4.8b the latent traits and the loadings are displayed, while in Figure 4.2 the resulting binary matrix is illustrated.

(a) Latent traits. (b) Loadings.

Figure 4.1: The latent traits and the loadings generated by the beta-CoRM BMF generative process with parameters $(\gamma, c, R, a, \alpha, \beta) = (2, 1, 5, 1, .1, .1)$.

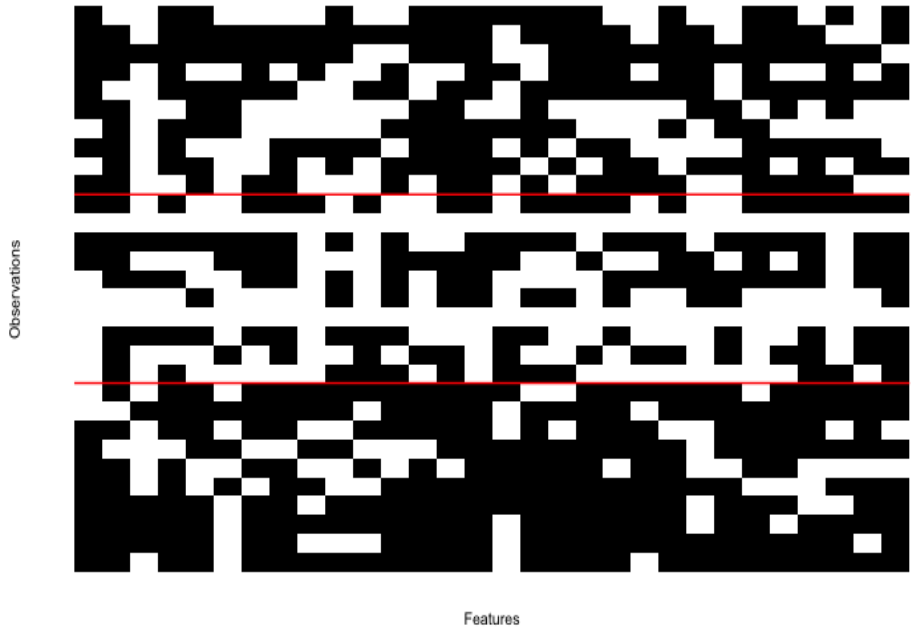


Figure 4.2: Synthetic data generated by the beta-CoRM BMF generative process with parameters $(\gamma, c, R, a, \alpha, \beta) = (2, 1, 5, 1, .1, .1)$.

The particular choice of $\alpha = \beta = .1$ allows us to have the loadings either close to one or to zero, making some of the latent traits predominant which has a clear impact on the data. For the posterior inference we consider for each parameter $a, \alpha, \beta, \gamma, c$ a weekly informative prior given by a gamma $(.001, .001)$ distribution. This yields the following conditional densities

$$f(c|\gamma, \mathbf{p}) \propto \frac{c^{.001-1}}{B(c,\gamma)^R} \exp\left[-c\left(.001 - \frac{\gamma}{R}\sum_{l=1}^{R}\log(p_l) - \left(1-\frac{\gamma}{R}\right)\sum_{l=1}^{R}\log(1-p_l)\right)\right]$$

$$f(\gamma|c, \mathbf{p}) \propto \frac{\gamma^{.001-1}}{B(c,\gamma)^R} \exp\left[-\gamma\left(.001 - \frac{c}{R}\sum_{l=1}^{R}\log\left(\frac{p_l}{1-p_l}\right)\right)\right]$$

$$f(a|\mathbf{M}) \propto a^{Rd+.001-1} \exp\left[-a\left(.001 - \sum_{l,j}\log(m_{jl})\right)\right]$$

$$f(\alpha|\beta, \mathbf{A}) \propto \frac{\alpha^{.001-1}}{B(\alpha,\beta)^{RM}} \exp\left[-\alpha\left(.001 - \sum_{l,i}\log(a_{li})\right)\right]$$

$$f(\beta|\alpha,\mathbf{A}) \propto \frac{\beta^{.001-1}}{B(\alpha,\beta)^{RM}} \exp\left[-\beta\left(.001 - \sum_{l,i}\log(1-a_{li})\right)\right].$$

From these conditional densities it is straightforward to see that the score parameter a has a gamma posterior distribution, while the rest of the hyperparameters require an adaptive Metropolis-Hastings algorithm on their respective log. To start the sampler we fix all the hyperparameters to 1 and then we initialise the matrix of latent traits to a random binary matrix. To discover the true values and to help the mixing of the chain, we run 120,000 simulations from which half of them correspond to the burning period. The remaining 60,000 are then thinned using a lag of 15, yielding an effective sample size of 4000. The thinned dynamics of the parameters and the initial and the last matrix of latent traits are respectively displayed from Figure 4.3 to Figure 4.8.

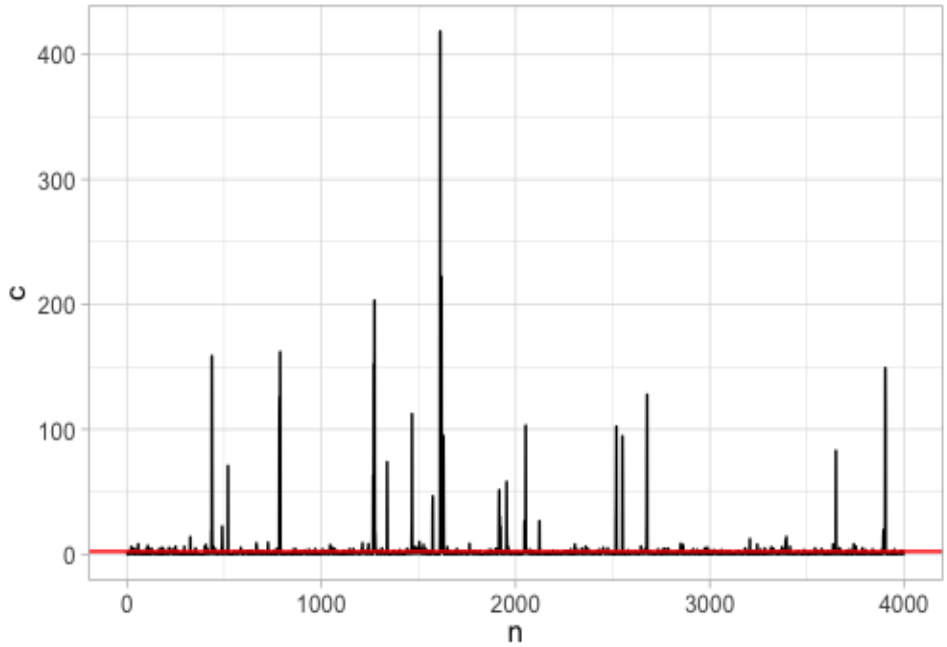


Figure 4.3: Posterior samples of c. The horizontal red line represents the mean.

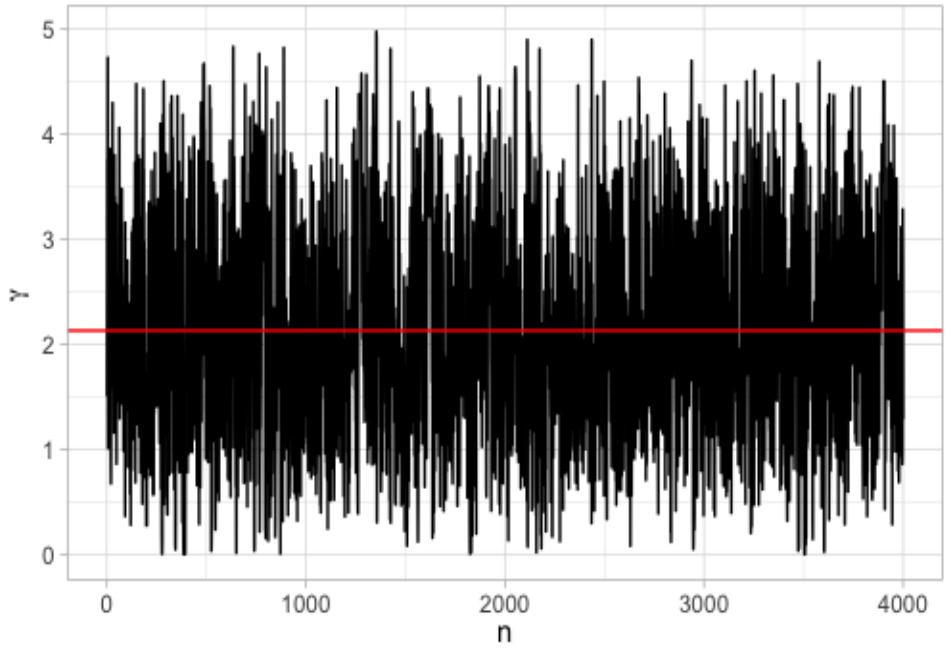


Figure 4.4: Posterior samples of γ. The horizontal red line represents the mean.

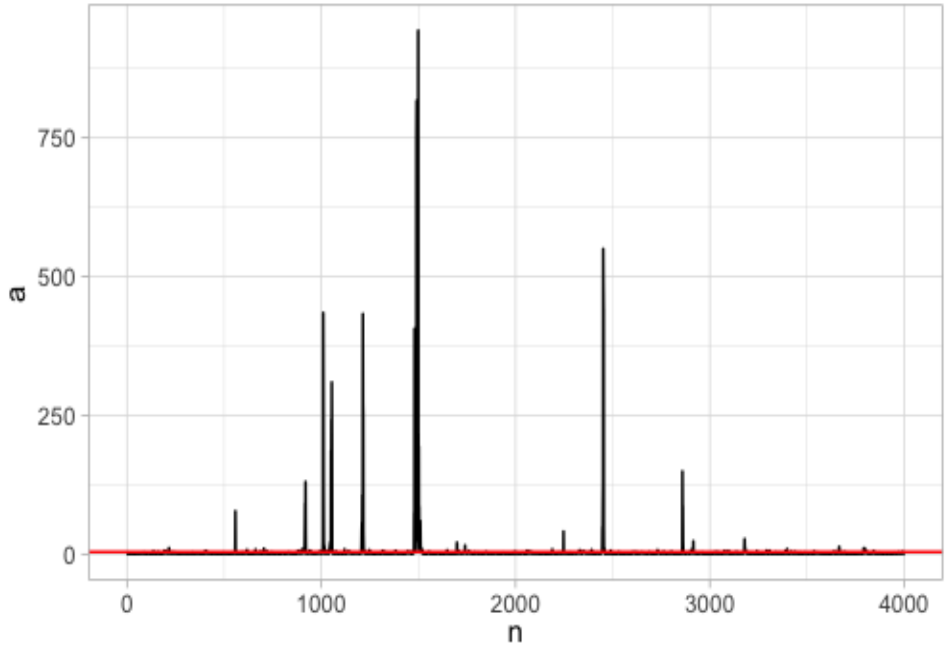


Figure 4.5: Posterior samples of a. The horizontal red line represents the mean.

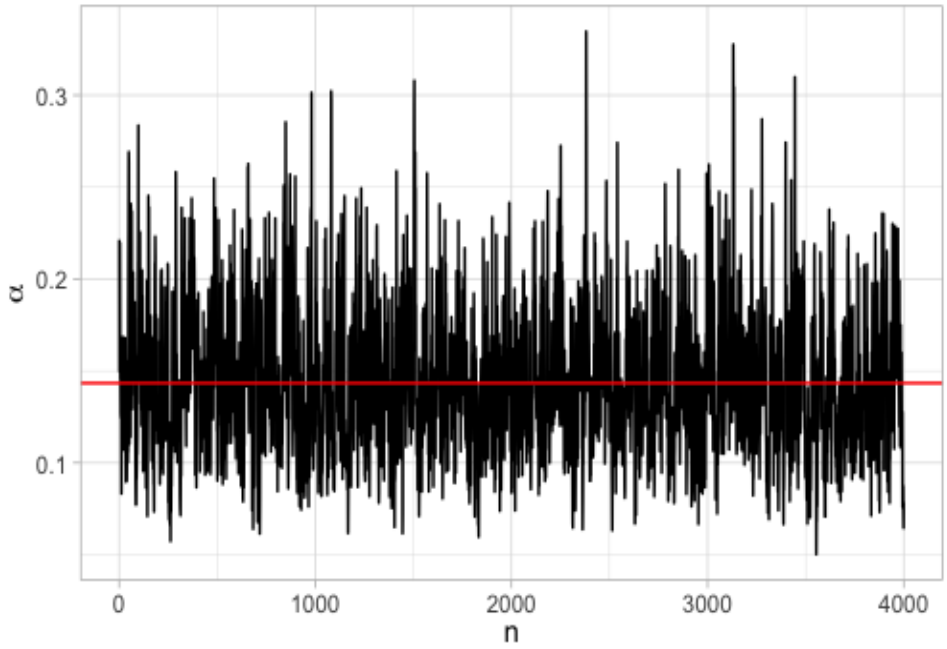


Figure 4.6: Posterior samples of α. The horizontal red line represents the mean.

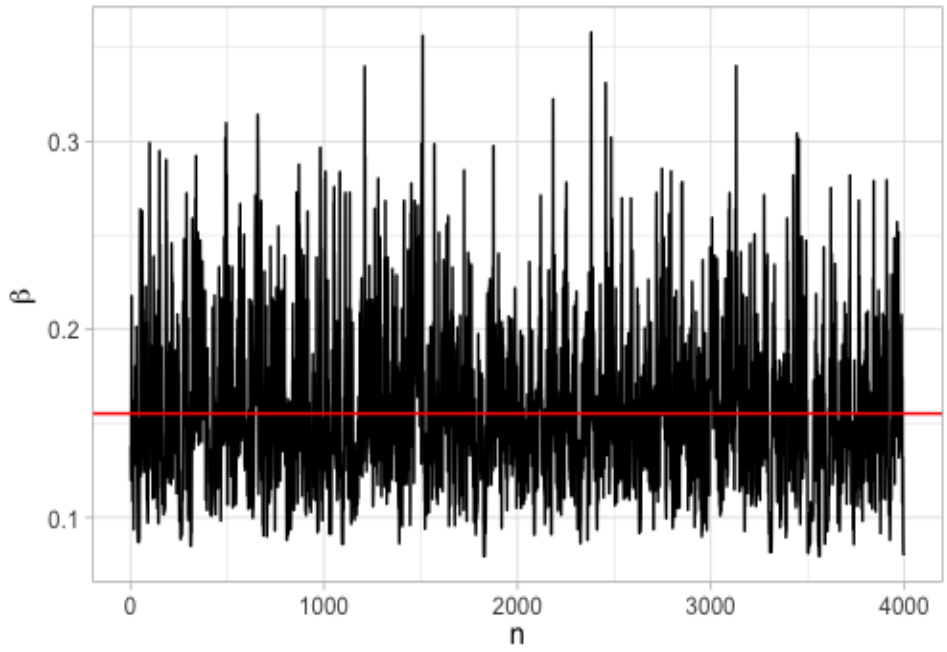
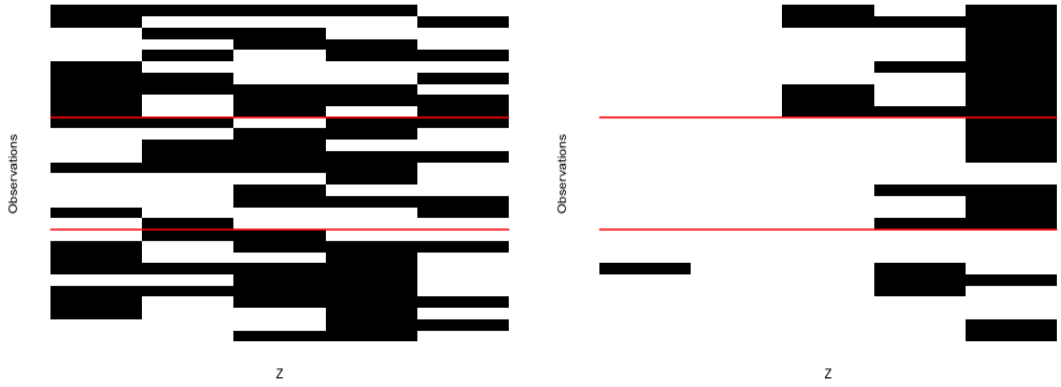


Figure 4.7: Posterior samples of β. The horizontal red line represents the mean.



(a) Initial latent traits. (b) Posterior latent traits.

Figure 4.8: Evolution of the dynamics of the binary matrix of latent traits Z.

Interesting conclusions can be derived from this example. First, it is clear that we are able to discover the true underlying latent traits. In this direction it is compelling to notice that they do not appear in the original order; however, due to the exchangeability framework this does not represent an issue. Secondly, we can also appreciate that the model is able to recover close estimates of the real values for most of the parameters, with the exception of a and c which still have journeys far from their original values. Interestingly, these are two characteristics that we have constantly observed in the multiple MCMC runs tried so far without regard of the initial values and the total number of simulations.

A second attractive approach to the beta-CoRM BMF is by letting, just as for the generalised beta-CoRM, each latent trait ω_l to have a unique score parameter a_l with common gamma(κ, ψ) prior distribution. It is straightforward to see that each a_l has a posterior gamma distribution with parameters $(d + .001, .001 - \sum_j \log(m_{jl}))$. And as previously mentioned, by giving gamma(.001,.001) priors to κ and ψ we have a conjugate model on ψ, whereas for κ an adaptive Metropolis-Hastings step could be used on its logarithmic transformation. This is something that we explore on the following section on a real malware data set.

4.6 Malware data set

In this section we present a preliminary analysis of the beta-CoRM BMF on a real malware data set. Due to the computational cost of this model, we have decided to start with a smaller subset compared to the beta-CoRM models described in Chapter 3 and Chapter 4. In this case we are using a balanced data set composed of 270 malware divided into the nine families considered previously and for which 335 unique 4-grams have been extracted. The graphical representation of the data can be seen in Figure 4.9. We can again appreciate how some of the families have a well-defined structure that allows us to differentiate them. However, it is also true that there are other families that seem to have a strong overlapping in their observations. Therefore, making this data set particularly interesting for the beta-CoRM BMF model to try to discover some underlying structure.

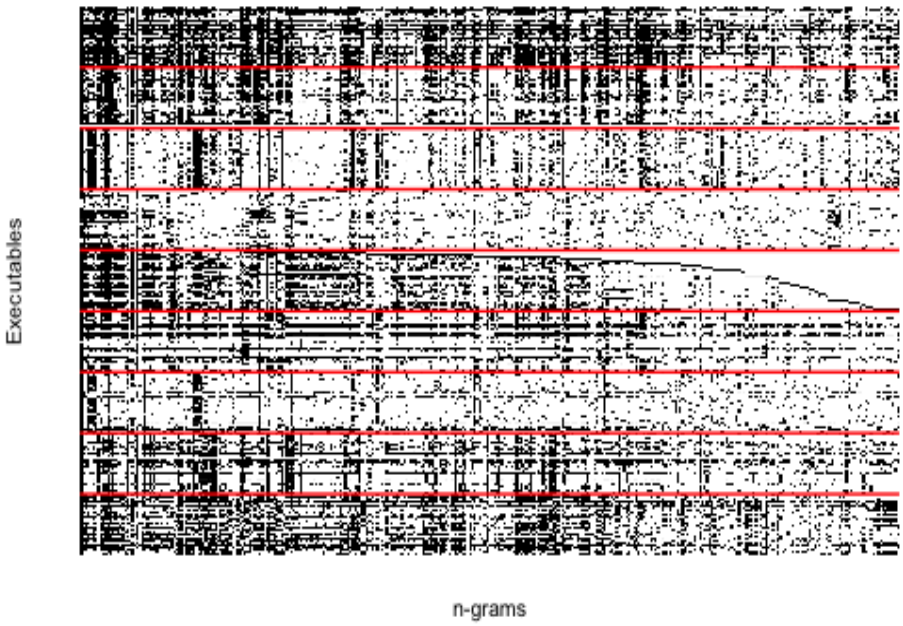


Figure 4.9: Graphical representation of the malware data composed of 270 observations and 335 features divided evenly into nine families separated by the red lines.

Due to the significant increment on the number of observations and features, we have found that the computational cost increases drastically. This is due to the large number of parameters and auxiliary variables that need to be sampled at each round. That is why we decided to use 5,000 iterations as our burning period and 7,500 samples from the posterior with a thinning of 15, yielding an effective sample size of 500 observations. At this point and now that we have introduced all the models considered in this thesis, that is, beta-CoRM, generalised beta-CoRM and the beta-CoRM BMF we believe it is an interesting exercise to demonstrate the increase on the computational cost. In Table 4.1 we present the time required for each of the three models to obtain a sample of the posterior inference using the MCMC specifications just described for the small malware data set illustrated in Figure 4.9.

	beta-CoRM	generalised beta-CoRM	beta-CoRM BMF
Time	20 sec	40 sec	3600 sec

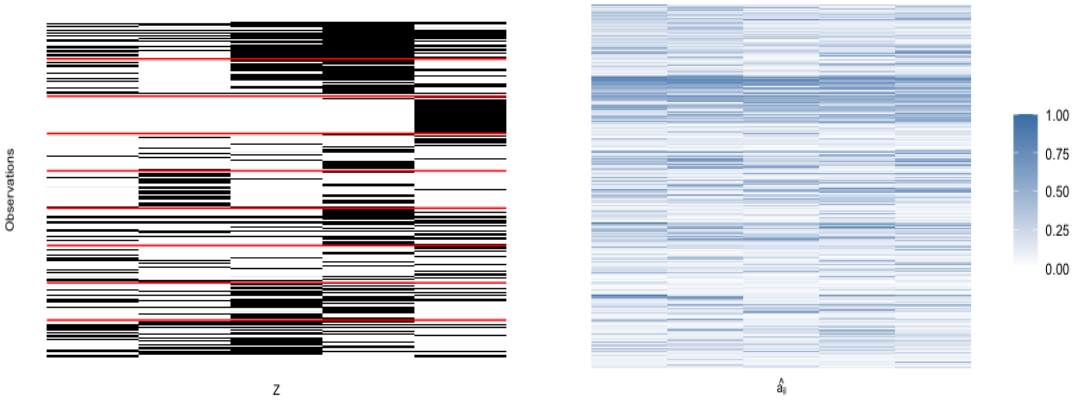
Table 4.1: Comparison of the time (in seconds) required for the beta-CoRM, generalised beta-CoRM and beta-CoRM BMF MCMC algorithms to obtain 12,500 rounds of simulations using the small malware data set (Figure 4.9).

It is clear from Table 4.1 the significant increase of the computation cost required for the beta-CoRM BMF model to obtain a sample from the posterior distribution. That is why until now we have only considered applying the beta-CoRM BMF model to the small malware data set. However, a final interesting comparison between the beta-CoRM and the generalised beta-CoRM can be made since both of them were applied to the complete malware data set illustrated in Figure 2.7. In this direction, the reader can appreciate in Table 4.2 the time required for these two models to obtain a posterior sample by running the MCMC algorithms for a total of 60,000 simulations.

	beta-CoRM	generalised beta-CoRM
Time	400 sec	950 sec

Table 4.2: Comparison of the time (in seconds) required for the beta-CoRM and the generalised beta-CoRM MCMC algorithms to obtain 60,000 rounds of simulations using the complete malware data set (Figure 2.7).

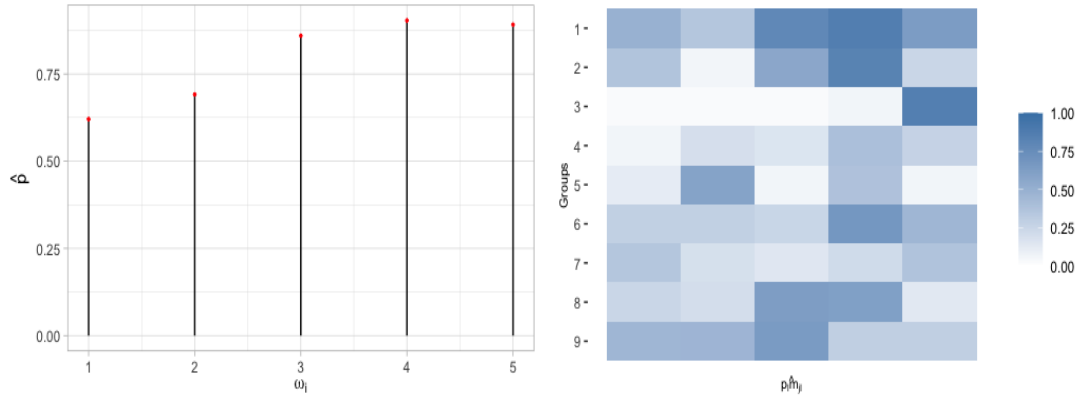
Turning back our attention to the results of the beta-CoRM BMF, we refer the reader to Figure 4.10 where the inferred latent traits and the estimated loadings are displayed.



(a) Latent traits. (b) Loadings.

Figure 4.10: The inferred latent traits and loadings for the malware data set described in Figure 4.9.

Interesting remarks can be made from Figure 4.10a. For example, we can appreciate how only one latent trait is present in the third family, on the other hand, the second family contains all but one of the latent traits. We can further appreciate, how observations in dense families such as the first and the last one also share most of the latent traits. Although, in the first family it is clear that one of the latent traits is always present. Furthermore, in Figure 4.11 we display the posterior estimates of the directing weights as well as the perturbations at a group level.



(a) Posterior estimates of the beta process' jumps.

(b) Posterior estimates of the groups' probabilities.

Figure 4.11: The posterior estimates of the weights of the directing beta process and the perturbed weights at a group level for the malware data set described in Figure 4.9.

A final interesting remark is that the posterior estimates of the score parameters $\widehat{a}_l$ for all the latent traits are quite similar to each other. Hence, and following the ideas described in Chapter 4, we could argue that the five latent traits considered are equally important, and in order to appreciate a significant difference, we should consider the presence of more latent traits. The posterior estimates of the score parameters are displayed in Table 4.3.

	$\widehat{a}_1$	$\widehat{a}_2$	$\widehat{a}_3$	$\widehat{a}_4$	$\widehat{a}_5$
Posterior estimate	.9308	.9254	.9233	.9363	.9281

Table 4.3: Posterior mean of the score parameters for the five latent traits considered in the beta-CoRM BMF model.

4.7 Concluding remarks

The beta-CoRM BMF is an interesting and novel approach to binary latent factor models. However, it is clear that there is still work to de carried out in this direction before completely applying this methodology for classification purposes.

From a theoretical point of view, there are several points we need to study more carefully. For example, there is a need to deeply analyse the effect of the parameters and the latent traits in the covariance structure described in equation (4.3). Also, there is a need to perform a truncation error analysis to throughly understand the impact of the truncation process made on the the directing beta process. This is something particularly interesting since it will allow us to determine the optimum number of latent traits to consider for the simulations and the posterior inference. Of course, we acknowledge, as explained in Section 4.1, that there exist alternatives to deal with the infinite representation of the beta process. In particular, we found attractive the idea of splitting the full conditional into a finite and an infinite part since it would allow us to consider a random number of latent traits and hence, no truncation process would be required.

As for the computational part, we need to have a deeper look into the MCMC procedure due to the heavy computational cost of the sampler. In this direction it could be particularly interesting to use directly the posterior distribution of the loadings described in equation (4.14), rather than using the second set of latent traits $\{v_{kjil}\}$. This is something that would dramatically reduce the number of variables to sample at each round, and with it a faster mixing could probably be achieved. Also there is a need to study other prior distributions for the mass and the concentration parameter. In practice the gamma distribution has been considered when the stick-breaking representation of the beta process is used due to its conjugacy. However, in our modelling, this conjugacy is not present, and hence, we could perhaps consider a different prior. In this direction another possibility would be to find optimal values for c and γ using a similar procedure as in Chapter 2 where optimal values of a and c were found by leveraging on the classification performance of the model.

Chapter 5

Future work

The Bayesian methods developed in this thesis are an attractive approach to discrimination tasks. We have found them to be particularly useful for supervised learning in situations where groups do not have a strong overlapping. However, we have also identified future areas of research with respect the beta-CoRM models presented in Chapters 2, 3 and 4. That is why in this chapter we present the future work we intend to realise to improving the beta-CoRM's theory and applicability to other cyber security detection problems. In this chapter, we also discuss a second Bayesian nonparametric factorial approach for the binary data that uses the Indian buffet process (IBP) as a prior. Finally, we also explore a second approach to malware detection that relies on the number of times a feature appears and what we think would be two interesting approaches to modelling this data.

5.1 Beta-CoRM models

For the beta-CoRM models there are several lines of future work that we are aiming to tackle in the forthcoming months. As we can recall from Chapters 2 and 3, the construction of the discrete beta-CoRM is based on the underlying theory of compound random measures (Griffin and Leisen, 2017). This approach is suitable for supervised learning and hence for discrimination tasks. However, there are still some interesting challenges that we would like to discuss and keep studying.

First of all, we are aiming to discuss more in depth the prior sensitive analysis. This is an important theoretical aspect that could provide insightful information on this stochastic process and hence on its modelling capabilities. This is particularly interesting due to the fact that we have used a discrete base measure; however, as seen in Chapter 4, this can be easily extended to consider a continuous measure instead. With this in mind, and using suitable inferential techniques, we would allow the discovery of new unobserved features. For a static approach to malware analysis, this would be particularly interesting due to the increasing number of malware in the wild, especially of obfuscated malware where normal code has been injected to avoid detection.

The second line of future research for the beta-CoRM model is to extend it to other class of cyber attacks, especially for the ones that rely on binary features like the n-grams. In this direction, it would be particularly attractive to use this Bayesian approach to n-gram profiles to detect masquerade attacks. In this kind of attacks, a person or computer uses a fake identity to gain unauthorised access to sensitive information using legitimate usernames and credentials. In practice, these attacks can be carried out through the means of a UNIX command line, then

the idea is to detect anomalies by comparing the commands that are being used to past normal behaviour. In order to detect intrusion detection at a command line, one could rely on the frequency of the commands, the transitions among them or through n-gram profiles like in Geng et al. (2011).

5.2 A Bayesian latent logistic model with binary predictors

In statistics, logistic regression analysis is used to model the probability of an event occurring that depends on a set of independent variables, also known as predictors and henceforth denoted by $\mathbf{Z} = \{z_{ij}\}_{i,j}$. Depending on the application, these predictors can be continuous, discrete or a mix of both. However, for our purposes we centre our attention on pure binary predictors, which indicate whether the i-th observation has the j-th characteristic. In medical studies, this is particularly useful for modelling the probability of an individual having (or being prone) to a disease depending on the gender, on underlying health conditions (e.g., heart disease, diabetes, etc.) among other binary characteristics. The dichotomic nature of the response variable and the predictors makes the logistic approach an ideal model for malware detection using n-grams. However, from Chapter 2 we can recall that the logistic regression model had some limitations with a real malware detection data set due to some predictors having null variation.

Fortunately, for the detection and classification of malware, the logistic regression can be used to model the probability of an observation having or not an n-gram. This can be achieved by using as predictors, $\mathbf{Z}$, a set of K latent variables that can be either continuous or discrete. A particularly interesting approach is to consider binary latent variables which can then be used to model

the probability of a vector of responses $\{x_i\}_{i=1}^N$ through the equation

$$\mathbb{P}(x_i = 1|\mathbf{Z}, w) = \frac{\exp(z_i w^T)}{1 + \exp(z_i w^T)}, \tag{5.1}$$

where $w = \{w_i\}_{i=1}^K$ is a vector of parameters that characterises the latent factors.

In some applications the number of K latent features is fixed and finite; however, from a Bayesian nonparametric approach it is usually assumed that the number of latent features is unbounded. This assumption yields a binary matrix $\mathbf{Z}$ with N rows and an infinite number of columns. In order to define a prior for this infinite-dimensional object, we will rely on the Indian buffet process since it will allow us to develop a collapsed Gibbs sampling for the posterior inference like in Ghahramani and Griffiths (2006).

5.2.1 The finite model

In order to define a Bayesian nonparametric approach that uses the IBP as a prior on $\mathbf{Z}$ for a logistic regression model, we first define a finite version of it. Hence, we assume that each object possesses the feature k with probability p_k and that it is independent from other features. This yields the following distribution on binary matrices

$$f(\mathbf{Z}|p) = \prod_{k=1}^{K} p_k^{m_k}(1 - p_k)^{N - m_k}, \tag{5.2}$$

where m_k is the number of observations having the k-th feature, that is $m_k = \sum_{i=1}^N z_{ik}$. We further assume a beta prior on $\mathbf{p}$, that in the first formulation of

the IBP is defined to be

$$p_k \sim \text{beta}\left(\frac{\alpha}{K}, 1\right),$$

where α is the mass parameter of the underlying beta process. This beta prior allows us to integrate out the vector of probabilities $\mathbf{p}$ in (5.2) so that

$$\mathbb{P}(z_{ik} = 1|\mathbf{Z}_{-(ik)}) = \frac{m_{-ik} + \frac{\alpha}{K}}{N + \frac{\alpha}{K}}, \tag{5.3}$$

where m_{-ik} are the set of observations having the k-th feature, without taking into account the i-th observation. In the infinite dimensional case this expression can be further simplified to $\mathbb{P}(z_{ik} = 1|\mathbf{Z}_{-(ik)}) = \frac{m_{-ik}}{N}$.

Bayesian inference on the logistic model can not be performed directly. However, a Polya-gamma augmentation technique can be used in order to have fast inference (Polson et al., 2013). This method allows us to express the likelihood of a single response (5.1) as

$$f(x_i|\mathbf{Z}, w, \theta_i) = 2^{-1} \exp\left[\left(x_i - \frac{1}{2} z_i w^T\right)\right] \exp\left[-\theta_i \frac{(z_i w^T)^2}{2}\right], \tag{5.4}$$

where θ_i is a Polya-gamma (1,0) random variable. Hence, the complete likelihood is given by

$$f(\mathbf{X}|\mathbf{Z}, w, \theta) = \prod_{i=1}^{N} 2^{-1} \exp\left[\left(x_i - \frac{1}{2} z_i w^T\right)\right] \exp\left[-\theta_i \frac{(z_i w^T)^2}{2}\right]. \tag{5.5}$$

Equation (5.5) allows us to choose a conjugate model for w by choosing a multivariate Gaussian distribution with mean vector μ and covariance matrix Σ. This will be particularly useful if the IBP is used as a prior in order to derive a collapsed Gibbs sampling procedure.

The posterior distributions for θ and w can be derived using some of the key properties of the Polya-gamma augmentation technique, yielding,

$$\begin{aligned} \theta_i | w, \mathbf{Z}, \mathbf{X} &\sim PG(1, z_i w^T) \\ w | \theta, \mathbf{Z}, \mathbf{X} &\sim N(\Sigma_1 \mu_1, \Sigma_1), \end{aligned} \tag{5.6}$$

where $\Sigma_1 = (\mathbf{Z}^T \Theta \mathbf{Z} + \Sigma^{-1})^{-1}$, $\mu_1 = \Sigma_1(\mathbf{Z}^T \kappa + \Sigma^{-1}\mu)$, $\Theta = \text{diag}(\theta_1, ..., \theta_N)$ and $\kappa = \{\kappa_i\}_i = \{x_i - 1/2\}_i$. As we discuss in the following subsection, for the infinite dimensional case it will be useful for computational purposes to set $\mu = 0$ and $\Sigma = \sigma \text{I}$.

In order to update $\mathbf{Z}$ we notice that due to the conjugacy we are able to integrate out the parameters $\mathbf{w}$ and with this in mind we have the following posterior probability (up to proportionality)

$$\mathbb{P}(z_{ik} = 1 | \mathbf{X}, \mathbf{Z}_{-(ik)}, \theta) \propto \left(\frac{m_{-ik} + \frac{\alpha}{K}}{N + \frac{\alpha}{K}} \right) \mathbb{P}(\mathbf{X} | \mathbf{Z}, \theta),$$

where

$$\begin{aligned} \mathbb{P}(\mathbf{X} | \mathbf{Z}, \theta) &= \int \mathbb{P}(\mathbf{X} | \mathbf{Z}, \theta, w) f(w) dw \\ &= \int \left(\prod_{i=1}^{N} 2^{-1} \exp\left[\left(x_i - \frac{1}{2} \right) z_i w^T \right] \exp\left[-\theta_i \frac{(z_i w^T)^2}{2} \right] \right) \\ &\times \frac{1}{(2\pi)^{K/2} \sigma^{K/2}} \exp\left[-\frac{1}{2} (w^T (\sigma \text{I})^{-1} w \right] dw \end{aligned}$$

$$
\begin{aligned}
&= \int \exp\left[-\frac{\theta_i}{2}\left(z_i w^T - \frac{(x_i - \frac{1}{2})}{\theta_i}\right)^2\right] \exp\left[-\frac{1}{2} w^T (\sigma \mathrm{I})^{-1} w\right] dw \\
&\times \frac{\prod_{i=1}^N \exp\left[\frac{(x_i - \frac{1}{2})^2}{2\theta_i}\right]}{2^N (2\pi)^{K/2} \sigma^{K/2}} \\
&= \int \exp\left[-\frac{1}{2}[(y - \mathbf{Z}w)^T \Theta^{-1} (y - \mathbf{Z}w) + w^T (\sigma \mathrm{I})^{-1} w]\right] dw \\
&\times \frac{\prod_{i=1}^N \exp\left[\frac{(x_i - \frac{1}{2})^2}{2\theta_i}\right]}{2^N (2\pi)^{K/2} \sigma^{K/2}},
\end{aligned}
\tag{5.7}
$$

with $y = \{y_i\} = \{(x_i - \frac{1}{2})\theta_i\}_i$. The term inside the exponential in (5.7) can be further written as

$$
\begin{aligned}
& (y - \mathbf{Z}w)^T \Theta^{-1} (y - \mathbf{Z}w) + w^T (\sigma \mathrm{I})^{-1} w \\
&= w^T (\mathbf{Z}^T \Theta^{-1} \mathbf{Z} + (\sigma \mathrm{I})^{-1}) w - 2 w^T (\mathbf{Z} \Theta^{-1} y) + y^T \Theta^{-1} y \\
&= w^T \Sigma_1^{-1} w - 2 \mu_1^T w + y^T \Theta^{-1} y \\
&= (w - \Sigma_1 \mu_1)^T \Sigma_1^{-1} (w - \Sigma_1 \mu_1) - \mu_1^T \Sigma_1 \mu_1 + y^T \Theta^{-1} y,
\end{aligned}
$$

where $\Sigma_1 = (\mathbf{Z}^T \Theta^{-1} \mathbf{Z} + (\sigma \mathrm{I})^{-1})^{-1}$ and $\mu_1 = \mathbf{Z}^T \kappa$. Hence (5.7) can be finally expressed as

$$
\begin{aligned}
\mathbb{P}(\mathbf{X}|\mathbf{Z}, \theta) &= \frac{|\Sigma_1|^{1/2} \prod_{i=1}^N \exp\left[\frac{(x_i - \frac{1}{2})^2}{2\theta_i}\right]}{2^N \sigma^{K/2}} \exp\left[\frac{1}{2} \mu_1^T \Sigma_1 \mu_1 - \frac{1}{2} y^T \Theta^{-1} y\right] \\
&\times \int \frac{1}{(2\pi)^{K/2} |\Sigma_1|^{1/2}} \exp\left[-\frac{1}{2}(w - \Sigma_1 \mu_1)^T \Sigma_1^{-1} (w - \Sigma_1 \mu_1)\right] dw \\
&= \frac{|(\mathbf{Z}^T \Theta^{-1} \mathbf{Z} + (\sigma \mathrm{I})^{-1})|^{-1/2}}{2^N \sigma^{K/2}} \prod_{i=1}^N \exp\left[\frac{(x_i - \frac{1}{2})^2}{2\theta_i}\right] \exp\left[-\frac{1}{2} y^T \Theta^{-1} y\right] \\
&\times \exp\left[\frac{1}{2} \kappa^T \mathbf{Z} (\mathbf{Z}^T \Theta^{-1} \mathbf{Z} + (\sigma \mathrm{I})^{-1})^{-1} \mathbf{Z}^T \kappa\right].
\end{aligned}
\tag{5.8}
$$

5.2.2 The infinite model

If we want to analyse what happens when $K \to \infty$ we need to show that (5.8) is well defined. In expression (5.8) $\mathbf{Z}$ appears two times and we can analyse them separately. In order to do so, we shall write $K = [K_+, K_0]$ where K_+ represents the features for which $m_k > 0$ and K_0 the features for which $m_k = 0$. This allows us to write $\mathbf{Z} = [\mathbf{Z}_+, \mathbf{Z}_0]$ where $\mathbf{Z}_0$ has zeros in all the entries. Therefore,

$$\begin{aligned}
& |(\mathbf{Z}^T\Theta^{-1}\mathbf{Z} + (\sigma \mathrm{I})^{-1})|^{1/2} \\
= & \left| \begin{pmatrix} \mathbf{Z}_+^T \\ \mathbf{Z}_0^T \end{pmatrix} \Theta^{-1} \begin{pmatrix} \mathbf{Z}_+ & \mathbf{Z}_0 \end{pmatrix} + \begin{pmatrix} \sigma^{-1}\mathrm{I}_{K_+} & 0 \\ 0 & \sigma^{-1}\mathrm{I}_{K_0} \end{pmatrix} \right|^{1/2} \\
= & \left| \begin{pmatrix} \mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+} & 0 \\ 0 & \sigma^{-1}\mathrm{I}_{K_0} \end{pmatrix} \right|^{1/2} \\
= & |(\mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+})|^{1/2}\sigma^{-K_0/2}.
\end{aligned} \tag{5.9}$$

The appearance of $\sigma^{-K_0/2}$ in (5.9) will not be a problem since this expression is multiplied by $\sigma^{K/2} = \sigma^{K_+/2}\sigma^{K_0/2}$ and hence canceled out leaving only an expression depending on the number of used features K_+.

The second part in (5.8) that we need to analyse is $\mathbf{Z}(\mathbf{Z}^T\Theta^{-1}\mathbf{Z} + \sigma^{-1}\mathrm{I})^{-1}\mathbf{Z}^T$. This expression can be easily simplified due to the large number of zeros present as

$$\begin{aligned}
& \mathbf{Z}(\mathbf{Z}^T\Theta^{-1}\mathbf{Z} + \sigma^{-1}\mathrm{I})^{-1}\mathbf{Z}^T \\
= & \begin{pmatrix} \mathbf{Z}_+ & \mathbf{Z}_0 \end{pmatrix} \begin{pmatrix} \mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+} & 0 \\ 0 & \sigma^{-1}\mathrm{I}_{K_0} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{Z}_+^T \\ \mathbf{Z}_0 \end{pmatrix}
\end{aligned}$$

$$
\begin{aligned}
&= \begin{pmatrix} \mathbf{Z}_+ & \mathbf{Z}_0 \end{pmatrix} \begin{pmatrix} (\mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+})^{-1} & 0 \\ 0 & (\sigma^{-1}\mathrm{I}_{K_0})^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{Z}_+^T \\ \mathbf{Z}_0 \end{pmatrix} \\
&= \mathbf{Z}_+(\mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+})^{-1}\mathbf{Z}_+^T.
\end{aligned}
\tag{5.10}
$$

Finally we can note that

$$
\prod_{i=1}^{N} \exp\left[\frac{(x_i - \frac{1}{2})^2}{2\theta_i}\right] = \exp\left[\frac{1}{2}\kappa^T\Theta^{-1}\kappa\right]
$$

and

$$
\exp\left[-\frac{1}{2}y^T\Theta^{-1}y\right] = \exp\left[-\frac{1}{2}\kappa^T\Theta\kappa\right].
\tag{5.11}
$$

Therefore, using (5.9), (5.10) and (5.11) we obtain that the $\mathbb{P}(\mathbf{X}|\mathbf{Z},\theta)$ is equal to

$$
\begin{aligned}
&\frac{1}{2^N\sigma^{K_+/2}|(\mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+})|^{1/2}} \exp\left[\frac{1}{2}\kappa^T\Theta^{-1}\kappa\right] \exp\left[-\frac{1}{2}\kappa^T\Theta\kappa\right] \\
\times\ &\exp\left[\frac{1}{2}\kappa^T\mathbf{Z}_+(\mathbf{Z}_+^T\Theta^{-1}\mathbf{Z} + \sigma^{-1}\mathrm{I}_{K_+})^{-1}\mathbf{Z}_+^T\kappa\right] \\
=\ &\frac{\exp\left[\frac{1}{2}\kappa^T[\Theta^{-1} - \Theta + \mathbf{Z}_+(\mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+})^{-1}\mathbf{Z}_+^T]\kappa\right]}{2^N\sigma^{K_+/2}|(\mathbf{Z}_+^T\Theta^{-1}\mathbf{Z}_+ + \sigma^{-1}\mathrm{I}_{K_+})|^{1/2}},
\end{aligned}
\tag{5.12}
$$

and hence,

$$\begin{aligned} & \mathbb{P}(z_{ik} = 1 | \mathbf{X}, \mathbf{Z}_{-(ik)}, \theta) \\ \propto \quad & \left(\frac{m_{-ik}}{N}\right) \frac{\exp\left[\frac{1}{2}\kappa^T [\Theta^{-1} - \Theta + \mathbf{Z}_+ (\mathbf{Z}_+^T \Theta^{-1} \mathbf{Z}_+ + \sigma^{-1} \mathrm{I}_{K_+})^{-1} \mathbf{Z}_+^T] \kappa\right]}{2^N \sigma^{K_+/2} |(\mathbf{Z}_+^T \Theta^{-1} \mathbf{Z}_+ + \sigma^{-1} \mathrm{I}_{K_+})|^{1/2}}. \end{aligned} \tag{5.13}$$

Finally, in the infinite dimensional case we also need to consider adding k new features when updating the corresponding set of latent variables for each observation. This is done with probability proportional to

$$\mathbb{P}(k) \quad \propto \quad \text{Pois}\left(k; \frac{\alpha}{N}\right) \mathbb{P}(\mathbf{X} | \mathbf{Z}_{\text{new}}, \theta), \tag{5.14}$$

where $\mathbf{Z}_{\text{new}}$ has k new columns in the i-th row all of them equal to 1. For these k new features we also need to consider sampling values from the prior since they will be required to update each θ_i. Hence, the following Gibbs sampling algorithm can be used for the posterior inference.

1. Initialise θ_i, w and $\mathbf{Z}$.

2. At step n,

   (a) sample $\theta_i^{(n)} \sim PG(1, z_i^{(n-1)} (w^{(n-1)})^T)$,

   (b) sample $w^{(n)} \sim N(\Sigma_1 \mu_1, \Sigma_1)$

   (c) update $\mathbf{Z}$ by sampling each entry using equation (5.13) and add new features through (5.14).

This procedure can be easily applied to other likelihoods, for example if we consider X_i to be a vector of M independent binary responses, i.e.,

$$\begin{aligned}\mathbb{P}(X_i|\mathbf{Z}, w) &= \prod_{j=1}^{M}\left(\frac{\exp(z_i w^T)}{1+\exp\left(z_i w^T\right)}\right)^{x_{ij}}\left(\frac{1}{1+\exp(z_i w^T)}\right)^{1-x_{ij}} \\ &= \frac{\exp(z_i w^T)^{n_i}}{(1+\exp(z_i w^T)^M},\end{aligned}$$

then we can apply the Polya-gamma augmentation technique with Polya-gamma random variables of parameters $(M, 0)$ and with $\kappa_i = n_i - M/2$. This likelihood would be certainly useful for the malware analysis through n-grams since each malware is completely characterised by the M binary features.

5.3 An n-grams counts approach to malware analysis

Throughout this thesis we have explored novel supervised Bayesian methodologies for binary data with specific applications to malware detection and classification through feature-engineered n-grams. With these probabilistic approaches, interesting results have been obtained regarding not only the classification task but also on the generative process of the data and the importance of the features across and within families. Moreover, further research opportunities in this direction have been gently described in the previous sections of this chapter. However, we would also like to describe another possibility on the analysis of malware, where the interest relies not only on whether a feature is present or not, but on the number of times it appears.

One of the main motivations of this counting approach is to overcome the overlapping present for some families while considering binary factors (Figure 2.7). In order to appreciate this more clearly, the reader can refer to Figure 5.1, where the presence intensity of each feature across families is displayed, that is, $\log_{10}(X_{ij}+1)$. This transformation is required since it would be impossible to appreciate the differences among families by directly plotting the counts due to the presence of features with a large count value.

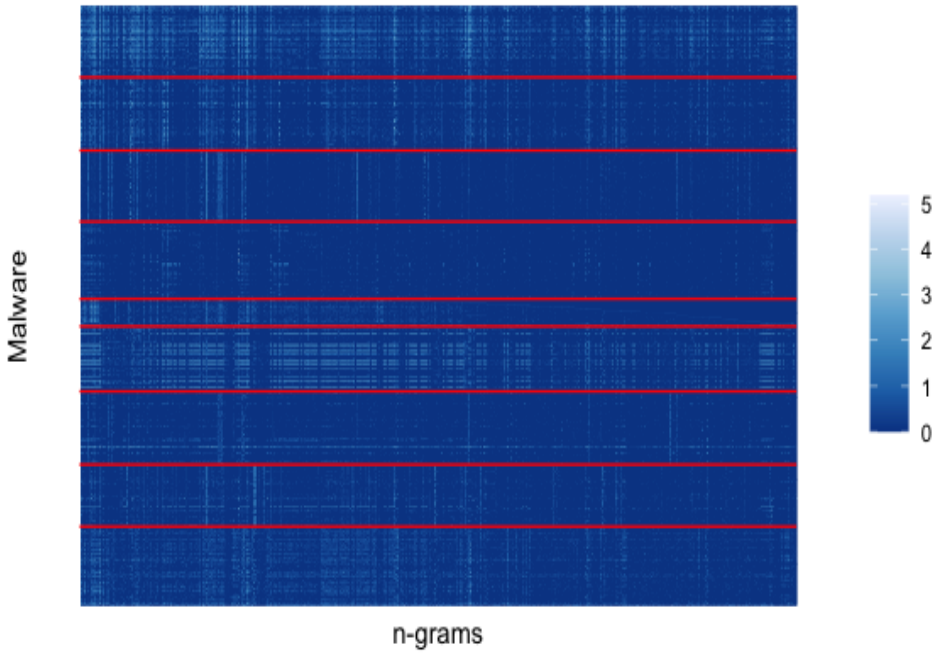


Figure 5.1: Intensity of each feature calculated as $\log_{10}(X_{ij}+1)$ across each of the nine families of malware separated by the red lines.

Looking at the intensities, it is clear that the overlapping among families might be diminished and hence, there is a real possibility of designing a supervised learning model with improved classification accuracy. In order to model the counts, there is a need to deal with the features that have a large count. One straightforward possibility is to introduce a truncation process. This is certainly feasible since the number of features having a large count represent a small

proportion of the data. For example, for the data illustrated in Figure 5.1 only 4.689% of the entries are greater than 10, hence, we could consider truncating the data at this value. The results of this truncation process are displayed in Figure 5.2.

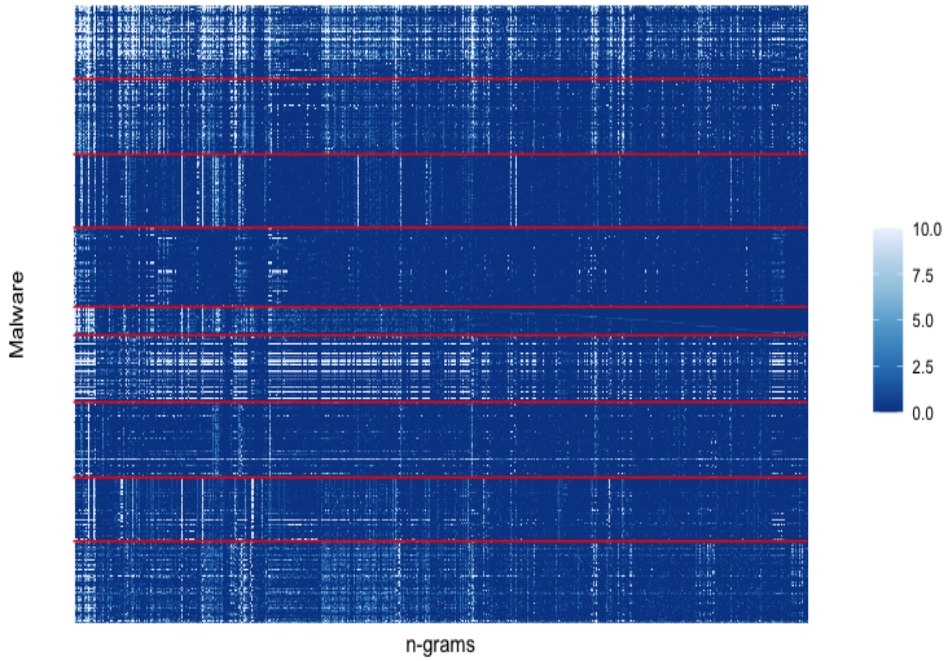


Figure 5.2: Matrix of truncated counts (at 10) across each of the nine families of malware separated by the red lines.

Looking at Figure 5.2, we can appreciate that overlapping seems to be diminished. Therefore, taking into account the number of times the feature appears might be a sensible choice to differentiate among the different families of malware. In order to develop a probabilistic model for this counts data, a natural first approach would be to consider a Poisson model with gamma parameters. In order to assess this assumption we can first obtain the index of dispersion for each feature across family as displayed in Figure 5.3.

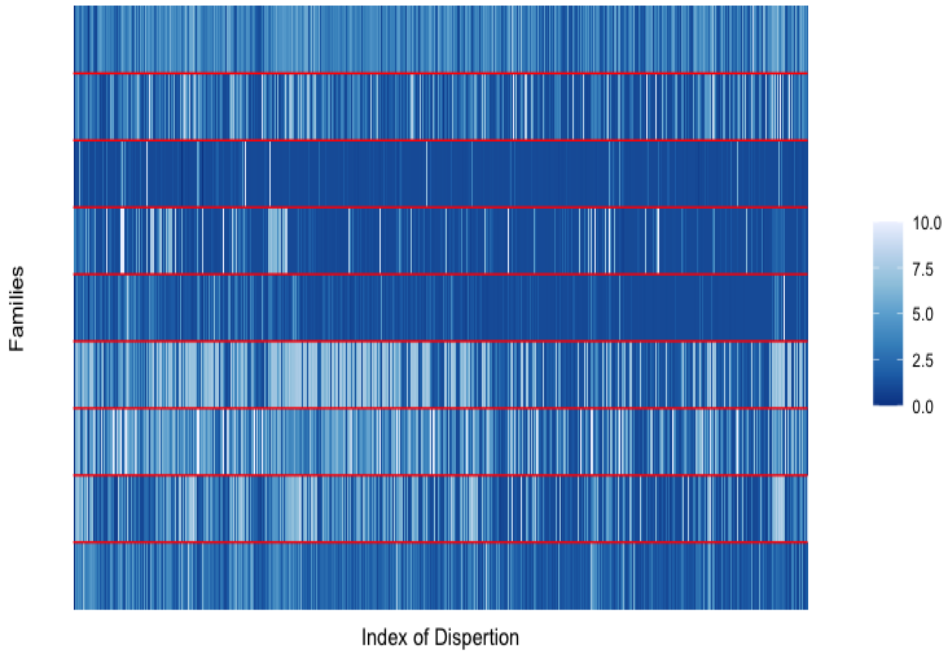


Figure 5.3: Index of dispersion for each feature across families.

If the Poisson distribution were to fit the data, the index should be close to 1, something that is clearly not happening except for families 3 and 5. The overdispersion found in the rest of the families can be easily explained by both the large number of zeros and the truncation procedure applied to the data. Hence, to properly model this data, one possibility would be to consider a mixture model that would allows us to capture both the zero-inflated part and the heavy tail part, as we further discuss in Section 5.3.1. Otherwise, we could follow a more direct approach by considering a categorical distribution on the data, as we further discuss in Section 5.3.2.

5.3.1 A zero-inflated truncated Poisson model

In statistics, overdispersion is usually thought as the presence of higher variability on the data than the one expected once a model has be assumed. In count data, overdispersion is usually concerned when we observe a higher amount of zeros compared to the non-zero ones. In order to model this kind of behaviour a whole theory on zero-inflated discrete distributions has been developed, the reader could refer to Lambert (1992), Ridout et al. (1998), Rodrigues (2003) and Ghosh et al. (2006) for an account on both frequentist and Bayesian approaches to zero-inflated distributions and their use in regression models. It is without a doubt interesting to notice that these discrete distributions are usually written as $V(1-B)$ where B is a Bernoulli random variable and V is a counting random variable such as Poisson, geometric or any other power series distribution.

These distributions have been certainly useful in order to model one-sided overdispersion; however, in other applications such as the malware count data, where a truncation process was performed, we would also need to consider overdispersion appearing in the tails of the distribution. To deal with this situation, we could consider a truncated version of a zero-inflated distribution. For the purposes of this section we restrict our attention to the Poisson assumption, and discuss a general Bayesian framework for the inferential process.

Henceforward, whenever we mention a truncated Poisson distribution (TPD) we are going to be referring to the upper tail truncation, that is, a discrete probability distribution on $\{0, ..., n\}$ where n is the level of the truncation, and

with probability mass function given by

$$\mathbb{P}(X = x) = \begin{cases} \frac{\theta^x e^{-\theta}}{x!} & \text{if } x \in \{0, ..., n-1\} \\ e^{-\theta} \sum_{x=n}^{\infty} \frac{\theta^x}{x!} & \text{if } x = n. \end{cases}$$

Using this truncation process on a zero-inflated distribution should allow us to capture the behaviour observed in the malware count data set. In particular, for a zero-inflated Poisson distribution we obtain the following probability mass function

$$\begin{aligned} &\mathbb{P}(X = 0|\theta) = p + (1-p)e^{-\theta} \\ &\mathbb{P}(X = k|\theta) = \frac{(1-p)e^{-\theta}\theta^k}{k!} \qquad\qquad k \in \{1, 2, ..., n-1\} \\ &\mathbb{P}(X = n|\theta) = (1-p)\left(e^{-\theta}\sum_{l=n}^{\infty}\frac{\theta^l}{l!}\right), \end{aligned}$$

where p is the probability of $B = 0$ and whenever $p \to 0$ we recover the truncated Poisson distribution. Now, considering a vector $\mathbf{X} = \{X_1, ..., X_N\}$ of counts and letting $K^i = \sum_i^n \mathbb{1}(X_i = i)$ be the number of observations equal to i the likelihood can be written as:

$$(p + (1-p)e^{-\theta})^{K^0}\left(\prod_{l=1}^{n-1}((1-p)e^{-\theta}\theta^l)^{K^l}\right)\left((1-p)e^{-\theta}\sum_{l=n}^{\infty}\frac{\theta^l}{l!}\right)^{K^n}.$$

In order to deal with this likelihood the data augmentation technique described in Rodrigues (2003) could be considered. In this approach, we let X to

be a random variable with distribution given by

$$\mathbb{P}(X = x|\theta, n) = \begin{cases} \delta_0^x & \text{if} \quad \theta = 0 \\ TPD(x; \theta, n) & \text{if} \quad \theta > 0, \end{cases}$$

and in order to have a zero-inflated model we let a discrete mixing distribution on θ so that $\mathbb{P}(\theta = 0) = p$ and $\mathbb{P}(\theta > 0) = 1 - p$ which yields the following mixture model

$$p(x|\theta, n) \quad = \quad p\delta_0^x + (1-p)TPD(x; \theta, n).$$

Finally, we let K^i to be defined as previously and for each observation we consider an indicator latent variable, I_i, telling us whether the respective observation comes from the degenerate distribution or from the truncated Poisson distribution, i.e.,

$$I_i = \begin{cases} 1 & \text{with probability } q(\theta, p) = \frac{p}{p+(1-p)e^{-\theta}} \\ 0 & \text{with probability } 1 - q(\theta, p). \end{cases}$$

Taking all of this into consideration and letting $S = \sum I_i \sim \text{bin}(K^0, q(\theta, p))$, the augmented likelihood can be finally written (up to proportionality) as

$$p^S(1-p)^{N-S}e^{-\theta(N-S)}\theta^{\sum_{i=1}^{n-1} iK^i}\left(\sum_{l=n}^{\infty}\frac{\theta^l}{l!}\right)^{K^n}. \tag{5.15}$$

From equation (5.15) we can directly observe that a beta prior on p could be used in order to have a conjugate model. As for θ we could be inclined to use a gamma prior; however, a Metropolis-Hastings step needs to be considered due to the last term of the product involving the infinite sum. The Gibbs sampling procedure would require updating S from the binomial distribution and then update p and θ.

As for the malware counts data set with a truncation at 10 counts, we could assume independence across features and use a truncated zero-inflated Poisson distribution on each of them. Hence, and following the notation of the beta-CoRM models, we could define the following hierarchical model

$$\begin{aligned} B_0 &= \sum_i q_i \delta_{\omega_i} \\ B &= \sum_i p_i \delta_{\omega_i} && p_i \sim \text{beta}(cq_i, c(1-q_i)) \\ B_j &= \sum_i m_{ij} p_i \delta_{\omega_i} && m_{ij} \sim \text{gamma}(a, b) \\ X_{kj} &= \sum_i z_{ikj} \delta_{\omega_i}, \end{aligned}$$

with

$$\begin{aligned} \mathbb{P}(Z_{ikj} = 0) &= (1 - p_i) + p_i e^{-m_{ij}} \\ \mathbb{P}(Z_{ikj} = n) &= \frac{p_i e^{-m_{ij}} m_{ij}^n}{n!} && n \in \{1, 2, ..., 9\} \\ \mathbb{P}(Z_{ikj} = 10) &= p_i e^{-m_{ij}} \sum_{l=10}^{\infty} \frac{m_{ij}^l}{l!}. \end{aligned}$$

5.3.2 Multivariate gamma-categorical model

For the multivariate gamma-categorical approach, we follow the same principles of the beta-CoRM models by defining a global random measure that is modified at a group level through a score distribution. In this case, however, the beta process cannot longer be used and we centre our attention on gamma-distributed random variables instead. In this approach we further assume that the counts have been truncated at a predefined level N and the objective is to create a profile for each family on the number of times a feature can appear. At this point we assume that the features are independent and identically distributed and in order to have sharing of information we define the following hierarchical model on each of the M features

$$\begin{aligned} p_i &\sim \text{gamma}(\alpha, \beta) && i \in \{0, ..., N\} \\ m_{ji} &\sim \text{gamma}(a, b) && j \in \{1, ..., d\} \\ x_{kji} &\sim \text{categorical}\left(\{0, ..., 10\}, \left\{\frac{p_i m_{ji}}{\sum_s p_s m_{js}}\right\}_i\right) && k \in \{1, ..., n_j\}. \end{aligned}$$

In this case it is interesting to remark that a normalisation of the random measure is required in order to obtain a probability measure. With this model the complete likelihood is given by

$$\mathbb{P}(\mathbf{X}|\mathbf{m}, \mathbf{p}) \;=\; \prod_{j=1}^{d}\prod_{i=0}^{N}\left(\frac{p_i m_{ji}}{\sum_s p_s m_{js}}\right)^{N_{ji}},$$

where N_{ji} is the number of times that the i-th possible value appears in the j-the group. In order to deal with the normalising constant we can condition the likelihood to a set of latent variables t_j that are gamma distributed with

parameters $(N_{j\cdot}, \sum_s p_s m_{js})$ in such way that

$$\begin{aligned}\mathbb{P}(\mathbf{X}|\mathbf{m},\mathbf{p},\mathbf{t}) &= \prod_{j=1}^{d}\prod_{i=0}^{N}\frac{t_{ji}^{(N_{ji}-1)}}{\Gamma(N_{ji})}\exp\left(-t_{ji}\sum_{s=0}^{N}p_s m_{js}\right)(p_i m_{ji})^{N_{ji}} \\ &= \prod_{j=1}^{d}\prod_{i=0}^{N}p_i^{N_{ji}}m_{ji}^{N_{ji}}\frac{t_{ji}^{(N_{ji}-1)}}{\Gamma(N_{ji})}\prod_{s=0}^{N}\exp\left(-t_{ji}p_s m_{js}\right).\end{aligned} \tag{5.16}$$

Using equation (5.16) we are now able to provide closed expressions for the conditional posterior distributions, since we can identify gamma kernels for the parameters and hence a conjugate model for the p_i's and the scores m_{ji}'s, that is,

$$\begin{aligned}p_i|\mathbf{t},\mathbf{m} &\sim \text{gamma}\left(\alpha+\sum_{j+1}^{d}N_{ji},\beta+\sum_{j=1}^{d}t_{ji}m_{ji}\right) \\ m_{ji}|\mathbf{p},\mathbf{t} &\sim \text{gamma}\left(a+N_{ji},b+t_{ji}p_i\right).\end{aligned}$$

Therefore, a straightforward Gibbs sampling procedure can be used for the inferential procedure. As for the hyperparameters gamma priors could be assigned to each one of them an an adaptive Metropolis-Hastings step could be used in order to update them, just as we did for the beta-CoRM models.

Chapter 6

Conclusions

In a highly connected world, cyber security has become an imperative concern for everyone. Nowadays, no one is completely safe of cyber criminals and new measures need to be taken since traditional detection methods can be overwhelmed due to the increasing number of unknown threats. That is why, statistical and mathematical methods applied to cyber security have become widely popular from a research point of view. In particular, we have observed that there is a large number of approaches based on classical and machine learning methods, and on recent years, there has been an increasing interest on Bayesian models. However, this framework is still not as deeply explored as the other approaches and it is clear that Bayesian statistics offers a wide range of models that are attractive for cyber security research.

In this work we have identified three main cyber security areas, volume traffic anomalies, network anomalies and malware detection and classification, where Bayesian models, both parametric and nonparametric, have been successfully used. In all three areas, we are (mainly) interested in modelling and statistically

characterising the patterns of normal behaviour so that large departures can be flagged as anomalies. These anomalies can occur at a traffic level for which a change-point detection framework is ideal. They can also occur at the network level where connections and authentication can be characterised as either a bi-partite graph or through the so-called network flows. Finally, the third area discussed was the detection and classification of malware. From a Bayesian perspective, most of the research has been focused on the first two areas. That is why, we have centred our attention to the task of correctly detecting and classifying malware.

Detecting and classifying malware is a crucial part of cyber security, especially now more than ever, due to the increasing number of unknown threats or slight modifications of known ones. To detect malware, there are two main approaches that have been studied: a static analysis or a dynamic one. In the first one we are interested in binary features extracted from the hexadecimal representation of the binary code, whereas in the second one we are interested in the set of instructions. Clearly both approaches have their advantages and disadvantages, however, both of them are important for a thorough understanding of the malware's characteristics. From a Bayesian perspective we found that the static approach had not been fully considered, and that is why we centred our attention on developing Bayesian models for detecting and classifying malware.

As we have already established, the static approach considers binary features built from the hexadecimal representation of the binary code, that completely characterise each malware. Under this approach, several challenges in terms of the dimensionality of the data arise, more specifically, even for small data sets, the number of binary features can easily reach billions. Therefore, most methodologies require a feature selection process before and/or during the inferential process. Once this process has taken place, the data can be represented as a

labelled binary matrix suitable for a supervised learning model. In practice, decision tree and its boosted versions have been widely used due to its high accuracy. However, discriminative models like these ones, do not provide a probabilistic characterisation of the data, which is quite important for a thorough analysis.

In this thesis we have proposed a novel methodology for supervised learning on binary data. Through an elegant approach based on compound random measure we have settled the foundations for a new class of bayesian nonparametric models. The flexibility exhibited by these mathematical objects has allowed us to provide an effective approach to discrimination and classification tasks. Moreover, a feature selection process has also been introduced as part of the inferential process, and we have extended the methodology to a latent factor model. The proposed *beta-CoRM* approach is certainly quite interesting and we believe it has the potential to be used on a wide range of cyber security problems and even in other areas like text classification.

Of course, there are still some theoretical and practical challenges that need to be fully addressed. In particular, for the *beta-CoRM BMF* which we are keen on keep studying the covariance structure, the impact of the parameters, and other theoretical issues regarding the truncation process. From a computational point of view, there is a need to keep developing efficient and faster inferential methodologies. This is crucial due to the large number of observations and features that can be used. In this direction there are a lot of options that could be considered, such as, variational methods, collapsed or accelerated samplers and embarrassingly parallel solutions. All of these are exciting ideas; however, we need to keep in mind their limitations and their applicability to the proposed models.

At this point, we would also like to remark that the task of detecting and classifying malware from a static perspective offers a wide range of modelling possibilities. First of all, if we consider binary features, there are other Bayesian approaches that could be used, for example, in a nonparametric setting the Indian buffet process has been widely used for unsupervised learning. This is something that we have started considering as we showed in our future lines of work. However, the binary approach is not the only one we could be interested in. The large overlapping seen in these classes might be diminished by considering the number of times a feature appears (or a suitable transformation, like its logarithm). This idea opens a whole new world of modelling possibilities and both theoretical and computational challenges, and it is something that we have also started to consider and that we believe to be a compelling approach as well.

Finally, and as our last remark, we would like to emphasise that to be well-protected and prepared against cyber criminals, there is a need to keep developing new anomaly detection methodologies. Throughout history, we have seen how cyber attacks have been designed to avoid detection, for example, there are already variations of malware able to detect if they are running in a sandbox environment, once they are aware of this they can shut down or start producing what we could consider normal patterns of instructions. Cybercrime is a billionaire industry that in many occasions seems to be ahead of us, that is why, cyber security research is more important than ever.

Bibliography

Adams, N. and Heard, N., editors (2014). *Data Analysis for Network Cyber-Security*. Imperial College Press.

Ahmed, I. and Lhee, K. (2011). Classification of packet contents for malware detection. *Journal in Computer Virology*, 7(4):279 – 295.

Alfaro, E., Gámez, a., and García, N. (2013). adabag: An R Package for Classification with Boosting and Bagging. *Journal of Statistical Software*, 54(2):1 – 35.

Angelino, E., Johnson, M. J., and Adams, R. P. (2016). *Patterns of Scalable Bayesian Inference*. Now Publishers Inc.

Antoniak, C. E. (1974). Mixtures of Dirichlet Processes with Applications to Bayesian Nonparametric Problems. *The Annals of Statistics*, 2(6):1152 – 1174.

Atchadé, Y. F. and Rosenthal, J. S. (2005). On adaptive Markov chain Monte Carlo algorithms. *Bernoulli*, 11(5):815 – 828.

Bennett, J. and Lanning, S. (2007). The Netflix Prize. In *KDD Cup and Workshop in conjunction with KDD*.

Berman, D. S., Buczak, A. L., Chavis, J. S., and Corbett, C. L. (2019). A Survey of Deep Learning Methods for Cyber Security. *Information*, 10(4).

Bernardo, J. M. (2003). Bayesian Statistiscs. In Viertl, R., editor, *Encyclopaedia of Life Support Systems*, Probability and Statistics, Oxford, UK: UNESCO.

Bishop, M., Crawford, R., Bhumiratana, B., Clark, L., and Levitt, K. (2006). Some Problems in Sanitizing Network Data. In *15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'06)*, pages 307 – 312.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational Inference: A Review for Statisticians. *J. Am. Stat. Assoc.*, 112(518):859 – 877.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993 – 1022.

Bolton, A. and Heard, N. (2018). Malware Family Discovery Using Reversible Jump MCMC Sampling of Regimes. *Journal of the American Statistical Association*, 113(524):1490 – 1502.

Broderick, T., Jordan, M. I., and Pitman, J. (2012). Beta Processes, Stick-Breaking and Power Laws. *Bayesian Analysis*, 7(2):439 – 476.

Buczak, A. and Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys Tutorials*, 18(2):1153 – 1176.

Cao, J., Cleveland, W., Lin, D., and Sun, D. (2003). *Internet Traffic Tends Toward Poisson and Independent as the Load Increases*, pages 83 – 109. Springer New York, New York, NY.

Cao, X., Chen, B., Li, H., and Fu, Y. (2016). Packet Header Anomaly Detection Using Bayesian Topic Models. *IACR Cryptology ePrint Archive*, 2016:40.

Catlett, C. (2008). A Scientific Research and Development Approach to Cyber Security. Report, U.S. Department of Energy.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly Detection: A Survey. *ACM Computing Surveys*, 41:1 – 72.

Chen, G., Wang, X., and Li, X. (2015). *Fundamentals of Complex Networks*. Wiley Publishing, 1st edition.

Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785 – 794, New York, NY, USA. Association for Computing Machinery.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., and Li, Y. (2021). *xgboost: Extreme Gradient Boosting*. R package version 1.5.0.2.

Chen, X., Irie, K., Banks, D., Haslinger, R., Thomas, J., and West, M. (2018). Scalable Bayesian Modeling, Monitoring, and Analysis of Dynamic Network Flow Data. *J. Am. Stat. Assoc.*, 113(522):519 – 533.

Chockalingam, S., Pieters, W., Teixeira, A., and van Gelder, P. (2017). Bayesian Network Models in Cyber Security: A Systematic Review. In Helger, L., Mitrokotsa, A., and Matulevičius, R., editors, *Secure IT Systems*, pages 105 – 122. Springer International Publishing.

Clausen, H., Briers, M., and Adams, N. (2018). Bayesian Activity Modelling for Network Flow Data. In *Data Science for Cyber-Security*, pages 55 – 76.

Cox, D. R. (1972). Regression Models and Life-Tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187 – 220.

Cramer, C. and Carin, L. (2011). Bayesian Topic Models for Describing Computer Network Behaviors. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE Press.

Damien, P., Wakefield, J., and Walker, S. (1999). Gibbs sampling for Bayesian non-conjugate and hierarchical models by using auxiliary variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(2):331 – 344.

Dantu, R. and Loper, K.and Kolan, P. (2004). Risk management using behavior based attack graphs. In *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.*, volume 1, pages 445 – 449.

de Finetti, B. (1930). Funzione Caratteristica Di Un Fenomeno Aleatorio. In *Memorie della R. Accademia dei Lincei*, volume 4, pages 86 – 133.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391 – 407.

Dhillon, I. S. (2001). Co-Clustering Documents and Words Using Bipartite Spectral Graph Partitioning. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 269 – 274, New York, NY, USA. Association for Computing Machinery.

Donoho, D. and Jin, J. (2004). Higher criticism for detecting sparse heterogeneous mixtures. *Annals of Statistics*, 32(3):962 – 994.

Doshi, F., Miller, K., Van Gael, J., and Teh, Y. W. (2009). Variational Inference for the Indian Buffet Process. In van Dyk, D. and Welling, M., editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 137 – 144.

Dunlavy, D., Hendrickson, B., and Kolda, T. (2009). Mathematical Challenges in Cybersecurity. Report, Sandia National Laboratories.

Eleazar, E. (2000). Anomaly Detection over Noisy Data using Learned Probability Distributions. In *In Proceedings of the International Conference on Machine Learning*, pages 255 – 262. Morgan Kaufmann.

Escobar, M. D. and West, M. (1995). Bayesian Density Estimation and Inference Using Mixtures. *Journal of the American Statistical Association*, 90(430):577 – 588.

Ferguson, T. S. (1973). A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):209 – 230.

Fisher, R. (1934). *Statistical Methods For Research Workers*. Olyver and Boyd, Edinburgh.

Freund, Y. and Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, pages 148 – 156, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232.

Frigault, M. and Wang, L. (2008). Measuring Network Security Using Bayesian Network-Based Attack Graphs. In *2008 32nd Annual IEEE International Computer Software and Applications Conference*, pages 698 – 703.

Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., and Rubin, D. (2013). *Bayesian Data Analysis, Third Edition*. Taylor & Francis.

Geng, D., Odaka, T., Kuroiwa, J., and Ogura, H. (2011). An N-Gram and STF-IDF model for masquerade detection in a UNIX environment. *Journal in Computer Virology*, 7:133 – 142.

Ghahramani, Z. and Griffiths, T. L. (2006). Infinite latent feature models and the Indian buffet process. In Weiss, Y., Schölkopf, B., and Platt, J. C., editors, *Advances in Neural Information Processing Systems 18*, pages 475 – 482. MIT Press.

Ghahramani, Z., Thomas L. Griffiths, T. L., and Sollich, P. (2007). Bayesian nonparametric latent feature models. In *Bayesian Statistics 8*, pages 1 – 25. Oxford University Press.

Ghosh, S., Mukhopadhyay, P., and Lu, J. (2006). Bayesian analysis of zero-inflated regression models. *Journal of Statistical Planning and Inference*, 136(4):1360 – 1375.

Goldstein, M. (2013). Observables and models: exchangeability and the inductive argument. In Damien, P., Dellaportas, P., Polson, N. G., and Stephens, D. A., editors, *Bayesian Theory and Applications*, pages 3 – 18. Oxford University Press.

Gopalan, P., Charlin, L., and Blei, D. M. (2014). Content-based Recommendations with Poisson Factorization. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3176 – 3184, Cambridge, MA, USA. MIT Press.

Grandini, M., Bagli, E., and Visani, G. (2020). Metrics for multi-class classification: an overview. 2008.05756.

Greenwell, B., Boehmke, B., Cunningham, J., and Developers, G. (2020). *gbm: Generalized Boosted Regression Models*. R package version 2.1.8.

Griffin, J. and Holmes, C. (2010). *Computational issues arising in Bayesian nonparametric hierarchical models*, pages 208 – 222. Cambridge University Press.

Griffin, J. and Leisen, F. (2018). Modelling and Computation Using NCoRM Mixtures for Density Regression. *Bayesian Analysis*, 13(3):897 – 916.

Griffin, J. E. and Leisen, F. (2017). Compound Random Measures and their use in Bayesian nonparametrics. *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 79(2):525 – 545.

Griffiths, T. L. and Ghahramani, Z. (2005). Infinite Latent Feature Models and the Indian Buffet Process. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, pages 475 – 482, Cambridge, MA, USA. MIT Press.

Griffiths, T. L. and Ghahramani, Z. (2011). The Indian Buffet Process: An Introduction and Review. *Journal of Machine Learning Research*, 12(32):1185 – 1224.

Gupta, M., Gao, J., Aggarwal, C., and Han, J. (2014). Outlier Detection for Temporal Data: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 26(9):2250 – 2267.

Hagberg, A., Kent, A., Lemons, N., and Neil, J. (2014). Credential hopping in authentication graphs. In *2014 International Conference on Signal-Image Technology Internet-Based Systems*. IEEE Computer Society.

Hall, E. (2000). *Internet Core Protocols: The Definitive Guide: Help for Network Administrators*. An owner's manual for the internet. O'Reilly Media, Incorporated.

Heard, N., Rubin-Delanchy, P., and Lawson, D. (2014). Filtering Automated Polling Traffic in Computer Network Flow Data. In *2014 IEEE Joint Intelligence and Security Informatics Conference*, pages 268 – 271.

Heard, N. A., Palla, K., and Skoularidou, M. (2016). Topic modelling of authentication events in an enterprise computer network. In *2016 IEEE Conference on Intelligence and Security Informatics*. IEEE Press.

Heard, N. A. and Rubin-Delanchy, P. (2016). Network-wide anomaly detection via the Dirichlet process. In *the Proceedings of the IEEE workshop on Big Data Analytics for Cyber-security Computing*.

Hewitt, E. and Savage, L. (1955). Symmetric measures on Cartesian products. *Transactions of the American Mathematical Society*, 80:470 – 501.

Hjort, N., Holmes, C., Müller, P., and Walker, S., editors (2010). *Bayesian Nonparametrics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.

Hjort, N. L. (1990). Nonparametric Bayes Estimators Based on Beta Processes in Models for Life History Data. *Annals of Statistics*, 18(3):1259 – 1294.

Hofmann, T. (2001). Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 42(1–2):177 – 196.

Hosmer, D., Lemeshow, S., and Sturdivant, R. (2013). *Applied Logistic Regression*. Wiley Series in Probability and Statistics. Wiley.

Hu, W., Liao, Y., and Vemuri, R. (2003). Robust Support Vector Machines for Anomaly Detection in Computer Security. pages 168 – 174.

Ishwaran, H. and James, L. F. (2001). Gibbs Sampling Methods for Stick-Breaking Priors. *Journal of the American Statistical Association*, 96(453):161 – 173.

Ishwaran, H. and Rao, J. S. (2005). Spike and slab variable selection: Frequentist and Bayesian strategies. *Annals of Statistics*, 33(2):730 – 773.

Jing, X. and Shelton, C. R. (2010). Intrusion Detection Using Continuous Time Bayesian Networks. *J. Artif. Intell. Res.*, 39(1):745 – 774.

Kao, Y., Reich, B., Storlie, C., and Anderson, B. (2015). Malware Detection Using Nonparametric Bayesian Clustering and Classification Techniques. *Technometrics*, 57(4):535 – 546.

Karagiannis, T., Molle, M., Faloutsos, M., and A. Broido, A. (2004). A nonstationary Poisson view of Internet traffic. In *IEEE International Conference on Computer Communications 2004*, volume 3, pages 1558 – 1569.

Kent, A. D. (2015a). Comprehensive, Multi-Source Cyber-Security Events. Los Alamos National Laboratory.

Kent, A. D. (2015b). Cybersecurity Data Sources for Dynamic Network Research. In *Dynamic Networks in Cybersecurity*. Imperial College Press.

Kingman, J. F. C. (1967). Completely Random Measure. *Pacific Journal of Mathematics*, 21(1):59 – 78.

Knoblauch, J., Jewson, J. E., and Damoulas, T. (2018). Doubly Robust Bayesian Inference for Non-Stationary Streaming Data with β-Divergences. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31, pages 64 – 75. Curran Associates, Inc.

Kolter, J. Z. and Maloof, M. A. (2004). Learning to detect malicious executables in the wild. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 470 – 478, New York, NY, USA. Association for Computing Machinery.

Kolter, J. Z. and Maloof, M. A. (2006). Learning to Detect and Classify Malicious Executables in the Wild. *Journal of Machine Learning Research*, 7:2721 – 2744.

Kruegel, C., Mutz, D., Robertson, W., and Valeur, F. (2003). Bayesian event classification for intrusion detection. In *19th Annual Computer Security Applications Conference, 2003. Proceedings.*, pages 14 – 23.

Lambert, D. (1992). Zero-inflated Poisson regression, with an application to defects in manufacturing. *Technometrics*, 34:1 – 14.

Lancaster, H. O. (1952). Statistical control of counting experiments. *Biometrika*, 39(3 - 4):419 – 422.

Lau, J. W. and Green, P. J. (2007). Bayesian Model-Based Clustering Procedures. *J. Comput. Graph. Stat.*, 16(3):526 – 558.

Masud, M., Khan, L., and Thuraisingham, B. (2007). A Hybrid Model to Detect Malicious Executables. In *2007 IEEE International Conference on Communications*, pages 1443 – 1448.

McGraw, G. and Morrisett, G. (2000). Attacking Malicious Code: A Report to the Infosec Research Council. *IEEE Software*, 17(5):33 – 41.

Metelli, S. and Heard, N. (2016). Model-based clustering and new edge modelling in large computer networks. In *2016 IEEE Conference on Intelligence and Security Informatics*. IEEE Press.

Metelli, S. and Heard, N. (2019). On Bayesian new edge prediction and anomaly detection in computer networks. *The Annals of Applied Statistics*, 13(4):2586 – 2610.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., and Leisch, F. (2021). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.7-9.

Meza, J., Campbell, S., and Bailey, D. (2009). Mathematical and Statistical Opportunities in Cyber Security. arXiv:0904.1616.

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill International Editions. McGraw-Hill.

Mitchell, T. J. and Beauchamp, J. J. (1988). Bayesian Variable Selection in Linear Regression. *Journal of the American Statistical Association*, 83(404):1023 – 1032.

Muñoz González, L., Sgandurra, D., Paudice, A., and Lupu, E. C. (2017). Efficient Attack Graph Analysis through Approximate Inference. *ACM Trans. Priv. Secur.*, 20(3).

Myhre, R. N. (2001). *Introduction to Networking and the OSI Model*. Prentice Hall.

Newman, M. (2010). *Networks: An Introduction*. Oxford University Press, Inc.

Olding, B. and Wolfe, P. (2014). *Inference for Graphs and Networks: Adapting Classical Tools to Modern Data*, pages 1 – 31. Imperial College Press, London.

Olson, D. and Delen, D. (2008). *Advanced Data Mining Techniques*. Springer Berlin Heidelberg.

Oppliger, R. (2001). *Internet and Intranet Security*. Artech House, Inc., USA, 2nd edition.

Paffenroth, R., Kay, K., and Servi, L. (2018). Robust PCA for Anomaly Detection in Cyber Networks. 1801.01571.

Paisley, J., Blei, D., and Jordan, M. (2012). Stick-Breaking Beta Processes and the Poisson Process. In Lawrence, N. D. and Girolami, M., editors, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 850 – 858.

Paisley, J., Carin, L., and Blei, D. (2011). Variational Inference for Stick-Breaking Beta Process Priors. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 889 – 896.

Paisley, J. and Jordan, M. I. (2016). A constructive definition of the beta process. 1604.00685.

Pauwels, S. and Calders, T. (2018). Extending Dynamic Bayesian Networks for Anomaly Detection in Complex Logs. 1805.07107.

Pearl, J. (1985). Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning. In *Proc. of Cognitive Science Society (CSS-7)*.

Pearson, K. (1933). On a Method of Determining Whether a Sample of Size n Supposed to Have Been Drawn from a Parent Population Having a Known Probability Integral has Probably Been Drawn at Random. *Biometrika*, 25(3 - 4):379 – 410.

Pektaş, A., Eris, M., and Acarman, T. (2011). Proposal of n-gram based algorithm for malware classification. *SECURWARE 2011 - 5th International Conference on Emerging Security Information, Systems and Technologies*, pages 14 – 18.

Perman, M., Pitman, J., and Yor, M. (1992). Size-biased sampling of Poisson point processes and excursions. *Probability Theory and Related Fields*, 92:21 – 39.

Pollak, M. and Tartakovsky, A. (2009). Optimality Properties of the Shiryaev-Roberts Procedure. *Statistica Sinica*, 19(4):1729 – 1739.

Polson, N. G., Scott, J. G., and Windle, J. (2013). Bayesian Inference for Logistic Models Using Pólya–Gamma Latent Variables. *Journal of the American Statistical Association*, 108(504):1339 – 1349.

Polunchenko, A. S., Tartakovsky, A., and Mukhopadhyay, N. (2012). Nearly Optimal Change-Point Detection with an Application to Cybersecurity. *Sequential Analysis*, 31:409 – 435.

Polunchenko, A. S. and Tartakovsky, A. G. (2011). State-of-the-art in sequential change-point detection. *Methodology and Computing in Applied Probability*, 14(3):649 – 684.

Poolsappasit, N., Dewri, R., and Ray, I. (2012). Dynamic Security Risk Management Using Bayesian Attack Graphs. *IEEE T. Depend. Secure*, 9(1):61 – 74.

Prasse, P., Machlica, L., Pevný, T., Havelka, J., and Scheffer, T. (2017). Malware Detection by Analysing Network Traffic with Neural Networks. In *2017 IEEE Security and Privacy Workshops (SPW)*, pages 205 – 210.

Price-Williams, M., Heard, N., and Rubin-Delanchy, P. (2019). Detecting weak dependence in computer network traffic patterns by using higher criticism. *Journal of the Royal Statistical Society: Series C*, 68(3):641 – 655.

Price-Williams, M., Heard, N., and Turcotte, M. (2017). Detecting Periodic Subsequences in Cyber Security Data. In *2017 European Intelligence and Security Informatics Conference*, pages 84 – 90.

Price-Williams, M., Turcotte, M., and Heard, N. (2018). Time of Day Anomaly Detection. In *2018 European Intelligence and Security Informatics Conference*, pages 1 – 6.

Pruteanu-Malinici, I., Ren, L., Paisley, J., Wang, E., and Carin, L. (2010). Hierarchical Bayesian Modeling of Topics in Time-Stamped Documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):996 – 1011.

Raff, E., Zak, R., Cox, R., Sylvester, J., Yacci, P., Ward, R., Tracy, A., Mclean, M., and Nicholas, C. (2016). An investigation of byte n-gram features for malware classification. *Journal of Computer Virology and Hacking Techniques*, pages 1 – 20.

Ridout, M., Demetrio, C., and Hinde, J. (1998). Models for counts data with many zeros. In *Proceedings of the XIXth International Biometric Conference*, pages 179 – 192.

Ripley, B. (2021). *tree: Classification and Regression Trees*. R package version 1.0-41.

Roberts, S. (1966). A comparison of some control chart procedures. *Technometrics*, 3:411 – 430.

Rodrigues, J. (2003). Bayesian Analysis of Zero-Inflated Distributions. *Communications in Statistics - Theory and Methods*, 32(2):281 – 289.

Ronen, R., Radu, M., Feuerstein, C., Yom-Tov, E., and Ahmadi, M. (2018). Microsoft Malware Classification Challenge. arXiv:1802.10135.

Rubin-Delanchy, P., Heard, N. A., and Lawson, D. J. (2019). Meta-Analysis of Mid-p-Values: Some New Results based on the Convex Order. *Journal of the American Statistical Association*, 114(527):1105 – 1112.

Rumao, P. (2016). Detect Malacious Executable (AntiVirus). Available at: https://archive.ics.uci.edu/ml/datasets/Detect+Malacious+Executable(AntiVirus). UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science.

Salakhutdinov, R. and Mnih, A. (2007). Probabilistic matrix factorization. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, pages 1257 – 1264, USA. Curran Associates Inc.

Sanna Passino, F. and Heard, N. A. (2019). Modelling dynamic network evolution as a Pitman-Yor process. *Foundations of Data Science*, 1:293 – 306.

Sato, K. (2013). *Lévy Processes and Infinitely Divisible Distributions*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2nd edition.

Schultz, M. G., Eskin, E., Zadok, F., and Stolfo, S. J. (2001). Data mining methods for detection of new malicious executables. In *Proceedings 2001 IEEE Symposium on Security and Privacy. S P 2001*, pages 38 – 49.

Shiryaev, A. N. (1963). On optimum methods in quickest detection problems. *Theory of Probability and Its Applications*, 1:22 – 46.

Storlie, C., Anderson, B., Vander Wiel, S., Quist, D., Hash, C., and Brown, N. (2014). Stochastic identification of malware with dynamic traces. *The Annals of Applied Statistics*, 8(1):1 – 18.

Stouffer, S. (1949). *The American soldier*. Studies in social psychology in World War II. Princeton University Press.

Tang, Y., Wu, Y., and Zhou, Q. (2010). AASC: Anonymizing network addresses based on subnet clustering. In *2010 IEEE International Conference on Wireless Communications, Networking and Information Security*, pages 672 – 676.

Tartakovsky, A. G. (2014). *Rapid Detection of Attacks in Computer Networks by Quickest Changepoint Detection Methods*, pages 33 – 70. Imperial College Press, London.

Tartakovsky, A. G., Rozovskii, B. L., Blaźek, R. B., and Kim, H. (2006a). Detection of intrusions in information systems by sequential change-point-methods. *Statistical Methodology*, 3(3):252 – 293.

Tartakovsky, A. G., Rozovskii, B. L., Blaźek, R. B., and Kim, H. (2006b). A novel approach to detection of instructions in computer networks via adap-

tive sequential and batch-sequential change-point detection methods. *IEEE Transactions on Signal Processing.*, 54(9):3372 – 3382.

Teh, Y. W., Grür, D., and Ghahramani, Z. (2007). Stick-breaking Construction for the Indian Buffet Process. In Meila, M. and Shen, X., editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 556 – 563.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101(476):1566 – 1581.

Thibaux, R. and Jordan, M. I. (2007). Hierarchical Beta Processes and the Indian Buffet Process. In Meila, M. and Shen, X., editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 564 – 571. PMLR.

Tippett, L. (1931). *The Methods of Statistics*. Williams and Norgate, London.

Turcotte, M., Moore, J., Heard, N., and McPhall, A. (2016a). Poisson factorization for peer-based anomaly detection. In *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*. IEEE Press.

Turcotte, M. J. M., Heard, N. A., and Kent, A. D. (2016b). *Modelling user behaviour in a network using computer event logs*, pages 67 – 87. World Scientific.

Turcotte, M. J. M., Kent, A. D., and Hash, C. (2018). *Unified Host and Network Data Set*, chapter 1, pages 1 – 22. World Scientific.

Valdes, A. and Skinner, K. (2000). Adaptive, Model-Based Monitoring for Cyber Attack Detection. In *Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection*, RAID '00, pages 80 – 92. Springer-Verlag.

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, New York, fourth edition. ISBN 0-387-95457-0.

Vidal, M., Orozco, J., Orozco, S., Lucila, A., Villalba, G., and Javier, L. (2017). Alert Correlation Framework for Malware Detection by Anomaly-Based Packet Payload Analysis. *Journal of Network and Computer Applications*, 97(C):11 – 22.

Willinger, W. and Paxson, V. (1998). Where mathematics meets the Internet. *Notices of the American Mathematical Society*, pages 961 – 970.

Yang, Y. and Liu, X. (1999). A Re-Examination of Text Categorization Methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 42 – 49, New York, NY, USA. Association for Computing Machinery.

Yang, Y. and Pedersen, J. O. (1997). A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412 – 420. Morgan Kaufmann Publishers Inc.