Major incident detection

MSc. Thesis Computer Science

Thomas Kolenbrander

Major incident detection

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

Thomas Kolenbrander

Software Engineering Research Group
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
www.ewi.tudelft.nl

AI for Fintech Research
ING Bank N.V.
Frankemaheerd 1
Amsterdam, the Netherlands
www.ing.nl

Major incident detection

Author: Thomas Kolenbrander
Student id: 4353137
Email: `t.j.kolenbrander@student.tudelft.nl`

Abstract

Incident management is one of the top priorities for IT companies. Within incident management the so-called major incidents, incidents with a severe impact on the company, require emergency actions to reduce this impact. An earlier detection of these major incidents will lead to a faster resolution time and this can be achieved by using software analytics methods. These methods have been used on incident management before which faces challenges with data quality and imbalance. However, software analytics have not been applied to major incidents in particular, which is what this thesis aims to do. To gain more insight into the possibilities and challenges of automated major incident detection, a case study at ING (a large global bank) was performed. For this case study a machine learning system has been created and assessed by eight experts through interviews. Following from these interviews, three novel challenges were identified. The first challenge is that the impact should be measured to make a more accurate prediction. The second challenge is combining multiple information sources and the third challenge is the explainability of the decision. Furthermore, two solutions to existing challenges were investigated during the creation of the machine learning system. The first being the suitability of different machine learning models for incident data, as no direct comparison is available in literature. It is shown that Logistic Regression is best suited for this use case while the Support Vector Machine and Neural Network also perform well on incident data. Finally, some findings on the pre-processing of the incident data are reported. It is shown that assumptions in literature about automatically generated incident data being easier to use, can not always be made and that imbalanced data still remains an unsolved problem as sampling is not suited. The main contribution of this thesis are the insights and challenges in the unexplored topic of major incident detection and general recommendations for handling incident data.

Thesis Committee:

Chair:	Prof. Dr. A. van Deursen, Faculty EEMCS, TU Delft
University supervisor:	Dr. S. Proksch, Faculty EEMCS, TU Delft
Committee Member:	Dr. D. Tax, Faculty EEMCS, TU Delft

Preface

First of all, I would like to thank my supervisor Sebastian Proksch for his support and the advice he gave me during my thesis. Even though you only joined me for the last few months you still gave it your all, for which I am very grateful. For the first part of the thesis I was guided by Georgios Gousios whom I would like to thank as well, for helping me define this topic and laying the foundation of the thesis.

I would also like to thank the people at ING for providing me the opportunity to carry out my research in industry. More specifically, I would like to thank Jerry Brons and Elvan Kula for their direct involvement in my thesis. Without them I would have been lost in the business world, luckily they were there to guide me. At ING I would also like to thank all members of the AI for Fintech Research lab for the inspirational talks and providing a community to discuss our research.

Finally, I would like to thank my friends and family for their unconditional support. For the much needed distraction or an insightful discussion, you were always there for me.

Thomas Kolenbrander
Delft, the Netherlands
February 10, 2021

Contents

Chapter 1

Introduction

IT Service management (ITSM), managing IT operations, is one of the top demanding factors that influences the revenue of companies [36]. One of the main processes in ITSM is incident management, the tracking and managing of all incidents. Research is being done on improving the incident management process as more efficient incident handling can reduce the losses caused by incidents. Examples of such improvements are retrieving similar incidents and automatic classification and routing of incidents. Within incident management there is the notion of 'major' incidents, these are the incidents that have a very severe impact on the business and should be resolved as soon as possible. It is therefore important to detect these major incidents as fast as possible to prevent or limit their (possible) impact. One way to do so is by applying software analytics methods to this problem. Software analytics tries to provide insightful and actionable information by performing data exploration and analysis [43]. A key tool for software analytics is machine learning, which will be applied in this research in order to aid the faster detection of major incidents.

The topic of major incident detection itself remains untouched in literature as most research regarding incidents is focused on incident classification, assigning a suitable category to an incident. The difference is in the extremely unbalanced and binary nature of the major incident detection problem. Since incident classification does require the same type of data, namely incident data, some of the insights gained are applicable to major incident detection as well. The first remark about the data is that there are two types of data: User generated and system generated. These should be processed differently but it is unclear what the exact effects of this distinction are, it is suspected that system generated data is easier to use. Another problem that arises with incident data is the fact that it can be unbalanced. This is especially true for major incident data as major incidents are rare occurrences.

Besides these difficult properties of the data, it is also unclear which machine learning model performs best on this type of data. A better understanding of the performance of machine learning models on incident data will lead to improvements in the whole field. While a lot of machine learning techniques have already been applied on incident data, there is a lack of direct comparison between them. It is therefore one of the goals of this research to determine which model is the best suited for incident data.

Research questions Based on the information provided above it is clear that there is research lacking in this topic. Following from this insight, the following research questions are formulated:

RQ1: What are the challenges in major incident detection? Since there is no research on detecting major incidents using historic data, the first question to answer would be on how such a system would work and which challenges it brings.

RQ2: Which machine learning models are best suited for incident data? There has been research using machine learning on incident data. However, due to the confidential nature of incident data it is not possible to make direct comparisons between the different models used. Answering this question will provide an overview of which machine learning techniques are best suited for incident data.

RQ3: What is the effect of splitting the data on its source? Previous research indicates two different sources for incidents: User generated and system generated. While it is speculated that splitting the data on these sources improves performance this has not been tested in practice.

RQ4: What is the best way to deal with data imbalance? Incident data is known to be imbalanced, even more so for major incidents, as they are rare occurrences. Some methods such as random oversampling and undersampling have been applied but a direct comparison is missing. The goal of this question is to provide a clear answer on the best to deal with imbalanced incident data.

Approach The first step is analyzing related work and determining which challenges are applicable to major incident detection as well. Following from this analysis a system to detect major incidents is designed. The system design will be implemented and solutions to the already identified challenges are evaluated. These solutions include determining the best machine learning model, the effect of splitting data on its source and dealing with data imbalance. During this implementation and evaluation new challenges that are found in practice will be investigated. The final step is to interview experts on the subject and evaluate their responses. Combining these three different perspectives (literature, the system in practice and experts) will give us a complete overview of the challenges in major incident detection.

Main Contributions Two different types of contributions have been made regarding major incident detection: Novel challenges were identified and solutions for existing challenges have been proposed. The novel challenges have been obtained by interviews with experts and are the following: Measuring impact, combining different data sources and the explainability of the decision. The first two to help in improving the performance of the system. The explainability is a requirement by users as it helps them in resolving major incidents. The solutions to existing challenges haven been evaluated by designing and implementing a major incident detection system. For this implementation several machine learning models have been compared using incident data. Following from this comparison

it is concluded that Logistic Regression, Support Vector Machine and Neural Networks are performing best on incident data. The Logistic Regression is best suited for major incident detection due to its high recall while the neural network might be suited for other use cases as there is still a lot of room for improvement. Imbalanced data is an already identified challenge and sampling was used to mitigate its effects. The results show that performance barely improved and more intelligent solutions are needed. Finally, the incidents were split on their source (user or system generated) and we conclude that automatically generated incidents are harder to classify than user generated incidents. This contradicts existing literature, opening up a new research opportunity.

Chapter 2

Background and Related Work

Online service systems such as online banking systems and e-commerce systems are increasingly popular and important in our society [26]. A growing number of people rely on these systems and their availability is therefore of the utmost importance. When a company fails to keep their system available this can have a large impact, both on their reputation and finances. An example of this impact is an Amazon AWS outage of 4 hours in 2017 which is estimated to have cost a total of $310 million dollars [39]. IT service management aims to ensure the availability of the services and minimize downtime. It is becoming one of the top demanding factors that influences the revenue of companies [36].

2.1 IT service management

IT Service Management (ITSM) is a discipline for managing IT operations [12]. ITSM is characterized by its emphasis on IT services, customers, service level agreements, and an IT department's handling of its daily activities through processes [17]. With ITSM being the concept of Service Management, there are several frameworks implementing these ideas in more specific guidelines. The most popular and industry standard [17] being the Information Technology Infrastructure Library (ITIL) [19].

”ITIL describes the best practices and approaches in IT Service Management, starting from strategy generation to the continual service improvements” [13]. The ITIL lifecycle is shown in Figure 2.1. The lifecycle is comprised of five different phases, which all interact with each other. Incident Management is part of the 'service operation' phase, which is defined as the stage that ”coordinates and carries out the activities and processes required to deliver and manage services at agreed levels to business users and customers” [15]. The Incident Management process focuses on tracking and managing all incidents, from opening until closure [37] and is one of the

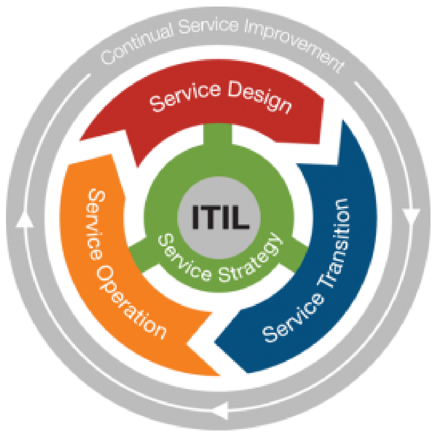

Figure 2.1: The ITIL v3 lifecycle
Source: *AXELOS, ITIL v3/2011*

main processes provided by ITIL [36]. It is even the most adopted service of ITIL with 95% of the companies using it in some form [28].

According to the ITIL framework an incident is defined as ”an unplanned interruption to an IT service – or a reduction in the quality of an IT service. The failure of a configuration item that has not yet had any effect on a service is also an incident” [1]. So situations that might lead to incidents should also be handled as incidents. The main goal of Incident Management is to restore normal service as quickly as possible. The resolution of incidents is usually started with the creation of a ticket (also called report) that contains information regarding the incident. This ticket is then assigned to a group responsible for solving the incident and reassigned or updated if necessary. Finally, the ticket is closed when the incident is solved [36]. The scope of incidents is rather broad as they can be reported by users, technical staff, third parties, automated monitoring tools or other processes. Some examples of incidents are: Network devices failing, applications showing error screens, printers not working or systems crashing. It is clear that some of these incidents require a more urgent response then others. For these incidents the concept of a ’major incident’ is introduced. Major incidents are incidents that have a very severe impact on the business. Thus, it differs from company to company how a major incident is defined. Major incidents will usually be actively managed by a team to ensure a fast resolution time.

2.2 Software analytics in incident management

The term software analytics was introduced by Zhang et al. [43] and defined as: ”enabling software practitioners to perform data exploration and analysis in order to obtain insightful and actionable information for data-driven tasks around software and services”. An important tool for software analytics is machine learning as it is able to learn hidden patterns or create predictive models from data.

Following from the ITIL framework every single incident must be recorded as these incident tickets are vital for trend analysis and reporting [1]. Incident tickets generally contain two types of data: Structured fields and free-text fields [42][22][8]. The structured fields contain data such as the date, reporting time and assigned resolver [27]. While these fields are used in most companies, some companies also include information such as the severity [5] and priority [40] as structured fields. The free-text fields often contain more detailed descriptions of the incident which can include affected IT components, symptoms, and diagnosis results [23]. These descriptions generally do not conform to certain formats [24] which makes them hard to process. Nevertheless, the free-text fields are considered to contain the most fundamental information of an incident [20].

In companies, a large number of tickets is created every day [37] thus providing a new and important data source for service management [26]. Due to the large amount of data being available in incident management it is a perfect target problem for software analytics research [26]. Two popular software analytics applications within incident management will be discussed in the sections below.

2.2.1 Known-incident association

Known-incident association tries to aid the incident resolution process by providing historically similar incidents. By leveraging the knowledge about solutions from past incidents we can help improve the effectiveness and efficiency of incident resolution [26]. A typical known-incident association system is comprised as follows: Once a new incident is presented, the system tries to retrieve similar cases. Problem features are extracted from the incident and the retrieval is performed based on a similarity computation between the incident and previous incidents [20].

The most fundamental information to retrieve similar incidents is the incident description [20]. By just using the incident description and filtering the words on a pre-defined dictionary Liu et al. [25] manage to achieve a reasonably positive result with a 78% success rate. More successful approaches also incorporate knowledge outside of the incident description such as transaction logs [9] and the hierarchical structure of the organisation [20].

2.2.2 Incident classification

Incident classification is the process that assigns a suitable category to an incident, so they are routed more accurately [37]. A more formal definition of incident classification is given by Maksai et al. [27]: ”incident ticket classification is a multi-class classification problem that aims at assigning the correct type to each ticket based on input features extracted from the ticket resolution text”.

One of the first attempts to create a system to automatically classify incidents was made in 2002 by Di Lucca et al. [7]. The first step is processing the textual description, which is done by removing stop-words and then using a stemmer to bring back each word to its radix and encoding the result using a bag-of-words approach. The data is then fed to one of the five machine learning models they evaluated. Namely: The vector space model, Naive Bayes, SVM, CART and k-nearest neighbours. Following from a case study with 6000 incidents each having (only) a 12 word description, the SVM classifier seemed to perform best at 84% accuracy, closely followed by k-nearest neighbours and the Naive Bayes model with respectively 82% and 80%. Due to the small dataset and short descriptions it is unsure whether these results still hold up today.

As part of their research on efficient incident management, Gupta et al. [14] also explore the possibility of automatic incident classification. They make an important distinction in the two types of incident tickets available, namely user reported and system generated tickets. Therefore, different techniques are required for their classification. A machine learning approach is deemed best suited for the user reported incidents while a rule-based approach is suggested for the system-generated incidents. The selected machine learning approach is comprised of a bag-of-words encoding combined with a Naive Bayesian classifier. Surprisingly, no other pre-processing or cleaning method is used. Their result of a 70% accuracy using 1000 features could thus possibly be improved by introducing cleaning as they already noted themselves that the textual description can be very ’dirty’ (using acronyms and service-specific expressions).

Xu et al. [42] use the same method of splitting the incidents tickets into these two groups but also incorporate extensive data cleaning and preparation. This step includes techniques such as stemming, stop-word removal, spelling correction, abbreviation normalization and the use of synonyms. According to them using domain specific knowledge is extremely important for accurate ticket classification. This can also be seen in the domain specific data preparation techniques used, which are: Keeping or discarding certain errors based on expert knowledge and a synonym library maintained by domain experts. The tickets are then represented by a bag-of-words for the system generated tickets and n-grams based on Part-Of-Speech (POS) tagging for the user generated tickets. For the classification a modified version of k-means clustering is used. The results show that in general the system generated tickets are easier to classify as they result in an F1-score of 0.93. The results for the user generated tickets were also very decent with an F1-score of 0.82.

A machine learning approach that has not been mentioned yet is the Gradient Boosting Machine (GBM), which is used by Bogojeska et al. [2]. They perform an extensive text cleaning by removing server names, punctuation, numbers, html formatting and more. The resulting text is then used in a bag-of-words model which is fed to the GBM. By doing so they are able to achieve great results of 94% and 91% accuracy on two different datasets. They do still identify some problems in incident classification as the data can be very unbalanced and there is a high variability in tickets.

The machine learning approaches discussed above all require a large amount of pre-classified data. Manually labeling this data can be a time-expensive task. To eliminate the need of pre-classified data Diao et al. [8] explore the idea of a fully rule-based classification system. While the biggest advantage is indeed not needing any labeled data, they are also able to show that well-constructed (simple) rules can still perform better than some supervised learning methods. It should be noted though that creating these rules for the system is also very time expensive and this could negate the advantage of not needing to manually label data.

Another approach to the data labeling problem is taken by Maksai et al. [27] where they try to reduce the labeling effort by intelligently selecting unlabeled incidents to be labeled. An hierarchical clustering approach is taken which is then also applied in the classification system. A very similar system was made by Roy et al. [35] using k-means clustering. The clustering approach makes use of the bag-of-words model for the incident descriptions. Their approach is compared to several other techniques such as Maximum Entropy (a modified Logistic Regression model), gradient boosting and clustering using Latent Dirichlet Allocation (LDA). The hierarchical clustering approach outperforms most of these techniques when there is only a small amount (100-150) of labeled tickets available. However, with a larger amount of labeled tickets the other approaches surpass the hierarchical clustering approach.

More recently, in 2018, Silva et al. [37] also focus on the incident classification problem using a purely machine learning based approach. Common pre-processing steps such as removing stop-words and stemming are applied first, which is followed by encoding the resulting text using TF-IDF. Two different machine learning models are evaluated, namely

Method	**Used by**
Keyword extraction	[32] [35]
Stop-word removal	[7] [37] [42]
Part-Of-Speech tagging	[35] [42]
Stemming	[7] [37] [42]
Filtering out numbers, names and formatting	[2] [42]
Spelling correction	[42]
Synonym dictionary	[42]
Abbreviation normalization	[42]
Error filtering	[42]

Table 2.1: Cleaning techniques

Support Vector Machines (SVM) and K-Nearest Neighbors (KNN). The SVM seemed to perform best with a 89% accuracy and this is supposedly caused by the fact that SVM is more suited for textual data. This is confirmed by their conclusion that the textual description of an incident is in fact the most important feature for classification.

Incident triage, automatically assigning an incident report to a suitable team [4], can also be seen as a form of incident classification where the classes are made up of teams instead of categories. One of the first attempts was based on keyword extraction from the incident report combined with a decision tree model [32]. While this already resulted in a 93% accuracy the keyword dictionary was hand-picked and thus not very generalizable and scalable to other datasets.

Chen et al. [4] take a different approach and try to solve this problem by applying bug triage techniques. Compared to bugs, incidents happen more frequently and also include automatically generated reports. Despite these differences, the bug triage techniques do perform relatively well on incident data. The best performing bug triage techniques were based on deep learning followed by Naive Bayes.

Cleaning the data is one of the first steps taken in incident classification. In the section above several different cleaning techniques have been discussed, which are all displayed in Table 2.1. We can see that stop-word removal and stemming seem to be the most popular cleaning techniques.

After the data has been cleaned, it needs to be encoded in order to be used in a model. In the reviewed literature only three different encoding seem to be used: Bag-of-words, TF-IDF and n-grams (see Table 2.2). Bag-of-words is by far the most popular. However, more recently word embedding is also becoming a popular encoding which still seems to be missing in incident classification.

The final step in incident classification is the actual classification. All classification techniques that have been used are shown in Table 2.3 and we can see that a wide range of techniques have been applied but clustering seems to be the most popular. Due to the lack of standard datasets and the confidential nature of the data it is very difficult to compare the

Method	Used by
Bag-of-words	[7] [14] [2] [27] [42]
TF-IDF	[37] [35] [7]
n-grams	[42]

Table 2.2: Encoding techniques

Method	Used by
Vector space model	[7]
Support Vector Machine (SVM)	[7] [37]
Classification and Regression Trees (CART)	[7]
Rule-based	[14] [8]
Naive Bayes	[7] [14] [4]
Gradient Boosting Machine (GBM)	[2] [27]
Logistic Regression	[27]
Clustering: k-nearest neighbours	[7] [37]
Clustering: hierarchical	[27]
Clustering: k-means	[35] [42]
Clustering using Latent Dirichlet Allocation (LDA)	[27]
Deep Learning	[4]
Decision Tree	[32]

Table 2.3: Models used

results of these techniques with each other. Two of our studies do use multiple techniques and they conclude that SVM is the best performing model for this type of data [7] [37]. In a literature study on ticket classification, Kubiak et al. [22] come to the same conclusion. Since the incident classification systems all make use of similar data which includes textual incident descriptions, we are able to identify some common challenges:

- **Different sources** In general incident tickets come from two different sources: User reported and system generated. These two sources require a different type of processing, where system generated tickets should be easier to classify. [14][27][22][42]
- **Data quality** The textual descriptions of the incidents can be of low quality. Especially the user reported tickets can contain acronyms [14], non-English words [42] and spelling and grammar errors [22].
- **High variability** There is a very high variability in the incident texts. This is due to a large vocabulary size and differences between teams [27]. The variability leads to sparse, high dimensionality data which is a big influence on performance [42].
- **Domain specific** The vocabulary used in incidents is very domain specific and it often requires expert knowledge to understand incident reports [22][27][22]. Due to this it is hard to create a generalizable solution to this problem.

- **Unbalanced** The number of incidents is very large and unevenly distributed over the different classes, leading to an unbalanced dataset [2][27].

2.3 Detection of major incidents

The detection of major incidents is a topic that remains untouched in literature. While incident classification is a similar problem (assigning a class to an incident), major incident detection differs due to its extremely unbalanced and binary nature. Another problem very similar to major incident detection is determining the priority of incidents, since a major incident is an incident with the highest priority. Determining the priority of incidents is called one of the very first steps in incident management [16] and is noted as future work [37].

Renners et al. [34] do try to solve this problem by prioritizing network security incidents based on their severity using model trees. While their results were reasonable they conclude that their approach was limited by the rather simplistic learner. Another factor influencing their results is the removal of free text fields, which are deemed 'unusable'. However, in Section 2.2.2 free text fields were considered one of the most important features in incident classification. The downside to using these free text fields is that they require extensive pre-processing.

2.4 Handling imbalanced data

Since major incidents are rare occurrences it is plausible to assume that the dataset will be unbalanced. As stated before, highly imbalanced ticket data is a big challenge for classification models [41]. In general there are two ways of dealing with imbalanced data: Undersampling and oversampling.

Undersampling works by removing data points from the larger class, which is the non-major incident class in our case. Data points are selected randomly and removed. Literature is contradictory on whether under-sampling improves the results as both positive [41] and negative [30] results are noted.

Oversampling reduces the imbalance by increasing the smaller class, so artificially creating more major incidents. There are two ways to do so: Randomly duplicating major incidents and generating 'new' major incidents, which is called artificial oversampling. The most established way of artificial oversampling is SMOTE [3] which generates new data points based on the interpolation of current data points. Oversampling using SMOTE has not been used on ticket data before and thus provides an interesting research opportunity.

2.5 Text classification

As seen in Section 2.2 quite some text classification techniques are used when dealing with incident data. The following section will provide background on and an explanation of these techniques.

2.5.1 Text encoding

The different textual encodings can be divided into two categories: Weighted words and word embeddings. The weighted words category contains methods that are all based on counting the words in some way. Word embeddings on the other hand are based on neural networks and transform each word to a vector with some semantic meaning. Word embeddings need a lot of data to be trained and are therefore only outperforming weighted words encodings when the dataset contains millions of entries [45]. Since previous research in the field of incident management was never performed on such large datasets we will be focusing our efforts towards weighted words.

Within weighted words, three methods have been used in the incident management domain before: Bag-of-words, n-grams and TF-IDF with bag-of-words being the most popular. The simplest technique is bag-of-words, which uses the frequency of each of the words as features. Both n-grams and TF-IDF are variations of this concept. N-grams uses combinations of terms and counts those, which preserves the order of terms which could be useful in some cases. The downside is the increased memory usage. TF-IDF uses single terms just like bag-of-words but divides them by their document frequency. By doing so stop-words get a lower score than words that are more unique and thus provide more meaning.

2.5.2 Machine learning models

The different machine learning models following from Table 2.3 will be discussed below to get a basic understanding on how they work. The rule-based model is left out as it is not related to machine learning and the vector space model is not discussed as it is not used in practice anymore.

Logistic Regression [38] Logistic Regression is a very popular and simple model which is why it is a great starting point for machine learning tasks. Logistic regression tries to fit a line in the data that separates the data best. It does so by making use of maximum likelihood.

Support Vector Machine (SVM) [6] As the name Support Vector Machine suggests, the model is based on support vectors. Support vectors are data-points close to the boundary of two classes. These support vectors are then used to construct a boundary line (or plane in case of more than 2 dimensions) equally afar from the support vectors to separate the classes. When a linear separation is not sufficient, support vector machines are also able to transform the data to a higher dimensionality and separate the data in that space. The model allows for several different functions (called kernels) to be used. The Support Vector Machine is a very robust machine learning model.

Naive Bayes [44] The Naive Bayes classifier is a classifier based on applying Bayes' theorem under the assumption that all features are independent. Naive Bayes is a very fast model but this comes with the downside that models are often over-simplified.

Decision Tree [33] A decision tree classifier tries to learn simple decision rules in order to classify the data. An example would be: ”if *shape* = *round* classify as orange”. Combining all these decision rules results in the final model. The model is very simple and easy to interpret. However, it is very prone to overfitting.

Gradient Boosting Machine (GBM) [10] In gradient boosting, more specifically gradient tree boosting, several decision tree models are built sequentially and they try to learn from the mistakes from the previous model. By doing so, several weak decision trees are able to produce a more powerful model together. Due to the sequential nature of this method, training can take a lot of time though.

Clustering There are many different algorithms available for clustering but they all try to divide the data into clusters where the data points are similar. It is especially useful in cases with multiple classes and unlabeled data. It is therefore not really applicable to the binary major incident detection problem.

Deep Learning Deep learning is a technique based on artificial neural networks inspired by the human brain. These networks are highly complex and can take notoriously long to train.

Chapter 3

System design

As stated in the approach a machine learning system will be created to classify new incidents as either major or non-major. The input to the system will be the created incident report at the time the incident is detected. Based on this report the system should decide if the submitted incident is major. The system will make this decision based on a machine learning model that is trained on past incidents. During this training the system will learn common properties in major incidents and use these to predict if a new incident is major. A simplistic design of the system is given in Figure 3.1. There are two different inputs to the system, namely past incidents, which are used for training, and new incidents which should be classified as either major or non-major. The ’filter’ step is only applied on the past incidents, while the ’clean’ and ’encode’ step are used by both and will be further explained in Section 3.1. The resulting encoded past incidents are used to train the machine learning model which will ultimately decide whether the new incident is major. The selection of the machine learning model is discussed in Section 3.2. Finally, the details on the implementation are given in Section 3.3.

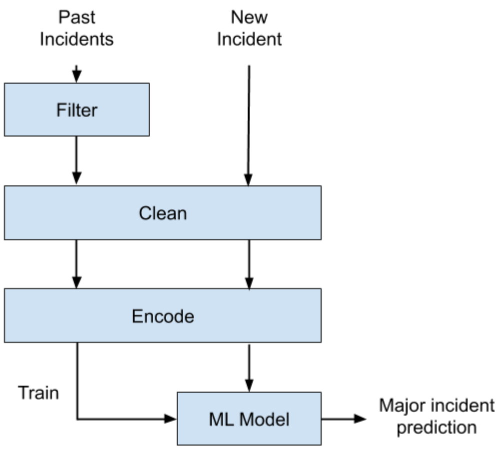

Figure 3.1: The design of the system

3.1 Pre-processing

As discussed in the previous chapter, the textual description of an incident is the most important factor for its classification. It is therefore important that we include the textual description in our system as well. Since the textual description requires the most pre-processing (due to its unstructured nature) this section will mostly be focusing on how this data is handled. However, other features that are available should be included as well and processing those will be discussed in the subsection on encoding.

3.1.1 Filtering

The filter step is only applied on the historic incident dataset as can be seen in Figure 3.1. This step is needed to ensure the quality of the training data. Since the description of an incident is one of the key components of our system, we need to make sure all of our descriptions are in the same language. The language of course depends on the company, but in a large enterprise setting we can assume this will be English. Thus, we only keep the incidents for which we can detect that the description is in English.

Furthermore, incident reports with missing data also need to be handled. When the description of an incident is missing, the incident will be removed from the dataset as the description is one of the most important features. For other attributes it depends on the value that is missing. For example with a feature like 'Possible cause' a default value of 'Unknown' can be defined when the data is missing. However, for something like 'External customer impact' where the value can either be true or false we cannot infer the right value. In cases where the right value cannot be inferred the incident should be removed from the dataset. Once again, this is done to ensure the quality of the dataset.

3.1.2 Cleaning

The cleaning process will be the same for both the past incidents and the new incident. As stated in the related work section, the data quality of textual descriptions can be rather low. This can be caused by spelling and grammar errors, acronyms or non-English words. A common first step is to remove all non-alphanumeric characters as their meaning is hard to interpret. After removing the non-alphanumeric characters all numbers still remain. A word that is a combination of both letters and numbers (which we will call a 'hash') is not a regular English word and it will thus very hard to determine its meaning. Therefore, we will replace any words containing a number with an identifier to signal to our system that it is a hash. By doing so the information that there is a hash in the description does not get lost but we do not need to worry about its meaning. The same holds true for numbers, which are also replaced by a general identifier. The final step is to transform all capital letters to their lowercase equivalents to make sure that words get recognized as being the same despite capitalization.

Another aspect that should not be overlooked is the high variability in the texts. The very large vocabulary, due to the domain specific terms, is a problem as it will lead to high dimensionality data, which has a big influence on the performance of machine learning models [42]. A solution to this problem is to remove words that occur too infrequent. When some words only occur a few times in the whole dataset we cannot draw conclusions based on these few occurrences and it is therefore better if they are removed from the text. The same holds true for stop-words which are words that occur too often. An example of this would be the word 'the'. Since it is very likely to occur in almost every English text it is better to remove these stop-words as they do not provide any value. Removing all these words lowers the variability which in turn should increase the performance of the machine learning model.

The final cleaning step that will be performed is stemming as it is one of the most popular techniques and helps reducing the variability of the data even more. Stemming is able to reduce the variability by identifying the base word (or stem) for a particular word. An example would be the word ”trees”, which is stemmed to its singular form ”tree”. A more difficult example is ”walking” and ”walked” which would both be stemmed to ”walk”. By using stemming the variability in words decreases which should increase the performance once again.

3.1.3 Encoding

The next step in the system is to encode the data so that it can be used by a machine learning model. For this step we distinguish three different types of data: Numerical, categorical and textual. The numerical data does not need any specific encoding and is ready to be used.

For categorical data there are two options regarding encoding: Integer and one-hot encoding. With integer encoding each category gets replaced with an integer corresponding to that category. The downside of integer encoding is that the relations between the integers are mapped to the relation of the categories. Take for example the category 'Environment' which can take the values 'Production', 'Testing' and 'Monitoring' as displayed in Table 3.1. These can be converted to the respective integers 1, 2 and 3. When interpreting these integers it looks like that production (1) is closer to testing (2) than to monitoring (3). However, in reality this might not be the case. The problem is that integers imply some natural ordering in the categories. To circumvent this problem we can use one-hot-encoding where each category is assigned its own column. When the incident is in the respective category a '1' will be placed in the column as is shown in Table 3.1. Since we do not have the problem of possibly unwanted relations between the categories, one-hot encoding will be used for the categorical data.

The different textual encodings have already been discussed in Section 2.5.1 and now a decision on which one to use will be made. Since n-grams have only been applied once before and are known to use a lot of memory, they will be discarded. Bag-of-words has the advantage of being the simplest which is also its biggest drawback. It is very easy to use but the representation lacks context. Context does get provided by TF-IDF as the term document frequency is taken into account as well. Therefore, TF-IDF will be the textual encoding of choice for the system.

	Original data	**Integer encoding**	**One-hot encoding**		
	Category	Category integer	Production environment	Testing environment	Monitoring environment
Incident 1	Production	1	1	0	0
Incident 2	Testing	2	0	1	0
Incident 3	Monitoring	3	0	0	1
Incident 4	Production	1	1	0	0
Incident 5	Monitoring	3	0	0	1

Table 3.1: Example of categorical encodings

3.2 Machine learning model

As discussed in Chapter 2 a variety of machine learning methods have been used in the incident management domain. Due to a lack of comparison it is unclear which performs best. However, the few instances where a comparison was made showed that the Support Vector Machine (SVM) performs the best. Since the SVM is particularly suited for text data, which we consider the most important data source, we choose to adopt the SVM as the initial model in our system. However, the model is easily exchangeable and which model performs best is still very much dependant on the dataset [45] and thus other machine learning models will be evaluated as well.

Training The dataset is split into two different datasets, one is used for training the model and the other is used for evaluating it. This is done to make sure that the learned parameters are also applicable to new data. The split of the data is performed randomly where 75% of the data is selected as training data and the remaining 25% is used as validation data. For any hyper optimization, such as is needed for the Support Vector Machine, no validation data will be used in order to prevent overfitting. Instead, the training data is splitted again where 75% will still be training data and the remaining 25% will be test data. The results on this dataset are then used to make a decision on the parameters.

3.3 Implementation

The system is built using Python[1] (version 3.8.3 64-bit) due to the wide range of machine learning libraries available. One of those libraries is scikit-learn [31]. Scikit-learn is considered the industry standard due to its excellent documentation and wide availability of algorithms. Therefore, scikit-learn (version 0.23.2) will be used in the system for implementing the machine learning models. All data is represented using pandas (version 1.0.5) dataframes as it offers extensive functionality and is compatible with scikit-learn. The filtering on language is implemented using the langdetect[2] library.

[1] https://www.python.org/

[2] https://pypi.org/project/langdetect/

Class weights Some of the machine learning models in scikit-learn allow the use of class weights. These class weights are able to help with the use of imbalanced data, which is the case for our type of data. Each class gets assigned a weight which will increase or decrease the cost of wrongly classifying data points in that class. In the case of unbalanced data that would mean that the majority class (the 'normal' incidents) gets a lower weight than the minority class (the 'major' incidents). By doing so the model is favouring the correct classification of major incidents. It is possible to specify these class weights yourself or use the 'balanced' function which adjusts weights inversely proportional to class frequencies. This function is implemented by scikit-learn and inspired by the weighting used by King et al. [21]. The following formula is used:

$$classweight = \frac{\text{\#samples}}{\text{\#classes * \#samples in class}} \tag{3.1}$$

For our dataset this means the following class weights are considered 'balanced':

$majority = 2,484,111/(2*2483078) = 0.5$
$minority = 2,484,111/(2*1032) = 1203.5$

Sampling The imbalanced-learn[3] (version 0.7.0) library is used to implement sampling as it offers both random over- and undersampling as well as artificial oversampling. The library allows the user to define a certain percentage and will then use the sampling technique specified until the minority class takes up that percentage in the full dataset. This allows for a very precise control over the sampling process.

Neural network While scikit-learn offers some simple neural networks, these are not intended for large-scale applications as no GPU support is offered. Since we have no GPU available the scikit-learn neural networks will have to suffice. Scikit-learn offers just one neural network classifier, the Multi-layer Perceptron classifier, which will be used as our neural network.

[3]`https://imbalanced-learn.readthedocs.io/en/stable/index.html`

Chapter 4

Case study

4.1 Context of ING

ING is a global bank with a strong European base and around 38.4 million customers in over 40 countries [18]. It has a strong online presence and large ecosystem regarding its applications which makes it a suitable target for research about incidents.

4.1.1 Incident management

Within ING an incident is defined as ”an unplanned interruption of the availability of a service or reduction in the quality of a service”. The purpose of incident management is to restore normal service operation as quickly as possible. The incident management process can be triggered in many ways:

- a user contacts the Service Desk through phone or e-mail
- a user completes a web-based incident-logging screen
- incidents are raised automatically by tools
- squads may notice potential failures themselves and raise an incident
- suppliers send some form of notification of a potential or actual difficulty.

Incidents thus come from a wide range of sources and are very diverse in their nature. Each incident is labeled with an appropriate prioritization code based on the estimated impact and urgency of the incident.

4.1.2 Major incidents

Incidents that are labelled with the highest prioritization code have the possibility of becoming a ’major’ incident, meaning that emergency action is required. A major incident is an incident with major impact on ING end customers or critical internal business processes and services. Major impact is defined as one of the following three consequences:

- Negative (media) attention
- The continuity of the primary business operation is or will be at risk
- Significant financial damages

The goal of ING is to minimize the impact of these incidents. There are three ways ING tries to do so:

1. **Prevention**: Prevent major incidents from happening again by investigating the cause of major incidents and fixing the underlying problem.
2. **Mitigation**: Avoid that regular incidents turn into major incidents.
3. **Reduce mean time to repair**: When major incidents happen, they should be resolved as fast as possible.

We will be focusing on the third item as a faster detection of major incidents means that process of resolving the incident can start earlier. Which in turn reduces the mean time to repair.

Process A squad is required to report all of their priority one incidents (the highest priority) to one of the two different roles within incident management. The decision whether an incident is major or a regular priority one incident is made by the incident management based on the available information. If the incident management is in doubt whether an incident meets the criteria of a major incident, the responsible stakeholders will be consulted. When it is decided the incident is major, the local stakeholders are informed and requested to report their impact immediately. The final step is to decide whether the major incident has local (one country) or global (multiple countries) impact. Depending on the outcome Global Major Incident Management might be involved as well.

4.2 Data

Within ING all incident management is done with an incident management system that provides technical management support. All incident reports must be registered with the platform and are also available in a SQL database for further analysis. The incidents available in the SQL database will be used to test and evaluate the system. The data ranges from November 2016 till July 2020. Incidents get updated over time as well but for this dataset only the first available versions were selected, by doing so the detection can happen in the earliest stage of an incident. The first step was to filter the incidents and only keep the incidents in English, as described in Section 3.1.1. This provided the advantage of removing incidents with an empty description as well, thus improving the quality of the dataset. After filtering, a total of 2,484,110 incidents remained of which 1032 were major incidents, which means that the dataset is very imbalanced.

As has been seen in literature before [14][27][22][42], the dataset contains both user generated and system generated incidents. According to this literature, splitting the data on this property increases results, mostly for the system generated incidents as their semi-structured text data should be easier to use for the machine learning model.

4.2.1 Feature selection

When reporting incidents several attributes need to be filled in. Some attributes have to be discarded as they are added after the incident, privacy sensitive or simply not available. Attributes that have more than 50% missing values are removed as well. Finally, attributes with a high cardinality, meaning that there are a lot of categories (a threshold of 50 is used) are removed as well. When using one-hot encoding, as described in Section 3.1.3, this will generate a lot of different features and this high dimensionality has a big impact on performance.

Included attributes

The following list shows the attributes which are included and will hence be used by the system.

Id The id will not be used by the model itself. However, when debugging the system it is easy to have a reference back to the incident.

Contact type The contact type is a categorical feature which describes how the incident was reported. There are six different categories: Event, e-mail, phone , self-service, walk-in and chat. While the majority of the contact types are in the event category, this feature will still be used in the system. It might for example be that more urgent incidents get reported in a different than usual way and this feature might thus be useful.

Notify by The notify by category describes how the reporter of the incident wants to be notified of changes. This attribute will also be used in the system based on the idea that more urgent incidents require different communication channels, similar to the 'contact type' feature.

Environment The environment describes the environment in which the incident occurs. A total number of 8 environments are included in the dataset with 'production' being the majority. Whether something happens in the production or testing environment makes a huge difference in the impact on customer and will thus definitely be included in the system.

Problem candidate The problem candidate states whether a follow-up by problem management is needed. This attribute is false in a large majority of all incidents which makes it look like a rather uninteresting attribute. However, for major incidents this value is 30% more likely to be true which means that this attribute can actually be used to distinguish between regular and major incidents and it is therefore included.

Description The description provides a full overview of the incident and its symptoms. According to literature the description is one of the most important features and therefore included.

Short description The short description is a clear and complete summary of the incident. For the short description the same holds as for the ’regular’ description. The short description might even be more valuable as users are forced to only write down the most essential information, leading to less irrelevant information

Incident state The possible incident states are ’new’, ’work in progress’, ’hand-over pending’, ’resolved’ and ’closed’. We are using the first reported version of the incidents but we might still encounter ’closed’ tickets as tickets can get created after the incident for administrative purposes. This information will be used by the system as closed tickets will very likely not be major incidents.

Impact - Urgency - Priority These three attributes provide an overview of the scale of the incident which is important information. Incident are rated on all three of these attributes.

Major incident This is the target attribute the system needs to learn.

Security Whether an incident is related to security or not could be impacting if it is a major incident and is included.

Incident escalated Incident escalation indicates a more serious incident which means it will be useful for the system.

External customer impact As described in Section 4.1 customer impact is one of the factors in determining the major status. It is expected that this will be a very meaningful feature.

4.2.2 Pre-processing

One of the goals of pre-processing is to reduce the number of features (the variability) to improve performance of the machine learning model. In order to do so the pre-processing steps, as described in Section 3.1, have been applied. This first resulted in a dataset with a total of 107,110 features mostly coming from the TF-IDF that has been applied to the Description and Short Description (80,635 and 26,422 features respectively). As stemming was already applied the feature space was reduced by removing infrequent and stop-words. The stop-words were removed if they were found in more than 20% of the descriptions and infrequent words were removed if they appeared less than 10 times. More strict parameters resulted in loss of performance. By doing so the number of features was reduced to 69,222 (51,308 and 17,861 for the Description and Short Description respectively) which is still a relatively large number.

4.3 Evaluation

To evaluate the system the data is randomly split into a training and validation set. The training set contains 75% of all incidents whereas the validation set contains 25%. Metrics also need to be defined in order to say something meaningful about the performance of the system. The most obvious metric would be the accuracy, the percentage of incidents that are assigned to the right class. However, due to the extreme imbalance of the dataset this would not be the right metric. For example, if 1% of the incidents is a major the system could achieve a 99% accuracy by predicting that none of the incidents are major. So while this would look like a very impressive system in theory, in practice it is basically useless as not even one major incident is detected. Two metrics that are more useful are the recall and the precision of the predicted major incidents. The recall quantifies how many major incidents could be detected and the precision quantifies how many of the detected major incidents are indeed major incidents. These metrics can be combined in the so-called F-score as shown in Equation 4.3. The factor β can be used to either favor the recall, the precision or weigh both equally. A factor of $\beta = 1$ is called the F1-score and a standard performance measure weighing both the recall and precision equally. As this F1-score is the most common in literature we will use it evaluate our system as well. In the case where two models have a comparable F1-score the model with a higher recall is preferred as missing a major incident is very costly.

$$Precision(P) = \frac{\text{\#Major incidents classified as such}}{\text{\#Incidents classified as major}} \tag{4.1}$$

$$Recall(R) = \frac{\text{\#Major incidents classified as such}}{\text{\#Major incidents}} \tag{4.2}$$

$$F_\beta = (1+\beta^2) * \frac{P * R}{(\beta^2 * P) + R} \tag{4.3}$$

4.3.1 Machine learning models

This section will provide an overview of the different machine learning models used. First, all models are evaluated individually and their best performing parameters are identified. To make a fair comparison, each of the models will use these optimized parameters in the final evaluation to decide which model performs best.

Logistic Regression The Logistic Regression implementation of scikit-learn allows the use of class weights. To asses the impact of using different class weights the following class weights will be evaluated: No class weights, 0.9 to majority class and 10 to minority class, 0.8 to majority class and 100 to minority class and a 'balanced' weight set as defined in Section 3.3. The comparison of the different class weights can be seen in Figure 4.1. In general, lower class weights seem to favour the precision while larger class weights seem to favour the recall. Both none and 0.0: 0.9, 1.0:10 have very similar F-scores while the others are clearly performing worse. As stated in Section 4.3, in the event of a similar F1-score a higher recall is preferred. In this case the class weight set 0.0: 0.9, 1.0:10 has the higher recall score and will thus be the preferred set of class weights which will be used for further evaluation.

There are two different loss functions available for Logistic Regression: *L1* and *L2*. Besides that there is also the option to use no loss function and to combine both of the functions,

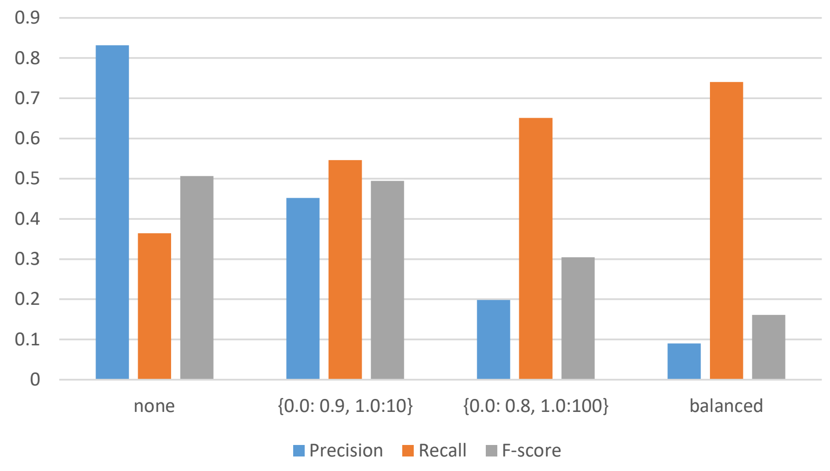

Figure 4.1: Comparison of the class weights for Logistic Regression

Loss function	**Precision**	**Recall**	**F-score**
none	0.26	0.33	0.29
L1	0.43	**0.57**	0.49
L2	0.45	0.55	0.49
elasticnet	**0.49**	0.56	**0.52**

Table 4.1: Performance of loss functions for logistic regression

this combined loss function is called ’elasticnet’. All of these options are evaluated in order to determine which loss function performs best. The ’elasticnet’ function is able to favor either *L1* or *L2* but we choose to experiment with an equal balance between the two. The results of these experiments are shown in Table 4.1. The highest F1-score is achieved by ’elasticnet’ just like the highest precision. The best recall is achieved by the *L1* loss function but ’elasticnet’ is a very close second. The winner of this comparison is the ’elasticnet’ loss functions due to its clearly higher F1-score.

Support Vector Machine The Support Vector Machine model supports several kernels and the first step is to decide which one to use. Since this is part of the hyperparameter optimization process, the validation data is not used, as described in Section 3.2. The results of this evaluation are shown in Table 4.2. As can be seen, both the ’linear’ and ’poly’ kernels seem to outperform the others based on the F1-score. Since a higher recall is more favourable, the ’linear’ kernel will be used for the Support Vector Machine.

Besides the kernel, the ’C’ parameter is also an important parameter that needs to be tuned in the hyperparameter optimization process. The results of experimentation with this parameter are shown in Figure 4.2. When looking at the graphs it is clear that a higher value

Kernel	**Precision**	**Recall**	**F-score**
linear	0.64	0.35	**0.45**
poly	**0.80**	**0.30**	0.44
rbf	0.79	0.26	0.39
sigmoid	0.76	0.08	0.15

Table 4.2: Comparison of different kernels for the SVM

Criterion	**Precision**	**Recall**	F-**score**
gini	0.44	0.36	0.40
entropy	**0.46**	**0.43**	**0.45**

Table 4.3: Comparison of criterion for Decision Tree

of C leads to a lower precision. The recall seems to increase as C increases but hits its peak at a C value of 30 and starts decreasing after that. The F1-score achieves the highest scores at C values of 1 and 3. As we favour recall, and recall increases with C, a C value of 3 is chosen.

Similar to Logistic Regression, the scikit-learn implementation of the Support Vector Machine also allows the use of class weights. The same class weights will be evaluated as for Logistic Regression: No class weights, 0.9 to majority class and 10 to minority class, 0.8 to majority class and 100 to minority class and a 'balanced' weight set as defined in Section 3.3. The comparison of the different class weights can be seen in Figure 4.3. The class weights do have a large effect on the precision of the model, the recall however does not seem to be affected that much. Using no class weights achieves the highest F1-score and will therefore be used in further experiments using the Support Vector Machine.

Decision Tree The Decision Tree allows for two different functions to measure the quality of the generated split: 'Gini' and 'entropy'. Gini is based on the Gini impurity while 'entropy' is based on information gain. Comparing both of these functions in Table 4.3 shows that 'entropy' is performing the best on all of the metrics. With the knowledge that the 'entropy' criterion performs best it is now time to determine the class weights. The same class weights will be used as seen before when evaluating the Support Vector Machine and Logistic Regression models: No class weights, 0.9 to majority class and 10 to minority class, 0.8 to majority class and 100 to minority class and the 'balanced' weight set. Results of this evaluation are shown in Figure 4.4. Both none and 0.0: 0.9, 1.0:10 achieve very similar F-scores with 0.0: 0.9, 1.0:10 scoring better on the recall. The class weight set 0.0: 0.9, 1.0:10 is therefore chosen for further comparison.

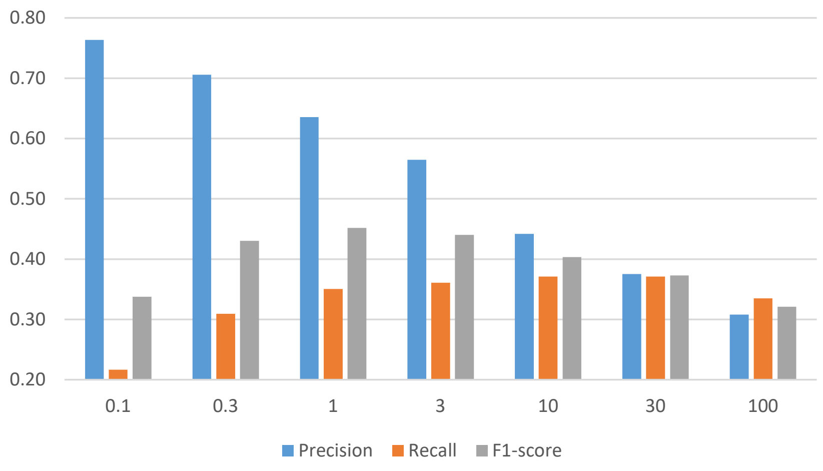

Figure 4.2: Experimentation with the C parameter for the linear SVM model

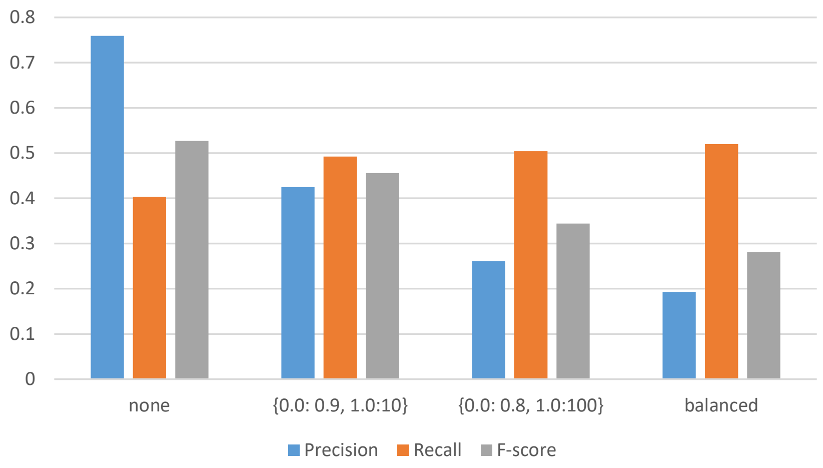

Figure 4.3: Comparison of the class weights for SVM

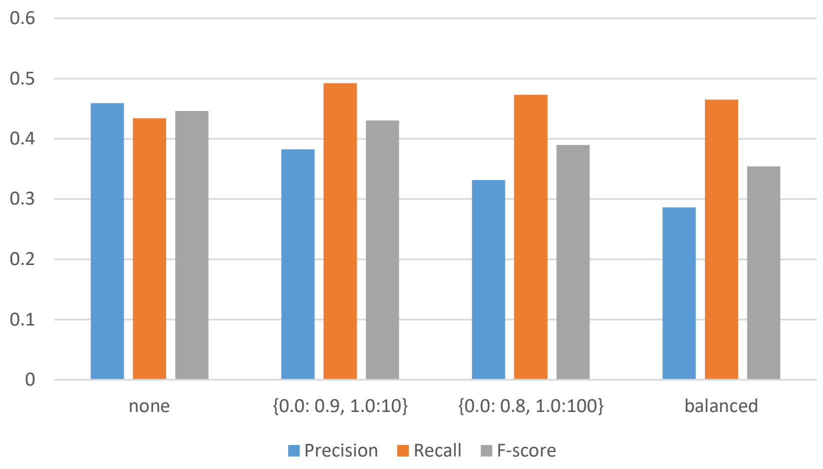

Figure 4.4: Comparison of the class weights for Decision Tree

GradientBoost GradientBoost is able to optimize for two different loss functions: Deviance and exponential. The exponential loss function is very sensitive to mislabeled examples and thus performs best on a high quality dataset. Since the quality of a dataset containing natural language is rather low, which is the case for our dataset, we do not expect great performance using the exponential loss function. This expectation is confirmed as the GradientBoost model using the exponential loss function was not able to correctly classify any major incident. The deviance loss function was better suited and its results are shown in Table 4.4.

Multi-Layer Perceptron As described in Section 3.3 the Multi-Layer Perceptron (MLP) is the chosen neural network. The network is initialized with one hidden layer containing 30 neurons. While this is a relatively small network (compared to state of the art models) the results are still very much comparable to the other models as can be seen in Table 4.5. No other network configurations were evaluated due to the run-time of this small network already being over 10 hours.

Naive Bayes For the Naive Bayes (NB) model the Multinomial NB and the Complement NB (an adaption of Multinomial NB) algorithms are selected. The initial results showed that the models were not fit for handling imbalanced data and therefore oversampling was applied. As can be seen in Table 4.6 the resulting F-scores are rather low. It is interesting to see that Complement NB scores even worse than Multinomial NB as it has been particularly adapted for imbalanced data sets. Due to the low scores achieved by NB it will not be considered a realistic option and thus not be included in the comparison.

Precision	**Recall**	**F-score**
0.26	0.17	0.21

Table 4.4: Results of GradientBoost with the deviance loss function

Precision	**Recall**	**F-score**
0.73	0.41	0.53

Table 4.5: Results of the MLP Classifier

Model	**Oversampling**	**Precision**	**Recall**	**F-score**
Multinomial NB	none	0	0	0
	0.1%	0.063	0.0039	0.0073
	1%	0.066	0.66	0.11
	10%	0.028	0.77	0.054
Complement NB	none	0	0	0
	0.1%	0.14	0.054	0.078
	1%	0.017	0.80	0.034
	10%	0.014	0.85	0.028

Table 4.6: Results of using Naive Bayes

	Precision	Recall	F-score
No oversampling	1.2E-05	0.012	2.4E-05
Oversampling 0.1%	1.2E-05	0.012	2.4E-05
Oversampling 1%	6.9E-04	0.99	0.0014

Table 4.7: Results of KMeans clustering

Clustering Even though clustering is best suited for multi-class problems in an unsupervised context (as opposed to the binary supervised problem of major incident detection) its performance was still evaluated as it is a popular model for incident classification. As clustering algorithm the popular KMeans algorithm was selected as implemented by scikit-learn. The algorithm did have a lot of trouble with the imbalanced data and therefore random oversampling was applied. More in-depth discussion about oversampling can be found in Section 4.3.3. The results of this experiment are shown in Table 4.7 and show that clustering, more specifically the KMeans algorithm, is not suited for the problem of major incident detection.

Comparison The best performing configurations of the different machine learning models are displayed in Figure 4.5. The models which achieve the highest F-scores are Logistic Regression, Support Vector Machine and the Multi-Layer Perceptron classifier. Since the Logistic Regression has the clear advantage on the recall, we consider this model the best suited model for major incident detection. The runtimes, the time it took to train the model, are included as well in Table 4.8. The model that was trained the fastest was the Support Vector Machine with roughly 3 minutes while the MLP Classifier took over 10 hours to train. Due to this extensive training time, the MLP classifier is not used for the further experiments involving pre-processing, different ticket types and sampling.

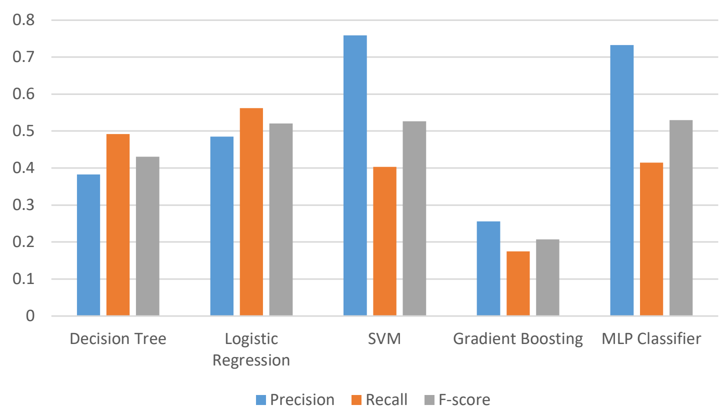

Figure 4.5: Comparison of the machine learning models performance

Model	Runtime (s)
Decision Tree	403
Logistic Regression	6391
SVM	189
Gradient Boosting	6584
MLP Classifier	37781

Table 4.8: Comparison of the runtime

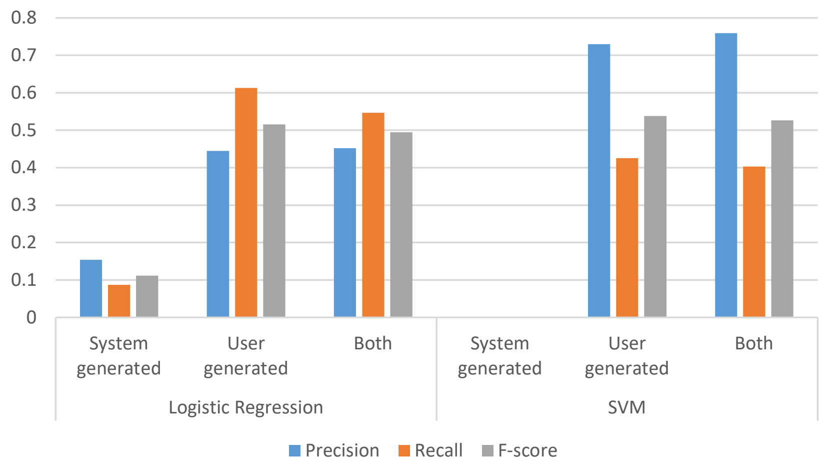

Figure 4.6: Comparison of user and system generated incidents

4.3.2 User and system generated incidents

The user and system generated incidents are split and the models are trained and evaluated on just one type of incident (so either system or user generated). The results using Logistic Regression and the Support Vector Machine are shown in Figure 4.6. It can be observed that the user generated incidents perform way better then the system generated incidents. While this is the case for both the Logistic Regression and Support Vector Machine it is especially evident in the latter as not even one system generated major incident gets classified correctly.

4.3.3 Data imbalance

As discussed in Section 2.4 on data imbalance there are two unknown factors with sampling: The effects of undersampling and artificial oversampling. To ensure that the effects are not only model related, the two best performing models (F-score wise) are selected: Logistic Regression and the Support Vector Machine.

The results of using undersampling are shown in Figure 4.7. The percentage in the second column represents the presence of the minority class (the major incidents) after sampling. So for a value of 1% for example, sampling is applied until the minority class makes up a total of 1% of the dataset.

By using undersampling the precision seems to decrease while the recall increases. The first step in undersampling with 0.1% achieves a similar F-score as using no undersampling for the Support Vector Machine and an even higher score for Support Vector Machine. After this step the F-score keeps steadily decreasing as more undersampling is applied.

The oversampling results can be found in Table 4.9. Both random and SMOTE oversampling are included in order to make a comparison between the two. In general, random and SMOTE oversampling seem to perform really similar. Only the first two steps with 0.1%

and 0.3% oversampling improve the F-score of Logistic Regression with further oversampling only worsening results. For the Support Vector Machine oversampling is never able to increase the F-score.

In general, the improvements achieved by sampling are very minor. Which means that sampling is not a sufficient solution for the data imbalance and it still remains a challenge for incident data.

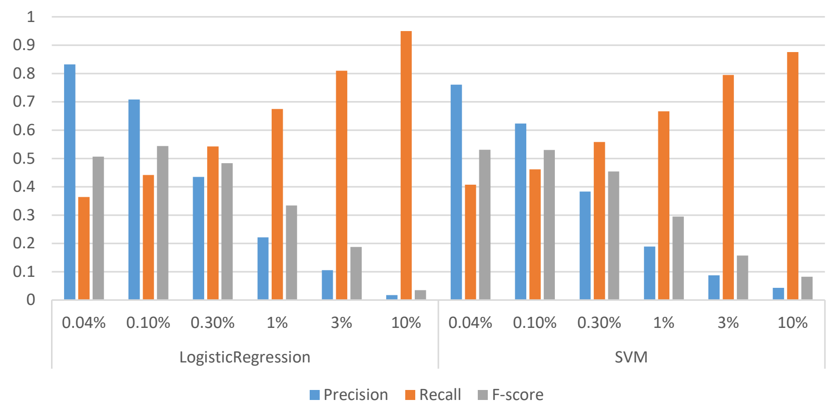

Figure 4.7: Performance of undersampling

		Logistic Regression			Support Vector Machine		
		Precision	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**
None	±0.04%	0.83	0.36	0.51	0.76	0.41	0.53
Random	0.10%	-0.11	0.07	0.03	-0.14	0.05	0.00
	0.30%	-0.30	0.16	0.02	-0.38	0.15	-0.08
	1%	-0.47	0.22	-0.06	-0.57	0.26	-0.24
	3%	-0.59	0.26	-0.16	-0.67	0.39	-0.37
	10%	-0.66	0.29	-0.23	-0.72	0.47	-0.45
SMOTE	0.10%	-0.11	0.08	0.04	-0.12	0.02	-0.02
	0.30%	-0.29	0.17	0.03	-0.29	0.08	-0.05
	1%	-0.48	0.21	-0.07	-0.39	0.09	-0.11
	3%	-0.58	0.24	-0.16	-0.48	0.09	-0.17
	10%	-0.65	0.28	-0.22	-0.51	0.09	-0.20

Table 4.9: Performance of oversampling compared to baseline (none)

Chapter 5

Involving expert knowledge

The next step in the process is to interview experts on the subject. The decision whether an incident is major is still very much a manual process where a lot of human interaction is needed. Therefore, to improve the system and identify challenges, we should also try to incorporate the knowledge and experience from the people making these decisions. We will do so by conducting exploratory semi-structured interviews with people involved in the major incident management process. First the set-up of the interviews will be discussed followed by the findings and its implications for the system.

5.1 Interview set-up

The interview is aimed at the people involved in the major incident management process and other processes related to it. The interview guide containing the questions and the interviewee selection process will be discussed below.

5.1.1 Interview guide

The interview guide contains a number of questions that should be answered during the interviews. Due to the semi-structured design of the interview, the order of the questions in the guide does not necessarily need to be followed throughout the interview. When a question naturally arises earlier in the interview than stated in the interview guide it is asked directly. There is also the possibility to further explore topics that are not included in the guide but are deemed relevant to the study.

The main goal of the interviews is to gain insight into the process of deciding when something is a major incident and thus the majority of the question will be focused on this topic. The interview guide is roughly divided into four segments:

1. Introductory questions: These questions allow the interviewee to introduce himself and provide context for the answers given.

2. Main questions: These are the questions related to the decision when an incident should be considered major.

3. Data quality questions: The questions in this segment are asked to gather more knowledge about the available dataset.

4. Process related questions: These question are about the major incident management process itself.

The full interview guide is available in Appendix A. The guide has also been translated into Dutch as a majority of the employees at ING are Dutch and might prefer the interview to be in their first language.

5.1.2 Interviewee selection

Interviewees were selected based on chain referral [29] where each interviewee is asked whether they know others that could provide information. The process was started by contacting one of the main figures in the global major incident management process and the first round started with interviewing his recommendations. With the selection of interviewees the goal was to explore different roles and domains in order to get a diverse view of the process. According to the ING major incident management process there are two roles responsible for the decision on declaring an incident major. These two roles are therefore deemed essential to this research and must be included.

The sample size was based on the concept of saturation which is reached when collecting more data produces little important new information or understanding [29]. The first round of 3 interviews did not provide a sufficient saturation as some domains and roles were still missing which could add new insights. After a second round, now totalling a number of 8 interviews, all roles and the most important domains had been covered and sufficient saturation was reached.

The interviews were planned by creating a Teams meeting of half an hour in the calendar of the interviewee and sending an accompanying e-mail. The e-mail contained a short introduction about the researcher and the lab. The confidentiality of the research was underlined as well. All of the interviews were recorded (with the appropriate permission) and transcribed for further analysis.

5.1.3 Analysis methodology

To analyse the data generated by the interviews, coding has been applied to the transcriptions. Coding is a method of indexing your data so that you can develop basic conceptual frameworks [29]. The coding approach followed for this analysis is based on the steps described by Newing [29]:

1. Read through the transcriptions and identify the main subjects, which will be used as codes.

2. Go through the transcriptions again and apply the codes to sentences that mention the subject.

3. Use the selected sentences for a given code to draw conclusions for the subject.

Following from step 1 in the process a number of 11 main subjects were identified. These subject together with the amount of times they were mentioned and how many individual interviewees are shown in Table 5.1. As can be seen for each of the subjects at least 3 interviewees had responded, providing a broad view on the subject.

5.2 Interview findings

As described in step 3 of the analysis process, the selected information by the coding process has been used to gain insight into these subjects. The following paragraphs will describe these insights.

Domain specific Detection of major incidents on the company level is deemed infeasible by experts and it is suggested to detect major incidents on the domain level.

According to the official process within ING, the decision whether an incident should be major is taken by incident management. The teams whose service or product is impacted are consulted as well. This is due to the fact that they know the service best and are therefore best-suited to assess the (possible) impact. An example from the business side is that the classification of an incident is dependent on the specific customer that is experiencing impact. Customers business models differ and therefore incidents can have a different impact. On the other hand, there is the more technical side where for example the number of successful transaction per second is used to determine the impact. These are both very different ways to look at impact but both are required depending on the type of incident. The main take-away is that the decision is taken on relatively low (management) level because very specific domain knowledge is needed to assess the incident.

Subject	**Code used**	**Number of interviewees**
Major incident definition	9	6
Decision making	31	7
Information sources	6	5
Tools	11	7
Reporting of incidents	14	7
Data quality	9	5
Change over time	5	5
Clustering	4	3
Process improvements	7	3
Challenges	7	3

Table 5.1: Coding of the interviews

Combining sources As ING is a very broad company there are other data sources besides the incident report that could be utilized as well.

Some domains, such as wholesale banking, do not only have a measurable technical impact but can also have a business impact. In these cases there also needs to be consultation with the business side of ING. Important aspects here are the agreed service levels in the contract with customers and the cut-off times for transactions between banks. These can differ per customer making it very hard to have general rules regarding the impact of incidents for the business side. However, work is currently being done on creating service level indicators to see if the contracts are being fulfilled. This information source would be very useful in determining the impact of the incident and thus whether it is major or not.

Measuring impact The main driver to decide whether something is a major incident is the customer impact. Determining the customer impact is based on the monitoring systems, knowledge of the internal workings of ING and experience. The interpretation of these different factors is done by experienced employees of ING.

What constitutes as a major incident is also subject to change over time. For a system that needs to detect major incidents it is important to understand what these changes are and why they happen. The first reason for change in major incidents is that the IT systems used at ING are subject to change themselves. Some service that can lead to major incidents today might not even exist anymore tomorrow. And a service that has yet to be implemented might lead to major incidents in the future. Due to the changing IT infrastructure at ING the services that lead to major incidents are subject to change as well. What is changing here is the cause of incidents. However, the impact that is caused by major incidents is still roughly the same, namely major customer impact. The focus of major incident detection should thus be on determining the impact of an incident and not classifying them based on cause.

Explainability The experts noted that the actual system should be explainable. In order to trust the system, they need to know why a certain incident is marked as major. *'We need to ensure that the system is using the right data and not drawing the wrong conclusions'*. The first step in incident resolution is always to verify the impact and for that it is necessary to know what the decision is based on.

Lacking data quality For a machine learning system the quality of the data is one of the most important aspects in its success. It was often mentioned that the quality of the incident data is very unpredictable as there are no guidelines as to what an incident description should contain. In general, the administration of incidents can be seen as a cumbersome task and is therefore sometimes lacking. It is important to make everyone in the bank aware of the importance of this administration. The prioritization of incidents was also noted to be a *'very gray area'*, where there are quite some difference between teams on the interpretation of the different priority levels. This leads to inconsistencies within the data making it harder to classify the incidents.

Another aspect that was mentioned which is influencing the data is that it is unsure when an incident actually happened. We know at what time the incident was created but this does not necessarily match with the time the incident actually happened. This makes it harder to actually assess the full impact of an incident. The problem can be solved by the automatic creation of incidents which is currently being done by an automated alerting tool, which will be explained in the next paragraph.

Different incident sources (user and system) The automated alerting tool is a platform where teams sent all of their monitoring events and based on these events certain pre-specified actions are performed. These actions include the sending of e-mails, calling certain team members or more relevant to us, the automatic creation of incidents. The impact, urgency and priority of these incidents is pre-specified by the team. The actual description of the incident is coming straight from the event sent by the monitoring of the team. This means the data in these descriptions is semi-structured. However, since every team is allowed to use the monitoring tools they prefer this means that the descriptions are still quite diverse as each monitoring tool reports differently, which leads to a high variability. This variability negates the advantage of more structure in automatically generated tickets, making them harder to classify.

Since the automated alerting tool is only a platform, that means teams are able to use in any way they want. The decision of what is logged as an incident and what not is still up to the individual teams. Meaning that for one team incidents get created for all events while for the other it only happens with priority one incidents. Thus the incidents created by the automated alerting tool do not provide a fully representative view of all that is happening within these teams. This difference between teams also comes forward when asked whether the automatically created incidents lead to major incidents. One the one hand we have someone that said that *'majors are rarely coming from automated tickets'* while someone else noted that *'it happens very often that incidents created by the automated alerting tool cause priority one or major incidents'*. In conclusion there is a lot of difference in the way that teams use automatically generated incidents, leading to an inconsistent dataset.

High variability The required domain specific knowledge also leads to a high variability in the incidents text. As each domain has its own expressions all these different expressions lead to more features. Another factor influencing the high variability is that the automatically generated incidents come from a wide array of tools.

Process related insights While gaining insights about the process is not the focus of the interviews it is still an important aspect to our problem and was therefore also often discussed. In general there was a positive sentiment about the major incident management process as a whole.

ING operates at a scale that results in multiple agile teams concurrently improving the systems and hence a complete single overview of all components is not always readily available. It was reported that this makes it difficult to see all the interactions between the different products and services, which impacts incident handling. Monitoring on the service level is becoming quite sophisticated but monitoring on the product level still seems to be lacking. Insight into these chains and implementing monitoring on the product level will allow a better assessment of the impact of an incident and will thus be very useful for the detection of major incidents.

Change over time In general there could be no trend seen in the type of incidents that become major over time. As stated earlier in the paragraph on impact, the cause of major incidents changes all the time, the impact is still roughly the same. A positive trend that was noted is that monitoring by the teams is becoming better and better which means that incidents tend to get detected earlier.

Clustering The interviewees noted that they do notice some clustering where one incident leads to multiple other incidents. However, these clusters often overlap making them hard to distinguish. For major incidents the clustering is not deemed relevant as only one of the incidents from a cluster is promoted to a major.

5.3 Improving the system

Following from the interviews, several ways to improve the system have been brought forward. While most of these improvements are very extensive and out of the scope of this thesis, we will perform an experiment with one of the improvements. Following from the section on the classification of major incidents the main take-away was that this decision is taken on the domain level as domain specific knowledge is required. To test this finding in practice, two experiments using two different domains have been performed.

Methodology The first step is the selection of the domains that will be used in the experiment. A requirement is that the domains should contain major incidents, where a larger number of major incidents is preferred as a small number of major incidents would lead to overfitting and thus a less generalizable result. Based on this requirement two different domains have been selected, which we will call domain 1 and 2. The properties of these domains regarding the number of (major) incidents can be seen in Table 5.2. In order to make a fair comparison between the experiments and the original system, the domain incidents are pre-processed in the exact same way as in the original system (as described in Section 3.1 and 4.2.2). Training of the models is also performed in the same way with 75% of the data being used as training data and the remaining 25% as validation data.

Results The results of the experiments can be found in Table 5.2. Both models show better performance on the domain level compared to the original system. The best results are achieved on domain 2. However, these improved results could also be explained by the relatively larger number of major incidents in these domains, meaning that the domain datasets are better balanced. This better balance could also be the cause of the improvements, so more extensive research is needed to verify the claim that major incident detection on the domain level is the better approach (compared to the company level).

			Logistic Regression			Support Vector Machine		
Domain	Incidents	Majors	**Precision**	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**
All	2,484,110	1032	0.45	0.55	0.49	0.76	0.40	0.53
1	902	37	0.41	0.78	0.54	0.44	0.78	0.56
2	4069	287	0.43	0.88	0.58	0.75	0.56	0.64

Table 5.2: Proof of concept for domain level system

Chapter 6

Discussion

6.1 Main findings

The performance of the implemented major incident detection system seems to be lacking compared to other research in the IT incident field. We were able to identify three reasons within our system that could explain this. The first is the unbalanced data problem, which was not solved by performing sampling. The second reason is the high variability of the data. As stated there were still 69,222 different features in the dataset, which is a large number compared to literature. Lastly, the automatically generated incidents were very hard to classify. The following paragraphs will compare these findings with the current knowledge in literature and the newly identified challenges following from the expert interviews.

The three challenges identified in the system implementation can already be found in existing literature. The unbalanced nature of the data was known beforehand but literature does not state how to solve this problem. In this research we tried to solve this challenge by performing sampling but this did not result in major improvements. Major incident data is extremely imbalanced, even more so then regular incident data which might explain why we are unable to solve this. The second challenge identified, the high variability, was also mentioned in literature. Proposed solutions to this challenge were pre-processing techniques such as stemming and stop-word filtering. These techniques have been applied and did reduce the variability but the variability still remained high. This can be partly explained by the large dataset that was used compared to previous research. However, since our dataset comes directly from industry it shows that the high variability is an actual challenge when working with incident data. It might be worthwhile to use more intelligent solutions to reduce the variability. Such advanced solutions are considered out-of-scope of this project and were therefore not explored. Another possibility is to reduce the variability by focusing on the domain level, which reduces the number of domain specific terms. While the improvements on the domain level look promising, a more detailed investigation has been left for future work. Finally, the automatically generated tickets proved to be a big challenge as the system was not able to classify them. This is not in agreement with literature as it was suggested that automatically generated tickets would be easier to classify. An explanation for this has been provided by the experts, which will be discussed later on.

The expert interviews resulted in a total of seven challenges for major incident detection. Three of which have not been identified in literature before, these are: The importance of using different sources, explainability and the notion of impact for major incidents. The use of different sources would provide additional information to the major incident detection system and is very likely to improve the results of the system. A possible source would be performance indicators. This does require some form of schema matching however, which is a whole field in itself and therefore not explored in this thesis. The importance of the explainability of the decision is also a new property and is especially relevant as more work is focusing on neural networks, which struggle with this. The last newly identified challenge is the question on how to measure impact. The major incident classification process is mostly driven by the (possible) impact of a certain incident. The experts state the if we would be able to use the impact of an incident it would become much easier to classify an incident as major. In order to do so the integration with different data sources, as mentioned earlier, is important as just an incident description does not contain enough information.

The remaining four challenges that have been identified can already be found in literature. The first of these challenges is the lacking data quality which was confirmed by the experts by saying that incident reporting can be seen as a cumbersome task by some. This sometimes results in insufficient incident descriptions. The second challenge is the domain specific nature of incidents. Almost all of the experts agreed that domain specific knowledge is required and literature is in agreement with this as well. To endorse this finding a small experiment using only incidents from certain domains was performed which greatly improvement results, once again underlining the importance of the domain focus. The third challenge, which was also found in the system, is the high variability of the data. This ties back to the domain specific knowledge required as each domain will use its own terms to describe the incident resulting in a more diverse dataset. Based on this notion we recommend again that major incident detection is performed on the domain level and not on the company level. This will allow the system to incorporate more domain knowledge and reduce the variability as incidents from other domains are left out. Finally, the difference in automatically and user generated incidents was confirmed by the experts as well. While literature states that the automatically generated incidents should be easier to classify we found the opposite to be true when implementing our system. The experts were able to clarify this finding. Within ING a large array of tools is used to automatically generate these incidents. This means the dataset is still very diverse and the advantage of more similarly structured incidents is taken away.

The evaluation of the different machine learning models, by applying them in the major incident detection system, showed that Logistic Regression, Support Vector Machine and the Neural Network were performing the best. Studies that compare different models on incident data also state that the Support Vector Machine performed best, so we can confirm those findings. Logistic Regression has only been used in one study before but despite its simplicity it seems to perform really good on incident data and should therefore not be overlooked in research. Finally, the Neural Network also showed very promising results in spite of the small network used. This shows that in cases where explainability is not an issue, Neural Networks might be the best option as improvements can still be achieved by optimizing the network. We suggest that future work further investigates the application of neural networks on incident data.

In literature it was identified that system generated incidents should be easier to classify than user generated incidents. We were not able to verify these statements and even found the opposite to be true: User generated incidents are easier to classify than system generated incidents incidents. The underlying reason for this finding followed from the interviews as it became apparent that the system generated incidents come from a wide variety of tools. There is a lack of standardization as each team is allowed to use its own tool for incident creation. This results in a very diverse dataset which is difficult to classify. Secondly, it is a matter of available data. There are just more user generated incidents available and the models can be trained on a larger dataset which typically leads to better results. In general, it always depends on the data and we show that assumptions which state that one type of incidents is easier to classify than another can therefore not be made.

To handle the data imbalance, oversampling and undersampling have been evaluated. Previous research is undecided whether undersampling can improve results. In our case random undersampling did not improve the results but performed very similar to random oversampling. We also compared random oversampling to artificial oversampling (SMOTE) which has not been used on incident data before. The results indicate that the performance is very similar and in terms of F1-score there is no real reason to prefer one over the other. It should be noted that artificially generating new incidents is more computationally expensive and random oversampling might thus be more favourable.

6.2 Limitations

Due to the scope of this thesis and the possible biases when performing interviews there are some limitation that should be noted. First, the limitations regarding the designed and implemented system will be discussed. This is followed by the threats to validity that might be present in the interviews.

System - Data As stated in Section 4.2 the first *available* version of an incident is used. This might however not be the first *reported* version of the incident. Within ING, the incident reports are exported to the database in batches. Between the creation of an incident and running the batch process there is still some time where the incident can be updated before it is exported to the database. It is therefore likely that we are not using the first reported version of all incidents and some extra knowledge about the incident might have been added in the mean time. While this might not be the best approach regarding the detection time of incidents it does not invalidate the results. Furthermore, only the incident reports were used as a data source for the system. Other data sources such as logs could be used as well but were considered out of scope as assigning the right logs to incident is a research topic in itself.

System - Generalization The results achieved by the system can not directly be generalized to other organizations due to several factors. The first factor is that the definition of what constitutes a major incident may differ from organization to organization. An organization with a different definition might get widely different results. The second factor is that incidents are often related to the underlying infrastructure of an organization. While the system might be able to recognize these components within the ING environment and optimize for these particular components this might not be the case for other organizations. Even other banks might use a completely different infrastructure. The final factor is the lack of publicly available incident datasets, as there are none. By testing the system on different incident datasets we might be able to make better estimates of the generalizability of the system. Another limitation regarding the results (and thus generalizability) is that the system was never put into practice. While using a validation set does provide a good indication of how the system performs the ultimate test would be to put it in practice. Unfortunately, due to a policy change, this was not possible anymore at ING. It will remain unclear how the system performs in a real world scenario.

Interviews The interviewee selection based on chain referral is a rather subjective method of gathering interviewees. While it is useful for our case, where it is unclear which people are involved, we can not say with certainty that all relevant people have been interviewed. There might still be more insight to be gained into the major incident management process, but to the best of our knowledge all relevant parties have been involved. The interview process itself is also susceptible to bias as the author interned at ING at the time of this study. To prevent bias in the interview questions, they have been formulated in correspondence with the thesis supervisor. While the questions were drafted in English, most of the interviews (7 out of 8) were held in Dutch as it would be more comfortable for the interviewees. As such, there is the opportunity for misinterpretation while translating (both for the questions to Dutch and the answers to English). The interviewees are also susceptible to bias, as they are part of ING and the incident management process itself. Due to their involvement in the process they might not be fully objective. Finally, there is the so-called social desirability bias [11] where interviewees have the tendency to answer question in a socially desirable way. To minimize this bias respondents were ensured that the interviews would be handled anonymously and in the unlikely event that they were to be quoted, explicit consent would be asked in advance.

6.3 Future work

The system created in this study is the first of its kind, which means that there is still a lot of room for improvement. The first improvement is that the system should focus on the domain level and not the company level as a small experiment has already shown this improves results. However, a full investigation is needed to verify these results in practice.

There is also the possibility of combining and incident classification system with a major incident detection system. The classification system would be used to determine the domain of an incident and forward it to the major incident detection system that has been trained for that particular domain.

The second improvement comes in the form of using additional data sources. This would require additional research on combining these different data sources. The third improvement would be to create a system that is explainable and can state why an incident is considered major. Explainability is becoming more and more important in different fields and major incident detection would be a great use case as the explanation of why an incident is major is really helpful information for the persons responsible for resolving the incident.

One of the challenges that was identified in the literature, system implementation and the expert interviews as well, was the high variability of the data. Several techniques such as stemming and stop-word removal have been applied but did not result in major improvements. High variability is known to limit the performance of machine learning models meaning that a solution for this problem is very much needed. Therefore, we suggest that more intelligent solutions that also take into account the context of the words are applied to incident data. This could be word embeddings combined with a neural network as this combination has not been used on incident data before.

Another one of the problems this research tried to address was the imbalanced data. While several sampling techniques have been applied none led to meaningful increases in performance. Therefore, more research on handling imbalanced data, more specifically in the case of incident data, is required.

While only discussed briefly, neural networks might bring big improvements in the field of IT Service Management. In this study only a very simple network was evaluated and its results were on the same level as the other top performing models. This suggests that with more optimization neural networks would be able to outperform all other machine learning models. So future research focusing on IT Service Management should include neural networks.

Chapter 7

Conclusion

Based on the research performed in this thesis, both the implementation of the system and the interviews with experts, the research questions will be revisited and answered.

RQ1: What are the challenges in major incident detection?

To answer the first research question a system to detect major incidents was implemented and experts were interviewed. The five challenges identified in literature were confirmed and three new challenges were identified. The first novel challenge is measuring impact. The impact is heavily used in determining the priority of an incident and will thus be very valuable for a major incident detection system. The second novel challenge is making use of different sources. By incorporating different sources the detection, in theory, should improve. The final novel challenge is that of explainability. The decision on why an incident is major should be explained so that the people responsible for solving the incident can start resolving it right away.

RQ2: Which machine learning models are best suited for incident data?

The second research question has been answered by evaluating the performance of different machine learning models. The three top performing models based on the F-score are: Logistic Regression, Support Vector Machine and the Multi-Layer Perceptron. While the Multi-Layer Perceptron is unsuited for our use case due to its lack of explainability, it might be a very good option for other problems regarding incident data especially since there is still a lot of room for optimization. For major incident detection the Logistic Regression model is considered the best as it has a similar F-score as the Support Vector Machine but higher scores on its recall, which is preferred over precision.

RQ3: What is the effect of splitting the data on its source?

Research question three has been answered by the means of an experiment. This experiment showed that the user generated incidents were easier to detect than the system generated incidents. While this result is unexpected compared to literature, two causes can be identified. The first cause is that the dataset contained more user than system generated incidents which means that there was more data available for the user generated incidents. The second cause was identified by the expert interviews where it was stated that there is a very high variability in the tools used to generate incidents. This high variability negates the perceived advantage of more structured data in the system generated incidents.

RQ4: What is the best way to deal with data imbalance?

In order to answer research question four both oversampling and undersampling have been applied. The results were found to be very similar but did not lead to an improvement compared to no sampling at all. Applying artificial oversampling (SMOTE) also did not manage to improve the results. Data imbalance, at least for incident data, still remains a large challenge and further research into solutions is needed.

Bibliography

[1] Claire Agutter. *ITIL lifecycle essentials: your essential guide for the ITIL foundation exam and beyond*. IT Governance Publishing, 2013.

[2] Jasmina Bogojeska, Ioana Giurgiu, David Lanyi, George Stark, and Dorothea Wiesmann. Impact of HW and OS type and currency on server availability derived from problem ticket analysis. In *IEEE/IFIP NOMS 2014 - IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World*. IEEE Computer Society, 2014. ISBN 9781479909131. doi: 10.1109/NOMS.2014.6838347.

[3] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[4] Junjie Chen, Xiaoting He, Qingwei Lin, Yong Xu, Hongyu Zhang, Dan Hao, Feng Gao, Zhangwei Xu, Yingnong Dang, and Dongmei Zhang. An Empirical Investigation of Incident Triage for Online Service Systems. In *Proceedings - 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice, ICSE-SEIP 2019*, pages 111–120. Institute of Electrical and Electronics Engineers Inc., may 2019. ISBN 9781728117607. doi: 10.1109/ICSE-SEIP.2019.00020.

[5] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 13-17-Augu, pages 785–794, New York, New York, USA, aug 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785. URL `http://dl.acm.org/citation.cfm?doid=2939672.2939785`.

[6] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, sep 1995. ISSN 0885-6125. doi: 10.1007/bf00994018.

[7] G. A. Di Lucca, M. Di Penta, and S. Gradara. An approach to classify software maintenance requests. In *Conference on Software Maintenance*, pages 93–102, 2002. doi: 10.1109/icsm.2002.1167756.

[8] Yixin Diao, Hani Jamjoom, and David Loewenstern. Rule-based problem classification in IT service management. In *CLOUD 2009 - 2009 IEEE International Conference on Cloud Computing*, pages 221–228, 2009. ISBN 9780769538402. doi: 10.1109/CLOUD.2009.80.

[9] Rui Ding, Qiang Fu, Jian Guang Lou, Qingwei Lin, Dongmei Zhang, Jiajun Shen, and Tao Xie. Healing online service systems via mining historical issue repositories. In *2012 27th IEEE/ACM International Conference on Automated Software Engineering, ASE 2012 - Proceedings*, pages 318–321, 2012. ISBN 9781450312042. doi: 10.1145/2351676.2351735.

[10] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[11] Adrian Furnham. Response bias, social desirability and dissimulation. *Personality and individual differences*, 7(3):385–400, 1986.

[12] Stuart D. Galup, Ronald Dattero, Jim J. Quan, and Sue Conger. An overview of IT service management. *Communications of the ACM*, 52(5):124–127, may 2009. ISSN 00010782. doi: 10.1145/1506409.1506439.

[13] Muhamet Gërvalla, Naim Preniqi, and Peter Kopacek. IT infrastructure library (ITIL) framework approach to IT governance. In *IFAC-PapersOnLine*, volume 51, pages 181–185. Elsevier B.V., oct 2018. doi: 10.1016/j.ifacol.2018.11.283.

[14] Rajeev Gupta, K. Hima Prasad, Laura Luan, Daniela Rosu, and Chris Ward. Multi-dimensional knowledge integration for efficient incident management in a services cloud. In *SCC 2009 - 2009 IEEE International Conference on Services Computing*, pages 57–64, 2009. ISBN 9780769538112. doi: 10.1109/SCC.2009.48.

[15] Ashley Hanna and Stuart Rance. ITIL®glossary and abbreviations English. *Jul-2011*, 2011. URL `https://www.axelos.com/corporate/media/files/glossaries/itil{_}2011{_}glossary{_}gb-v1-0.pdf`.

[16] Marina Danchovsky Ibrishimova. Cyber incident classification: Issues and challenges. In *Lecture Notes on Data Engineering and Communications Technologies*, volume 24, pages 469–477. Springer, oct 2019. doi: 10.1007/978-3-030-02607-3_43.

[17] Jon Iden and Tom Roar Eikebrokk. Implementing IT Service Management: A systematic literature review. *International Journal of Information Management*, 33(3): 512–523, jun 2013. ISSN 02684012. doi: 10.1016/j.ijinfomgt.2013.01.004.

[18] ING. ING at a glance, 2020. URL `https://www.ing.com/About-us/Profile/ING-at-a-glance.htm`.

[19] U K ItSMF. *ITIL Foundation Handbook*. The Stationery Office, 2012.

[20] Yong Bin Kang, Arkady Zaslavsky, Shonali Krishnaswamy, and Claudio Bartolini. A knowledge-rich similarity measure for improving IT incident resolution process. In *Proceedings of the ACM Symposium on Applied Computing*, pages 1781–1788, 2010. ISBN 9781605586380. doi: 10.1145/1774088.1774466. URL `http://www.cryer.co.uk/glossary,`.

[21] Gary King and Langche Zeng. Logistic regression in rare events data. *Political analysis*, 9(2):137–163, 2001.

[22] Patrick Kubiak and Stefan Rass. An Overview of Data-Driven Techniques for IT-Service-Management. *IEEE Access*, 6:63664–63688, 2018. ISSN 21693536. doi: 10.1109/ACCESS.2018.2875975.

[23] Haochen Li and Zhiqiang Zhan. Machine learning methodology for enhancing automated process in IT incident management. In *Proceedings - IEEE 11th International Symposium on Network Computing and Applications, NCA 2012*, pages 191–194, 2012. doi: 10.1109/NCA.2012.28.

[24] Derek Lin, Rashmi Raghu, Vivek Ramamurthy, Jin Yu, Regunathan Radhakrishnan, and Joseph Fernandez. Unveiling clusters of events for alert and incident management in large-scale enterprise it. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1630–1639, New York, New York, USA, 2014. Association for Computing Machinery. ISBN 9781450329569. doi: 10.1145/2623330.2623360. URL `http://dl.acm.org/citation.cfm?doid=2623330.2623360`.

[25] Rong Liu and Juhnyoung Lee. IT Incident Management by Analyzing Incident Relations. In *LNCS*, volume 7636, pages 631–638. Springer, Berlin, Heidelberg, nov 2012. doi: 10.1007/978-3-642-34321-6_49. URL `https://link.springer.com/chapter/10.1007/978-3-642-34321-6{_}49`.

[26] Jian Guang Lou, Qingwei Lin, Rui Ding, Qiang Fu, Dongmei Zhang, and Tao Xie. Experience report on applying software analytics in incident management of online service. *Automated Software Engineering*, 24(4):905–941, dec 2017. ISSN 15737535. doi: 10.1007/s10515-017-0218-1.

[27] Andrii Maksai, Jasmina Bogojeska, and Dorothea Wiesmann. Hierarchical Incident Ticket Classification with Minimal Supervision. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, volume 2015-Janua, pages 923–928. Institute of Electrical and Electronics Engineers Inc., jan 2014. ISBN 9781479943029. doi: 10.1109/ICDM.2014.81.

[28] Mauricio Marrone, Francis Gacenga, Aileen Cater-Steel, and Lutz Kolbe. IT service management: A cross-national study of ITIL adoption. *Communications of the Association for Information Systems*, 34(1):865–892, feb 2014. ISSN 15293181. doi: 10.17705/1cais.03449. URL `https://aisel.aisnet.org/cais/vol34/iss1/49`.

[29] Helen Newing. *Conducting research in conservation: social science methods and practice*. Routledge, 2010.

[30] Cristian Paduraru and Mihaela Elena Breaban. Dealing with data imbalance in text classification. In *Procedia Computer Science*, volume 159, pages 736–745. Elsevier B.V., jan 2019. doi: 10.1016/j.procs.2019.09.229.

[31] 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.

[32] Padej Phomasakha Na Sakolnakorn and Phayung Meesad. Decision tree-based model for automatic assignment of IT service desk outsourcing in banking business. In *Proc. 9th ACIS Int. Conf. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2008 and 2nd Int. Workshop on Advanced Internet Technology and Applications*, pages 387–392, 2008. ISBN 9780769532639. doi: 10.1109/SNPD.2008.71.

[33] J. Ross Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27(3):221–234, 1987.

[34] Leonard Renners, Felix Heine, and Gabi Dreo Rodosek. Modeling and learning incident prioritization. In *Proceedings of the 2017 IEEE 9th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2017*, volume 1, pages 398–403. Institute of Electrical and Electronics Engineers Inc., nov 2017. ISBN 9781538606971. doi: 10.1109/IDAACS.2017.8095112.

[35] Suman Roy, Durga Prasad Muni, John-John Yeung Tack Yan, Navin Budhiraja, and Fabien Ceiler. Clustering and Labeling IT Maintenance Tickets. In Quan Z Sheng, Eleni Stroulia, Samir Tata, and Sami Bhiri, editors, *Service-Oriented Computing*, pages 829–845, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46295-0.

[36] Saeed Salah, Gabriel Maciá-Fernández, Jesús E. Díaz-Verdejo, and Leovigildo Sánchez-Casado. A Model for Incident Tickets Correlation in Network Management. *Journal of Network and Systems Management*, 24(1):57–91, jan 2016. ISSN 10647570. doi: 10.1007/s10922-014-9340-6. URL `https://link.springer.com/article/10.1007/s10922-014-9340-6`.

[37] Sara Silva, Ruben Pereira, and Ricardo Ribeiro. Machine learning in incident categorization automation. In *Iberian Conference on Information Systems and Technologies, CISTI*, volume 2018-June, pages 1–6. IEEE Computer Society, jun 2018. ISBN 9789899843486. doi: 10.23919/CISTI.2018.8399244.

[38] Juliana Tolles and William J Meurer. Logistic regression: relating patient characteristics to outcomes. *Jama*, 316(5):533–534, 2016.

[39] Shannon Vavra. Amazon outage cost S&P 500 companies $150M - Axios, mar 2017. URL `https://www.axios.com/amazon-outage-cost-sp-500-companies-150m-1513300728-aaff3a9e-d5de-4700-aded-55614cf7852c.html`.

[40] Bogdan Walek. Intelligent System for Ordering Incidents in Helpdesk System. In *IC-SEC 2017 - 21st International Computer Science and Engineering Conference 2017, Proceeding*, pages 224–228. Institute of Electrical and Electronics Engineers Inc., aug 2018. ISBN 9781538607879. doi: 10.1109/ICSEC.2017.8444003.

[41] Jian Xu, Liang Tang, and Tao Li. System situation ticket identification using SVMs ensemble. *Expert Systems with Applications*, 60:130–140, oct 2016. ISSN 09574174. doi: 10.1016/j.eswa.2016.04.017.

[42] Jian Xu, Hang Zhang, Wubai Zhou, Rouying He, and Tao Li. Signature based trouble ticket classification. *Future Generation Computer Systems*, 78:41–58, jan 2018. ISSN 0167739X. doi: 10.1016/j.future.2017.07.054.

[43] Dongmei Zhang, Yingnong Dang, Jian Guang Lou, Shi Han, Haidong Zhang, and Tao Xie. Software analytics as a learning case in practice: Approaches and experiences. In *ACM International Conference Proceeding Series*, pages 55–58, New York, New York, USA, 2011. ACM Press. ISBN 9781450310222. doi: 10.1145/2070821.2070829. URL `http://dl.acm.org/citation.cfm?doid=2070821.2070829`.

[44] Harry Zhang. The optimality of Naive Bayes. In *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2004*, 2004. ISBN 1577352017.

[45] Xiang Zhang, Junbo Zhao, and Yann Lecun. Character-level Convolutional Networks for Text Classification *. Technical report, 2015.

Appendix A

Interview guide

Introduction
First of all, am I allowed to record this meeting? The recording will only be used by me to transcribe the interview.

I will start with a short introduction of myself. My name is Thomas and I am a part of the AI4FinTech lab, which is a collaboration between the TU Delft and ING. Currently I am doing research on detecting major incidents using machine learning. That is why I like to talk to people involved in major incidents which is why I have planned this interview.

Introductory questions

- Could you introduce yourself?
- What is your involvement in major incidents?
- How often do you encounter major incidents in your domain?
- How would you define a major incident?
  - Follow-up: How would you define major impact?
  - Follow-up: How many services need to be impacted?

Main questions

- Could you please describe the process of deciding whether an incident is major or minor?
  - Follow-up: How long does this process usually take?
- How do potential major incidents get reported to you?
  - Follow-up: Who reports these to you?
- What are the sources of information you use to make this decision?
- Are there any threshold values that guide your decision?

- Do you use any tools to assist you in making this decision?
- Do you consult with other people when making a decision? If so, who?
- How do you assess the impact of the incident?
- Let's say there was a new tool to support you in the decision making process. What features would you want/need?
- When is something definitely not a major incident?

Data quality questions

- Who creates the incidents?
- Are there incidents that get created automatically?
- Does it happen that it is forgotten to make an incident?

Process related questions

- Are the type of incidents that become major incidents changing over time?
- Where do you see opportunities to improve the major incident management process?