Expressive Prediction and Forecasting of Alarms within a National Telecommunications Network

Chris Wrench

2018

December

Abstract

This thesis concerns the prediction of faults in a telecommunication network, this is done in an expressive way that offers engineers insight into the cause of a fault to help guide mitigating action. The thesis is a detailed account of the preprocessing steps applied to the data and an exploration of Rule Induction techniques resulting in a novel method of Rule induction. A method is developed to enable classification algorithms to forecast events. The result of this work is a system that can forecast critical events with high precision.

Telecommunications are a vital part of modern society. They are relied upon for the running of both businesses and personal lives and so minimising the disruption to this network is very important. To this end there is a system of alarms in place that detail faults and warnings that engineers can respond to in a timely manner. A method of forecasting these alarms would allow engineers more time to take action. If these forecasts took the form of feature rich expressive rules then engineers would be offered a greater insight into the cause of an issue and may be able to mitigate the oncoming fault in the future.

To create more expressive rules a method is introduced to hybridise event forecasting and event classification, allowing a classifier to produce forecasting rules. This offers a number of benefits over traditional forecasting techniques. This work also contributes an accompanying novel abstaining classifier adapted to producing rules with a broad range of features.

This a very desirable feature when dealing with alarm data as other forecasting techniques are not able to capitalise on the potentially valuable data held within each alarm outside of the overarching alarm type. Rules that can incorporate these details will offer an engineer more information to work with.

Dedication

Is life worth so many questions? - Aramis

My thanks to Fred for his patience through an unexpected number of years, I was very relieved not to break your run of successful defences. To my sponsor, BT, and in particular Detlef and Vidya, who made me welcome there and made sense of what I was doing. To my lab mates for setting the bar so high and to all my parents and siblings both here and abroad. Finally to Krissy without whom, I feel certain, I would not have made the last set of hurdles.

Declaration

Declaration: I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Chris Wrench

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to the Problem of Forecasting Network Alarms

Data is generated at an increasing rate as smart devices are further incorporated into society and our daily lives. Data often has a great deal of value that can be extracted given the right tools. It can be used to predict human buying habits, raise alerts when safety parameters are breached, aid traffic flow around cities or networks and more. For this reason data is collected and stored for eventual processing and benefit. This is becoming the status-quo for all data, whether the data's value is known at the time or not.

Data can be generated sporadically, periodically or constantly, it can be a complex record of multiple features or a single stream of digits, it can be machine generated or manually recorded. The great variety makes the job of the Data Miner to extract worth from this data a difficult and often bespoke process.

This thesis is concerned with predicting future alarms using a data set of historic events from the UK national telecommunication network. An alarm is a message or an event sent across the network to alert an engineer of a fault. These events are generated from devices within the network to alert network managers to a potential fault or anomaly in the network device. There are a wide variety of alerts raised by the network, each indicating a different

level of severity.

Some alarms represent a disruption in service, a clearly undesirable circumstance for customers who increasingly depend on their connectivity to the World Wide Web for their business and leisure. These alarms are considered to be more severe and avoiding these is a priority.

Customer satisfaction is an important metric for any business and so errors must be kept as low as can be and dealt with quickly. Predicting the most severe events ahead of time would aid in both of these aspects. Given sufficient warning there could be an automated or manual intervention to avoid an outage. Failing this a prediction would give an engineer a head start in remedying the issue.

Further to this, it would lighten the workload of both network monitors and network engineers. The potential saving for the service provider, through a combination of saving time for engineers, call-centres and reducing churn rates are in the order of millions[84].

The network alarms are just one data set of the many that make up the national telecommunication network, other data sets such as the underlying log data and recorded metric data may be required to augment the produced system further. This work is primarily exploratory in nature, an attempt to answer the question of whether this data has sufficient predictive value, discovered through the application of a great many techniques.

A telecommunication network is a collection of interconnected switches, routers, gateways and servers that operate a number of protocols to pass data from an originating address to it's destination. The structure of the network is often vast and complex with components in need of upgrading or replacing. It is often the case in older networks that the inventory of devices and connections is not complete. To aid in running the network, smart devices monitor themselves and generate events to be passed to the operations centre. Normally the action taken on these events is left to human experts who are able to see the event in its wider context and make an informed decision on how to respond. The granularity of the network ranges from an overarching national level through to regions, cities, Virtual Local Area Networks (SVLAN), down to the individual components that make up a devices.

Neighbouring devices are expected to have an influence on each other, for example a router failure will lead to a shift in traffic pathways and the load on some devices will increase.

A number of techniques have been used during this research, taken largely from the field of machine learning. Initially the project focused solely on Rule Induction approaches. These are set apart in machine learning from *black box* approaches such as Artificial Neural Networks (ANN) or K-Means Clustering. There are a large variety of approaches that share the common goal of producing a model based on human readable rules. Each rule comes with one or more conditions (formally known as the antecedent, head, or Left Hand Side (LHS)) that, when met, trigger a consequent (tail or Right Hand Side). These give a clear advantage over statistical methods, in terms of this project, in that each fired rule is accompanied by a rationale as to why it fired. This is a large benefit for any engineer or operator looking to understand how or why an alarm was raised. In turn this can help with any attempt at mitigating the cause of the alarm in the future. A variety of Rule Induction algorithms were examined in this work, a description of them is provided in Chapter 2.

1.2 Research Aims and Objectives

The overall aim of this project is to determine if it is possible to produce human readable rules that forecast network alarms ahead of time in a telecommunication network. The objectives can more precisely be defined as:

1. Is it possible to forecast critical network alarms.
    - This work is an exploratory work to see if the data supports predictions, answering this question with confidence is important and further work can be built upon it.
2. Assuming it's possible can we do this via a Rule Induction or similar *white box* approach to offer an explanation behind the raising of an alarm.
    - The system produced should generate a rule based model that can be used by network operators and engineers to diagnose and correct a fault, as well as mitigate

any further faults.

3. Ideally produced rules should contain an expected time before the fault occurs and potentially provide a location.

    - Predicting faults ahead of time would allow the engineers greater time to intervene and pre-vent a loss of connectivity to the customer.
    - Producing a location would also be important information for engineers, ideally to a fine granularity.

The ideal system should produce a rule similar to:

IF (DeviceA and SymptomA) and (DeviceB and SymptomB) THEN DeviceC and Critical SymptomC [Time T]

The rule above points an engineer to a network device C that is likely to fail within an estimated time T. Beyond that it also provides the engineer with the rational behind the prediction, giving the engineer more insight into the problem and helping them take the correct action to pre-event the fault or mitigate it's impact on the network.

1.3 Methodology

This thesis will examine Rule Induction approaches to forecasting alarms. Rule Induction algorithms are a collection of methods that classify an instance by matching a subset of it's features to a rule, an operation that is fully comprehensible to a human user who could use these rules to classify an instance themselves. This property has led to their inclusion in the *white box* family of algorithms. Algorithms within Rule Induction differ in approaches, the advantages and disadvantages of them will be explored in Chapter 2. They are important in this study as a desired outcome of this project is to produce a model that engineers can understand and help create further understanding of a vastly complicated network.

Before modelling can take place the data must be examined, it is common to find the raw data does not lend itself to a modelling problem without first undergoing a transformation

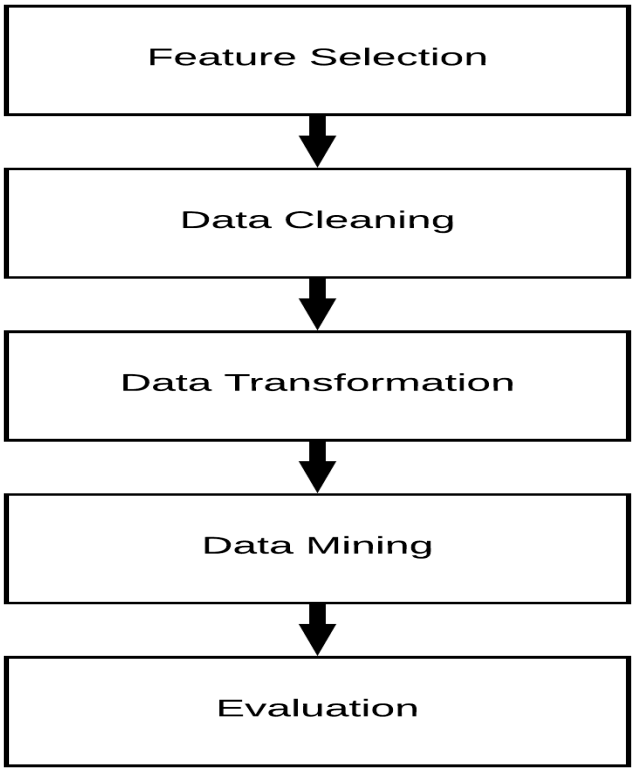

Figure 1.1: The five steps of Knowledge Discovery Process

to help expose key concepts. This is referred to as Data Cleaning and Feature Selection, two early steps on what is referred to as the Knowledge Discovery and Data Mining (KDD) work-flow, depicted in Figure 1.1.

The primary data set consists of just under 1.7 million rows from 2 months of operation ordered by the time the alarm was generated. Each row is an event generated by a network device that indicators some abnormal behaviour. Devices vary in granularity from the high level of a server rack to a component on a switch or router, i.e. a port number or network card. Each row has 37 features. Of these there are several features that would be key to prediction though others may contain a less explicit value. The readily identifiable key features are:

Opened Date, **Element Class Name**, **Event Name**, **Device ID**, **Location**

Event Name, (Element) Class Name and Device ID are key in describing the nature of the event. Opened Date and Location provide contextual information that can be used to determine how these events possibly affect each other.

As mentioned, the data has a high number of dimensions and contains a mix of manual and automated entries. This means that the data may be subject to attribute noise, missing or erroneous values that make may create bias in any model produced if not mitigated[62]. There may be features that have no impact on the target class that they may be removed. The system follows the steps above to clean and prepare the data beginning with feature selection. Feature selection is employed using a mix of statistical techniques along with input from domain experts.

Feature Selection is intended to decrease the dimensionality of the data by removing features which make the lowest contribution to the prediction process. There are an number of reasons why it may be desirable to remove a feature:

- Features with low to zero variance, that is features that do not change, add no value to the data set
- Features that are highly correlated create redundancy in the data. Redundant features can be removed or merged to create a new feature
- Features that have little to no correlation with the target class as these are likely to inject noise into the data set

Data cleaning is the process of examining the data for inconsistency or sparsity. It can involve addressing missing values in the data or checking features for extreme values that might constitute an error. Manually updated features are vulnerable to human error and are likely to contain missing or inconsistent values which will need to be checked for. Certain numerical features will also need validating to see if they reflect real world values.

Data Transformation may involve scaling numerical features; placing values into bins to reduce the number of distinct values; changing how the values are represented, for example some models are dependent on ordinal data values, in which case a decision has to made as to how to transform categorical features.

These steps are designed to produce a clean data set better suited to processing by a data mining algorithm. As a rule of thumb approximately $\frac{2}{3}$ of the work is dedicated to

data preparation. Once this is completed the algorithm can be applied. Following this it is important to evaluate the outcome. This step is highly dependent on the type of data mining undertaken. Traditional methods include various forms of hold-out testing.

The final step of the KDD process is the evaluation of the model. This is defined by what the problem domain and what the projects goal is. An appropriate metric, or set of metrics, must be selected that are applicable to the model and provide the best measure of how well the system meets the criteria. The system this thesis is concerned with will be designed to alert engineers to a problem so they may take steps to mitigate it. With this in mind the focus will be on the precision of the system. Precision is the ratio of the positive values correctly classified against the total number of classifications. It can be expressed as:

$$\frac{TP}{TP + FP} = \frac{TP}{P} \tag{1.1}$$

Where TP and FP are the number of true positives and False Positives respectively. By maximising the precision of the system false alarms are minimised. Any action taken by an engineer as a result of the system will have an associated cost, minimising the false positives will keep this cost low and will also help to instil confidence in the system. Operator compliance (the uptake of a system and subsequent use of the output) is more heavily impacted by a false positive than by a missed alarm[36].

Further to the quantitative evaluation, the system will also be evaluated qualitatively. To help engineers understand the network the system should produce rules of *interest*, which is highly subjective. Whilst all rules should be logical and intuitive, a subset of the rules should also be interesting and provide new insight into a cause of an alarm. Whilst metrics exist that can determine the information content of a rule, and these shall be looked at, it is not possible to fully quantity something so domain specific. The value of a rule at testing stage can be best determined by a domain expert and so part of the evaluation will be by such an expert from BT. This thesis has employed all the above steps in its attempt to meet the objectives, they will be more thoroughly examined in Chapter 3, Chapter 4, and Chapter 5.

1.4 Contributions

The overall contribution of this work is the development of a combination of methods to produce expressive rules to predict future events accounting for a minority class without artificially altering the class bias. This can be further decomposed into the following:

1. An empirical evaluation and analysis of off-the-shelf expressive algorithms to determine their effectiveness at making predictions from the data. A number of approaches were tried to address the question of whether forecasting events in this data set is possible.

2. A novel way of pre-processing event stream data to allow the application of descriptive data mining algorithms to a predictive problem. This is a done to meet the requirement of producing expressive rules along with a prediction and it's outline.

    - This method allows the application of descriptive Data Mining algorithms that work across a range of features to a forecasting problem. This kind of problem is normally addressed with an item based predictor.
    - This method is important in fault diagnosis as they present the engineers with information gleaned from a more varied range of features.

3. The development of a two stage classifier for forecasting infrequent network alarms. In conjunction with the above data transformation this system allows both predictions of events and expressive rules.

    - This system is designed to address the class imbalance of the data without the need to skew the sampling or create artificial instances of the minority class. This avoids the need for additional interference in the data and is expected to be a more truthful representation of the problem.
    - The system allows the combination of a statistical method with powerful predictive properties with a Rule Induction approach to produce the human readable rules required by both the users and objectives.

A number of publications have been produced in the course of this project. These, in order of publication are listed below:

- **A Review of Real Time Complex Event Processing**[86] - Chris Wrench, Frederic Stahl,Guiseppe Di Fata, Detlef Nauck, Vidhyalakshmi Karthikeyan in Enterprise Big Data Engineering, Analytics, and Management published by IGI. An overview of Data Mining applications in both Complex Event Processing, and Event Stream Processing.

  The book chapter examines the thinly defined differences in the two fields and looks at the way these streams are handled in Data Stream Mining for various industries including Telecommunications.

- **Towards Expressive Rule Induction on IP Network Event Streams**[85] Chris Wrench, Frederic Stahl, Giuseppe Di Fatta, Vidhyalakshmi Karthikeyan, Detlef Nauck in Research and Development in Intelligent Systems XXXII by Springer, presented at SGAI AI 2015 in Cambridge.

  A positioning paper for a new Data Stream Mining system designed to produce expressive rules from the data set and handle concept drift.

- **A Method of Rule Induction for Predicting and Describing Future Alarms in a Telecommunication Network**[87] - Chris Wrench, Frederic Stahl, Giuseppe Di Fatta, Vidhyalakshmi Karthikeyan, Detlef Nauck in Research and Development in Intelligent Systems XXXIII by Springer, presented at SGAI AI 2016 in Cambridge. An empirical study of a batch based predictive Rule Induction system incorporating pre-events and a JMeasure based classifier.

  This system performed well in prediction overall but suffered from a number of false positives. False Positives, ie. forecasting a serious alarm that never comes, is a potentially costly problem if manpower is assigned to fix the non-problem. This is taken into account for subsequent works.

1.5 Structure of this Thesis

This thesis is arranged in the following order:

Chapter 2 contains background information on Telecommunication Networks, other approaches taken to solve similar problems and a more in depth description of the pre-existing algorithms employed during this work.

In Chapter 3 a more thorough description of the data is provided, largely through statistical overviews. Time is taken to describe what in the data represents a target alarm and the context surrounding it.

Chapter 4 contains a record of a number of experiments conducted during an exploratory stage of the research as the problem is narrowed down.

Chapter 5 presents the system arrived at along with a thorough empirical evaluation and answers the questions posed in the original objectives.

Chapter 6 wraps up the thesis with some concluding remarks and outlines some open research topics.

1.6 Summary

This chapter has introduced the problem at hand- the need for an expressive method of forecasting alarms in a telecommunications network. It has described the network in brief along with the data that forms the focus of this study and some of the techniques to be explored.

The objectives of this project have been formally set out along with the intended contributions of this project. Finally, an overview of the road map to meet these objectives has been given in the form of the thesis structure.

Chapter 2

Literature Analysis

A set of objectives were established in the previous section, in brief these were:

1. Establishing if it possible to forecast from these particular sets of alarms
2. Create a forecasting system that generates an accompanying rationale behind the forecast
3. Determine how far in advance an alarm can be forecasted as well as providing an approximate location

The initial phase of investigation has two aspects to it: to understand the data and to survey the surrounding literature to establish approaches have been explored previously. These two parts are both important and have a cyclic influence on each other. In this chapter we will focus on the literature surrounding the forecasting of events and other works focussing on telecommunication. This begins with a look at telecommunication networks to establish the problem domain, this is followed by a look at event based data mining. The literature around data mining an telecommunication is examined, this is broken down into two categories, those approaches that forecast alarms (largely frequent pattern based), and those that describe alarms (including classification and clustering). There is a more general look at rule based approaches, a description of some event processing techniques and finally the most promising avenues of research are discussed in the concluding remarks. The next

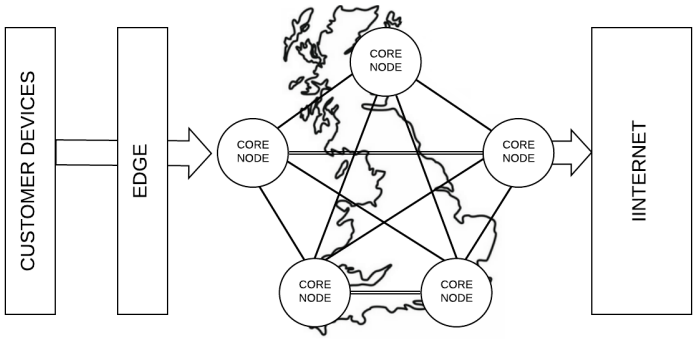

Figure 2.1: A simplified diagram of the national Core Network

section is a brief introduction to the telecommunication network the primary data set is collected from.

2.1 Telecommunication Networks

The National Telecommunication Network consists of many layers, this work is concerned with the core network and connected metro nodes. The edge layers that provide the connections to the cabinets and customers, as well the connections to the outside internet are not directly concerned, see Figure 2.1. The core network is a collection of IP devices that are responsible for routing traffic from the exchange up until the network edge where it leaves BT's jurisdiction. Core nodes are placed across the country for maximum connectivity and connected in full mesh for resilience. To limit the amount of routing needed to deliver these packets across the country devices are grouped into SVLANs designated for carrying a subset of packets labelled with the corresponding SVLAN tag. If devices fail then network protocols such as Rapid Tree Spanning (RTS) are in place to reroute the traffic, avoiding outages but placing greater strain on other parts of the network. Most outages that affect customers occur outside the core network between the cabinet and the home, referred to as the 'last mile'[31].

There has been a national telecommunication network in some form for nearly one hundred years and the dependency upon it has steadily grown. A publication from the UK Government[12] outlines the criticality of the network and potential threats it faces. These

include physical damage to the network by nature, accident of malicious intent; cyber attacks such as a man in the middle or Denial of Service attack, overloads and hardware failures. This list is not exhaustive and is provided for scope, this work will only go towards addressing some of them.

As a business there are additional concerns for the network operator, BT. A level of service must be maintained for customer satisfaction. Falling below this service level is likely to result in an increased customer churn rate and action from the regulatory body the Office of Communications, or Ofcom[63].

2.2 Event Based Data Mining

Data Mining by necessity covers a broad range of approaches to match the large variety of data sets. Understanding the data set and the end goal are important factors in effective data mining. As mentioned in Chapter 1, the target data set is made up of events generated from a national network of IP devices, because of this there are some important features that affect the final approach. To begin with the nature of an event must be looked at, a more in depth look at Events and Event processing is available in [86].

Events have a number of properties that distinguish them from other types of data sets[65, 35]. Events can be multivariate or singletons. The primary data set has a very high dimensionality but there is value in representing them as a series of categorical attributes. This is because events have an explicit temporal component and ordering. Meaning can be derived from this ordering by studying the events which precede or follow the event. At the same time events are highly compartmentalised and self-contained, unlike a timeseries, there is not an assumed autocorrelation. Events often conform to a Poisson distribution, their occurrences can seem random or they may appear in bursts of great activity[43]. It can be very sparse or empty for long periods of time or it can be noisy with an average interval measure in nanoseconds. These aspects set event mining apart from other kinds of Data Mining, leading to a new field of Event Processing and Complex Event Processing, alternatively a series of events can be restructured and approached as a different problem.

In the following, events have been treated in a number of different fashions. This is an account of predictive analytics as applied to, primarily, the domain of telecommunication networks.

2.3 Predictive Analytics as Applied to Telecommunication Networks

Fault prediction in networks is a problem that has been studied in the past with a range of methods, some of these studies encompassing or specialising in telecommunication networks.

Telecommunications were amongst the first to use Data Mining and Data Stream Mining. This work is concerned with fault prediction within telecommunication but there are other areas of interest within this domain. [82] defined two additional applications as Marketing (detecting likely adopters of new services or value customers) and Fraud Detection (identifying unscrupulous accounts to prevent losses to both the company and victimised customers).

A collection of works focussing on fault prediction follows, a more general review of predictive analytics in telecoms is available from[82]. A number of approaches for other IP based networks are included for their potential compatibility. The next section is divided into two headings, those works that attempt to define and describe the relationships between alarms (Descriptive) and those that attempt to forecast alarms (Forecasting). Both fields of research are relevant to this thesis.

2.3.1 Alarm Description

Alarm description techniques are derived from classification and clustering problems from the broader data mining field. The goal is to generate new information to describe each data point usually in the form of labelling through either the assignment of a target class or the placement into a cluster of similar points. The authors of [27] applied ITRULE to generate rules for an expert system in order to automate network management and the networks

response to alarms. ITRULE induces rules using the J-measure and uses beam search to limit the search space. [50] present a very general method of detecting a root cause using topology data by finding the most likely failed component between pairs of links. The output is a hypothesised failed link in what is termed a *silent failure* or *black hole*, a similar problem is described in [42] and combated using a modified Bayesian Network.

[41] presents eXpose which learns dependency rules using the J-measure to help diagnose faults. The J-measure, also known as cross entropy, is a measure of the theoretical information content of an *IF THEN* rule. exPose uses the information within packet traces as its basis for rule learning, it gathers these within time windows of 1 second, network packets being far busier than the telecommunication data set which consists of many sparse alarm channels. Relations between packets sent through the network are assumed to be unidirectional (as communications are nearly always two way) and so the unweighted J-measure is used as it does not infer a direction. The J-measure is further modified to remove the negative component so as to only score positive relations between events (ie. dismissing the relation between event A happening and event B **not** happened). The rule set is also pruned so that only rules whose LHS and RHS occur with similar frequencies are kept. These rules are then marked as significant if their support is over a threshold. exPose uses many interesting ideas and though the granularity of the problem is very low (windows of seconds) this system could be adapted for alarm prediction.

TASA[32] produces episodic rules from alarm data with a goal to provide new insight into alarm relations, the method of producing these rules is similar to Apriori and Association Rules Mining (ARM). The LHS of these rules are *alarm predicates* defined as any 'expression that can be evaluated from a single occurrence of an alarm', not simply alarm types but any predetermined combination of an alarms features selected by a domain expert. These predicates have either a strong or weak ordering within their set and are given to the system by the user along with the desired window width, experimentally found to be between 5 seconds and 10 minutes. The method depends on finding frequent patterns with sufficient support before clustering rules by their consequents and presenting them for investigation.

Two final approaches that focus on describing relations are mentioned here. [18]'s TP Mining searches for repeated event patterns within a time window and promotes those with a high Topographical Proximity (TP), a metric derived from the relative position of a source device to other devices. As geographical data is provided for a number of these devices this method of rule promotion could be beneficial, the suitability of the approach will depend on presence of patterns with sufficient support in the event data. [45] uses Ant Colony Optimization to produce time based rules, these are rules created by traversing an N by N matrix of all feature values starting with a time to reach a target class, producing and evaluating IF THEN rules with the nodes selected. Node selection is biased towards the arbitrary ordering of the features within the graph and a high dimensionality would make the method very computationally expensive. Further more there is an assumption that the time of an alarms generation is always a defining feature. This assumption is unlikely to hold, however, this method may be adaptable.

2.3.2 Alarm Forecasting

The authors of [40, 61] focus their work around fault prediction in the Pakistan Telecom network. They approach the problem with Decision Trees, an adaptation of Association Rules (termed temporal rules), and Neural Networks. To predict the chosen network events (in this case limited to three types), Apriori is used to identify patterns leading up to the event in a restricted time window. The data is separated according to device type and a large number of alarms deemed non-critical are filtered out before processing. The rules produced are non-descriptive and the restriction to producing rules by device type is a more narrow problem then the one we are presented with. The features space is also much smaller.

In [83] produced a genetic algorithm named Timeweaver that specialises in predicting rare events from a telecommunications alarm data set. It follows a two step pattern very similar to Apriori and details a bespoke language enabling patterns to be produced from ordered events, unordered events and wild-cards. The fitness function used is a combination of it's F1 score (a score derived from precision and recall) along with a metric to promote diversity within

the rule set, a property important in converging to a good solution traditionally managed by breeding and/or mutation.

[47] investigates an enhancement to the algorithm TASA, using sliding windows to find both Association Rules and Episodic Rules. Episodes are frequently occurring sequences of event types that exist within a window that occur in a time interval. It produces human readable rules along with a confidence value. Expert domain knowledge is then required to analyse the great quantity of rules produced by this system before it is applied to a live system. The system has been evaluated by these same domain experts and, amongst the Episodic Rules produced, several unknown patterns have been reported. The system handles only alarms that occur momentarily but with some adaptation could deal with durable events, they are also very short term but have some level of expressiveness including both Episodic and Association Rules.

In [88, 49] a Markov based codebook approach is used, manually identifying problem alarms (alarms of interest) and labelling the succeeding alarms as potential symptoms. Correlation graphs are formed from the ordering of the symptoms, with some aggregation for repeated sub-sequences, these are then vectorised and presented as a codebook. Hamming distance is then used to detect re-occurrences of each code in the network with some resistance to noise. The system is designed to assist operators with root cause analysis over a complex method rather than forecasting though prediction may be possible if the approach were reversed. The authors claim a significant speed increase over rule based systems though no empirical evidence of this is presented. In [20] an alarm prediction method over an IP network is proposed using SVM and Singular Value Decomposition (SVD)[26]. Each window of events is converted into a representative discretised vector and the collection of windows form an event-by-window matrix where each column represents a window. SVD is applied and the first k columns of the resulting v matrix become a new data set to train the SVM with a Radial Basis Kernel (RBF). An online version is proposed using an incremental version of SVD. The optimum window size for the data is investigated through mapping error rates to window sizes from 5 to 100 minutes. The error rate falls and plateaus at the 35 minute mark.

This work does produce accurate predictions for a specific alarm type, additional SVD-SVMs must be included for other target classes. There is no rationale behind the rules produced but there are methods of extracting rules from black box systems that will be explored in 2.5.3.

A number of approaches reoccur during this survey that may be applicable to this survey. They have been identified as:

- Rule Induction techniques that classify an instance using a model consisting of human readable rules. ITRULE and the J-measure have made frequent appearance in the literature as well as Ant Colony optimisation and the bespoke algorithms TASA and TP Mining,

- Alarm Correlation allows for forecasting of alarms ahead of time. These often use an adaptation of Association Rules or a transformation of an event series into a different format during preprocessing.

2.4 Gaps in Research

Though [47] goes some way to fully expressive rules, the examples used are almost immediate (time-scales of roughly 5 seconds). The way the rules are produced depend on compartmentalising the alarm series into several overlapping windows and taking these to be the item sets. From the data analysis it is clear that the events in the focus data set occur burstily, that is that the stream is sparse but windows containing events contain a very high concentration. An Association Mining approach will be explored to produce rules but unless it is adapted for bursty data then it is likely to be affected by a lot of noise. This problem applies to all approaches that apply Frequent Pattern Mining or ARM to the data set.

[47] also creates very large rule sets that analysts have found useful for uncovering patterns but, although they are capable of predicting future events, they are not used so. The method produced here will ideally provide enough warning for human intervention.

There are several mentions of the J-measure as a metric for evaluating or creating the

models used for classifying event date. The J-measure is a metric designed to promote a balance of entropy and likeliness to occur and is one possible way of promoting rules that lead to infrequent events. However, as mentioned, one driving component is how likely a rule is to fire and as such may not be best suited to creating models that predict rare events. In [41] an altered version of the J-measure is used to remove the likelihood component though possibly altering the usefulness of one of the most useful properties of the J-measure, Jmax. This will be expanded upon later in this chapter.

2.5 Rule Based Data Mining

A number of the approaches in the literature from the previous sections have used adaptations of pre-existing Rule Based techniques. In this section a more detailed look at some of these algorithms is provided. These include Frequent Pattern Mining algorithms such as Apriori and Rule Induction techniques such as ITRULE and finally a section on methods of extracting rules from black box systems.

2.5.1 Frequent Pattern Mining and Association Rule Mining

A popular analogy to demonstrate the value in Data Mining relates to Market Basket Analysis. Customers upon reaching the checkout have a collection of items to purchase that form a transaction set. Frequent Pattern Mining[29, 25] is a collection of techniques to locate frequently occurring subsets which, as the second stage of ARM, can be converted to human readable rules.

The most famous of these algorithms is Apriori[2]. Support and confidence are important metrics in Frequent Pattern Mining so they are defined here. Support is the probability of items appearing together out of the transaction set as per equation 2.1:

$$P(X \cap Y) \tag{2.1}$$

and Confidence is the probability of the complete subset occurring out of those transactions

containing an incomplete subset, see equation 2.2:

$$\frac{P(X \cap Y)}{P(Y)} \tag{2.2}$$

First all item sets of size 1 are found and their supports calculated. Those that are above a minimum support are expanded upon and the process repeats until all item sets are maximal. The Apriori principle is used here to limit this otherwise large search space, if a subset support is below the threshold (i.e. it can be said to be infrequent) then it's superset must also be infrequent. ARM goes one step further to divide frequent item sets into a LHS and a RHS of association rules to find a set of rules with a given confidence.

2.5.2 Rule Induction Classification Methods

One of the most effective ways of producing a rationale for a classification is via Rule Induction. For this reason a summary of some of the most popular algorithms is included here. Rule Induction is a branch of Machine Learning dedicated to producing Rule Based models to predict the data. Rules typically take the form of *IF THEN* rule sets, meaning if the condition of the LHS is met than the RHS is likely (often with some accompanying probability or support). The model consists of a variable number of rules that are compared to the data to find the best match. They have their roots in decision trees such as the early AQ[58]. The field is now quite large but there are some recurring traits. Rules begin as a single node and are expanded upon. This expansion is driven by one of many available metrics and continues until a stopping criteria is reached optionally followed by a pruning phase to generalise lengthy rules and prevent over fitting.

Like many models the metric driving the Rule Induction process can be varied, careful selection of this can promote precise rules, infrequent rules or another desired quality. The authors of [24] draw equivalences between a range metrics used in separate and conquer algorithms and group them into two categories. Precision based metrics, those that focus on increasing the local AUC of an ROC curve such as recall, precision and the weighted combinations of both, and cost weighted difference such as GINI. The searching method will

have an impact on the resultant rule set. Trees follow a covering approach, one which will increase the rule set until every combination in the training set is covered by a rule. The antithesis of this is to produce a set of n rules at the cost of only classifying instances that fit these patterns. One final decision to mention here is to incorporate pruning and select a metric to drive this. To measure rule quality there are several statistical metrics relating to how many instances are covered (coverage), missed (consistency) or incorrectly classified (accuracy) by a rule. These are quite powerful methods of measuring a rules use but do not quantify the descriptive power of a generalised rule.

The following details a number of important entries into the rule induction family beginning with Decision Trees. Each algorithm is capable of producing a rationale for classification and all should be considered.

Decision Trees are a covering approach to Rule Induction following what is known as Top Down Induction. Starting with a root node each possible split is evaluated via the driving metric. When the optimal split is found a child node is created with all the instances covered by the rule so far. This process is continued until there is a child node of just one target class, this is labelled a leaf of the tree and the process is repeated until the all classes are covered or until another stopping criteria is reached. There have been many varieties of and additions to decision trees. Ross Quinlan proposed three of the most popular varieties [69, 70], ID3, CART and C4.5 respectfully, introducing the ability to process both categorical and numerical values whilst making the tree more resistant to outliers.

Trees are simple to understand and can be easily visually represented. Each path from the root to the leaf can also be represented as a human readable rule.

A popular iteration of decision trees are the ensemble learners Random Forests[9]. Ensemble learners combine many distinct instances of a trained classification algorithm, referred to as *weak learners*. Each tree is trained on a different sample of the data. Instances are then run through the forest and a classification is produced based on the votes from each tree. They have been proven to be a very effective tool in classification, the method of training gives them a resistance to noise and their structure lends itself to parallelisation.

ITRULE was developed by Goodman and Smyth [75] and produces generalised rules from batch data consisting of many nominal attributes. It evaluates every combination of possible rule terms using the J-measure, Equation 2.3 this is the product of the probability of the LHS of the rule occurring and the cross entropy of the rule, Equation 2.4. The J-measure is a uni-directional metric and so well suited to rule induction where cause and effect are encoded into the metric. The maximum information content of a given rule is bounded by the function Jmax, the maximum possible J-measure the rule can reach, Equation 2.5. Evaluating optimal partial rule sets from branch and bound approaches has received a good deal of attention in the literature. The authors of [80] define two types of partial rule evaluation, total cost and priority, the former evaluates potential gain of expanding on a rule term further whilst the latter evaluates the rule term alone. Priority evaluation is the far less computationally expensive of the two but can lead to suboptimal solutions. Calculating the value of Jmax enables ITRULE to stop specialising a rule when the J-measure approaches or reaches it's upper bound, which is a computationally efficient compromise between the two evaluations.

It uses Beam Search to keep the search space to a manageable size, only selecting the top N rules from the first iteration to expand upon. Additional rule terms are then appended to the LHS of the rule, specialising the rule. When every combination of the next phase of rules have been produced the top N are again selected and process continues until JMax is reached.

$$J(X : Y = y) = p(Y).j(X : Y = y) \tag{2.3}$$

$$j(X : Y = y) = p(x|y).\log(\frac{p(x|y)}{p(x)}) + 1 - p(x|y).\log(\frac{p(x|y)}{1 - p(x)}) \tag{2.4}$$

$$J_{max} = p(y).\max\{p(x|y).\log(\frac{1}{p(x)}), 1 - p(x|y).\log(\frac{1}{1 - p(x)})\} \tag{2.5}$$

An issue with ITRULE, and specifically the Beam Search approach, is that it is prone to falling into local maximums through partial rule dominance. This weakness may be amplified when applying it to noisy data. As mention above, however, some modification of the J-measure may improve performance when tested on a minority class such as a critical

alarm. An advantage of ITRULE over PRISM is that ITRULE does not attempt to classify all the training instances whilst PRISM will repeat until it has separated the classes into as many groups as required. Some versions also utilise a default class which can hamper it's reliability[37].

Prism Algorithms are a family of algorithms that have been developed around the original Prism algorithm [10]. Prism was developed in response to a problem with decision trees known as the repeated subclass tree problem. A tree is unable to represent disjointed rules without producing intermediary branches and nodes. The Prism approach can produce these rules without this excess rule production.

Still operating with a covering approach it distinguishes itself against the decision tree by trading the 'Divide and Conquer' for 'Separate and Conquer'. Separate and conquer, like decision trees, involves building a rule incrementally one term at a time. Every rule term covers a narrower subset which are removed from the main data set for further specialisation. When only one target class is left in this focus data set the rule is complete and the algorithm returns to the original data set without the instances covered by rules so far. This continues until all the data points are covered.

Other versions of Prism exist such as PrismTCS[8], which always partitions the data on the minority class; Random Prism[76], an ensemble learner based on the Prism family; and two adaptations for streaming data, eRules[77] and G-eRules[54].

Both ITRULE and Prism have the ability to abstain from the making a classification. Decision trees do not inherently have this ability as their structure requires a classification. Abstaining is needed to prevent a blind classification[37] which can increase the error rate of classifier in the absence of bespoke method of assigning a class when the value is uncertain [14]. This can occur when unseen feature values are presented to a trained model, when an instance falls inside a decision boundary or far from any class boundary[5]. Abstaining is not restricted to Rule Induction methods, Neural Networks naturally abstain if an instance does not result in sufficient stimulus. [71] .

A number of Rule Based techniques have been described above that learn a set of rules

from training data that can be used to predict the class of an unseen data point. The logic behind the classification is explicit within the model making these classification approaches *white box*. There exist many algorithms that are not transparent with their classifications though they are capable of producing accurate classifications. The next section describes a number of approaches taken from the literature that attempt to extract rationales from these black box models.

2.5.3 Rule Extraction Algorithms From Black Box Systems

There has been much work done on extracting rules from a black box classifier[38]. The goal of these techniques is to produce a white box system to closely mirror the black box system. To this end the systems accuracy is looked at alongside it's *fidelity*. The latter term is a measure of how well the two classifications match, independent of how well the predictions match the real class labels. Comprehensibility is a third evaluation criteria, it is a problematic measure as it is highly subjective and possibly unquantifiable. Quantifying aspects of the rule set, such as the number of terms used, may indicate a level of comprehensibility however this kind of evaluation may lend itself to a qualitative approach.

The focus of a lot of research on rationalising black box methods are on Artificial Neural Networks (ANN)[57] and Support Vector Machines (SVMs)[81]. ANNs aim to model a simplistic brain using perceptrons, neurons with inputs and a firing threshold. The connections between neurons are weighted allowing the ANN to be trained to produce a classification, methods of rule extraction for ANN are available from [78]. SVMs are designed to solved a margin optimisation problem by finding the points that form the decision boundary between two classes (the support vectors). Their widespread use stems from the computationally efficiently use of kernels to transform non-linearly separable problems to a different space.

There has been a considerable amount of work on the extracting rules from SVMs. A direct approach is to treat the SVM as a black box model and interpret it's output using a white box approach trained with the transformed output of the SVM[34]. To what extent this approach approximates the kernels decision function is not measured and a successful

application is dependent in the transformed labels creating a solvable problem through a white box approach. An advantage of the approach is that it is not algorithm specific and that the black box can be seen as a method of removing noise from the target class.

An alternative approach is to extract the support vectors and use these to define areas of the input features space[4]. Each support vector can be reclassified after the model has been trained to determine if it defines a positive or negative boundary and by determining the features that contribute the most to this classification an expressive rule can be produced. A number of these approaches are dependent on the input space being a set of boolean values[67]. This is often the case when the nominal features have been transformed into boolean vectors before inputting but would require an additional step to restore the data set to its original form to produce viable rules. Further techniques are described in [4].

2.6 Event Processing

This work is focussed around an event data set and so some time must be spent looking at how event data can be handled in a live environment. A contribution of this work is a book chapter on the subject[86], a summary is presented here. Two large fields of study around event data have been established, though the two fields are becoming increasingly similar: Event Stream Processing (ESP) and Complex Event Processing (CEP). CEP is the discovery of repeated patterns in low level events and the subsequent aggregation of these into more complex events. These complex events are assumed to be more meaningful and can then be passed on for further processing, perhaps to more machine learning[74]. Event Stream Processing is a branch of technologies that perform data mining on live event streams. Standard databases are not able to support the frequent polling required for ESP, instead they are built upon Active Databases or Data Stream Management Systems with a host of new event reactive query languages[11].

Where ESP focuses on the temporal relations between the events (the order in which events arrive is paramount), CEP takes a more abstract view enabling it to deal with larger and more complicated cloud based structures. CEP operates at a centralized location on

Partially Ordered Sets (POSET) where temporal relations are relaxed to a degree. Through this, problems such as network delays, a common occurrence in large, noisy networks, are less of a problem. CEP can both incorporate Machines Learning and/or support it. For example, Complex Events can be fed to a Frequent Pattern Mining algorithm as a cleaner and more abstract set of transactions, generating more general patterns across the breadth of the stream. Frequent Pattern Mining could also be incorporated in CEP to guide the amalgamation process. The same as been done using clustering[55]. Notably CEP can also be recursive and feed Complex Events back to itself for recombination with other primitive or complex events. Both these fields operate row-wise in that the other values contained within an event are not utilized, only the arrival times and type of event.

Complex Event Processing is a useful tool when dealing with bursty and noisy event data sets. Typical methods of dealing patterns include Association Rule Mining and Apriori[2] which work on item or transaction sets. The discrete values (items) are very similar to how events are approached. Apriori searches for frequent sets of items regardless of ordering (it's origins are in a time insensitive domain). FP-Growth [79] has been applied to event data in much the same way. FP-Growth differs from Apriori in that it uses a depth, not breadth, first approach which can yield larger transaction sets at lower supports. There are expansions to the basic algorithm where ordering is maintained. [51] proposes a way to represent bursty event sets as complex events by iteratively merging overlapping events to capture co-occurrence.

2.7 Conclusion

The goal of this work is, once it has been determined to that prediction is possible from this data, to produce human readable rules forecasting network alarms. This chapter has outlined some of the work done on event based data mining in the telecommunication (and more general) network domains. As the primary data set consists of alarm event data, a look at Event Processing was beneficial. A number of works in telecommunication have drawn from this field, particularly in terms of Frequent Pattern Mining. Aggregating events as seen

CEP can be used as an effective form of preprocessing for a Data Mining algorithm, especially if dealing with a noisy data set.

Some short comings in the existing works have been identified and possible solutions to these have been drawn in from research from other domains.

In section 2.3.2 were several approaches that produced accurate alarm prediction. A number of approaches [61] were not expressive and would require significant adaptation to become so. In 2.5.3 a number of methods to extract rules from black box systems were described, demonstrating that such an approach may be adaptable. [47] produced descriptive predicted rules accompanied by an estimated interval time by adapting Association Rules. This method produces one of the most expressive forms of rules but is dependent on producing frequent item sets with sufficient confidence which may not be feasible in this data set due to the bursty nature of events. The same is true of [45] though the base confidence of a rule may be increased by the inclusion of wild card variables. Alternatives to the frequent pattern approaches in forecasting are presented in [49][88][51][79].

In section 2.3.1 a number of expressive approaches to classifying and describing alarms were looked at. Approaches incorporating the J-measure have been used as a means to promote rules from a noisy data set [27][41]. The systems produce IF THEN rules and the beam search method of ITRULE offers a simple way to create an abstaining classifier, a desirable property for an algorithm classifying in a bursty environment. [18] uses Topographical Proximity for rule promotion, as focus data set contains some geographical location data this may be a good way to refine the rule set.

The outputted system will ideally produce descriptive rules that forecast and do so with data rich rules. The forecasting algorithms do not meet this criteria as they are and depend largely on an event type for their forecasting. A method of converting an expressive classifier in the line of [27][41] to forecast rules will be looked at in Chapter 4. In the following chapter a more detailed look at the alarm data and the pre-processing steps is looked at.

Chapter 3

Data Description

In this chapter the data sets that form the focus of this work will be outlined. Due to sensitivity of the data exact excerpts will not be given but statistical overviews and descriptions are permitted. This is done to give the reader an insight into some of the problems the data presents us with and some additional context for the over all problem of forecasting in a nationally gathered data set, provided by BT.

This chapter covers the segregating of the data into logical geographical clusters with the intention to strengthen the underlying concepts. The selection and transformation of features to those more suited to the Data Mining process. Special attention is paid to the transformation of time based attributes as these are expected to contain valuable information. Finally a description of the final data set is provided.

3.1 Data Description

By the end of this work a number of data sets were made available to work with. Figure 3.1 depicts the basic schema of the database used, each table contains descriptors for IP devices from the core network of the national telecommunication network, these devices are responsible for the transmission of packets from the cabinet across the backbone infrastructure. These are the alarms generated by each device to indicate a potential problem. Location data contains primarily the Northing and Easting values for each device that can be used

to determine inter-device distances. Incidents contain references to severe alarms that have been examined by engineers that can be classed as the most severe faults. Topology contains the SVLAN IDs for some of the devices, this can be used to examine the connected neighbourhood of each device.

Figure 3.1: Data Schema for main data sets

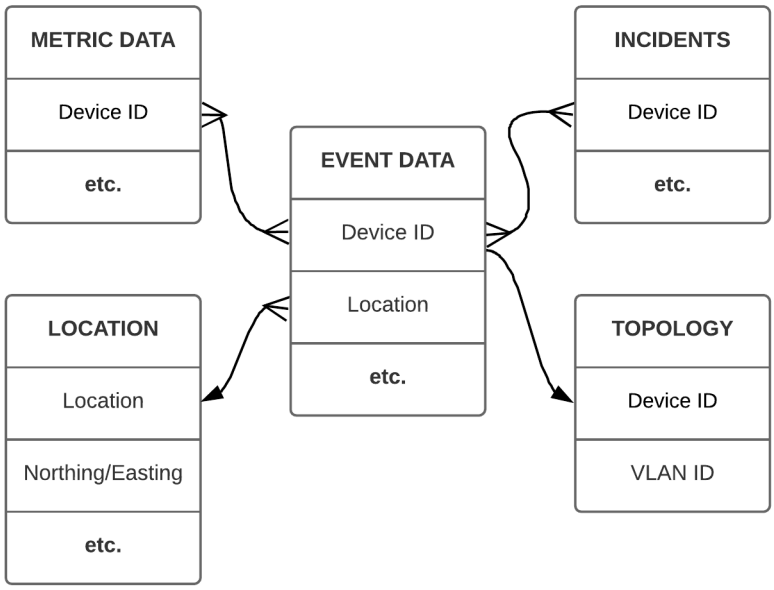

Table 3.1: Feature breakdown by data type

Data Type	Number of Features
Time	3
Integer	18
String	14

3.2 Data segmentation through GeoLocation features

The data set was divided into smaller sets based on their geolocation under the assumption that, given the scale of the network, there may be region dependent variation. Factors such as the density of devices in a cabinet, the age of the network and the climate will all vary depending on the location. It is expected that the vast majority of devices will be found in densely populated urban areas with less devices located in rural environments. Producing

clusters based around these population centres will capture both the dense urban and sparse rural network devices at the centroids and edges respectively. It possible that clusters with similar populations will create adequate out of sample sets for validation.

As the goal is to produce an unknown number clusters respectful of density a density based clustering method is best suited, DBScan[22] was used. DBScan is a bottom up density based clustering method that merges instances to form clusters based on two parameters: a distance from the point (η) and the number of other points that fall within this distance (ϵ). Some erroneous points can be removed as noise by checking their values against the physical boundaries of the UK.

The parameters were experimentally set with the goal of maximising the silhouette score, a measure of the average distance from each point each centroid. The highest silhouette score was achieved with an ϵ values of 20,000 and η of 600. Under these parameters 5 clusters are produced. Calculating the centroid of these points yields the locations of 5 major UK cities which is to be expected as these correspond to areas of high density. These cities loosely correspond to the highest population densities in the UK and it follows that the density of infrastructure would also be at it's highest. Points were then assigned a cluster based on the shortest distance to one of the 5 centroids as described in Table 3.2, the approximate clusters and sizes are depicted in Figure 3.2.

Table 3.2: Populations and radius of clusters

Region	Number	Population
Cardiff	0	169,056
London	1	780,043
Birmingham	2	182,934
Manchester	3	324008
Glasgow	4	155,059

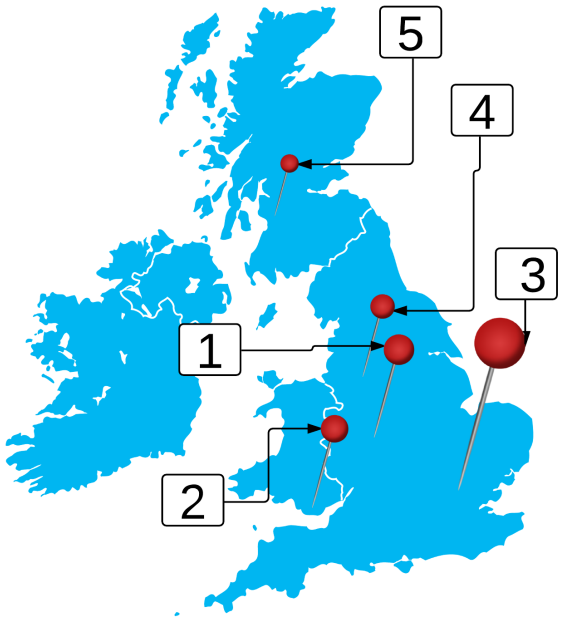

Figure 3.2: Approximate location of centroids in the UK

3.2.1 Feature Selection

The most influential method of feature selection came from domain experts. A number of features consist of manual entries and, as such, take place after the event was generated. These were identified and removed as they were not present at the time of event generation and may introduce an unwanted bias into the data set. Likewise, redundancy exists among some string attributes, as these features contain information found in other columns these were removed to decrease training times.

3.2.2 Locations Codes and String Entries

The easting and northing values of each event is provided by the location table, the key matching is performed over a location code. This code is in the format of [Area//Cabinet or Building//Room// etc.], manually entered and so often erroneous. 480 entries are lacking locations (0.02%). The precision of the location is reduced to only the area and cabinet. This loss of granularity allows a greater number of matches when joining the location table to the alarm data. The number of distinct locations is reduced from 2,650 to 2,360 by altering these location codes.

Some of the remaining unmatched location codes could be corrected through string matching using Levenshtein distance. Where the correction lies in the least significance part of the location code this carries very little risk. Corrections to the area code have a high associated risk of substantially relocating the event. Where there is more than one candidate for the corrected code this risk can be quantified by the maximum distance the event may move and a decision made as to whether to correct or exclude this event.Attempts to correct the remaining entries by string matching were unsuccessful due to many strings having multiple similar strings

Element Type is a very descriptive field in the alarm data set that contains a large number of distinct string attributes that can be costly to used for modelling. Unlike the location entries these do not all follow a predefined structure, they are, however, still hierarchical. A dictionary of root words was populated by collecting different strings for the same device and using sub-string matching to find the most simple form. Replacements were then made using this dictionary across the data set. This reduced the number of distinct entries from 127 to 25.

3.2.3 Binning of Numerical Features

Two numerical features were discretised into bins so they may be represented as categorical.

The boundaries were set intuitively from the features distributions. The distribution being roughly Poisson, there is more information in larger values than smaller. To that end larger bins were used for the higher, more sparse numbers, see Table 3.3.

Label	Duration	Occurrence
0	0	-
1	1-240	1
2	241 - 5995	2
3	5996+	3 - 888
4	-	889+

Table 3.3: Binning Boundaries for the numerical attributes Duration and Occurrence

3.3 Network Trends

Network performance is strongly linked to network usage, in-line with this there is an expected seasonality to the data. Figure 3.3 demonstrates the hourly and daily variation in the level of event generation. A timeseries for each day of the week is plotted from the aggregates of all events by hour. It is worth noting that the experiments detailed in this thesis are constrained to these 2 months of data, it may well be the case there is a wider seasonality to the data that cannot be seen from this sample.

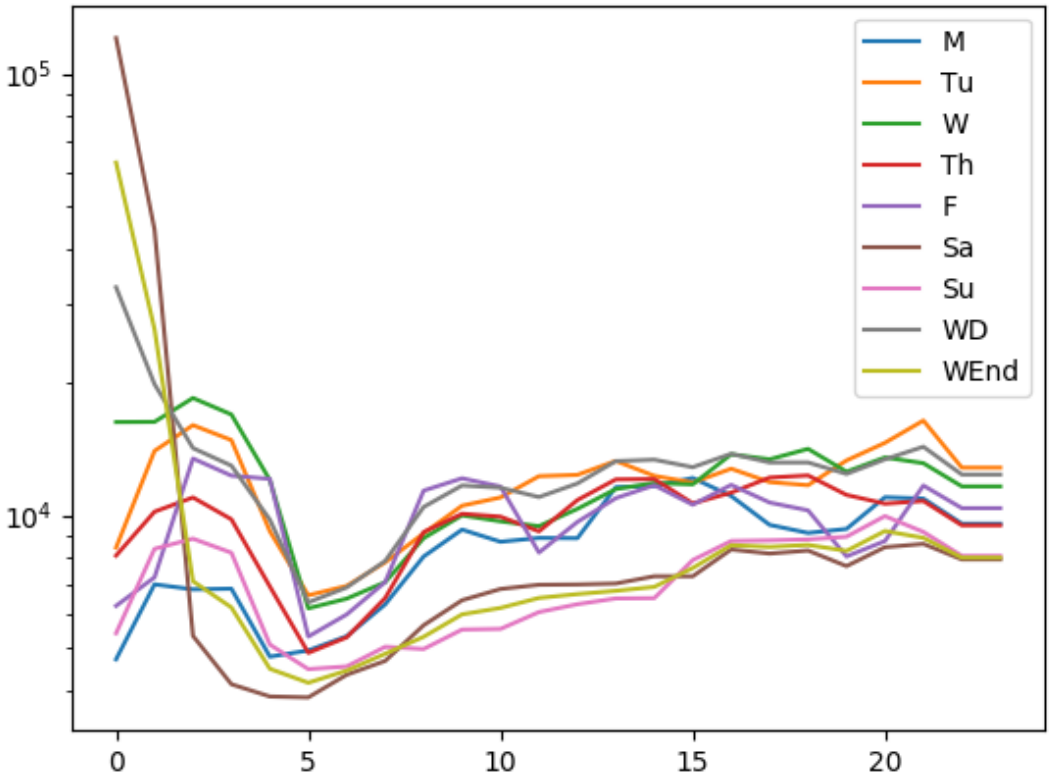

Figure 3.3: Aggregate of network events for the cluster 4 region by hour, Saturday and Sunday are identifiable

It can be seen from Figure 3.3 that there are two underlying trends, shown more clearly in Figure 3.4 which depicts smoothed lined for the weekend and weekday data.

Table 3.4 contains the number of events broken down by the day on which they occurred. On the weekend there are fewer alarms than the weekday, this is to be expected as there

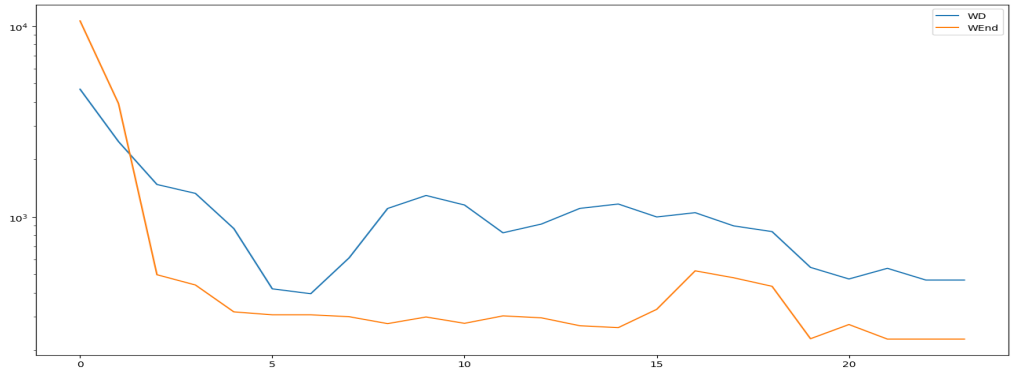

Figure 3.4: Smoothed average of the weekend (orange) and weekday events (blue) demonstrating two different trends

Table 3.4: The total number of events by the day of the week with a drop of nearly 50% from Friday to the weekend

Day	Count
Monday	208,466
Tuesday	290,371
Wednesday	275,977
Thursday	230,598
Friday	351,656
Saturday	157,379
Sunday	168,021

is a reduction in network traffic from businesses on the weekend. This can be represented in the data as a boolean feature to distinguish the week form the weekend. There is also a recurring pattern of peek times that should be reflected in the event data. There are peaks of error rates between 0pm and 5am. During this time a low usage is expected. This may still be the case as it is known that device restarts and engineering work takes place at night to avoid disturbing customer[84].

To capture this behaviour in the data a boundary for each part of the day was needed. Change Point detection was used to segment each daily timeseries into periods of similar behaviour. The data was divided into 8 timeseries of equal length representing the 8 weeks that make up the data set, from Monday to Sunday. A Bottom Up Clustering approach based on those outlined in [44] was used to create segments for each 24 hours with the start and ends of each segment registering as change points. There are 3 distinct observable behaviours

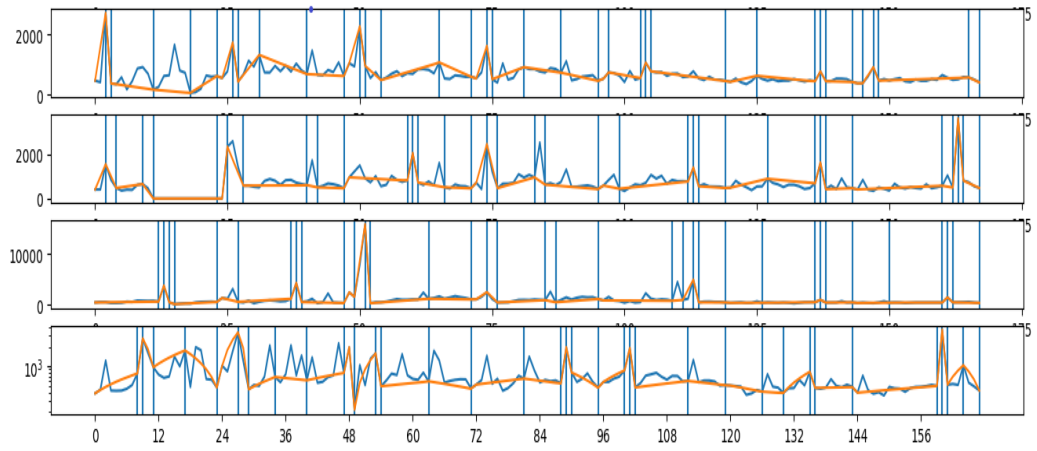

Figure 3.5: Change points along each week of the alarm data set. Orange lines denote the clustered projection of the line with the blue lines denoting the segment boundaries produced by a bottom up clustering algorithm.

in the network and so to capture these different behaviours 3 change points are needed. The algorithm is also guided by a manually set minimum distance between the change points, set conservatively to 2 hour. This is to encourage the algorithm to search the breadth of the series for change points rather than locating a particularly turbulent 4 hours. Figure 3.5 contains an image of the clustering process.

Figure 3.6 is a plot of the frequency of each change point to show the optimal set of boundaries. These boundaries were included in the data as an additional time based feature called 'Period' with the values *Night*, *Morning*, and *Peak*. Weekday was also included as a boolean value.

From Figure 3.7 a skewing influence can be seen on day 12, a Saturday, whose effect can be seen in Figure 3.3. It can also be seen by this breakdown that there is a good deal of variation from one day to the next.

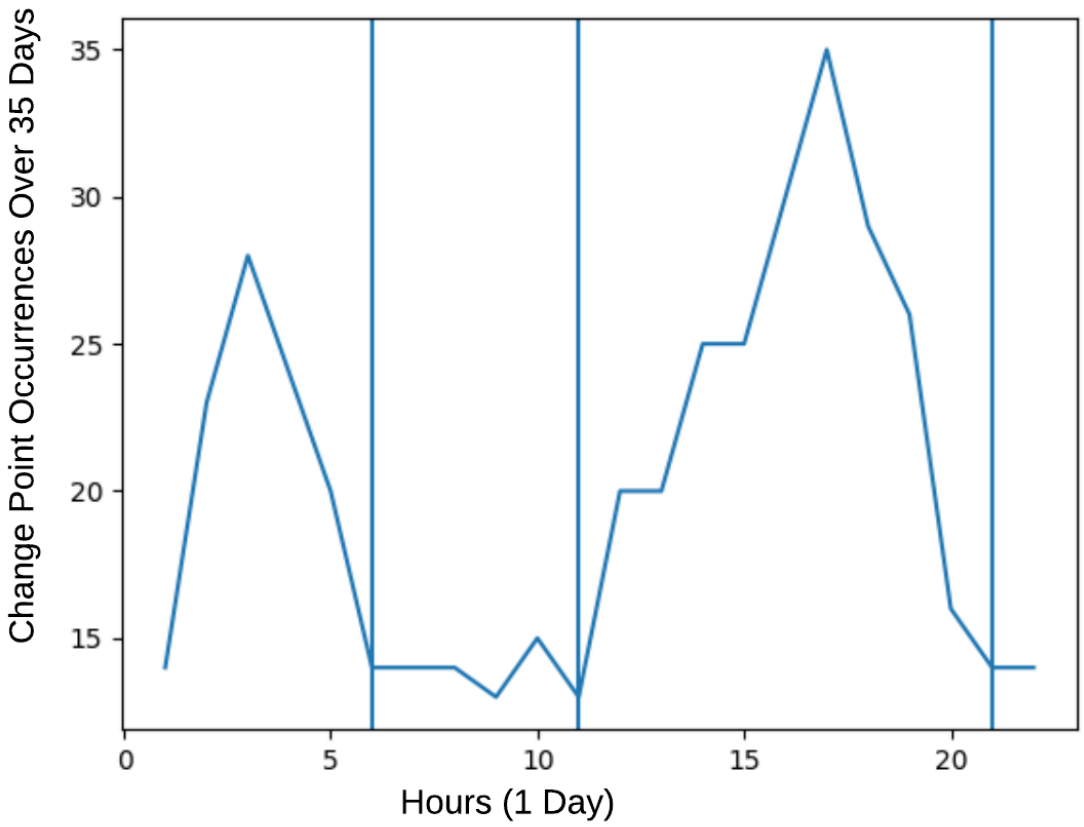

Figure 3.6: Plot of frequently occurring change points across each week of the data sample. Three different periods are observable.

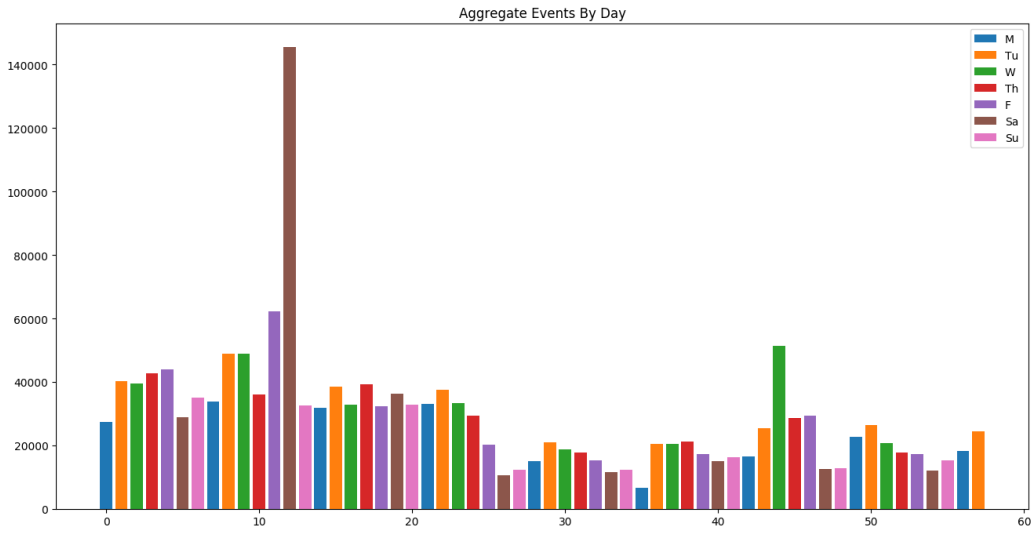

Figure 3.7: Daily event rates across the 2 month sample, the second Saturday is a strong influence on any aggregate plot.

3.4 Event Filtering

To detect and remove outliers from the static data set 3 measures were looked at.

- Total appearance - the proportion of times an event type appears in the whole data set
- Variance of daily occurrences
- Burst parameter - a measure to describe the burstyness of a series ranging from -1 (periodic) to 1 (random) with 0 equating to a regular Poisson distribution

Together these metrics give an indication of which events are likely to be background noise. Events are generated when a metric recorded on the device breaches a threshold. These thresholds are set manually by engineers and are often not tuned[84]. An imprecisely placed threshold can lead to a large number of alarms being generated as the device is often operating close to the set threshold. A low daily variance indicates that the event appears consistently with similar levels, this combined with a high probability of occurrence is a strong indicator of background noise. A similar indicator is a strong negative burstyness. A plot of

all three parameters can be seen in Figure 3.8, the events have little spread across the axis, with the majority having similarly low appearances, low variance and positive burstyness. Figure 3.9 is a histogram featuring those event types whose daily variance is less than 1. It can be seen that the majority of this sample belongs to the 0-0.1 bin, pruning these will result in the loss of a large amount of information so the decision was made to retain these alarms.

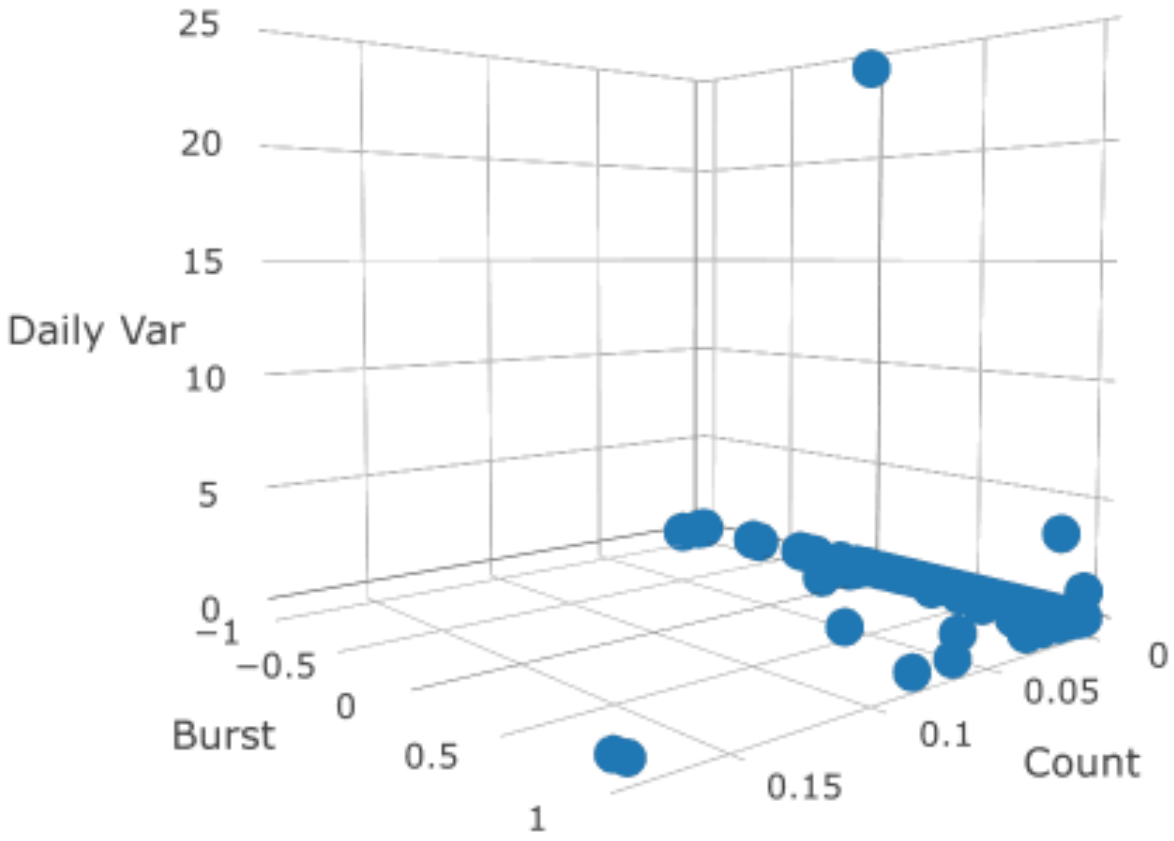

Figure 3.8: Identifying outliers through their scaled variance, burst parameter and count of all distinct events

The goal of the eventual system will be to predict an alarm representing a serious fault on a device. These are presented by a number of alarm types that will be generally referred to in this work as *Down* events. The relative frequency of this combined event class is high, Figure 3.10 shows three groups of events that make up the compete corpus. Two dominant event types are visible, Down and HighUtilization, the third group is the remaining 287 types. The Down event group can be broken down even further into device type and model, see Figure

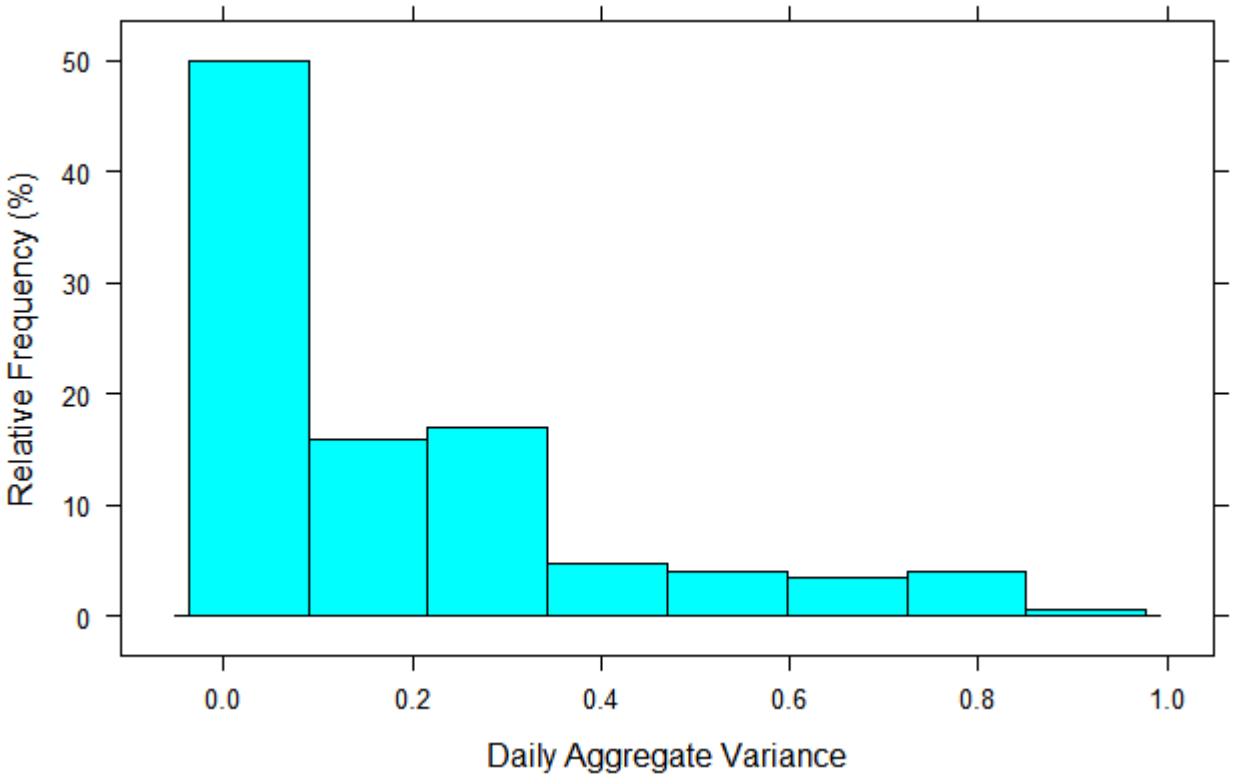

Figure 3.9: Selection of variances of daily aggregates below 1.0 with their relative frequencies, over 50% of this data sample has a variance close to 0

3.11. At this scale it is apparent that one model of device is responsible for the majority of down events, so much so that they may be considered background noise.

The outliers were selected using the criteria from Figure 3.8, these outliers include the noisy down events. This reduces the number of instances in the event set from 1,682,468 to 916,517 a reduction of 46%. The new event type balance is displayed in Figure 3.12.

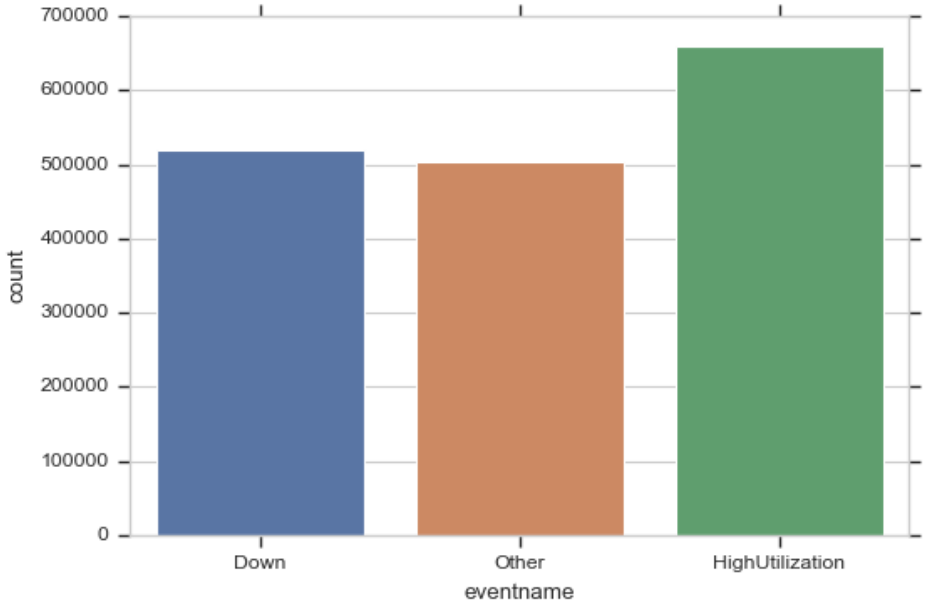

Figure 3.10: Volume of event name classes, Down, HighUltilization and Other with number of subclasses for each class

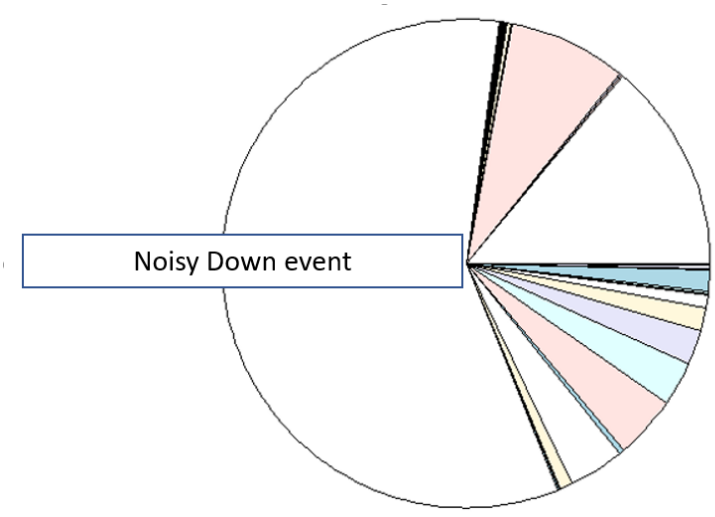

Figure 3.11: Relative frequencies of Down events broken into device type and model

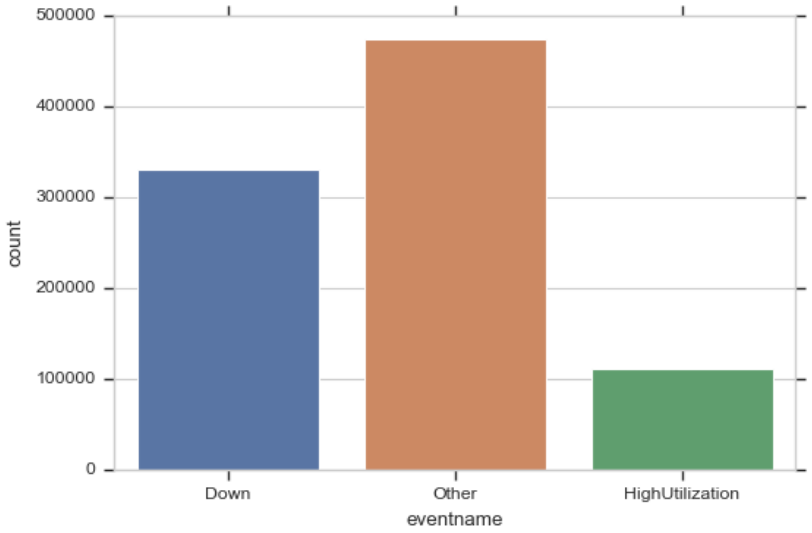

Figure 3.12: Volume of event name classes, Down, HighUtilization, and Others with number of subclasses for each class for data set after filtering

3.5 Interval Merging

Repeated events are merged if they have the same event type, host device and timestamp. The duration, end date and occurrence count were altered to reflect these merges. The occurrence count has very little correlation with the duration, suggesting that this number is not reliable.

3.6 Discussion of Data

This chapter has detailed the steps taken to transform the data from its raw format to a state more conducive for pattern extraction. A large number of features with no value have been removed to lower the dimensionality of the data, reducing the problem in complexity and making it more efficient to fit a model with.

A number of additional features were also introduced that capture some of the temporal and geo-spatial information in categorical features, making them trivial to represent in a human readable rule. In the next chapter this data set will be used to train and test a number of models to determine how well it lends itself to a classification problem. A further

transformation will also be detailed that enables a model to forecast alarms.

Chapter 4

Rule Induction Approaches to Forecasting Critical Alarms

In Chapter 3 the data set was introduced along with a description of the pre-processing steps that were applied to the data set ready for experimentation. In this chapter a number of these experiments will be described. The goal of this chapter is to produce a system that demonstrates that the forecasting of down events is an achievable goal. In line with the requirements of this project this system must be highly expressive and with a high precision.

This chapter will begin with the rationale behind the choice of algorithm derived from Chapter 2. Some problems are highlighted and a solution implemented resulting in a new Rule Induction classifier. This is evaluated against the original implementation. Afterwards a transformation to the data that allows these algorithms to forecast events is experimented with.

In the previous chapter a number of approaches were identified from the literature that could be used to meet this projects goal. These were examined for their suitability and a case was made for the use of ITRULE as a rule induction technique. The algorithm has been associated with alarm description successfully before and it has a number of properties that make it more suitable for the approach outlined in this chapter. ITRULE is an abstaining classifier, it is not forced to classify ever instance given to it which can a restriction of other white box models such as decision trees. As the data this work is concerned contains a

majority target class whose prediction is not of interest, this abstaining property is desirable. The goal is to produce expressive rules that can forecast an event. The forecasting criteria will be approached through a transformation to the data outlined in Section 4.3.1. The goal of the transformation is to allow the forecasting of events both expressively and using a variety of features. The event forecasting techniques looked at in the previous chapter limit the feature space to an *alarm type*, i.e. a primary descriptor for that alarm. It may be the case that other features are also important to the forecasting process, the pre-event method of forecasting is designed to forecast using a wider feature space. This may, in turn, allow the forecasting of events that the previous methods are unable to. This method runs the risk of introducing a lot of noise in the data set, the ability to abstain will reduce the amount of noise held in the model.

4.1 CEP on the Whole Data Set (OGRI)

Online Generalised Rules was proposed as a method of extracting informative human readable rules from a data stream. It was divided into multiple sections as seen in Figure 4.1.

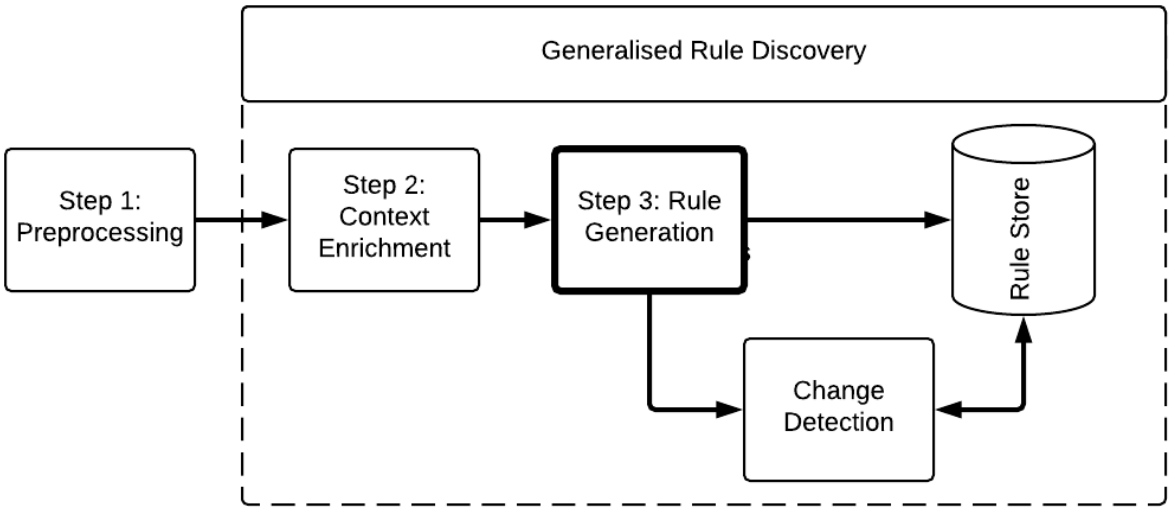

Figure 4.1: System diagram for Online Generalised Rule Induction

1. Pre-processing - Each event in the stream under goes Pre-Processing, as discussed in

Chapter 3, to transform the raw features into a form more suitable for pattern discovery.

2. Context Enrichment - This refers here primarily to co-occurring events, events are active at the same time. This is not necessarily because they originate at the same time point but an events duration may cause it to overlap with another event. As the BT data set has an number of proximity based features these can also be used to promote classifications.

3. Generalised Rule Discovery:

   (a) Rule Generation - Generalised Rule discovery refers to a particularly expressive form of rule induction where by both the LHS and RHS of the rule contain multiple terms. The rules can be generated in turn by two different forms of rule induction.

   (b) Rule Store - These rules, once generated, are placed in a rule store for monitoring and updating.

   (c) Change Detection - This is used to monitor any drifts in the produced rules, these are important to monitor from a network maintenance perspective, as concept drifts may indicate potential problems, and to ensure that the model is kept up to date. The ideal change detection algorithm will monitor the produce rules, two examples of such an algorithm are TRCM [1] and the event change technique proposed in [56].

The design of the initial system was laid out in the authors paper [85]. In this paper ITRULE is selected as the rule induction component. It was chosen in part for of it's use of the J-measure to promote rarely occurring events over populous ones which is also an implicit method of ranking the rules produced. It is also not forced to make a classification as in the case of decision trees. In a large complex network it is unlikely that the complete set of feature values will be captured in the training set, ITRULE is able to cope with this though previously unseen values should be captured elsewhere in the framework. Examples of ITRULE's use in other classification problems are outlined in Chapter 2.

All experiments in this chapter are conducted using alarm data from the 4th cluster as defined in Chapter 3 and using 3 fold cross-validation. Cluster 4 was chosen as it is of comparable size to two other clusters, providing an option to use an out of sample data set for final testing. It is also, of the 3 similarly sized clusters, the most removed, making it less likely to be impacted by events in other clusters under the assumption that the topographical and geographical network are similar. Results in this thesis are often expressed as scatter plots and a fitted logistic regression along with a shaded confidence boundary of this regression. This instance of ITRULE is based on the original paper [75] with an additional hyper-parameter to limit the maximum length of a the LHS of a rule for cases where rule's J-measure does not approach Jmax, this is set to 4.

ITRULE was found to converge very quickly to local optima, a problem with the beam search approach caused by the greedy evaluation of early partial rules. Beam search is a technique designed to limit the search space for combinatorial problems. After evaluating all possible root nodes they are ranked and only the top n are retained and expanded upon. This value n is referred to as the beam width. Figure 4.2 demonstrates the performance of this algorithm through it's precision and recall as the maximum beam width parameter is increased. The maximum beam width is the upper bound on the number of rules that can be taken forward for specialisation at each stage (see Figure 4.4). Figure 4.3 shows the tentative accuracy and accompanying abstain rate for the same classifier. The abstain rate is expected to decrease as the beam width expands as the additional rules allow for a wider classification. These graphs indicate that ITRULE generates a very strong first rule and that subsequent rules only diminish the accuracy. This in inferable from the downward trend in 4.3 as more rules are added to the beam.

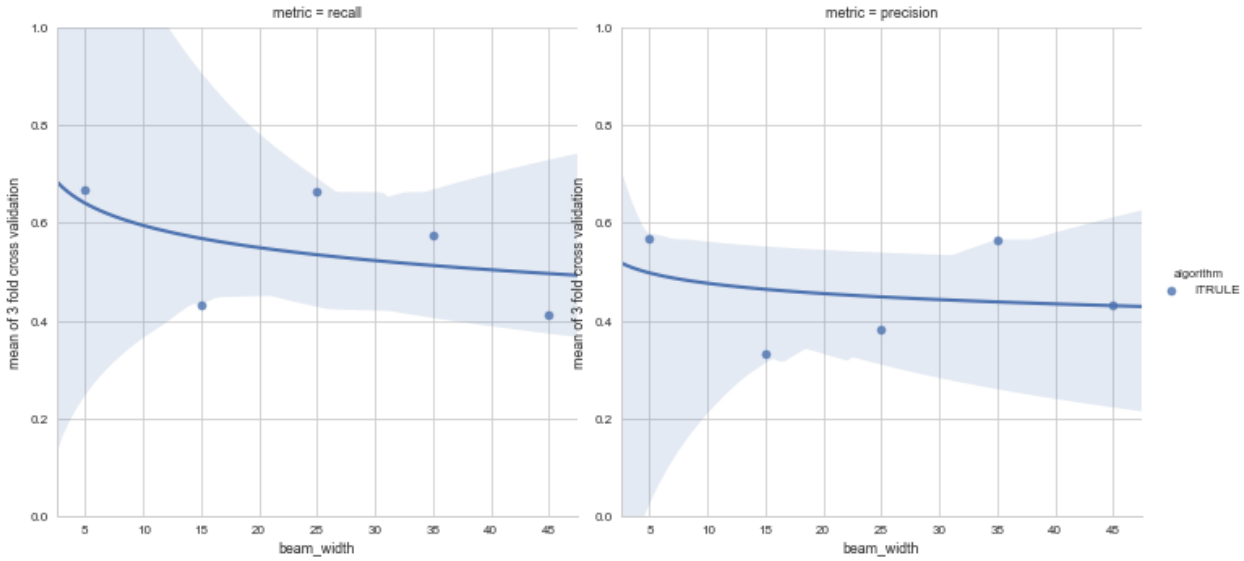

Figure 4.2: Precision and recall of the ITRULE algorithm across a range of beam widths

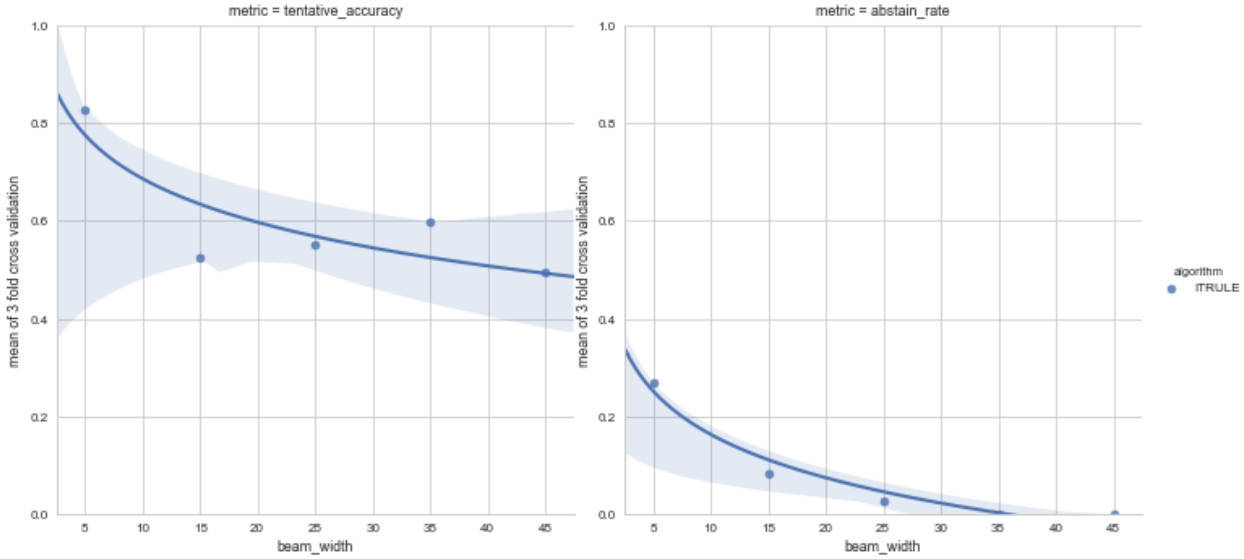

Figure 4.3: Abstain rate and tentative accuracy of the basic ITRULE classifier on the event data

Beam Search evaluates each rule without regard to the strength of the global rule set[17]. Key to this is the idea of dominance [16], when a solution does not weaken the rule set as a whole but offers an improvement in at least one case. In such a scenario the rule that offers the marginal improvement is said to dominate the other. Using the Beam Search heuristic, a candidate rule may dominate another but this niche case offering the marginal improvement is not evaluated. A rule is only added to the beam based on the driving metric, if it is not

added then a partial rule may be irrecoverably lost as it's root node, or rule term, is left out of the beam[80]. This applies in reverse as a candidate rule with a higher value will cause the displacement of a dominating rule within the beam already.

The problem is compounded when there is a large disparity between the assigned worth or goodness of a feature value. For example, if there is a large difference between the number of distinct values belonging to an attribute so that the probability of $P_a >> P_b$. As the J-measure is a product of the probability of the LHS of a rule, those features with fewer distinct values contribute to much higher J-measures than those with many values. If not accounted for, these feature values will repeatedly be selected for further specialisation to the extent that they appear in a disproportionate number of rules within the beam. This in turn leads to rules which share very similar conditions and in turn cover a very narrow range of features. This is computationally wasteful as the same tests are carried out between rules with shared terms (a problem shared with decision trees[10]) and leads to a very narrow coverage of the feature set, which in turn will decrease the algorithms accuracy.

For example, in the below, feature values A_1 and B_1 have a high probability of both occurring and being selected in the top N rules. This leads to very similar rules with low coverage:

$$\text{IF } A_1 \text{ AND } B_1 \text{ AND } C_1 \text{ THEN } X$$

$$\text{IF } A_1 \text{ AND } B_1 \text{ AND } C_2 \text{ THEN } X$$

$$\text{IF } A_1 \text{ AND } B_1 \text{ AND } C_3 \text{ THEN } X$$

This rule set has a large amount of redundant information and, given the assumption that the feature space has been exhaustively represented, space could be preserved in the beam for other rules if they could instead be presented as the root rule: :

$$\text{IF } A_1 \text{ AND } B_1 \text{ THEN } X$$

This can be addressed either during induction or with the inclusion of a pruning stage. The pruning stage does not allow for the recovery of lost rules making an alteration to the induction process the preferred method. A hypothesis going forward is that the an increase in the variety of features in the beam is likely to improve ITRULE's performance,

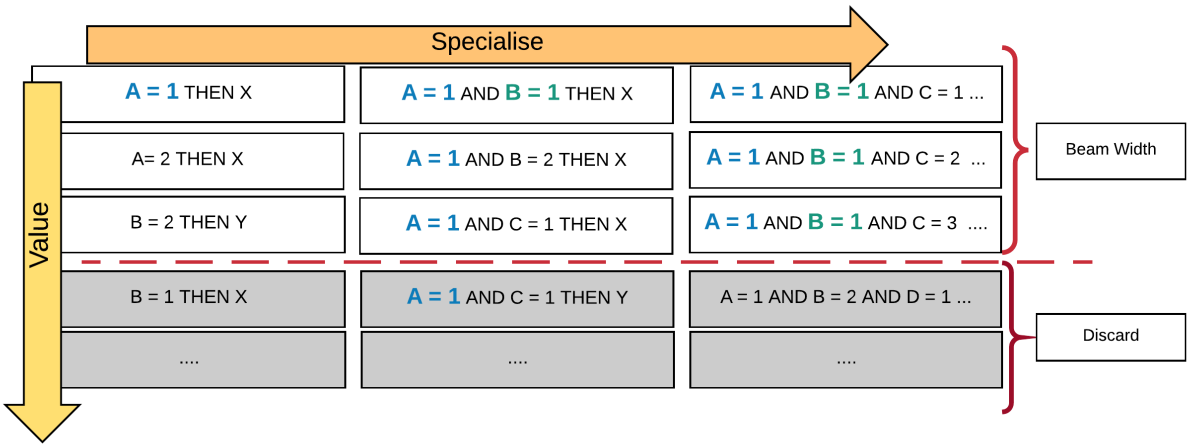

Figure 4.4: The beam search approach

a pruning stage will have no benefit in this regard either. The following section details some alterations to the ITRULE algorithm in order to mitigate against this problem and produce more meaningful rule sets.

4.2 Rule Induction Prediction and Local Maximas

As mentioned in the previous section there are issues with local maxima that can hamper the predictive capability of ITRULE. In this section a number of alterations to the ITRULE algorithm are described.

To demonstrate the local maxima problem the top five rules generated by ITRULE on the Cars[19] and Congressional[73] sets are listed below in Table 4.1 and Table 4.2. Particularly in the Congressional data set the top 4 rules consist of the same feature and values. Only when the cartesian product of this feature has been exhausted is a term featuring an alternative feature included into the rule set. This is particularly problematic as rules 2 and 4 will never fire as their LHSs' already appear in a preceding, higher priority rule.

Table 4.1: Top 5 rules generated by ITRULE on the cars data set

Rank	LHS	RHS	P(x| y)	P(x|ny)	pY	pX	J
1	maint: med	acc	0.2662	0.7337	0.25	0.2222	0.0019
2	buying: med	vgood	0.06018	0.9398	0.25	0.0376	0.0021
3	maint: med	vgood	0.06018	0.9398	0.25	0.0376	0.0021
4	maint: low	vgood	0.0601	0.9398	0.25	0.0376	0.0021
5	buying: high	unacc	0.75	0.25	0.25	0.7002	0.0022

Table 4.2: Top 5 rules generated by ITRULE on the congressional data set

Rank	LHS	RHS	P(x| y)	P(x|ny)	pY	pX	J
1	h infants: F	D	0.4322	0.5677	0.5425	0.6137	0.0526
2	h infants: F	R	0.5677	0.4322	0.5425	0.3862	0.0526
3	h infants: T	R	0.1657	0.8342	0.4298	0.3862	0.0718
4	h infants: T	D	0.8342	0.1657	0.4298	0.6137	0.0718
5	synfuels : T	R	0.14	0.86	0.3448	0.3862	0.0736

This pattern is seen again when examining the output rules from the ITRULE experiments run on the alarm data set as shown in the following rules:

Rule 1: **IF** elementtype: elementtypeA **THEN** isproblem: 0

P(x|y) 0.39363 P(x|ny) 0.60636 pY 0.22476 pX 0.62157 j 0.153006 J 0.03439 JMax 0.19106

Rule 2: **IF** eventname: eventnameB **THEN** isproblem: 1

P(x|y) 0.000008 P(x|ny) 1.0 pY 0.05298 pX 0.37842 j 0.68596 J 0.03634 JMax 0.03634

Rule 3: **IF** eventname: eventnameB **THEN** isproblem: 0

P(x|y) 1.0 P(x|ny) 0.0 pY 0.22476 pX 0.37842 j 0.68599 J 0.03634 JMax 0.03634

Rule 1 is a very high performing rule as seen from downward curve in Figure 4.3, each subsequent rule detracts from all 4 displayed metrics. Rule 2 is a low performing rule and rule 3, shares the same LHS as the preceding rule and so never fires.

Randomness is often employed to non-exhaustively explore the search space and find alternative candidates, approaches described in Chapter 2 employ techniques from Evolutionary Computation[83] or Random Forests[9]. There are several places such an alteration could be made to the ITRULE algorithm. Rather than populate the beam with systematically generated candidates (prone to partial rule dominance) a warm start could be employed with

a random set of partial rules. This approach is unlikely to have an affect on the outcome as the partial rules are phased out through subsequent iterations resulting in a the same partial rule dominance issue.

A simplistic approach to handling this is to monitor the features and values already inside the beam width and prevent the addition of new rules whose terms already make up a threshold percentage of the beam. Such a variant of ITRULE was developed during this research and was included in the publication [87], one of the contributions of this research.

To mitigate against partial rule dominance a form of pre-pruning was incorporated. When selecting the candidates for the next iteration of rule specialisation, the rules are ranked according to their respective J-measures and the top N are added into the beam for specialization. When a rule is added the distribution of the features within the beam is updated. Here a rule is only added if this rule's individual member features do not surpass a threshold proportion of the beam (set to 40% for these experiments), otherwise it is deemed to be predominant and not included. It is possible to exhaust the list of candidate rule terms with this method if the beam size is sufficiently large or the candidate list is sufficiently small. This being the case, the list (minus the rules already selected) is iterated again until a sufficient number of rules have been selected and the threshold deterministically relaxed with each iteration. The algorithm is referred to in the publication as ITRULE with Partial Rule Dominance correction (PRD). This approach leads to a more varied rule set though it has the disadvantage of on occasion excluding a rule with high a J-measure in preference for a new minority rule term with a lower J-measure. By enforcing a wider variety of features in the beam the algorithm is more resistant to local maxima, this may increase the number of rules that are of interest to the domain experts.

In [87] the ITRULE PRD algorithm was compared with the original on a classification task with three different target classes from a transformed version of the BT data (this transformation is outlined in more detail in Section 4.3.1). These three classes were chosen due to their varying level of class imbalance, Table 4.3 details the aliased class labels and their proportion of the data set. For these experiments a beam width of 45 was used and the

Table 4.3: Class proportion of target events

Data Set	Class Breakdown	Percentage Target Class
A	A_i	7.41
	A_{ii}	6.92
	A_{iii}	4.24
B	B_i	69.55
	B_{ii}	35.50
	B_{iii}	66.16
C	C_i	0.68
	C_{ii}	0.73
	C_{iii}	0.51

value of Jmax was set to 80% of its full value to combat over fitting.

Table 4.4 displays these results. In terms of accuracy and tentative accuracy the results are very similar, the accuracy for predicting A events are higher due to a much reduced abstain rate. The trade-off in the variance in execution time is higher in all cases, this is due to a large amount of additional tests needed to populate the beam size in such a heavily skewed data set. Even with the inclusion of Partial Rule Dominance correction the overall accuracy is low and the abstain rate is high.

Table 4.4: Results for ITRULE and ITRULE PRD

	Basic ITRULE			ITRULE PRD		
Data Set	Acc%	Tent-Acc%	Abs Rate %	Acc %	Tent-Acc%	Abs Rate%
A_i	23.03	93.75	75.44	36.63	92.87	60.57
A_{ii}	35.92	92.94	61.35	49.07	95.93	48.85
A_{iii}	33.73	93.47	63.92	30.51	93.49	67.37
B_i	99.16	99.31	0.15	76.83	99.14	22.51
B_{ii}	99.49	99.49	0.00	92.56	99.47	6.96
B_{iii}	95.13	99.26	4.17	43.54	99.16	56.09
C_i	0.92	45.75	98.00	5.43	68.52	92.08
C_{ii}	0.37	18.25	98.00	0.42	18.66	97.75
C_{iii}	1.20	59.01	97.98	0.78	37.05	97.90

The method employed by ITRULE PRD is a brute force approach and in these test does not offer a notable improvement over the base algorithm. In the following section a more successful extension to ITRULE PRD is described.

Simulated Annealing and ITRULE

Simulated Annealing[60, 46] is an approach inspired by metal work that uses a cooling temperature and an alpha parameter to dynamically alter the probability of an event. A natural progression from ITRULE PRD is to incorporate the annealing mechanic to determine the probability of a rule being accepted into the beam. With every accepted rule term the global temperature cools and the acceptance probability decreases, keeping the search space broad for the high energy initial stages where the local optima problem is more severe. This version of the algorithm is referred to as *ITRULE Annealing*.

ITRULE Annealing introduces two new parameters to the original ITRULE algorithm, α and *temperature* to control the rate of cooling. If a rule's value, in this case the J-measure, is not sufficiently high to be added to the beam width an additional test is carried out using these parameters. The acceptance probability (see equation 4.1 and 4.2) is calculated from the difference between the lowest J-measure inside the beam and the candidate rule's J-measure.

$$\delta = newJmeasure - oldJmeasure \tag{4.1}$$

$$P = \exp(\delta / temperature) \tag{4.2}$$

If this value is greater than p, a randomly generated value drawn from a uniform distribution then the candidate rule replaces the lowest valued rule in the beam width. The temperature is reduced by the coefficient α upon every successful addition to the beam width through a cooling strategy making each subsequent addition less likely. Algorithm 1 outlines the program flow of ITRULE Annealing. A high starting temperature combined with a high value of *alpha* will result in a very slow cooling algorithm, the energy will remain high for longer allowing lower scoring rules into the beam more frequently. Vice versa a low starting temperature and a low *alpha* will cool quickly and the acceptance probability will tend towards 0, creating in effect the basic ITRULE algorithm. The rate of cooling is also tied

into the beam width and cardinality of feature values, i.e. the number of tests made for each iteration of the algorithm.

Algorithm 1 The algorithm for ITRULE with Simulated Annealing. An additional step to the original algorithm is added at line 11

```
1: Data set D with Target Classes C_n and Attributes A_n
2: for Every Class C_i in D do
3:   while D contains classes other than C_i do
4:     for Every Attribute A_i in D do
5:       for Every Attribute Value A_iv in A_i do
6:         Generate Rule R_n with Rule Term A_iv
7:         Calculate j
8:         J_max ← J_max * ThresholdT
9:         if j > J_min(S) then
10:          add R_n to Candidate Rule Set
11:        else if p < P then
12:          add R_n to Candidate Rule Set
13:          Temperature = Temperature * α
14:        end if
15:      end for
16:    end for
17:    Select R_n where j(R_n) is maximised
18:    Remove Instances not covered by R_n
19:    if j(R_n) > J_max(R_n) then
20:      Rule Complete
21:      Break
22:    end if
23:  end while
24:  for all Rule Terms RT_i : in Rule R do
25:    if Confidence( RT_i) <Confidence(RT_{i-1}) then
26:      Remove RT_i from R
27:    end if
28:  end for
29:  C_i ← Remove Instances Covered by R_n from D
30: end for
```

The metric used to establish the acceptance probability need not be the same metric that drives the basic algorithm. The values Jmax and Jdistance, both derived from the J-measure, can be used instead.

Jdistance was another contribution from the paper[87], there it was used as an evaluation metric for the rules produced on the BT alarm data set. It is calculated as the difference between a rules J-measure and Jmax, a theoretical measure of how optimised a rule was. In

Table 4.5: Class proportions for BT data set problem target class

positive proportion	negative proportion
0.392	0.608

practice a rule does not always approach it's maximum possible J-measure (Jmax), which can be an order of magnitude higher. While Jmax and Jdistance should promote rules with a high maximum information content, Jdistance will also promote rules that have a larger potential gain in J-measure which is fitting for a partial rule term.

The ITRULE Annealing algorithm was run on the BT data set to predict problem events to establish the most effective parameters. Problem events are alarms manually labelled by engineers with a minor class imbalance (see Table 4.5). The algorithms were trained and tested on alarm data from the 4th cluster using 3 fold cross validation and varying the values of α, starting temperature, beam width and the driving metric. Table 4.6 contains the mean and standard deviations of the key metrics across the tests run aggregated over a range of beam widths. *t_accuracy* is the tentative accuracy of all classifications made, it differs from accuracy in that an abstain from classification does not penalise it. Tentative accuracy's lower bound is the accuracy and it's upper bound is 1.

Tables 4.6 report very similar results across the three driving metrics. The average recall of the system is quite low and the abstain rate is high (which will always result in a low accuracy). The precision is higher along with the standard deviation, both precision and recall will be effected by the change in class imbalance created by increasing the beam width so this deviation is expected.

Four plots in Figure 4.5 show the variation of the precision under different starting temperatures. Higher starting temperatures keep the acceptance probability high in the initial stages and, in this case, do not seem to have had a large impact on the reported precisions. The high number of distinct feature values will create a large number of tests for each iteration of the beam, the starting temperature on the exponential cooling schedule will quickly decrease. A possible avenue of research to improve this algorithm would be to tie these parameters into the number of expected tests, a value correlated to the dimensionality of the

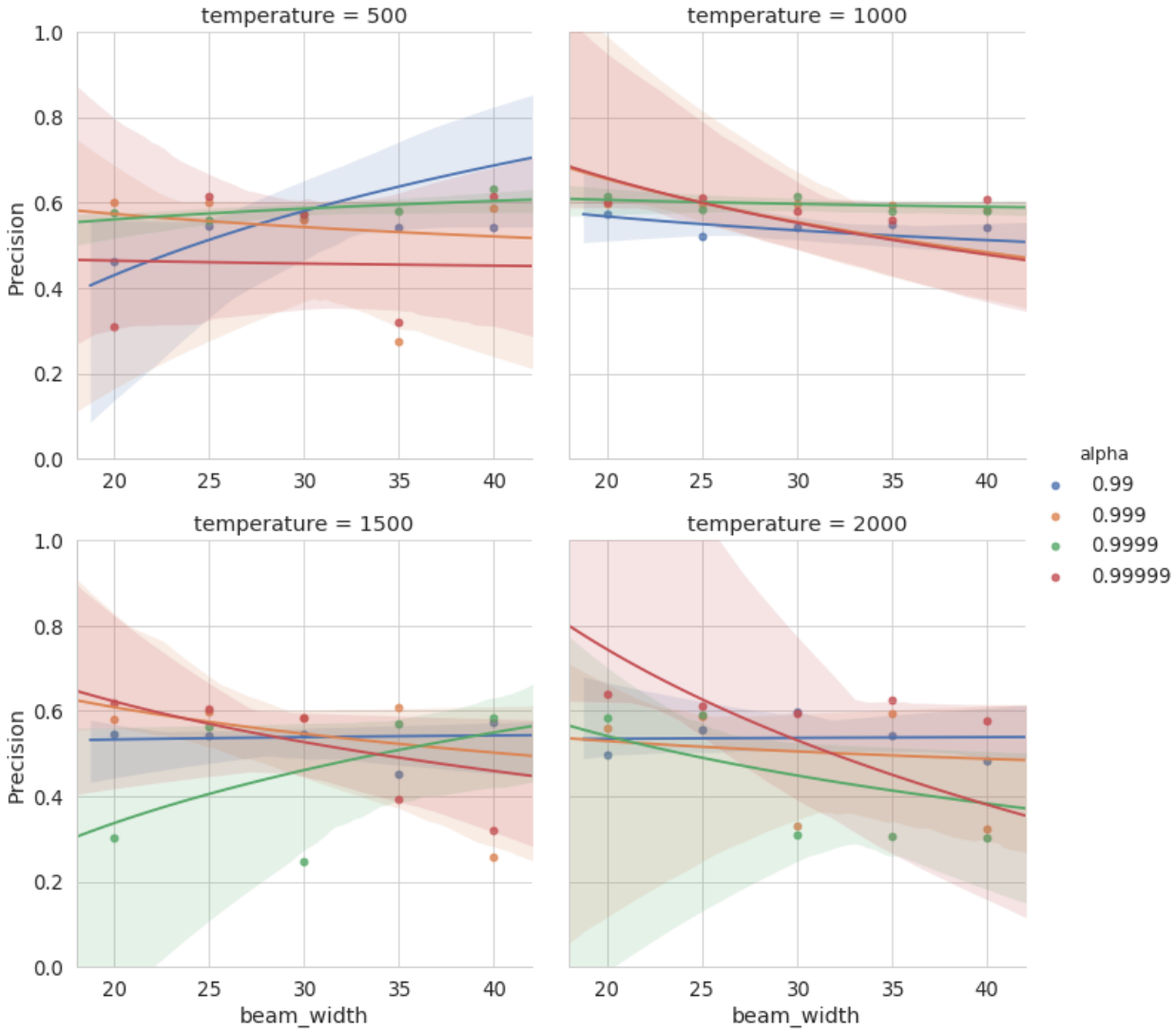

Figure 4.5: Precision of ITRULE Rule with Simulated Annealing with varying parameters for alpha and starting temperature over a range of beam widths

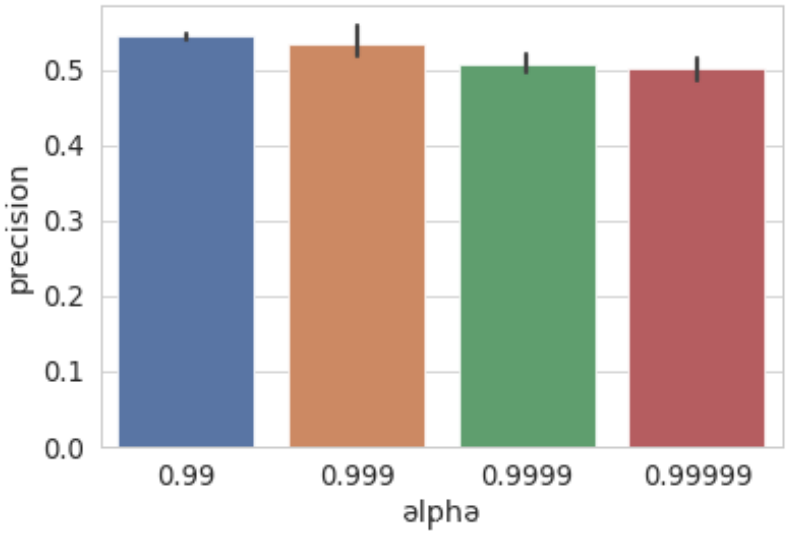

Figure 4.6: Mean precision of each alpha aggregates across starting temperatures and beam widths

data. Each series in the plot is generated from the algorithm run under a different α value. As α controls the rate at which the temperature decreases this has a large impact on the results. The highest value of α, 0.99999, creates a nearly random set of rules and this is visible in the lack of consistency in the recorded precisions. This is because the temperature is never sufficiently cooled to make the inclusion of a beam an unlikely possibility. In some cases, particularly when the temperature is high, it produces the highest precisions but when aggregated across the range of parameters it has a marginal shortfall in precision (see Figure 4.5).

This variation is greater still in the reported recall in Figure 4.7 though the recall at an α of 0.99 looks to have resulted in a more stable system. 4.8 demonstrates the variation in cooling rates given a temperature and α against then number of iterations across the BT data search space. Any value temperature above the 10^{-1} is likely to result in immediate acceptance into the beam so an α must chosen that quickly lowers the temperature past this point. Other cooling strategies have been proposed in recent years that may be beneficial to try[39]. This is further illustrated in Figure 4.9, a heat map of the standard deviation of the precision with varying starting temperatures and αs.

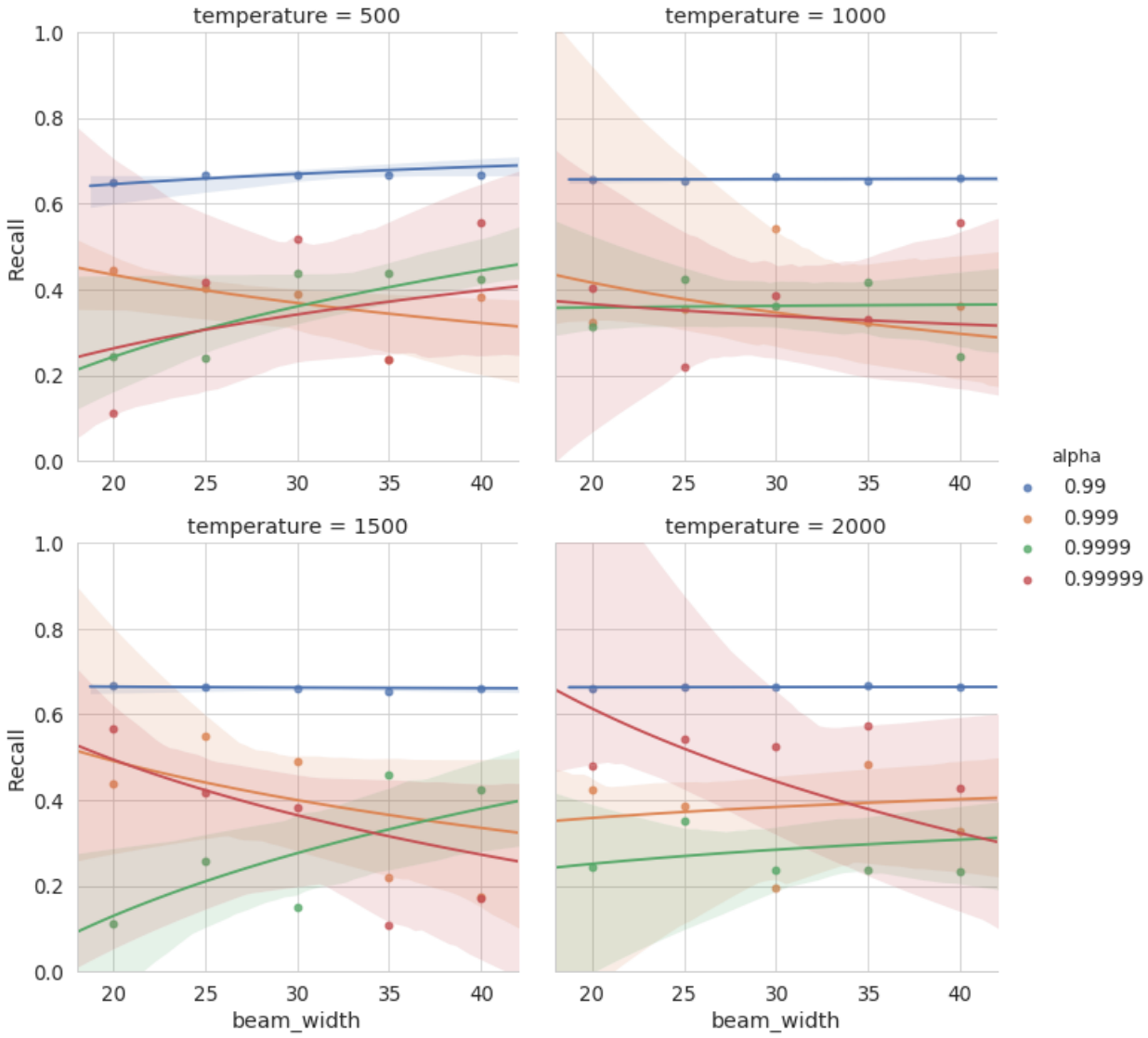

Figure 4.7: Recall of ITRULE Rule with Simulated Annealing with varying parameters for alpha and starting temperature over a range of beam widths

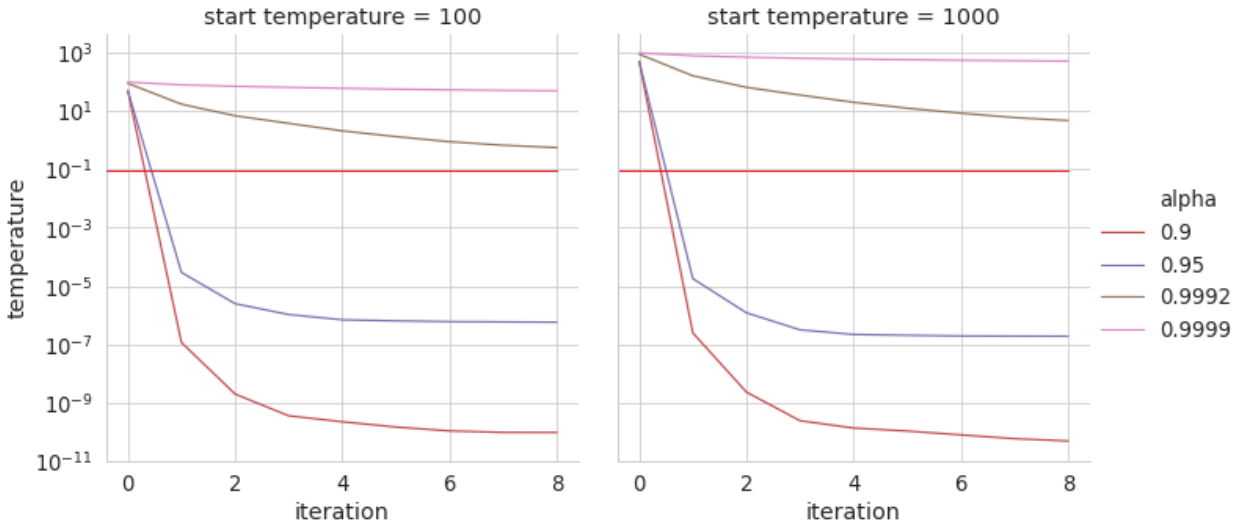

Figure 4.8: Cooling strategy at different starting values and α

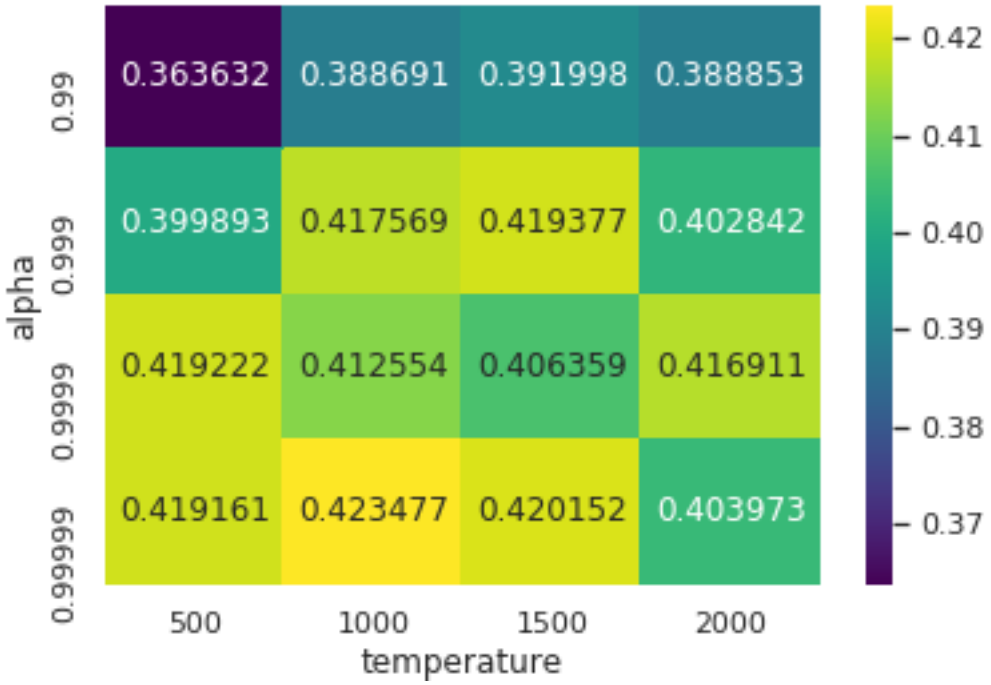

Figure 4.9: Heat map displaying the reported standard deviations in the precision based on the varying in starting temperature and α

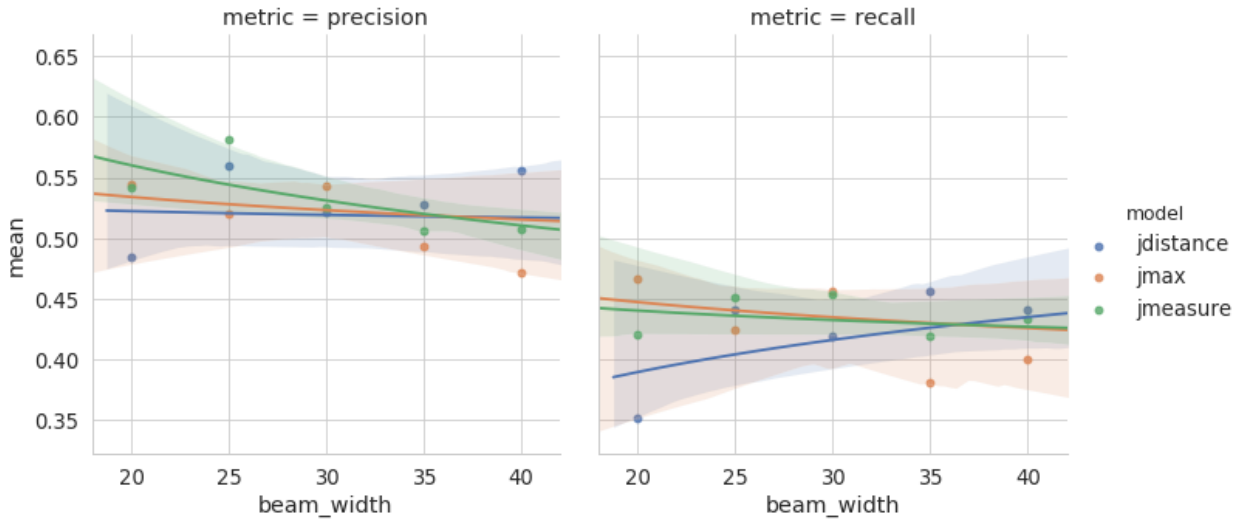

Figure 4.10: The affect of varying the beam width on ITRULE using Simulated Annealing. The acceptance probability is driven by one of JDistance, JMax and J-measure with a starting temperature of 1000 and an α of 0.99

Varying the driving metric does not seem to have had a large effect on the precision or recall as seen in Figure 4.10. This can be verified for all metrics in Table 4.6. This may be because of the order of magnitude of the metrics being similarly small that they do not produce sufficiently large variations in the acceptance probability. It may also be because the acceptance probability is too high in the initial instance that acceptance into the beam is very likely across all three. Finally the tentative accuracies in Figure 4.11 are very high, particularly again for the lower values of *alpha*. This is expected as the longer it takes to cool the closer to the global optimum the solution will be, at the cost of an increased running time[13]. This concludes the parameter tuning of ITRULE Annealing, the next section reintroduces a further version of ITRULE from the literature before an comparative evaluation is made.

4.2.1 ITRULE Positive

In this section another variation of ITRULE available in the literature is briefly described. The authors of [41] produced a different version of ITRULE with the aim of finding correlations between network events. It calculates an altered J-measure using Equation 4.3, this implementation is referred to in this work as *ITRULE Positive*. A benefit of J-measure when

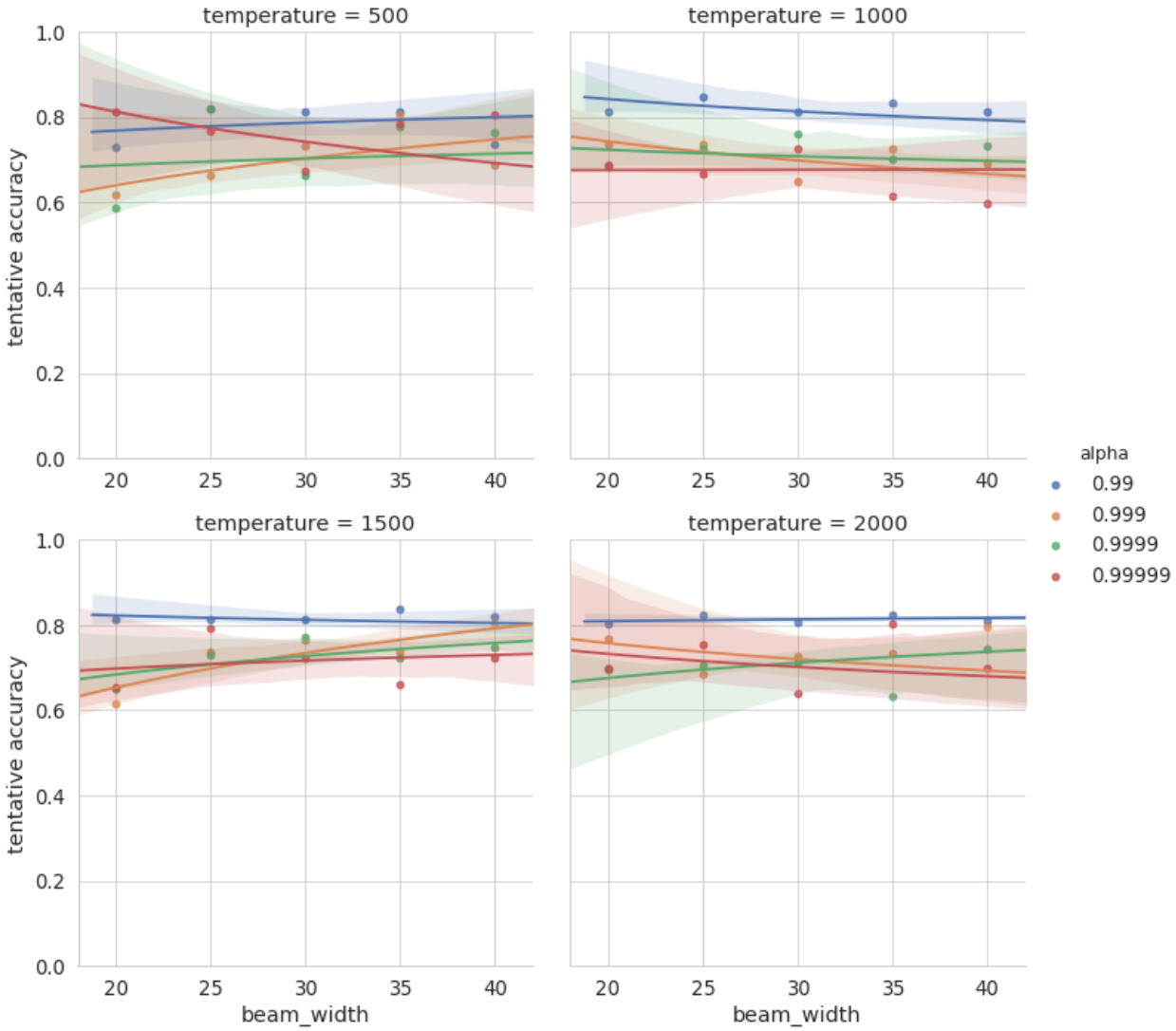

Figure 4.11: Tentative Accuracy for ITRULE Annealing for varying values of α and starting temperature

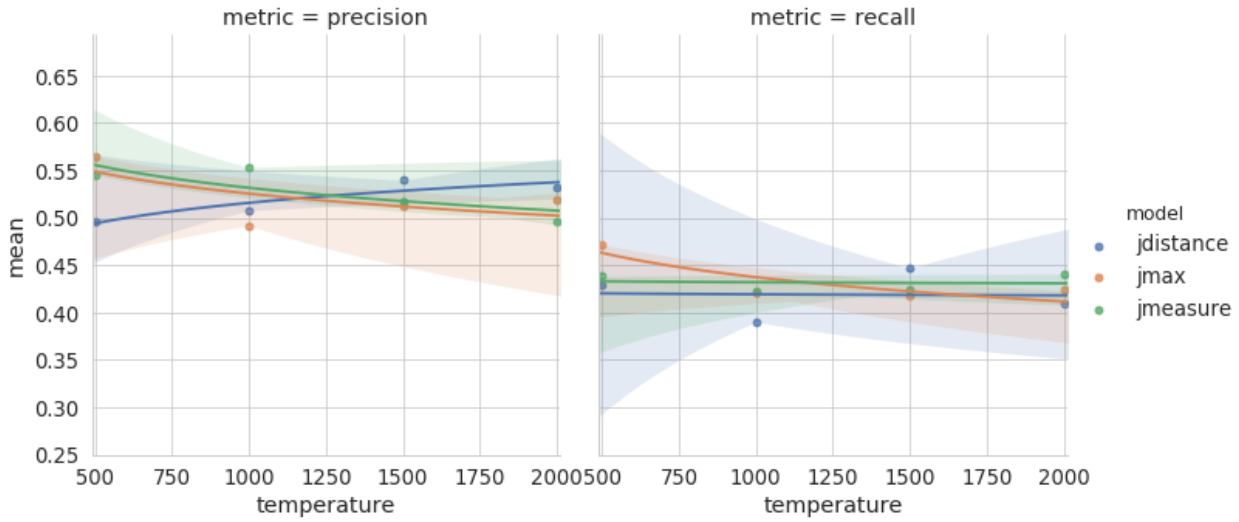

Figure 4.12: The effect of varying the starting temperature on the mean recall and precision in ITRULE Annealing where rule acceptance is driven by J-measure, JDistance and JMax

Table 4.6: Top precisions across all parameters for ITRULE annealing

model	metric	mean	std
Jdistance	abstain_rate	0.699505	0.055724
Jmeasure	abstain_rate	0.680406	0.067036
Jmax	abstain_rate	0.680202	0.064932
Jdistance	accuracy	0.173452	0.115898
Jmeasure	accuracy	0.185314	0.124468
Jmax	accuracy	0.188460	0.121462
Jdistance	confidence	0.492104	0.304272
J-measure	confidence	0.489430	0.316811
Jmax	confidence	0.499106	0.298263
Jmeasure	Jmax	0.506784	0.188706
Jmax	Jmax	0.539343	0.183456
Jdistance	Jmax	0.505548	0.159106
Jmax	Jmeasure	0.096194	0.018269
Jdistance	Jmeasure	0.098417	0.016198
Jmeasure	Jmeasure	0.099563	0.019094
Jmax	precision	0.519609	0.400384
Jdistance	precision	0.520515	0.408271
J-measure	precision	0.524562	0.411919
J-measure	recall	0.436274	0.350953
Jmax	recall	0.440970	0.347173
Jdistance	recall	0.403655	0.327087
Jmax	tentative_accuracy	0.738580	0.124374
Jdistance	tentative accuracy	0.731592	0.111820
J-measure	tentative accuracy	0.736578	0.127325

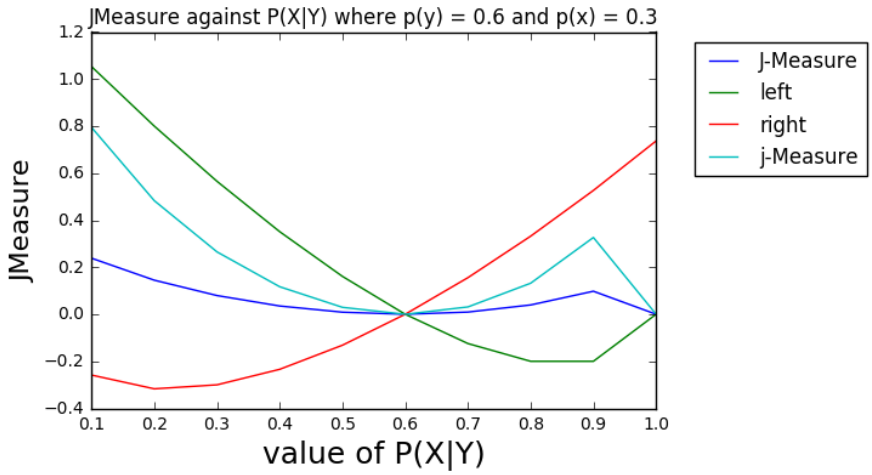

Figure 4.13: J-Measure as calculated by ITRULE and ITRULE Positive under different values of p(Y) and $p(X|Y)$. The left and right plot use a high and low value of p(X) respectively.

used to discover causal rules is that it specifies the direction of the relationship. The version of J-measure used here removes the entropy value of the exclusivity relationship, when an antecedent makes a consequent less likely. Figure 4.13 demonstrates the relationship between the positive and negative parts of the J-measure calculation. It can be seen that the modified equation will, in some cases, produce higher J-measures without the negative weighting of the right hand-side. It also enables a negative J-measure when P(X|Y) becomes likely whilst the J-measure has a lower bound of 0 and there is no uplift for results at the extreme end of the relationship. This will have an effect on the ranking of rules that have a high causal relationship but these rules are expected to be few.

$$j(X : Y = y) = p(x|y).\log(\frac{p(x|y)}{p(x)}) \tag{4.3}$$

4.2.2 Base comparison of ITRULE for alarm classification

There are now four variants of the ITRULE algorithm to test against the BT data set. Two from the literature, the original algorithm from [75] and ITRULE Positive from [41], and two developed to counter the partial rule dominance issue outlined as seen on this data set in [87]. In this section all four are trained and tested with the classification problem used in the previous section, two variants of ITRULE Annealing were included in this test, the

initial variant and where the acceptance probability is controlled by the Jdistance. The values of the starting temperature and α are 1000 and 0.95 respectively. The mean precision and recall of these experiments are displayed in Figure 4.14. There is a large variation in ITRULE Annealing's results compared to the other three methods, Positive and ITRULE perform fairly poorly, the alteration to the J-measure in ITRULE Positive makes a positive classification more likely and as such may increase the algorithms recall. Without the negative weighting a rule can quickly reach it's Jmax value keeping the rules more general. Here, however, the recall is comparable to basic ITRULE and the precision is very low. This is likely due to high scoring negative rules dominating the additional lower scoring positive rules. Low scoring positive rules will increase the number of positive classifications but low scores are often the result of low prior probabilities (see Figure 4.13), making the rule less accurate. ITRULE_PRD, the precursor to ITRULE Annealing, consistently displays the highest precision but the lowest recall, indicating that it is correctly classifying an extreme minority of available target alarms and that it has been over-fit.

Figure 4.15 displays the accuracies and tentative accuracies produced by this experiment. Both variants of the ITRULE Annealing algorithm show less variation here and consistently produce the best tentative accuracy whilst having a very low overall accuracy, indicating that it abstained for more instances than it classified but was largely correct when classifying them. Table 4.7 contains the means for all recorded metrics for each algorithm aggregated across the beam width. It shows, as before, that the difference between the versions of ITRULE Annealing under different acceptance probability methods is very small. The original ITRULE has performed very poorly in these experiments, the low precision and recall are indicative again of local maximums. ITRULE Annealing has overcome this by using the Annealing method to maintain a variety of features in the beam. Figure 4.16 is a Gaussian Kernel Density plot of the feature distribution of the ITRULE and ITRULE Annealing that demonstrates this variety. The feature distribution of ITRULE is presented by a large peak over a small subset of features whilst ITRULE Annealing has a much larger range of features and much smaller peak indicating the features that are repeated within the beam.

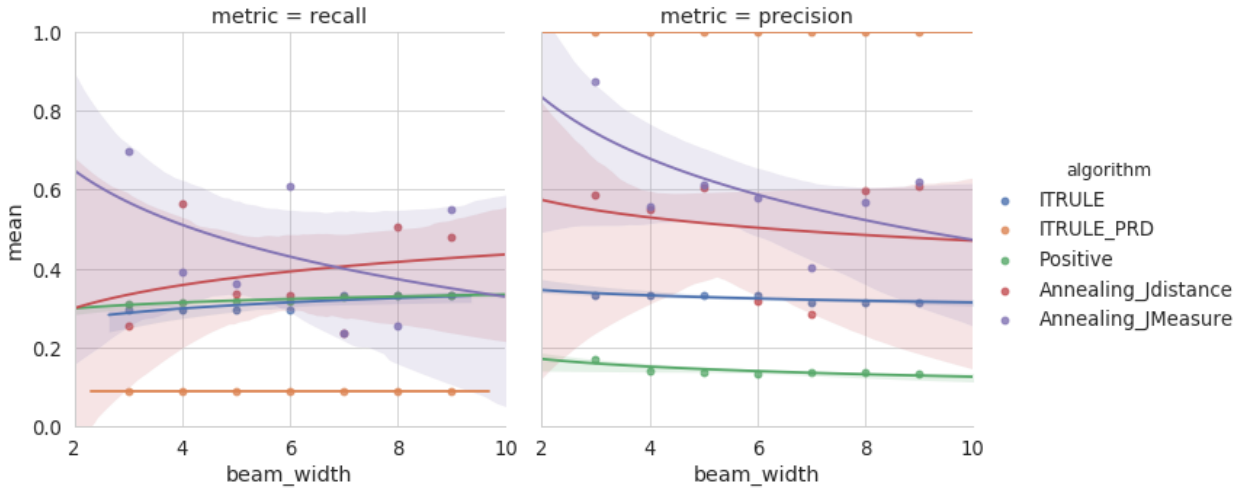

Figure 4.14: Precision and Recall for the variants of ITRULE on the BT data set

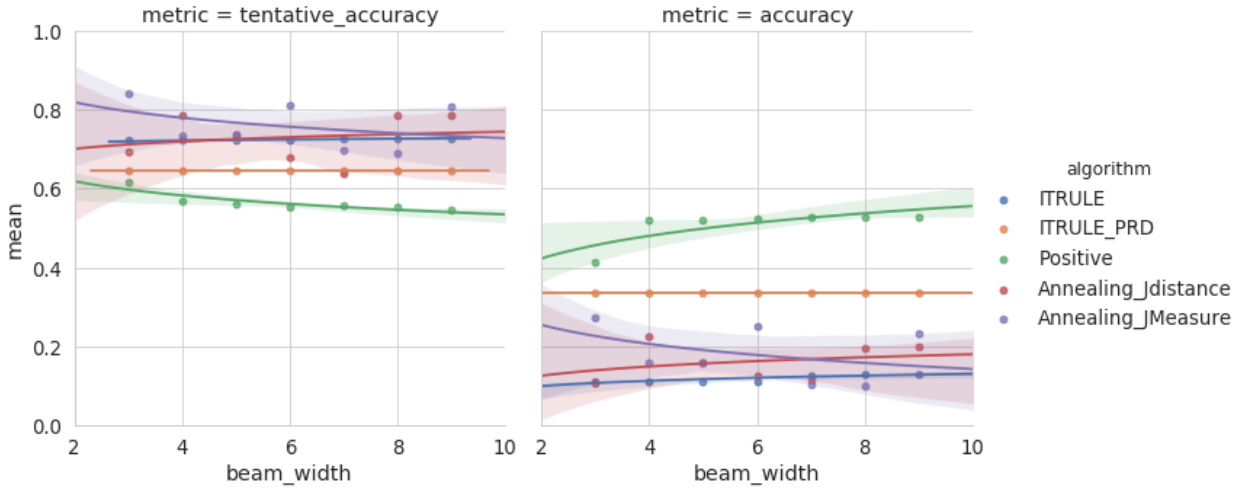

Figure 4.15: Tentative accuracy and accuracy for the variants of ITRULE on the BT data set

Table 4.7: Mean results across all beam widths for ITRULE variants

algorithm	metric	mean	std
Annealing J-measure	abstain rate	0.706917	0.059048
Annealing J-measure	accuracy	0.183247	0.108899
Annealing J-measure	confidence	0.549703	0.206654
Annealing J-measure	Jmax	0.241563	0.117654
Annealing J-measure	J-measure	0.039180	0.013268
Annealing J-measure	precision	0.601787	0.354879
Annealing J-measure	recall	0.443129	0.304336
Annealing J-measure	tentative accuracy	0.760515	0.099950
Annealing Jdistance	abstain rate	0.742089	0.076877
Annealing Jdistance	accuracy	0.161498	0.130882
Annealing Jdistance	confidence	0.479694	0.279691
Annealing Jdistance	Jmax	0.188582	0.127113
Annealing Jdistance	J-measure	0.040124	0.016678
Annealing Jdistance	precision	0.507891	0.420027
Annealing Jdistance	recall	0.387667	0.350474
Annealing Jdistance	tentative accuracy	0.728994	0.132600
ITRULE	abstain rate	0.636898	0.019155
ITRULE	accuracy	0.120263	0.167032
ITRULE	confidence	0.538919	0.323453
ITRULE	Jmax	0.038955	0.032901
ITRULE	J-measure	0.468750	0.003420
ITRULE	precision	0.325818	0.460776
ITRULE	recall	0.312525	0.441977
ITRULE	tentative accuracy	0.724433	0.172497
ITRULE PRD	abstain rate	0.665267	0.215945
ITRULE PRD	accuracy	0.334733	0.215945
ITRULE PRD	confidence	0.604990	0.097803
ITRULE PRD	Jmax	0.017517	0.002819
ITRULE PRD	J-measure	0.010938	0.006228
ITRULE PRD	precision	1.000000	0.000000
ITRULE PRD	recall	0.091269	0.013868
ITRULE PRD	tentative accuracy	0.644167	0.020939
Positive	abstain rate	0.061093	0.051710
Positive	accuracy	0.509031	0.101323
Positive	confidence	0.593118	0.088425
Positive	Jmax	0	0
Positive	J-measure	0	0
Positive	precision	0.142575	0.201632
Positive	recall	0.322206	0.455667
Positive	tentative accuracy	0.564995	0.061830

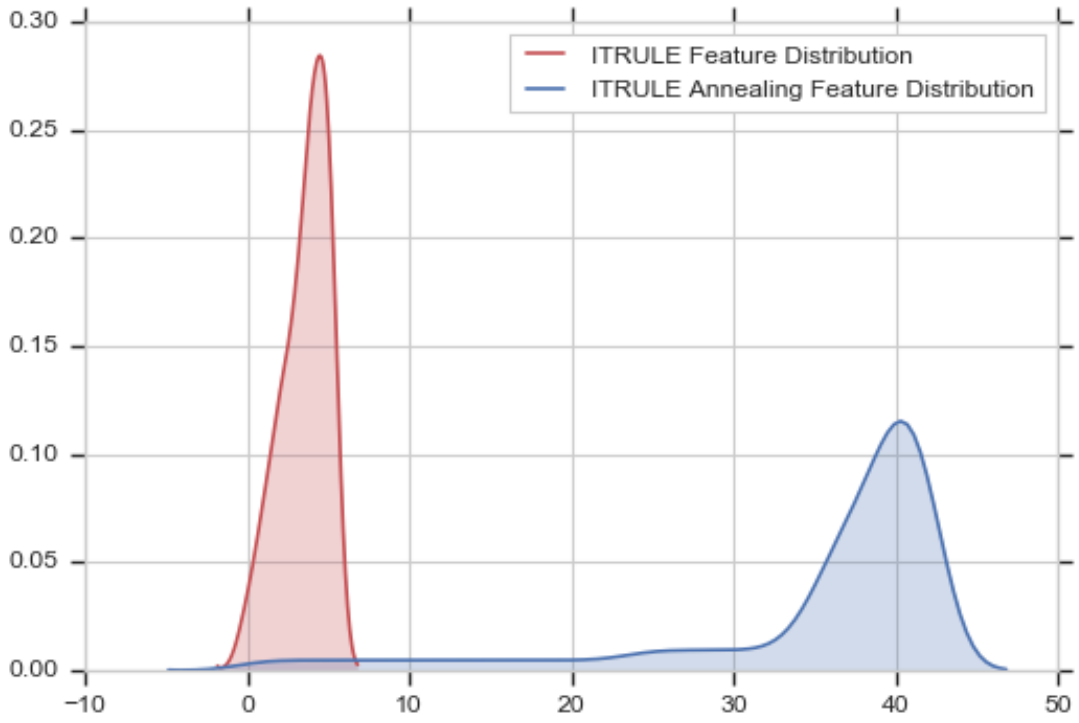

Figure 4.16: KDE Plot of feature distribution across ITRULE and ITRULE annealing

4.3 Expressive Forecasting of Down Events

In the previous a section a number of adaptations to the beam search based classifier ITRULE were described and tested with a classification problem based on the BT alarm data set. These classifiers are able to produce rules that are human readable and contain a variety of features. The next step is to use these classifiers to forecast alarms rather than simply classify them. To best represent the feature space in these rules an adaptation is made to the data set instead, outlined in the following section. This expressive method of prediction was one of the contributions from this project and published in[87]. Experiments are conducted to determine if forecasting these events with the algorithms explored already is possible and finally a method to forecast the arrival time of an event is trialled.

4.3.1 Pre-Event Marking

Figure 4.17 demonstrates Rule Induction across two axis, forecasting on the vertical axis and classification on the horizontal. Each are limited entirely to their own axis and so a forecaster cannot utilise additional features in it's predictions, this is can be a disadvantage if the concept is not entirely contained within one feature. Pre-event Marking is a technique designed to allow an algorithm on the horizontal axis to be trained on a target class that represents the vertical.

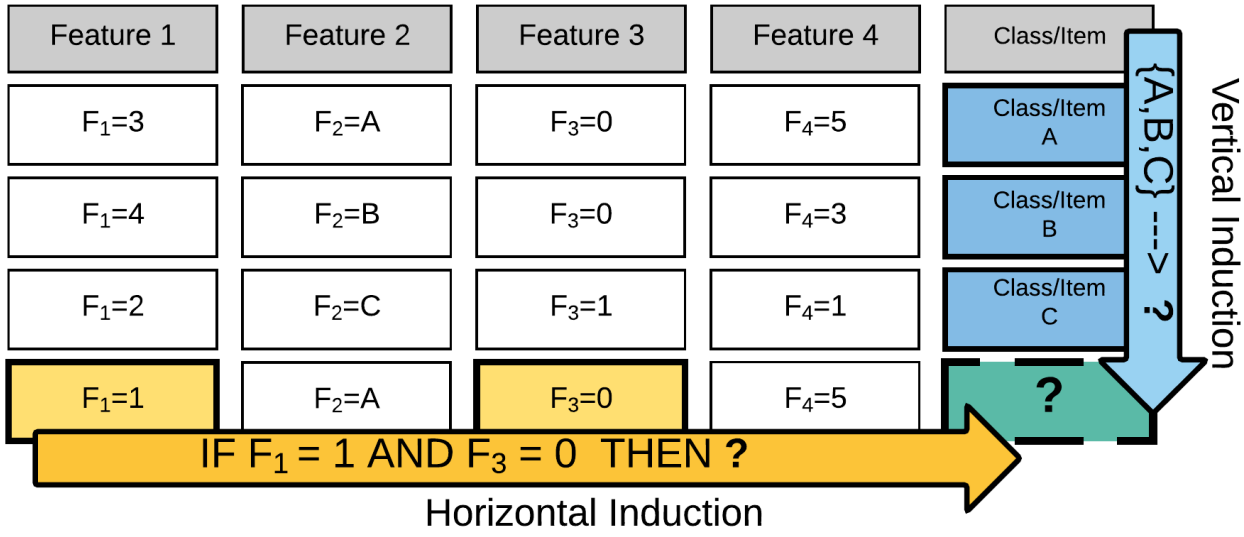

Figure 4.17: Axis of Rule Induction)

This is a very basic transformation of the target class from a descriptor of the instance to an indicator of a future event, in this case still a down event.

A time window of length w is set, terminating at an event of interest, A, and starting at $time_A - w$. Events falling within this time window are marked as a pre-event, retaining their own event type regardless of it's event type. In this way down events can also be marked as a pre-event, which is a logical step as faults often beget additional faults in a complex system. This transformation allows a Rule Induction algorithm to produce rules that forecast an event rather than simply describe the containing event. This approach is simplistic and some methods to refine the transformation are covered in Chapter 5. There are issues with heavily altering the class balance of the data set and the optimal window with which to

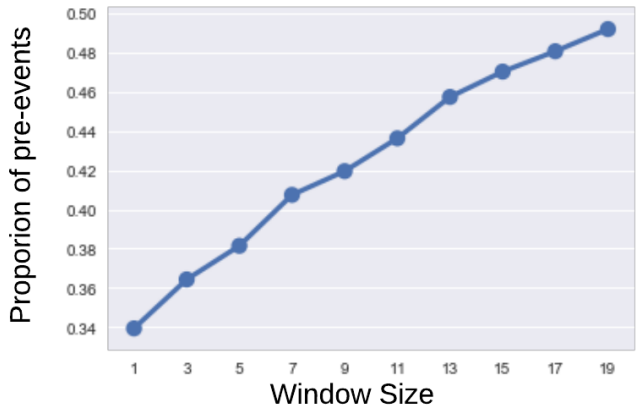

Figure 4.18: The mean proportion of events marked as pre-events against interval size across SVLANs (filtered)

produce these rules must be determined. In this section experiments will be conducted over a range of windows and finally an attempt will be made to forecast the expected time of an event.

Figure 4.18 depicts the proportion of class labels as the pre-event window increases from 1 to 19 minutes in the data set. They show in both a cases a nearly linear relationship, a binary classification problem with a data set using a 19 minute window has an almost 50% split between the classes. A benefit of this is that algorithms that suffer from class imbalance may perform better, however, this transformation has likely introduced a substantial amount of noise into the data set which will be further addressed in Chapter 5. Pre-event marking is more respectful of the original distribution as the time window tends towards 0, though the time window would not too short an interval for any mitigating action to be taken by the user to prevent the predicted event.

4.3.2 Pre-event Prediction

In this section the ITRULE variants are tested using the pre-event prediction transformation. Precision is again the key performance indicator as the goal is to predict whilst limiting false positives. False positives, as outlined in Chapter 2, have a far greater impact on a systems uptake than false negatives. The recall of events will also be recorded as a secondary metric.

A perfect classifier will have a precision and recall of 1, in this case a recall of 1 is not desirable as a number of the events marked as pre-events are likely to be incidentally included, this is expanded upon in Chapter 5. In [87] this method was also combined with variants of the Prism algorithm, another expressive abstaining classifier which performed well, these are included here for comparison. This paper also contributed an alteration to Prism called JPrism which, though not included here, is presented in Appendix A.1. The hyper-parameters for ITRULE and ITRULE Positive are the same as before with a maximum rule length of 4 and a beam width of 45. ITRULE Annealing, having been configured in the previous section, has the same beam width.

Results from the pre-event show a much better precision using the ITRULE Annealing whilst ITRULE performs very poorly. This is likely because of the lack of variation in the ITRULE's rules as ITRULE performs poorly against minority classes [87]. The change made to the J-measure calculation in ITRULE Positive assigns a higher value to rare events and makes it more likely for a rule predicting the minority class to be included in the final beam, because of this the predictions from ITRULE Positive have the highest precision across time windows. Overall the precision of each algorithm has dropped when predicting the pre-events due to the increase in label noise.

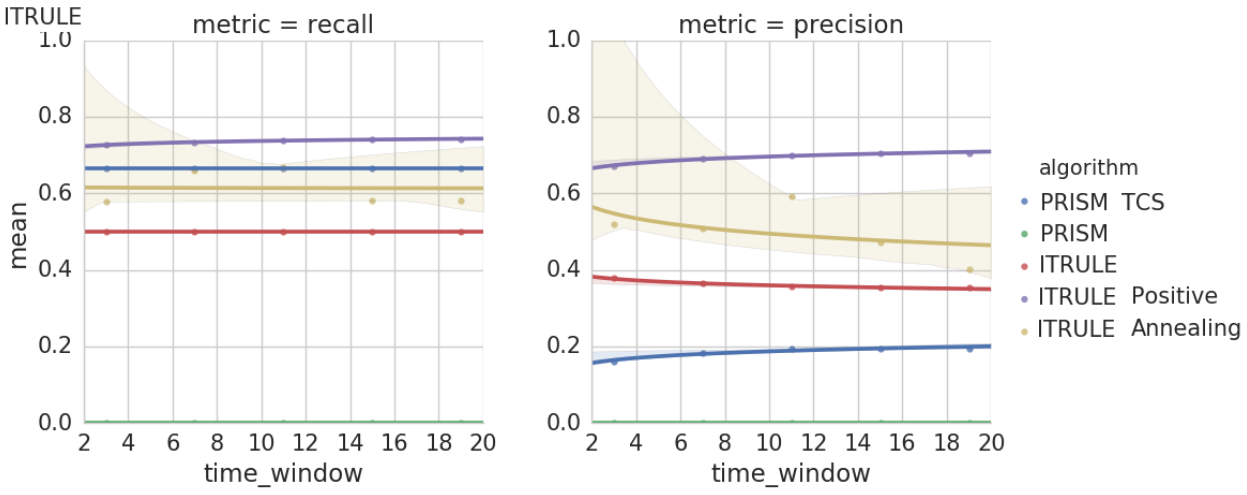

Figure 4.19: Precision and Recall for predicting down events using ITRULE and PRISM

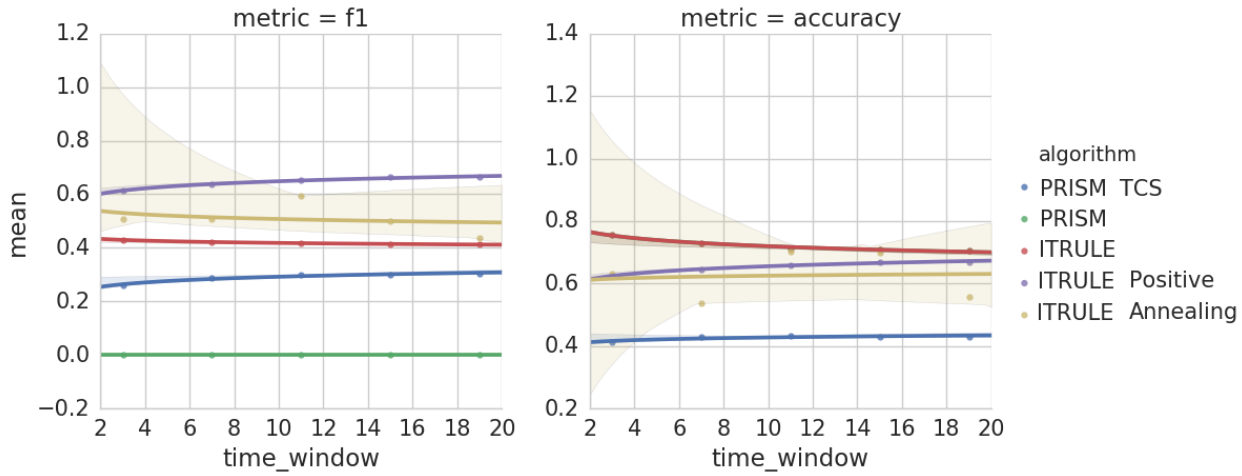

Figure 4.20: Accuracy for predicting down events using ITRULE and Prism

4.3.3 Predicting Time Windows

In the experiments above the time length of the time window has been a variable to be optimised. The previous section has demonstrated that ITRULE Positive and Annealing are the strongest predictors of those tested. The precision and recall for both of these algorithms remains fairly constant (around 0.7 for the positive whilst annealing fluctuates between 0.6 and 0.5). The transformation of events into pre-events converts the lateral classifier into a predictive algorithm, it may also allow the algorithm to express the time window in which an event is expected to take place. This is accomplished by removing the pre-event target class and replacing it with the time window variable.

The process is described in Algorithm 2.

Algorithm 2 Creating class labels for time window prediction

```
n is a time interval, E_T C is the target class of event E, A = ∅
for Every time window T_0, T_1 to T_n do
    for Every E in T_n do
        if E is not in A then
            E_T C = n
            Add E to A
        end if
    end for
end for
```

Figure 4.21 displays the frequency of pre-events across the range of time windows. Pre-

events are assigned to the shorter time window in which they occur and so there is expected to be a large disparity between the early intervals and the later. The difference is of several magnitudes, creating a severe class imbalance. From the overall distribution (see Figure 4.22) it can be seen that the additional events gained by increasing the window size decreases with each minute. As a result the higher time windows are so few as to make prediction difficult.

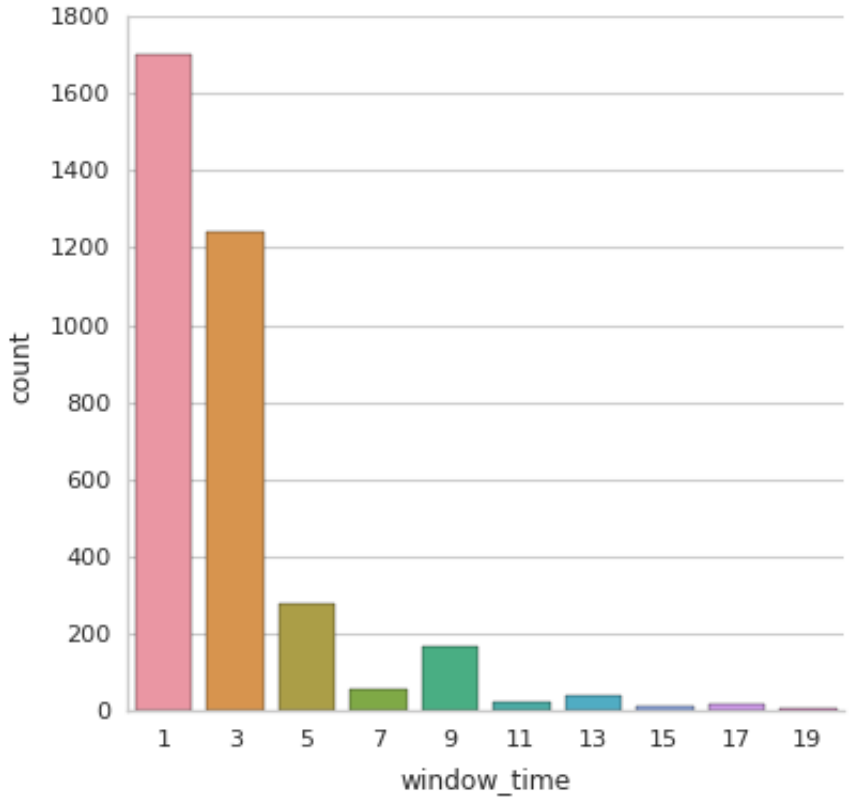

Figure 4.21: Distribution of time window target class

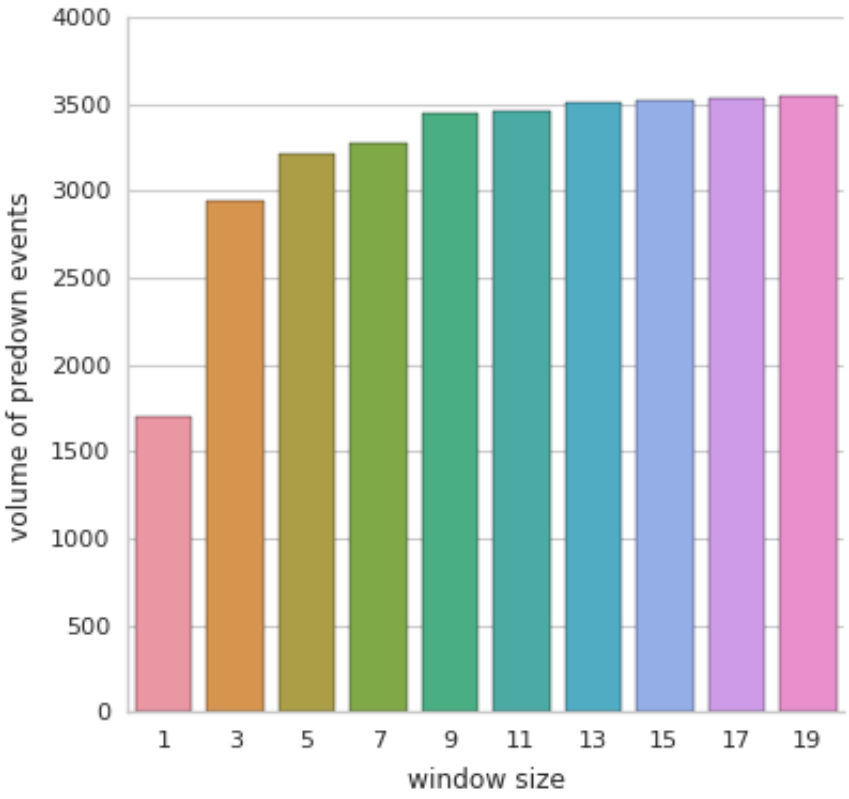

Figure 4.22: Cumulative distribution of event window times

These experiments were run with the initial time window class assignments and again with the class label being withheld up for any time equal to or below 5 minutes. This last change was made for two reasons, it helps address the heavily skewed class distribution by redistributing events that appear in under five minutes to their first appearance after that window. It is also a logical change in terms of the end goal of helping engineers mitigate events; the below 5 minute set of events are likely too soon for an engineer to act on. The results are displayed in Tables 4.23 and 4.24. The performance of an algorithm often drops from a binary classification problem to a multi-label classification so the lowered precisions are to be expected. In their current state none of the algorithms are able to predict a time window with any reliability. ITRULE Positive performs very well when predicting the earliest of the events and has the strongest performance on the subsequent three labels though the precision is too low to be actionable.

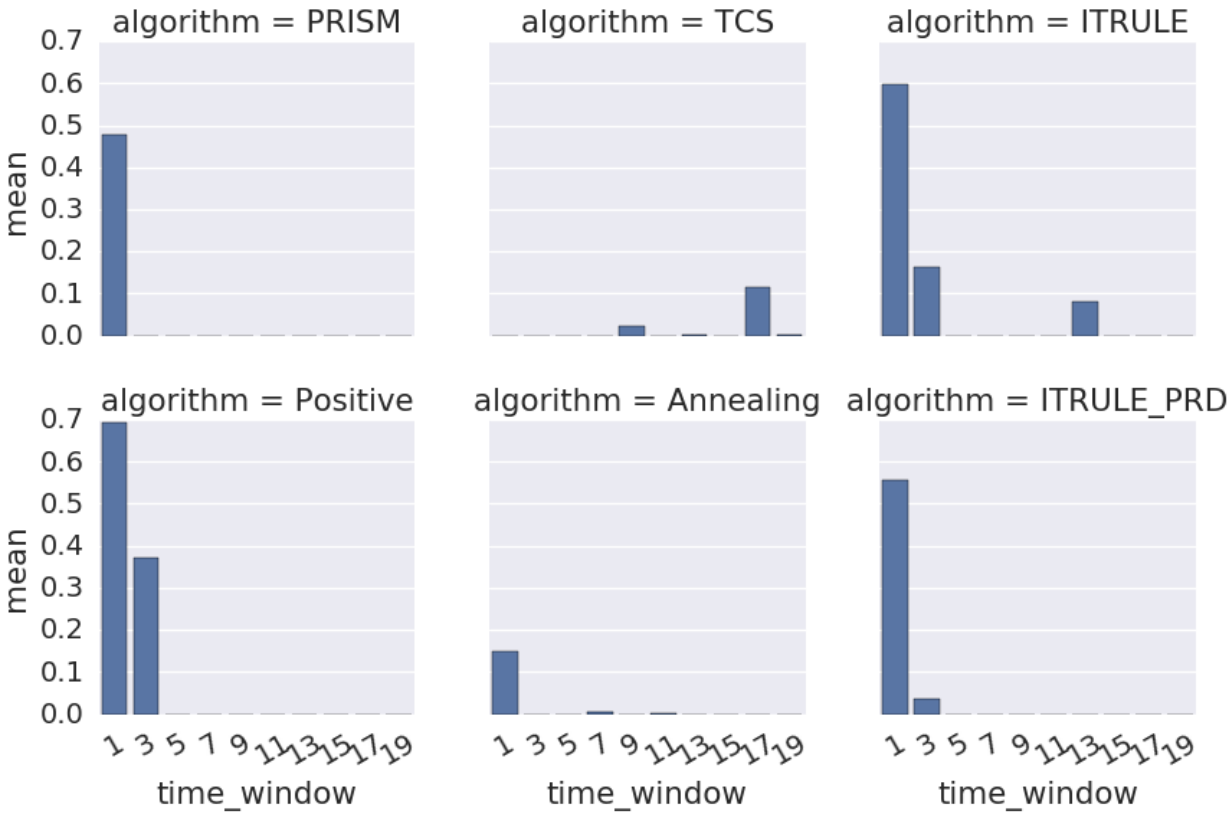

Figure 4.23: Precision of different rule induction algorithms for each time based target class

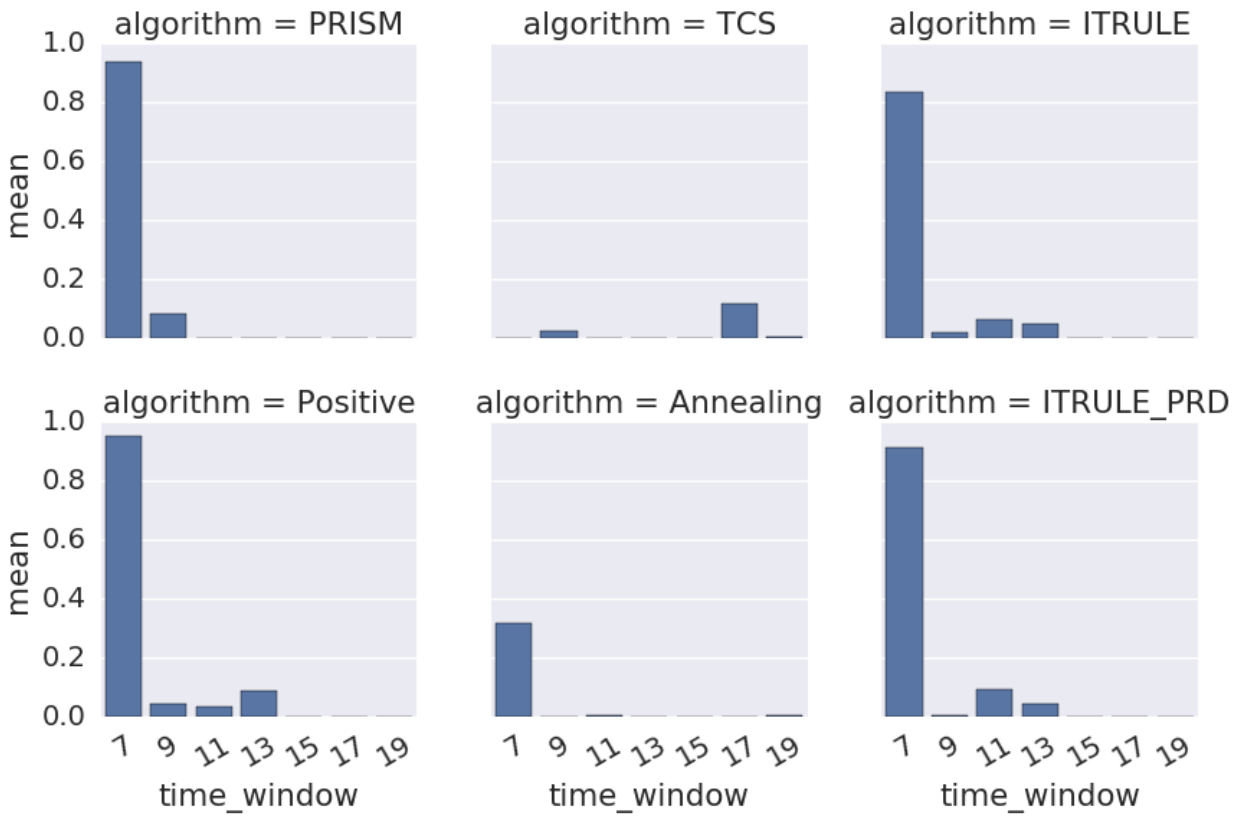

Figure 4.24: Precision of different rule induction algorithms for each time based target class for time above 5 minutes

4.4 Discussion

The purpose of this chapter was to establish if a rule induction algorithm was able to forecast down events in the BT alarm data. To this end a number of rule induction algorithms were trialled, both existing algorithms from the literature and incremental forms. The algorithms were first trialled on the alarm data as a classification problem to predict alarms that have been marked by engineers as problematic. The results are very encouraging with all variants of ITRULE achieving a high tentative accuracy though only ITRULE Annealing coupled this with a high precision, suggesting that the other variants of ITRULE predict a negative event in the majority of cases.

The next phase was to trial the algorithms on a transformed version of the data set to establish if these algorithms can forecast an alarm. These tests also include Prism And PrismTCS. Prism base algorithms are another family of abstaining classifier with PrismTCS

being a special adaptation to predict the minority class which may perform well here. A benchmarking of the Prism algorithms against the ITRULE on stock data set is available in appendixA.1).

As the algorithms are by nature descriptive, successfully forecasting the down events would go along way to meeting this works primary objective. The transformation of the data set is quite crude and is likely to introduce alot of noise. This is offset with the choice of an abstaining classifier for prediction as it is not forced exhaustively learn all the concepts within the data. The primary evaluation metric for these problems are the precision of the classifications with tentative accuracy and recall as secondary targets. With this in mind the results from the pre-event tests were very good with ITRULE Positive and ITRULE Annealing producing workable precisions. Recall was also high for these algorithms along with PrismTCS. The other algorithms trialled performed less well though the tentative accuracy of the base ITRULE was very high this is likely because it was predicting the majority class in nearly all instances. Rules that predict the majority class are important as they help prevent false positives during training but in actuality a prediction of the majority class is of no value.

The algorithms were also trained to predict the expected time window of a fault. This again involved a transformation of the target class and has likely introduced more noise into the class labels, compounding the noise introduced in the previous steps. This combined with the expected loss of predictive power on a multi-label classification accounts for the across the board poor performance in predicting these events. The expected arrival time of an event would be a very valuable feature for the system as an engineer will have a time frame to work to. A further attempt at this will be made in the following chapter.

One of this chapters contributions to knowledge, the adapted ITRULE Annealing has performed very well in the classification and pre-event forecasting tasks. ITRULE PRD, introduced in [87], was a brute force approach to introduce variety into the rule that was not very successful. Annealing introduces a more statistical based set of controls around the acceptance of rules that was rewarded in the recall and precision of the scores.

Chapter 5

Two Stage Classification of PreEvents

In Chapter 4 a number of rule induction options were developed and explored that forecasted down events with human readable rules to a degree of accuracy, though there are some improvements that can be made to the system. The process of labelling events as pre-events allows a rule induction classifier to become a rule induction forecaster but it introduces a lot of noise into the artificial target class. This is demonstrated in Figure 5.1. In the window preceding a down event there will be contributing events and incidental events. The contributing events should be labelled as pre-down events and it's features used to predict the oncoming down event. Incidental events do not have a causal relationship to the down event though it happens to be present in the SVLAN within the time window.

These incidental events contribute noise to the system which alters the underlying concept that the model is being trained on and may limit it's predictive capability. When labelling the classes there is no test in place to check if a duplicate event has already been seen and what label it was given, resulting in events with matching LHS but dissimilar RHS, creating a class noise. This chapter details some approaches to refine the pre-event marking process. Several approaches are explored and their effect on the precision of the classifications are monitored. The strongest performing method is then refined and tasked with predicting time windows to meet the final goal of this work.

Section 5.1 focusses on several methods that were employed to remove the noise introduced by pre-event marking. These are separated into three broad approaches: statistical, frequent

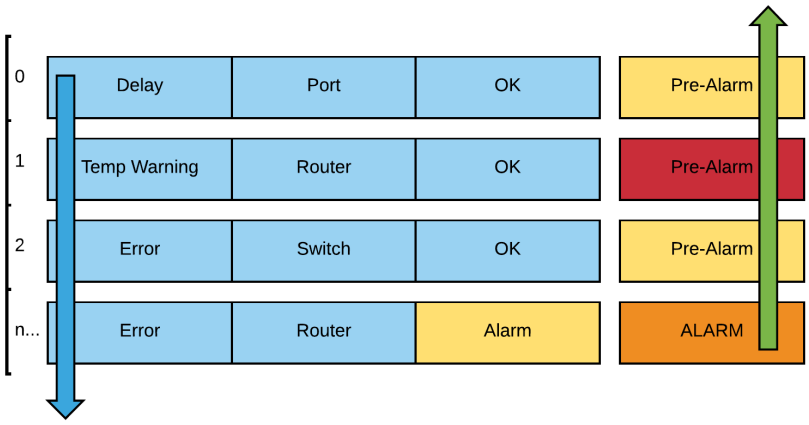

Figure 5.1: Pre-event marking marks all events within the preceding window as a pre-event. If an event does not contribute to the eventual down event it will still be marked and create label noise. A method is needed to remove this noise.

pattern based and model based. The result of these experiments is a high precision Two Stage modelling based approach to predicting pre-events that is further refined in Section 5.2.

In this chapter all experiments are performed using alarms from the BT data set. These alarms are for the most part taken from the 4th cluster as defined in Chapter 3 and the target class used is the pre-event label from the previous chapter.

5.1 Methods to reduce pre-event label noise

Class label noise can create instances that are indistinguishable besides their class label resulting in a *clash* that may hamper the performance of a system[77]. In these cases it can be assumed that the information necessary to differentiate these instances has not been captured in the data set. As these instances are not possible to separate by their feature values alone, a decision must be made based on a wider context within the model. A clash can be resolved in an number of ways including discarding the instance or relabelling it with a designated target class[7, 53]. This problem can be avoided if a classification is not required at the modelling level and a class likelihood value is output instead, allowing the user to set their own intuitive threshold. This is possible with ITRULE by allowing an instance to check

itself against all the rules and taking an average classification, this approach was explored with ITRULE Annealing but an instance matching multiple rules was a very rare occurrence and so there was very little impact made.

Assigning the majority class of the BT data set would in most cases assign the instance a *no event* label which would discard information about a possible oncoming critical alarm. This section explores some modifications to the pre-event labelling to address clashes in the data and improve pre-event prediction. These approaches fall into the following categories:

- Majoratative Based - retaining only the pre-events that have no clashes or who present the majority class in the clash
- Frequent Pattern Mining - labelling only events that match pre-generated patterns generated from Sequence Mining
- Model based Filtering - using modelling to assign pre-events

All of the above techniques are likely to reduce the number of the pre-events and so will result in a greater class imbalance, changing the distribution seen in Chapter 4. All the experiments in this section are performed with three fold cross-validation. Each technique is applied to a training set and the unfiltered data used to test the models explored in Chapter 4.

5.1.1 Majoratative Based

Population based filtering comprises of two different approaches to resolving a clash when they occur referred to here as *majoratative* and *unique* filtering. The first is to only mark pre-events when there is no clash, this is the equivalent of labelling using the population majority class as a default. The second is to only mark pre-events when the pre-event is the majority class within the clash which is a widely used approach to combat clashes [10, 70]. The two present a trade off between the amount of noise retained in the data set and the level of class imbalance being introduced.

Of the three types of approaches these methods represent the simplest to apply. The effect each technique has on the population of the various clusters is visible in Figure 5.2. As would be expected, the population of pre-events is reduced under the unique sampling approach. The number of distinct rows in the pre-event set has also been reduced but by a lesser extent.

Figures 5.3 and 5.4 display the affect on the precision, recall and accuracy of each filtering technique against the original baseline. There has been a general drop in performance using these filtering methods. The removal of so many instances of the target class was likely to lead to a greater class imbalance problem and the accompanying over-fitting problem. This trade-off is visible through the different responses the algorithms have to the unique and majoratative methods. The majoratative method impacts the class imbalance less and the basic ITRULE, which is more susceptible to over-fitting, has been impacted by this the least. The opposite response was seen from ITRULE Positive and ITRULE Annealing, which have seen large drops in precision from both filtering methods. As events are removed from the target class previously learnt concepts are lost, a level of noise may be beneficial to these learners.

There are additional methods that could have been explored here listed in [23], including Bayesian methods of assigning confidence to each resolution. These were not explored as the methods so far have only decreased the precision of the algorithms with only few exceptions. The next set of approaches in Subsection 5.1.2 are those based in Frequent Pattern Mining.

5.1.2 Alarm Pattern Based Filtering

Rather than mark pre-events using a time window, events could be marked using a template approach. The process would first require the generation of a number of patterns that can be tuned by setting a minimum support and confidence.

Frequent Pattern Mining is a field of machine learning focussed on finding relationships between sets of items, both ordered and unordered. Unlike the majoratative approaches they require an exploration phase to locate the patterns, often taking many passes of the data.

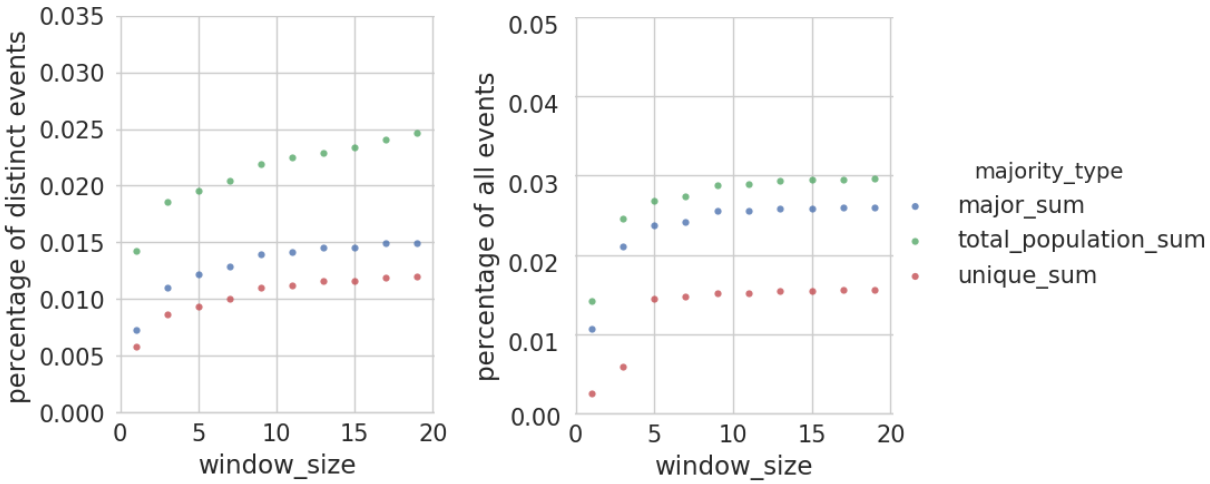

Figure 5.2: Population Size of Pre-events after simple majority and unique clash resolution by total events

The author of [47] used Apriori to generate a set of association rules in a telecommunication data set of a similar duration and cardinality as the BT data set. They then used a subset of alarm features, known as alarm predicates, from alarms captured in these rules to create more expressive rules. This process is dependent upon generating enough high confidence association rules in the initial stage. Any generated patterns will need to be evaluated to determine their suitability as a template. This section details the results of experiments to find these initial patterns for use in the pre-event transformation. Two approaches are considered from the literature, Association Rule based approaches [2, 48] and a codebook approach[49].

Codebook based filtering approach

In [49] a codebook approach is used as a pre-processing method ahead of alarm correlation based on the method presented in [89]. The process has two major steps: creating a codebook or transition matrix and then using this to decode the event stream.

To build the correlation matrix a subset of events are identified as exception or problem events, leaving other events as symptomatic though allowing for problems themselves to be symptoms. For each problem P we generate a correlation vector P_i either with the observed probabilities of an event following an alarm or, deterministically, a boolean value indicating

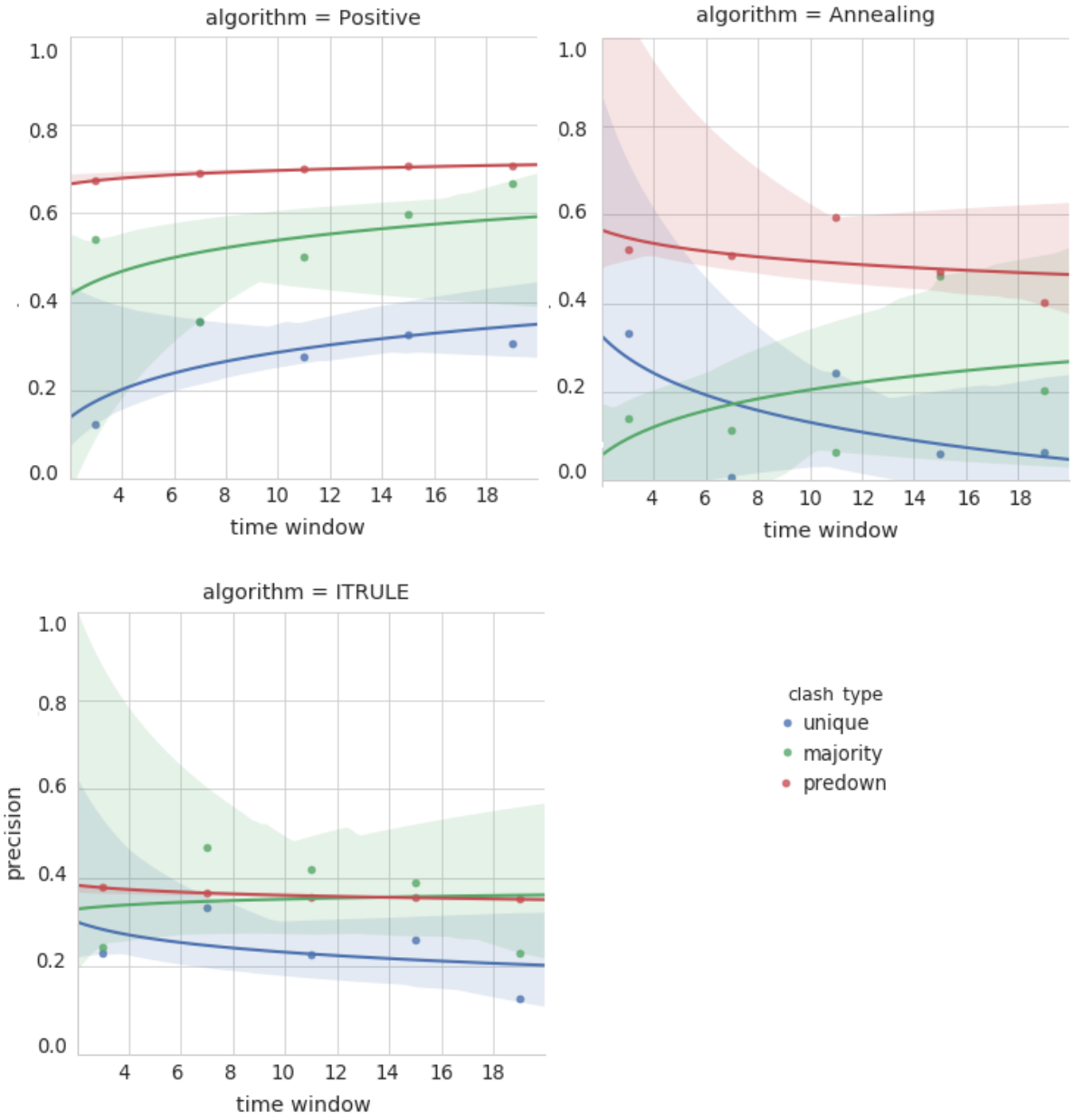

Figure 5.3: The effect on the precision of pre-event prediction using unique and majoratative filtering over a range of time windows.

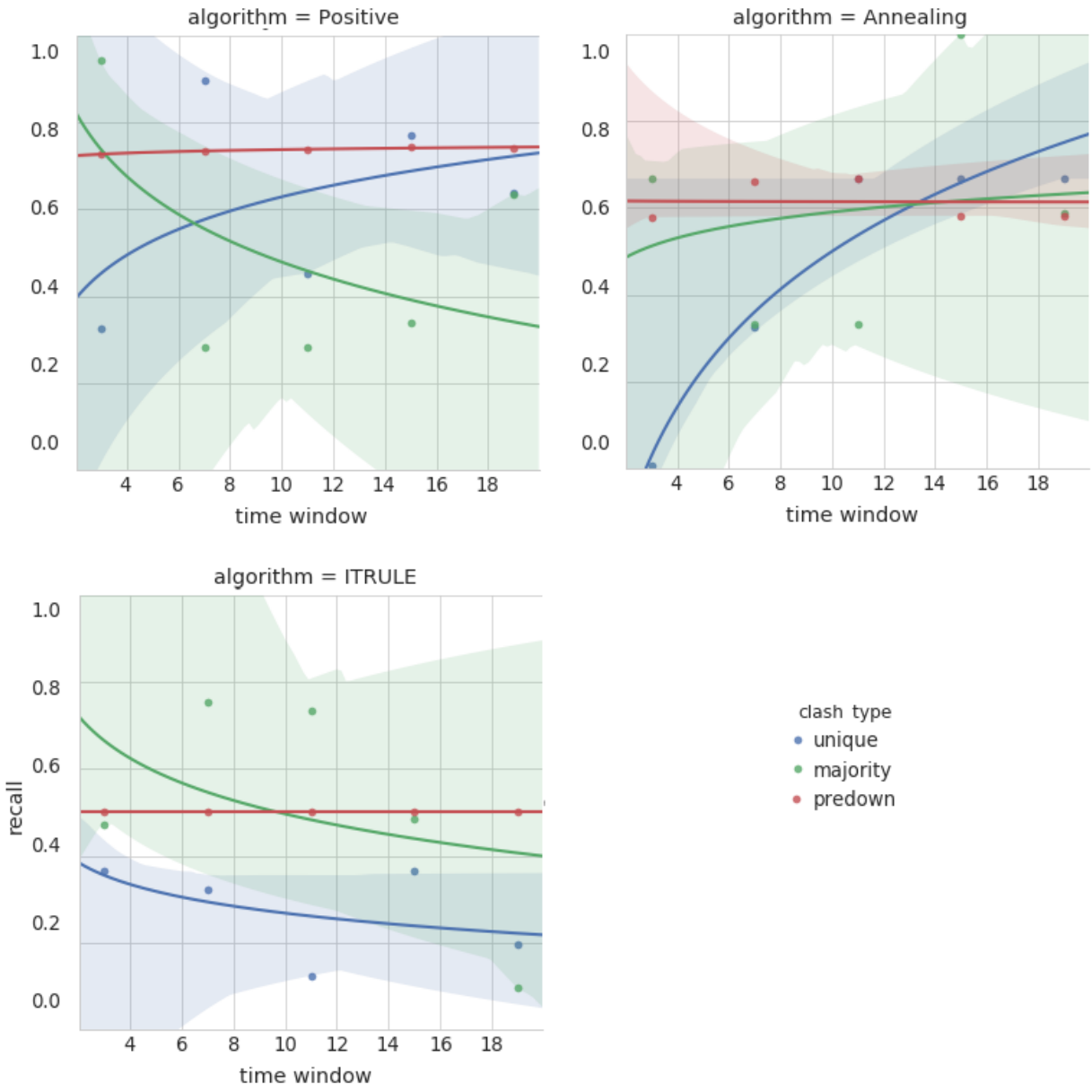

Figure 5.4: The effect on the recall of pre-event prediction using unique and majoratative filtering over a range of time windows.

if it was ever present in the window. This vector is referred to in [49] as the code of the problem. These form a maximal codebook CM.

The alarms first undergo a filtering process to reduce cyclic events into one event, much like the removal of repeating events applied to the BT data in Chapter 3. The codes for each problem are likely to be overly long and in need of refining to remove noisy events. A new codebook C is produced by systematically selecting rows from CM and adding *spurious* events.

Both a deterministic and probabilistic codebook are presented in [49] with the reduction being performed via a threshold value T, calculated as one half of the minimal Hamming distance, or log-likelihood for the probabilistic approach, between pairs of codes. This distance must be experimentally set to balance the resistance to noise with the variation in patterns.

To achieve a codebook with the desired radius the alarms are ranked by their total volume in ascending order. Alarms are then added to an empty codebook and the radius recalculated sequentially. This continues until the threshold is exceeded. The algorithm is laid out in Algorithm 3. The function radius is the Hamming distance [28] for the deterministic approach or the log likelihood for the probabilistic codes, see Equations 5.1 and 5.2 respectively for how these are calculated.

Algorithm 3 Algorithm to form codebooks

```
1: P_n = Problem vector n
2: Codebook C = ∅
3: Maximal Codebook CM = (P_0 P_1 ... P_n)
4: radius(CM) = 0
5: distance threshold = T
6: for alarm i in CM do
7:    if Var(i) > 0 then
8:       C_i = i
9:       if radius(C) > T then
10:         return CM
11:      end if
12:   end if
13: end for
```

$$dist(P_i, P_i) = \sum_{k=0}^{n-1} [P_{i,k} \neq P_{j,k}] \tag{5.1}$$

$$dist(P_i, P_i) = \sum_{k=0}^{n-1} \log \frac{P_{i,k}}{P_{j,k}} \tag{5.2}$$

The codebook is readily adapted to the filtering problem presented here with the proviso that other problem events can be identified with which to filter against. These additional events are necessary to remove event patterns that occur with sufficient support as to be considered noise. In the implementation outlined below the search is for events that lead to problems, as opposed to tracing symptoms back to their source, in actuality only the terminology changes.

In order to filter out noise or *spurious* events a set of additional problem events are needed. To select these events the top n event types ranked by their aggregate estimated impact are taken. The estimated customer impact is a manually populated field that has been excluded from the modelling process as it is added some time after event generation. It is set by domain experts and is an indication of the level of disruption an event causes to customers. The log distribution of these values are plotted in Figure 5.5 and the aliased values are listed in Table 5.1. There are two drops in magnitude for the impact values, denoted on Figure 5.5, that can be used as natural boundaries between problem events and the other alarms (those events that fall to the right and left of the line respectively). The minimum Hamming distance decreases exponentially as additional problem vectors are included, see Figure 5.6. The cardinality of the set of problem events will be inversely proportional to the amount of noise in the codebook and so the left line was chosen as the boundary. These other events must also occur with a frequency similar to the target events otherwise the other vectors in the codebook will be too sparse to set any reasonable level for the filter.

The maximum average Hamming distance for a vector occurs when each of it's problem alarms are unique. This is achievable by removing any row with a clash but this will leave a very small subset of alarms.

Both deterministic and probabilistic filtering codebooks were generated and applied to the event data set. The precision and recall for both approaches are available in Figures 5.7 and 5.8. The filtering has a large impact on the recall of the system, as the number

Table 5.1: Top 7 events ranked by customer impact as assigned by engineers

Rank	Event	Aggregate Minimum Impact (# Customers)
1	A	3587036.0
2	B	18514.0
3	C	6173.0
4	D	3494.0
5	E	2652.0
6	F	2309.0
7	G	745.0

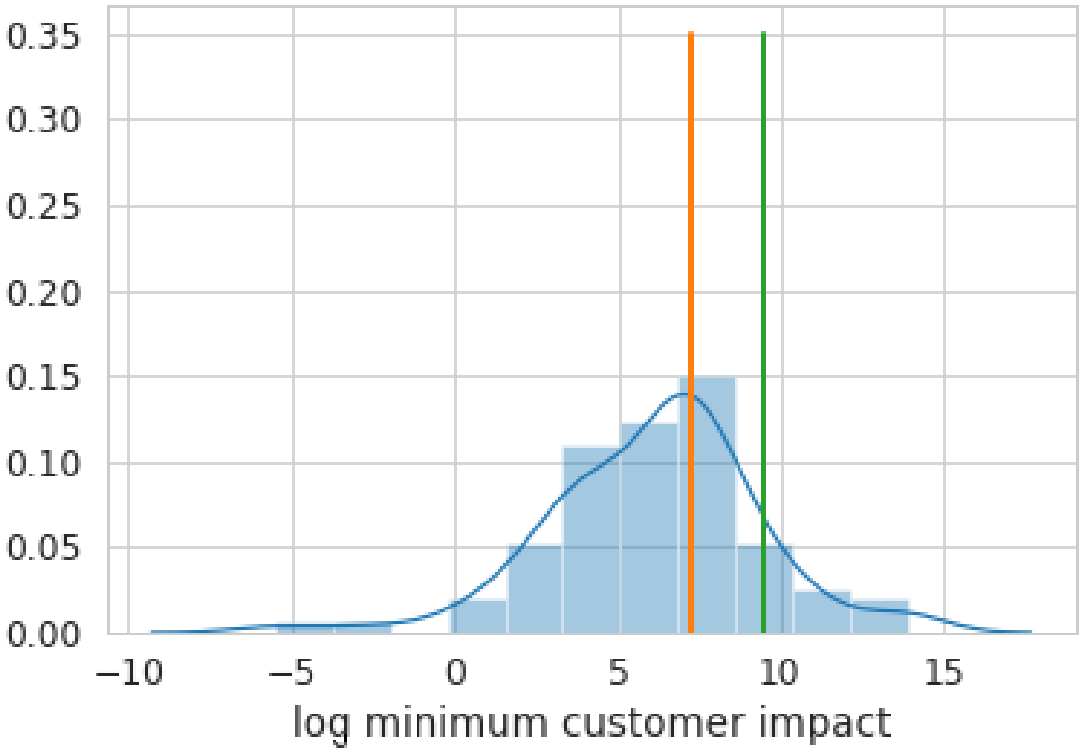

Figure 5.5: Log distribution of the feature 'customer impact' to determine secondary target classes for the codebook approach with a cut off line denoting problem events to the right

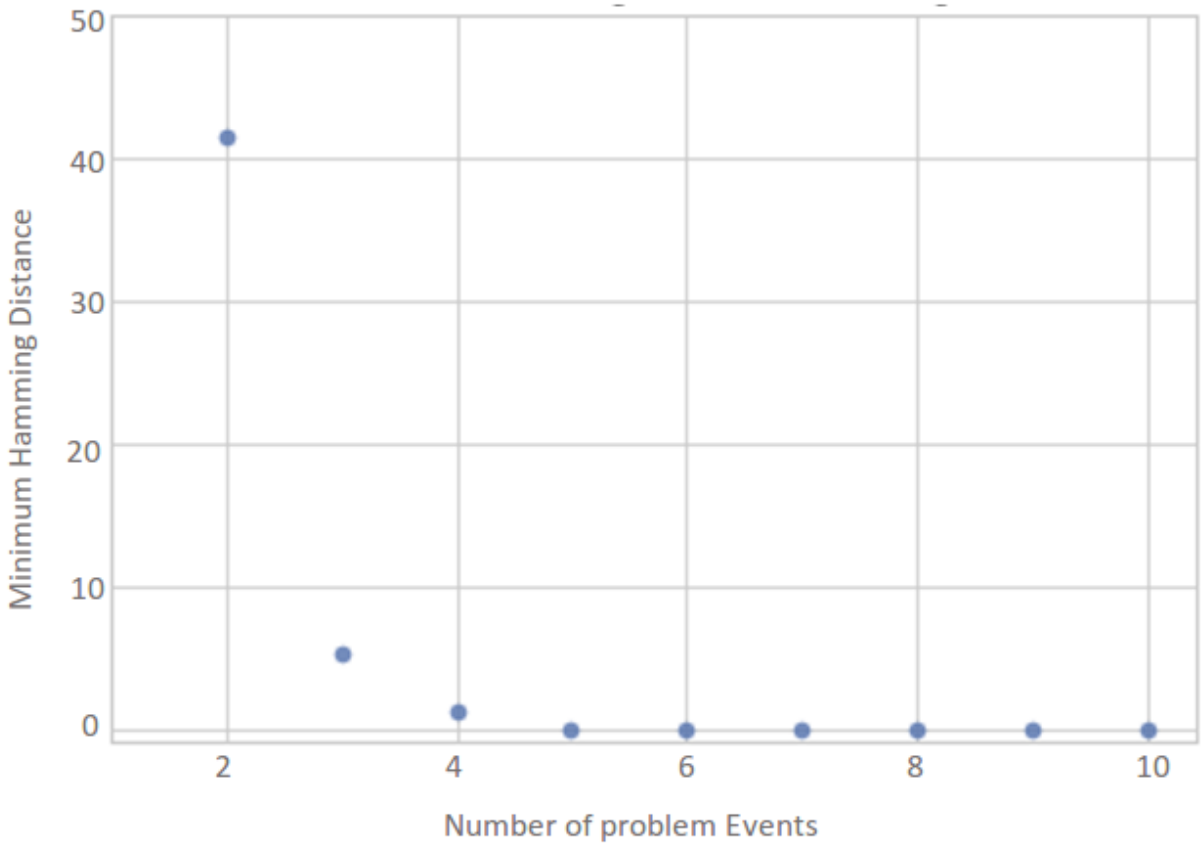

Figure 5.6: The affect on the minimum Hamming Distance between problem vectors when against number of problem vectors

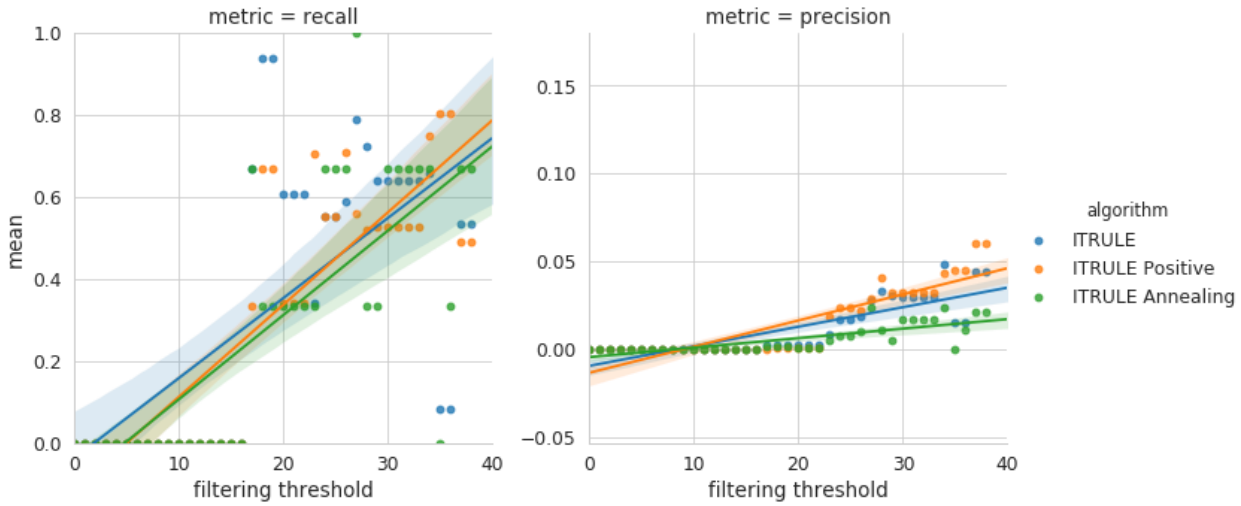

Figure 5.7: Precision and recall from the deterministic filtering approach

of possible positive instances decreases it follows that the recall increases. The precision has been lowered across the board with both the probabilistic and deterministic approaches yielding a precision of no more than 0.2. This approach has made the system less effective. It is possible that this approach could be improved. Some options include: using different problem vectors, altering the algorithm for creating a codebook, or using a different distance measure. The problem alarms were selected for their similar volume and disruptiveness based on a manually entered value. An alternative would be to select the problem alarms randomly or based on alternative feature values. Other distance measures are available such as Levenshtein distance[59]. Hamming distance was chosen as it features in the original paper and is specific to boolean vectors. The reported precisions have dropped significantly when compared to the previous filtering method and the precisions achieved before filtering was applied. The shortfall in precision is large enough to suggest that this method is not worth exploring further.

Association Rule Mining

Association Rule Mining (ARM) is a large research area within Machine Learning. It focusses on extracting recurring patterns from collections of items. Ordering restrictions can range from weak in the case of Frequent Pattern Mining, or strong in the case of Sequence Mining,

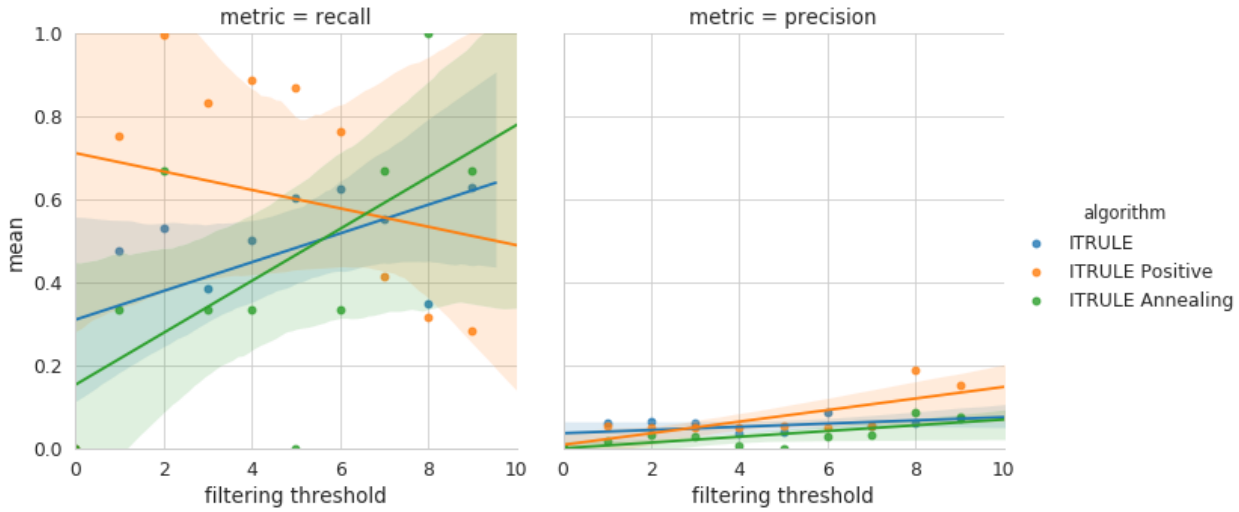

Figure 5.8: Precision and recall from the probabilistic filtering approach

with some algorithms operating under the assumption of partial ordering[86, 30]. One of the earliest algorithms designed for frequent pattern mining was the Apriori algorithm designed for detecting small, high confidence item sets. A later development was the FPGrowth algorithm[79], which uses a depth first approach to generate longer patterns with a lower confidence. These longer patterns are more suited to this task as the transaction sets will take the form of long sequences of events. The implementation of the ARM algorithms used in this section were from the Python package Fim[6] initially with a minimum support of 10.

The event data first needs to be compartmentalised into transactions as seen in[41]. To do this the event series was divided using a threshold applied to the inter-event arrival times. The thresholds were set at each 10^{th} percentile from a sample of SVLANs to give a board coverage.

The distributions of the transactions under each threshold was examined in order to the find the optimal value. The number of items in a set decreases with the value of the threshold. If the transactions are homogeneous then the number of distinct event types will also decrease. Tables 5.2 and 5.3 demonstrate that the transactions sets produced are predominantly singularities with very high support. Alarms with very high support, as established when looking at the codebook approach, are likely to be noise events.

Table 5.4 shows the output of the FPGrowth algorithm with a lower support boundary

Table 5.2: The average lengths of the transaction sets based on inter event arrival times (time boundaries)

percentile	boundary	avg length	avg set length	max length	max set length
10	60.0	1.049146	1.001207	44	4
20	240.0	1.085810	1.004728	104	8
30	327.0	1.137938	1.006343	113	8
40	614.0	1.164693	1.009908	121	12
50	1271.0	1.210368	1.016034	121	15
60	3009.0	1.287583	1.028528	171	20
70	5126.0	1.389394	1.047441	188	25
80	10822.0	1.564482	1.078456	299	26
90	21726.5	1.797264	1.116214	483	30

Table 5.3: Mean length of transactions and the average number of unique event types against threshold (measured in seconds)

	avg confidence	avg length	avg support	boundaries
1	0.0	1.0	28293.00	181.0
2	0.0	1.0	28095.50	298.0
3	0.0	1.0	27637.25	689.0
4	0.0	1.0	25507.00	3377.0

of 10 when trained over the transaction sets. Only one pair of transactions occurred with any frequency and each boundary condition produced the same 4 sets of rules. The results are identical with those produced by the Apriori algorithm .

Table 5.4: Mean length and support of frequent patterns generated through FPGrowth threshold (measured in seconds)

	avg confidence	avg length	avg support	boundaries
1	0.0	1.0	28293.00	181.0
2	0.0	1.0	28095.50	298.0
3	0.0	1.0	27637.25	689.0
4	0.0	1.0	25507.00	3377.0

The alarm data is sparse and unevenly spread, a behaviour describable by the term *bursty*. The probability of a network alarm in a window is relatively small but rises sharply given a network alarm in the preceding time window. The authors of [51] proposed an interval transformation for bursty data. The transformation reduces the number of repeated events by replacing each burst with a set of the events along with the burst's start and end time.

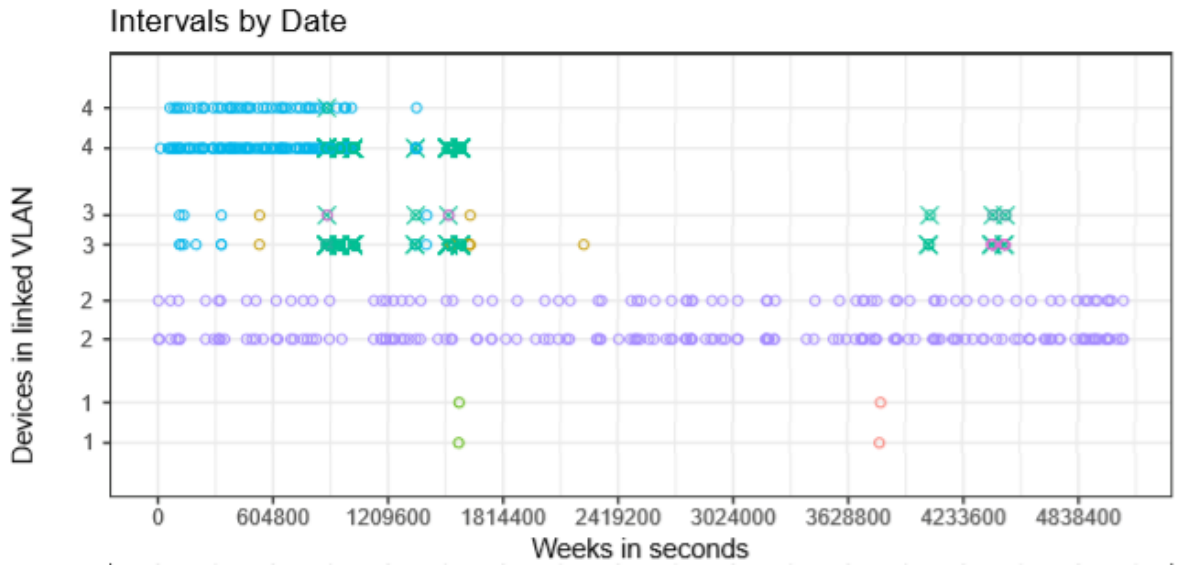

Figure 5.9: Examples of bursty, congested and sparse event streams by devices in a share SVLAN, each series is displayed with filtering and without

These events can then be treated as new transaction sets or, by using the start and end times, as complex events to be merged into new transaction sets using the approach above.

Figure 5.10 depicts a set of alarms over a short window before and after the transformation. The following is description of the process followed in this digram to transform the bursty events into complex events. From left to right across the time window, as event A overlaps with events B and C these are merged into two different complex events. The contents of each event is defined by the active events at each events time of termination. Event C is momentary, i.e., it starts and terminates at the same time with a duration of 0. Events A and B are ongoing at the point of C's closure leading to the creation of complex event ABC with an interval of A_{t0} to t. Likewise when A terminates the event AB is created from the same opening point, A_{t0}, to the updated time t. Event D is also momentary and terminates at the same time as event B, creating event ABD whilst the second occurrence of event A is open, presumably to be closed outside of the window depicted. In this way the five events are reduced to four complex events. The algorithm from [51] is laid out in Algorithm 4, the effect on the alarm data from the BT data set is displayed in a plot of events sampled from three devices on the same SVLAN in 5.9.

Table 5.5 contains the results of the same sample data using the burst transformation.

Algorithm 4 The algorithm transcribed from [51] to convert bursty sequences into a complex form

```
transaction = ∅
result = ∅
adding_phase = TRUE
for opening To and closing times Tc of all events in sequence S do
    items = S(To)
    if items ≠ ∅ then
        trans = trans ∪ items
        adding_phase = TRUE
    end if
    items = S(Tc)
    if items ≠ ∅ then
        if adding_phase = TRUE then
            result = result ∪ trans
            adding_phase = FALSE
        end if
        trans = trans - items
    end if
end for
```

events as self contained transaction sets. As the transformation is dependent on overlapping events there is no need to set a boundary threshold to compartmentalise the data. The maximum length of a rule is still 2 but the proportion of length 2 rules is higher, there are also a great number of patterns with sufficient support to produce rules. The length of these rules are still too short to be useful in serving as a filter to resolve pre-event clashes.

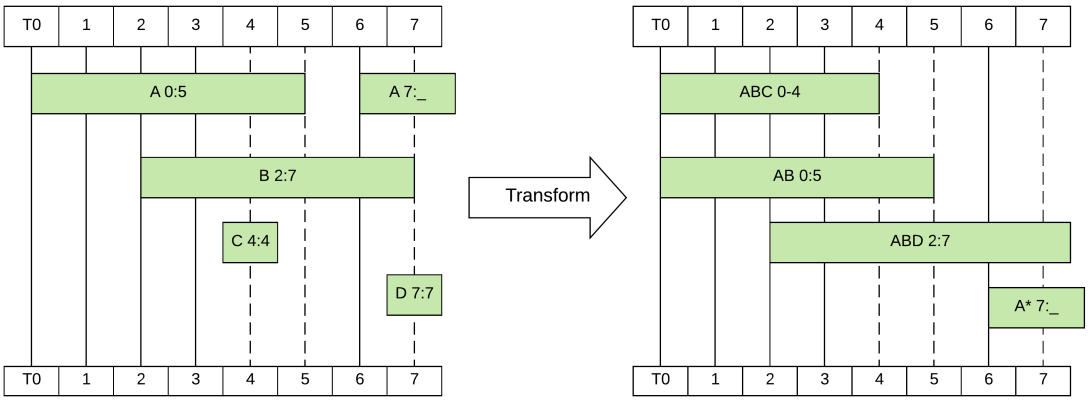

Figure 5.10: Transformation of bursty event data to burst format. Multiple smaller events are absorbed into longer complex events with little information loss

Table 5.5: Mean length and support of frequent patterns by frequent pattern algorithms on the burst transformed data

	algorithm	avg_confidence	avg_length	avg_support
1	Apriori	0.308694	1.533333	31472.933333
2	FPGrowth	0.308694	1.533333	31472.933333
3	Eclat	0.308694	1.533333	31472.933333

Using the same process as before, the data set is split into transactions based on the inter-event arrival times, this time using the complex burst representation. From Tables 5.6 and 5.7 it can be seen that the new transaction sets share the same issues as the original. Singularities and repeated events make the bulk of the transaction sets and no sets with a confidence or support above the set threshold have been produced. An improvement to the results can be made if a minimum transaction set of 2 is imposed on the data set as seen in Tables 5.8 and 5.9 but there is no improvement in average confidence.

Table 5.6: The average lengths of the bursty event based transaction sets based on inter event arrival times (time boundaries)

avg length	avg set length	boundary	max set	max set length
1.049146	1.001207	60.0	4	44
1.085810	1.004728	240.0	8	104
1.137938	1.006343	327.0	8	113
1.164693	1.009908	614.0	12	121
1.210368	1.016034	1271.0	15	121
1.287583	1.028528	3009.0	20	171
1.389394	1.047441	5126.0	25	188
1.564482	1.078456	10822.0	26	299
1.797264	1.116214	21726.5	30	483

Table 5.7: Confidence and support of frequent sets produced by FPGrowth using the bursty event based transaction sets

avg confidence	avg length	avg support	boundaries
0.0	0.0	0.0	60.0
0.0	0.0	0.0	240.0
0.0	0.0	0.0	327.0
0.0	0.0	0.0	614.0
0.0	0.0	0.0	1271.0
0.0	0.0	0.0	3009.0
0.0	0.0	0.0	5126.0
0.0	0.0	0.0	10822.0
0.0	0.0	0.0	21726.5

Table 5.8: Confidence and support of frequent item sets produced by FPGrowth using the bursty event based transaction sets where transactions of cardinality one or less are removed

avg confidence	avg length	avg support	boundaries
0.0	0.0	0.0	60.0
0.0	0.0	0.0	240.0
0.0	1.0	716.0	327.0
0.0	1.0	735.0	614.0
0.0	0.0	0.0	1271.0
0.0	0.0	0.0	3009.0
0.0	1.0	962.0	5126.0
0.0	1.0	1360.0	10822.0
0.0	1.0	1061.5	21726.5

Table 5.9: Confidence and support of the sets produced by FPGrowth using the bursty event based transaction sets where transactions of cardinality one or less are removed using Aprioi, FPGrowth and Eclat

algorithm	avg confidence	avg length	avg support
apriori	0.0	1.0	735.0
fpgrowth	0.0	1.0	735.0
eclat	0.0	1.0	735.0

5.1.3 Model Base Filtering

The third approach to noise removal is a model based approach. Some models have an inbuilt resilience to label noise. Studies of class label noise in [23, 62] describe some black

box methods with such properties. In [62] a number of algorithms were tested on data sets with noise introduced to the target class. These data sets were notably smaller than the BT data set with a less severe class imbalance. The Naive Bayes proved the most resilient to noise overall whilst the SVM produced the largest variation in results across niche cases. As this data contains an unknown amount of class noise in both the test and training set both were investigated at, these same properties of the data set make a perfect classification impossible. The SVM is of particular interest as it has adaptations for class imbalance that can be explored later, however, it can perform poorly on noisy data as it relies on specific instances to form the decision boundary. Naive Bayes in contrast uses prior probabilities from the wider training set and should be less impacted by noise. Table 5.10 details the precision, recall and accuracy of both these models when run on the BT data. The SVM in this case has been trained using a Radial Basis Function as the kernel and the data is from cluster 4. The implementation of both these models are from the package Scikit-Learn[64].

The Naive Bayes has not performed as well as could be expected from the literature, this can be attributed to the data set violating the two assumptions the model makes. Naive Bayes assumes that each feature is independent and that all the features are present. Neither of these can be guaranteed as features have both been deterministically and heuristically pruned in Chapter 3 and alarms are likely to contain related features due to their specificity of location, time and even manufacturer. The SVM has performed very well in these tests and is explored further in the following section.

5.2 Two Stage Classification of Pre-events with Support Vector Machines

An SVM was not considered in the early stages of this work as it is a black box system and as such violates the key requirement of producing human readable rules. This is to give engineers an insight into how predictions were produced and the underlying causes of a fault. It also requires all the features to be numerical, as the BT data's predominant feature type

Table 5.10: Precision, recall and accuracy for Naive Bayes and SVM Classifiers

model	accuracy	precision	recall	time window
GaussianNB	0.570164	0.237988	0.994371	1
GaussianNB	0.655120	0.407149	0.995736	3
GaussianNB	0.727434	0.488985	0.989341	5
GaussianNB	0.736283	0.501949	0.980019	7
GaussianNB	0.712010	0.494595	0.984753	9
GaussianNB	0.418458	0.326776	0.993750	11
GaussianNB	0.426802	0.332249	0.992933	13
GaussianNB	0.421492	0.330887	0.992077	15
GaussianNB	0.422250	0.332061	0.991236	17
GaussianNB	0.428066	0.334419	0.986888	19
SVM	0.872314	0.759259	0.076923	1
SVM	0.863717	0.785408	0.585288	3
SVM	0.865487	0.780899	0.673450	5
SVM	0.864981	0.790123	0.669838	7
SVM	0.869785	0.828228	0.678924	9
SVM	0.870038	0.830065	0.680357	11
SVM	0.867509	0.824094	0.682862	13
SVM	0.870544	0.826360	0.695423	15
SVM	0.869279	0.825678	0.693252	17
SVM	0.869532	0.826403	0.694930	19

is categorical this will require transforming the data set. Approaches exist to extract human readable rules from SVMs as covered in Chapter 2. One of these techniques is to use the SVM as a filtering black box model to transform the class labels ahead of Rule Induction, the authors of SVM_DT[4] specifically use a decision tree as the second stage.

Transforming categorical attributes to numerical attributes is most simply done using hot-point encoding, or creating dummy vectors. This process converts each distinct feature value into a new boolean feature indicating it's presence or not for that observation. This technique has a number of disadvantages, foremost is the large increase in dimensionality which increases training time and can impact the performance of a model. It is likely that a large number of the additional features created through this process will add no value to the data, an additional phase of feature selection could be employed to lower the training time of the SVM.

Figure 5.12 demonstrates the work-flow of this Two Stage system. The SVM is trained on the data containing clashes. A second set of training data is then passed through the SVM where pre-events are relabelled based on the SVM's classification. These are then passed to the rule induction algorithm for training. As a final stage a whole data set is passed through the system so the performance of the Rule Induction can be evaluated. SVM filtering decreases the number of pre-events in the data set by a marginal amount, see Figure 5.11, increasing the class imbalance by a small margin than previous methods.

In a live system the set up is substantially simplified as after training both algorithms the SVM is no longer required and the rule induction algorithm can classify events without filtering, though the SVM would be required again to retrain the algorithm if, through changes to the network or consumer behaviour, the learnt concepts become invalid.

Figure 5.13 shows the precisions of all the rule induction algorithms across the time windows as the output of the Two Stage system. The basic CART[70] decision tree outperforms the other methods of rule induction while ITRULE PRD has been heavily penalised by the change in class labels. It has been previously established that this method is prone to over fitting on the majority class as it has done here. ITRULE Annealing and ITRULE Positive

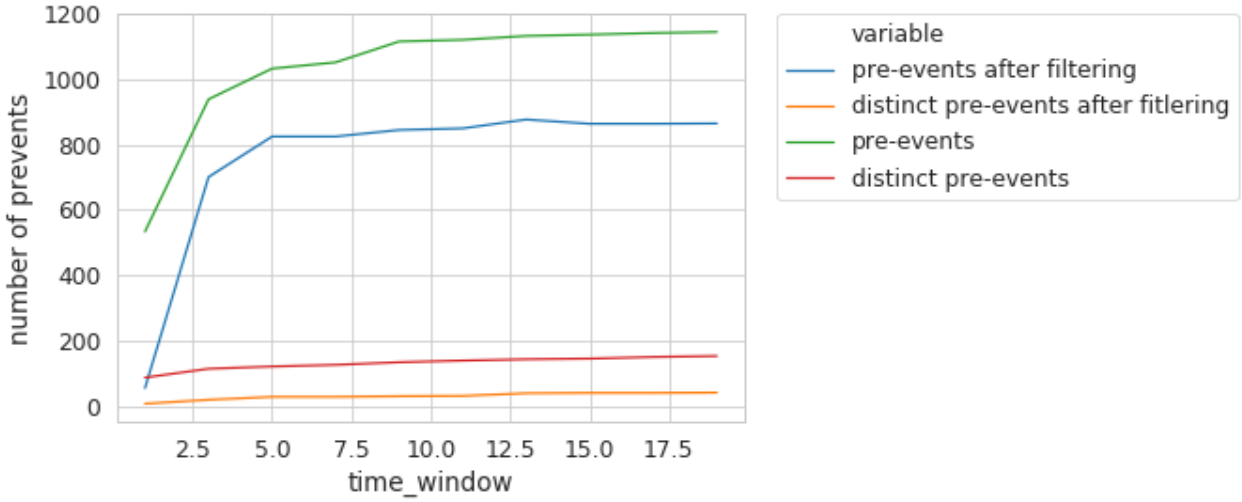

Figure 5.11: The effect of the SVM filter on the pre-event population across all time windows

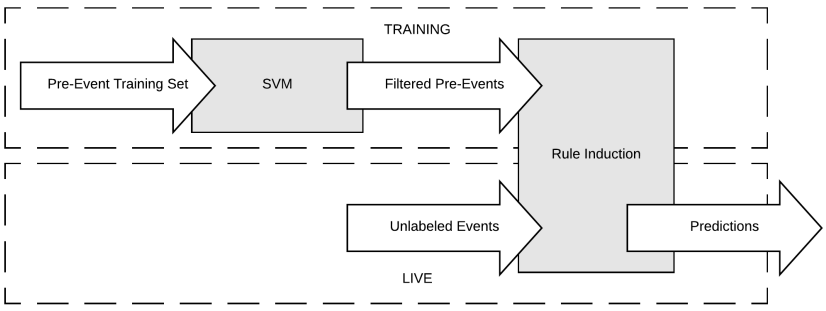

Figure 5.12: SVM resolves clashes training the rule induction model. The model is then able to predict pre-events with a greater precision.

both report precisions lower than before filtering, the rules have been over generalised by the simplification of the problem and the average rule length tends towards 1. This results in a large number of false positives. The Two Stage SVM filtering approach is by far most effective of those explored. In the following section the SVM filtering algorithm will be explored further.

To optimise the system the same experiments were run with different kernels. Figures 5.13 and 5.14 depict the precision and recall for each kernel across different time windows. A summary of the performance of each is provided in Table 5.11. The kernel choice has little difference on the precisions with the exception of the sigmoid kernel under which the CART algorithm does not perform as well. The kernels do, however, have a large effect on the recall of the system though this is secondary to the evaluation under precision. There is

a large variation in results for the polynomial and RBF kernel. This suggests that the rules responsible for the high recall are in close competition with other rules.

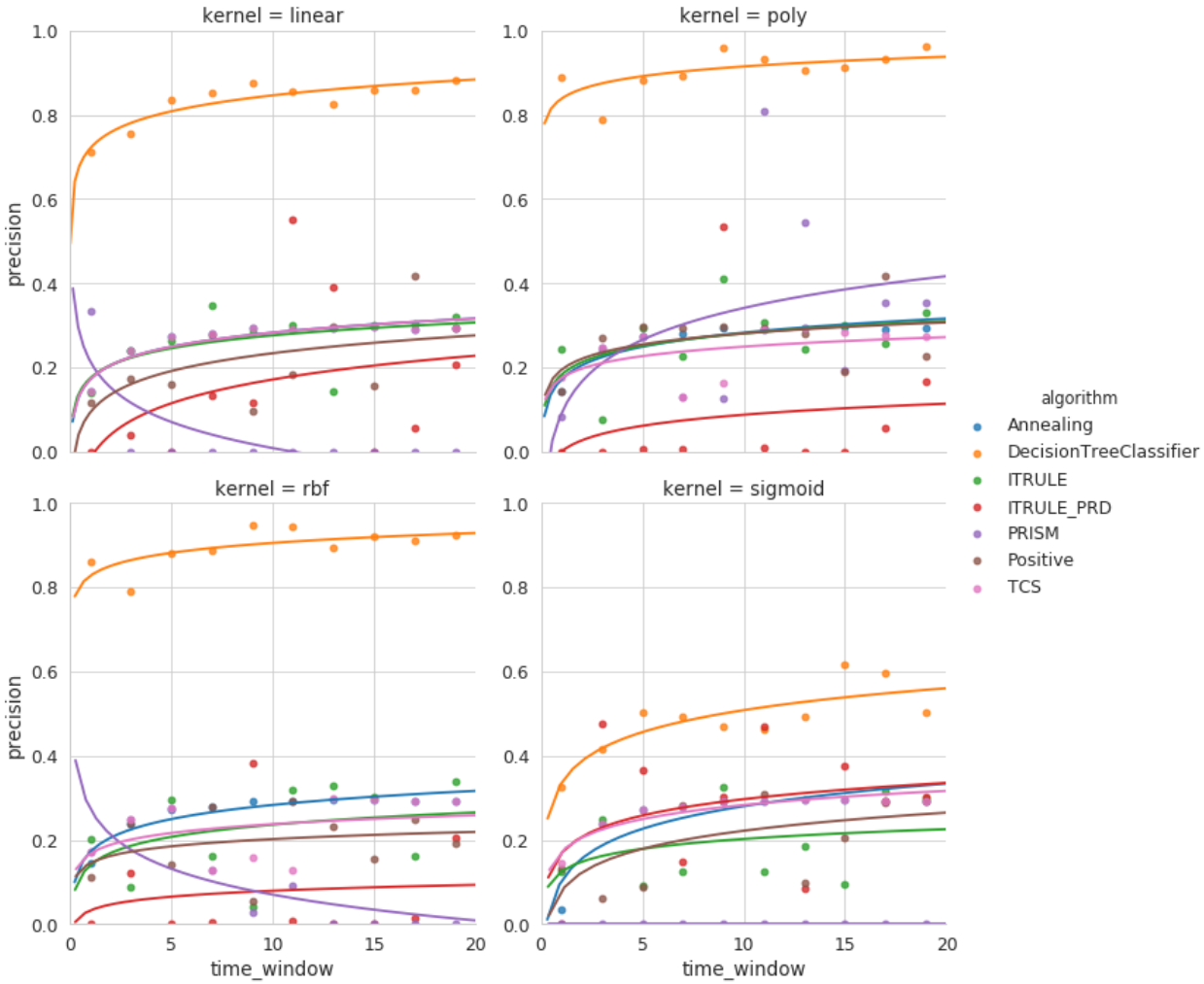

Figure 5.13: Precision of the Two Stage classifier under different kernels

5.2.1 One-Class SVM

The authors of [72] proposed an extension to the SVM to classify anomalies from unlabelled data. Where the original SVM determines the support vectors that produce an optimal margin between two classes, the One-Class or Unary SVM produces a hyperplane to maximise the distance from a high density of normal points to the origin, any point falling outside this plan are categorised as novel. During this process a class label for the other data points is not specified given it a similar resilience to new data points as found in an abstaining classifier.

Table 5.11: Mean results over all time windows for a two stage classifier varying kernels and rule induction models

algorithm	kernel	accuracy	precision	recall
ITRULE Annealing	linear	0.269803	0.269785	1.000000
ITRULE Annealing	poly	0.269777	0.269777	1.000000
ITRULE Annealing	rbf	0.269777	0.269777	1.000000
ITRULE Annealing	sigmoid	0.300986	0.258905	0.910406
DecisionTreeClassifier	linear	0.888923	0.831425	0.695305
DecisionTreeClassifier	poly	0.881917	0.905826	0.604988
DecisionTreeClassifier	rbf	0.895751	0.895853	0.666622
DecisionTreeClassifier	sigmoid	0.733485	0.487562	0.486057
ITRULE	linear	0.359560	0.264398	0.799775
ITRULE	poly	0.444512	0.269358	0.636508
ITRULE	rbf	0.443955	0.224744	0.584396
ITRULE	sigmoid	0.402580	0.193069	0.428411
ITRULE_PRD	linear	0.664264	0.149453	0.087450
ITRULE_PRD	poly	0.643854	0.077689	0.066094
ITRULE_PRD	rbf	0.658599	0.073700	0.074288
ITRULE_PRD	sigmoid	0.606272	0.281463	0.475431
Prism	linear	0.730197	0.033333	0.000176
Prism	poly	0.529995	0.311696	0.479527
Prism	rbf	0.522585	0.094302	0.323219
Prism	sigmoid	0.730223	0.000000	0.000000
ITRULE Positive	linear	0.457562	0.217441	0.451218
ITRULE Positive	poly	0.416464	0.270959	0.730776
ITRULE Positive	rbf	0.377795	0.195153	0.487249
ITRULE Positive	sigmoid	0.436899	0.204826	0.581314
ITRULE TCS	linear	0.270182	0.269818	0.999824
ITRULE TCS	poly	0.258447	0.240875	0.853467
ITRULE TCS	rbf	0.271624	0.229169	0.786998
ITRULE TCS	sigmoid	0.269777	0.269777	1.000000

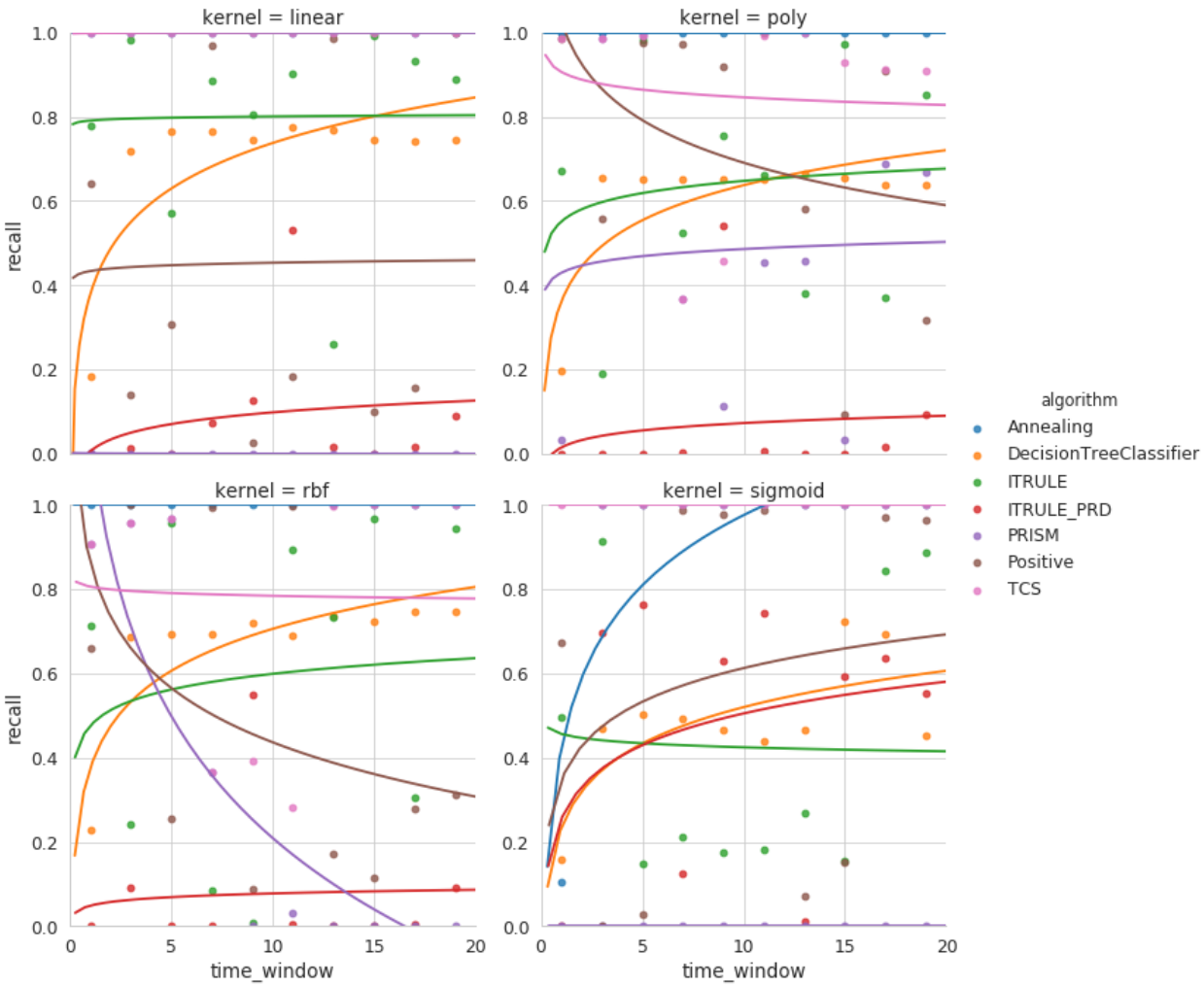

Figure 5.14: Recall of the Two Stage classifier under different kernels

This adapted SVM has successfully been used in anomaly detection for applications including vibration classification detect potential issues in jet engines[33] and image ranking[15]. Support Vector Domain Description (SVDD) is a second extension to the SVM for purposes of novelty detections novelty detection that minimises the radius of a hypersphere around the normal points, under a RBF kernel these two approaches are identical [52].

Table 5.12 describes the results of the Two Stage system with the One-Class SVM against the regular SVM. Under every kernel the accuracy and precision are higher with the SVM. This may be because the One-Class SVM depends on a density of points to train by that may not exists within the normal data points. The distance between precisions under the RBF suggest that the SVDD will perform similarly with other kernels.

The decision tree classifier does not directly produce human readable rules and an addi-

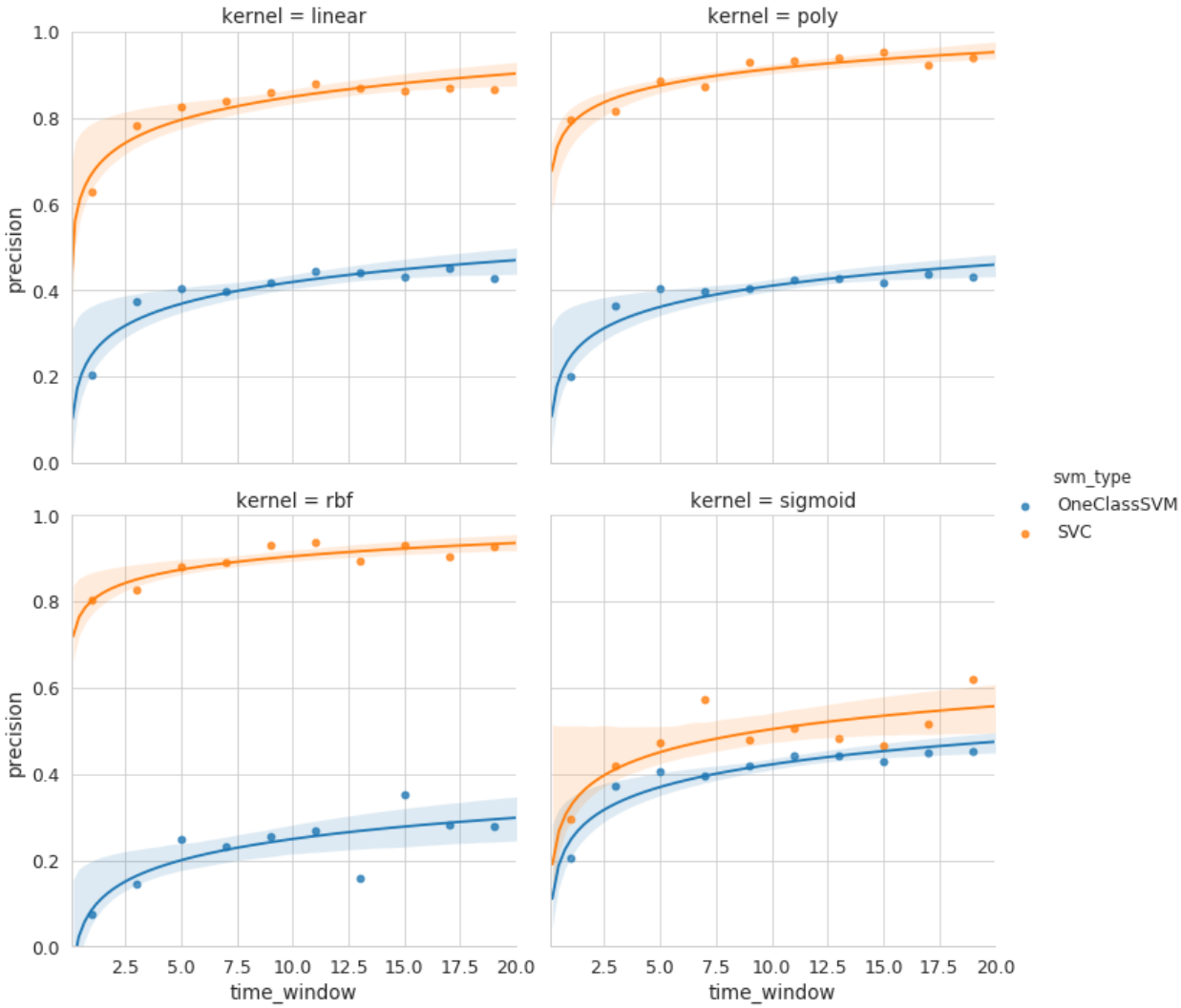

Figure 5.15: Precision of the Two Stage set up after with a regular SVM and a One Class SVM

tional stage is needed to extract them from the model. To extract the rules a search of the tree is performed to identify the path to every root node. As there is no danger of rules over lapping as with the ITRULE approaches the nodes that correspond to a negative classification can be removed, creating in effect an abstaining classifier. There is also no need to rank these rules as using the full depth of the tree will never result in clashes.

5.2.2 Predicting Time Windows

The above Two Stage system is capable of forecasting down events across a range of time windows but it still unable to supply an estimated arrival time of the forecasted event. Here

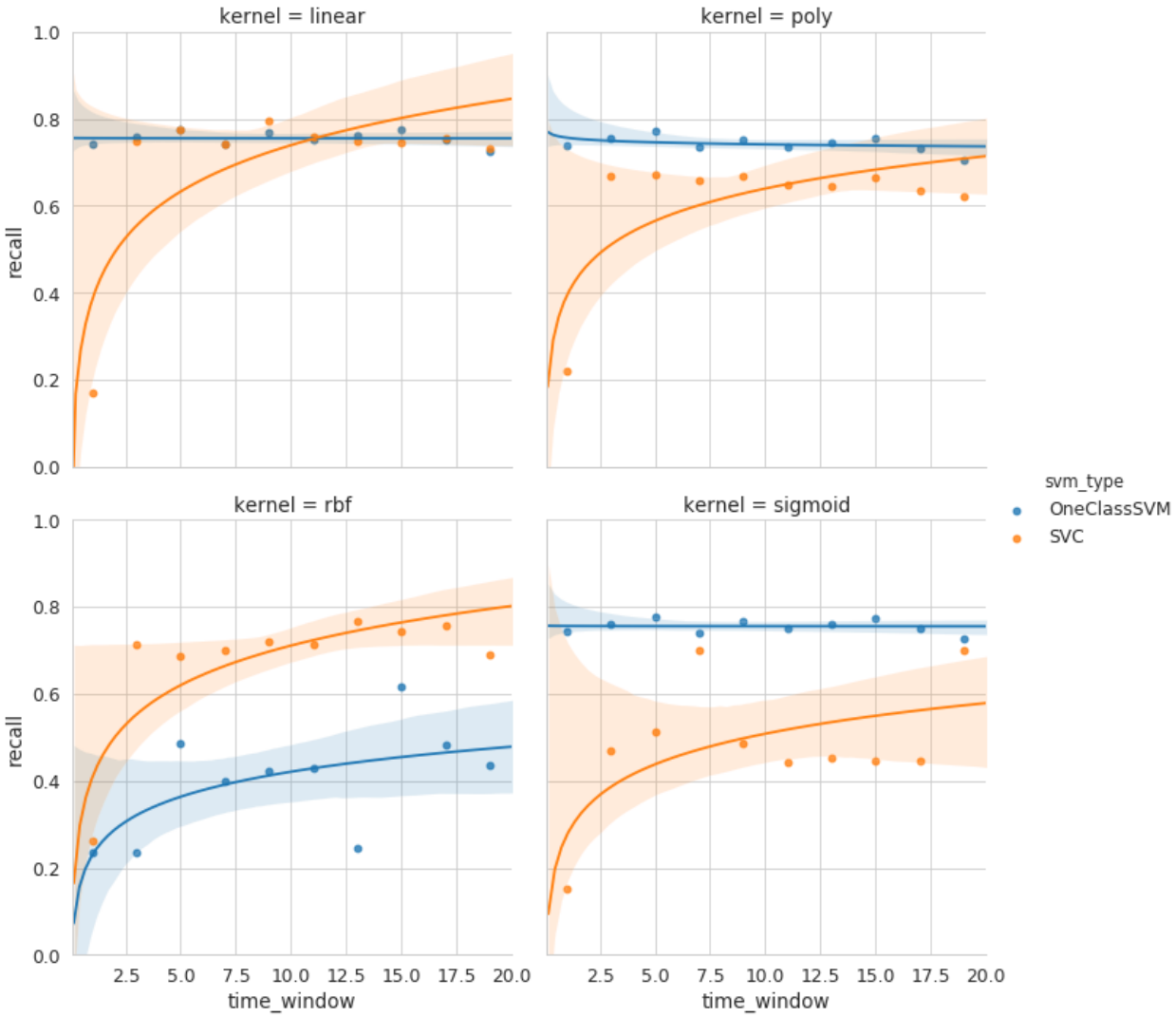

Figure 5.16: Recall of the Two Stage set up after with a regular SVM and a One Class SVM

Table 5.12: Performance of a classical SVM and a One Class SVM averaged over all time windows

	kernel	svm_type	accuracy	precision	recall
1	linear	OneClassSVM	0.641249	0.409355	0.756308
2	linear	SVM	0.890921	0.829129	0.701879
3	poly	OneClassSVM	0.624810	0.394666	0.746375
4	poly	SVM	0.884648	0.902688	0.614125
5	rbf	OneClassSVM	0.539201	0.314574	0.610516
6	rbf	SVM	0.900303	0.893688	0.684764
7	sigmoid	OneClassSVM	0.641426	0.409590	0.756954
8	sigmoid	SVM	0.738796	0.492098	0.491580

Table 5.13: Performance of algorithms for predicting time windows with SVM filtering

algorithm	mean_absolute_error	mean_squared_error	average_precision
Positive	0.471177	1.655150	0.005610
ITRULE	1.667191	4.932868	0.200365
ITRULE_PRD	1.945400	1.608659	0.010184
Annealing	0.081236	0.848363	0.089361
DecisionTreeClassifier	0.060030	0.337306	0.088260
PrismTCS	1.100600	3.820031	0.085355
Prism	1.087871	0.434097	0.084959

the experiments with time windows in Chapter 4 are repeated to see if this has become viable with the Two Stage system. As before the MSE and MAE are recorded. Figure 5.17 shows a large drop in error when using the Decision Tree and ITRULE Annealing. Though the reported errors are very low, a look at the rules produced by them demonstrates that the higher valued target classes, those labels that indicate the onset of a down event after a longer interval, do not appear. The low error rate is due to the large number of correctly classified events at the lower end of the scale as well as non-events. If these non-events are removed from the classification the mean absolute and mean squared errors are much higher, see Figure 5.18. The improvement offered by the annealing process is due to the decision tree over fitting on the majority class. A look at the rules produced by the tree shows that the nodes were limited to the labels 0 or 2, producing large errors when the target class was of a high value. ITRULE Annealing behaves similarly with the majority of the beam width containing rules that produce a 0 classification. Unlike the decision tree, the alternative classifications are not limited to one label and a number produce higher valued classifications, though these values are limited to 5, which may be too small an interval to be of use.

5.2.3 Predicting Critical Alarms in an Event Streams

In this section the system developed above is tested over an event stream replicating it's use in a live system. The trained model is passed one instance at a time to classify and with each classification an accompanying ground truth is produced. This ground-truth represents the presence of a critical alarm in the subsequent time window on the same SVLAN. The

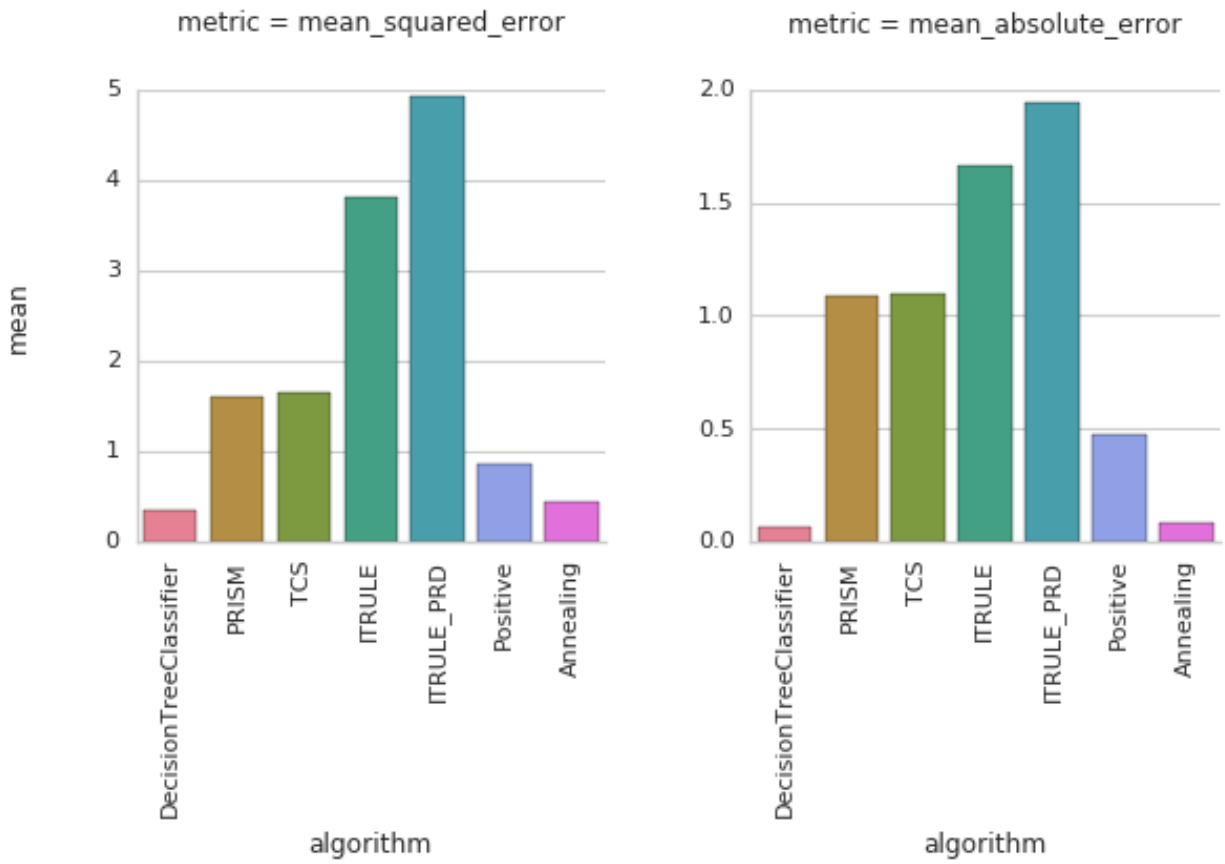

Figure 5.17: MSE and MAE cost of predicting time windows with SVM filter

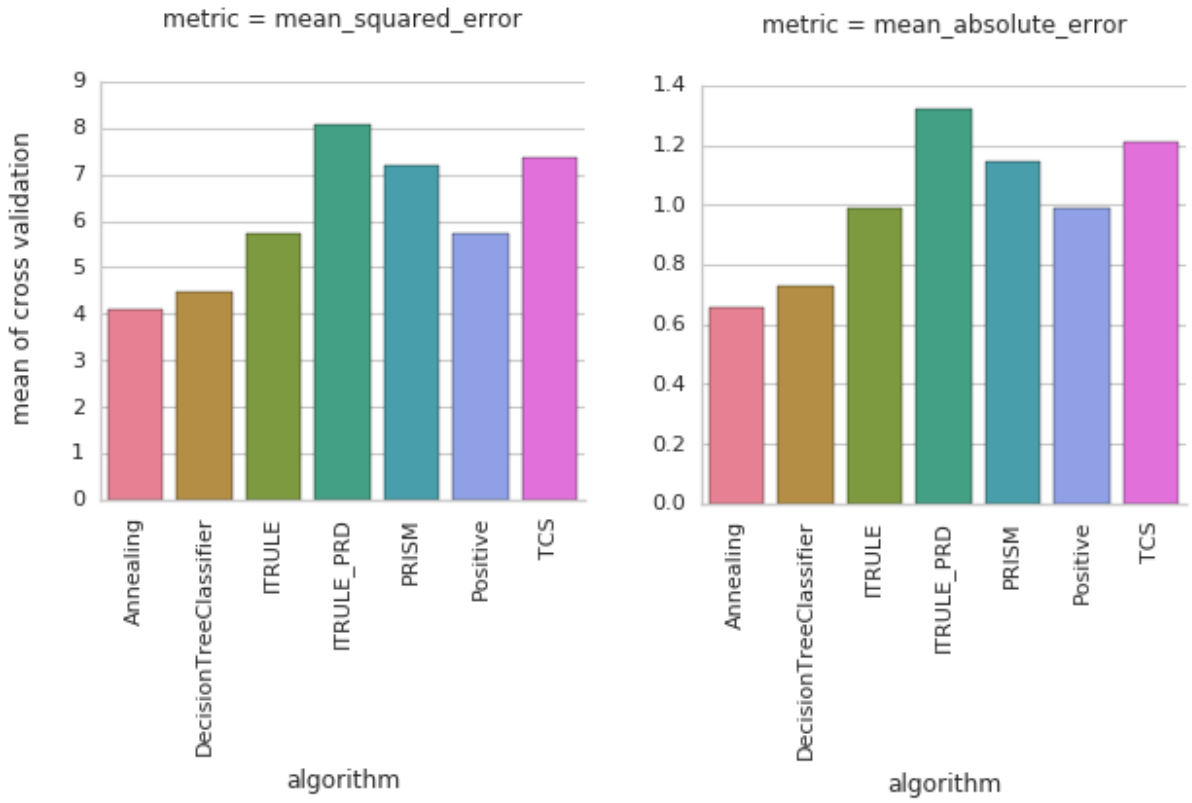

Figure 5.18: MSE and MAE for predicting time windows where predictions of non-events are discounted

output is a set of predictions and a set of ground-truths with which to verify. These tests were conducted with data from cluster 4 and further verified with a model trained and tested on data from cluster 2. The results are displayed in Figure 5.19. From this it can be seen that the precision, recall and accuracy are all high with only a few exceptions. The recall for predicting events under 5 minutes is low but in contrast the precision for these events is very high. The drop in recall for the 5 minutes window may indicate that the concepts required to predict these alarms were not available so close to the alarms generation.

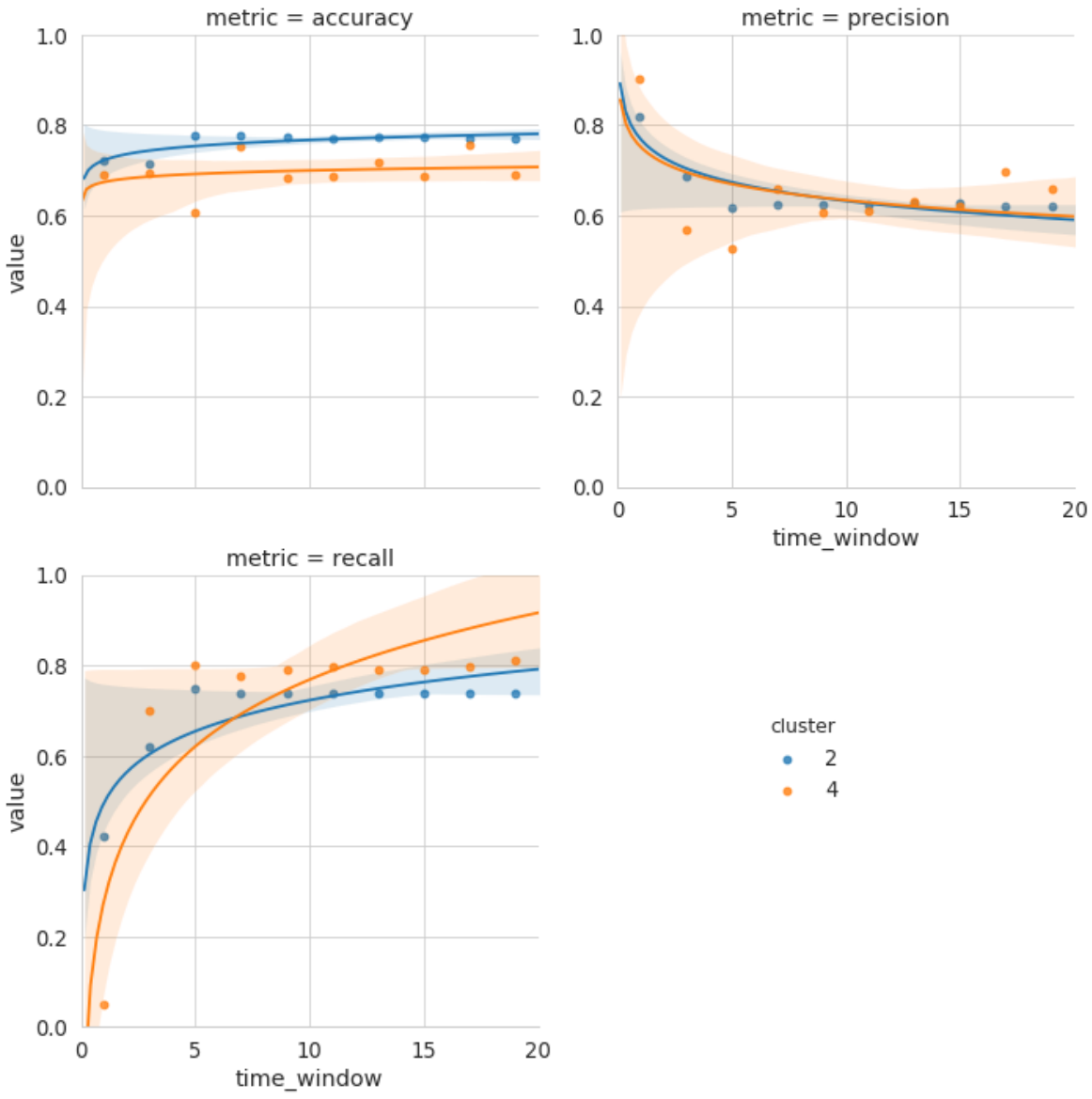

Figure 5.19: Precision, recall and accuracy of the rules produced by the Two Stage system

Training a model on the pre-event data ensures that the model is kept general across

SVLANs as the events seen are not SVLAN specific. As SVLANs are likely to contain attributes peculiar to themselves this is a strength of the system. An alternative would be to produce a model for each SVLAN which would be computationally inefficient and likely to lead to over fitting. A problem with this method of evaluation is that not all SVLANs will produce down events. Under such a circumstance any positive prediction will immediately create a precision of 0.0. A potential improvement to the system would be to monitor the number of times the rules fire. This could be examined and a confidence level for the alarm prediction produced.

5.3 Discussion of Results

In this chapter a number of options to improve the precision of the pre-event prediction method by refining the labelling process. The approaches considered were:

- The assignment of a default class based on the population of a split
- Producing templates with Frequent Pattern mining
- The application of models to filter out clashes

The direct population based clash resolution offered no improvement in precision, likely due to the reduction of the minority class labels leading to a lack of rule generation. This loss of rule generation for pre-events events is supported by the drop in recall. The complete removal of noise is also likely to contribute to over fitting which would lead to a drop in accuracy and precision. Approaches with frequent patterns also failed to produce an improvement in precision. The codebook approach produced the lowest precisions seen in these experiments using both deterministic and probabilistic approaches. This method relies on the tuning of a distance parameter to distinguish events preceding a down event from other events that appear with similar regularity. As this distance threshold was increased the recall and precision also increased, however, as with the previous approach too few alarms are left to produce a broad set of rules capable of predicting down events in unseen data. Frequent

pattern based approaches, though the basis of many approaches in the literature, were not applicable to this data due to the lack of repeated patterns with sufficient confidence, or indeed any pattern. This is possibly due to the bursty nature of the data but the application of a method to extract patterns from bursts also produced very few repeated patterns.

The most successful approach was the modelling based filtering, specifically with an SVM and an RBF Kernel. This method employs a black box model to relabel instances and remove clashes. As with previous relabelling approaches this comes with the problem of over fitting and reducing the minority target class below a critical point. In these experiments the previous versions of ITRULE that performed well on the initial data saw a drop in precision and a rise in recall. As it has been established that a number of these events are incidental, a rise in recall is not desirable for pre-event prediction, the goal is instead to discern between the incidental and actual pre-events. The number of pre-downs being caught in the rules increased but the rules now capture a larger number of false positives. Following the set up laid out in[4] a decision tree was also employed as the rule induction algorithm. This system produced the highest precisions, especially with larger time windows. Decision trees have a number of disadvantages compared to the beam search approaches examined, including the need for an additional rule extraction phase, but the resulting precisions would seem to meet the demands of the project.

Finally the SVM system was trained on time windows in an effort to produce rules that predict the interval of the oncoming event. These experiments produced high MAE and MSEs for all algorithms involved suggesting that this method is not reliable. This could potentially be addressed by adapting the algorithms for regression as this is no longer strictly a classification task, allowing for more intuitive feature value selection.

Chapter 6

Conclusions and Future Work

In this concluding chapter the contributions to knowledge made by this work are described and the extent to which the project aims and objectives have been met is examined. Throughout this thesis there have been opportunities to expand on the experiments and methods used, these are gathered together and summarised as future potential avenues of research.

6.1 Contributions

There have been a number of contributions to knowledge during this work including novel algorithmic developments and steps made towards the ultimate goal of the project. As mentioned in Chapter 1, there have been three publications made during the course of the research on the subject of machine learning as applied to telecommunication data. These include a book chapter on event processing techniques, a positioning paper laying out the OGRI system described in Chapter 4 and an empirical evaluation of some of the new algorithms and techniques created to meet the ends of this project.

The OGRI system for classifying events in a data stream has not been fully realised and this will be touched on in the following Future Work section. All the distinct components of the system save two have been realised outside of this work and the rule induction component was part of the focus of Chapter 4. With this said the structure is novel and stands as a contribution.

The algorithm work mentioned above refers to the variants of Rule Induction that have been developed for the purpose of predicting events with feature rich expressive rules. The paper [87] introduced two new algorithms: ITRULE PRD and JPRISM for this purpose. Both are driven by the J-measure metric and are focussed on promoting rare events, although they have struggled with class imbalance. The third algorithm developed was an extension to the ITRULE PRD algorithm incorporating the Simulated Annealing a means to overcome the local maxima issue that occurs in the original ITRULE implementation, and indeed on other algorithms that incorporate the beam search heuristic as this where the problem lies. A thorough examination has been made of ITRULE Annealing and the relationship between the hyper-parameters α and temperature. It has been demonstrated that the annealing approach is an effective way of keeping the feature space represented within the rule set broad and by extension enables it to predict a wider range of instances. This has a positive effect on the recall, precision and tentative accuracy of the system when compared to the other forms of ITRULE in this work. It has also been shown that ITRULE Annealing can incorporate a number of different metrics to control the acceptance probability. Trialled in this work were Jdistance (another minor contribution) and Jmax.

There were issues using traditional forecasting approaches from the Frequent Pattern Mining family of algorithms on the BT data set. This can be attributed to either the severe bursty nature of the data or the lack of a viable concept within the alarm type class that Frequent Pattern Mining requires. The pre-event transformation is a contribution from this work that is designed to convert classifiers into forecasters. The transformation has been successfully validated on a data stream from two clusters. It is a simple transformation that has issues with introducing class label noise but the advantages of a successful implementation are:

- Algorithms are able to forecast using a range of features enabling forecasting when the pertinent concepts are not contained within the alarm type
- It allows a far greater range of features to be included in the rules, adding to their expressiveness

- It is not algorithm specific, allowing a large range of learners to be used in a forecasting problem

To refine the pre-event marking process an exploration of filtering techniques was made in Chapter 5. This resulted in the adaptation of a number of pre-existing techniques including a method to transform bursty events and a codebook method which, as to the authors knowledge, has not been applied to a prediction problem before, albeit it was not successful. These refinements lead to the final contribution of this work: a two stage classifier that produces high precision expressive rules to predict alarms in the BT data set.

6.2 Conclusion

In this section the outcome of this thesis is broadly discussed, beginning with the extent to which this work has met the stated objectives in Chapter 1 and followed by a discussion of some of the key learnings. To begin with the objectives are restated and the work towards each one is described.

6.2.1 Research Aims and Objectives

The overall aim of this project is to determine if it is possible to produce human readable rules that forecast network alarms ahead of time in a telecommunication network. The objectives can more precisely be defined as:

The system produced should generate a rule based model that can be used by network operators and engineers to diagnose and correct a fault, as well as mitigate any further faults. The Two Stage classifier performed best when paired with a decision tree which, although a white box algorithm, is not a rule based classifier. However, the Decision Tree can be represented as a set of expressive rules which meets this objective.

Ideally produced rules should contain an expected time before the fault occurs and potentially provide a location. Experiments to produce an expected time interval

with rule induction were not successful. Although it was established that the system can forecast an alarm ahead of time, and that this was the case for a range of time windows, it is not possible to claim this objective was met.

Having re-established the initial objectives the next section details any additional findings produced by this thesis.

6.2.2 Research Findings

One of the largest contributions of this work was a study of ITRULE and the problems with the beam search approach. It was hypothesised that the issue with ITRULE was that it was prone to locating a local maxima, producing partial rules that limit its predictive power. This was addressed with the development of ITRULE Annealing that was able to increase the feature represented by the rule set and in turn outperform ITRULE across all recorded metrics.

A number of filtering approaches were trialled unsuccessfully to refine the pre-event method. Amongst these were a number of Frequent Pattern Mining algorithms. It was demonstrated that these could not locate a pattern of sufficient support and confidence in the data to be utilised in the prediction process. The moderate success of pre-event prediction has demonstrated that Frequent Pattern Mining, although a well established field, is limited by the assumption that the underlying concept can be represented by an alarm type or key feature.

ITRULE Positive was an alteration made to ITRULE by an earlier work[41] used to discover patterns in a network stream. In this work it has been applied to a classification task and has on occasion outperformed all other applied algorithms due to it's minor but effective alteration to the J-measure that promotes rarer events.

6.3 Future Work

Throughout this work there have been a number of opportunities to widen the scope of the investigation into various aspects. A number of these have been left unexplored but may prove to be promising areas of research. These are presented here as potential future work to expand upon this thesis further or as an independent offshoot of this investigation.

6.3.1 Improvements to the Two Stage System

An issue with the Two Stage classifier presented is it's reliance on Hot-Point Encoding to convert the categorical features into a numerical feature set applicable to an SVM. This is an expensive process that greatly increases the feature space and creates a large amount of redundant features. Alternatives are available that do not have the same short comings that may yield better results than those achieved here. Of note are cp-coding, a method that encodes each categorical value with a probabilistic value that describes its influence on the target class. This has been used with great effectiveness in the Cat-Boost regressor[21]. The transformation is applied in place and so there is no increase in dimensionality. A similar method based on signal encoding is presented here[66]. Other methods include a phase of feature selection on the produced boolean vector to either discard or merge sparse features. There may also be gains to be made using alternative dimensionality transformations in place of Hot-Point Encoding and possibly the SVM. Random Forest's can be used to represent an instance as a vector representing a collection of leaf nodes. This method does not guarantee a reduction in dimensionality as it is directly linked to the number of leaves available in the Random Forest but the transformation may save processing time and further expose the alarm data's concepts.

In Chapter 2 a number of methods for extracting rules from a SVM[3] were described. In this thesis the SVM_DT[34] approach was applied to great effect. It has been shown that the rules can extracted from the support vectors by either training a model on these data points or using them in a clustering algorithm to assign class labels. There are also wide array of SVM implementations available. Some, such as[68], are designed specifically to be resilient

to noise, these could be explored.

6.3.2 ITRULE Investigation

ITRULE Annealing and ITRULE Positive proved the most successful implementations of ITRULE when predicting down events. These two algorithms are differentiated by two features that are not mutually exclusive and may offer further benefits if combined. Incorporating the two would mean a small change to ITRULE Annealing to alter the J-measure calculation which may, as it has done for ITRULE Positive, allow it to better predict rare events.

ITRULE Annealing uses Simulated Annealing to moderate the distribution of features within the beam. It has been demonstrated that the parameters for α and temperature must be carefully chosen. There are opportunities for experimenting with alternative cooling strategies as opposed to the strategy originally implemented. It is also clear that an optimal cooling strategy should be linked to the number of expected challenges, i.e. the number of times a rule will seek admission into the beam. This can be calculated to some extent with prior knowledge of the feature space and beam width.

6.3.3 OGRI Improvements

The OGRI system was never fully realised in this work although it is only missing a component to handle additional contextual information. In Chapter 2 the algorithm TP Mining[18] was described that promoted rules based on their topographical information. As the rule sets are actively managed in OGRI this could be incorporated to promote rules that are generated from similar SVLANs or geographical locations.

6.4 Concluding Remarks

This chapter has summarised the work and contributions made throughout this thesis. Although not all objectives were fully met (producing an estimated time of arrival for events

was not successful) the majority of aims have been achieved. Furthermore there have been many novel contributions through the development of new techniques. The final section has outlined some additional avenues of research that could be pursued if this project were to continue in another form.

Bibliography

[1] Mariam Adedoyin-Olowe, Mohamed Medhat Gaber, and Frederic Stahl. TRCM: A methodology for temporal analysis of evolving concepts in Twitter. In *Artificial Intelligence and Soft Computing*, volume 7895 LNAI, pages 135–145. Springer, 2013.

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(2):207–216, 1993.

[3] Nahla Barakat and Andrew P Bradley. Neurocomputing Rule extraction from support vector machines : A review. *Neurocomputing*, 74(1-3):178–190, 2010.

[4] Nahla H. Barakat and Andrew P. Bradley. Rule extraction from support vector machines: A sequential covering approach. *IEEE Transactions on Knowledge and Data Engineering*, 19(6):729–741, 2007.

[5] Jerzy Blaszczynski, Jerzy Stefanowski, and Magdalena Zajaçc. Ensembles of abstaining classifiers based on rule sets. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5722 LNAI:382–391, 2009.

[6] Christian Borgelt. Efficient Implementations of Apriori and Eclat. In *Workshop of Frequent Item Set Mining Implementations*, pages 1089–1114, 2003.

[7] Max Bramer. Automatic Induction of Classification Rules from Examples Using N-Prism. *Research and Development in Intelligent Systems XVI*, pages 99–121, 2000.

[8] Max Bramer. An information-theoretic approach to the pre-pruning of classification rules. *Intelligent information processing*, pages 201–212, 2002.

[9] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

[10] Jadzia Cendrowska. PRISM: An algorithm for inducing modular rules. *International Journal of Man-Machine Studies*, 27:349–370, 1987.

[11] Sharma Chakravarthy, V. Krishnaprasad, Eman Anwar, and Seung-Kyum Kim. Composite Events for Active Databases: Semantics, Contexts and Detection. In *VLDB*, pages 606–617, 1994.

[12] Lorna Christie and Alison Tully. Security of UK Telecommunications. *POST Notes: Houses of Parliament Office of Science and Technology*, (584):1–6, 2018.

[13] Carlos A. Coello Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems*, 1(3):269–308, 1999.

[14] William W Cohen. Fast Effective Rule Induction. In *Machine Learning Proceedings 1995*, pages 115–123. Elsevier, 1995.

[15] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (Csur)*, 40(2):5, 2008.

[16] Kalyanmoy Deb. Multi-objective optimization using evolutionary algorithms: an introduction. *Multi-objective evolutionary optimisation for product design and manufacturing*, pages 1–24, 2011.

[17] F Della Croce, M Ghirardi, and R Tadei. Recovering Beam Search: Enhancing the Beam Search Approach for Combinatorial Optimization Problems. *Journal of Heuristics*, 10(1):89–104, 2004.

[18] Ann Devitt, Joseph Duffin, and Robert Moloney. Topographical proximity for mining network alarm data. *Proceeding of the 2005 ACM SIGCOMM workshop on Mining network data - MineNet '05*, page 179, 2005.

[19] Dua Dheeru and Efi Karra Taniskidou. {UCI} Machine Learning Repository, 2017.

[20] C. Domeniconi, C. Perng, R. Vilalta, and S. Ma. A classification approach for prediction of target events in temporal sequences. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 125–137. Springer, 2002.

[21] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

[22] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xu Xiaowei. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *KDD*, 96(34):226–231, 1996.

[23] Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, 2014.

[24] Johannes Fürnkranz and Peter A. Flach. An Analysis of Rule Evaluation Metrics. *Proceedings of the 20th International Conference on Machine Learning (ICML-2003)*, pages 202–209, 2003.

[25] Bart Goethals. Survey on Frequent Pattern Mining. *Manuscript*, pages 1–43, 2003.

[26] G. H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14(5):403–420, Apr 1970.

[27] Rodney M Goodman, Barry Ambrose, Hayes Latin, and C T Ulmer. *Noaa: An Expert System Managing The Telephone Network*. Springer, 1995.

[28] R W Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29(2):147–160, apr 1950.

[29] Jiawei Han, Hong Cheng, Dong Xin, and Xifeng Yan. Frequent pattern mining: Current status and future directions. *Data Mining and Knowledge Discovery*, 15(1):55–86, 2007.

[30] Jiawei Han, Guozhu Dong, and Yiwen Yin. Efficient mining of partial periodic patterns in time series database. In *Proceedings 15th International Conference on Data Engineering (Cat. No.99CB36337)*, pages 106–115, mar 1999.

[31] Lydia Harriss. Telecommunications Infrastructure: Cabling, Ducts and Poles. *POST Notes: Houses of Parliament Office of Science and Technology*, (24), 2017.

[32] Kimmo Hätönen, Mika Klemettinen, Heikki Mannila, Pirjo Ronkainen, and Hannu Toivonen. Rule Discovery in Alarm Databases. Technical Report C-1996-7, University of Helsinki, Helsinki, 1996.

[33] P Hayton, B Schölkopf, L Tarassenko, and P Anuzis. Support vector novelty detection applied to jet engine vibration spectra. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 946–952. MIT Press, 2001.

[34] Jieyue He, Hae Jin Hu, Robert Harrison, Phang C. Tai, and Yi Pan. Rule generation for protein secondary structure prediction with support vector machines and decision tree. *IEEE Transactions on Nanobioscience*, 5(1):46–52, 2006.

[35] Annika Hinze, Kai Sachs, and Alejandro Buchmann. Event-based Applications and Enabling Technologies. In J.N. Kok, J. Koronacki, Lopez de Mantaras, S. R., Matwin, and D. Mladenic, editors, *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, pages 1:1—-1:15, New York, New York, USA, 2009. ACM Press.

[36] Kevin Anthony Hoff and Masooda Bashir. Trust in automation: Integrating empirical evidence on factors that influence trust. *Human Factors*, 57(3):407–434, 2015.

[37] Jens Christian Hühn and Eyke Hüllermeier. FR3: A fuzzy rule learner for inducing reliable classifiers. *IEEE Transactions on Fuzzy Systems*, 17(1):138–149, 2009.

[38] Johan Huysmans, Bart Baesens, and Jan Vanthienen. Using Rule Extraction to Improve the Comprehensibility of Predictive Models. *SSRN Electronic Journal*, pages 1–55, 2006.

[39] Eugene Isaacson. Numerical Recipes in C: The Art of Scientific Computing (William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling); Numerical Recipes: Example Book (C) (William T. Vetterling, Saul A. Teukolsky, William H. Press, and Brian P. Flan. *SIAM Review*, 31(1):142–142, mar 1989.

[40] Mohammad Jaudet, Amir Hussain, and Kamran Sharif. Temporal Classification for Fault-prediction in a real-world Telecommunications Network. pages 209–214, 2005.

[41] Srikanth Kandula, Ranveer Chandra, and Dina Katabi. What's going on?: learning communication rules in edge networks. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 87–98. ACM, 2008.

[42] Srikanth Kandula, Dina Katabi, and Jean-philippe Vasseur. Shrink: A tool for failure diagnosis in IP networks. *Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 173–178, 2005.

[43] M Karsai, K Kaski, A L Barabasi, and J Kertesz. Universal features of correlated bursty behaviour. *Scientific Reports*, 2:7, 2012.

[44] E. Keogh, S. Chu, D. Hart, and M. Pazzani. An online algorithm for segmenting time series. *Proceedings 2001 IEEE International Conference on Data Mining*, pages 289–296, 2001.

[45] Imran Khan, Joshua Z Huang, and Nguyen Thanh Tung. Learning Time-based Rules for Prediction of Alarms from Telecom Alarm Data Using Ant Colony Optimization. *International Journal of Computer and Information Technology (ISSN: 2279 – 0764)*, 03(01):139–147, 2014.

[46] Scott C Kirkpatrick, D Gelatt, and M P Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[47] Mika Klemettinen:. *A Knowledge Discovery Methodology for Telecommunication Network Alarm Databases*. Number October. 1999.

[48] Mika Klemettinen. *A Knowledge Discovery Methodology for Telecommunication Network Alarm Databases*. Number October. 1999.

[49] S Kliger, S Yemini, and Y Yemini. A coding approach to event correlation. ... *Network Management IV*, pages 1–12, 1995.

[50] Ramana Rao Kompella, Jennifer Yates, Albert Greenberg, and Alex C. Snoeren. Detection and localization of network black holes. In *Proceedings - IEEE INFOCOM*, pages 2180–2188, 2007.

[51] Alexander Lachmann, Mirek Riedewald, Zhuolun Zhang, Sufen Wang, Alexander Lachmann, and Mirek Riedewald. Finding relevant patterns in bursty sequences. *Proceedings of the VLDB Endowment*, 1(1):78–89, 2008.

[52] Christoph H. Lampert. Kernel Methods in Computer Vision. *Foundations and Trends® in Computer Graphics and Vision*, 4(3):193–285, 2007.

[53] Thien Le, Frederic Stahl, Mohamed Medhat Gaber, João Bártolo Gomes, and Giuseppe Di Fatta. On expressiveness and uncertainty awareness in rule-based classification for data streams. *Neurocomputing*, 265:127–141, 2017.

[54] Thien Le, Frederic Stahl, João Bártolo Gomes, Mohamed Medhat Gaber, and Giuseppe Di Fatta. Computationally Efficient Rule-Based Classification for Continuous Streaming Data. In *Research and Development in Intelligent Systems XXIV*, page 2008. Springer International Publishing, 2014.

[55] O-joun Lee, Eunsoon You, Min-sung Hong, and Jason J Jung. Adaptive Complex Event Processing Based on Collaborative Rule Mining Engine. In Raymond Kosala Ngoc Thanh Nguyen, Bogdan Trawiński, editor, *Intelligent Information and Database Systems*, volume 9011, pages 430–439. Springer International Publishing, 2015.

[56] Duen Ren Liu, Meng Jung Shih, Churn Jung Liau, and Chin Hui Lai. Mining the change of event trends for decision support in environmental scanning. In *Expert Systems with Applications*, volume 36, pages 972–984, Beijing, 2009. Elsevier.

[57] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

[58] Ryszard S. Michalski. On the Quasi-Minimal Solution of the General Covering Problem, 1969.

[59] Frederic P. Miller, Agnes F. Vandome, and John McBrewster. *Levenshtein Distance: Information Theory, Computer Science, String (Computer Science), String Metric, Damerau?Levenshtein Distance, Spell Checker, Hamming Distance*. Alpha Press, 2009.

[60] Hamid Mohamadi, Jafar Habibi, Mohammad Saniee Abadeh, and Hamid Saadi. Data mining with a simulated annealing based fuzzy classification system. *Pattern Recognition*, 41(5):1841–1850, 2008.

[61] Nacem Iqbal Mohammad Jaudet. Neural networks for fault-prediction in a telecommunications network. *8th International Multitopic Conference, 2004. Proceedings of INMIC 2004.*, pages 315–320, 2004.

[62] David F. Nettleton, Albert Orriols-Puig, and Albert Fornells. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review*, 33(3-4):275–306, 2010.

[63] UK Office of Communications. Communications Act 2003, 2003.

[64] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[65] Louis Perrochon, Walter Mann, Stephane Kasriel, and David C. Luckham. Event Mining with Event Processing Networks. *Methodologies for Knowledge Discovery and Data Mining. Third Pacific-Asia Conference, PAKDD-99 Beijing, China, April 26–28, 1999 Proceedings*, pages 474–478, 1999.

[66] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing.

[67] Mirko Polato and Fabio Aiolli. Boolean kernels for interpretable kernel machines. (April):25–27, 2018.

[68] T Prayoonpitak and Sarawan Wongsa. A robust one -class support vector machine using gaussian -based penalty factor and its application to fault detection. *International Journal of Materials, Mechanics and Manufacturing*, 5:146–152, 08 2017.

[69] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.

[70] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, 1993.

[71] Ulrich Rückert and Stefan Kramer. Towards tight bounds for rule learning. *Twenty-first international conference on Machine learning - ICML '04*, page 90, 2004.

[72] Bernhard Sch, Robert Williamson, Alex Smola, John Shawe-taylor, and John Platt. Support Vector Method for Novelty Detection. In *NIPS*, pages 582–588, 1999.

[73] Jeffrey Curtis Schlimmer. Concept acquisition through representational adjustment. 1987.

[74] Poul Nicholas Schultz-Moller. *Distributed Detection of Event Patterns*. PhD thesis, Imperial College of Science, Technology and Medicine, 2008.

[75] Padhraic Smyth and Rodney M. Goodman. An Information Theoretic Approach to Rule Induction from Databases, 1992.

[76] Frederic Stahl and Max Bramer. Random prism: A noise-tolerant alternative to random forests. *Expert Systems*, 31(5):411–420, 2014.

[77] Frederic Stahl, Mohamed Medhat Gaber, and Manuel Martin Salvador. eRules: A modular adaptive classification rule learning algorithm for data streams. *Res. and Dev.*

in Intelligent Syst. XXIX: Incorporating Applications and Innovations in Intel. Sys. XX - AI 2012, 32nd SGAI Int. Conf. on Innovative Techniques and Applications of Artificial Intel., pages 65–78, 2012.

[78] Brian J. Taylor and Marjorie A. Darrah. Rule extraction as a formal method for the verification and validation of neural networks. *Proceedings of the International Joint Conference on Neural Networks*, 5:2915–2920, 2005.

[79] Risto Vaarandi. A breadth-first algorithm for mining frequent patterns from event logs. *Intelligence in Communication Systems*, pages 293–308, 2004.

[80] Jorge M S Valente and Rui A F S Alves. Filtered and recovering beam search algorithms for the early / tardy scheduling problem with no idle time *. *Computers & Industrial Engineering*, 48(351):363–375, 2005.

[81] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer New York, New York, NY, 2000.

[82] Gary M Weiss. Data Mining in the Telecommunications Industry. *Data Mining and Knowledge Discovery Handbook*, pages 1189–1201, 2005.

[83] Sholom M Weiss and Nitin Indurkhya. *Predictive data mining: a practical guide*. Morgan Kaufmann, 1998.

[84] Chris Wrench. personal communication with BT.

[85] Chris Wrench, Frederic Stahl, Giuseppe Di Fatta, Vidhyalakshmi Karthikeyan, and Detlef Nauck. Research and Development in Intelligent Systems XXXII: Incorporating Applications and Innovations in Intelligent Systems XXIII. chapter Towards Ex, pages 191–196. Springer International Publishing, Cham, 2015.

[86] Chris Wrench, Frederic Stahl, and Guiseppe Di Fata. A Review of Real Time Complex Event Processing. (December):1–7, 2014.

[87] Chris Wrench, Frederic Stahl, Thien Le, Giuseppe Di Fatta, Vidhyalakshmi Karthikeyan, and Detlef Nauck. A Method of Rule Induction for Predicting and Describing Future Alarms in a Telecommunication Network. In *Research and Development in Intelligent Systems XXXIII*, pages 309–323. Springer International Publishing, Cham, 2016.

[88] Shaula Alexander Yemini, Shmuel Kliger, Eyal Mozes, Yechiam Yemini, and David Ohsie. High Speed and Robust Event Correlation. *IEEE Communications Magazine*, 34(5):82–90, 1996.

[89] Yechiam Yemini, Shaula Yemini, and Shmuel Kliger. Apparatus and method for event correlation and problem reporting, jun 2001.

Appendix A

A.1 PRISM

An additional contribution was published in the paper [87] was *JPrism*. JPrism was developed to promote the induction of interesting rules using the J-measure as the driving metric as opposed to straight probability. In some cases JPrism returned higher accuracies then either of it's parent algorithms. The algorithm for JPrism is laid out in Algorithm 5. To keep the rules general early stopping is controlled using Jmax and pruning is performed using confidence.

Algorithm 5 JPrism algorithm

```
1: Data set D with Target Classes C_n and Attributes A_n
2: for Every Class C_i in D do
3:   while D contains classes other than C_i do
4:     for Every Attribute A_i in D do
5:       for Every Attribute Value A_iv in A_i do
6:         Generate Rule R_n with Rule Term A_iv
7:         Calculate j
8:         J_max ← J_max * ThresholdT
9:         add R_n to Candidate Rule Set
10:      end for
11:    end for
12:    Select R_n where j(R_n) is maximised
13:    Remove Instances not covered by R_n
14:    if j(R_n) > J_max(R_n) then
15:      Rule Complete
16:      Break
17:    end if
18:  end while
19:  for all Rule Terms RT_i : in Rule R do
20:    if Confidence( RT_i) <Confidence(RT_{i-1}) then
21:      Remove RT_i from R
22:    end if
23:  end for
24:  C_i ← Remove Instances Covered by R_n from D
25: end for
```

Base tests with ITRULE and PRISM on two stock training sets, the cars data set and the congressional used in the original ITRULE paper, both available from [19] are laid out below in Table A.1 and Table A.2. The congressional voting set contains 16 boolean features (with missing values) relating to votes cast by one member and a class label relating to the party of that member. There is a slight class imbalance in this data set as the 'republican' class label makes up only 38.6% of rows. The cars data set contains 6 nominal attributes and a nominal class label. These models were built without reference to a positive or negative class value and the Precision, Recall and F1 scores are the unweighted mean of all class values. All values are 3 fold cross validated and the mean and standard deviation reported.

Table A.1: ITRULE on the Cars data set

	fold 1	fold 2	fold 3	mean	std
abstain rate	0.000000	0.000000	0.000000	0.000000	0.000000
accuracy	0.229167	0.215278	0.222222	0.222222	0.005670
confidence	0.077958	0.088839	0.082540	0.083112	0.004460
Jmax	2.376887	2.682172	2.496835	2.518631	0.125581
J-measure	0.367232	0.358698	0.350695	0.358875	0.006752
tentative_accuracy	0.229167	0.215278	0.222222	0.222222	0.005670

Table A.2: PRISM on the Cars data set

	fold 1	fold 2	fold 3	mean	std
abstain rate	0.000000	0.000000	0.000000	0.000000	0.000000
accuracy	0.229167	0.217014	0.251736	0.232639	0.014386
confidence	1.940701	2.987654	2.527363	2.485239	0.428454
tentative_accuracy	0.229167	0.217014	0.251736	0.232639	0.014386

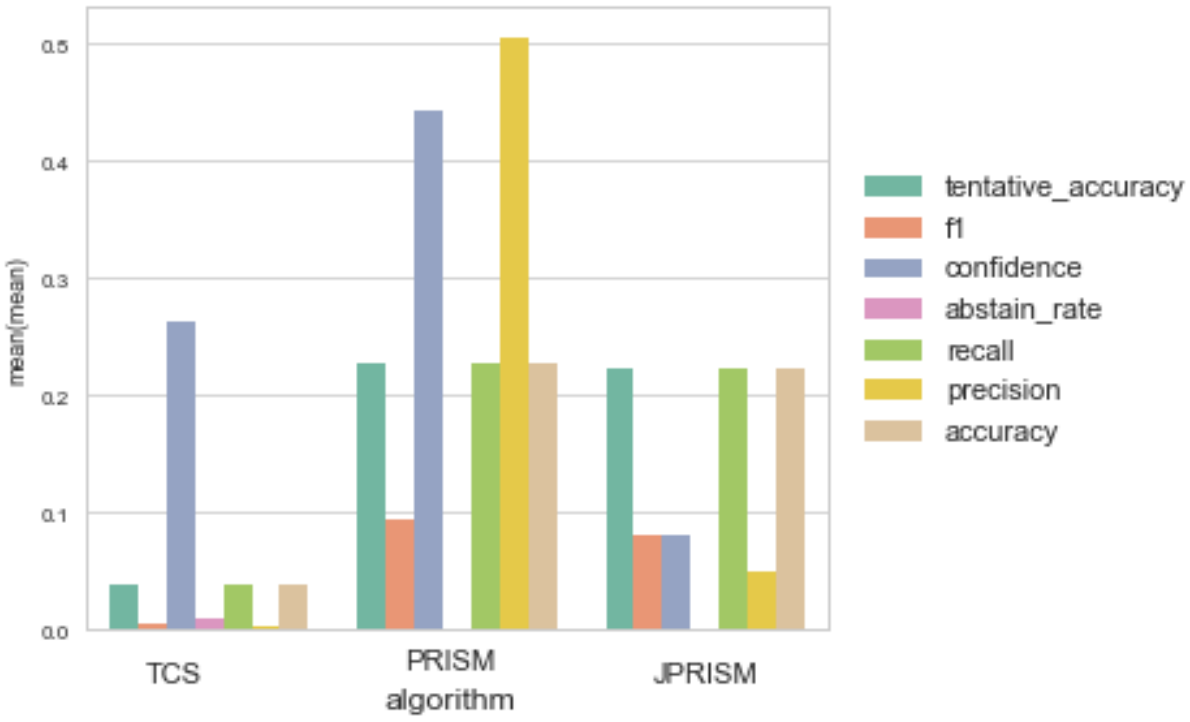

Figure A.1: Performance of Prism variants on Cars, a multi-label data set.

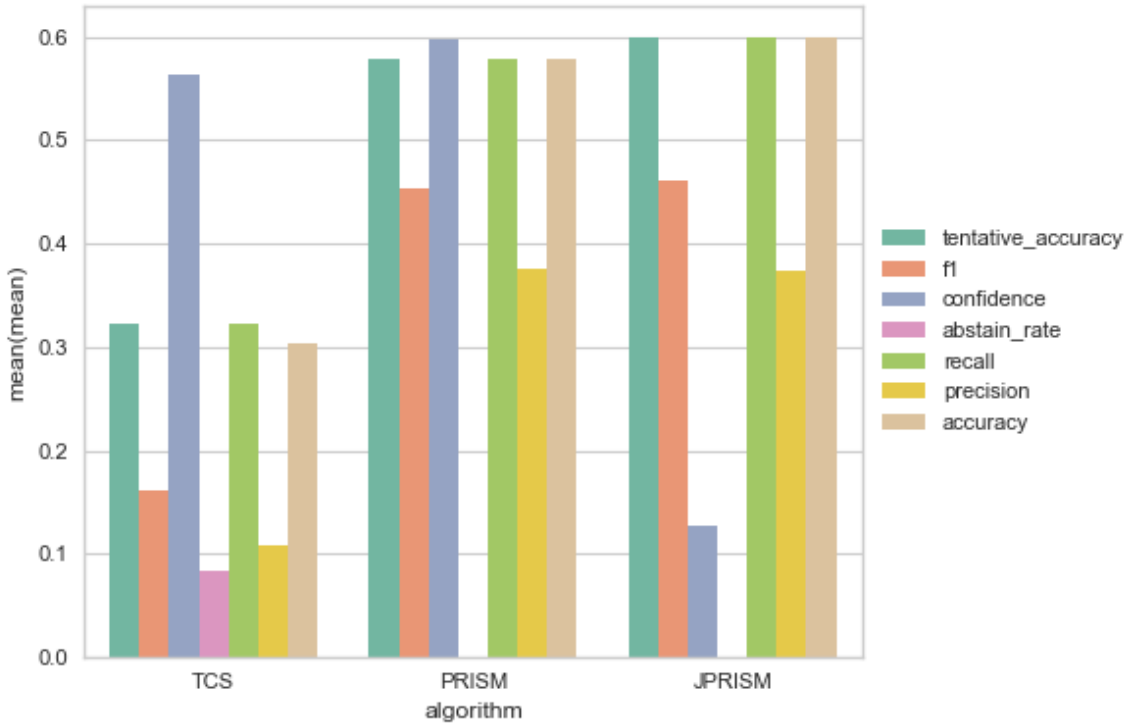

Figure A.2: Performance of Prism variants on the congressional data set, a two class problem

From these tests it can be seen that the performance of JPrism and Prism are very similar in both cases, PrismTCS under performs in both data sets. For the congressional data, Figure A.2, PRISM and JPrism are very close in precision based metrics, depending on the fold Prism may out perform each other. For the multivariate Cars data set in Figure A.1 there is a clear difference in precision between the two. This could be due to the strong influence the value $P(Y)$ has on the J-measure, JPrism performs poorly in cases when there is little variation between these values.