Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Johan Selnes

Automated Analysis of the Impact of Security Incidents on Company Value

Master's thesis in Master in Information Security
Supervisor: Kyle Porter
May 2022

NTNU
Norwegian University of Science and Technology

Johan Selnes

Automated Analysis of the Impact of Security Incidents on Company Value

Master's thesis in Master in Information Security
Supervisor: Kyle Porter
May 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

NTNU

Kunnskap for en bedre verden

Abstract

This thesis develops a new approach for automating and analyzing the impact of security incidents on company value. The method is based on detecting security incidents in news articles and then using an event study to determine the change in stock price resulting from the incident. Furthermore, the method tests if modern natural language processing models based on pre-trained neural networks perform well with a classification task based in the cyber security domain. The method shows promise with calculated impact in line with previous studies at a 4 per cent loss in company value 50 days after a security incident is present in news articles. However, the modern natural language processing techniques used to classify security incidents did not outperform simpler statistical models, which performed exceptionally well.

Sammendrag

Denne oppgaven utvikler en ny tilnærming for å analysere og automatisere kostnaden av sikkerhetshendelser på selskapets verdi. Metoden er basert på å oppdage sikkerhetshendelser i nyhetsartikler og deretter bruke en hendelsesstudie for å bestemme endringen i aksjekursen som følge av sikkerhetshendelsen. Videre tester metoden om moderne naturlig språkbehandlingsmodeller basert på forhåndstrente nevrale nettverk fungerer godt med klassifiseringsoppgave basert i cybersikkerhetsdomenet. Metoden virker lovende med en kalkulert effekt i tråd med tidligere studier. Det kalkulerte resultatet ble en 4 prosent tap i selskapsverdi 50 dager etter at en sikkerhetshendelse er nevnt i nyhetsartikler. De moderne prosesseringsteknikkene for naturlig språk som ble brukt til å klassifisere sikkerhetshendelser overgikk imidlertid ikke enklere statistiske modeller, som presterte eksepsjonelt bra.

Contents

Figures

Tables

Code Listings

Chapter 1

Introduction

Companies are affected by cyber security incidents all the time. News articles about hackers, data leakage, or downtime due to ransomware attacks are increasingly becoming part of the news cycle. At the same time, companies, in general, are increasingly transforming more and more into IT companies as information technology plays an ever-increasing role in the world and everyday operations. This integration of IT systems means that companies must allocate resources to securing their cyberinfrastructure. This allocation process is an exercise in risk management and, ultimately, a cost-benefit analysis that everyone with a digital footprint must perform. However, estimating the actual cost of security incidents is complicated since significant security incidents are so rare for an individual company that it is difficult to get reliable data.

Additionally, many different areas incur costs whenever a cyber security incident occurs. Examples of such areas are tangibles, like lower earnings due to recovery costs, reduced productivity due to unavailability, or fines from regulatory bodies. Still, there are also intangible costs such as loss of consumer trust, reputation damage, etc. Furthermore, many of the companies impacted by major security incidents choose to share as little information as possible from the incidents and, if possible, might decide not to disclose anything at all. This secrecy makes it difficult to extrapolate costs.

Therefore looking at the movement in stock market prices could be interesting for deriving the potential costs. A company's stock price can be viewed as the future expected returns and a discount rate based on how far into the future those returns are expected. All active participants in the stock market are effectively competing to estimate the value of a company, in real-time, based on public information. Therefore, market participants are financially incentivized to take into account all impacts on both tangibles and intangibles and produce a best-effort analysis of the total effect of the security incident as quickly as possible based on all available information. Furthermore, listed companies are legally required to disclose events of material impact to shareholders; therefore, the average effect on listed companies should function as a reasonable proxy/estimate of the average cost of security incidents.

1.1 Problem Statement

This thesis aims to develop a method to automate the analysis of security incidents' impact on affected companies' stock prices. This method requires a comprehensive approach where steps from the collection, classification and the potential impact are explored and studied. The results from this study will attempt to answer the following hypotheses:

- **h1** State of the art NLP models can be trained to classify text concerning security incidents.
- **h2** State of the art NLP models achieves better results when classifying news articles concerning security incidents than more straightforward machine learning methods.
- **h3** News articles concerning security incidents can be used to conduct an event study and give insight into the impact of security incidents on stock prices.

1.2 Unique Contributions

The attempt to automate the security incident identification and classification process using natural language processing (NLP) methods is novel. Previous studies into security incident stock impact have primarily used handpicked or insurance data due to the difficulty of manually reviewing all newsworthy events from 10s of thousands of companies. As a result of this problem, most event studies into security incidents' impact on stock performance include a line stating that the sample size should ideally be bigger. An example of this is the study "*The Effect of Data Breaches on Shareholder Wealth. Risk Management and Insurance Review*" by Gatzlaff and McCullough [1]. Which states: "*Future research involving an even larger sample ... would be useful in drawing broader conclusions*" [1]. Such studies usually have a sample size of 100-200 handpicked events.

Secondly, the application of automation in selecting samples might yield a different result than samples chosen by humans, either because of other biases or sample size.

Thirdly, suppose modern NLP models prove to be helpful in the domain of cyber security. It might open the door to new avenues and applications as the domain knowledge might transfer into different problems and applications within the field.

Fourthly, gathering and labelling a dataset from news articles in the domain of cyber security incidents is new.

Chapter 2

Related Work

2.1 Effects of Cybersecurity Incidents

Due to the lack of self-reporting, data regarding cyber security incidents are difficult to obtain. Such incidents, by their nature, can be expected to have a negative short term impact on the stock price. This negative impact means that companies' executives and decision-makers are incentivized not to publicize incidents because they are often compensated based on stock performance. On the other hand, laws such as General Data Protection Regulation (GDPR) require disclosing and notification if the incident affects personal/user data. On a similar note, most public stock exchanges require companies to disclose events of material impact on the company. However, given few incentives to announce such events, one can expect companies to make a restrictive interpretation of what "material" means to reveal as little as possible.

One of the first studies into the effect of incidents was *The economic cost of publicly announced information security breaches: empirical evidence from the stock market* by Campbell et al. (2003) [2]. They looked into the effect of incidents published in major newspapers from 1995 to 2000. The total sample size obtained over the period was 43 events from 38 public companies, where they studied the price movement in a 3-day window surrounding the time of announcement. When looking at all the samples, there was a slight indication of an adverse reaction to incidents but not with a good p-value ($p = 0.1393$). When splitting the data and only looking at incidents involving unauthorized access to data, they found that the adverse reaction was much more pronounced and highly significant. The authors suggest that the type of incident is of material importance concerning the negative effect of the incident.

In the paper *The effect of data breaches on shareholder wealth* by Gatzlaff and McCullough (2010) [1] they looked at the impact of incidents involving leakage of personal information. The paper identified 77 events in the period 2004 to 2006 which was used in the study. They estimated the performance of a given stock relative to the market using a 1-year window before the incident. This estimated performance was compared to the stock's performance up to 120 days after the

incident became public. They found that the overall impact of disclosing such events is negative and that the result is statistically significant, with a 0.84 per cent loss in company value. Furthermore, they also found that the dropdown effects occur for up to 40 days after disclosure before performance returns to normal and starts tracking the market again. Additionally, they remark that growth stocks (higher book to value) are more affected than other types of stocks.

In the paper *Do Firms Underreport Information on Cyber-Attacks? Evidence from Capital Markets* [3] from Amir, Eli, et al. in 2018 dug a bit deeper into data about security incidents and used it to estimate the likelihood of a security incident disclosure happening voluntarily from the affected company, as opposed to it being disclosed by an unrelated 3rd party. They manually combined and verified data from two sources and ended up with 276 samples from 2010 to 2015. Then they divided the data into self-disclosed and discovered by others. Then they categorized the data that was discovered by others into material and immaterial. They show that the immaterial set does not affect the stock price meaningfully. They argue that the samples present in the material set should have been reported/disclosed by companies. At the same time, the immaterial group did not warrant the need for disclosure. They show that in cases where a firm rapidly discloses the incident, the company's value drops by 0.33% in the first three days and 0.72% over the entire month after the breach. For the case where firms did not voluntarily disclose the incident, but it was discovered by someone else, they find that the drop in value is 1.47% and 3.56% for the first three days and the entire month, respectively. Based on the data, the authors suggest that managers are more likely to withhold severe incidents and only choose to disclose the incident when investors already suspect(with a 40% confidence) that an incident has occurred.

The newest and most comprehensive paper with regards to sample size is the paper *A Comprehensive Analysis of Cyber Data Breaches and Their Resulting Effects on Shareholder Wealth* from 2020 by Hogan et al. [4]. They obtained data regarding personal information leakage from a proprietary dataset from the insurance data provider Advisen Ltd. claims that the samples in their cyber loss data have been manually gathered and or verified. The Adivsen cyber loss data yielded 3991 samples, by far the largest of any known study. They also find that short term adverse effects but not as pronounced as in some of the smaller earlier studies. Furthermore, they also find significant negative long term results with an average loss of 7.46% with a p-value < 0.0001, 250 trading days or about a year after the incident.

As highlighted, several studies all conclude adverse effects as a reaction to security incidents. However, all studies ultimately depend on manually gathered and/or proprietary data, except for Hogan et al.(2020), all studies use few samples.

2.2 Text Classification

There has been plenty of work on machine learning models to understand and classify text. In the last ten years, there have been considerable developments in

the usage of deep learning models for this task. Rashid and O'Keefe explored Bidirectional Long Short-Term Memory (BiLSTM), Convolutional Neural Network (CNN), and Recurrent Neural Network Models [5] in 2014 with promising results. However, the major disadvantage to these models is that they are slow to train. Part of the reason behind the slowness is that input data needs to be processed sequentially through the network. Sequential processing is necessary because the previous state's inputs are required to process the current state. This kind of sequential processing does not leverage modern computing hardware such as graphics cards and multi-core processors, which are all designed to process data in parallel for optimal efficiency; this is especially true for graphics cards. Transformer models came out of a Google research by Vaswani et al. in 2017 [6] in the paper *Attention Is All You Need*. They present a new type of machine learning model that does not rely on convolution and other recurrent mechanisms such as the ones used by ALRashdi and O'Keefe. Instead, transformers rely solely on the concept of attention. The advantage of transformer models is the training speed which allows for larger models and the ingestion of more data. The transformer architecture employs an encoder-decoder architecture similar to Recurrent Neural Networks. The difference is that the input sequence is passed to the transformer architecture in parallel. As an example, in a traditional neural network, a sentence such as "The quick brown fox jumps over the lazy dog" is passed to the encoder one word at a time. This is because the word embeddings has dependencies on the word preceding them. In a transformer encoder, the entire sentence is passed in at once, and the word embedding dependencies are calculated on the whole sequence simultaneously. This process is called self-attention. A simplified explanation is that an attention weight is calculated towards every other token in the input sentence for every token in the input. This weight represents the influence each word has on one another. The model developed by Vaswani et al. in 2017 [6] was developed for the task of language translation. It works by taking a sentence from the input language as input into the encoder block, which(from a high level) computes word embedding containing the meaning of the sentence. This computation is then passed as input to the decoder. The decoder takes the previously generated word in the target language and the output from the encoder and generates the following word in the target language. Thus one can extrapolate that the encoder learns what word means and their context for a single language. In contrast, the decoder learns how to map words/the meaning from the encoder to the target language words. One way to convert such an architecture to a classifier is to remove the decoder part and only rely on the encoder, which produces the text's " meaning ". Then add additional layers, which eventually collapse the result to the classes one wants to classify.

Following transformer models, Google proposed The Bidirectional Encoder Representation of Transformer (BERT) in a paper by Devlin et al. [7]. The idea behind BERT is that it is pre-trained to connect bidirectional relations on unlabeled text. Since human text/language transcends many domains and applications, the pre-trained model can be deployed to many different scenarios with little addi-

tional training and significant effect. The Bidirectional **Encoder** Representation of Transformer (BERT) is en effect lots of encoder blocks as seen was introduced by Vaswani et al. in 2017 [6] stacked together. Creating a BERT model for a specific task can then be split into two. Firstly the model undergoes pretraining. This training is done by training BERT on two tasks simultaneously. One of the tasks is masked language modelling (MLM), and the other is next sentence prediction (NSP). The MLM task is to predict a masked word in a sentence. The NSP task is to predict the following sentence in a paragraph. The NSP training is accomplished by feeding BERT two sentences and having the model output a 0 or 1 depending on whether the second sentence is the following sentence. Note that sentence in the BERT context is not an actual sentence but a sequence of words that can contain 0 or multiple "human" sentences. The output of the pretraining is that, hopefully, the BERT model has some generalized understanding of the language. After the BERT model has undergone pretraining, the next step is to fine-tune the model. The fine-tuning is done by training the model on a task that is not the MLM or NSP task. In a generalized sense, the process identifies the task, creates a new dataset, connects some untrained network layers at the end of the pre-trained BERT model, and trains the model using the new dataset. In theory, this process should be fast as the model is already trained on the pretrained, and one is only interested in updating the weights on the new layers to suit the new task.

Another development in the field is the Generative Pre-trained Transformer (GPT), such as GPT-2 [8]. GPT models are like BERT models, attention-based transformers. There are some differences; more specifically, BERT comprises transformer encoder blocks while GPT uses transformer decoder blocks. One of the significant differences between the two is that GPT is auto-regressive. Auto-regression means that each token (word) has the context of the previous tokens (words) when predicting text. This context implies that GPT operates one token at a time. On the other hand, BERT works on all the text at once. As a result, when BERT is "learning", it has the context of the words ahead and behind, but it uses a dropout rate to limit the information available to the model. In contrast, GTP can only look back but has the context of all the words behind the current token. Delvin et al. argue in their paper [7] on BERT that this lack of context on what lies ahead is a significant weakness of GPT models. They highlight question answering problems where they believe that giving the model the context of what lies ahead is a considerable strength. Even when compared to models such as the BiLSTM, which in effect are separately learning left to right and right to left and then concatenating the results and thus are not truly bidirectional.

The application of large transformer models as a classifier of security incidents in the news is unexplored territory. Still, BERT has shown great improvements in various NLP tasks. BERT or BERT derivatives occupy many spots on the General Language Understanding Evaluation (GLUE) benchmark[1]; therefore, this model is probably a good fit for the suggested classification task. An example paper demonstrating BERT as a classifier is the CrisisBERT paper [9]. In this paper,

[1] `https://gluebenchmark.com/leaderboard`

the researchers used BERT together with text from social media to identify crisis events with great accuracy.

Chapter 3

Method

3.1 Data Collection

3.1.1 News Articles

Due to the topic's novelty, there is no good dataset or a standard approach to data collection for classifying security incidents from news articles. Therefore, data collection must be done from scratch and due to the volume, this data collection must be done programmatically. At a high level, this can be described with the following steps:

- Download a list of all New York Stock Exchange (NYSE) and Nasdaq tickers.
- Query Reuters for the company's page based on the ticker.
- Iterate through the news articles on the ticker's page and extract the URL.
- Download the news articles using the URL and the Newspaper Library.[1]
- Save the article as a JSON file containing the article's title, URL, publish date, stock ticker, and text.

Reuters removes news articles that are older than 2-3 years. Therefore, the resulting dataset is limited to this time frame.

3.1.2 Labeling News Articles

Due to the expected volume of the dataset, manual labelling of the news articles is unfeasible. Therefore, all news articles will be labelled as "not a security incident". A keyword/sub-string search combined with a manual review will determine which articles are security incidents. This approach means that there is the possibility of false negatives, but all positives should be true positives. Furthermore, the true positives are expected to have quite distinct keywords, making them unique. The actual negatives are expected to outnumber the false negatives massively; thus, the false negatives will only represent a tiny fraction of the negative class.

[1] `https://github.com/codelucas/newspaper`

Given the likely resulting imbalance of the dataset, this approach, while not perfect, is still a good approach given the constraints of the problem, especially since we are most interested in the positives.

An analysis of the inherent properties in the resulting dataset and possible implications on both NLP classification and event study will also be performed.

Defining Security Incident

When doing this manual labelling, the definition used will be security incident as defined by the National Institute of Standards and Technology (NIST):

> *An occurrence that actually or potentially jeopardizes the confidentiality, integrity, or availability of an information system or the information the system processes, stores, or transmits or that constitutes a violation or imminent threat of violation of security policies, security procedures, or acceptable use policies.*[2]

This definition means that security incidents include data breaches, data leaks, DDoS attacks, ransomware attacks, insider breaches, espionage, etc. Note that there will not be an attempt to determine the relationship between the companies mentioned in the security incident. As a result, victims of security incidents will represent a subset of the identified security incidents. For example, an article might mention a company assisting with the security incident or describe how a company has discovered a hacking campaign. As a result, a "security incident" label will not imply that the companies mentioned in the article are victims of the security incident. Instead, the victims will be a subset of the identified articles containing security incidents. The intention behind this is to simplify the machine learning task. Furthermore, no distinction is made with the labelling of articles concerning the same security incident. For example, one data breach might have multiple articles written about it as new information becomes available, but it only represents one security incident. This lack of granularity in the labelling can have the effect of overweighting the specific incident in the event study, as there will be events that are counted twice when calculating the average impact of all the incidents, assuming that the related articles are within the studied time window. If, on the other hand, the associated news articles are not within the studied time window, the effect would be a reduction in the aggregate impact. However, in aggregate, the negative signal should still exist as the unrelated events will be a random walk around the market return and effectively cancel each other out.

Thus the expected effect is difficult to assume as there are situations that will both increase and decrease the aggregate impact of security incidents with multiple news articles. Still, as more severe incidents are more likely to have numerous news articles in the time shortly after disclosure, one might expect the overweighting to happen more often.

[2] `https://csrc.nist.gov/glossary/term/security_incident`

3.1.3 Stock Prices

Stock prices will be downloaded from the alpha vantage API from the TIME_SERIES_DAILY dataset[3], in which the daily closing price is extracted and used as the stock price for a given date.

3.2 Data Classification

3.2.1 Creating the Classification Model

Pre-processing the Data

The text used for classification will be retrieved from the downloaded articles and will be a combination of the title + the text of the article. Typically in NLP tasks, the input text undergoes extensive pre-processing. Due to BERT usage, pre-processing techniques such as removal of stop words, stemming and lemmatization are not used. This is because BERT is a contextual model; thus, stop words and stems can provide information about the context of the text. In the paper "Understanding the Behaviors of BERT in Ranking" [10], the authors looked at what kind of "words" BERT gives attention to. The authors found that stop words receive just as much attention as non-stop words suggesting that BERT finds them important. On the other hand, the authors also found that the removal of stop words did not affect the model's performance in some tasks. Thus, pre-processing is limited to loading the text and tokenizing it using the BERT tokenizer.

The BERT tokenizer contains several options on how to tokenize the text. The tokenizer can be set to lowercase, remove punctuation, remove stop words, remove digits, remove special characters, etc. Ultimately the task of the tokenizer is to convert "human" text into a sequence of tokens which is understandable by the BERT model. Here there is the possibility to do a sweep of different tokenization options and find the best set of options. But due to time constraints, this will not be done. The tokenizer will be configured to use the uncased version meaning that all capitalization information will be removed. Furthermore, the BERT model is trained to handle 512 input tokens. This limitation means that it can not look at more than 512 tokens at a time. As a result, the text will be truncated to the first 512 tokens and discarding the rest of the text from a given article. This truncation means that the model will not be able to look at the text after the 512 first tokens, but hopefully, this will not affect performance much. Another option would be to split the text into multiple 512 token chunks and feed them to the model. Then perform averaging or a combination with the subsequent 512 tokens for each article. This would be a more complex task and would require more time, and will therefore not be explored.

Note that a character is not equivalent to a token. A token is a numerical representation of "something" that the BERT model can understand. This "something"

[3] `https://www.alphavantage.co/documentation/#daily`

can be a complete word or a part of a word. BERT uses WordPiece embeddings which were presented by Wu et al. in the paper "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation" [11]. The total vocabulary size is about 30000 tokens. This means that all the characters are converted into tokens representing one or more characters. Therefore, it is difficult to quantify the average percentage of an article that will be passed on to the model. At the very least, some news articles will be truncated when encoded. It is also worth noting that if no mapping of characters to token exists, the encoded characters will be encoded using a UNK or unknown token.

Building the Model

When fine-tuning BERT on a new task, starting with a model trained on another task is usual. In this case, we will use the DestilBERT model. The DestilBERT model was created by Sanh et al. [12] in an effort to improve the performance and reduce the size of the traditional BERT model. All the pre-trained layers are locked, and new layers are added to the end of the model. This is done to allow the model to learn the new task without having to rewrite the existing neurons. The model is configured with a dropout of 0.2. This dropout is used to prevent overfitting. This is accomplished by randomly dropping out a certain percentage of the neurons. This means that the model is forced to send the signals which represent the desired information on multiple neurons which should make the signal more robust and ensure that it looks at more of the input. The final layer contains a single neuron which is the output layer. The output layer is a sigmoid function. Usually, the output layer is a softmax function, but the softmax function is, in effect, an extension of sigmoid for multiclass logistic regression. However, since we are doing two-class logistic regression, the sigmoid function is sufficient. Sigmoid gives an output between 0 and 1, which in this case represents the probability that the text is about a security incident according to the model.

Training the Model

Once the pre-trained model has been combined with blank layers and configured for our needs, the next step is to start training the model. The goal of this training is to influence the new layers to learn to represent the new task better. This is done by running the one sample of the training set forwards through the model. Then the correct output is backpropagated through the model. There are several methods and algorithms that need to be selected when training the model. One such parameter is the optimization algorithm; examples include Adam, stochastic gradient descent (SGD), Ftrl, and RMSprop. For this model, Adam was used. Adam is a recent optimization algorithm and has shown promising results when compared to other alternatives. According to Kingma et al. [13], Adam is "*computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/parameters*". This matches the model's requirements quite well.

Another parameter is the loss function. This function represents the quantity that the model should seek to minimize during training. In this case, the loss function used is the binary cross-entropy loss function. This function is intended for binary classification applications, which is what we are doing. The binary cross-entropy loss function is given by the following equation:

$$\text{Binary Cross-Entropy Loss} = -(y \log(p) + (1-y)\log(1-p))$$

Some of the other parameters that are used in training the model are the learning rate, the number of epochs, and the batch size. The learning rate is the amount of change that is applied to the model during training. The learning rate will be set to 0.00002. This learning rate is relatively low compared to what is standard for untrained models. This is because we want to fine-tune the last layers, not re-train the model. The number of epochs is the number of times the training set is used to train the model. For this project number of epochs used will be 10. The optimal number of epochs is reached by when the validation performance starts to diverge from the performance on the training data. The batch size is the number of samples that are used to train the model at a time. The batch size used during the training of this model was 32. The effect of adjusting the batch can vary depending on the model and dataset. Some papers looking into this are; *Don't decay the learning rate, increase the batch size* by Smith et al. [14], and *Train longer, generalize better: closing the generalization gap in large batch training of neural networks* by Hoffer et al. [15]. Generally, there is some optimal batch size between 1 and the number of samples in the training set. There are several forces impacting what might be optimal. Larger batch sizes allow for greater computational efficiency(parallelism in hardware), and in convex optimization problems, larger batch sizes can lead to better solutions and a batch size of the entire dataset guarantees convergence to the global optima. However, this also dramatically increases the chance of overfitting, and this is especially true for non-convex optimization (neural networks), which are more prone to overfitting. Therefore the selection of batch size has many comparisons to training with regards to underfitting and overfitting, but it is not precisely the same. For this model, the batch size used that will be used is 32. This still represents a relatively small batch size when compared to the total number of samples in the training set. This batch size is standard, and if it is much larger, one quickly runs into hardware constraints due to GPU memory usage. The GPU will be an Nvidia GTX 1070 with 8GB of memory.

3.2.2 Evaluating the Trained Model

Evaluation of the trained model is a key step in analyzing the viability of the proposed method. Traditionally classification models are evaluated using the following metrics:

- Accuracy
- F1-score

- Precision
- Recall
- ROC AUC

The results will also be displayed in a confusion matrix. Furthermore, a comparison of the performance of the model with other techniques will be made. The different metrics are the result of the True-Positive (TP), False-Positive (FP), True-Negative (TN) and False-Negative (FN) predictions in the following equations:

$$\text{Accuracy } = \frac{TP + TN}{TP + TN + FP + FN}$$

$$F1 = \frac{2 * \text{ Recall } * \text{ Precision}}{\text{Recall } + \text{ Precision}}$$

$$\text{Precision } = \frac{TP}{TP + FP}$$

$$\text{Recall } = \frac{TP}{TP + FN}$$

Furthermore, an analysis with regards to overfitting and underfitting will be done. Overfitting is the result of the model being too specific to the training data and causing a resulting drop in performance on validation and testing data. Underfitting is the result of the model being trained enough on the data mining that there is still performance to be extracted if more training is done. This will be done by graphing the model performance on the training data and the performance on the validation data over the training epochs. Additionally, the model performance will be compared to a Naive Bayes classifier and a logistic regression classifier. Given the likely imbalance of the dataset, a more discretionary assessment will also be performed as it is highly relevant with regard to the practical viability of the model.

3.3 Event Study

To understand the impact of security incidents on stock prices and thus to discover if there are any significant changes in the stock price due to the security incident, an event study is performed. The list of security incidents generated in the previous section is used as the event list for this study, and it will be assumed that the classifier is able to classify all security incidents identified in the news articles correctly. This is due to the limited amount of data and the fact that the classifier is trained on the majority of the data. Furthermore, this assumption also allows for an analysis of the viability of the classification and event study parts separately.

An event study is, in essence, based on the efficient market hypothesis, which states that all publicly available information will be incorporated into the price of a stock as it becomes available [16]. When performing the impact analysis, traders have to effectively make a best-effort estimate of the total impact of the

event, including intangibles such as reputation and so on. Therefore, the daily changes in the price of a particular stock should reflect the perceived total impact of daily events and information on the stock's expected current and future performance. Hopefully, this analysis will remove random movements and impact from unrelated events and isolate the studied event when performed on multiple events across multiple stocks.

Fama, Fisher, Jensen, and Roll defined the traditional event study methodology in the 1969 paper *The Adjustment of Stock Prices to New Information* [16]. This methodology is essentially the same as today(with some minor modifications). This event study will follow the method described by Mackinlay, A in the paper *Event Studies in Economics and Finance* in 1997 [17]:

1. Calculate the daily abnormal returns (ARs) for each stock in the days surrounding the event. There are multiple models for calculating the ARs:
    - The Market Model, which looks at what the market return is over the same period.
    - The Constant Mean Return Model, which looks at what the market return is on average over a more extended period than the event itself.
    - The net-of-characteristic matched portfolio return adjusts the comparison portfolio such that the characteristics of the included companies are similar to the ones that make up the events.
    - An equilibrium asset pricing model which does not use stock price; instead, it uses an asset pricing model and then looks at the price changes; an example of such a model is the capital asset pricing model (CAPM).

    The AR model's purpose is to estimate the stock's expected return over the studied time horizon. Thus one can compare the actual return with the expected return to get the abnormal return.
    This study will use the market model to estimate the market return. The market model is simply the market's return, also called the market portfolio. The market model is the most common and also fits well with our data as the events can occur in a wide range of businesses with different characteristics.
    The market model calculates the expected return of stock i at time t relative to the market return at time t. This can be expressed in the following equation:

    $$E\left(R_{i,t}\right) = b_0 + b_1 \cdot E\left(R_{M,t}\right)$$

    b_0 and b_1 represents the estimation window, which is usually from -225 days to -25 days before the studied event. Their value can be found by performing a least-squares regression on the stock returns. This allows us to estimate how the stock typically moves. This, in turn, allows for plugging in the return of the market in the event window to get the predicted return of the stock using the market model. R is the return, E is the expected return, and M is the used model.

The market model gives the expected return of the stock; when comparing this to the actual return of the stock, we can get the abnormal return(AR). This can be expressed in the following equation:

$$AR_{i,t} = R_{i,t} - E\left(R_{i,t}\right)$$

2. Calculate the average abnormal return (AAR) for each day in the event window. This aggregation is done by taking the average of the ARs for all stocks to find the average AR for a given time in the event window. This is given by the following equation:

$$AAR_t = \frac{1}{N}\sum_{i=1}^{N} AR_{i,t}$$

Where N is all the stocks in the event window, this can help eliminate idiosyncratic differences in the stock returns due to individual stocks.

3. Sum the AAR for each day in the event window. This is the total abnormal return for the event. This can be expressed in the following equation:

$$CAAR_T = \sum_{t=1}^{T} AAR_t$$

The CAAR helps in understanding the aggregate effect of the stock returns. This is the total abnormal return for the event. This is particularly useful if the event window is not exclusively on the event date itself. This can be expected to be the case for security incidents in the news. This is because it is unlikely that the news agencies are the first to discover the incident. Furthermore, more information can become available without warranting a new news article.

This CAAR, along with metrics such as the T-stat, P-value, and confidence intervals, will be analyzed during the event window. Note that AR and connected CAAR numbers will be calculated using the log return, making the maths more straightforward when aggregating across periods. For minor return figures, which is expected for this event study, log returns are effectively interchangeable with percentage returns.

Chapter 4

Results

4.1 Data collection

4.1.1 List of Tickers

First, the list of tickers was downloaded from Nasdaq as shown in 4.1. Then duplicates and non-stock tickers were removed, which produced a list of 8331 stock tickers. Then each ticker in that list was used as a parameter in a search on Reuters. This search was able to find 7993 matches on Reuters, meaning that 338 tickers did not map directly to a company on Reuters. The tickers that were not found are listed in appendix A. A sample review shows that most of the tickers in appendix A represent small companies and, as a result, probably do not have any news articles on Reuters. However, there are some exceptions, particularly with shares with multiple classes where mapping listed ticker to Reuters ticker does not work. An example of this is BRK (Berkshire Hathaway) which has both A and B class shares, but Reuters does not find the company when searching on either ticker. Nevertheless, this represents a relatively small portion of the total number of tickers; therefore, it was not attempted to correct this mismatch. Instead, the list tickers used will be the 7993 tickers available without issues.

Code listing 4.1: Retrival of tickers from Nasdaq

```
# excerpt from GenData.ipynb

def downloadFTPData(ftp_url, file_name):
    ftp = ftplib.FTP(ftp_url)
    ftp.login()
    ftp.cwd("symboldirectory")
    ftp.retrbinary("RETR␣" + file_name, open(file_name, "wb").write)
    ftp.quit()


if not os.path.exists("nasdaqlisted.txt"):
    downloadFTPData("ftp.nasdaqtrader.com", "nasdaqlisted.txt")

if not os.path.exists("otherlisted.txt"):
    downloadFTPData("ftp.nasdaqtrader.com", "otherlisted.txt")
```

4.1.2 News Articles

News articles were scraped on a per ticker basis from Reuters. The implementation follows the steps described in 3.1.1. Due to limitations in how fast Reuters allows requests and the usage of a full browser client, the scraping process was relatively slow. It took between 2-3 days to scrape all stock tickers for articles and enrich them. Reuters only keeps articles for three years; therefore, all articles are in that timeframe. 4448 of the 7993 tickers had articles in the last three years. The articles were saved as JSON files, one for each ticker with the structure as seen in listing 4.2. When de-duplicating on the combination of URL and ticker, the total number of articles becomes 63126. When de-duplicating on text, that number drops to 55327. The total size of the dataset is 259,2 MB of text content.

Code listing 4.2: Example News Article

```
{
    "authors": [
        "Reuters Staff"
    ],
    "keywords": [
        "company",
        "..."
    ],
    "publish_date": "12/01/20",
    "stock_ticker": "ERJ",
    "summary": "FILE PHOTO: An Embraer E-175 jet sits outside...",
    "text": "FILE PHOTO: An Embraer E-175 jet sits outside...",
    "title": "Brazil planemaker Embraer says hackers gained...",
    "url": "https://www.reuters.com/article/us-brazil..."
}
```

4.1.3 Labeling News Articles

The data was labelled as described in 3.1.2. The keywords "Cyber attack", "Cyberattack", "data breach", "ransomware", "hacker", "security flaw", and "gained access" was used to identify articles for manual review. In total, this yielded 217 news articles classified as a security incident. De-duplicating the events on text reduced the number of security incidents to 204. De-duplicating security incidents on ticker and date reduced the number to 170. This gives a highly skewed distribution of events where only 0.4% of events are security incidents, as seen in figure 4.1.

4.1.4 NLP Classification Viability

The length distribution of the two classes is shown in figure 4.2. There is a pronounced difference between the two classes as there is a significant spike of events

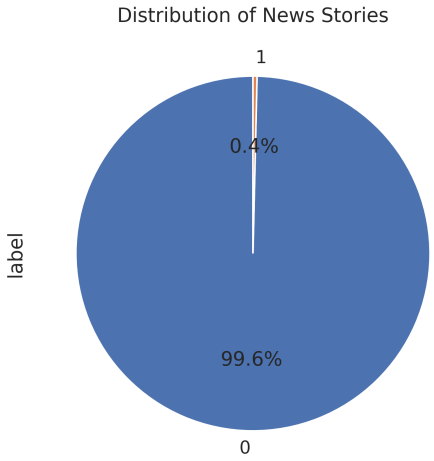


Figure 4.1: Distribution of labels in the dataset

labelled as not a security incident in the interval of 0-1000 characters. This spike is due to a particular type of article connected to earnings announcements that will never contain security incidents. Of the longer news articles, which presumably have more natural language and represent news in general, there does not seem to be much difference between the two classes.

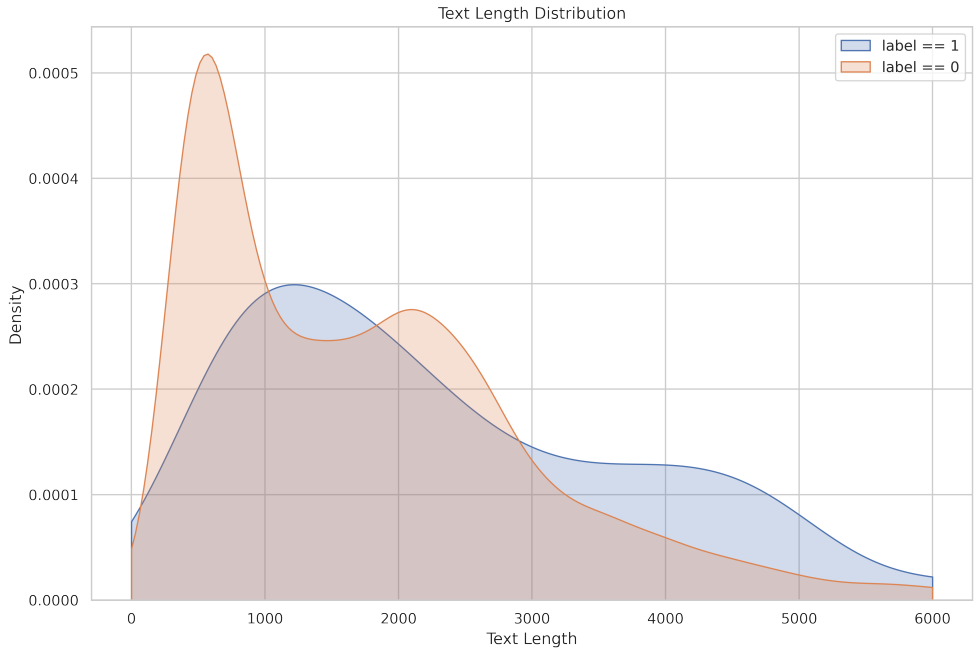


Figure 4.2: Distribution of text length for the two classes of news articles

When looking at the frequency of words in the two classes (with stopwords

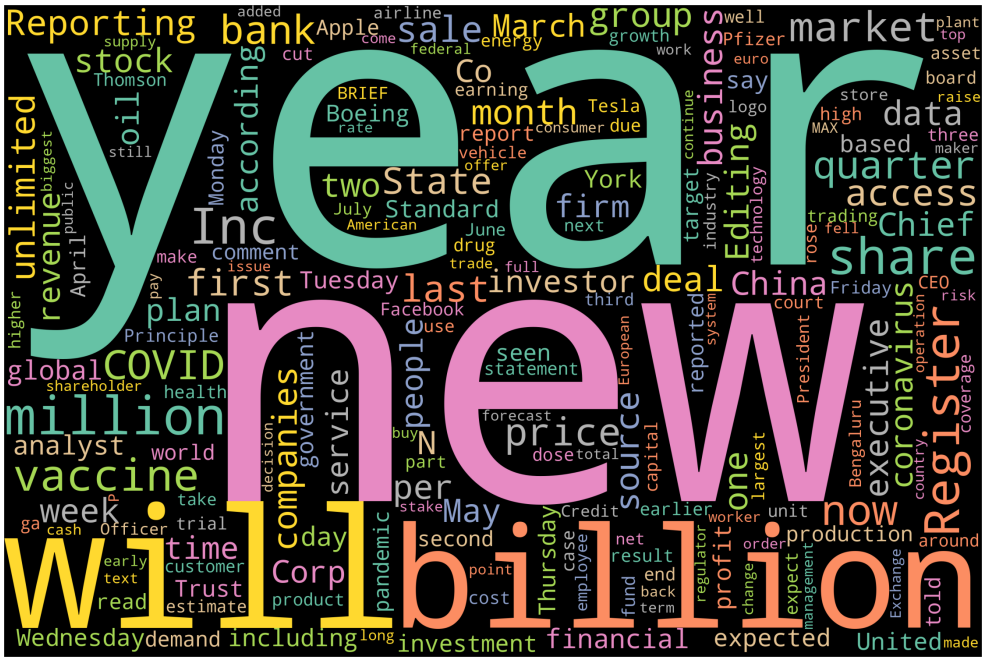


Figure 4.3: Word cloud of all words in non-security-related news

removed), we get the word clouds in figure 4.3 for non-security news and figure 4.4 for security-related news. This difference indicates dissimilarities between the two classes, which bodes well for training a classifier on the data.

4.1.5 Event Study Viability

When conducting an event study, there are only two data points which are of interest, the date and the stock ticker; an example of the data identified and processed for an event study is shown in table 4.1. One of the drawbacks of the collected data is that there is no way of knowing if the article's date represented when the security incident became public knowledge. Furthermore, there is no guarantee that the mentioned company is the victim of the security incident. Nevertheless, the article's date is used as a best-effort proxy for the date of the security incident. Another drawback is that the collected events do not necessarily represent unique events. A given event may be associated with multiple articles, and new news articles can be written years after the security incident as fallout, such as litigation, comes to a close.

Table 4.1 shows an excerpt from the collected data. In the table, we can observe that the ticker MGM has two articles within two days. This proximity of dates makes it highly probable that the events are related. Similarly, there is a high probability that several of the events associated with the MAR ticker are related to the same security incident. No attempt at correcting this will be made as the goal is to see if the automated method is viable. Furthermore, the identified

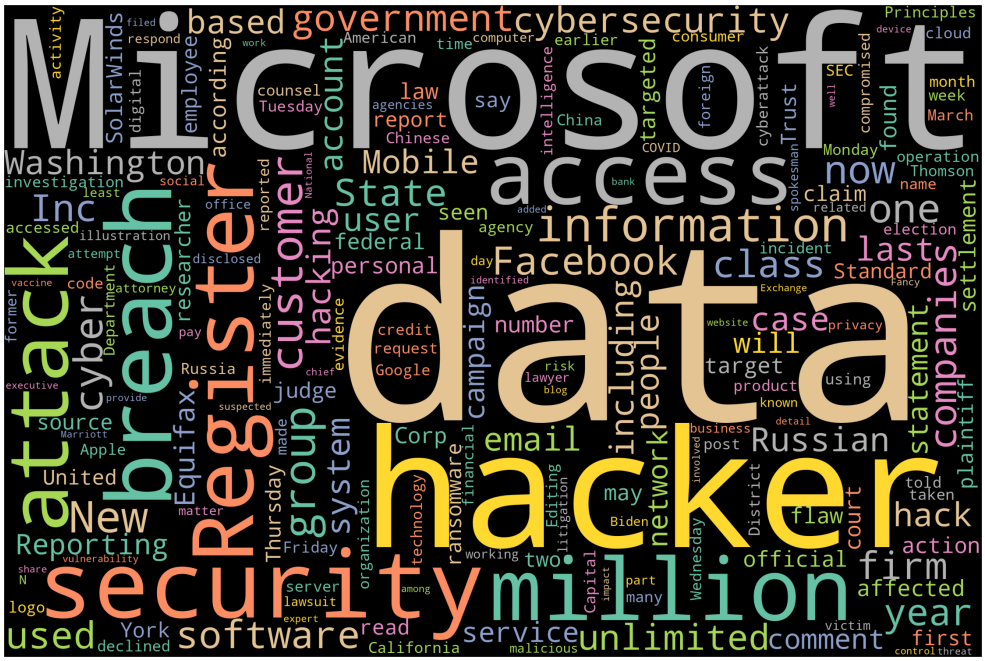


Figure 4.4: Word cloud of all words in the security-related news

drawbacks will not make the impact signal disappear. It might reduce the aggregate observed impact, but it should still be possible to observe some of the effects, as removing and isolating the random noise is one of the strengths of an event study.

Table 4.1: Table of excerpt events

Ticker	Date
HD	2020-11-24
IBM	2021-04-14
IBM	2020-12-03
INTC	2019-05-14
KR	2021-07-01
MAR	2021-03-04
MAR	2020-03-31
MAR	2019-07-09
MAR	2019-08-05
MAR	2020-08-19
MAR	2020-10-30
MCD	2021-06-11
MCO	2019-06-27
MGM	2020-02-20
MGM	2020-02-22
MIME	2021-01-12
MPC	2021-05-10
MRNA	2020-12-15
MS	2021-07-08

4.2 Data Classification

4.2.1 Evaluating Model Performance

Figure 4.5 shows the confusion matrix for the trained model. Unfortunately, the model's performance is not ideal. We can see that of the entire set; the model had a 0.07 per cent false-negative rate and a 0.36 per cent true positive rate. This performance means that model misses one out of every six security incident. Furthermore, the number of false positives is 1.92 per cent, meaning that a predicted positive is more than five times as likely to be a false positive than a true positive. It is possible to play around with thresholds to adjust the model to increase precision, but that would also increase the false-negative rate.

The test set suffers from a low sample size. The dataset was split into an 80 15 5 division between the training, validation, and test data, which means that the total number of positive samples in the test set is only 12. As a result, there is a large degree of uncertainty in the resulting performance metrics due to a single false negative or true positive massively impacting the calculated performance of the model.

The performance metrics of the model can be seen in table 4.2. We can also see the performance metrics for classification using a naive Bayes classifier and logistic regression. The confusion matrix for the naive Bayes classifier is shown

in figure 4.6. The logistic regression model is shown in figure 4.7. Interestingly, the naive Bayes model predicts all events to be negative. The logistic regression model almost predicts everything to be negative except for two samples which are predicted to be positive, one which is a false positive, and one is a true positive. This negative prediction bias is because both models do not handle the imbalance in the dataset well.

When adjusting the logistic regression model to handle the imbalance in the dataset, the model was able to improve the performance metrics substantially, as seen in table 4.2 and the resulting confusion matrix is shown in figure 4.8.

The logistic regression model performs better than the trained BERT model by avoiding the large number of false positives predicted by the BERT model. This performance advantage makes sense as more traditional statistical models tend to perform better when the sample size is smaller. In contrast, transformer models tend to gain the edge when the complexity of the task increases and the number of data samples is large.

There are two possible explanations for the performance advantage seen in the logistic regression model. Firstly, the sample size might be so small that the model cannot learn and generalize the meaning of what text constitutes a security incident. Secondly, the complexity of the data might be so low that the more straightforward statistical methods can accurately identify and model the meaning. Thus the complexity of the BERT model only results in the fact that more data is required to model the same meaning. We can see some indication of this in the word clouds in figure 4.3 and 4.4, where there is a clear distinction in words used between the two classes. This distinction suggests that a significant amount of information can be derived from the words used rather than the context in which they are used, making the classification task simpler.

Ultimately, the results indicate that state of the art NLP models can be trained to classify text concerning security incidents as stated in **h1** in 1.1, but not in a strong sense and not necessarily with good enough accuracy to be used for the proposed method of automated security incident impact analysis. Furthermore, **h2** was disproven as the better performance was achieved using logistic regression. However, a better result might have been achieved given more data or better tuning of the BERT model, but this depends on whether the result was mainly due to the simplicity of the attempted classification class of security incidents or the low sample size.

Model	**Accuracy**	**F1**	**Precision**	**Recall**	**ROC AUC**
BERT Fine-tuned	0.9801	0.2667	0.1587	0.8333	0.9070
Naive Bayes	0.9957	0.0000	0.0000	0.0000	0.5000
Logistic Regression(LR)	0.9957	0.1429	0.5000	0.0833	0.5415
LR weighted	0.9957	0.7586	0.6471	0.9167	0.9572

Table 4.2: Model Performance

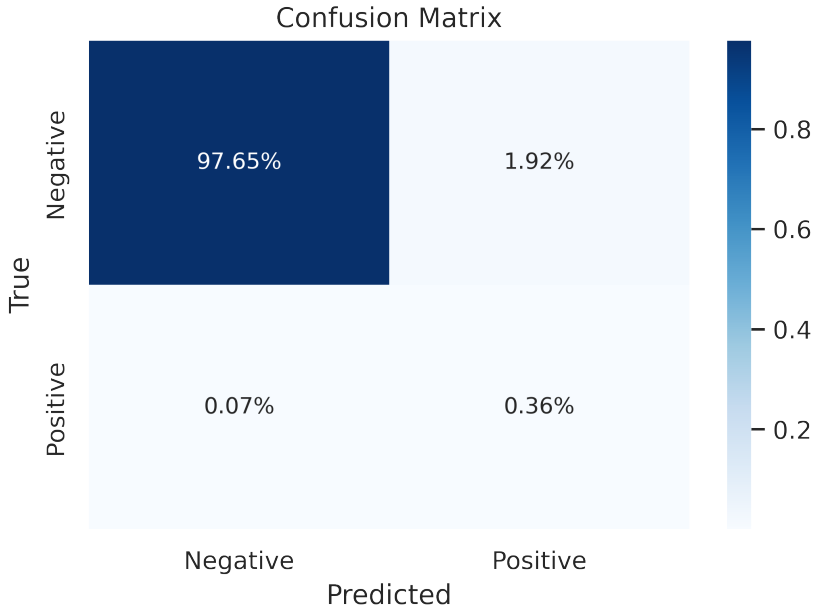


Figure 4.5: Confusion Matrix for BERT model

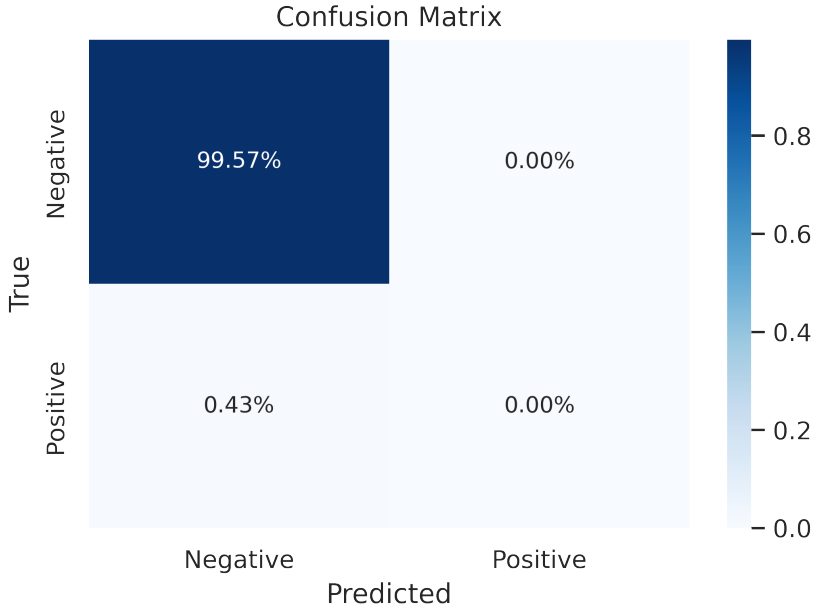


Figure 4.6: Confusion Matrix for Naive Bayes model

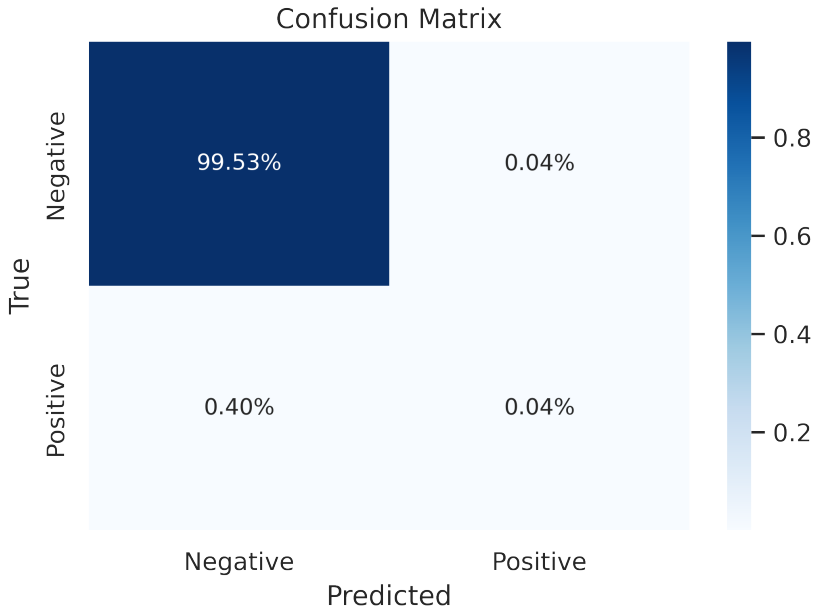


Figure 4.7: Confusion Matrix for Logistic Regression model

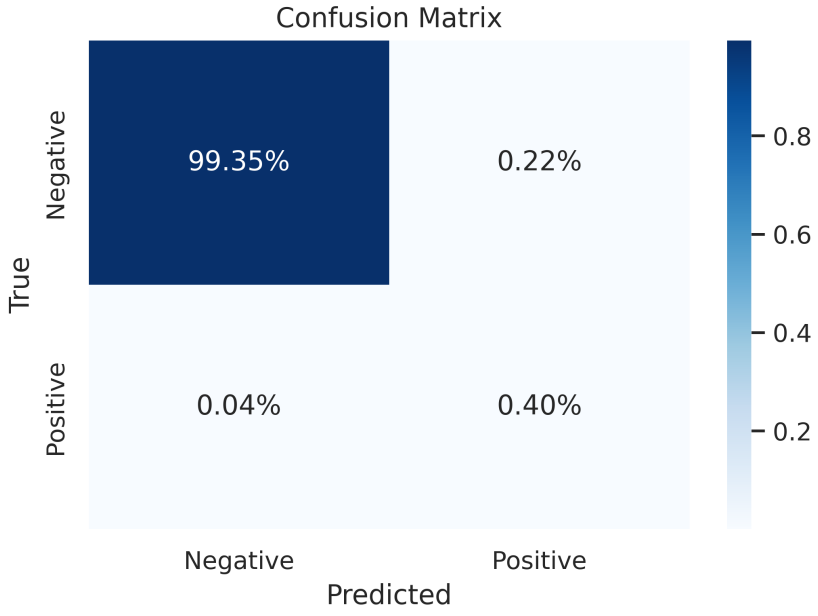


Figure 4.8: Confusion Matrix for Logistic Regression model with class imbalance adjustments

4.2.2 Model Architecture and Training

Figure 4.9 and figure 4.10 shows the accuracy and the loss of the model over the training epochs. As we can see from the graph, the model undergoes significant improvement in accuracy over the first couple of epochs. We see a gradual and steady reduction in the loss over the training period. The accuracy figure 4.9 converges and flattens towards the end of the training period. This behaviour is usually a good thing as it indicates that the model has generalized itself such that the performance on the training data is transferred to the validation data. It will be a sign of overfitting if we see the validation performance drop while the accuracy increases; this behaviour is not present in the result. The loss graph in figure 4.10 is steadily decreasing, which again is a good sign that the training is effective. It is worth noting the difference between accuracy and loss. Validation accuracy measures the differences between the model's predictions and the actual label. In comparison, loss measures the absolute difference between the predicted value and the actual label. For example, if the model predicts a 0.85 security incident in its last neuron, it will be rounded to 1 and compared to 1 when measuring accuracy; this gives no negative impact. However, when measuring loss, the absolute value of 0.85 is used, and thus the loss is 0.15 even if the prediction would have been correct. The model is training on accuracy; therefore, it is possible to have a situation where the loss is increasing, but the model's predictive accuracy is still growing.

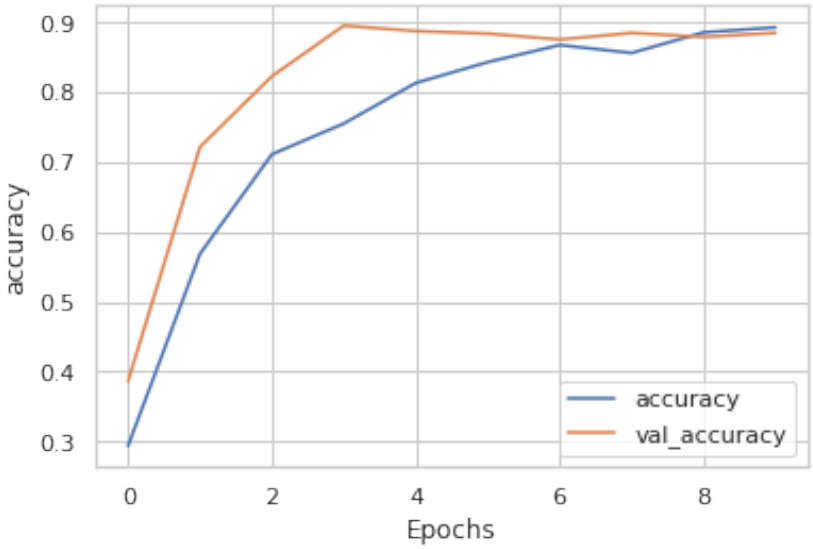


Figure 4.9: BERT Model Accuracy over Epochs

The final model architecture can be seen in figure 4.11. It consists of a single 768 neuron layer before collapsing to a single neuron layer which indicates the class. Multiple more complex architectures with additional layers were tested, but they all had equal or worse performance. This again indicates either a lack of data

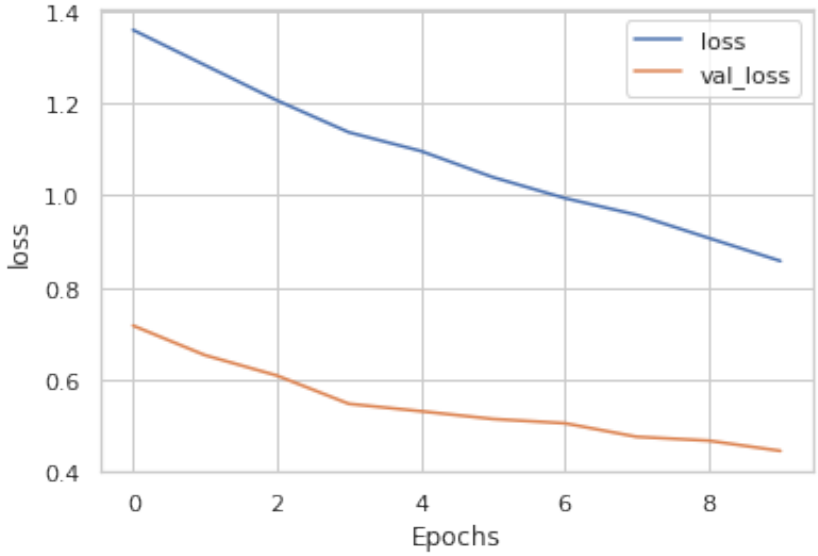


Figure 4.10: BERT Model Loss over Epochs

or a lack of complexity in the classification task.

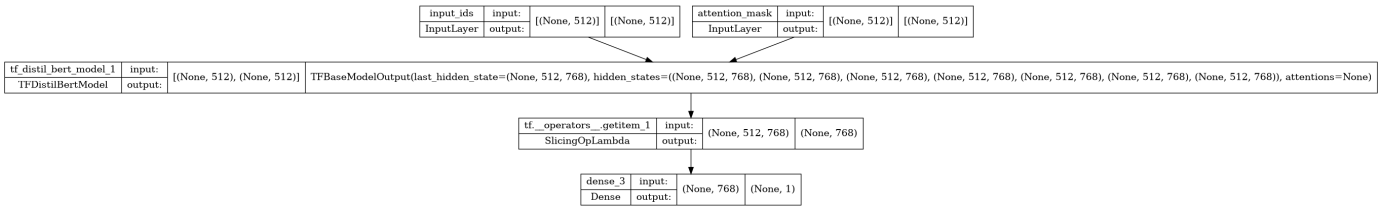


Figure 4.11: Final Bert Model Architecture

4.3 Event Study

4.3.1 All Events

Overall, the event study shows that the impact of security incidents on stock prices is negative, with a p-value of 0.01. The CAAR for all the 170 identified unique events was -2.3 per cent after ten days and -4.3 per cent after 50 days. This can be seen in table 4.3 and figure 4.12 for the 15 day event window, and table 4.4 and figure 4.13 for the 75 day event window. The asterisk behind the CAAR number indicates a significance level >99 for ***, >95 for **,and >90 for *. Interestingly, as can be seen from figure 4.12 the impact is notable sometime before the event date. There are two possible explanations for the earlier observed impact. The first is that news agencies are not the first to discover the information regarding security incidents. The second is that one security incident might contain

multiple news articles, thus shifting the aggregate observed impact on the event window timeline. Both explanations are likely to be present in this dataset. The event study also shows what we would expect regarding removing random noise. When looking at the 75-day event window in figure 4.13 the CAAR line tracks the market until it gets close to the event date, and then it starts to drop until it again stabilizes and starts tracking the market again.

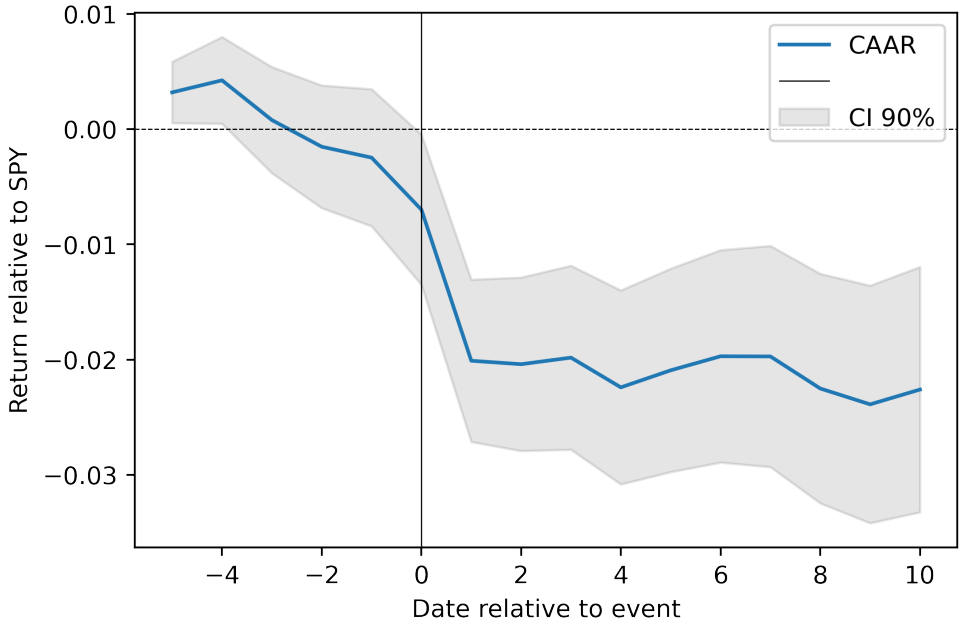


Figure 4.12: CAAR of all events. Event window: (-5, 10)

4.3.2 Big Cap Tech Stocks

A quarter of the input events are about large-cap tech firms. When reviewing the news articles that make up these events, they do not seem to represent a security incident in which the large-cap firms are heavily involved or affected. When isolating the events affecting these firms (Microsoft, Apple and Google), we get 46 events. When performing an event study on these events, we get table 4.7 and figure 4.14, which shows that there is no statistically significant impact from the events on the stock price. This observation also holds true when looking at a longer event window as can be seen in figure 4.15 and table 4.8.

Furthermore, when performing an event study with the events from large-cap tech stocks (Microsoft, Apple and Google) removed, see figure 4.16 and table 4.5. This observation of no effect also holds true when moving to longer event windows as can be seen in figure 4.17 and table 4.6. These results show that the impact of the events on the stock price is not significant for the larger firms that are heavily involved with technology. This fact also coincides with the observation that

Table 4.3: Event study of all events. Event window: (-5, 10)

Day	AAR	Std. E. AAR	CAAR	Std. E. CAAR	T-stat	P-value
-5	-0.003	0.00208	-0.003	0.00208	0.12	0.83
-4	-0.001	0.00208	-0.004	0.00294	0.15	0.71
-3	-0.003	0.00208	-0.001	0.00360	0.22	0.83
-2	-0.002	0.00208	-0.002	0.00415	0.37	0.71
-1	-0.001	0.00208	-0.002	0.00464	0.53	0.59
0	-0.005	0.00208	-0.007	0.00509	1.37	0.17
1	-0.013	0.00208	-0.02 ***	0.00549	3.66	0.00
2	-0.000	0.00208	-0.02 ***	0.00587	3.47	0.00
3	0.001	0.00208	-0.02 ***	0.00623	3.18	0.00
4	-0.003	0.00208	-0.022 ***	0.00657	3.41	0.00
5	0.001	0.00208	-0.021 ***	0.00689	3.04	0.00
6	0.001	0.00208	-0.02 **	0.00719	2.74	0.01
7	-0.000	0.00208	-0.02 **	0.00749	2.63	0.01
8	-0.003	0.00208	-0.023 ***	0.00777	2.90	0.00
9	-0.001	0.00208	-0.024 ***	0.00804	2.97	0.00
10	0.001	0.00208	-0.023 **	0.00831	2.72	0.01

actual news events did not have the large tech firms as victims or heavily affected. This connection strengthens the hypothesis that we can retrieve security incidents from news articles and that the news articles are not just random noise. Still, at the same time, it highlights a case where the assumption that a large portion of events represents security incidents of material impact on the mentioned firm is not true.

4.3.3 Reliability of Results

The results seem reliable; however, it is worth noting that the t-value is a bit misleading. This is because the security incidents impact the market a bit before the studied event date. Therefore the t-value is, in fact, more significant than what is shown in the calculations. As an example, we can see in table 4.6 that the t-value starts somewhere around two and moves to somewhere around -2; thus, the actual t-value, if the event date is adjusted to when the impact in the market appears, is going to be more significant.

The P-value is at 0.01 for both the studied event windows and the set including all stocks and the set with large-cap tech stocks removed. This indicates that the event study is reliable and that the signal we wanted to retrieve is indeed present. Furthermore, the fact that this result is not present in the large-cap tech stocks matches the expected results when looking at the contents of the news articles that make up the event study.

Hogan et al. [4] conducted an event study looking into data breaches which

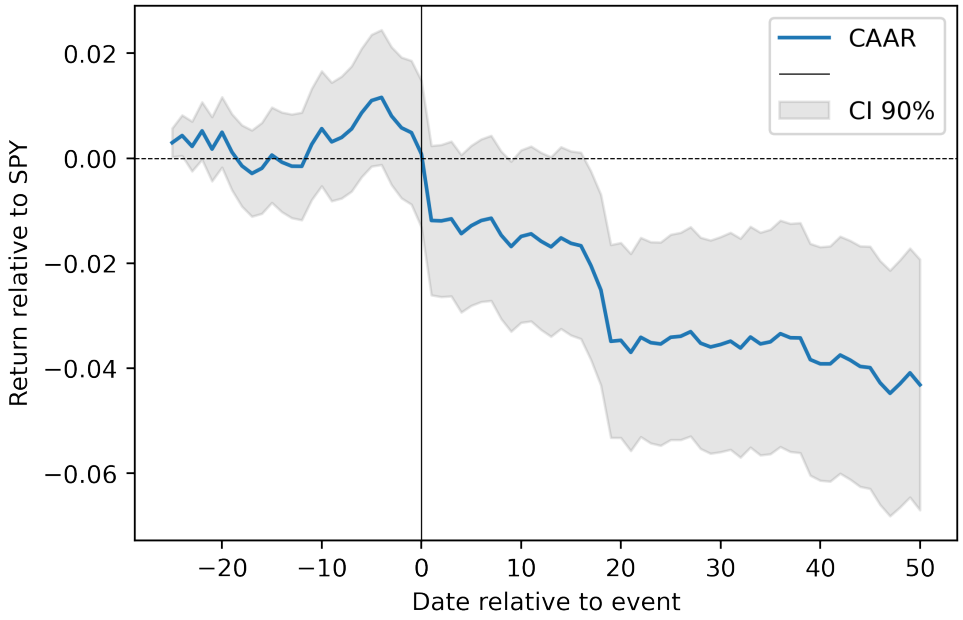


Figure 4.13: CAAR of all events. Event window: (-25, 50)

is a subset of the events that are studied here. The results can be seen in table 4.9. The loss percentage of 2% and 4% in the event windows of (-5, 10) and (-25, 50) are somewhat in line with previous studies. Hogan et al. looked at events that occurred until 2019, while the event study we are conducting looks at events from march 2019 to march 2022. Therefore there is likely little overlap between the two event studies regarding input events. Nevertheless, the results are very similar. They both have a slow initial loss, followed by a more significant loss over an extended period. The time this occurs is shorter in the event study we are conducting. One possible explanation for this is that the market is becoming more efficient and can more quickly and accurately price the impact. Hogan et al. found a more significant loss percentage is also expected. There are several possible explanations for this:

1. Companies are becoming more mature and therefore do not have as significant losses when a security incident occurs as they did in the past.
2. There is an increase in disclosure laws that are more strict than in the past, forcing companies to disclose incidents that would not have been disclosed in the past.
3. The subset of events that Hogan et al. are looking at (data breaches) is more severe in terms of costs than the broader term security incidents.
4. This event study will likely have events where the studied company was not affected, which will reduce the loss percentage.

These effects are likely to be present in the event study. However, determining

Table 4.4: Event study of all events. Event window: (-25, 50).

Day	AAR	Std. E. AAR	CAAR	Std. E. CAAR	T-stat	P-value
-25	-0.003	0.00214	-0.003	0.00214	1.39	0.16
-24	-0.001	0.00214	-0.004	0.00302	1.43	0.15
-23	-0.002	0.00214	-0.002	0.00370	0.61	0.54
-22	0.003	0.00214	-0.005	0.00427	1.22	0.22
-21	-0.003	0.00214	-0.002	0.00478	0.37	0.71
...	...	...	...	...	...	...
46	-0.003	0.00214	-0.043 **	0.01813	-2.36	0.02
47	-0.002	0.00214	-0.045 **	0.01825	-2.45	0.01
48	0.002	0.00214	-0.043 **	0.01838	-2.34	0.02
49	0.002	0.00214	-0.041 **	0.01850	-2.21	0.03
50	-0.002	0.00214	-0.043 **	0.01862	-2.32	0.02

how much each of these effects contributes to the loss percentage is not easy. Also, looking into more long-term effects is impossible due to the recency of most of the events. There is also a possibility that the news as a source will bias events to be more severe since the more severe an event is, the more likely there is for a news article to be written. Similarly, a more severe event is more likely to have multiple news articles; if these all fall within the studied event window, the expected effect is an increase in the weighting of the event by the number of articles written when the average impact is calculated.

Ultimately it appears that the event study supports hypothesis **h3** as described in 1.1 and that events extracted from news articles can be used in an event study to gain insight into the cost impact of security incidents. Furthermore, the results fall in line with previous studies, which further strengthens the confidence in **h3**.

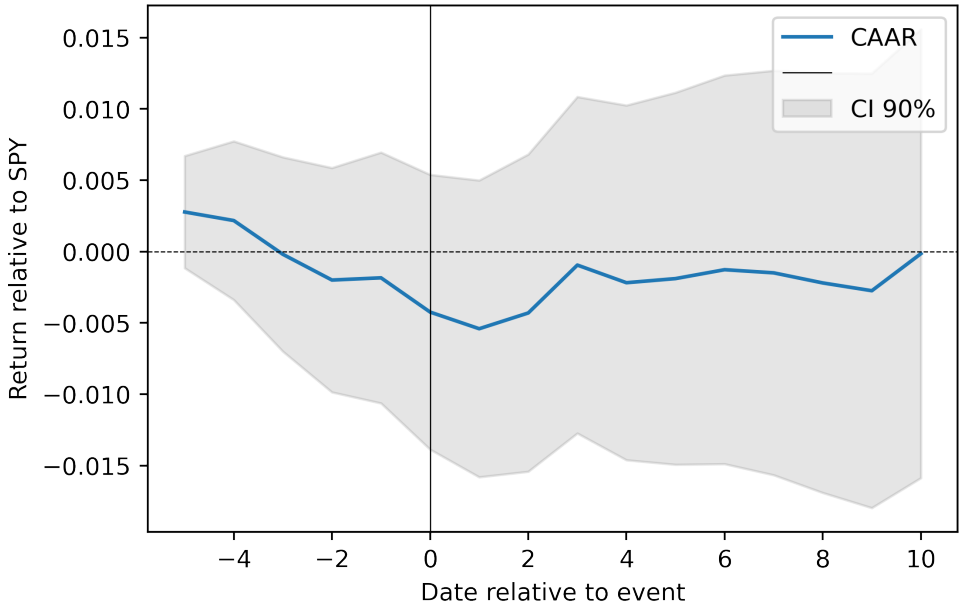


Figure 4.14: CAAR of big tech stock events. Event window: (-5, 10)

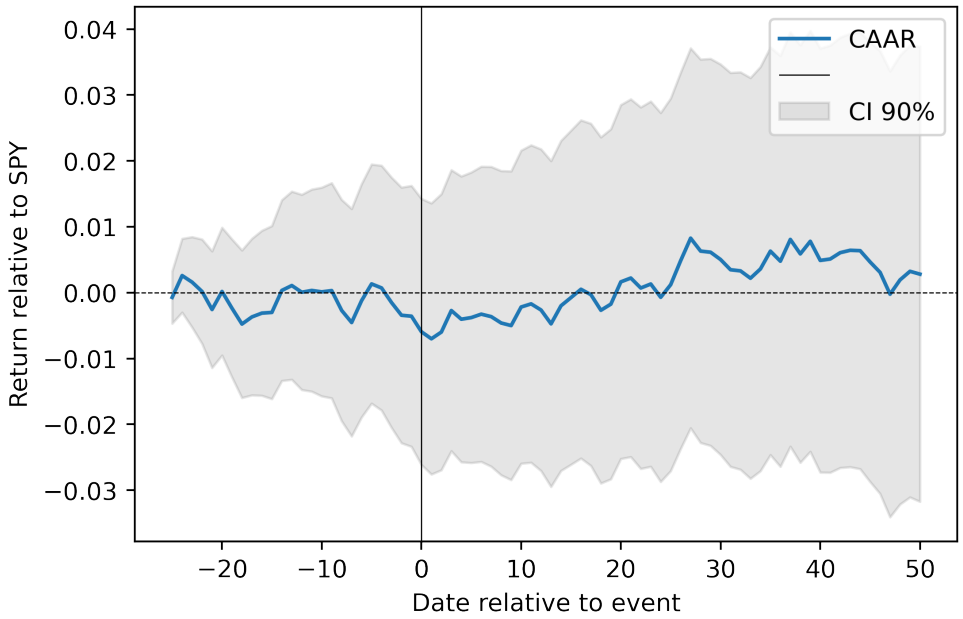


Figure 4.15: CAAR of big tech stock events. Event window: (-25, 50)

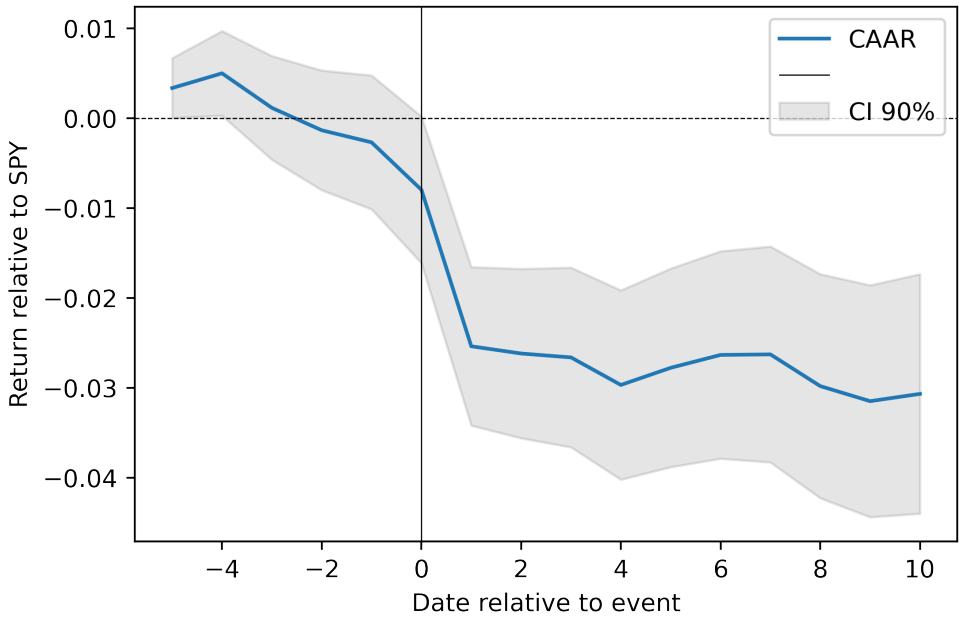


Figure 4.16: CAAR without big tech stock events. Event window: (-5, 10)

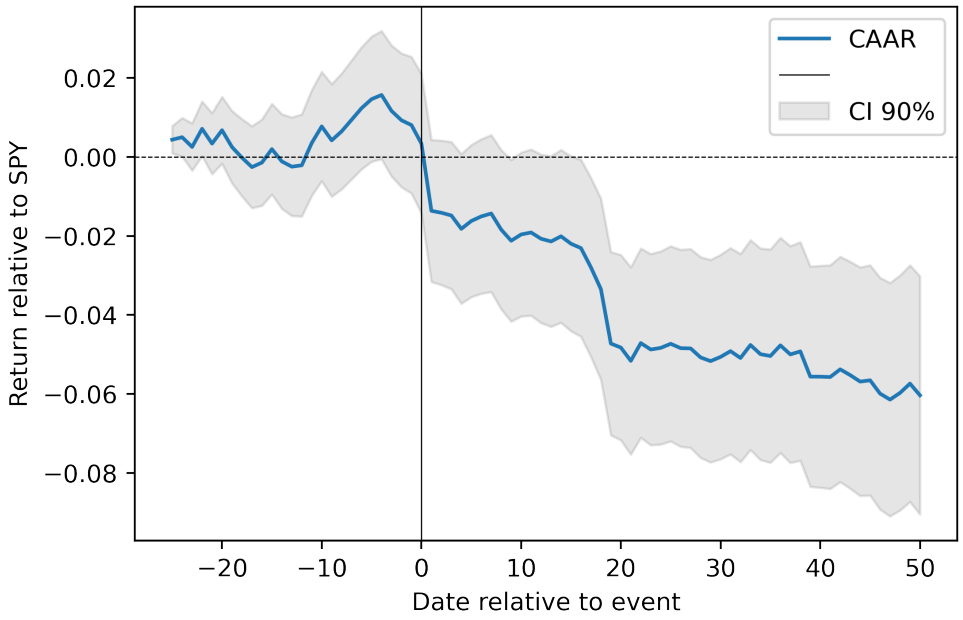


Figure 4.17: CAAR without big tech stock events. Event window: (-25, 50)

Table 4.5: Event study without big tech stock events. Event window: (-5, 10)

Day	AAR	Std. E. AAR	CAAR	Std. E. CAAR	T-stat	P-value
-5	0.003	0.0026	0.003	0.00260	1.29	0.20
-4	0.002	0.0026	0.005	0.00368	1.36	0.17
-3	-0.004	0.0026	0.001	0.00450	0.25	0.80
-2	-0.002	0.0026	-0.001	0.00520	-0.26	0.79
-1	-0.001	0.0026	-0.003	0.00581	-0.46	0.64
0	-0.005	0.0026	-0.008	0.00637	-1.25	0.21
1	-0.017	0.0026	-0.025 ***	0.00688	-3.69	0.00
2	-0.001	0.0026	-0.026 ***	0.00735	-3.56	0.00
3	-0.000	0.0026	-0.027 ***	0.00780	-3.41	0.00
4	-0.003	0.0026	-0.03 ***	0.00822	-3.61	0.00
5	0.002	0.0026	-0.028 ***	0.00862	-3.22	0.00
6	0.001	0.0026	-0.026 ***	0.00901	-2.93	0.00
7	0.000	0.0026	-0.026 **	0.00937	-2.80	0.01
8	-0.004	0.0026	-0.03 ***	0.00973	-3.07	0.00
9	-0.002	0.0026	-0.031 ***	0.01007	-3.13	0.00
10	0.001	0.0026	-0.031 ***	0.01040	-2.95	0.00

Table 4.6: Event study without big tech stock events. Event window: (-25, 50)

Day	AAR	Std. E. AAR	CAAR	Std. E. CAAR	T-stat	P-value
-25	0.004	0.0027	0.004	0.00270	1.61	0.11
-24	0.001	0.0027	0.005	0.00382	1.31	0.19
-23	-0.002	0.0027	0.003	0.00468	0.54	0.59
-22	0.005	0.0027	0.007	0.00540	1.32	0.19
-21	-0.004	0.0027	0.003	0.00604	0.56	0.58
...	...	...	...	...	...	...
46	-0.003	0.0027	-0.06 **	0.02290	-2.62	0.01
47	-0.002	0.0027	-0.061 **	0.02306	-2.67	0.01
48	0.002	0.0027	-0.06 **	0.02322	-2.57	0.01
49	0.002	0.0027	-0.057 **	0.02338	-2.46	0.01
50	-0.003	0.0027	-0.06 **	0.02353	-2.57	0.01

Table 4.7: Event study with big tech stock events. Event window: (-5, 10)

Day	AAR	Std. E. AAR	CAAR	Std. E. CAAR	T-stat	P-value
-5	0.003	0.00307	0.003	0.00307	0.90	0.37
-4	-0.001	0.00307	0.002	0.00434	0.50	0.62
-3	-0.002	0.00307	-0.0	0.00531	-0.04	0.97
-2	-0.002	0.00307	-0.002	0.00614	-0.33	0.74
-1	0.000	0.00307	-0.002	0.00686	-0.27	0.79
0	-0.002	0.00307	-0.004	0.00752	-0.56	0.57
1	-0.001	0.00307	-0.005	0.00812	-0.67	0.50
2	0.001	0.00307	-0.004	0.00868	-0.50	0.62
3	0.003	0.00307	-0.001	0.00921	-0.10	0.92
4	-0.001	0.00307	-0.002	0.00970	-0.23	0.82
5	0.000	0.00307	-0.002	0.01018	-0.19	0.85
6	0.001	0.00307	-0.001	0.01063	-0.12	0.90
7	-0.000	0.00307	-0.001	0.01106	-0.14	0.89
8	-0.001	0.00307	-0.002	0.01148	-0.19	0.85
9	-0.001	0.00307	-0.003	0.01188	-0.23	0.82
10	0.003	0.00307	-0.0	0.01227	-0.01	0.99

Table 4.8: Event study with big tech stock events. Event window: (-25, 50)

Day	AAR	Std. E. AAR	CAAR	Std. E. CAAR	T-stat	P-value
-25	-0.001	0.00309	-0.001	0.00309	0.81	0.37
-24	0.003	0.00309	0.003	0.00437	0.59	0.55
-23	-0.001	0.00309	0.002	0.00535	0.30	0.77
-22	-0.001	0.00309	0.0	0.00618	0.03	0.98
-21	-0.003	0.00309	-0.003	0.00691	-0.37	0.71
...	...	...	...	...	...	...
46	-0.002	0.00309	0.003	0.02622	0.12	0.91
47	-0.003	0.00309	0.0	0.02641	0.01	0.99
48	0.002	0.00309	0.002	0.02659	0.07	0.94
49	0.001	0.00309	0.003	0.02677	0.12	0.90
50	-0.000	0.00309	0.003	0.02694	0.10	0.92

Table 4.9: Event study results from Hogan et al. [4].

Day	CAR	p-value
-1,+30	-0.22%	0.099
-1,+60	-0.81%	0.001
-1,+180	-3.64%	0.001
-1,+250	-7.46%	0.001

Chapter 5

Future Work

The primary drawback of this study is the resulting positive sample size, which was around 200, and has the possibility of some impurity in the dataset. Therefore the most important thing to look into when iterating on this work is increasing the sample size. Particularly concerning improving the performance of the BERT model, this could provide significant improvements. Furthermore, looking into different parameters and configurations of BERT model could provide additional insights.

Another major drawback of this study is the lack of a fully integrated approach. This integrated approach would entail starting with unclassified news articles, classifying them as security incidents, and passing the positive events into an event study. This approach would also entail using test data from start to finish. Not attempting an integrated result was done due to the limited sample size. Still, it would be fascinating to examine how both the BERT and the logistic regression models perform when the raw predictions are passed to an event study where the sample size is large enough to provide a statistically significant result.

Another avenue worth looking into is revisiting the current data later and looking at longer event windows, as the repercussions of security incidents often play out over long time horizons.

The event study is quite decent, but possible improvements could be using a complete market benchmark, a Fama 5-factor benchmark, or a representative peer benchmark based on the victim companies.

Chapter 6

Conclusion

The primary goal of this thesis was to develop a method to analyse the impact of security incidents automatically. The technique developed in this thesis covers the following steps:

- How to acquire the data of which a subset is security incidents.
- How to extract security incidents from the data.
- How to compute the impact from the extracted incidents.

Overall the method shows promise and would be interesting to revisit with out-of-sample data.

Additionally and related, the thesis investigated the following hypothesises:

- **h1** State of the art NLP models can be trained to classify text concerning security incidents.
- **h2** State of the art NLP models achieve better results when classifying news articles concerning security incidents than more straightforward machine learning methods.
- **h3** News articles concerning security incidents can be used to conduct an event study and give insight into the impact of security incidents on stock prices.

Confidence in the first hypothesis(**h1**) was strengthened, but the resulting performance was worse than what would be ideal.

The second hypothesis(**h2**) was refuted, as better performance was achieved using logistic regression. This worse performance indicates that the classification task is simple or that the sample size of positive samples was too small.

Confidence in the third hypothesis(**h3**) was strengthened, and the results were in line with previous studies.

Bibliography

[1] K. M. Gatzlaff and K. A. McCullough, 'The effect of data breaches on shareholder wealth,' *Risk Management and Insurance Review*, vol. 13, no. 1, pp. 61–83, 2010.

[2] K. Campbell, L. A. Gordon, M. P. Loeb and L. Zhou, 'The economic cost of publicly announced information security breaches: Empirical evidence from the stock market,' *Journal of Computer security*, vol. 11, no. 3, pp. 431–448, 2003.

[3] E. Amir, S. Levi and T. Livne, 'Do firms underreport information on cyber-attacks? evidence from capital markets,' *Review of Accounting Studies*, vol. 23, no. 3, pp. 1177–1206, 2018.

[4] K. M. Hogan, G. T. Olson and M. Angelina, 'A comprehensive analysis of cyber data breaches and their resulting effects on shareholder wealth,' *Available at SSRN 3589701*, 2020.

[5] J. Pennington, R. Socher and C. D. Manning, 'Glove: Global vectors for word representation,' in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, 'Attention is all you need,' in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[7] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, 'Bert: Pre-training of deep bidirectional transformers for language understanding,' *arXiv preprint arXiv:1810.04805*, 2018.

[8] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, 'Language models are unsupervised multitask learners,' *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[9] J. Liu, T. Singhal, L. T. Blessing, K. L. Wood and K. H. Lim, 'Crisisbert: A robust transformer for crisis classification and contextual crisis embedding,' in *Proceedings of the 32nd ACM Conference on Hypertext and Social Media*, 2021, pp. 133–141.

[10] Y. Qiao, C. Xiong, Z. Liu and Z. Liu, 'Understanding the behaviors of bert in ranking,' *arXiv preprint arXiv:1904.07531*, 2019.

[11] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, 'Google's neural machine translation system: Bridging the gap between human and machine translation,' *arXiv preprint arXiv:1609.08144*, 2016.

[12] V. Sanh, L. Debut, J. Chaumond and T. Wolf, 'Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter,' *arXiv preprint arXiv:1910.01108*, 2019.

[13] D. P. Kingma and J. Ba, 'Adam: A method for stochastic optimization,' *arXiv preprint arXiv:1412.6980*, 2014.

[14] S. L. Smith, P.-J. Kindermans, C. Ying and Q. V. Le, 'Don't decay the learning rate, increase the batch size,' *arXiv preprint arXiv:1711.00489*, 2017.

[15] E. Hoffer, I. Hubara and D. Soudry, 'Train longer, generalize better: Closing the generalization gap in large batch training of neural networks,' *arXiv preprint arXiv:1705.08741*, 2017.

[16] E. F. Fama, L. Fisher, M. Jensen and R. Roll, 'The adjustment of stock prices to new information,' *International economic review*, vol. 10, no. 1, 1969.

[17] A. C. MacKinlay, 'Event studies in economics and finance,' *Journal of economic literature*, vol. 35, no. 1, pp. 13–39, 1997.

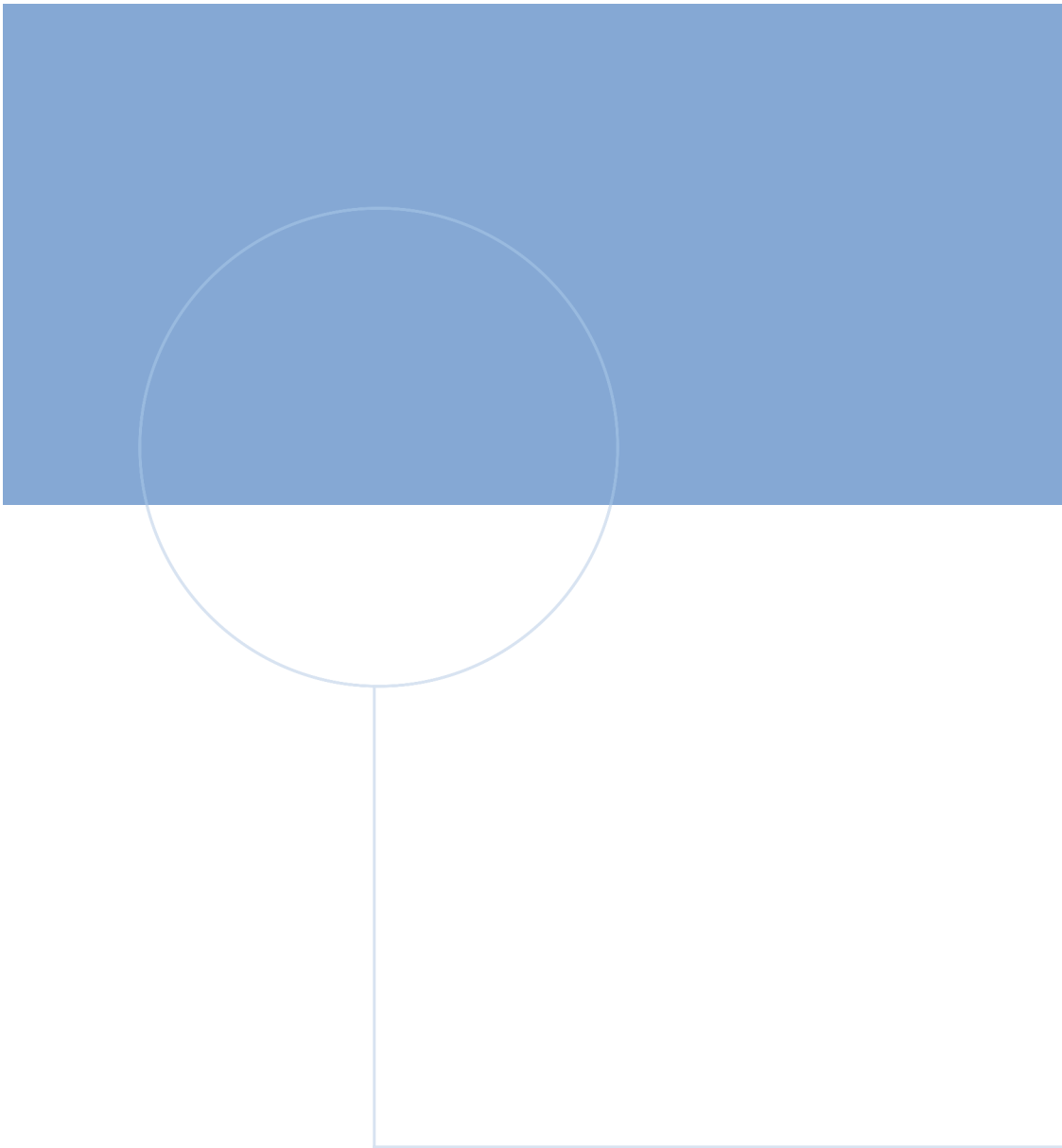
Appendix A

Listed Tickers Not Found on Reuters

- ABVC
- ADN
- AGGR
- AGRI
- AIP
- AKU
- ALRM
- ALR
- AMZN
- ANGI
- APAC
- ASAX
- ASO
- AUR
- AVCT
- AXON
- BFI
- BIOS
- BLU
- BNR
- BRPM
- BSY
- BTB
- BWV
- CALT
- CCB
- CENN
- CERT
- CHRW
- CHX
- CIFR
- CLST
- CMMB
- COOP
- CYN
- CYT
- DAVE
- DDI
- DH
- DICE
- DISA
- DSP
- EAR
- ECOR
- EMBC
- EPAY
- ERIC
- EVO
- FCNCA
- FCCO
- FBMS
- FFHL
- FLWS
- FRBA
- FSTR
- GATE
- GBS
- GDS
- GEG
- GET
- GGR
- GP
- GRAY
- GREE
- GROW
- GROM
- GTH
- HERA
- HOUR
- HYW
- IBCP
- IBRX
- IBEX
- INDI
- INAB
- INDB
- INM
- INVO
- IRCP
- ISLE
- IVA
- JD
- JRSH
- KRON
- KSCP
- KSI
- LI
- LIAN
- LUNG
- LUXA
- MACA
- MACAU
- MASS
- MAYS
- MF
- MILE
- MNMD
- MOGO
- MON
- MOR
- MQ
- MTP
- NARI
- NEXI
- NVAC
- OB
- OLB
- ONYX
- OPEN
- ORIA
- OSTK
- OTEC

- OXAC
- PARA
- PBHC
- PCOM
- PCX
- PEAR
- PECO
- PFC
- PIK
- PNT
- POET
- PRLD
- PROC
- PROF
- PRSO
- PRVA
- QSI
- RAIN
- RCRT
- RENO
- RGF
- RIDE
- RILY
- RILYG
- RILYN
- RILYO
- RILYP
- RILYZ
- RMCF
- ROSE
- SAGAR
- SANB
- SANG
- SEAT
- SERA
- SEV
- SHC
- SLAM
- SNT
- SOFI
- SOHU
- SRSA
- SSAA
- SSP
- STAB
- STRC
- STRN
- SV
- TASK
- TATT
- TCFC
- TCOM
- TERN
- TIGO
- TIL
- TMC
- TROW
- TSP
- UCL
- UK
- UPC
- VELO
- VEON
- VIEW
- VIRC
- VLON
- VS
- WIX
- WMG
- WRAP
- XM
- XTLB
- ZEAL
- ADEX
- AGM.A
- AKO.A
- AKO.B
- AKA
- ALLG
- AMR
- AMUB
- AOS
- APN
- ATCO
- ATC
- ATEK
- AUD
- AUS
- BDCZ
- BEKE
- BF.A
- BF.B
- BH.A
- BHG
- BILL
- BIO.B
- BITE
- BODY
- BRD
- BRK.A
- BRK.B
- BRMK
- BROS
- BUR
- CARR
- CARS
- CCO
- CELG^
- CIG.C
- CION
- CLAS
- CLIM
- CMCL
- CND
- COMP
- COOK
- COUR
- CRD.A
- CRD.B
- CRM
- CTK
- CWEN.A
- DC
- DDL
- DEN
- DESP
- DGP
- DGZ
- DHI
- DLY
- DMS
- DNZ
- DTC
- DZZ
- EBR.B
- ELF
- EP
- EURN
- FACT
- FBGX
- FEDU
- FNB
- FSR
- FVT
- GEF.B
- GENI
- GFX
- GIA
- GIC
- GTN.A
- HEI.A
- HIMS
- HPX
- HSBC
- HUGS
- HVT.A
- IAA
- IBO
- IDW
- IH
- IHS
- INFA
- IVT
- JILL
- JOE
- KD
- KIND
- LEAP
- LEN.B
- LEV
- LGF.A
- LGF.B
- LTH
- LU
- MDH
- MIO

- MIT
- MKC.V
- ML
- MLPB
- MOG.A
- MOG.B
- MP
- MSC
- NAPA
- NETC
- NETI
- NILE
- NTB
- NWG
- NYC
- OLO
- OPA
- ORLA
- OSI
- OTIS
- OWL
- OZ
- PAY
- PBR.A
- PICC
- POLY
- PTA
- PV
- QS
- RAAS
- RBA
- RCC
- RCFA
- RDY
- RMO
- ROSS
- SAK
- SII
- WDI
- WE
- WNS
- WOLF
- WRB
- WSO.B
- YETI
- YSG
- ZTEST
- ZTO
- ABVC
- BON



NTNU

Norwegian University of
Science and Technology