Detecting Protest Repression Incidents From Tweets

Fatma Elsafoury

Submitted in fulfilment of the requirements for the
Degree of Master of Science (Research)

School of Engineering
College of Science and Engineering
University of Glasgow

May 2019

Abstract

Protests are considered a threat to governments and political elites, that is why protesters are likely to be faced with repression. For social scientists to study protest repression, they need protest repression datasets. Currently, social scientists depend on news reports to build protest datasets and political conflict datasets. Although news reports provide a source of information that gives access to historical and international events, they have limitations like the coverage of small protest events and the delay in reporting incidents. This research explores the use of social media posts, especially Twitter, to build protest repression dataset and to overcome the limitations of using new reports. We use supervised machine learning models with a dataset of tweets that were sent during the Turkish Gezi Park protest in 2013 to detect tweets that report protest repression events. To accomplish this, we run a crowdsourcing experiment to build a training dataset of tweets and their corresponding labels as protest-related or not and violent or not. Then, we use this dataset to train two baseline machine learning models: Support Vector Machine(SVM) and Multinomial Naive Bayes(MNB) with different text representation models: Bag of Words(BOW), TF-IDF and word Embedding(WE). The empirical results of the experiments show that Crowdsourcing with the right settings and quality measures provides a fast and cheap way to hand label datasets to train machine learning models. The results also show that baseline machine learning models perform well in tweets classification tasks in terms of good AUC scores (high true positive rate and low false-positive rate).

Keywords: Protests, Violence, Protest repression, Twitter, Machine learning, Text classification, Support vector machine (SVM), Naive Bayes (NB), Crowdsourcing, Figure-Eight,

Contents

List of Tables

List of Figures

Declaration

The work leading to this Master Thesis has been conducted in the school of Computing science. This work was supervised by Dr. Bjørn Sand Jensen and Dr. Simon Roger from Computing science school and Dr. Christopher Claassen from Political science school.

With the exception of chapters 1 and 2, which contain introductory material and literature review, all work in this thesis was carried out by the author unless otherwise explicitly stated.

Acknowledgements

This work would not have been done without the help of my supervisors: Dr. Bjørn Sand Jensen, whom I learned a lot from, Dr. Simon Roger and Dr. Christopher Claassen. I also want to thank Prof. Sarah Birch for her support and advice in many situations.

A huge thanks goes to the great women in Computing science school for their support: Patrizia Di Campli San Vito (Paddy), Gozel Shakeri, Frances Cooper, Ornela Dardha, Sofiat Olaosebikan and Ekaterina Alexandrova.

Finally, thanks to family, friends, colleagues and office mates.

Chapter 1

Introduction

In Venezuela 2017, police forces and pro-government militias killed 50 people and arrested 2700 during a series of protests against the government [32]. Similar events happened during the Arab spring in 2011, the occupy movement protests in the US in 2011, Gezi Park protests in Turkey in 2013 and Euromaidan protests in Ukraine in 2014. Ortiz shows that between 2006 and 2013, 843 protests took place around the world and the majority of them were faced with state repression (violence carried out by state agents like the police or army forces) [55].

For social scientists to help human rights organisations in developing methods to reduce the repression, they need to study protest repression. Measuring and studying protest repression requires having a dataset that includes incidents of protest repression. As far as we are aware, there is no political dataset dedicated only to protest repression events. Usually, protest repression events are part of protest datasets or political conflict datasets like the Global Database of Events, Language, and Tone (GDELT), the Integrated Crisis Early Warning System (ICEWS) and the Social, Political and Economic Event Database Project (SPEED). Most of these datasets use news reports as a source of information. However, some studies argued that it is biased to depend on news stories because they have problems with coverage bias, accuracy issues, censorship and duplication [26] [19].

Since 2011, social media has played a role in spreading information about protests and protest repression in different countries e.g., Egypt, Ukraine, Turkey, Libya, Greece, Spain and the US. Social media has been used as an alternative news media, especially for the younger generation. To some extent, it overcomes some shortcomings with traditional media like censorship in some countries. However, with that potential, governments became aware of the threat and some of them took pre-cautious steps like cyber censorship in China or blocking Twitter altogether in Iran. Yet in most countries, social media provides an abundant real-time source of information with precise details like the time and the location of events [82].

In recent years, social scientists have started using automated methods like machine learning and text classification to detect events from news articles. These models save time and money, especially when coding millions of news stories. In this research, we are developing a machine learning system to detect incidents of state repression against protesters from social media platforms like Twitter. Having such a tool helps in automating the process of building protest repression dataset which in turn will help in studying protest repression. As mentioned before, studying protest repression is important as most of the protesting groups will experience protest repression at least once and because state repression against protesters is a reflection of a higher level of systematic repression [26]. In this research, we propose the use of Twitter, as a real-time source of information to detect protest repression events to overcome some of the above-mentioned problems in using traditional news media. We use crowdsourcing to build a labelled dataset to train machine learning models to automatically detect tweets that report protest repression incidents.

1.1 Research aims

This research has three main aims:

1. Developing a tool to detect protest and protest repression incidents. Such a tool is important for building a protest repression dataset which can help social sciences researchers to study protest repression as a phenomenon, to propose and to implement tools that can predict repression events in the future and help international human rights organisations to direct resources to where they are needed the most.

2. Using Social media, especially Twitter, as a near real-time, cheap and accessible source of information, to detect protest events and incidents of protest repression to overcome the drawbacks of traditional news media like coverage bias, censorship and duplication.

3. Building a machine learning systems to detects protest events and protest repression incidents from tweets. Starting from building a labelled training dataset using crowdsourcing platforms, training baseline machine learning models, comparing their performance and choosing the best machine learning model that fits our data.

1.2 Research Contribution

The contribution of this research lies in investigating the possibility of using machine learning models with social media posts (tweets) to detect protest and protest repression events.

1.3 Research Design

To achieve our research aims, we design our research in two steps:

1. **Building a training dataset:** Machine learning models learn to detect events from a text by learning a certain pattern associated with each label in the dataset. In our dataset, We have four labels:

   - **protest:** This label describes the tweets that are related to what is happening on the ground in the Turkish Gezi Park protest.
   - **non-protest:** This label describes the tweets that are not directly related to what is happening on the ground in the Turkish Gezi Park protest. For example, if the tweet is talking about the Turkish elections and using the protest hashtag, it is considered non-protest.
   - **violent:** This label describes the tweets that report violent incidents regardless if they are related to the protest or not. By violent we mean physical violence like shooting or beating up.
   - **non-violent:** This label describes the tweets that do not report physical violent incidents.

   Each tweet in the dataset received two labels a protest label either **"protest"** or **"non-protest"** and a violence label either **"violent"** or **"non-violent"**. We use the combination of these labels to decide if a tweet reports an event of protest repression or not; for example, a tweet labelled as **"protest"** and **"violent"** is considered a tweet that reports an event of protest repression. While a tweet labelled as **"non-protest"** and **"violent"** means that the tweet reports a violent event but not protest-related. In other words, the tweet does not report a protest repression event. On the other hand, if a tweet is labelled as **"protest"** and **"non-violent"**, then the tweet is related to the protest but does not report a protest repression event. In case of a tweet labelled as **"non-protest"** and **"non-violent"**, then the tweet is not related to the protest and does not report a violent event.The last type of tweets are the ones we used to teach the model the pattern in the negative examples (do not report protest repression and are not protest-related).

   To make sure that the model learns these patterns, we feed it with a sample of data that correspond to each label. This step is aimed at building a training dataset with a high enough number of training samples that allows the model to distinguish between different patterns and different labels. To carry out this task, it consumes time and money to hire an expert (people who are familiar with the protest and protest repression concepts) to do the labelling.

To address this issue we crowdsource the task to crowd workers online after giving them instructions and examples of how to label the tweets. This way we save time as more than one person will be working on the data, save money because it is cheap to hire crowd workers compared to students. By hiring more than one person to do the same tweet and by implementing quality assurance mechanisms, we can hypothesis that the crowd workers would give data of high enough quality to be used in training the machine learning model. However, given that the data are tweets, which are unstructured human-generated text, and they have some grammatical and misspelling mistakes, we expect incidents of confusion, disagreement and mistakes in labelling the tweets, especially that the crowd workers might not be familiar with some idioms related to the protest event in Turkey.

2. **Text classification models:** The second issue to address is what is the best machine-learning model that fits our training dataset and will be able to generalise to new unseen data in our tweets collection. We try two of the baseline models that are known to perform well in text classification tasks: Support vector machines (SVM) and Multinomial naive bays (MNB). We investigate the performance of each model with different parameters and different text representations.

   Then, we compare the performance of the two models. Again given the short nature of tweets with the grammatical and misspellings, we expect the model to relate some words to certain labels even if this is not the case all the time. This will lead to cases of misclassifications. We investigate how strong the influence of these cases on the general performance of the model is and if the models are good enough to be used with new data to detect protest and violent tweets as a first step to detecting the events.

1.4 Research Questions

To work towards our research aims, we need to answer the following research questions:

- RQ1: What is the agreement level internally between the crowd workers on labelling the data? Meaning that for the same tweet how many crowd workers agreed on giving the same label to the tweet?

- RQ2: What is the best baseline machine-learning model between SVM and MNB to classify tweets as protest/non-protest tweets or non-protest/non-violent tweets?

1.5 Thesis Statement

We are investigating the possibility of social media posts especially tweets that are sent during protests to be used as an alternative source of information to detect protest repression events instead of traditional news media. We use machine learning models to detect the tweets that report protest repression. This is illustrated through the case study of the Gezi Park protest in Turkey in 2013. By comparing the days when the number of violent tweets peaked to the actual violent days of the protest as reported in news media, we found that the violent tweets detected by the model are responsive to what happened on the ground during the protest.

1.6 Thesis Outline

This thesis is composed of the following chapters:

- **Chapter 2 - Background:** This chapter starts with defining protests and conceptualising state repression against protesters by reviewing the literature. Then, it provides background on how machine learning models work by explaining the text classification pipeline and the different approaches for performing each step.

- **Chapter 3 - Data collection:** This chapter describes the data collection of tweets from the Turkish Gezi protest 2013. Then, we describe the crowdsourcing experiment used to create the labelled training dataset. We analyse the results of the experiments in terms of the agreement between the crowd workers. Then, we investigate how to detect the reliability of the workers and the accuracy of their labels. Finally, we provide examples of the mislabelled tweets by the crowd and why does that happen, the task design and the workers. This chapter answers the first research question.

- **Chapter 4 - Text classification:** This chapter describes two text classification steps: the first is to detect protest-related tweets and the second is to detect violence-related tweets. For each step, we run a group of experiments to find the best parameters and text representations that fit the data and gives the best performance. We run this group of experiments with both models SVM and MNB. Then we compare the performance of the two models to decide which model, which parameters and which text representation to move forward with for protest classification and violence classification. Finally, we provide a basic analysis of the tweets collection after predicting their labels. This chapter answers the second research question.

- **Chapter 5 - Conclusion:** In this chapter, we highlight the contributions of this thesis, summarise the experiments, provide answers to the research questions and discuss challenges, recommendations and future work.

Chapter 2

Background

This research is interdisciplinary between the two fields political science and computing science. The political science part is concerned with the research aim which is detecting protest repression events from tweets. On the other hand, the computing science part is concerned with the method of how to achieve that research aim. To carry out the research we first to define and explain certain concepts from the two fields and how they interact together.

In this chapter, we introduce these concepts starting with defining what a protest is and what protest repression is. Then, we introduce the conceptual framework of protest repression, which details the characteristics of the events we are interested in. This framework has three dimensions: the actors (perpetrators and victims), the action (killing, shooting or arresting) and the characteristics of the action (public, private, direct or indirect). In other words who did what to whom when and where. Then we review the body of literature on the existing political datasets, what are the used sources of information and extraction methods. We also provide the pros and cons of each of these sources and methods and the potential in using social media and machine learning to detect the protest repression events.

The second part of this chapter deals with the computing science part of this research. It explains concepts like machine learning (ML), text classification and feature extraction. Then, we explain the main steps of the text classification process and the methods and techniques used in each step. Finally, we explain the performance measures and statistical tests used to evaluate the used models.

2.1 Protest Repression

Throughout history and across nations, we can find several incidents of protest repression, e.g. the civil rights protests in the USA in the 60s and the 70s. The repression takes different forms and is done by different actors. In this section, a typology of state repression toward protests is

covered. A widespread definition of protests and social movements repression is introduced by Stockdill who defines it as "any actions taken by authorities to impede mobilization, harass, and intimidate activities, divide organizations, and physically assault, arrest, imprison, and/ or kill movement participants" [75]. However, Jennifer Earl believes that this definition is limiting as it ignores further conceptualization of repression. According to Earl the use of this definition led the research in repression to focus on the severity of repression more than the type of repression like it is done in [21]. Earl finds Tilly's definition is more inclusive to different types of repression. Tilly's definition is "repression is any action by another group which raises the contender is the cost of collective action" [80]. Earl uses Tilly's definition to conceptualize repression based on three key dimensions: Actors, action and visibility [24]. In the following section, we describe this framework in detail.

2.1.1 Protest Repression Typology

Who/Actor (The identity of the repressive agent): The actor of the repressive action could have different types of connections to the state and the political elites. Earl distinguishes between three types of repression actors based on their connection to the state:

- **State-actors:** actors that are directly connected and controlled by the state and the political elites like military and police agencies in authoritarian regimes or national police agencies in democratic regimes. For example, the Irish civil rights protests for Northern Ireland are a minority Catholic community in the mid-60s, one of the actors of the violence that took place was the state 's security forces [88].

- **Loose-state-actor:** actors who are not directly controlled by the state and political elites like local police agencies in the United States. For example, the violence done by the Southern sheriffs against the civil rights marches in the USA in the 50s and the 60s [11].

- **Non-state-actors:** private citizens or groups who are a countermovement. This group could be indirectly hired by the state/political elites. They also could be an independent countermovement. For example, the pro-state Protestant paramilitary who used violence against the Catholic civil rights protests in Ireland because they were afraid that the Catholic gains come at their expenses [88].

How/Action (The character of the repressive action): Here, Earl uses two contrasting models to characterize a repressive action:

- **Coercion:** is the use of violence in the form of police and military action like violent harassment(intimidation), Violent repression (kidnapping or assassinations) or violent campaign(mass physical killings) [76]. For example, Between the mid-60s and the late 80s, Military dictatorships used violent repression in the form of torture, murders and

Dimension	Type		
Actor	State-actor (e.g., police or military)	Loose-state-actor(e.g.,local police department in the US)	Non-state-actors (e.g., counter-movements)
Action	Coercion (e.g., the use of tear gas)	Channelling (e.g., restrictions on 501 social movement organizations)	
Visibility	Observable (overt)	Un-observable (convert)	

Table 2.1: Protest repression typology. The shaded cells are the characteristics of the protest repression events this research is interested in.

disappearance in Chile, Uruguay and Argentina [46]. Violent harassment actions like arrests were the main coercive actions in the USA states in the 50s and the 60s [11].

- **Channeling:** is constraining the ability of movements to organize protest events by regulating a key resource flow to the movement. Taxes on a non-profit organization is a form of restriction. For example, the obstruction of funds to the Catholic church in Chile [46]

What (The observability of the repressive action): Although there is a spectrum of levels of the visibility of a repressive action, Earl simplifying it by using the two extremes on the spectrum:

- **Observable:** When the repressive action (usually exercise) is meant to be visible to the protestants and the public. All the violent actions used in the examples above are observable [50].

- **Unobservable:** When the repressive action (coercive or channelling) is meant to be unknown to the public. For example, the tax regulations in the USA are applied to social movement organizations (SMO) and non-SMO alike. However, it diminishes the activities of SMOs [50].

Table 2.1 summarizes the typology set by Earl for social movements repression. The shaded parts of the table are the interest of this research. As mentioned before, we are interested in protest repression incidents. These events are the ones that are Coercive, overt and carried out by State-actors.

2.1.2 Collecting Political Events

From the definition and the typology of protest repression, we can find the main elements that form a protest repression event. The definition can be broken down in the following way: repression is any action (what is the action?) by another group (who is the actor of repression?) raises

the contender's (who is the victim?) cost (how did the repression take place?) of collective action. To detect protest repression events, we need to extract the answers for these questions from the available data. Two more important elements to be extracted are: the location and the time of the event. In this section, the literature is reviewed for the different sources of information used by academic researchers to detect protest repression events, the events extraction methods, the coding methods and the existing political conflict datasets.

Surveys

Surveys are the traditional way to collect information in social sciences. They were also used for collecting information about protest repression. For example, [9] used different surveys to collect information about three protests in Turkey, Brazil and Ukraine. They used surveys provided by third parties. The first survey is the Konda survey from Turkey during the Gezi protest 2013. The survey asked 4000 demonstrators and was conducted in Gezi Park. In Brazil, they used surveys provided by Datafolha, which also asked around 4000 protesters on-site in San Paulo city in Brazil 2013. The survey took place after the first repression cycle and at the peak of the subsequent uprisings. In Ukraine, they used a survey conducted by Kiev International Institution of Sociology (KIIS). The KIIS survey included 1000 protesters in the Maidan Square during the Euromaidan protests in Kiev, Ukraine in 2013. The surveys asked questions about the actors, victims and actions. The repression is expressed in answers like "seeing repressive acts of the police" when the participants are asked questions like "What was the most important reason to join the protests?" or "why are you here? ".

There are advantages and disadvantages when using surveys to collect information. The advantages are: they provide a relatively fast and inexpensive data collection process giving access to a wide range of participants. Surveys are accurate if sampling is probabilistic and ethical. On the other hand, the disadvantages of using surveys are: it is expensive to cover wide geographic areas and to ensure representative data, data may not be valid because of self-reporting problems, poor sampling or response bias, it is difficult to find surveys to cover different time periods [48].

Most of the scholars studying protests and social movements use news wire reports like Reuters, BBC, Agency France Press (AFP) as their source to detect political conflict datasets. Social Conflict Analysis Dataset (SCAD) includes protests, riots, strikes, inter-communal conflict, government violence against civilians and other forms of social conflict. It relies on news reports from The Associated Press (AP) and Agency French Press (AFP). It covers areas like Africa, Mexico, the Caribbean and Central America. It has around 20,000 social conflict events between the period of 1990 and 2015 [67]. The Uppsala Conflict Data Program (UCDP) is another dataset that provides data on organized violence and armed conflicts. It covers events between 1989 and 2018 from different parts of the world. It uses a mix of news wire reports like Reuters, BBC, Agency France Press (AFP) and newly published books and reports [31]. [26]

uses daily editions of the New York Times as a source to build their dataset collection of 1905 protest repression events. They focus on news reports between 1968 and 1973 in the state of New York. Similarly, [14] uses independent newspapers and recorded TV interviews with some of the participants in the protests in Egypt during the period of 2007-2011.

The main advantage of using new papers as a source of information to detect protest events is that they provide information on historical and cross-national protest repression events. On the other side, the disadvantage of using new paper as a source of the dataset according to the media is a business which means that some protests might not be covered because the event is small. There were some issues related to the accuracy of the report as sometimes the reporter does not witness the event but is informed of instead. The freedom of the press varies from country to another. Sometimes certain protests are not covered or reported from the perspective of the state.

Social media have been widely used during the Arab Spring, the Occupy movement, the Gezi protests in Turkey and the Euromedian protests in Ukraine [77] [57] [25] [8]. In studying the role of Twitter players during protest events, Earl et al., analyzed a dataset of 30,296 tweets collected during the protests surrounding the G20 meetings held in Pittsburgh 2009. The protests faced state repression in the form of the arrest of over 100 participants, the use of smoke canisters and firing rubber bullets. The tweets used the hashtag #G20 and collected in the week of the meeting on 24-25 September 2009. They found that Twitter is more used during the protest event than other social media platforms like Facebook which might be used prior to the protest to spread information and collect people. Twitter is mainly used to broadcast information and updates on what is happening on the ground.

Another interesting finding is that Twitter is used to report information about police and protest policing (repression) [25]. The same findings are supported by Poell and Borra in their study of using Twitter, YouTube and Flicker as alternative news sources during the same G20 protest event. They found that 57% of top retweets about the G20 reports police activity. They also argue that Twitter is the most promising social media platform for crowdsourcing alternative reporting [57]. Similar results were found in the Turkish Gezi protests 2013 and the Ukrainian Euromedian protests 2014 where the number of tweets bursts increased significantly with the police attack on the protesters. In the Turkish case, on 31/05 and 01/06 when police forces used tear gas and water cannon against protesters. The number of tweets related to the protest on those days reached 3,500,000 tweets compared to 96000 likes on the Facebook pages related to the protests. In Ukraine, the 24th of January was when the first fatalities among the protesters took place by police repression. The Number of tweets related to the protests on that day reached 200,000 when the number of Facebook likes was 1,561 [82].

Although Twitter has not been used to detect events of state repression against protests, it has

been used in detecting sports, musical and political events like presidential debates [61]. It was also used in situational awareness event detection like [66] which used Twitter to detect earthquake events and it could report detailed information about earthquakes even before the national TV in Japan.

Pros and Cons:

One of the main advantages of social media plays an important role in news reporting when the state censors what is really happening on the ground during protests like in the Turkish Gezi protests and Egyptian Tahrir square protests. Social media was even used by some Human rights organizations like Amnesty International as a source of information [8]. Moreover, with wide internet access and mobile phones, social media posts are almost real-time. The accessibility nature of social media allows people to report events in small villages or events that do not attract news wire attention. The existing facilities that social media platforms offer like enabling geo-location and date and time provide a level of granularity when detecting the events. On the contrary, there are some disadvantages:

- **Cost:** The availability of a big number of tweets is an advantage and disadvantage at the same time because having a big number of data to process costs huge processing machines and complex machine learning algorithms to extract the important information from the tweet. This could be overcame by the revolution in the hardware industry where GPU and big servers became cheaper to use.

- **Location:** Although the availability of geolocation function on Twitter, only around 0.85% of all the tweets use this feature which makes it harder to extract location information from the tweet [72]. There are some trials to work around this by using the content of the tweet to extract location information, the use of the IP address or the use of the user's location from their profile's information [56] [86] [17] [66].

- **Pre-processing:** Tweets uses slang language, emoticon and grammatical and spelling mistakes. Most of the tweets are retweeted which leads to duplication in the dataset. This needs pre-processing steps to remove duplication and to clean the noise.

- **Repetition:** The same event could be reported more than one time by different people either by re-posting the same tweet or tweeting the same event many times. This again leads to duplication in the dataset.

- **Validation:** Social media is a space where rumours can spread quickly. A validation step needs to be taken to verify the accuracy of the reported events [12].

2.2 Text Classification

Starting this research, we did not have ground-truth data for incidents of violence during protests. We did not know how many tweets reporting violence would be sent during protests. We could not use unsupervised machine learning, as this needs lots of data points. That is why we chose to build a training dataset and to use supervised machine learning models. Text classification is a process with certain predefined steps Figure 2.1 shown the blueprint of the text classification process. In this section, we explain the main steps of a workflow for a supervised text classification system.

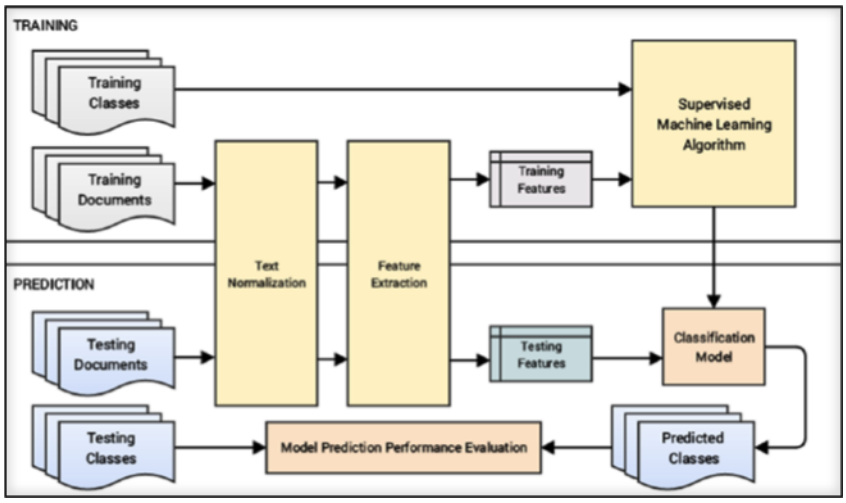

Figure 2.1: Text Classification Pipeline [68]

2.2.1 Data Preparation

Data is one of the main components in any machine-learning model. The data affects the model's performance. After collecting the data, it needs to be ready for training and testing the machine-learning model. There are three steps to prepare the data.

Pre-Processing

Because in text classification task we deal with human generated text, we need to clean the text before starting the training process and this step is called pre-processing or text normalization. The pre-processing step helps reducing the size of a document by removing unnecessary words. There are general main pre-processing tasks that are shared by all text classification tasks and other more specific ones related to the types of text. Here, we preview the general steps and later we will explain the specific pre-processing steps for tweets.

1. **Tokenization:** Words are the basic unit in a text classification task. The first step step in pre-processing is to divide each sentence in the document into words (tokens). Although

this task may seem easy, it is challenging for computers. This task depends on the language and what are the delimiters in this language. For example, a language like English, tokenization algorithms use spaces and some special characters like () < > ! ? . , : ' as delimiters. Some of these special characters are tricky sometimes. For example, the special character (.) dot could be the end of a sentence or could be used with abbreviations like U.N. or colons between number 3:30. In such cases, these special characters will be treated as delimiters, which means extracting the wrong tokens. The solution is to consider the type of text we have and use the most suitable tokenizer (e.g. smart tokenizers which can identify the abbreviations) or customize the tokenizer to the fit the data to be classified [87]

2. **Removing Special Characters:** As discussed in the previous section, special characters could be a source of confusion for the algorithms and in text classification task; there is no information for the model to learn. It is best to remove them especially, for text structure like tweets, full of mention symbols @ and hash tags #.

3. **Removing Stopwords:** Stopwords are the words that are most often occurring in nearly every document in any language for example; in English words like is, the, a, an, but, if, his, etc. Removing these words help in reducing storage space and speeds up the training process [2] [87].

4. **Stemming and Lemmatization:** This step reduces the words groups in a document by having less word variations. For example, the word "book" and "books" belong to the same group so it is better to remove the s from books and treat them as one word. There are two types of stemming inflectional stemmers (morphological analysis) and to-root stemmers (lemmatization). The first type (inflectional stemming) is used to normalize the plural to singular and past to present. The to-root stemmers (lemmatization) normalize the word to its root by removing the suffixes and the prefixes. For example, the word "meeting" is normalized to "meet". There are some popular stemming algorithms like Porter stemmer and Snowball stemmer. Lemmatization is more complicated as it needs more information to perform well [2] [87].

5. **Case Normalization:** This task converts the entire document to either upper case or lower-case. This makes the upper-case word at the beginning of a sentence treated as the lower-case words in the middle of a sentence [2] [58].

Train/Test Split

To measure the model's performance, we need to make sure that the model is capable of generalizing the learned pattern from the training dataset to new unseen data. If the model learns only the pattern in the training dataset and fails to detect similar but not exactly the same pattern in

new dataset, then the model would be over-fitting. This means that the model fails to generalize well to the new data. On the other hand, if the model fails to detect any pattern in the training dataset, the model would be under-fitting. To test the model is ability to generalize, we must test the model on a labelled dataset that is new to the model. To do that we split the original training dataset into train set and test set. In the literature, it is recommended to use 70% of the original labelled dataset to train the model and 30% to test the model.

Cross Validation

When the dataset is small, it is difficult to have a test set that is representative of the data. Cross validation helps to over come this problem by dividing the training dataset into K equally sized blocks. For K times the model is trained on all the blocks except for one $K-1$ then tested in the left out block. Then the average of the performance measure is taken over the K times.

Feature Extraction

As mentioned before, during the training process, the model learns the unique attributes and characteristics associated with each class/label. These attributes are called features. The third step of the text classification process is to extract the features of each document and feed them to the model along with the corresponding labels. For example, features could be a group of words like “protest”, “GeziPark” and “Police”, these words most of the time are associated with the label “protest”. Machine learning models do not understand the textual format of documents. To extract features from documents, we must first transform the textual features into numerical features. One of the basic concepts in feature extraction is the Vector Space Model (VSM). VSM is a mathematical model for transforming and representing text documents as numerical vectors. Here, we describe three models for representing text data in the corpora:

Bag Of Words model (BOW): The BOW model is a simple and effective model in which all the unique terms in the corpora are extracted and converted into numerical vectors. These numerical vectors represent each term in the corpora. There are three ways to represent the terms:

- **Binary representation:** In which each word is represented by a binary value 1 or 0. This number indicates the occurrence of each term in each document [87].
- **Integer Count representation:** Similar to Binary representation but relapsing the 1s with the number of occurrences of a term in each document [87].
- **Weighted model:** It replaces a term count by a term weight, which represents the importance of the term in the document. To calculate the weight of each term in the document,

we use a weighting scheme. The most popular scheme is the TF-IDF. It is based on equation 2.1 and 2.2. TF-IDF is calculated by multiplying the number of occurrences (TF) by the importance scalar (IDF). In case that the IDF is small this means that the term does not show up in many documents which means it is important. In other cases when IDF is big this means that the term is occurring in many documents. Words like the, is, they, he, but, etc. are examples of words that might occur in many documents and these words are not important [87] [2] [1] [41].

$$IDF(t,D) = \log \frac{D}{DF(t)} \tag{2.1}$$

$$TF-IDF(t,d) = TF(t,d) \times IDF(t,D) \tag{2.2}$$

$TF(t,d)$: The number of occurrences of term (t) in document (d).
D : The number of documents in the corpus.
$DF(t)$: The total number of documents with the term (t)

The Bag Of Words (BOW) model is simple and effective in some cases but their main problem is that words are treated as independent units with no relationship to one another. They also produce sparse multidimensional vectors (a vector with many entries with the value 0), which takes big memory space and long processing time [3].

N-gram models

A ML model learns the semantics of the text better if it learns the relationship between sequences of words. For example, the co-occurrence of words like “Gezi” and “park” together gives the higher probability that the tweet label is “protest” than the occurrence of “park” as a standalone word. The co-occurrence of words can be represented using the N-gram model. The N-gram model is a statistical language model that uses Markov Chain Models of order $n-1$ to calculate the probability of a certain sequence of words [28]. N-grams can be unigrams which are sequences containing one word, bigrams which are sequences containing 2 words and trigrams which are sequences of words containing 3 words.

Distributed representation

In this representation, the model learns the distributed representation of each word and the probability function for words sequences from each sentence in the training dataset [13]. Built on that, [53] proposed what is called word2vector model with two architectures for learning distributed representation of words with minimum computation complexity: the Continuous Bag of Words (CBOW) model and the Continuous skip-gram model. The CBOW model predicts the current word based on the context while the Continuous skip-gram model learns the context (surrounding words given the current word). The word2vector models produce word vectors that

represent the semantics of words to the level that if algebraic operations are performed on the word vectors, the resulting vector would have similar semantics compared to the input vectors. For example,

$vector(\text{"}King\text{"}) - vector(\text{"}Man\text{"}) + vector(\text{"}Woman\text{"}) = vector(\text{"}Queen\text{"})$

To produce the vector representation of each document, the words vectors of that document can be combined either by concatenation, average weighting or TF-IDF weighting [68].

2.2.2 Machine Learning Models

After preparing the training dataset and extracting the important features, the model is ready to be trained. There are many machine learning algorithms to choose from for text classification problems. According to the literature, there are two famous machine learning algorithms that proved effective in text classification problems: Multinomial Naive Bayes (NB) and Support Vector Machine (SVM) [68] [41]. This is why these two algorithms are used in this research. In this section, we provide a brief explanation of these models.

Notation

Before we start explaining the models we will use in this research, it is important to explain the Mathematical notations used in this chapter and the following chapters.

$m =$ the number of training samples

$n =$ the number of features

$K =$ the number of classes in the training dataset

$i =$ training sample index where $i \in \{1,2,3,..,m\}$

$j =$ feature index where $j \in \{1,2,3,..,n\}$

$k =$ the class index where $k \in 1,2,3,..,K$

$x =$ a vector of features. For example $x =$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

In text classification, the features are the numerical representation of a word. For example, the count BOW representation of the document "The protest is crushed by police #gezi #gezipark" is: words $x =$

$$\begin{bmatrix} protest \\ teargas \\ gezi \end{bmatrix}$$

$x^T =$the transpose of the vector $x = [x_1, x_2, x_3]$.

$x_i =$the feature vector of all the features in the (i^{th}) training sample.

x_i^j =the value of feature (j) in the $(i^{(th)})$ sample.
X =a matrix of (m) training examples and (n) features.

$$\begin{bmatrix} x_1^0 & x_1^1 & x_1^2 & \cdots & x_1^n \\ x_2^0 & x_2^1 & x_2^2 & \cdots & x_2^n \\ x_3^0 & x_3^1 & x_3^2 & \cdots & x_3^n \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ x_m^0 & x_m^1 & x_m^2 & & x_m^n \end{bmatrix}$$

w =a vector of weights. For example $w =$

$$\begin{bmatrix} b \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

w^T = the transpose of the vector $w = [b, w_1, w_2, , w_n]$
w_i^j = the value of weight (j) corresponding to feature (x_i^j) where $(j) \in \{1,2,3,..,n\}$ and $(i) \in \{1,2,3,...,m\}$
$l =$ labels vector. For example $l =$

$$\begin{bmatrix} l_0 \\ l_1 \\ l_2 \\ \vdots \\ l_m \end{bmatrix}$$

l_i = the label (i) corresponding to the training example x_i where $i \in \{1,2,3,..,m\}$ and $l_i \in \{c_1, c_2, c_3, .., c_k\}$. For the protest classification, the label $(l) \in \{protest, non-protest\}$ and for violence classification $(l) \in \{violent, non-violent\}$

Support Vector Machine (SVM)

Support vector machines (SVM) models have proved successful in many application, especially where the number of attributes is larger than the number of examples (e.g. Text classification). To understand SVM models, we need to have a brief look at linear models first.

Linear Models

Linear models learn a linear relationship between the features and the corresponding labels from

the labelled training dataset. For a training dataset X and l labels where $f: X \rightarrow l$. This linear relationship can be modelled using the following linear equation:

$$\hat{l}_i = wx_i + b \tag{2.3}$$

Where $\hat{l}$ is the predicted class for the i^{th} example, x is the feature vector for the training example (i), w is the weights vector and w_0 is the bias value. For all the training examples (m) and all the features (n) the model can be expressed as:

$$\hat{l}_i = b + w^j x_i \tag{2.4}$$

where $i \in \{1,2,3,..,m\}$ and $j \in \{1,2,3,..,n\}$

$$\hat{l} = w^T x_m + b \tag{2.5}$$

SVM

SVM models aim to separate the different classes using a threshold. If $f(x)$ is above that threshold, then the data point belongs to class A or else the data point belongs to class B. SVM models do this by trying to draw an optimal decision boundary that linearly separates the classes. For a binary classification task, SVM model represent the linear decision boundary as:

$$w^T x_i + b = 0 \tag{2.6}$$

Any data points above that decision boundary is labelled as A

$$w^T x_i + b > 0, \hat{l}_i = 1 \tag{2.7}$$

And if it lies below the decision boundary then it is labelled as B

$$w^T x_i + b < 0, \hat{l}_i = -1 \tag{2.8}$$

SVM models use the following decision function to predict new data:

$$\hat{l}_{new} = Sign(w^T x_{new} + b) \tag{2.9}$$

Now we have a hyper plane with a decision boundary $(w^T x_i + b = 0)$, upper bound $(w^T x_i + b = 1)$ and lower bound $(w^T x_i + b = -1)$. If we pick two points on the upper and the lower bounds of the hyper-plane, the perpendicular distance between these two points **(called support vectors)**

and the decision boundary is called the margin γ [62].

$$\gamma = \frac{2}{||w||^2} \tag{2.10}$$

The learning process in SVM models aim at finding the largest possible margin by maximizing the margin value or (computationally easier) minimizing the margin's inverse. The learning objective function now looks like

$$argmin_w = \frac{1}{2}||w^2|| \tag{2.11}$$

This objective function is trying to find the weights vector (w) that maximizes the margin. This optimization problem is constrained by the upper and lower bounds of the hyperplane as the weights found must make sure that all data points above the upper bound must satisfy $(w^T x_i + b >= 1)$ and all the points below the lower bound must satisfy $(w^T x_i + b <= -1)$. This adds new parameter's vector to the objective function (α_m). It adds a parameter α for each data point. This parameter's value is proportional to the influence of each training point in the decision function. If $\alpha > 0$, the associated points are used in the decision function which is the case for the support vectors [62].

There are two types of margins: hard margins, where all training examples must sit on the right side of the decision boundary. A hard margin is an example of a model overfitting. The second type of margins is a soft margin where some training examples are allowed to sit close to the decision boundary or on the wrong side of it. This can be controlled by a control parameter (C) that controls to what extent the training samples are allowed to sit within the margin or on the wrong side of the decision boundary. $0 \leq \alpha_m \leq C$, C imposes an upper bound to the influence of the data points in the decision function meaning that if C is small, then, α_m becomes small and more training samples become active in the decision function. In other words, we get more support vectors, some of them are on the wrong side, which makes the margin soft. Likewise higher C value makes the margin harder [62].

SVM models can also fit non-linear data. This can be achieved by using a kernel function (K). A kernel function transforms the features into another a higher dimensional space in which the features can be linearly separable. There are different kernel functions. The most famous are linear, Radial Basis Function (RBF), *polynomial* and *sigmoid*. Non-linear functions use an extra parameter gamma γ. γ affects the decision boundary that separates the transformed features. High γ value increases the complexity of the decision boundary and a low γ value makes the decision boundary flexible. It is important to mention that C and γ affect the model in a coupled manner so later when we choose the values for both C and γ we have to find the best combination of the two that gives the best results with our data [62].

Multinomial Naive Bayes

Naive Bayes (NB) classifiers are simple and efficient models, especially for small training datasets. NB classifiers are easy to implement, fast and accurate [52] [59] [68]. NB classifiers assume that the features in the training dataset are conditionally independent given the class that is why they are called naive and they are based on the Bayes theorem as in Equation 2.12.

$$P(l_k|x_i) = \frac{P(l_k) \times P(x_i|l_k)}{P(x_i)} \tag{2.12}$$

$P(l_k|x_i)$ called "Posterior probability" is the probability that a training sample (i) belongs to class (k) given the feature vector x_i where $k \in \{1,2,3,..,k\}$ and $i \in \{1,2,3,...,m\}$.

For example, given a tweet (i) = "gezi park on fire, police fires teargas in gezi park". The features are $[gezi, park, fire, police, teargas]$.

NB models estimate the probability that the tweet (i) belongs to the "protest" class $P(protest|x_i)$ and the probability that tweet (i) belongs to the "non-protest" class $P(non-protest|x_i)$. The decision then is made based on the higher probability. If $P(protest|x_i) > P(non-protest|x_i)$, The decision is that the tweet belongs to the protest class. The training process aims to maximizing the probability that a new unseen sample (i) with features vector (x) belongs to class (k) [59].

$$\hat{l_i} \leftarrow argmax_{x_{i=1,2,3,..,m}} P(l_k|x_i) \tag{2.13}$$

To estimate the probability, the model calculates:

- $P(x_i)$, "Normalization scalar", is the probability of observing that the feature vector (x_i) is independent from the class label.
- $P(l_k)$, "Prior probability", is the probability that class (k) is found in the dataset. Two popular ways to set the value of the prior:
  - $P(l_k) = \frac{1}{k}$, Where (K) is the total number of classes. For a binary classification $K = 2$ and $P(l_{(k=protest)}) = 0.5$ and $P(l_{(k=non-protest)}) = 0.5$.
  - $P(l_k) = \frac{M_k}{M}$ Where M is the total number of samples in the training dataset and M_k is the number of training samples that belong to class k. For example, the number of tweets belong to the "protest" class is 10 and the total number of tweets in the training dataset is 100, $P(l_{protest}) = \frac{M_k}{M} = \frac{10}{100}$ and $P(l_{non-protest}) = \frac{M_k}{M} = \frac{90}{100}$
- $P(x_i l_k)$, "Likelihood" or "class conditional probability", is the probability of observing the feature vector (x) given that it belongs to class (l_k). This can be estimated using Maximum likelihood as in equation 2.14.

$$P(x|l_k) = P(x^1|l_k) \cdot P(x^2|l_k) \cdots P(x^n|l_k) = \Pi P(x_i^j|l_k) \tag{2.14}$$

The individual likelihood of each feature in the feature vector is estimated by equation 2.15.

$$\hat{p}(x^i|l_k) = \frac{N(x^j, l_k)}{\sum_{i=1}^{m} N(x_i, l_k)} \tag{2.15}$$

Where $N(x^j, l_k)$ is the sum of number of times the feature value (x^j) appears in training samples with label (l_k) and $\sum_{i=1}^{m} N(x_i, l_k)$ is the sum of the counts of all features in samples with label (l_k).

For example, a new tweet (i) = "gezi park" needs to be classified with the NB model that has been trained on a dataset of 500 tweets where 100 tweets are protest tweets. The number of times the words "gezi" and "park" appeared are 70 and 10 times respectively within the sample of 100 protest tweets and the sum of the count of all the words in the tweets labelled as protest is 1000.

The features are [gezi, park]
$P(x_i = [gezi, park]|l_k = protest) = P(gezi|protest) \cdot P(parkprotest) = \frac{70}{1000} \cdot \frac{10}{1000} = 0.0007$

The previous expression can be calculated using maximum likelihood. There are two problems that could happen. First, if the new tweet has words that the model did not see before in the training dataset. Then $N(x^j, l_k) = 0$ and $\hat{p}(x^j|l_k) = 0$ and the outcome of equation 2.12 is 0.

The second problem is if we have too many features and too many samples, for example, the number of features (n) is 7000 and the number of training samples(m) is 5000, this will cause what is called the "curse of dimensionality". Fitting too many dimensions in any probability distribution function will lead to too many parameters to fit and to find the likelihood. To solve these two problems, we first use a smoothing parameter α to increase the feature count by a certain amount and to make sure that the whole expression does not give 0 in case the new samples have new words. There are two ways to do smoothing: Laplace smoothing in which $\alpha = 1$ and Lidstone smoothing where $\alpha < 1$. For the second problem, the "Naive " assumption becomes useful. Because it assumes that each feature is independent of other features. This assumption can be used to make the estimation of the likelihood a bit simpler by calculating the probability distribution function for each feature vector independently and this makes the number of parameters to fit reasonable [59].

There are different probability distribution functions to estimate the likelihood. Multinomial Naive Bayes uses the multinomial distribution function to estimate the probabilities of each feature in the feature vector of the new unseen data that gives the predicted class l_k. The maximum likelihood estimate of the likelihood term in the multinomial naive Bayes model after adding the

smoothing parameter α is

$$\hat{p}(x^i|l_k) = \frac{N(x^j, l_k) + \alpha}{\sum_{i=1}^{m} N(x_i, l_k) + \alpha \cdot V} \tag{2.16}$$

Where $N(x^j, l_k)$ is the sum of number of times the feature value (x^j) appears in samples with label (l_k) and $\sum_{i=1, i \in k}^{m} N(x_i, l_k)$ is the sum of the counts of all features in samples with label (l_k) and (V): the number of feature, which is the size of the vocabulary in our dataset [59].

2.2.3 Evaluation

After training the classification model, we test the performance on the test set as explained in section 2.2.1. There are different measures to evaluate the performance of the different models. In this section, we discuss some of these measures and how to use statistics to select the best performing [62].

Performance Measures

There are four outcomes that summarizes the classification results:

- **True Positive (TP):** the number of test data points that are correctly labelled as positive. The true label $l = 1$ and the predicted label $\hat{l} = 1$. For example, protest tweets are labelled as protest-related and violent tweets labelled as violent.

- **True Negative (TN):** the number of test data points that are correctly labelled as negative. The true label $l = 0$ and the predicted label $\hat{l} = 0$. For example, non-protest tweets are labelled as non-protest and non-violent tweets labelled as non-violent.

- **False Positive (FP):** the number of negative test data points that are mis-classified as positive. The true label $l = 0$ and the predicted label $\hat{l} = 1$. For example, non-protest tweets are labelled as protest and non-violent tweets labelled as violent.

- **False Negative (FN):** the number of positive test data points that are misclassified as negative. The true label is $l = 1$ and the predicted label $\hat{l} = 0$. For example, protest tweets are labelled as non-protest and violent tweets labelled as non-violent.

For a binary classification problem, these values can be visualized in a confusion matrix as in table 2.2.

Sensitivity and specificity

Sensitivity, also called recall or true positive rate (TPR), measures the proportion of all the positive data points in the test set $(TP + FN)$ that are correctly classified as positive (TP). It is

	True Labels		
Predicted Labels		1	0
	1	TP	FP
	0	FN	TN

Table 2.2: Example of a confusion matrix

calculated as:

$$S_e = \frac{TP}{TP+FN} \tag{2.17}$$

The complementary value of sensitivity is called the False Negative Rate (FNR) $FNR = 1 - TPR(S_e)$. It is the proportion of all the positive data points in the test set $(TP+FN)$ that are misclassified as negative (FN).

Specificity, also known as the True Negative Rate (TNR), measures the proportion of all the negative data points in the test set $(TN+FP)$ that are correctly classified as negative (TN). It is calculated as :

$$S_p = \frac{TN}{TN+FP} \tag{2.18}$$

The complementary value of specificity is called the False Positive Rate $(FPR) = 1 - TNR(S_p)$. It is the proportion of all the negative data points in the test set $(TN+FP)$ that are misclassified as positive (FP). S_e and S_p values range from 0 to 1. If the model classifies all the test data points as positive (all tweets are violent tweets), Sensitivity $S_e = 1$ but that would mean that $S_p = 0$ as all the negative data points misclassified as positive (all the non-violent tweets are misclassified as violent tweets). This is a bad model, instead, we want a model that achieves an acceptable ratio between sensitivity and specificity. This acceptable ratio depends on the application, a better measure would be to combine sensitivity and specificity into a single measure. This leads to the next measure [62].

The Area Under The ROC Curve (AUC)

The models we reviewed in this chapter MNB and SVM use thresholds to classify the data. For example, the NB model classifies a data point as positive if the probability of the positive class is higher than 0.5. Another example, the SVM model classifies a data point as positive if the linear model returns 1. The receiver operating characteristic (ROC) curve allows us to measure how the model's performance is changing as the threshold changes [62].

The sensitivity and specificity are calculated for different threshold values. The sensitivity (TPR) is then plotted against the complementary of specificity (FPR). This gives the curve in Figure 2.2. Low sensitivity $(TPR = 0)$ and high specificity $(FPR = 0)$ means that the model does not classify anything as positive $(TP = 0)$ and $(FP = 0)$ but it classifies all the points as negative $(TN > 0)$ and $(FN > 0)$. On the other hand, high sensitivity $(FPR = 1)$ and low

specificity $(FPR = 1)$ means the model does not classify anything as negative $(TP > 0)$ and $(FP > 0)$ while $(TN = 0)$ and $(FN = 0)$. If the model's performance generated a plot that is close to the straight line $(0,0)$ to $(1,1)$, this means that the model is randomly guessing that accurately labelling the data. We want a threshold that gives us a high sensitivity $(TPR = 1)$ and high specificity $(lowFPR = 0)$.

We aim for the top left corner in Figure 2.2. This means that there is a threshold that makes the classifier performs perfectly. We use the area under the curve (AUC) score to quantify the model's performance. The better the model the closer it is AUC score would be to 1 (top left corner of the graph). On the contrary, AUC score of 0 means $(TPR = 0)$ and $(FPR = 1)$. If the AUC = 0.5, this means the model is performance of close to the straight line $(0,0)$ to $(1,1)$. Sometimes the dataset we are working with is imbalanced, meaning that the number of samples that belong to one class is higher than the number of samples that belong to the other class. The AUC score is a ratio between the TPR and FPR that is why it is a good measure to use in such cases. The AUC score is the measure that we use to measure the performance of the MNB and SVM in this research.

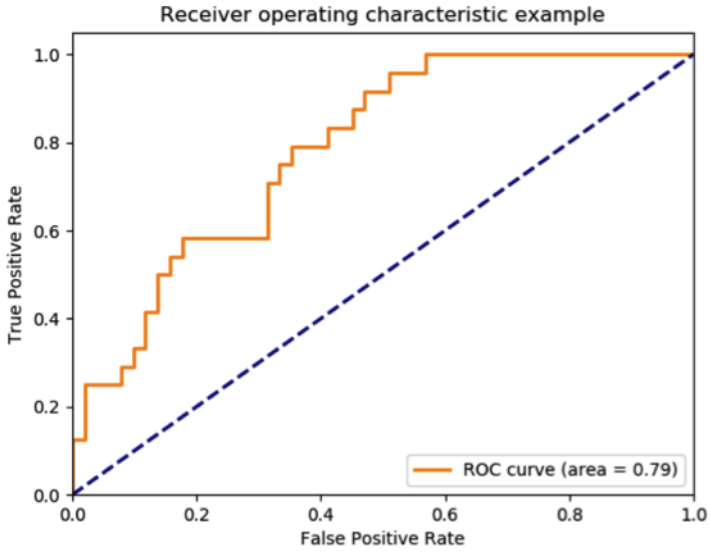

Figure 2.2: Example of ROC Curve [70]

Model Selection

We investigate the use of two baseline classification models (MNB and SVM) to measure protest repression related tweets. We also are going to try out different ways of feature representation (BOW, TF-IDF and WE). After training the two models on the different representations, testing them and evaluating their performance using the AUC score, we have to choose the best performing model with the best text representation and then to continue with the data analysis

chapter. That is why we will need a proper statistical test to compare the models.

In this section, we describe statistical hypothesis testing and the approach that will be used in this research. Hypothesis testing is a method of making a decision from sample data. We compare our observations from the sample to what we expected (Null hypothesis). Null hypothesis (H_0) is a statement about the population data. We use the sample data to decide to accept it or reject it. In our case, the null hypothesis is that there is no difference between the performance of the MNB and SVM in terms of AUC scores on our dataset.

We use a statistical test to compute a test statistic value from the sample to decide whether to accept or reject the null hypothesis. The test statistic value is then used to obtain the p-value, which is the probability of getting the same test static if the null hypothesis is true. There is a probability to reject the null hypothesis even when it is true. This probability is called the significance level. This value depends on the statistical test used. Most of the time significance level is set to 0.05, which means that there is a 1 in 20 chance to reject the null hypothesis when it is true. To test our hypothesis we compare the p-value obtained from our test statistic to the significance level. If p-value < significance level we reject the null hypothesis which means that one of the baseline models performs better than the other. There is no standard way to perform the significance test to compare classifiers [79].

The most commonly used method to compare between two classifiers are: the k-fold cross-validated t-test and the McNemar test. McNemar test compares the errors made by each model, which means the test uses TP, FP, TN and FN values of one test set. McNemar test is a good choice when the size of the $FN + FP$ is greater than 50 and the variance between the different folds on which the model is trained is small [60]. In other words, it is better to use McNemar test when a model is trained and tested once without cross-validation [23]. Others argue that the non-parametric Wilcoxon signed ranks test is a good choice. Wilcoxon test compares the AUC scores of each model on different test sets. The Wilcoxon test is a good choice when the sample size (AUC scores for each model) is close to 15 and without the assumption that the data is normally distributed [22]. In this research, we use one of the two tests when it is appropriate. We use Wilcoxon test to compare the different variations of one model e.g. to test the different combinations of parameters for SVM or MNB. To compare between the performance of SVM and MNB on different datasets, McNemar test is used.

Conclusion

In the first section of this chapter, we discussed how important protests are in influencing the political lives and democracy and how state repression towards protests is not only violation to human rights but also a threat to democracy. We introduced the definition of protest, protest

repression, the dimensions of protest repression typology in political science literature and the types that we are interested in in this research: state actor repression towards protesters using observable coercive actions. The literature also shows the drawbacks of using newspapers and surveys as a source to collect protest repression events. It shows the advantages of using social media as a source of near real-time data with accurate geographical coverage, wide participation from different backgrounds, and accepted accuracy. In the second section, we introduced the basic concepts of text classification and machine learning process starting from data preprocessing, feature extraction, model training, model evaluation and model comparison. We introduced the techniques that are used for each step in the experiments in chapter 4.

In the next chapter, we use the defined concepts (protests and protest repression) and state observable coercive state repression typology as guidelines in the data collection and data labelling process.

Chapter 3

Data Collection

In the previous chapter, we discussed the limitations of using news reports as a source of information to build political conflict datasets like coverage bias, censorship and duplication. We also discussed the potential in using social media and Twitter as an alternative source of information and the advantages of using machine learning models to extract political events from the text over human coders to save money and time. We also explained that machine-learning models need a labelled training dataset to learn the patters corresponding to certain labels and to be able to generalize to new unseen data. To use machine-learning models to detect protest and violence-related tweet, we need two training datasets. The first dataset is a collection of tweets labelled as protest or non-protest to train the protest classification model. The second dataset contains the same tweets but labelled as violent or non-violence to build the violence classification model.

In this chapter, we are describing the process of building these training datasets. As mentioned before it is expensive and time-consuming to hire an expert to label the tweets. That is why we use crowdsourcing to label the tweets. It is a challenging task to make sure that we hire the right crowd workers. We provide clear instructions and offer good incentives for the workers to do the job. There are quality issues related to the workers as some of them tend to be spammers who answer the questions randomly without reading the instructions or even the question. We first review the literature body on using crowdsourcing with machine learning to find out the main parts of the crowdsourcing task that influence the quality of the results. We also use the pilot study to explore and analyse the quality control settings and task design and how to enhance them to finish the task as fast as possible and with an acceptable level of quality. Then, we hire crowd workers through an online platform called Figure-Eight and asked them two questions for each tweet: is the tweet related to Gezi protest 2013 or not? And does the tweet report a violent incident? We use quality settings inspired by the literature and the pilot experiment. Finally, we analyse the agreement internally between the workers and the agreement between the workers and me. We investigate the disagreement between the workers and how to mitigate their influence, the mistakes that workers did and why did that happen, how good the labels are, and if

they are good enough to use for training the machine-learning model.

3.1 Data

The dataset was collected during the Gezi Park protest in Turkey in summer 2013. The tweets collection covers a month from 31/05/2013 to 30/06/2013. Researchers at the New York University Social Media and Political Participation laboratory[1] collected the dataset. First, they developed a list of keywords[2] and most used hash-tags by observing sample tweets, for example, occupygezi and #direnankara. They used the list of keywords to query Twitter API to collect the tweets in real-time, which provides 1% of all tweets during a given time window. This means that sometimes if the limit is reached for some keywords or hash-tags, the API will not return the full set of tweets that include this keyword or hash-tag. Another source of bias is that not all the protest-related tweets use hash-tags and on the other hand, some promotional tweets use the most trending hash-tags, which at the time of data collection contained hash-tags related to the protest. The collected dataset contains 30 million tweets. 74% of the tweets are in Turkish and the majority of the rest are in English. They made the dataset available online by making a list of Tweets IDs[3] .

This list of IDs was then used to collect the actual tweets using the Python package tweepy[4]. The returned tweets were filtered by language to retrieve only English tweets. That decision was made because if the model is trained on English dataset, the model can be used to predict other tweets from other countries if the tweets are in English, which is the most used language on Twitter according to [10] up to 90% of Twitter posts are in English. During the protest, some protesters post in English to attract the international community to their cause like the green protests in Iran, Arab spring and Gezi park Turkey [33] [15] [73]. This decision has its consequences, as the model may not be able to detect violent tweets in other languages as accurately. This is similar to the coverage bias as in news sources. However, the fact that people tweet in English to attract international attention to their cause might reduce the effect of this bias in Twitter. Due to Twitter privacy policy, if the user deleted the post or deleted their account, the tweet will not be available anymore. That is why the total number of the tweets eventually collected reduced from 30,000,000 to 1,290,451. This is a dataset collection that will be labelled later using the machine-learning model and then will be used for analysis.

[1] https://wp.nyu.edu/smapp/

[2] https://pdfs.semanticscholar.org/58fc/28bcd69e078710203f56c5107e31754b328b.pdf

[3] https://dataverse.harvard.edu/file.xhtml?persistentId=doi:10.7910/DVN/WCXK3Z/DGQVK8&version=1.1

[4] http://docs.tweepy.org/en/v3.6.0/

3.2 Crowdsourcing

The word Crowdsourcing is a portmanteau of the words "crowd" and "outsourcing". It means outsourcing a problem to a crowd rather than an expert [35]. This is because in some cases, there are difficulties in finding the experts to solve the problem due to limitations in finding the expert, money or time. In crowdsourcing, the problem is broadcasted to a crowd so that potential solvers can step in and propose solutions. With the wide use of the Internet and Information technologies, outsourcing problem to crowds became easier which means more use of crowdsourcing [81]. Crowdsourcing is built on: an organization (Customer) releases a task online through a crowdsourcing platform to a crowd of outsiders. Then, a group of interested people (Contributors) perform and submit the task to the same crowdsourcing platform for a fee or any other incentives. Crowdsourcing is not only limited to business organizations. Non-profit organizations and academic organizations also use crowdsourcing [94].

The traditional way of collecting information from a crowd is traditional surveys. In traditional surveys, a representative sample is selected and contacted via telephone, mail or face-to-face interviews. Crowdsourcing, on the other hand, could be considered as an internet-based survey through crowdsourcing platforms where the crowds are easily contacted for a smaller fee [20]. With entering the era of big data, there is a lot of digital material that could be used to train machine-learning models. Machine Learning (ML), text mining and Natural Language Processing (NLP) used crowdsourcing to obtain labels for the unlabeled data because it is an easy, cheap and fast way of collecting data [74]. It gives access to larger and more diverse annotators which according to [64] leads to less biased data and more labelled data for the same cost to employ smaller number of experts to do the same task with similar results [16] [51] [36] [29] [84]. The availability of more ongoing-labelled dataset can benefit supervised and semi-supervised machine learning models [45]. There are several crowdsourcing platforms to use. Depending on the nature of the task you can choose the one that suits you best. The platforms that are usually used for research purposes are Amazon Mechanical Turk[5] , Figure-Eight [6] (used to be known as Crowd-Flower), Click-Worker [7] and Prolific Academia [8] . For business purposes, there are Up-work[9] and Top-Coder[10].

One of the main challenges in crowdsourcing is quality. Especially for Machine Learning (ML) tasks like text classification where the quality of labelled training dataset affects the performance of the model. Quality control, we mean making sure that contributors actually read the instruc-

[5] https://www.mturk.com/
[6] https://www.figure-eight.com/
[7] https://www.clickworker.com/
[8] https://prolific.ac/
[9] https://www.upwork.com/
[10] https://www.topcoder.com/

tions and the questions and give the right label/answer to the given text/question. This is to make sure that the contributors are not spammers and they understand the task well. The quality, in this case, depends on the agreement between the contributors. In cases of disagreement, the quality of the answer depends on the reliability of the contributor who gave that answer. From the perspective of Natural Language Processing (NLP) and text classification, there are three factors that affect the quality of crowdsourcing labelling task: the data, the contributor and the task design [6] [4] [45] [64].

3.2.1 Data

The clarity of the data used in the crowdsourced task affects the contributor's understanding, which in turn affects the quality of their answers to the questions in the task and eventually the labelling of the tweets. NLP tasks process human-generated textual data. the data might contain misspellings and grammatical mistakes and this could be a source of confusion to the contributor. This can be seen on a large scale in tweets.

3.2.2 Contributor

Some contributors could be lazy to read the instructions or to read the actual question. They might be spammers who answer randomly or copy and paste answers online. In a study, 100-crowdsourcing workers were interviewed about the tasks they have done. 27% of them claimed that they continued to complete more than 50% of all the tasks that were unclear to them [30]. There are two strategies in the literature to ensure the contributor is integrity, consistency and comprehension of the task [54] [64].

- **Filtration:** the first strategy is to filter the contributors before starting the task to make sure that they are fit for the task by:
  - **Location or Language:** this filter makes the task only accessible to contributors from specific location or speak specific language depending on the requirements of the task [69] [63] [90].
  - **Prior Performance:** this is a filter that uses is a contributor is score that tell how good a contributor did in previous takes [90] [37].
  - **Screening Test:** is a competence test to make sure that the contributors have the skills to complete the task [51] [37].
    Although filtration is used to make sure that the task is only open the right contributors, it's main drawback is that it slows down the crowdsourcing task [64]. There is also no guarantee that once the contributor passed the filtration step, they will perform the task accurately [54].

- **Evaluation:** the second strategy is to evaluate the contributor is performance and train them while the task is running. There are two approaches to do that:
  - **Estimating Contributor Trust:** this approach is used to infer the contributor's performance. It is based on the Expectation-Maximization algorithm that uses maximum likelihood [40]. this approach is implemented in Amazon Mechanical Turk (AMT). The drawback of this approach is that it is expensive because it needs to collect an excessive number of judgments [54].
  - **Gold Standard:** This is the performance detect used by Figure-Eight. Gold standards or gold units are a subset of the actual data items uploaded to the crowdsourcing task but along with their right labels and the reason for the labels. The contributor is performance can be estimated by randomly adding gold units to the actual data units. If the contributor gives the wrong answer, they are corrected and shown the reason for the right answer as a way to train them. After a few trials, the contributor 's performance starts to get recorded. If the contributor is accuracy goes below the specified threshold for the task, they are rejected from the task and their labels became not trusted [54] [27] [69] [37]. The drawback of this approach is the time needed to prepare the gold units. Some argue in the literature that the best practice is to have 20% of the total number of data units as gold units. This would be challenging for tasks with big data collections [64] [54].

3.2.3 Task Design

Task design is an important step and has a direct influence on the quality of the results. The clarity of the instructions in terms of length and language and providing examples affects the contributor is understanding of the task which affects the quality of their performance [30]. For ML related tasks, the agreement between the contributors was higher for the task with simple binary labels [6]. It is advised to not to have more than seven labels to choose from [64].

3.3 Figure-Eight Experiment

In this section, we describe the crowdsourcing experiment that we ran on the Figure-Eight platform. Figure-Eight, previously known as Crowd-Flower, is a crowdsourcing platform specialized in providing solutions related to artificial intelligence and data science applications like search relevance evaluation, sentimental analysis and data classification [4]. Figure-Eight is based on the idea of creating millions of simple online tasks to create a distributed computing machine. Figure-Eight provides tools for the customer to upload their data and design their task. Then, the task becomes available to the contributors (the crowd). Figure-Eight provides tools

to set the desired quality control settings, the speed and the cost of the task. We used Figure-Eight because it implements most of the quality control mechanisms discussed in the previous section [69] [83] [27] [37]. For this research, we run two experiments on Figure-Eight. The first is a pilot experiment to find out how long the task would take and how much it would cost and what are the best quality settings for the task. Developing a pilot crowdsourcing experiment is a good practice to learn how easy the task is to the crowd and the best quality settings to get the task done with good quality and in a reasonable time [27] [63] [51] [69] [37]. The second experiment is the main user study that is used to label the data collection that will be used to train the machine-learning model in the next chapter. For both experiments, we created a task to label a subset of the tweets collection described in section 3.1. we discuss the main crowdsourcing tasks in terms of the quality control factors discussed in section 3.2: data preparation, the quality settings to monitor the contributors'performance and task design. In the following section, for each quality factor, we first explain how this factor is implemented in Figure-Eight. Then, we describe how we used this in our experiment, what are the values we used for the different settings and how the pilot experiment inspired these values.

3.3.1 Data

Figure-Eight Options

Figure-Eight allows the users to upload their data in a UTF-8 encoded CSV file TSV, XLSX, and ODS file formats are also acceptable and also require UTF-8 encoding. All data files must have a column header for each column.

Experiment's settings

A subset of 10,000 tweets was randomly selected from the tweets collection described in section 3.1 to be used in the main user study. We randomly selected another 1214 tweets from the 10000 tweets to be used in the pilot study. To test the agreement between the crowds labelling and an expert (me in this case), we hand labelled the pilot study dataset. The rest of the tweets were uploaded for the actual crowdsourcing task but for financial reasons we could only afford 6693 tweets. This is the dataset used later in training the model.

3.3.2 Contributors

As mentioned before, there are two ways to guarantee the contributors's integrity and quality: filtering out unsuitable contributors before the task and evaluating the contributor 's performance during the task. This section discusses which of these strategies is implemented in the Figure-Eight platform and what are the values that are used in this experiment.

Figure-Eight Options

Filtration in Figure-Eight Figure Eight implements two of the filtration methods: location and language by allowing the customer to set a geographical option to limit the participation to the task. Prior performance is also implemented in Figure-Eight to limit the task to contributors with a certain level of experience in two ways:

- **Internal or External Contributors:** this option allows the customer to either set the task to only in house experts by choosing internal contributors or getting more diverse contributions by allowing external non-expert crowd contributors to do the task.
- **Contributor's experience Level:** even when the crowdsourcing task is open to the external crowd contributors, the customer can set the level of experience of the contributor based on their performance in old crowdsourcing tasks. There are three levels of experience from level 1 to level 3 with level three is the most experienced. The higher the level of experience the smaller the numbers of the contributors who will have access to the task. This makes the task slower to finish.

Evaluation in Figure-Eight To evaluate the performance of the contributors throughout the task, gold standards, discussed in section 3.3.2, are implemented in Figure-Eight and are called Test questions.

- **Test Questions:** these test questions are used to both train and detect the contributor's performance. By default, there is one test question per page of data items (tweets), which means each contributor can only do as many pages of data as test questions there are in the task. This to make sure that the contributor does not see the same test question more than one time, which prevents cheating. However, if the contributor does the same test question several times separated by other test questions, this might be a good test to their consistency.

Experiment's settings

Filtration

To find the best technique to filter out the unfit contributors, we limited the task to external workers from Turkey and with experience level 2. We made these choices based on some studies found that a higher number of non-expert contributors can achieve the same levels of accuracy as small number of experts with more diversity [74] [45] [64]. For 1214 tweets, it took 145 hours (6 days) to finish the task. To test the effectiveness of the filter we sat on the contributors, we evaluate the quality of the labels they produced. we did that by comparing their aggregated label to our label. The standard deviation of the contributors's aggregated protest labels is 0.5 and The standard deviation of the contributors's aggregated violence labels is 0.33. The standard deviation of our protest labels is 0.49 and our violence labels is 0.39. We then

detect the inter-annotator agreement score between the contributors'aggregated labels and our labels. By agreement here, we mean the agreement between the contributors on the label/answer to the questions in the task. To calculate the inter-annotator agreement, we use the Krippendorff-alpha [42] for nominal data. The Kripperndorff-alpha scores for the protest label is 0.51 and the score for the violence label is 0.52. According to [78] the agreement score for nominal data using Krippendorff-alpha is fair if it is above 0.6. This means that the agreement between the contributors'labels and our labels is not high. Assuming that our labels are correct that means that the filtration technique used did not lead to high-quality results. This means that restricting the participation to only contributors from Turkey with high experience did not necessarily guarantee high-quality data and it took a long time to complete. That is why in the main user study; we chose to remove the geographical and the experience limitations allowing contributors from all over the world with an experience level of 1 to do the task.

Evaluation

The recommended number of test questions on Figure-Eight is 50. Others say that it is best practice to have 20% of the data as test questions, which is impractical sometimes when the number of data points is high. We randomly selected 116 tweets from the task tweet of 10000 and hand labelled them and uploaded them as a test question. Using 116 tweets allows the task to be done by as many contributors as possible, which provides diversity.

3.3.3 Task Design

Task design in crowdsourcing is an important step as creating a clear task with clear instructions contributes to the quality of the results [30] [6]. Labelling tasks with small number of categories are less confusing for the contributors and takes less time to answer [64] [5] [4] [83] [74] [44]. Some argue that attention should be paid to which controls to use in the design as it could be used to prevent cheating. In Figure-Eight, Task design is not only designing the user interface for the crowdsourcing task but also setting the threshold values to keep track of the contributor's performance, the quality of generated labels, time to be spent on the task and other settings that contribute to the quality of the generated labels. The data labelled in this crowdsourcing experiment (the main user study) will be used to train a text classification model to detect protest-related tweets and violence-reporting tweets that were posted during the Turkish Gezi protest 2013. To detect the protest tweets, the model needs to be trained on tweets that are not spam and is directly related to the protest event happening on the ground. To detect violent incidents, the model needs to be trained to capture the textual pattern (features) of tweets that report violence that use keywords related to violence like "teargas". This means that at the classification level we do not need to discriminate between tweets that report violent protest or protest repression incidents. That will come later when we do the information extraction. This will be reflected in the task design. The task design is built to collect two pieces of information for each tweet by

asking two questions. The first is about if the tweet is related to the Gezi protest in Turkey 2013 or not? The second is to ask about if the tweet reports a violent incident or not? Regardless if it is done by the police or the protesters.

Quality Measures

Figure-Eight uses the following two methods to keep track of the contributor is performance and the quality of the label/answer for the given crowdsourcing task.

Figure-Eight Options

Contributor is Trust

The contributor's performance is detected by calculated their trust score based on their performance in answering the test questions. After the contributor sees 4 test questions, their trust is calculated as follows: contributor is $trust = \frac{1-N_{missed}}{N_{shown}}$ Where N_{missed} is the number of test questions the user did not answer correctly and N_{shown} is the number of test questions the contributor has seen. The contributor is trust is a value between 0 and 1. The closer the value to 1 the more trusted the contributor is. If the value is low, then the contributor is not trusted and their labels should be rejected.

Confidence Score

The confidence score of each label/answer is the level of agreement between the contributors for each row of data weighted by the contributor is trust. To calculate the confidence score, the following steps are taken: Calculate the sum of the trust scores of the contributors responsible for each response. For example, given three judgments for each tweet. We calculate the sum of the contributors'trust scores for all violent tweets and non-violent tweets sum of contributors trust scores (violence) and the sum of contributors trust scores (non-violence)

1. Violence Trust Score (VTS)

$$VTS = \sum_{C=1}^{N} trustScore(C) \tag{3.1}$$

   Where C is the contributor who gave violent label and N is the number of contributors who gave the violent label.

2. Non Violence Trust Score (NVTS)

$$NVTS = \sum_{C=1}^{N} trustScore(C) \tag{3.2}$$

   Where C is the contributor who gave non-violent label and N is the number of contributors who gave the non-violent label.

3. Violence Confidence Score (VCS)

$$VCS = \frac{VTS}{VTS + NVTS} \tag{3.3}$$

4. non-Violence Confidence Score (NVCS)

$$NVCS = \frac{NVTS}{VTS + NVTS} \tag{3.4}$$

The confidence score value also is between 0 and 1. If all the trusted contributors agree on one label, then this label is score will be 1 and the opposite label is score will be 0. The confidence score (violence) is compared to the confidence score (non-violence) and the label with the highest score is the one that is more probable to be right.

Results

After the task is done, Figure-Eight provides the results in two formats: the full report and the aggregated report.

- **Full Report:** a full report contains row id, every single judgment for each row of data, each contributor is id and their trust score.
- **Aggregate Report:** this report aggregates all the judgments of each of the data row into one row. So instead of having 3 rows for each tweet, we have only one label for each tweet. There are different options available on Figure-Eight to aggregate the judgments. The most common and the default is the "Best Answer" which returns the judgment with the highest confidence score.

Experiment's settings

Task Interface

Based on the literature discussed before, we first designed a pilot crowdsourcing study with two questions:

- Is this tweet related to Turkish protests 2013? The user is given two options: Yes or No.
- Does this tweet report/discuss violent incidents happened during the protest? With three options Yes, No or not sure.

The second question depends on the first. If the answer for the first is yes, then the second question is activated. Otherwise, the second question remains dimmed. Although some scholars argue that using radio buttons opens the doors for the spamming [44], we collected the contributors' answers using radio buttons. This decision was made because this would be the fastest

and easiest way for the contributors to give their answers. In the pilot crowdsourcing study, the number of contributors who gave a label/answer for each question range from 3 to 6. The alpha score for inter-annotator-agreement between the contributors for the protest label is 0.64 and for the violence label is 0.61, which is a fair agreement. This means that the majority of the contributors understood the instructions similarly and gave close labels to the same tweets. But when the aggregated answer was chosen and the inter-annotator-agreement was calculated between that answer and our label the score for the protest label was 0.51 and for the violence label 0.57. This might have happened for 2 reasons:

1. The task instructions were not clear.

2. The task design was not the best to serve the study.

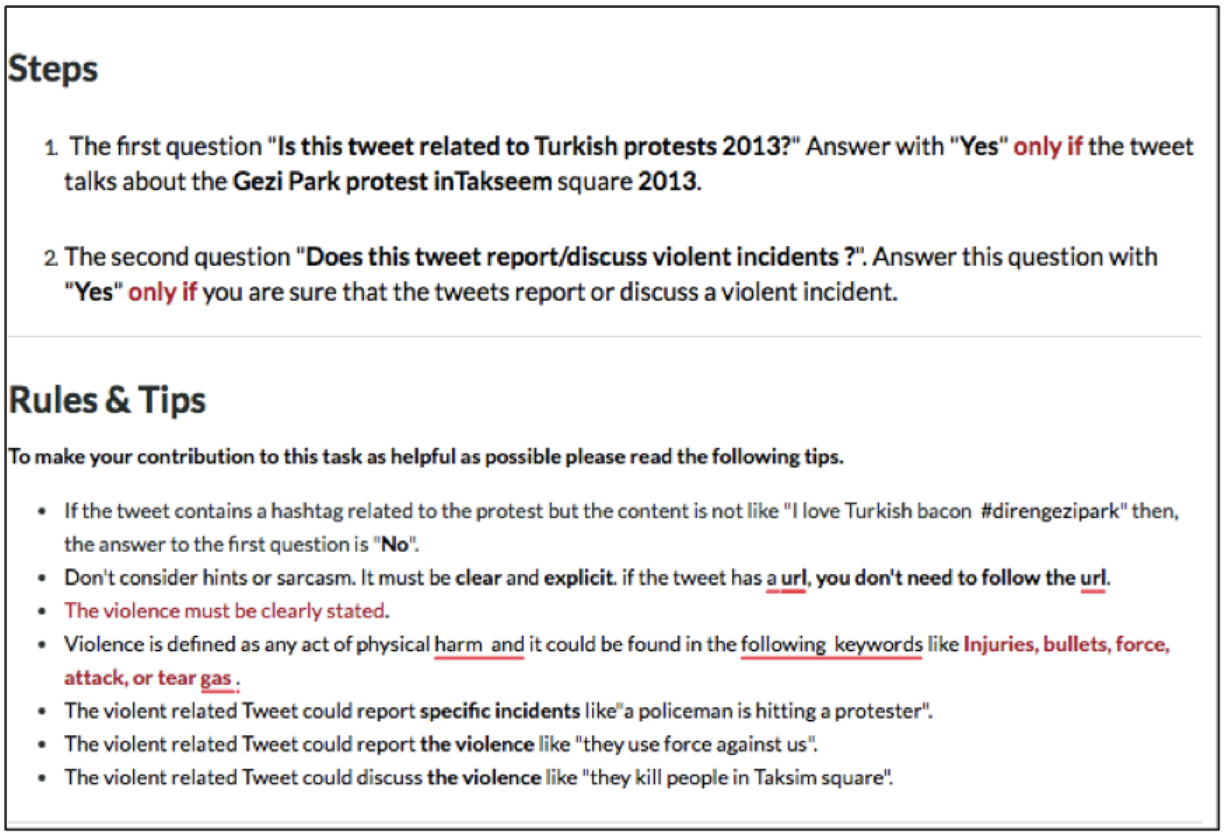

Figure 3.1: Instructions and tips we provided with the crowdsourcing task on Figure-Eight

Learning from the pilot study, clear instructions were provided to the users with highlights on the important information. Different examples that cover different cases with an explanation to the answers were provided too as shown in Figures 3.1 and 3.2. Also, we changed the task design, as for each tweet the contributor is asked two questions independently as shown in Figure 3.3:

- Is this tweet related to Gezi Park protest in Turkey 2013? The user is given two options to choose from “Yes” or “No”

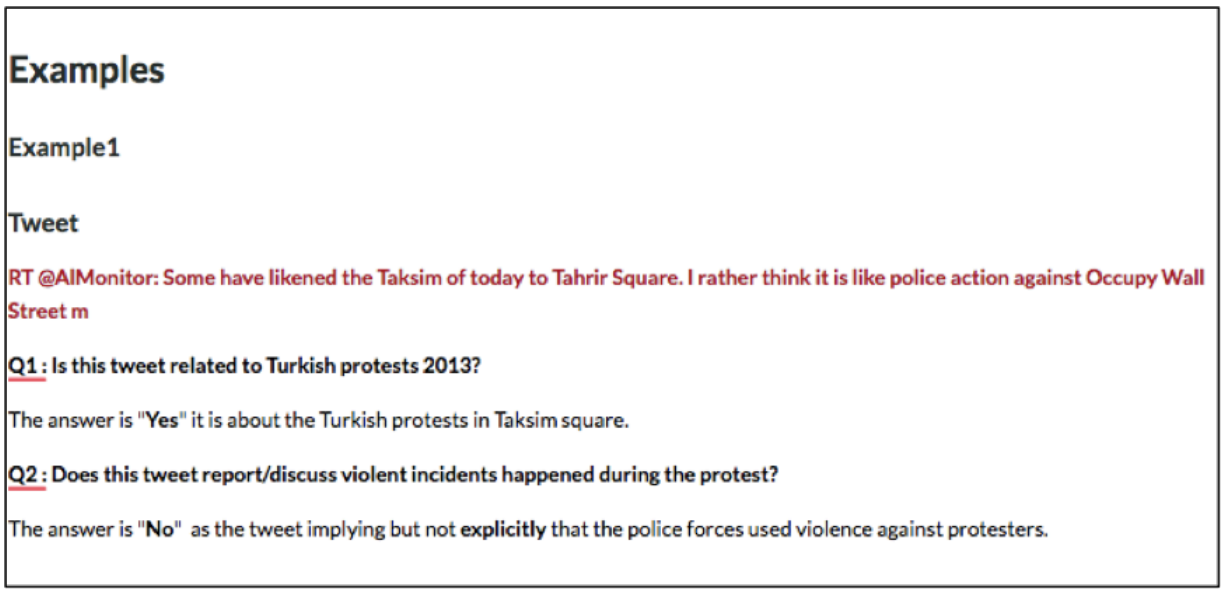

Figure 3.2: Examples of the correct answers to the asked questions in the crowdsourcing task on Figure-Eight and the reasons for those answers

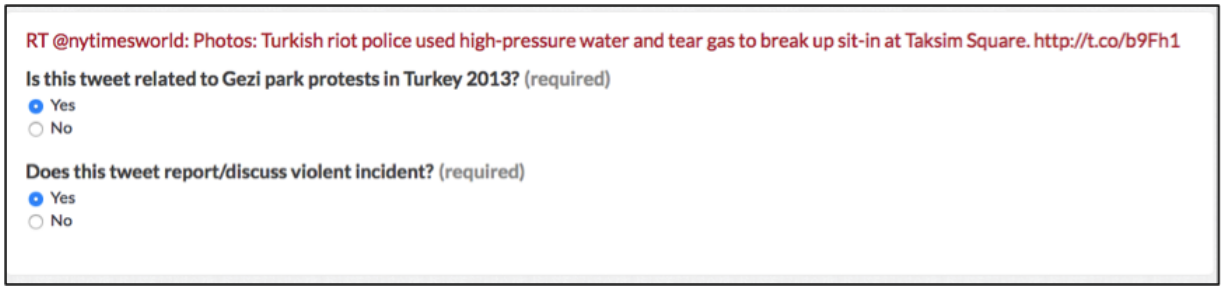

Figure 3.3: How the task looks like to the crowd worker

- Does this tweet report/discuss violent incident? The user is given two options “Yes” or “No”

We also removed the “Not sure” option because even if sometimes the tweets are confusing. It opens the door for lazy contributors to choose this answer instead of reading the tweet. From the pilot study using radio button did not seem to affect the contributors’reliability that is why we continued using radio buttons here as they take less time and we depend on other quality control settings to filter untrusted contributors.

Quality Measures

To ensure the quality of the task, Figure-Eight allows the customers to set the values that best suit their requirements.

1. **Minimum Contributor is Trust:** Figure-Eight sets the minimum contributor ’s trust to 0.7 by default. The contributors must maintain this score during the task to continue

working on the task otherwise they will be removed from the task and their answers will be disregarded. In the pilot experiment, the default contributor 's trust value of 0.7 was used. The average of trust scores of the contributors who participated in the pilot experiment is 0.98. This could be because a level of experience of 2 filtered the contributors. In the main crowdsourcing task as we reduced the level of contributor 's experience to 1, we set the contributor 'confidence level to 0.75.

2. **Dynamic Judgment Collection:** This option allows the platform to automatically collect more judgments on a row (data item) if the contributors disagree on a label/answer until a certain confidence score for the row is achieved or a specific number of judgments is met. Judgment here means a contributor 's answer. The more judgments required the more contributors would do that row of data. In the pilot experiment, this option was active and the number of judgements collected per rows ranged from 3 to 6 judgements. This enquired more money and more time to finished the task and in the same time when the inter-annotator-agreement score [42] was calculated between the best answer and our labels the score was 0.52, which is below fair. That is why in the main experiment we deactivated this option to save money and time.

3. **Judgments per row:** The customer sets the number of judgments to be collected for each row (data item). The default is 3 judgments per each row of data, which is overwritten if the dynamic judgments option is active. For the main experiment, we used the default value of 3 judgments. The more judgments you add to the task, the more time, money it costs and in some cases you get more accuracy.

4. **Row(s) Per Page:** The customer sets the number of data items in one page of work. As mentioned before the task design affects the quality of the work. To make the page of work more readable, we chose 5 rows per page with 2 questions for each row. So each page of work has 10 questions.

5. **Price Per Page:** The customer sets the price he is willing to pay per page of work to the contributors for a completed task if not rejected or found untrusted. [64] argues that the reward influences the time and quality of the task. In the pilot experiment, we set the reward to 7 cents per page but as we chose to deactivate the dynamic judgments we increased the payment to 10 cents per page of work to motivate the workers.

6. **Minimum Time Per Page:** The customer sets the minimum time, in seconds, that a contributor should take to complete a page of work. The default value is 10 seconds. To make sure that the contributor reads the tweet is content and not only the hash-tags, but we also set the min time to 100 seconds for 10 questions which are 10 seconds per question.

7. **Maximum Judgments Per Contributor:** This option limits the number of judgments that any contributor can do the job. By default, the maximum number of judgments any

contributors can submit is limited by the number of active test questions. As the contributor can not do more pages than the number of test questions available with the task. In this case, the contributor can not so more than 116 tweets. With each page containing 5 tweets, one is a test question, the contributor can not do more tweets than 468 tweets.

8. **Contributor Answer Distribution Rule:** Activating this option helps to monitor the distribution of the answers submitted by a contributor. This setting ensures that a contributor is removed if they tend to favour a specific answer. This rule is active by default in Figure-Eight and kept active for this user experiment.

Results

To find the best way to aggregate the results to train the model, we use the following different aggregation methods on the results from the pilot experiment. With reference to our labels as the correct labels, we calculate the AUC scores (discussed in chapter 2) of the labels of each of the aggregation methods. The AUC score here compares the ratio of true positive rate (TPR) to false-positive rate (FPR) between the aggregated label and our labels. The AUC score is a value between 0 and 1 with 1 is the perfect match and 0 is no match between the labels.

- **Best Answer:** the default-aggregated report by Figure-Eight which is based on the confidence score of each label. This dataset contains 6696 unique tweets.
- **Min Confidence Score:** here we use only the records with minimum confidence score 0.7. This dataset contains 3745 unique tweets for protest label with min confidence 0.7 and 5029 tweets for violence label with min confidence 0.7.
- **Full Agreement**: this dataset contains only the tweets, which gained full agreement between the three annotators. For the protest label, the dataset contains 3860 tweets and for the violence label 5248 tweets. This means that the confidence score for the labels is 1.

The AUC scores as shown in Figures 3.4 and 3.5, the AUC scores of the Full agreement aggregation and 0.7 aggregations are the best and almost the same for both protest and violence labels. We choose the full agreement aggregation method.

3.3.4 Results Analysis

Contributors

The total number of participants in the task number is 1554. The number of trusted contributor (with a trust score higher than 0.75) is 835. The trust values range from 0.77 to 1 and the mean trust value of 0.98 and the standard deviation is 0.047. The distribution of the contributor 's trust is shown in Figure 3.6. The average number of test questions that each contributor did is 11

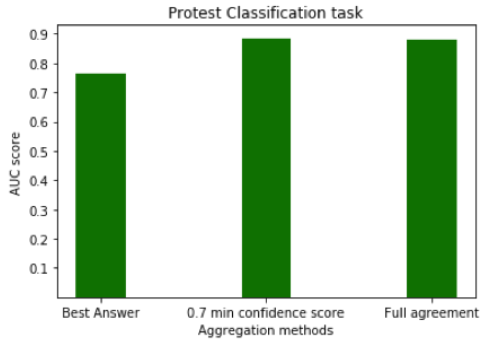

Figure 3.4: the AUC scores achieved by the different aggregation methods (protest dataset)

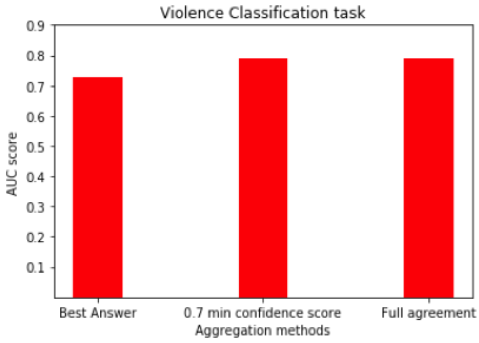

Figure 3.5: the AUC scores achieved by the different aggregation methods (violence dataset)

and ranges from a minimum value of 6 test questions to a maximum value of 61 test questions. The average number of judgments submitted by each contributor is 36 judgments range from 10 judgments to 285.

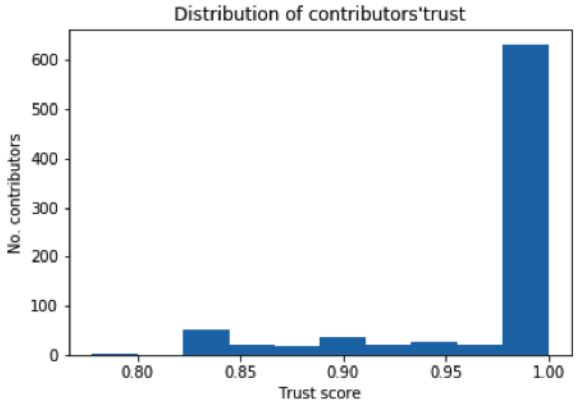

Figure 3.6: The distribution of contributors trust scores on the task

Test Questions

The golden answers (our answers) distribution of the 116 test questions for the protest label is 31 (26.49%) positive (with the answer "Yes" to the protest question) and 86 (73.5%) negative (with the answer "No" to the protest question). The distribution for the violent label is 10 (8.5%) positive (with the answer "Yes" to the violence question) and 107 (91.45%) negative (with the answer "No" to the violence question). The test questions received 10,439 judgments with an average number of 89 judgments per test question. The distribution of received answers for the protest label is 28% positive labels and 72% negative labels and for the distribution for the violence, the label is 9% positive and 91% negative. This is similar to the golden answers. 19 test questions received incorrect judgments (missed questions). Table 3.1 shows a group of the most missed test questions. The first question should be protest-positive labelled tweet and violence

Tweet	Total Votes	Pos. Protest Votes	Neg. Protest Votes	Pos. Violence Votes	Neg. Violence Votes	No. Times Missed
RT @cey-lanozbudak: 152.000 new flowers, 30 new trees were planted to #Gezi Park/ #Taksim after it was cleaned of the #protestors http://t.c	113	86	27	3	110	30
RT @selingirit: In the meantime: “Court orders demolition of park considered sacred in Tunceli, #Turkey stirring outcry ” via @HDNER http://	87	23	64	13	74	36

Table 3.1: Sample of missed test questions (incorrectly answered) by the crowd workers.

	Protest(Yes)	Protest(No)
Violence (Yes)	734	2130
Violence (No)	2306	3523

Table 3.2: The number of answers given by the crowd workers to the protest and violence questions

negative. It received 27 judgments as protest negative. This could be because it is unclear for the contributor that the tweet is discussing the clearing of the park by force. It received 3 out of 86 positive violence labels. This might be because the sentence “cleaned of #protesters” could be taken as violent, but the tweet does not explicitly indicate the use of violence (as stated in the instructions, such a tweet would not be considered violent). The second and third tweets received positive protest labels and positive violence labels while the golden label is negative-protest and negative-violent. The positive protest labels are because of the confusion that the tweet is related to the protest, albeit indirectly. In other words, the tweets do not discuss something related to what is happening on the ground in the protest. The positive violent tweets could be spam, or a simple misinterpretation of words appearing in the tweets, like “demolition”.

The mean confidence score of protest questions “Is this tweet related to Gezi protests in Turkey 2013?” for all the test questions is 0.873. And of the violence questions, “Does this

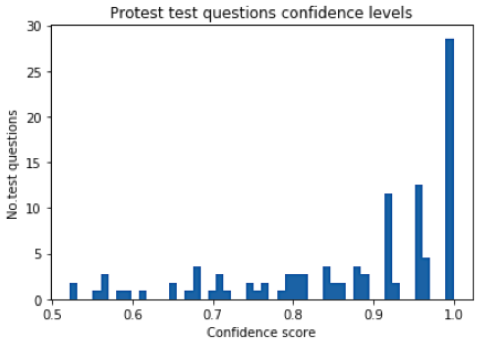

Figure 3.7: The distribution of confidence scores over test questions (protest question)

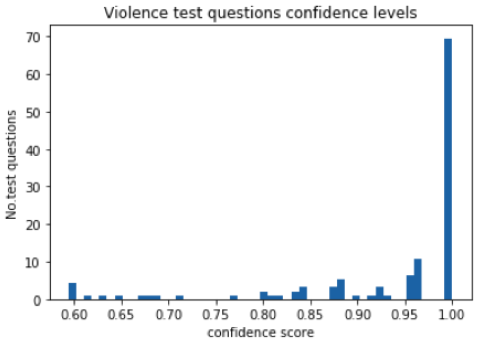

Figure 3.8: The distribution of confidence scores over test questions (violence question)

tweet report/discuss violent incident?" for all the test questions is 0.933. Figures 3.7and 3.8 show the distribution of confidence scores for the protest and violence test questions. The confidence scores range from 0.55 to 1 for the protest questions with the higher number of test questions with the confidence score 0.9 and 1. The confidence scores for the violence question range from 0.60 to 1 with the majority of test questions with confidence score 0.95 and 1.

The number of judgments received on the task is 20635 for 6693 tweets. Each of the tweets received two labels one for the protest question "Is this tweet related to Gezi protests in Turkey 2013?" and one for the violence question "Does this tweet report/discuss violent incident?" Both questions are answered with either "Yes" or "No". Table 3.2 shows the distribution of answers.

Protest Question

For the first question about the protest "Is this tweet related to Gezi protests in Turkey 2013?", the total number of "Yes" answers (positive label) is 3040 (45.42%) and the total number of "No" answers (negative label) is 3653 (54.57%). The distribution of the answers is shown below in Figure 3.9 The mean confidence score of all the answers is 0.859. The total agreement between the contributors for all the protest-related tweets is 57.7%: 35% of all the tweets were the 3 contributors agreed on the "No" answer. And 22.7% of all the tweets where the 3 contributors agreed on the "Yes" answer. On the other hand, 42% of the tweets showed disagreement on the label of the tweet. In 22.7% tweets, 2 contributors agreed on the "Yes" answer and 19.6% of the tweets only one annotator thought the tweet is protest-related.

This is reflected in the distribution of the confidence scores of the answers as shown in Figure 3.10 where 3860 tweets have protest label with confidence score 1, which means they received the total agreement. 2833 tweets have protest confidence score less than 1 because of disagreement between the contributors and based on the trust score of the contributors the tweet

received the label with the highest possible confidence score. When the Krippendorff alpha was calculated to detect the inter-annotator-agreement for the protest label with nominal data, Krippendorff alpha was 0.428 which is less than fair for nominal data according to [78]. The disagreement could because some tweets are not directly related to the protest but still uses one of the protest hash-tags that may have caused confusion for some contributors. For example, this tweet is mislabelled as protest positive "the coolest living creature ever, Tilda Swington is supporting us as well #occupygezi".

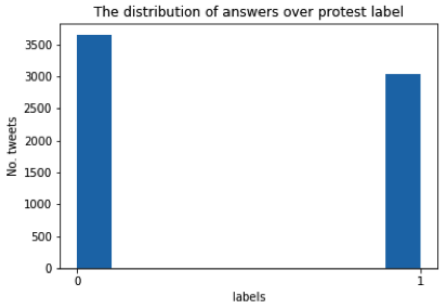

Figure 3.9: Answer distribution given by the crowd workers (protest question)

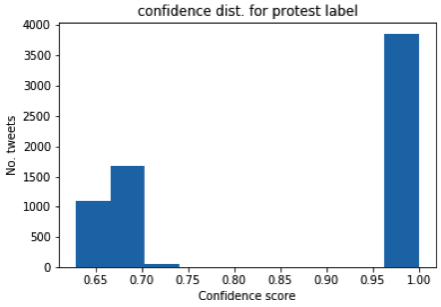

Figure 3.10: The distribution of confidence scores of the answers to the protest question

Violence Question

For the second question about the violence "Does this tweet report/discuss violent incident?", the total number of "Yes" answers is 889 and the total number of "No" answers is 5829. The distribution of the answers is shown below in Figure 3.11. The mean confidence score of all the answers is 0.928. The distribution of the confidence scores of the answers is shown in Figure 3.12. The total agreement between the contributors for all the violent label is 78.4%: 73.6% of all the tweets were answered by all the 3 contributors as "No" and for 4.8% of all the tweets, all the 3 annotators agreed on the answer "Yes". On the other hand, 21.6% of the tweets showed disagreement on the label of the tweet. In 8.1% tweets, 2 annotators agreed on the "Yes" answer and 13.5% of the tweets only one annotator thought the tweet is violent.

The distribution of the confidence scores of the answers as shown in Figure 3.12 also shows that 5248 tweets have violence label with confidence score 1, which means they received total agreement and 1445 tweets have violence confidence score less than 1 because of the disagreement between the contributors and based on the trust score of the contributors the tweet received the label with the highest possible confidence score. The Krippendorff alpha for the violence label is 0.427, which is again less than fair. Given that violence is a subjective label, it is hard to agree on violence. For example this tweet "legitimate protest, police overreact, media dumb, gov

retreats to autocracy, sigh, but Turkey is a democracy" people disagreed on "police overreact" as violent or not.

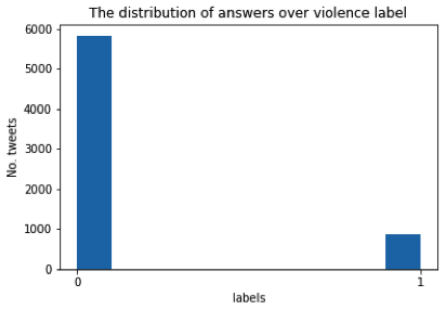

Figure 3.11: Answer distribution given by the crowd workers (violence question)

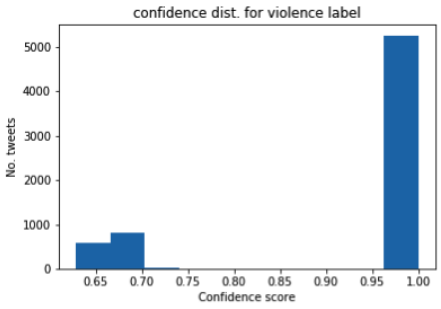

Figure 3.12: The distribution of confidence scores of the answers to the violence question

The pilot experiment the Krippendorff-alpha inter-annotator-agreement score between contributors range from 3 to 6 for each tweet is 0.64 for the protest question and 0.61 for the violence question. This shows fair agreement level for nominal data. And from the main experiment, the Krippendorff-alpha inter-annotator-agreement score between the 3 contributors for the protest label is 0.428 and for the violence label is 0.427, which is not as good as having more contributors. The two experiments showed that a decent level of agreement between non-experts could be reached for an event as violent or not.

From the reviewed literature and the 2 experiments we found that for the Figure-Eight platform, the most important quality settings of the task are:

1. **Gold units (test questions)**, they are a good way to test the performance of the contributors. It is best to provide a variety of test questions. It is better if more than one expert labels the test questions and then the best label is aggregated and uploaded the task.

2. **Contributor's trust**. The threshold for the performance of the contributor to the test question affects the confidence score of each label. Again it is a trade-off a very high trust score will guarantee high-quality data but also will take longer to find contributors.

3. **The number of judgment** as shown in the previous question's answer affects the agreement level on the label. The higher the number the higher the agreement scores which in turns affects the score of the final label that the tweet takes. Last but not least, the confidence score, the main experiment shows that labels with scores higher than 0.7 improved the text classification model 's performance. Even when the number of training dataset went down to almost 50% less, the model still did better.

Conclusion

In this chapter, we described the dataset collection and built the training dataset that will be used in training the machine learning model next chapter. We described the crowdsourcing experiment on Figure-Eight to collect answers for two questions for each of the tweet in a dataset collected during the Turkish protest Gezi park in 2013. The two questions are: 1) is the tweet related to the Turkish Gezi park protest 2013? 2) Does the tweet report/discuss violent incident? From the data experiments done and the data collected, we could provide the following answers to the research questions.

Now we have the dataset ready to train the model however, one of the concerns that will have a great influence on the model training process is the ratio between positive and negative examples. As in the original data, for the protest label, 39% of the tweets are protest-positive and 61% are protest-negative. On the other hand, the violence label makes the dataset highly skewed as only 13% of the tweets are violent positive and 87% of the tweets are violence negative. When the dataset is limited to full agreement dataset, the dataset for the protest label became almost balanced 42% protest positive and 58% protest negative. For the violence labels the gap between the positive and negative became even bigger 6% are violence positive tweets and 94% violence negative. This will have an impact on the machine-learning model, used in this research. Also, some tweets were confusing to the contributors because some words were misinterpreted. Sometimes violent and non-violent tweet would have similar words. This will have an impact on the model, as it'll be a source of confusion to associate a certain pattern with a certain label. However, as we use tweets that received full agreement we expect that impact to be small.

Chapter 4

Text Classification

In this research, we aim to automatically detect protest repression events from tweets. We propose using supervised machine learning models to detect protest-related tweets and violent tweets from the data described in chapter 3. Using supervised machine learning saves time and cost, especially when dealing with millions of tweets. We used crowdsourcing to build a labelled dataset to train the machine-learning model. In chapter 3, we discussed the data collection process and the crowdsourcing experiment to build the training dataset. Now we can train the machine learning models to detect tweets that report protest repression.

In this chapter, we try to answer the second research question, which is what is the best baseline machine-learning model between SVM and MNB to classify tweets as protest/non-protest tweets or violent/non-violent tweets? By describing the experiments done to train the machine-learning model to label the tweets. We run two groups of text classification experiments: one for protest classification and another for violence classification. In protest classification, the model learns to distinguish between the tweets that are related to the Turkish Gezi protest from other unrelated tweets. In violence classification, the model is trained to distinguish the violence-related tweets. In the two groups of experiments, we try different parameters and different text representations (BOW, TF-IDF and WE) with the two baseline models Multinomial Naive Bayes (MNB) and Support Vector Machines (SVM) and choose the best parameters and the best text representation with each model.

We evaluate the performance of each model in terms of AUC scores on different datasets and investigate the percentage of wrong and right predictions of each model and why the mode is performing in a certain way. Then, we compare the results of the two models MNB and SVM for the two task protest classification and violence classification and determine which one performs the best on the dataset. After that, we use the best performing model in protest classification to predict the protest labels and the best performing model in violence classification to predict the violence labels of the whole tweets collection. Finally, we provide a basic analysis of the time-

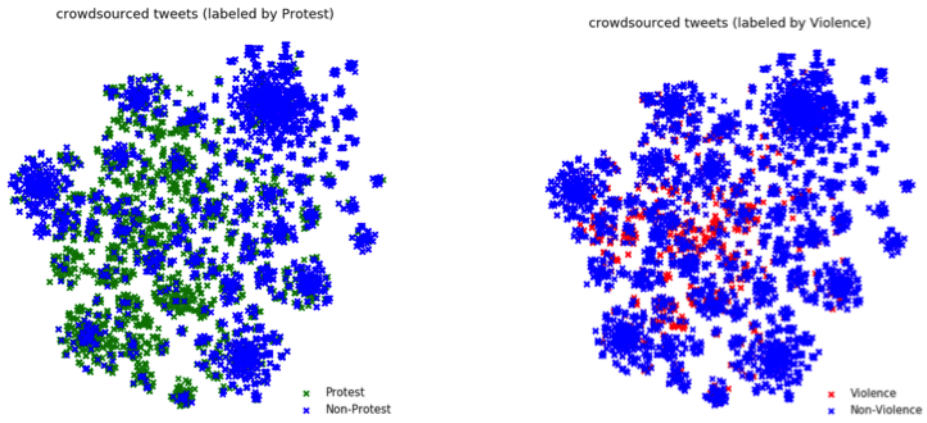

Figure 4.1: Words features in tweets collection protest dataset (left) and violence dataset (right)

line of the protest and violent tweets and the number of occurrences of the violence keywords throughout the protest period.

4.1 Data

We use the tweets dataset collected during the Turkish Gezi protest in 2013. The dataset was then labelled by crowd workers using the Figure-Eight platform. There were two questions per tweet, one asked if the tweet is related to the Gezi protest or not. The second question asked if the tweet reports any act of violence or not. Each tweet received three judgements from three different crowd workers. The final dataset was aggregated based on the agreement between the crowd workers. Only tweets with the full agreement between the three crowd workers were used. This resulted in two datasets: the protest dataset and the violence dataset. Each dataset is described in the next paragraph. The data can be visualized using t-SNE to project high dimensional data like text into a two-dimensions [47] shown in Figure 4.1.

4.1.1 The Protest Dataset

This dataset contains all the tweets that received full agreement between the three crowd workers on the protest question. The dataset has 3860 tweets with 1518 (39.3%) positive tweets (related to protest) and 2342 (60.67%) negative tweets (not-related to protest). The most occurring words in both protest positive and protest negative tweets can be seen in Figure 4.2. The figures show that there are protest-related words in the positive tweets e.g. “occupygezi”, “police”, “protests” and “protesters”. These are some of the search keywords that were used in the process of collecting the tweets as mentioned in chapter 3. Some of these search keywords exist also in the negative tweets like “turkey ” and “police” but they come with non-protest related words

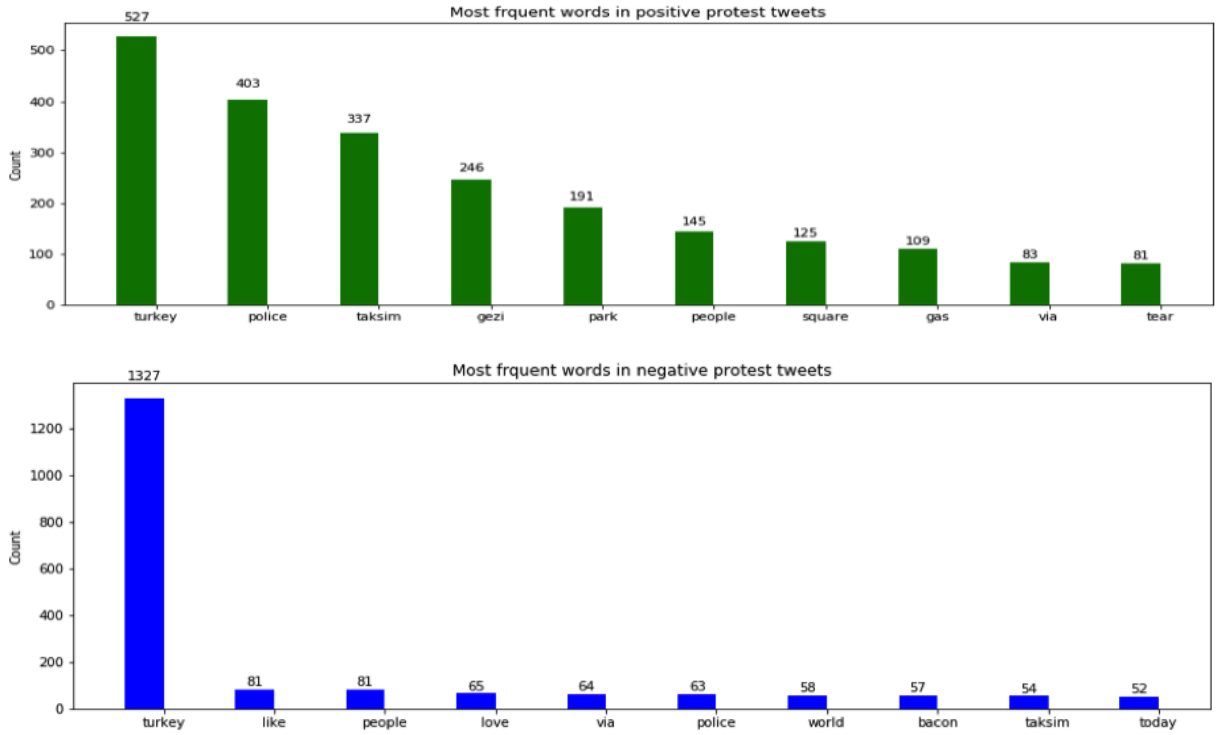

Figure 4.2: Most frequent words in protest dataset positive (top) and negative (bottom)

like “via”, “love” and “bacon”. Also, the number of occurrences of the word “police” is much higher in the protest positive tweets (403 times) compared to the negative tweets (63 times).

4.1.2 The Violence Dataset

This dataset contains all the tweets that received full agreement between the three crowd workers on the violence question. The dataset has 5248 tweets with 323 (6.15%) positive tweets (report violence) and 4925 (93.84%) negative tweets (do not report violence). Figure 4.3 shows the most frequent words in both the positive and negative tweets of the violence dataset. Here we find more common words between the negative and positive tweets like “park”, “police”, “taksim” and “gezi”. This is because these words are protest-related and some of them are among the search keywords used to collect the data. The tweets in the negative subset could be related to the protest but not necessarily reports violence. Violence related words like “tear”, “gas” and “water” are among the most frequent words in the violent positive tweets as they show the type of violence used against the protesters like tears gas bombs and water canons. We expect these common words between the positive and negative subset in both protest and violence dataset to be a source of confusion for the model as they were for some of the crowd workers in Figure-Eight. We try to reduce the source of this confusion in the feature extraction process.

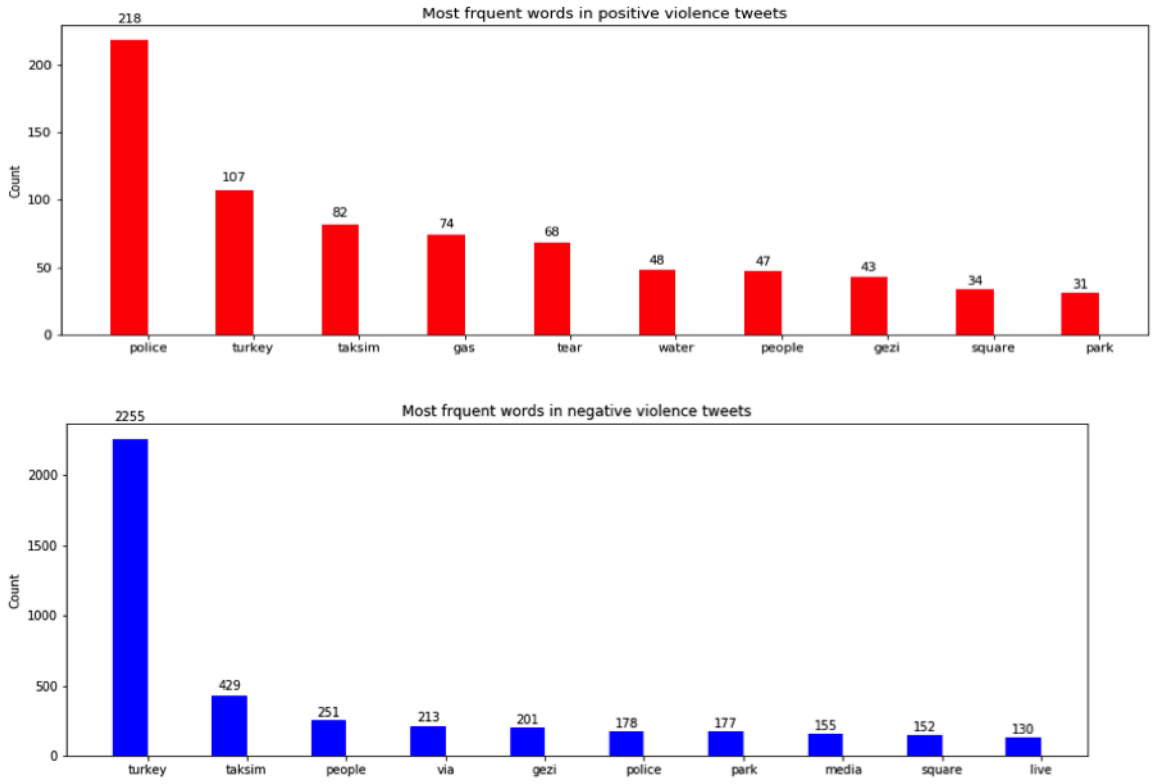

Figure 4.3: Most frequent words in violence dataset positive (top) and negative (bottom)

4.1.3 The Test Dataset

In our experiment, we test the models on 3 datasets. The first test set is a randomly chosen 50% held out dataset from the tweets collection (protest and violence datasets). The second one is ground truth data collection (GT1) of 116 test questions labelled by me and the third one (GT2) has 1214 tweets labelled in the crowdsourcing pilot experiment labelled by me. A fourth set (GT3) is the same as GT2 but labelled by crowd workers and aggregated by the best answer on Figure Eight as described in chapter 3. We use these datasets to test the model's performance on new unseen data. Another thing to see is how a model trained on crowd-sourced data would perform on a dataset labelled by me.

4.2 Feature Extraction

As explained in chapter 3, machine learning models can not work with text and words (features) must be represented as numbers. We use three different approaches to represent the text dataset as numbers. We then compare the different approaches of text representation to find out the best way to represent our dataset. We use a simple BOW model because it is the simplest, the fastest and proved to perform well. Tweets were collected using search keywords as mentioned in section 4.1, which causes that many words exist in all the tweets even if they do not add

information like the hash-tag #gezi. That is why we also use TF-IDF to penalize these common words and use only important features. We also use word vectors or word embeddings (WE) as a distributed representation because some words tend to come together like “tear gas” or “gezi park” and using word vectors here will create a vector for each word taking into consideration the words surrounding that word.

4.2.1 Pre-processing

Before we start extracting the features from the text, we need to clean the text first. Because tweets are short messages of 140 characters that might include mentions (@UserName), http addresses (http//:www.anything.com), hash-tags (#Keyword), digits and the actual textual content of the message. Sometimes Twitter users copy the same message after adding a mention or a hash-tag to it. Twitter treats it as a new one and this is a source of duplication in our dataset. So we start by normalizing all the tweets. We use regular expressions to replace user mentions, http links and digits with the keywords “usrid”, “httpaddress” and “dd”. Then, for each tweet we tokenize the tweet, convert each word to lower case and remove the normalized keywords because they occur a lot in the tweets and they do not carry valuable information for this research. We also remove stop words, any words that is less than 2 characters, emotion icons, punctuation and special characters like “#”. Finally, we make sure that each tweet after the cleaning step is at least 3 words long. We remove all other tweets that are less than three words long, as they do not have enough information. Then, we remove duplicated tweets. We pre-process the tweets in both the protest and violence datasets. The output of the pre-processing step is: the protest dataset now contains 3666 tweets, 1497 (48.83%) positive tweets and 2169 (59.1%) negative tweets and the violence dataset now has 4975 tweets, 322 (6.4%) positive tweets and 4653 (93.52%) tweets.

4.2.2 Bag Of Words (BOW)

We use the feature extraction CountVectorizer[1] function in the sklearn package[2]. As we do the pre-processing step separately, we do not pass any special parameters to this function. This function generates the integer count representation for each word in the tweet as explained in chapter 2.

[1] `https://scikitlearn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html`

[2] `https://scikit-learn.org/stable/`

4.2.3 TF-IDF

We use the TfidfVectorizer[3] function in the sklearn package. Here, we also do not pass any parameters to the function because we did the pre-processing step separately. This function generates the weight of each word in the tweet. This weight represents the importance of that word in the tweet and the tweets collection. As tweets are short, it is unlikely that the word frequency (TF) in one tweet would be high, however, it is expected that for some words related to the protest the word frequency in the corpus (DF) would be high.

4.2.4 Word Embedding (WE)

We use the word2vector function in the Gensim package[4] to produce the word embedding representation of the tweets. The word2vector function takes two important parameters. The first is the vector size and the second is the window, which is the number of words surrounding the current word to take into consideration while creating the vector for this word. In a pilot experiment, we chose different combinations of parameters' values to test on our data. The parameters' values were chosen from this study [89] because it also worked on classifying tweets. The different parameter values were dimension sizes = {200,500,800} and window sizes = {1,3,5} with SVM. The best performance was associated with window = 3 and size = 800 and this is the one we use here. Then we calculate the average vector for all the words' vectors in one tweet. Now each tweet is represented as one vector. We train the word2vector model for 100 epochs to generate the vectors.

4.3 Model Training

In this section, we explain the python packages used to implements the machine learning models along with the different parameters of each python methods and the values we use for these parameters.

Support Vector Machine (SVM)

We use the SVC[5] function in the sklearn.svm package. The function is an implementation of the SVM model descried in [34] [18]. According to the description provided in chapter 2, there are three parameters: the kernel, which defines the type of the model linear or non-linear. The second parameter is the C parameter, which decides how soft the margin that separates the two classes is. The third parameter is Gamma γ, which affects how complex the decision boundary

[3] https://scikitlearn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[4] https://radimrehurek.com/gensim/models/word2vec.html

[5] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

	No. Features	No. Samples
Protest dataset	7750	3666
Violence dataset	8993	4152

Table 4.1: The number of features and the number of samples in the protest and the violence dataset.

is. γ is influential only if the kernel is non-linear. According to the literature if the number of features is higher than the number of the training data points, then the linear kernel is the best to use with SVM [38]. This is the case in our data as shown in Table 4.1. This is why we investigate the different margin C values with linear kernel SVM.

Multinomial Naive Bayes (MNB)

We use the Multinomial Naive Bayes function[6] in the sklearn package which provides an implementation of the Multinomial Naive Bayes models described in [49] [52]. The function takes three parameters. The first is alpha α, the smoothing parameter, and its default value is 1 (Laplace smoothing). If alpha $\alpha > 0$, then the count of each feature (word) will increase by α, this will prevent having zero probabilities for words that are not present in the training sample. If alpha $\alpha < 1$, this is called Lidstone smoothing. We set the following values to $\alpha = \{0.01, 0.1, 1, 2\}$, with $\alpha = 0.01$ means that the sum of the number of times a feature appears in the likelihood estimate equation (Equation 2.16) increases by 0.01 and if $\alpha = 2$, the feature count increases by 2. The different values will have different effects on the performance of the model. The second parameter is *fit-prior*, which is a Boolean parameter. If set to true the model learns the *class-prior* probabilities. The default value is true which is what we use. The third parameter, *class-prior*, is a list of the prior class distribution.

- Uniform prior $= 1/C$ with C is the number of classes.
- Class size prior $= N_c/N$ where N_c is the number of samples in the training set that belong to class C and N is the number of samples in the training dataset.

We run a grid search to choose the best values for the parameters. We also investigate the performance of MNB with different text representations: BOW and TF-IDF. We cannot use MNB with WE directly because WE could contain negative weights and this is not compatible with Naive Bayes. There are some workarounds to get MNB to work with WE but this is not investigated here.

[6] https://scikit-learn.org/stable/modules/naive_bayes.html-multinomial-naive-bayes

	C = 1	C = 10	C = 100	C = 1000
BOW	**0.8954**	0.8901	0.8881	0.8895
TF-IDF	**0.8965**	0.8807	0.8827	0.8808
WE	0.8726	**0.8757**	0.8695	0.8629

Table 4.2: The AUC scores of the linear SVM model with different margin C values for each text representations {BOW, TF-IDF, WE} on the protest dataset

4.3.1 Protest Classification

In this section, we start investigating the performance of SVM and MNB models on the protest dataset in terms of AUC scores. We aim to find the best text representation, model parameters for each baseline model, in terms of AUC scores, on the protest dataset. The outcome of this investigation will be used in predicting the protest labels of the rest of the unlabelled tweets.

Support Vector Machine (SVM)

We start with investigating the model's parameters to find out how good the model would be in generalizing to new unseen data. We run a grid search with the following parameters values: $C = \{1, 10, 100, 1000\}$. We randomly split the dataset into 2 equal sizes: a training dataset and a test dataset using the train-test-split[7] function in the sklearn package. We train the linear SVM on a held-out fold in 10 folds cross-validation on the training set with the different margin values C. Then, we test it on the test datasets. We do the same process for BOW, TF-IDF and WE. We repeat this process 10 times and then average out the resulted AUC scores. The best performing parameters are summarised in Table 4.2. According to the results, BOW and TF-IDF representations are better than WE.

Then, we investigate if there is a significant difference between the AUC scores of the different representations. We use the Wilcoxon statistical test [22] to compare the AUC scores. We repeat the train/test split process for 30 times and each time we record the AUC scores of the SVM model with different text representation on the test set, then we perform the Wilcoxon test on the sample of 30 runs. The null hypothesis here is that there is no difference between the different text representations (BOW, TF-IDF and WE). The results as shown in Table 4.3, the p-value is less than 0.05 for all the comparisons which means that TF-IDF is significantly better than BOW and WE.

Now, we test the model's performance with the selected parameters: Kernel = linear and $C = 1$ and TF-IDF representation on the different test sets mentioned in section 4.1.3. We train the model on the training dataset which contains 1833 protest tweets with 741 (40.4%) positive

[7] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html`

Model 1	Model 2	p-value
Tf-IDF	WE	<0.05
TF-IDF	WE	<0.05
BOW	TF-IDF	<0.05

Table 4.3: Wilcoxon significance test results between the the best AUC scores of the linear SVM model with each text representation {BOW, TF-IDF, WE} of the protest dataset

.

	Positive samples	Negative samples	AUC
Test set (1833)	756 (41.2%)	1077 (58.7%)	0.8991
GT1 (116)	31 (26.72%)	85 (73.2%)	0.840
GT2 (1213)	515 (42.456%)	698 (57.54%)	0.817
GT3 (1213)	590 (48.63%)	623 (51.3%)	0.828

Table 4.4: Linear SVM model's performance (AUC scores) with $C = 1$ and TF-IDF on each test set

tweets and 1092 (50.957%) negative tweets. Then, the model is tested on the test set, GT1, GT2 and GT3. Table 4.4 displays the size and percentage of positive and negative samples in each test set and the model's performance in terms of AUC scores.

As discussed in chapter 2, the AUC score is the ratio between the True Positive Rate (TPR) and False Positive Rate (FPR). TPR is the rate of positive samples that are correctly labelled as positive and the FPR is the rate of negative samples that are misclassified as positive. The closer the AUC score to 1, the better is the model's performance. On the different datasets: Test set, GT1, GT2 and GT2, the model gives scores ranging from 0.81 to 0.89Ṫhis means that the model has high TPR and low FPR. The model gives the highest AUC score on the test set because they come from the same distribution as the training set. GT1 is labelled by us and received 0.840 AUC score. This might be because the percentage of positive samples is small 26.7%. GT2 and GT3 are the same dataset but labelled by us and crowd-workers, respectively. The results are close to 0.817 and 0.828.

To investigate more, Table 4.6 and Table 4.7 show examples of correctly classified and misclassified tweets.The first tweet in Table 4.6 has the words: "turkey", "blue", "cheese" and "apricot". These words are not related to the protest and the model did not see them in the

	No. right predictions	No. wrong predictions
Test set (1833)	1649 (90%)	184 (10%)
GT1 (116)	101 (87%)	15 (13%)
GT2 (1213)	981 (81%)	232 (19%)
GT3 (1213)	1004 (83%)	209 (17%)

Table 4.5: Comparison between the number of right and wrong predictions by the Linear SVM model with $C = 1$ and TF-IDF on each test set

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
recipe turkey panini blue cheese cranberry apricot chutney	0	0	0.97	0.03
Turkish police have blocked out id numbers on their helmets at #occupygezi #dierengezipark why?	1	1	0	1

Table 4.6: Examples of correctly labelled tweets by the Linear SVM model is with $C = 1$ and TF-IDF

training dataset associated with positive labels hence the model gives them a negative weight. That is why the model correctly labelled the tweet as negative. In the second example, the tweet has the words "Turkish", "police", "occupygezi" and "dierengezipark" which are related to the protest and received positive weights from the model. That is why the model gave a positive label to the tweet. In the misclassified tweets (Table 4.7), the first tweet has the words "police" and "Istanbul" which are related to the protest and have positive weights that are why the model gave positive label even when the tweet is actually not related to the protest. The second tweet is one of the tweets in GT1 dataset, which is labelled by us. It is a bit confusing as it is related to the protest but not directly related to what is happening on the ground. It is one of the tweets that are confusing to the human labeller as well. For the model, the tweet has only one word with positive weight "direngeziparki". The rest of the words do not have a high positive weight that is why the model gave it a negative label that is, in this case, closer to the right label than the human labeller. Another explanation of the wrong predictions is that the margin is soft $C = 1$ which makes the number of support vectors is high and the model is flexible enough to allow samples to be on the wrong side of the decision boundary.

In the previous examples, the model is confident as it assigns high probabilities to the predicted labels and low probabilities to the other opposite label. These probabilities are a good indicator of how certain the model is. If the model gives high probabilities to the predicted labels, it means that there is a pattern in the data that the model was able to detect even with some wrong predictions sometimes. If the model gives close probabilities to the assigned labels and the other label, this means that the model is confused. This might happen because the tweets have similar words with close positive and negative weights. To investigate the model's overall certainty, we show the probability distribution of the model's predictions on the different test sets in Figures 4.8, 4.9 and 4.10.

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
auslander raus istanbul police tells claudia roth go back would tell cem	0	1	0.2	0.8
frightening coming from #direngeziparki appears to be the main hashtag used	1	0	0.71	0.28

Table 4.7: Examples of mis-classified tweets by the Linear SVM model is with $C = 1$ and TF-IDF

Positive support vectors	798 (10% of the total number of words features)
Negative support vectors	1112 (14.3% of the total number of words features)

Table 4.8: The number of positive and negative support vectors of the Linear SVM model is with $C = 1$ and TF-IDF on the protest dataset

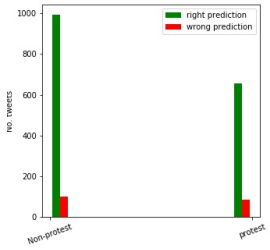

Figure 4.4: The No. of right and wrong predictions of each label (Non-protest and protest) of SVM on protest Test set

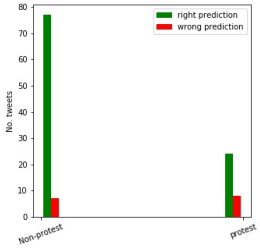

Figure 4.5: The No. of right and wrong predictions of each label (Non-protest and protest) of SVM on protest GT1

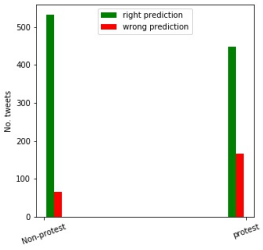

Figure 4.6: The No. of right and wrong predictions of each label (Non-protest and protest) of SVM on protest GT2

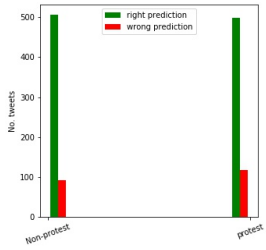

Figure 4.7: The No. of right and wrong predictions of each label (Non-protest and protest) of SVM on protest GT3

The figures show that the model is most certainly in most of the predictions as the majority of the negative and positive predictions associated with high probabilities apart from some cases where the model gives close probabilities to the negative and positive predictions. We show a sample of tweets in each set that received close prediction probabilities and wrong prediction in Table 4.9. We can see that the first tweet in Table 4.9, has the word “tweeting”, which has a negative influence on the model and the word “gezi” that has a positive influence on the model. This is why the mode gives close prediction probabilities. In the second example, the words police that have a positive influence on the model but the model gives a higher probability to the

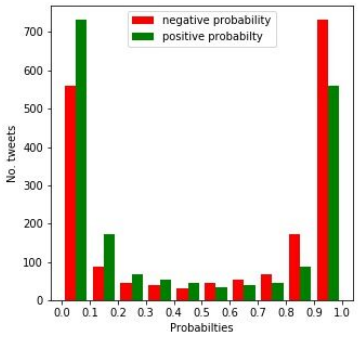

Figure 4.8: SVM probability distribution on Test set

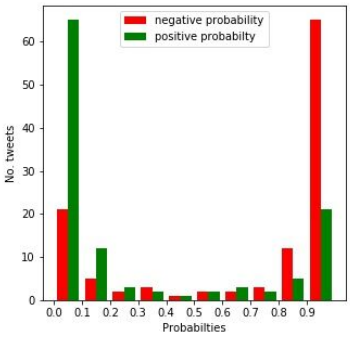

Figure 4.9: SVM probability distribution on GT1

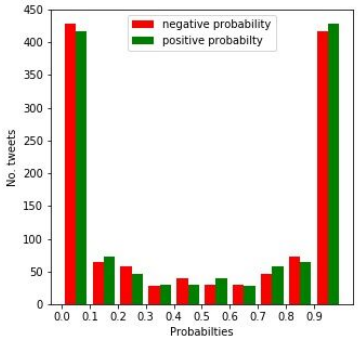

Figure 4.10: SVM probability distribution on GT2

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
everyone tweeting organizations gezi led someone go ask campers	1	1	0.39	0.60
watch riots police brutality and revolution in turkey june 2013 #nwo turkey #policestate	1	0	0.64	0.35

Table 4.9: Examples of tweets prediction with close probabilities by the Linear SVM model's with $C = 1$ and TF-IDF

negative label. This could be because words like "nwo", "june" and "2013" which have negative weights. Examples of words with positive and negative influence on the model are given in Figure 4.11.

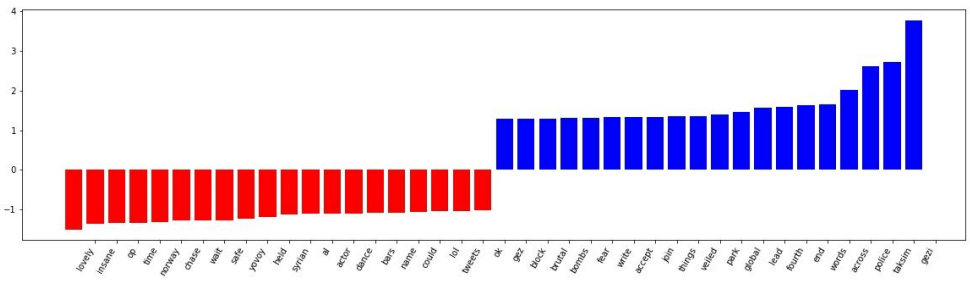

Figure 4.11: The most influential 20 words on the Linear SVM model is with $C = 1$ and TF-IDF on the protest dataset. Blue bars are words with positive influence and red bars are words with negative influence on the model

Although there are cases or misclassifications and close prediction probabilities, the model

still performs well on the unseen data and the percentage of the close perdition probabilities are small. In the next section, we test the performance of the MNB model on the same dataset and compare between the two models.

Multinomial Naive Bayes (MNB)

Now we start investigating the best parameters of Multinomial Naive Bayes model that fit the protest dataset. We run a grid search with the following values: $\alpha = \{0.01, 0.1, 1, 2\}$
$Classprior = \{Uniform - prior, Class - size - prior\}$
$Uniform - prior = \{0.5, 0.5\}$
$Class - size - prior = \{Positive = 0.4, Negative = 0.59\}$

We follow the same split technique used before with SVM which is randomly splitting the protest dataset into a training set (50%) and test set (50%). We run 10 folds cross-validation on the training dataset and report the AUC score on the validation set. Then, we run the different combinations on the test set. Table 4.10 shows the performance of the MNB model's AUC scores with the different parameters with each text representation. The best results are in bold and come from the following combinations:
BOW : $\alpha = 2$,$Classprior = Uniform - prior$
TF-IDF : $\alpha = 2$,$Classprior = Uniform - prior$

Next, we run the statistical test on the model performance on BOW and TF-IDF. We randomly split the dataset to equally sized train and test set and measure the model's performance in terms of AUC scores on the different representations. We repeat this 30 times. We set the null hypothesis to be that there is no difference in the model's performance between the BOW and the TF-IDF representations. Then, we run the Wilcoxon test on the sample of 30 AUC scores for both BOW and TF-IDF. The statistical test shows that the p-value is less than 0.05 so we can reject the null hypothesis. This means that BOW is significantly better than TF-IDF.

Now, we test the model's performance MNB ($\alpha = 2$ and $classprior = uniformprior$) on the different test set. We train the model on BOW representation of the protest data, which has 1833 protest tweets with 741 (40.4%) positive tweets and 1092 (50.957%) negative tweets.

The model is evaluated by testing it on the test set, GT1, GT2 and GT3. The model's performance is shown in Table 4.11. The results show high performance on the test set because it is coming from the same distribution as the training dataset. It also performs well on GT1 (test questions), which is labelled by us. On the other hand, the performance drops on GT2 (labelled by crowd-workers) and GT3 (labelled by us). The number of right predictions that the model makes compared to the number of wrong predictions is shown in Table 4.12 and figures 4.12, 4.13, 4.14 and 4.15. Examples of correctly classified tweets and misclassified tweets

Text representation	Class prior	$\alpha = 0.01$	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 2$
BOW	Uniform prior	0.8110	0.8110	0.8531	**0.8563**
	Class size prior	0.7993	0.82348	0.8425	0.8362
TF-IDF	Uniform prior	0.77781	0.7781	0.8358	**0.8463**
	Class size prior	0.7741	0.7940	0.7741	0.7394

Table 4.10: The MNB model's performance in terms of AUC score with the different parameter and text representations of the protest dataset

	Positive samples	Negative samples	AUC
Test set (1833)	756 (41.2%)	1077 (58.7%)	0.857
GT1 (116)	31 (26.72%)	85 (73.2%)	0.864
GT2 (1213)	515 (42.456%)	698 (57.54%)	0.76
GT3 (1213)	590 (48.63%)	623 (51.3%)	0.79

Table 4.11: MNB model is performance (AUC scores) with ($\alpha = 2$ and $classprior = uniform - prior$) and BOW representation on each test set

	No. right predictions	No. wrong
Test set (1833)	1542 (84%)	291 (16%)
GT1 (116)	100 (86%)	16 (14%)
GT2 (1213)	898 (74%)	315 (26%)
GT3 (1213)	955 (79%)	258 (21%)

Table 4.12: Comparison between the number of right and wrong prediction by the MNB with ($\alpha = 2$ and $classprior = uniform - prior$) and BOW representation on each test set

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
hopping brothers turkey repectful	0	0	0.74	0.25
bbc news turkey police clash istanbul gezi park protesesters	1	1	0	1

Table 4.13: Examples of correctly labelled tweets by the MNB model with ($\alpha = 2$ and $classprior = uniform - prior$) and BOW representation

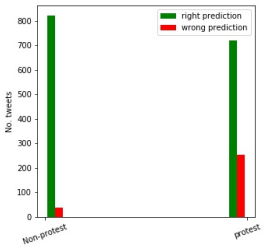

Figure 4.12: The No. of right and wrong predictions of each label (Non-protest, protest) using MNB on protest Test set

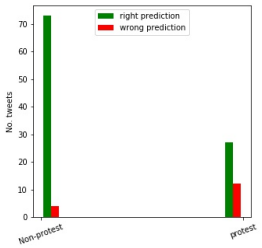

Figure 4.13: The No. of right and wrong predictions of each label (Non-protest, protest) using MNB on protest GT1

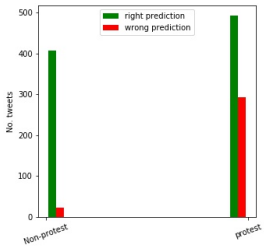

Figure 4.14: The No. of right and wrong predictions of each label (Non-protest, protest) using MNB on protest GT2

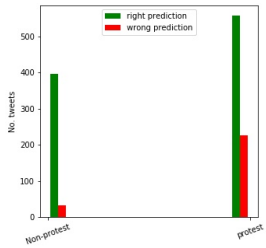

Figure 4.15: The No. of right and wrong predictions of each label (Non-protest, protest) using MNB on protest GT3

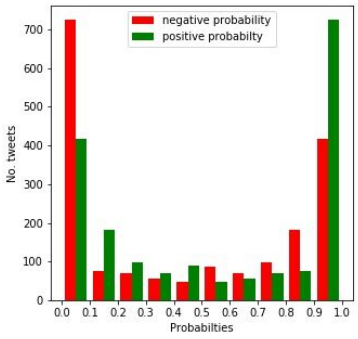

Figure 4.16: MNB probability distribution on Test set

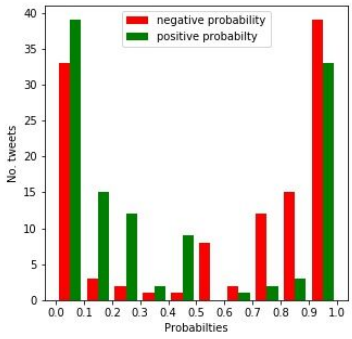

Figure 4.17: MNB probability distribution on GT1

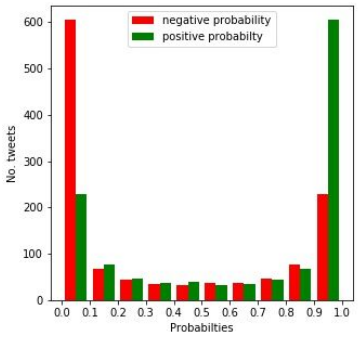

Figure 4.18: MNB probability distribution on GT2

are provided in Tables 4.13 and 4.14. The examples of correctly classified tweets in Table 4.13 follow the same pattern as the SVM model. The negatively labelled tweets (not related to the protest) has majority of words are not related to the protest which gives them low weights like “hopping” and “brothers” while positively labelled tweets (related to the protest) have majority of words with high weights like “police”, “clash”, “istanbul”, “gezi” and “protesters”. On the other hand, the misclassified tweets do not show a pattern. Although the first example has words like “canergelmis”, “init”, “many”, “watching” and “trust”, the tweet is classified as positive. These words are not related to the protest and expected to have low weights and a negative label.

This might be because: $\alpha = 2$, as these words are not related to the protest, it is unlikely that the model saw them in the training set which makes the number of their occurrences 0 or very small and here α adds 2 which makes their weight a bit higher. In the second example of the misclassified tweet in Table 4.14, the model gives a negative label to the tweet while the tweet has words related to the protest like “erdogan”, “criminal” and “arrested”. By looking at Figure 4.16 we see that the model gives the word “erdogan” negative weight, this might be because in

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
canergelmis cnn init many turks watching us trust media dir	0	1	0.08	0.916
accordingly erdogan committed criminal acts arrested fake social media accounts	1	0	0.72	0.27

Table 4.14: Examples of mis-classified labelled tweets by the MNB model with ($\alpha = 2$ and $classprior = uniform - prior$) and BOW representation

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
details taksim quora	0	0	0.58	0.41
feel ankara get enough support life stream staystrong ankara	1	0	0.50	0.49

Table 4.15: Examples of labelled tweets that received close predication probabilities by the MNB model with ($\alpha = 2$ and $classprior = uniform - prior$) and BOW representation

the training dataset the word “erdogan” existed in both positive and negative tweets and because the number of negative samples is higher than the number of positive samples.

To investigate the model’s probability distribution of the model’s predictions on the different test sets, we show the probability distribution of the model’s predictions on the different test set in Figures 4.16, 4.17 and 4.18. Table 4.15 shows examples of tweets that received predictions with close probabilities. The first tweet in Table 4.15 received prediction with close probabilities (0.6 and 0.4). This might be because the tweet has only 3 words. One of the words “taksim” is one of the most occurring in the protest dataset in both positive and negative samples but more in the positive ones. The other two words “details” and “quora” are not among the most frequent words, especially in the positive samples. That is why the model gives them low weights and a low probability.

The second tweet is not directly related to the protest (what is happening on the ground) that is why the right label should be 0. The model gets confused because the words “ankara” occurs twice in that tweets and it is one of the words with high weights. However, the rest of the words mostly have low weights that are why the model gives very close prediction probabilities (0.41, 0.49). Every time the model gives almost equal probabilities like this, the model gives a negative

label to the tweet.

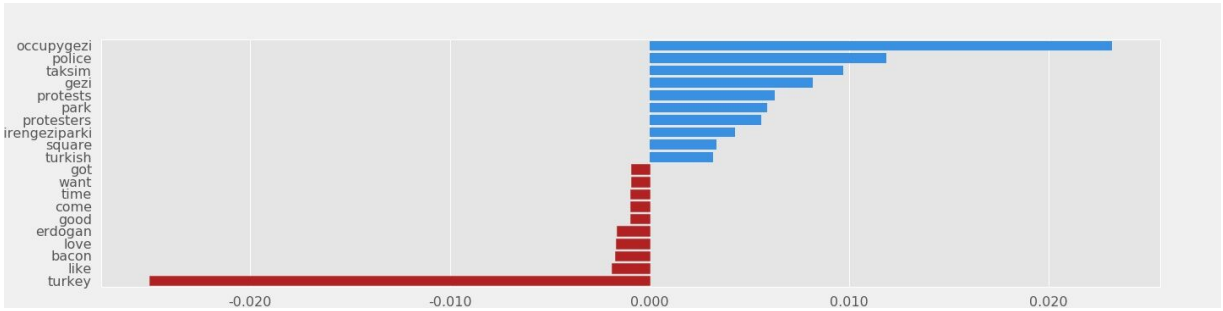

Figure 4.19: The most influential words on the MNB model with ($\alpha = 2$ and $classprior = uniform - prior$) and BOW representation. Blue bars are words with positive influence and red bars are words with negative influence on the model

The overall performance of MNB in terms of AUC scores is good on the test set and GT1 (0.85 and 0.86 respectively) and the performance on GT2 and GT3 (0.76 and 0.79 respectively) shows a tendency for higher False Positive Rate (FPR). So far the SVM model did better than the MNB on the protest dataset. In the next section, we investigate the statistical difference between the SVM and the MNB on the protest dataset.

Model Selection

Now we compare the performance of the SVM (kernel=linear, C=1) with TF-IDF representation of protest dataset against the performance of MNB ($\alpha = 2$ and $class - prior = uniform - prior$) with BOW representation of protest dataset. Here we use McNemar statistical test. We set the null hypothesis to be that there is no difference in performance between the MNB and SVM models. We compare the AUC scores of each of MNB and SVM on each of the test datasets, GT1, GT2 and GT3 and show the results in Table 4.16. Because McNemar requests that the number of instances in the False Negative(FN) and the False Positive (FP) must be larger than 50, in the case of GT1 we use the binomial distribution to mitigate for the small number of values [60].

The statistical test shows that we can reject the null hypothesis for the test set, GT2 and GT3 and accept the null hypothesis for GT1. This means that SVM significantly outperforms MNB on the test set, GT2 and GT3 while MNB insignificantly outperforms SVM on GT1. This could be because of the small size of the dataset. After comparing the performance of the two models, we decide to use SVM (kernel= linear, C =1) with TF-IDF to predict the protest labels of the rest of the unlabelled tweets in the original tweets collection described in chapter 3.

	Positive samples	Negative samples	MNB	SVM	P-value
Test set (1833)	756 (41.2%)	1077 (58.7%)	0.857	**0.899**	<0.01
GT1 (116)	31 (26.72%)	85 (73.2%)	**0.864**	0.840	1.00
GT2 (1213)	515 (42.456%)	698 (57.54%)	0.76	**0.817**	<0.01
GT3 (1213)	590 (48.63%)	623 (51.3%)	0.79	**0.828**	<0.01

Table 4.16: Comparison between MNB model is performance (AUC scores) with ($\alpha = 2$ and $classprior = uniform - prior$) and BOW representation versus Linear SVM model is with($C = 1$) and TF-IDF on each test set

4.3.2 Violence Classification

Here we investigate the performance of the selected baseline models (SVM and MNB) on the violence dataset. We start with investigating the best parameters to use with the SVM and MNB models to fit the violence dataset using different text representations BOW, TF-IDF and WE.

Support Vector Machine (SVM)

As mentioned before, when the number of features is higher than than the number of training samples, it is best to use linear SVM. Here, we use linear SVM and run a grid search to find the best margin parameter (C). C = {1, 10, 100, 1000}

I follow the same split technique used with the violence dataset. We randomly split the dataset into two: the training dataset 50% and the test set 50%. We run SVM on a held-out fold in 10 folds cross-validation on the training set with different margin (C) values. Then, we test the same margin values on the test set with different text representation models (BOW, TF-IDF and WE). We generate the WE using the same settings used in the protest classification: dimension size = 800 and context window = 3. We repeat this process for 10 times and we calculate the average of the resulted AUC scores.

The results of the SVM model using different parameters are summarised in Table 4.17. Table 4.17 shows that the best performance of the Linear SVM model on violence dataset with BOW is when the margin value $C = 1$ but with TF-IDF and WE, the best performance obtained with bigger margin $C = 10$. The results also show that similar to protest classification, BOW and TF-IDF representations are better than WE. We use the Wilcoxon statistical significance test to compare the AUC score of BOW and TE-IDF. We follow the same method of running the model for 30 times and run the Wilcoxon test on the sample of 30 AUC scores. The null hypothesis here is that there is no difference between the two text representations (BOW and TF-IDF). The results as shown in Table 4.18, the p-value is less than 0.05 for all the comparisons which means that TF-IDF is significantly better than BOW and WE. Now, we test the model's performance (linear kernel with C =10) by training it on TF-IDF representation of the violence dataset.

	C = 1	C = 10	C = 100	C = 1000
BOW	**0.8073**	0.8018	0.7963	0.7972
TF-IDF	0.7689	**0.8127**	0.8007	0.7860
WE	0.7555	**0.7689**	0.7597	0.7585

Table 4.17: The AUC scores of the linear SVM model with different margin C values for each text representation {BOW, TF-IDF, WE} on violence dataset

Model 1	Model 2	p-value
Tf-IDF	WE	<0.05
BOW	WE	<0.05
BOW	TF-IDF	<0.05

Table 4.18: Statistical comparison between the the best AUC score of the linear SVM model with different margin C values for each text representation {BOW, TF-IDF, WE} on violence dataset

The model is evaluated by testing it on the four test sets we have: Test set, GT1, GT2 and GT3. The model's performance is shown in Table 4.19. The results show higher AUC score on the test set than the rest (GT1, GT2 and GT3). That is because it is coming from the same distribution as the training dataset.

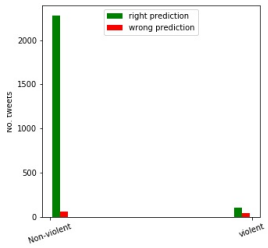

Figure 4.20: The No. of right and wrong predictions of each label (Non-violence and violence) using SVM on violence Test set

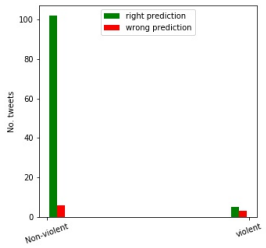

Figure 4.21: The No. of right and wrong predictions of each label (Non-violence and violence) using SVM on violence GT1

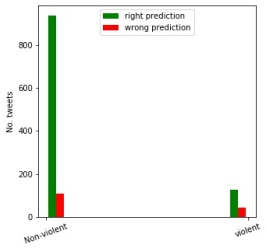

Figure 4.22: The No. of right and wrong predictions of each label (Non-violence and violence) using SVM on violence GT2

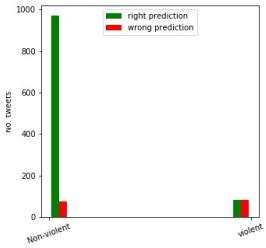

Figure 4.23: The No. of right and wrong predictions of each label (Non-violence and violence) using SVM on violence GT3

	Positive samples	Negative samples	AUC
Test set (2488)	164 (6.9%)	2324 (93.4%)	0.8127
GT1 (116)	11 (9.4%)	105 (90.5%)	0.7129
GT2 (1213)	235 (19.37%)	978 (80.6%)	0.7439
GT3 (1213)	158 (13.02%)	1055 (86.97%)	0.7260

Table 4.19: SVM model is performance (AUC scores) with ($C = 10$) and TF-IDF representation on each test set

	No.right prediction	No.wrong prediction
Test set (2488)	2391 (96%)	97 (4%)
GT1 (116)	107 (92%)	9 (8%)
GT2 (1213)	1060 (87.4%)	153 (12.6%)
GT3 (1213)	1055 (86.9%)	158 (13%)

Table 4.20: The number of right and wrong predictions of the linear svm with ($C = 10$) and TF-IDF representation on each test set

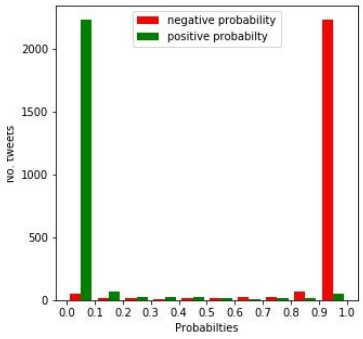

Figure 4.24: SVM probability distribution on Test set

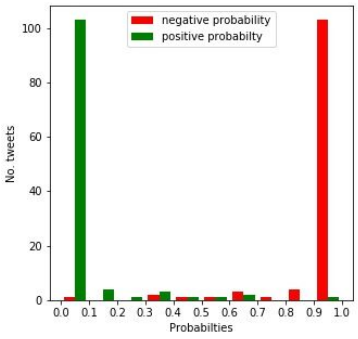

Figure 4.25: SVM probability distribution on GT1

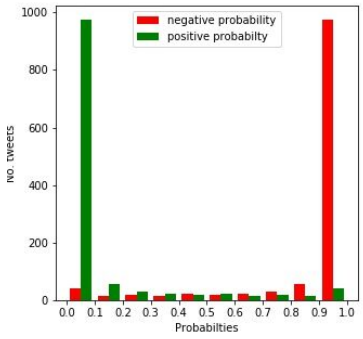

Figure 4.26: SVM probability distribution on GT2

Tables 4.21 and 4.22 show examples of right and wrong predictions of the model. The first tweet in the correctly classified tweets, Table 4.21, has the words “read”, “comprehensive” and “news”, which are not related to violence, this is why the model gave them negative weights which mean they lay on the negative side of the decision boundary. That is why the model correctly labelled it as negative. In the second example, the tweet has the words “killing”, “dying”, “police”, “protester” and “occupygezi” which are related to violence get positive weights and lay on the positive side on the decision boundary. That is why the model gave a positive label. In Table 4.22, the first tweet has the words “dying” and “occupygezi” which is related to the violence and has positive weights. It also has the words “press”, “working”, “people”, “help”, “prayforturkey” which are not related to violence and have negative weights. That is why the model decides to give a negative label while the tweet is positive (reports violence).

Now, to investigate the model’s prediction certainty, we show the probability distribution of the model’s predictions on the different test sets in Figures 4.24, 4.25 and 4.26. The figures show that the model is most certainly in the model predictions as the majority of the negative and positive predictions associated with high probabilities 0.9. It is even doing better than the protest prediction. Like before we investigate an example of each dataset that received close prediction probabilities by the SVM model. Table 4.23 shows these examples.

In the first example in Table 4.23, the words “video”, “police”, “journalists”,“gezi ” and

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
read comprehensive news	0	0	0.98	0.012
people killing people dying children hurt hear crying practice preach orlando supports occupygezi youranonnews must watch turkish protester hit police panzer	1	1	0	1

Table 4.21: Examples of correctly classified labelled tweets by the linear SVM model with ($C = 10$) and TF-IDF representation

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
turkish press working people dying streets help occupygezi prayforturkey	1	0	0.98	0.019

Table 4.22: Examples of mis-classified labelled tweets by the linear SVM model with ($C = 10$) and TF-IDF representation

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
video teargassed lobby police also targeting journalists street medics gezi	1	1	0.5	0.5
#ankara does not have a single demo point such as taksim after police started to yield at kizilay sq we m demos disper	0	1	0.57	0.42

Table 4.23: Examples of labelled tweets that recieved close prediction probabilities by the linear SVM model with ($C = 10$) and TF-IDF representation

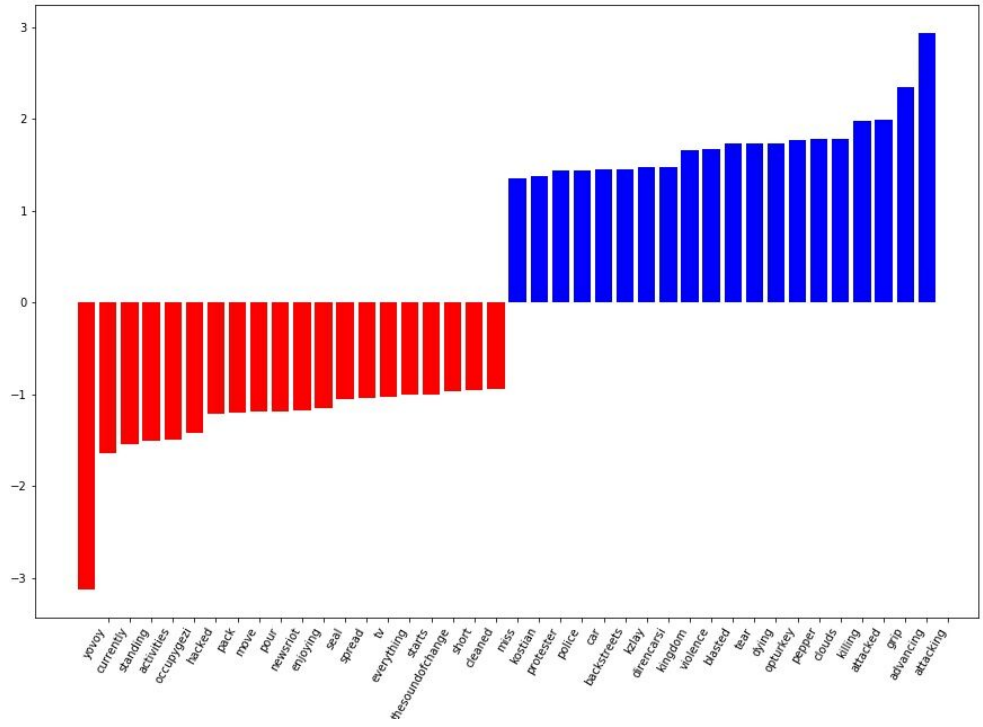

Figure 4.27: The most influential words on the SVM model with ($C = 10$) and TF-IDF representation. Blue bars are words with positive influence and red bars are words with negative influence on the model

"medics" are in the top 1000 words that have a positive influence on the model meaning that the model associates them with positive labels. Each of the words has a positive weight. On the other hand, the words "lobby", "also", "street" and "target" have a negative influence on the model. The tweet contains 5 words that give positive influence and 4 words that give a negative influence. The total sum of the weights of the positive words equals the weights of the negative words. That is why the model gives 0.5 positive probabilities and 0.5 negative probabilities to the positive and negative labels. Similarly, the second example in the table contains the words "ankara", "police", "started" and "demos" that are the positive words. And the words "point", "kizilay" and "sq" that are negative words. Although the positive words are more than the negative words, the weights of the negative words are higher than the positive words and that is why the model gives a slightly higher probability to the negative label. On the third example, the words "international" and "turkey" are negative words and the words "news", "showing", "riots" and "direngeziparki" are positive words. The number and the weights of the positive words are higher than the number and the weights of negative words that is why the model gives a slightly higher probability to the positive label. This shows how the model makes it's predictions and what is the influence of certain words on the model. Figure 4.27 shows the most influential 20 words on the model.

Although there are incidents of misclassifications and close prediction probabilities, the model still performs well on the unseen data and the percentage of the close prediction probabilities are small. The AUC scores on the different datasets range from 0.71 to 0.81, which means that the True Positive Rate (TPR) is high and False Positive Rate (FPR) is low. This means the model's prediction of true violence tweets is higher than the false positive tweets. In the next section, we test the performance of the MNB model on the violence dataset and compare between the two models.

Multinomial Naive Bayes (MNB)

Now, we start investigating the best parameters of Multinomial Naive Bayes model that fits the violent dataset. We run a grid search with the following values:
$\alpha = \{0.01, 0.1, 1, 2\}$
Class-prior = {Uniform prior, Class size prior}
Uniform prior = {0.5,0.5}
Class size prior = {positive = 0.06, negative=0.93}
Text representation = {BOW, TF-IDF}

We split the dataset into two random equal-sized sets: the training set and the test set. We run 10 folds cross-validation on the training dataset and report the AUC score on the validation set. Then, we run the different combinations on the test set. We repeat this process 10 times and report the average AUC score. Table 4.24 shows the best AUC scores of parameters on the test set with each text representation. The table shows that the best parameters to use for MNB are ($\alpha = 0.1$ and class prior = uniform prior) with BOW and TF-IDF. The results show the higher performance of the MNB with TF-IDF than MNB with BOW. Next, a statistical test is done to compare the two performances. We randomly split the dataset to equally sized train and test sets and measure the model's performance in terms of AUC scores on the different representations. We repeat this 30 times. Then we run the Wilcoxon statistical test on the sample of 30 AUC scores for both BOW and TF-IDF. The results show that there is a significant difference between BOW and TF-IDF. That is why we are going to use TF-IDF with MNB ($\alpha = 0.1$ and Class prior = uniform prior).

Now, we test the model's performance for MNB ($\alpha = 0.1$ and class prior = uniform prior) by training it on the TF-IDF representation of the data, which is 2487 violence tweets with 158 (6.3%) positive tweets and 2329 (93.6%) negative tweets. The model is evaluated by testing it on the Test set, GT1, GT2 and GT3. The model's performance is shown in Table 4.25.

The number of right predictions that the model makes compared to the number of wrong

Text representation	Class prior	$\alpha = 0.01$	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 2$
BOW	Uniform prior	0.8167	**0.8167**	0.7906	0.6865
	Class size prior	0.8094	0.8150	0.7922	0.7507
TF-IDF	Uniform prior	0.8189	**0.8189**	0.6893	0.6142
	Class size prior	.7803	0.5835	0.5324	0.5113

Table 4.24: The MNB model is performance in terms of AUC score with the different parameter of parametes and text representations of the violent dataset

	Positive samples	Negative samples	AUC
Test set (2488)	164 (6.9%)	2324 (93.4%)	0.8189
GT1 (116)	11 (9.4%)	105 (90.5%)	0.78
GT2 (1213)	235 (19.37%)	978 (80.6%)	0.80
GT3 (1213)	158 (13.02%)	1055 (86.97%)	0.82

Table 4.25: MNB model is performance (AUC scores) with ($\alpha = 0.1$ and uniform prior) on TF-IDF representation on each test set

	No.right prediction	No.wrong prediction
Test set (2488)	2233 (90%)	255 (10%)
GT1 (116)	104 (90%)	12 (10%)
GT2 (1213)	1045 (86%)	168 (14%)
GT3 (1213)	1030 (85%)	183 (15%)

Table 4.26: The number of right and wrong predictions of the MNB model is performance (AUC scores) with ($\alpha = 0.1$ and uniform prior) on TF-IDF representation on each test set

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
arresting people tweeting guess talk	0	0	0.992	0.007
update police fire tear gas istanbul taksim square protestors turkey	1	1	0	1

Table 4.27: Examples of correctly classified labelled tweets by MNB model is performance (AUC scores) with ($\alpha = 0.1$ and uniform prior) on TF-IDF representation

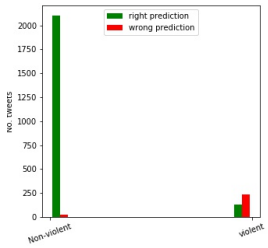

Figure 4.28: The No. of right and wrong predictions of each label (Non-violence and violence) using MNB on violence Test set

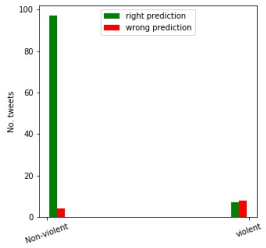

Figure 4.29: The No. of right and wrong predictions of each label (Non-violence and violence) using MNB on violence GT1

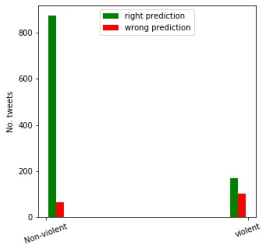

Figure 4.30: The No. of right and wrong predictions of each label (Non-violence and violence) using MNB on violence GT2

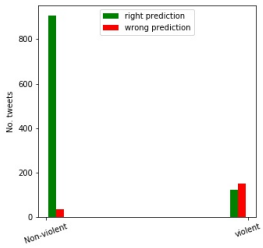

Figure 4.31: The No. of right and wrong predictions of each label (Non-violence and violence) using MNB on violence GT3

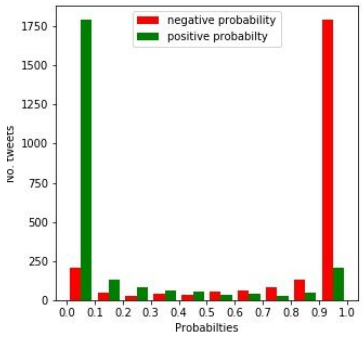

Figure 4.32: MNB probability distribution on Test set

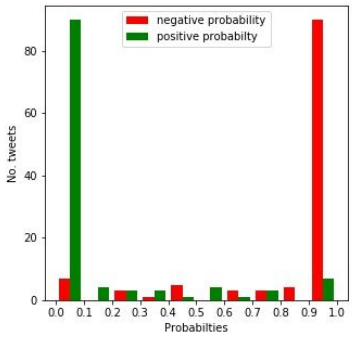

Figure 4.33: MNB probability distribution on GT1

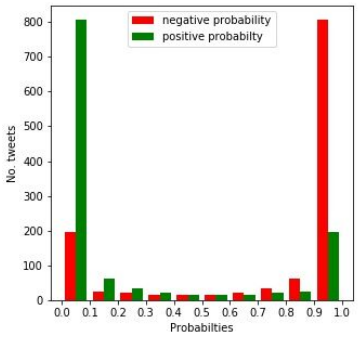

Figure 4.34: MNB probability distribution on GT2

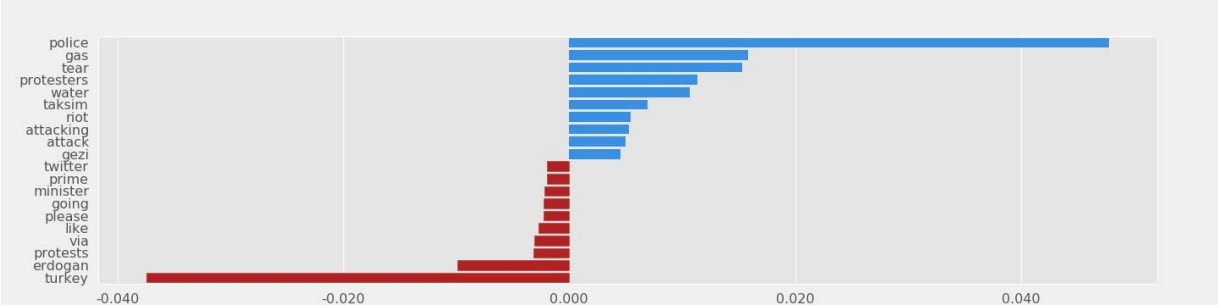

Figure 4.35: The most influential words on the MNB model with ($\alpha = 0.1$, Uniform prior) and TF-IDF representation. Blue bars are words with positive influence and red bars are words with negative influence on the model

predictions is shown in Table 4.26. The MNB model has a different pattern than the SVM model on the violence datasets. The SVM model tends to predict higher FNR while the MNB tends to predict higher FPR as shown in Figures 4.28, 4.29, 4.30 and 4.31. This could be, as mentioned in section 4.4.1, because the number of positive samples in the datasets is low. The

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
turkish police break occupy taksim park protest istanbul tandem	0	1	0.004	0.995
whole city cloud pepper many cities turkey joined	1	10	0.82	0.17

Table 4.28: Examples of mis-classified labelled tweets by the MNB model is performance (AUC scores) with ($\alpha = 0.1$ and uniform prior) on TF-IDF representation

Tweet	Actual Label	Predicted Label	Negative Prob.	Positive Prob.
ustream li ve park istanbul	0	0	0.55	0.44

Table 4.29: Examples of labelled tweets that received close prediction probabilities by the MNB model is performance (AUC scores) with ($\alpha = 0.1$ and uniform prior) on TF-IDF representation

positive likelihood estimate is high and the probability of the positive label is high. Tables 4.27 and 4.28 show examples of correctly classified and misclassified violence tweets by the MNB model. To investigate the model's performance, we show the probability distribution of the model's predictions on the different test sets in Figures 4.32, 4.33 and 4.34. The figures show a small number of tweets that received probabilities close to the negative and positive label in each dataset. The model shows a slightly higher number of tweets with close prediction probabilities than SVM. Although there are misclassifications and close prediction probabilities, the model still performs well on the unseen data and the percentage of the close prediction probabilities are small. The AUC scores on the different datasets are high, which means that TPR is high and FPR is low. This means the model is the prediction of true violence tweets is higher than the false positive tweets.

Model Selection

Now, we compare the performance of the linearSVM ($C = 10$) with the TF-IDF representation of the violence dataset against the performance of MNB ($\alpha = 0.1$ and class prior = uniform prior) with the TF-IDF representation of the violence dataset. We run the McNemar significance test with the null hypothesis that there is no difference between the MNB and SVM on the violence datasets. Although Table 4.30 shows that MNB outperforms SVM on all the datasets, the p-value of the McNemar significant test shows that the two models show the same proportion of errors. This means we accept the null hypothesis that there is no significant difference between MNB and SVM on the violence dataset. The higher AUC scores of the MNB could be because the number of positive examples in the dataset is low. After comparing the performance of the

	Positive samples	Negative samples	MNB	SVM	P-value
Test set (1833)	756 (41.2%)	1077 (58.7%)	**0.8189**	0.8127	0.1668
GT1 (116)	31 (26.72%)	85 (73.2%)	**0.78**	0.7129	0.50
GT2 (1213)	515 (42.456%)	698 (57.54%)	**0.80**	0.7439	0.06
GT3 (1213)	590 (48.63%)	623 (51.3%)	**0.82**	0.7260	0.28

Table 4.30: Comparison between MNB model is performance (AUC scores) with ($\alpha = 0.1$ and class prior = uniform-prior) and TF-IDF representation versus Linear SVM model is with($C = 10$) and TF-IDF on each test set

two models and because MNB outperforms SVM, we are going to use MNB ($\alpha = 0.1$ and class prior = uniform prior) with TF-IDF to predict the violence labels of the rest of the unlabelled tweets in the original tweets collection described in chapter 3.

4.4 Results Analysis

4.4.1 Tweets Predictions

Now, we apply the best performing models to the rest of the tweets collection to predict their protest and violence labels. The tweets collection contains 1,290,451 unlabelled tweets. For protest classification, we use linear SVM ($C = 1$) with the TF-IDF representation and for violence classification; we use MNB ($\alpha = 0.1$ and class prior = uniform prior) and TF-IDF. At first, we apply the same pre-processing step mentioned before in section 4.2. This resulted in 854,313 tweets. Then, we run the protest prediction and violence prediction models. Protest prediction resulted in 308,645 tweets labelled as positive protest (related to the protest), which is 36% and 545,668, are labelled as a negative protest (not related to the protest), which is 64%. The violence prediction resulted in 128,831 (15%) violent tweets and 724,337 (85%) non-violent tweets. Within the protest-positive tweets, the number of violent tweets is 100,387 (33%) and 207,613 (67%) non-violent tweets. Within the negative protest tweets, the percentage of violent tweets is even smaller 5% and 95% non-violent tweets.

4.4.2 Tweets Timeline

When we grouped the predicted tweets by the date, we got 27 groups for 27 days. As in Figures 4.36 and 4.37, we found that the number of tweets peaked on key dates. The key dates are the dates when force was used intensely by the police during the protest as shown in Table 4.31. We filtered out the protest positive tweets and violence positive tweets. This resulted in 100,387 tweets. Then, we extracted the most frequent 27 words in the tweets. The resulted words are “ga”, “tear”, “attack”, “water”, “riot”, “clash”, “cannon”, “violenc”, “fire”, “peac”, ”, “forc”, “brutal”, “report”, “injur”, “street”, “demonstr”, “right”, “kill”, “bomb”, “tearga”, “stop”, “violent”, “continu”, “hotel”, “chemic”, “arrest”. The words are stemmed (suffixes from

Date	Event
31/05/2013	The beginning of the protest and the use of force by police including tear gas and water cannons against protesters.
11/06/2013	Police forces make an attempt to clear Gezi square by force.
15/06/2013	The square is successfully cleared from protesters.

Table 4.31: Violent days during the Gezi protest with the violent events.

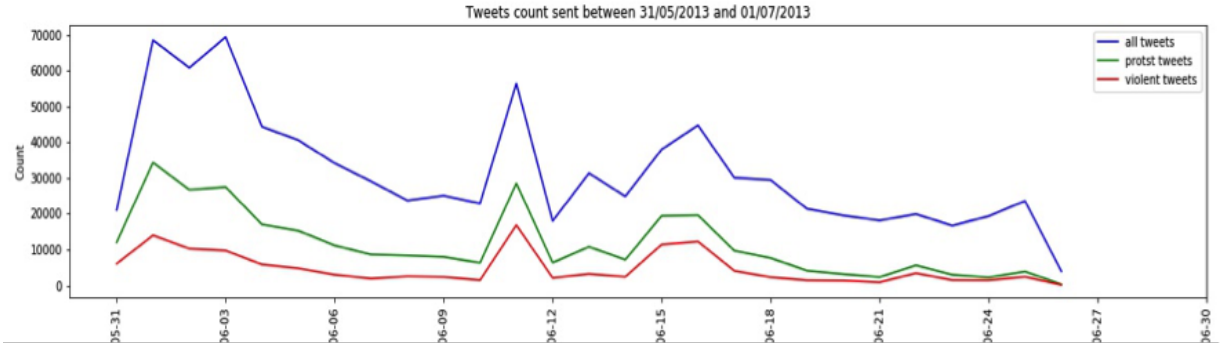

Figure 4.36: Timeline of the number of tweets, protest tweets and violent tweets sent during the Gezi protest perios from 31/05/2013 to 27/06/2013

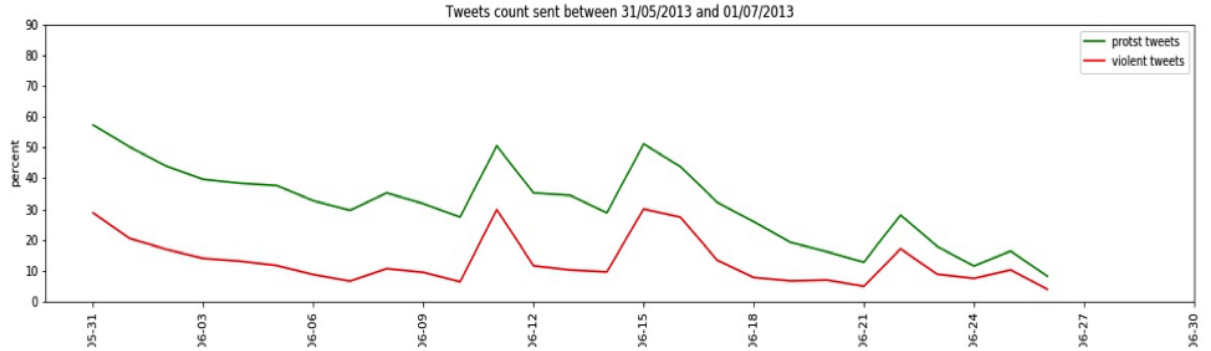

Figure 4.37: Timeline of the percentage of protest tweets and violent tweets sent during the Gezi protest perios from 31/05/2013 to 27/06/2013

words like es, s, ed , ing) that is why words like “ga” means “gas”. The words show the type of violence that was used during the protest. Then, we showed the number of occurrences of these words in each day of the 27 days in Figure 4.38. The figure shows that the number of occurrences of the words is the highest on the most violent days of the protest (the same days of the spikes in Figure 4.36 and 4.37).

Conclusion

In this chapter, we ran a group of experiments to answer the second research question: What is the best baseline machine-learning model between SVM and MNB to classify tweets as protest/non-protest tweets and violent/non-violent tweets? We tried two different baseline models SVM and MNB with different parameters and different text representations: BOW, TF-IDF and word embeddings (WE) to find the best model, the best parameter and the best text representation model to fit both the protest and the violence datasets. We found that the best model for protest classification is linear SVM with margin C = 1 with the TF-IDF text representation. And for violence classification, the best model is MNB with an alpha score (α=0.1) and uniform prior with the TF-IDF text representation. We then used these models to predict the labels of the rest of the tweets collection of 1,290,451 tweets. The date clusters of these tweets show that violent tweets peaked on the days when the Turkish police used force against the protesters. The same pattern showed when we used the number of word occurrences per day. We found that the number of occurrences of violent words increased on the violent days.

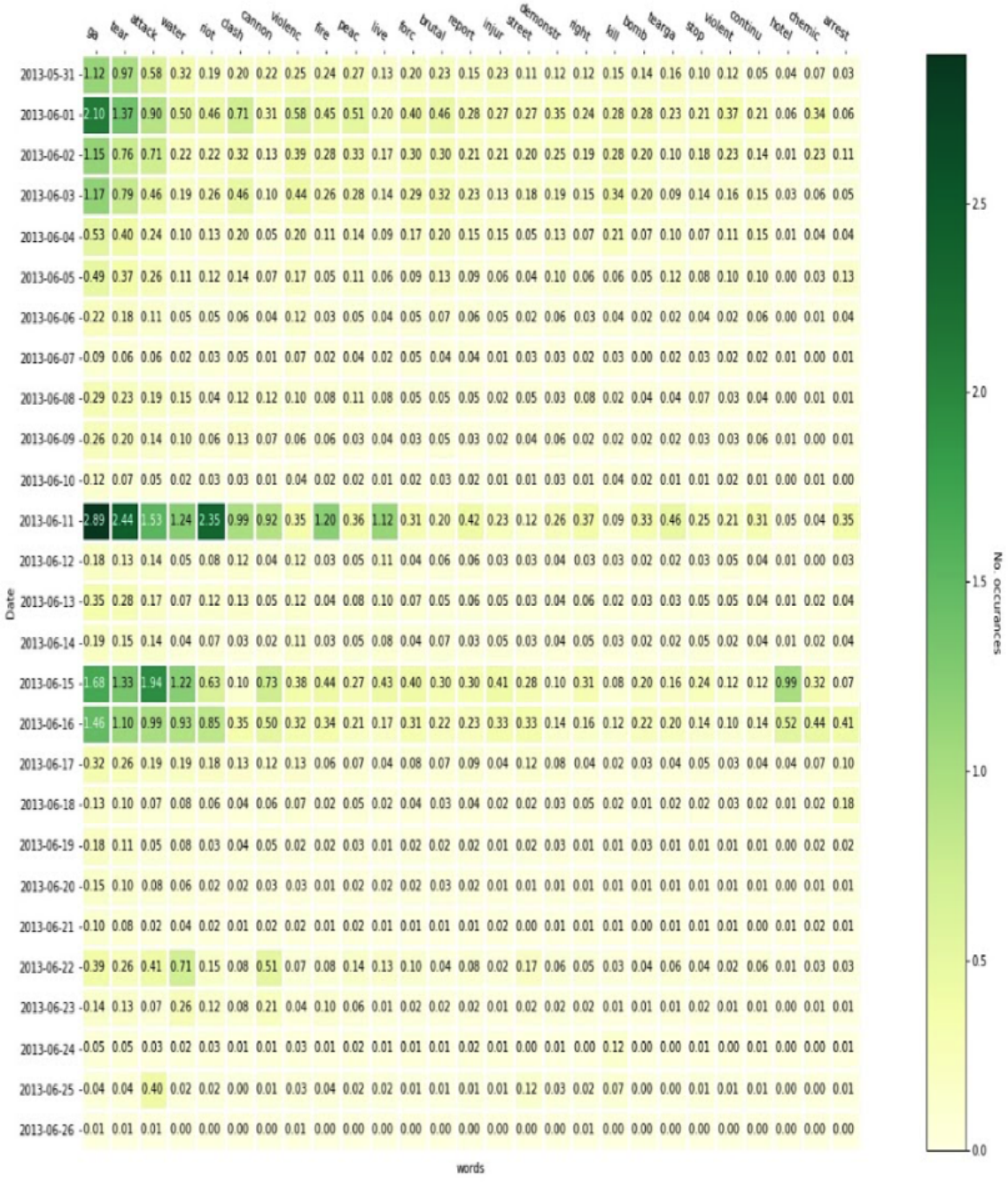

Figure 4.38: The number of occurrences of the most frequent words during the protest days from 31/05/2013 to 27/06/2013

Chapter 5

Discussion and Conclusion

In this research, we investigated using Twitter as a source of information to detect tweets that report incidents of protest repression. We used a machine-learning model to automatically detect violence reporting tweets. We used a collection of tweets that were sent during the Turkish Gezi Park protest in 2013. A subset of this tweets collection with their correspondent labels as protest-related or not and violent or not (training dataset), is required to train the machine learning model. To build this training dataset, we created a crowdsourcing experiment and asked the workers to label the tweets. After that, we used different baseline machine learning models with different text representation models with the crowdsourced dataset and chose the best text representation and the best baseline model that fit the data. We investigate the use of machine learning models with Twitter to detect protest and protest repression incidents.

Twitter provides a real-time source of information that reports the incidents as soon as they happen or soon after compared to news reports. Twitter also gives a chance to everyone with Internet access to report what he or she sees regardless where he or she is. This overcomes one of the main issues with news reports, coverage bias, as news articles tend to report what is happening in big important cities, not small towns or villages. We set out this research to investigate the possibility of using machine learning models with social media posts (tweets) to detect protest and protest repression events.

To build that machine learning model, two more research questions need to be answered through experiments: 1) What is the agreement level internally between the crowd workers on our data? 2) What is the best baseline machine-learning model between SVM and MNB to classify each tweet in the dataset as protest/non-protest tweet or violent/non-violent tweet? In this chapter, we conclude this research by providing answers to the research questions from the experiments'results. Then, we follow up with a discussion on the empirical findings, how they fit with the existing body of literature. Finally, we conclude with the research's challenges, recommendations and directions for future work.

5.1 Discussion

5.1.1 Empirical Findings

We start with providing a summary of the experiments done in this research and a discussion on the empirical results of these experiments.

Crowdsourcing Experiment

This experiment tries to answer the following research question: **w**hat is the agreement level internally between the crowd workers on the dataset of tweets collections? The experiment was designed to ask the workers two questions for each tweet. The first question is protest-related, "Is the tweet related to the Gezi park protest or not? " and the second question is violence-related "Does the tweet report violence or not?". We used a collection of 6693 tweets to be labelled besides 116 tweets were used as test questions to detect the workers 'performance.

The total number of workers is 1554 workers who submitted 20635 judgments. 54% of the workers proved to be trusted by achieving performance score above 75% on the test questions and those workers are the ones whose submissions were used. The experiment results answer the first research question: What is the agreement level internally between the crowd workers on our data? Using Kripperndorff-alpha score to detect the inter-annotator agreement, we found that the agreement score between the workers for the protest question is 0.428 and for the violence question is 0.427. According to [78], the score is fair if it is above 0.6. This shows that the agreement between crowd workers on the labels is not fair. This is because of some of the tweets that are not directly related to the protest and still contain keywords related to the protest. Similarly, with the violence questions, People might misinterpret some tweets as violent while others do not. To overcome this problem, use only the tweets that received full agreement between all the workers.

This leaves us with a dataset of 3860 tweets with fully agreed protest labels and a dataset of 5248 tweets with fully agreed violence labels. The current literature on crowdsourcing experiments for machine-learning task suggests that quality of the results can be controlled through a well-designed task, clear data and trusted worker. our experiment supports that suggestion and stresses on the importance of the quality settings related to the workers. One of the important settings that affected our experiment is the number of judgment per row of data (tweet). We recommend having at least 5 judgments per row of data, even though it costs more money, as it affects the agreement level between the crowd workers and in turn the quality of the resulted label. The second import setting is the worker's trust. It is important to grant access only to workers who proved not to be spammers and to understand the task. It is a trade-off as the higher the trust score, the less the number of workers who can do the task which will make the task slower to finish. This takes us to the next important setting which is the Golden units (test

questions). Based on these questions the worker's trust score is measured. That is why it is important to provide many test questions and more importantly to get more than one expert to label these golden units and aggregate the most agreed label to avoid confusing the workers.

Text Classification Experiment

After building the training dataset using crowdsourcing, in this experiment, we trained different machine learning models with different settings to find the best model that fits our data. we ran two experiments: protest classification to detect protest-related tweets and violence classification to detect violence reporting tweets. For each group, we trained two machine-learning models: Multinomial Naive Bayes (MNB) and Support Vector Machine (SVM). We also used different text representation models Bag Of Words (BOW), TF-IDF and word embeddings. We then chose the best text representation model and the machine-learning model that best fits the data.

For the protest classification experiment, we train the models on the protest dataset, which has tweets with full agreement protest labels. This dataset contains 3860 tweets with 61% (2342) negative tweets (not related to the Gezi park protest) and 39% (1518) positive tweets (related to the Gezi park protest). Similarly, for the violence classification experiment, we train the models on the violence dataset, which has tweets with full agreement violence labels. The violence dataset is imbalanced with only 6% (323) positive tweets (report violence during the protest) and 95% (4925) negative tweets (do not report violence incidents during the protest).

After training the models, we tested them on four datasets: Test set contains 50% randomly selected tweets from each dataset (protest dataset and violence dataset); GT1 contains the 116 tweets labelled by us and used as test questions in the crowdsourcing experiment; GT2 that contains 1214 tweets which were labelled by crowd workers in a pilot crowdsourcing experiment and GT3 which is the same as GT2 but labelled by us. This experiment tried to answer the second research question: What is the best baseline machine-learning model between SVM and MNB to classify tweets as protest/non-protest tweets or violent/non-violent tweets? The results showed that for protest classification, linear SVM with a margin value C =1 and TF-IDF text representation was the best that fit the protest dataset. The SVM achieved AUC scores of 0.899 on the Test set, 0.840 on GT1, 0.817 on GT2 and 0.828 on GT3. On the other hand, MNB with uniform prior and an alpha score of 0.1 and TF-IDF text representation gave the best AUC scores on the violence dataset. The MNB model gave AUC score of 0.8189 on the Test set, 0.78 on GT1, 0.80 on GT2 and 0.82 on GT3.

These two models, with the best AUC score, were used to predict the protest and violence labels in the rest of the tweets collection of 1,290,451 tweets. The recent body of literature on machine

learning suggests that deep learning models outperform baseline machine learning models especially with data like tweets [7] [32] [39] [43] [91] [93]. This experiment shows that, for our data, baseline machine learning models like SVM and MNB perform well in text classification tasks with tweets but with the faster training process and fewer parameters to tune. It also shows that basic text representation models, like BOW and TF-IDF, outperform word embeddings. The analysis of the predicted tweets shows that the number of violence tweets peaked on the violent days of the protest when the Turkish police attacked Gezi Park to empty it from the protesters. Also the use of violence-related words like "tear", "gas", "attack" and "arrest" increased on the violent days.

5.2 Conclusion

This research is concerned with developing a tool to automatically detect protests and protest repression incidents. As the literature suggests, protesting groups face protest repression at least once. This is why protest repression needs to be studied, to implement methods and to assign resources that can help in reducing the threat of people being exposed to that kind of violence.

Our research showed that there is a potential in Twitter to be used as a real-time source of information to detect protest repression incidents and to measure protest repression. However, there are challenges come with people posting tweets about the same incident more than one time and in new tweets not only as re-tweets. Also, people not necessarily witness the incident themselves, they post about incidents they heard of or read about in other social media sources like Facebook or Blogs and this could be a source of accuracy bias. There are also challenges related to the nature of tweets, which are human-generated with grammatical and spelling mistakes. One of the main challenges we faced in this research, is the subjective nature of violence as it was confusing sometimes if the tweet reports violence or not. This confusion showed in the crowdsourcing task and in turn in the results of the classification model. Another challenge is the lack of ground truth data on protest repression to validate the results of automatic protest repression detection.

The contribution of this research lies in investigating the possibility of using machine learning models with social media posts (tweets) to detect protest and protest repression events. This investigation showed that:

1. Non-expert (not political scientists) crowd workers, to some extent, can agree on what is protest-related or not and what is protest repression or not. And with implementing the right quality measures, we can get high levels of agreement from trusted crowd workers.

2. the crowdsourced labelled dataset can be used to build a good performing machine learning model in terms of high AUC scores.

3. Comparing the model's results to protest repression incidents happened on the ground, showed potential in Twitter to be a good source of information. However, more investigation must be done here.

5.2.1 Recommendations and Future work

This dissertation can be improved in future work through enhancing the dataset that was used to train and test the model and through improving the model itself.

The Data

Data availability is the most challenging part of most of the studies. This challenge was manifested in our study in two ways. Firstly, although some people from different parts of the world tweet in English during protests to attract the attention of the international community, we still miss out a lot of tweets written in the native language of of the country of the protest. This issue can be addressed in the future work by collecting tweets from different countries in different languages and use a translator to translate the tweets into English or to use Multi-lingual Transfer Learning models [85].

Another issue with our dataset is that the tweets were collected from only one protest, the Turkish Gezi Park protest, which limited the scope of our study and made the machine model not generalizable to other protests or other violence cases. This can be fixed in future work by collecting tweets from different protests in different parts of the world. Another way to extend the scope of the study could be through training the model to detect violence in general and then categories the type of violence, for example, protest repression or electoral violence.

The second challenge we faced was the availability of reliable ground truth to test the model. This issue can be addressed in future work by scraping news articles related to protests where repression happened. Political conflict datasets like GDELT could be used. However, they do not provide the actual text of the incidents. Instead, they provide the links to the articles and sometimes the links are broken and in other times, the articles are either not related to the protest at all or do not report a violent incident but provide a summary to what happened or a brief history of the reasons behind the protest.

The model

The future work to improve the model could be through using the images that are sent with tweet sometimes. [92] demonstrated that adding image classification step to the detection pipeline

improved identifying collective action events from tweets. Also, deep learning models like CNN [71] or LSTM [65] can be experimented with to test its effect on improving the performance in future work. Finally, a wider investigation of how useful social media is, in building protest repression datasets as a source of information, needs to be conducted after building more reliable training dataset and ground truth datasets.

Bibliography

[1] Chapter 2 - The Seven Practice Areas of Text Analytics. In Nisbet and G. Miner, editors, *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*, pages 29–41. Academic Press, Boston, 2012.

[2] Chapter 4 - Applications and Use Cases for Text Mining. In G. M. D. E. F. H. A. Nisbet, editor, *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*, pages 53–72. Academic Press, Boston, 2012.

[3] C. C. Aggarwal. Text Preparation and Similarity Computation. In *Machine Learning for Text*, pages 17–30. Springer, 2018.

[4] O. Alonso, C. Marshall, and M. Najork. Crowdsourcing a subjective labeling task: a human-centered framework to ensure reliable results. *Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2014âĂŞ91*, 2014.

[5] O. Alonso, C. C. Marshall, and M. Najork. Are some tweets more interesting than others?# hardquestion. page 2. ACM, 2013.

[6] O. Alonso, C. C. Marshall, and M. Najork. Debugging a crowdsourced task with low inter-rater agreement. pages 101–110. ACM, 2015.

[7] R. ALRashdi and S. O'Keefe. Deep Learning and Word Embeddings for Tweet Classification for Crisis Response. *arXiv preprint arXiv:1903.11024*, 2019.

[8] A. Anisin. Repression, spontaneity, and collective action: the 2013 Turkish Gezi protests. *Journal of Civil Society*, 12(4):411–429, 2016.

[9] S. E. AytaÃğ, L. Schiumerini, and S. Stokes. Protests and Repression in New Democracies. *Perspectives on Politics*, 15(1):62–82, 2017.

[10] G. Barata, K. Shores, and J. P. Alperin. Local chatter or international buzz? Language differences on posts about Zika research on Twitter and Facebook. *PloS one*, 13(1):e0190482, 2018.

[11] S. E. Barkan. Legal control of the southern civil rights movement. *American Sociological Review*, pages 552–565, 1984.

[12] J. Beieler, P. T. Brandt, A. Halterman, P. A. Schrodt, and E. M. Simpson. Generating political event data in near real time. *Computational Social Science*, page 98, 2016.

[13] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[14] D. Bishara. The politics of ignoring: Protest dynamics in late Mubarak Egypt. *Perspectives on Politics*, 13(4):958–975, 2015.

[15] A. Bruns, T. Highfield, and J. Burgess. The Arab Spring and social media audiences: English and Arabic Twitter users and their networks. *American Behavioral Scientist*, 57(7):871–898, 2013.

[16] C. Callison-Burch and M. Dredze. Creating speech and language data with Amazon's Mechanical Turk. pages 1–12. Association for Computational Linguistics, 2010.

[17] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. pages 759–768. ACM, 2010.

[18] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[19] C. Davenport and P. Ball. Views to a kill: Exploring the implications of source selection in the case of Guatemalan state terror, 1977-1995. *Journal of conflict resolution*, 46(3):427–450, 2002.

[20] J. De Winter, M. Kyriakidis, D. Dodou, and R. Happee. Using CrowdFlower to study the relationship between self-reported violations and traffic accidents. *Procedia Manufacturing*, 3:2518–2525, 2015.

[21] D. Della Porta. Social movements and the state: Thoughts on the policing of protest. 1995.

[22] J. DemÅąar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

[23] T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998.

[24] J. Earl. Tanks, tear gas, and taxes: Toward a theory of movement repression. *Sociological theory*, 21(1):44–68, 2003.

[25] J. Earl, H. McKee Hurwitz, A. Mejia Mesinas, M. Tolan, and A. Arlotti. This protest will be tweeted: Twitter and protest policing during the Pittsburgh G20. *Information, Communication & Society*, 16(4):459–478, 2013.

[26] J. Earl, S. A. Soule, and J. D. McCarthy. Protest under fire? Explaining the policing of protest. *American sociological review*, pages 581–606, 2003.

[27] T. Finin, W. Murnane, A. Karandikar, N. Keller, J. Martineau, and M. Dredze. Annotating named entities in Twitter data with crowdsourcing. pages 80–88. Association for Computational Linguistics, 2010.

[28] G. A. Fink. n-Gram Models. In G. A. Fink, editor, *Markov Models for Pattern Recognition: From Theory to Applications*, pages 107–127. Springer London, London, 2014.

[29] K. Fort, G. Adda, and K. B. Cohen. Amazon mechanical turk: Gold mine or coal mine? *Computational Linguistics*, 37(2):413–420, 2011.

[30] U. Gadiraju, J. Yang, and A. Bozzon. Clarity is a worthwhile quality: On the role of task clarity in microtask crowdsourcing. pages 5–14. ACM, 2017.

[31] J. Gellerman. Repression and Protests: A Comparative Case Study on the Causes of Protest Violence. 2018.

[32] Q. Gong, Y. Chen, X. He, Z. Zhuang, T. Wang, H. Huang, X. Wang, and X. Fu. DeepScan: Exploiting deep learning for malicious account detection in location-based social networks. *IEEE Communications Magazine*, 56(11):21–27, 2018.

[33] L. Grossman. Iran protests: Twitter, the medium of the movement. *Time Magazine*, 17, 2009.

[34] I. Guyon, B. Boser, and V. Vapnik. Automatic capacity tuning of very large VC-dimension classifiers. pages 147–155, 1993.

[35] G. Haciyakupoglu and W. Zhang. Social media and trust during the Gezi protests in Turkey. *Journal of ComputerâĂŘMediated Communication*, 20(4):450–466, 2015.

[36] M. Heilman and N. A. Smith. Rating computer-generated questions with Mechanical Turk. pages 35–40. Association for Computational Linguistics, 2010.

[37] J. Hong and C. F. Baker. How good is the crowd at real WSD? pages 30–37. Association for Computational Linguistics, 2011.

[38] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A practical guide to support vector classification. 2003.

[39] H. Hu, N. Phan, J. Geller, H. Vo, B. Manasi, X. Huang, S. Di Lorio, T. Dinh, and S. A. Chun. Deep Self-Taught Learning for Detecting Drug Abuse Risk Behavior in Tweets. pages 330–342. Springer, 2018.

[40] P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM, 2010.

[41] T. Joachims. Text Classification. In *Learning to Classify Text Using Support Vector Machines*, volume 668 of *The Springer International Series in Engineering and Computer Science*, pages 7–33. Springer US, Jan. 2002.

[42] K. Krippendorff. Computing Krippendorff's alpha-reliability. 2011.

[43] S. Kudugunta and E. Ferrara. Deep neural networks for bot detection. *Information Sciences*, 467:312–322, 2018.

[44] F. Laws, C. Scheible, and H. SchÃijtze. Active learning with amazon mechanical turk. pages 1546–1556. Association for Computational Linguistics, 2011.

[45] M. Lease. On quality control and machine learning in crowdsourcing. *Human Computation*, 11(11), 2011.

[46] M. Loveman. High-risk collective action: Defending human rights in Chile, Uruguay, and Argentina. *American Journal of Sociology*, 104(2):477–525, 1998.

[47] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[48] T. Mathiyazhagan and D. Nandan. Survey research method. *Media Mimansa*, 4(1):34–45, 2010.

[49] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. volume 752, pages 41–48. Citeseer, 1998.

[50] J. D. McCarthy, D. W. Britt, and M. Wolfson. The institutional channeling of social movements by the state in the United States. *Research in Social Movements, Conflicts and Change*, 13(2), 1991.

[51] B. Mellebeek, F. Benavent, J. Grivolla, J. Codina, M. R. Costa-Jussa, and R. Banchs. Opinion mining of spanish customer comments with non-expert annotations on mechanical turk. pages 114–121. Association for Computational Linguistics, 2010.

[52] V. Metsis, I. Androutsopoulos, and G. Paliouras. Spam filtering with naive bayes-which naive bayes? volume 17, pages 28–69. Mountain View, CA, 2006.

[53] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[54] D. Oleson, A. Sorokin, G. P. Laughlin, V. Hester, J. Le, and L. Biewald. Programmatic Gold: Targeted and Scalable Quality Assurance in Crowdsourcing. *Human computation*, 11(11), 2011.

[55] I. Ortiz, S. L. Burke, M. Berrada, and H. CortÃl's. World Protests 2006-2013. 2013.

[56] S. M. Paradesi. Geotagging Tweets Using Their Content. 2011.

[57] T. Poell and E. Borra. Twitter, YouTube, and Flickr as platforms of alternative journalism: The social media account of the 2010 Toronto G20 protests. *Journalism*, 13(6):695–713, 2012.

[58] M. Rajman and M. Vesely. From text to knowledge: Document processing and visualization: A text mining approach. In *Text Mining and its Applications*, pages 7–24. Springer, 2004.

[59] S. Raschka. Naive bayes and text classification i-introduction and theory. *arXiv preprint arXiv:1410.5329*, 2014.

[60] S. Raschka. Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. 2018.

[61] A. Ritter, O. Etzioni, and S. Clark. Open domain event extraction from twitter. pages 1104–1112. ACM, 2012.

[62] S. Rogers and M. Girolami. *A first course in machine learning*. CRC Press, 2016.

[63] S. Rosenthal, W. Lipovsky, K. McKeown, K. Thadani, and J. Andreas. Towards Semi-Automated Annotation for Prepositional Phrase Attachment. 2010.

[64] M. Sabou, K. Bontcheva, L. Derczynski, and A. Scharl. Corpus Annotation through Crowdsourcing: Towards Best Practice Guidelines. pages 859–866, 2014.

[65] H. Sak, A. Senior, and F. Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*, 2014.

[66] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes Twitter users: real-time event detection by social sensors. pages 851–860. ACM, 2010.

[67] I. Salehyan, C. S. Hendrix, J. Hamner, C. Case, C. Linebarger, E. Stull, and J. Williams. Social conflict in Africa: A new database. *International Interactions*, 38(4):503–511, 2012.

[68] D. Sarkar. Text Classification. In D. Sarkar, editor, *Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data*, pages 167–215. Apress, Berkeley, CA, 2016.

[69] A. B. Sayeed, B. Rusk, M. Petrov, H. C. Nguyen, T. J. Meyer, and A. Weinberg. Crowdsourcing syntactic relatedness judgements for opinion mining in the study of information technology adoption. pages 69–77. Association for Computational Linguistics, 2011.

[70] scikit learn. Receiver Operating Characteristic (ROC). `https://sklearn.org/auto_examples/model_selection/plot_roc.html#sphx-glr-auto-examples-model-selection-plot-roc-py`, 2007.

[71] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

[72] L. Sloan and J. Morgan. Who tweets with their location? Understanding the relationship between demographic characteristics and the use of geoservices and geotagging on Twitter. *PloS one*, 10(11):e0142209, 2015.

[73] B. G. Smith, R. L. Men, and R. Al-Sinan. Tweeting Taksim communication power and social media advocacy in the Taksim square protests. *Computers in Human Behavior*, 50:499–507, 2015.

[74] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. pages 254–263. Association for Computational Linguistics, 2008.

[75] B. C. Stockdill. Multiple Oppressions and Their Influence on Collective Action. 1996.

[76] S. Straus and C. Taylor. Democratization and Electoral Violence in Sub-Saharan Africa, 1990-2007. 2009.

[77] C. S. Suh, I. B. Vasi, and P. Y. Chang. How social media matter: Repression and the diffusion of the Occupy Wall Street movement. *Social science research*, 65:282–293, 2017.

[78] J. Taylor and D. Watkinson. Indexing reliability for condition survey data. *The Conservator*, 30(1):49–62, 2007.

[79] S. Taylor. *Business statistics: for non-mathematicians*. Macmillan International Higher Education, 2007.

[80] C. Tilly. Collective violence in European perspective. 1978.

[81] C. L. Tucci, A. Afuah, and G. Viscusi. *Creating and capturing value through crowdsourcing*. Oxford University Press, 2018.

[82] J. A. Tucker, J. Nagler, M. MacDuffee, P. B. Metzger, D. Penfold-Brown, and R. Bonneau. Big Data, Social Media, and Protest. *Computational Social Science*, page 199, 2016.

[83] R. Voyer, V. Nygaard, W. Fitzgerald, and H. Copperman. A hybrid model for annotating named entity training corpora. pages 243–246. Association for Computational Linguistics, 2010.

[84] A. Wang, C. D. V. Hoang, and M.-Y. Kan. Perspectives on crowdsourcing annotations for natural language processing. *Language resources and evaluation*, 47(1):9–31, 2013.

[85] Z. Wang, Z. Li, J. Li, J. Tang, and J. Z. Pan. Transfer learning based cross-lingual knowledge extraction for wikipedia. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 641–650, 2013.

[86] N. Wanichayapong, W. Pruthipunyaskul, W. Pattara-Atikom, and P. Chaovalit. Social-based traffic information extraction and classification. pages 107–112. IEEE, 2011.

[87] S. Weiss, N. Indurkhya, T. Zhang, and F. Damerau. From Textual Information to Numerical Vectors. In *Text Mining*, pages 15–46. Springer New York, Jan. 2005.

[88] R. White. Comparing State Repression of Pro-State Vigilantes and Anti-State Insurgents: Northern Ireland, 1972-75. *Mobilization: An International Quarterly*, 4(2):189–202, 1999.

[89] X. Yang, C. Macdonald, and I. Ounis. Using word embeddings in twitter election classification. *Information Retrieval Journal*, 21(2-3):183–207, 2018.

[90] T. Yano, P. Resnik, and N. A. Smith. Shedding (a thousand points of) light on biased language. pages 152–158. Association for Computational Linguistics, 2010.

[91] M. Yu, Q. Huang, H. Qin, C. Scheele, and C. Yang. Deep learning for real-time social media text classification for situation awarenessâĂŞusing Hurricanes Sandy, Harvey, and Irma as case studies. *International Journal of Digital Earth*, pages 1–18, 2019.

[92] H. Zhang and J. Pan. Casm: A deep-learning approach for identifying collective action events with text and image data from social media. *Sociological Methodology*, 49(1):1–57, 2019.

[93] Z. Zhang, D. Robinson, and J. Tepper. Detecting hate speech on twitter using a convolution-gru based deep neural network. pages 745–760. Springer, 2018.

[94] Y. Zhao and Q. Zhu. Evaluation on crowdsourcing research: Current status and future direction. *Information Systems Frontiers*, 16(3):417–434, 2014.