2020

Evaluating flood disaster tweets through the use of text mining

ISMAIL BHAMJEE

Dissertation submitted in partial fulfillment of the
requirements for the degree of master by advance study in
Big data science and technology

Department of computer science
University of bradford

Contents

Abstract

With the increased usage of social media, information regarding major events has become more abundant. During these times people express their views on what is happening in their local surroundings via various mediums.

This gives rise to the potential of using such data to better understand and react to events such as floods.

Whereas traditional Methods such as image analysis are not suitable for understanding the impact such events have on the local community. Thus, we opted for tweet analysis as they are more personal to the event and community.

This paper hopes to develop system in which tweets are used and processed until the level of assigning a risk factor to them. This would subsequently aim to help local authorities in the west Yorkshire primarily the bradford city council understand the damage alongside future risks. This paper will traverse through the necessary steps which are needed for the project to work. Alongside any meaningful discoveries.

Introduction

Natural disasters occur across the word and often they impact humans in terms of damage to property or loss of lives. For example, floods which can be caused by various methods such as river floods, costal floods or drainage problems. Next these disasters bring problems in many ways such as information transfer between people affected and rescue workers. A solution for this problem would be access to real time information in order to get the best possible help where needed such as aid relief and medical care.

Technology and the use of internet has allowed information which was previously unavailable such as daily updates or pre-emptive warnings to play a larger role in information distribution. This would also include the use of social media such as twitter and Facebook. Where is it possible to share media in terms of micro-blogs or images. As any individual can do this it becomes a source of information based on public input. However, our focus will be on twitter.

Twitter is platform which allows micro blogging by users. In this there can be many benefits and disadvantages. Benefits such as tweets are published instantly which make it a real time source of information. Next the volume of tweets produced in a disaster can increase by 6 times as shown in the development of hurricane sandy. Naturally with the increase of tweets comes more verity in their content. For example, the language used varies from user to user, this can also be in form of dialect changes and slang alongside their outtake on the event such as sarcasm or use of emojis. These problems must be taken into consideration.

Twitter is a reactive platform in terms that an event would have occurred for the sending of a tweet such as life event or a public event. This would the render opinion mining tools obsolete as they are event driven rather than topic driven. An example of this is a user will not express sentiment about a topic at random such as a political view. However, if a political event did occur then said user would be more like to react to it.

In order for us to analyse the flooding of Yorkshire we would have look at key events which would help us locate appropriate responses. Thus, providing a better source of data to work with.

In this work we aim to collect and analyse tweet data during the Yorkshire floods of 2015 and 2016 for the possible implementation of a detection/warning system.

In this period the west Yorkshire economy suffered a loss of £170 million.[1] During this people were left without help or information. However, people were communicating through the use of social media.

The tasks conducted are comprised of several parts: deciding the parameters of data, obtaining the data, cleaning the data, creating meaningful vectors and clustering. This will contribute to a better understanding of the severity of the floods, with addition of discovery of new aeras affected.

Background

In late 2015/early 2016 the region of Yorkshire was hit by severe flooding. This was due to multiple storms [2]

The damage produced by the storms lead to loss of housing, business and infrastructure. Over the course of the floods the local economy was at a loss as for every £1 of damage another £0.6 was lost.

The following damages occurred in the Calderdale borough

- 45% of premises suffered structural damage.
- 75% of local business lost stock.
- 46% of office business lost stock.
- Over 1600 local business affected by flooding
- Over 2800 houses were affected.

These floods took roughly double the time to recover as compared to the 2012 floods. With business experiencing 1.35 times more damage with losses also doubling. The knock-on effect would mean that businesses will become uninsurable. [3]

In this rapidly changing condition correct information and news was not readily available. During this challenge traditional reliance of structural measures was no long enough to support a community in their risk management. This introduces the concept of non-structural management such flood proofing, forecasting and warning.

The success of this risk assessment is dependent on the publics willingness to provide data on the issues. However, this comes with many issues as the public have different cultural norms, language barriers, household structures and perceptions of risk. These differences need to be taken into account for a successful interpretation of their views.[4]

Understanding though the lens of a community helps planners better the structural measures, as they would have access to reliable on the ground data for future prevention.[5]

Social media has been widely used for crisis communication. For our selected task we used twitter as they have an instant repose time. Furthermore, they have a greater capacity compared to regular mass media as they implement a two-way system which allows users to interact with each other via a blog thread. Although this gives room for insincere application of tweets. However, service providers and researches remain open to its benefits as seen by the plethora of studies conducted using tweet data. [6] (Lindsay, 2011)

Furthermore, a comparison would be hurricane Sandy's tweet output from before formation to after impact.

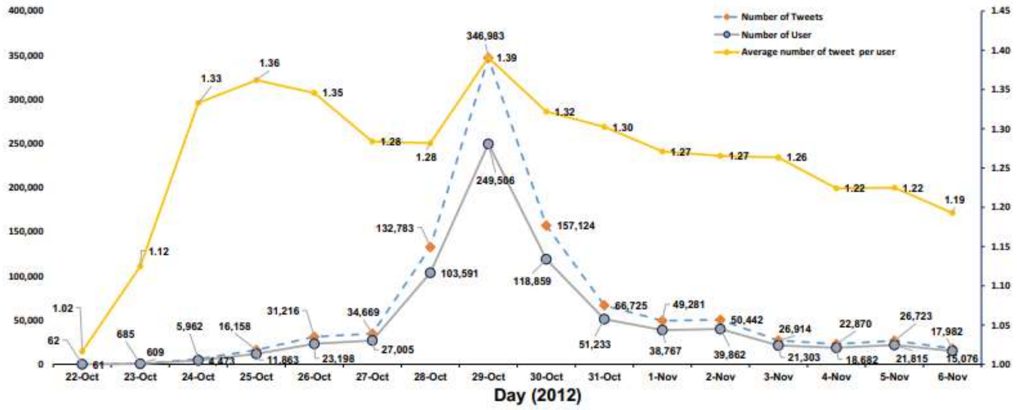


Figure 1volume of tweets during hurricane sandy

As shown the number of tweets increased rapidly and didn't subside to pre event levels. [7]

Information categories	Description	(%)
Disaster/events update	This type of information is used to report updates about the disaster, such as location, wind power	33.93
Information source tips/ recommendations	This use refers recommend the public members where to get the latest disaster information	23.77
Response/preparing report	This use is to report how official's response to or prepare disasters	15.72
Situation report	This type of information is to report people's own situation or how people are affected by this disaster, such as death, power lose	12.40
Response/preparing tips	This use is to offer tips about how to prepare and respond to Hurricane Sandy	9.90
Discussions	Include presidential elections, origin of disaster names, etc	2.81

Information collection	This use refers to collect how people response to disasters and their situation information	0.83
Expressing wishes and memorializing	This type of tweet is used to advertise products to people, such as insurance, generator	0.13
Requesting help	Request help from other, such as distributing information	0.13
Help tips	This use is to direct people how to offer hands during disasters	0.06

Table 1, type of tweets sent out during hurricane sandy

Tweet information type during disasters.

Next the paper" Crisis information distribution on Twitter: a content analysis of tweets during Hurricane Sandy" gave a comprehensive understanding of what types of tweets were sent out during that crisis as shown above. [7]

Literature review

In this section we will consider applications of tweet mining already implemented or theorised to gain a better understand of what the wider scientific community thinks of it.

Mining Twitter to Inform Disaster Response:

Introduces new tool called Tweetr which is able to Extract actionable information for disaster relief worker using natural disasters. Pipeline contains 3 parts classification, clustering and extraction. Classification used to identify damage or casualties. Clustering used for merging tweets which are similar. Specific tokens and phrases which report information on infrastructure damage, damage types, and casualties. Lastly, they validated tweets from 12 different crises in the US

They Manage to extract information from noisy social media data.
Results show that it's possible to draw out small information from heterogenous twitter data using small set of labels data. Using their methods, they were able to derive actionable info from real time tweets. [8]

Real-Time Detection of Traffic From Twitter Stream Analysis

Social media has been used for event detection. This uses road traffic congestion and car accidents.
Real time monitoring system for traffic. Fetches tweets from search criteria.
Process and classification Aim to attach each class label to each tweet. However, it is Unused on the Italian road network Before online new or websites.
Support vector machine was used in classification with an accuracy of 95.75%
Able to see whether traffic was caused by external events or not Accuracy at 88.89%
built on a Service oriented architecture. able to fetch and classify streams. and to see if users are able to discriminate between events. accuracy of 95.75%, for the 2-class problem. 88.89% for the 3-class problem. [9]

Large-scale Twitter Mining for Drug-related Adverse Events

For the use of adverse benefit of the drugs. For people who used new/ on shelf drugs. Used by pharma intelligence, Existing methods rely on patient reports. source for finding potential adverse events. using Natural Language Processing Support Vector Machine (SVM) classifiers are used
High performance computing was used for the 2 billion tweets
using MapReduce results show that daily social networking data could help early detection of important patient safety issues.

Key parts of the classification are the features. Textual and semantic features were considered. Used same feature extraction Both used the same feature extraction, In drug user classification only drug related tweets were used.
While the scope was expanded in the AE classification. Textual features were used such as:

- Number of hashtags
- Number of user mentions
- Number of words
- Number of URLS
- Number of pronouns
- Number of occurrences

[10]

Twitter as a Distributed Sensor System

Social media feeds are often used for the contribution and distribution of information. In them exists information which often includes refences to specific events and its location. In this paper the spatial and temporal characteristics of a twitter feed responding to a 5.8 magnitude earthquake.
In this they argued that such feeds can be used as a sensor system to identify and locate impacted areas. The experiment supported the use of people as sensors to provide data which improved situational awareness, understanding and response to events.
The conclusion of this paper suggest that tweets evolved from simply saying the event happened to news about the event. Furthermore, it claims that within the first 10 minutes a "fast and good" approximation can be derived. [11]

Determining disaster severity through social media analysis.

This paper justifies the use of social media as a prominent information source through the use social medias diverse user base. However, it states that current analytic tools are not suitable to measure the severity of the disaster in local communities using real time data. To improve the current applications, they proposed their own framework in this they manage to identify differences in events. Furthermore, they identified different types of tweets during a disaster time period which would provide an insight to what is going on. Lastly, they reinforced the usage of geographically locate tweets as this would provide a better insight and how reduction of these type of tweets provided constraint to the project. [12]

Methodology

As with any project or operation we would need a plan or a method of conducting activities. We used anaconda's distribution which allowed to import the needed libraries with ease. For development we used Jupiter notebook as it allows ease of use and it was fairly simple to use regarding executing code blocks.

1. The first step would be to retrieve key words which reflect the events which occurred during the flooding period of 2015.

2. Second would be to retrieve tweets which contain relevance to the key words and events which ae also suitable for text mining.

3. evaluate and process the tweets which includes cleaning and making them eligible for word vectorisation.

4. Next the tweets are vectorised using different methods word2vec, fast text and doc2vec. With addition of visualisation of vectors in relation to each other.

5. perform unsupervised classification on the tweets to see how further they can be grouped together.

6. Next, we will perform topic modelling which will enable us to derive suitable topics from the clusters.

7. Lastly, we will test tweets via attempted allocation to the correct cluster and then topic.

Text mining

Text mining is the discovery of new information by a computer which would have been hidden to the human eye via the use of explicit rules. This is done via the extraction of information from written sources which are non-structed. which is done by linking the extracted information to new facts or theories which can be explored later by conventional means.

The text which is used for text mining is of a higher quality. Which makes all aspects of the text valuable? Such as in our case. The use of specific tweets with key words.

This is variation of data mining which focuses on fining patterns in existing structured databases. Such as customer buying habits though a constructed database. These databases are designed for programs to be able to access them using query-based logic.

Whereas text mining the patterns are extracted from natural language text such as articles and papers. The inherent use of natural language in texts means that programs cannot explicitly “read” and “analyse” the data given. [13]

Natural language processing

Natural language processing is a large area of research so a unified definition is not available and would not encompass the many interpretations of it. however, the following is a bare bones definition which are applicable to all.

Natural language processing is a range of computational processes for analysing and representing natural texts at different levels of linguistic analysis for the purpose of obtaining human like language processing for a range of tasks.

Parts of this definition could be defined further.

- “range of computational processes” this would mean there exists multiple methods to achieve a specific type of language analysis.
- “natural texts” are texts which exist in any domain or language. These can be written as documentation or as speech. However, the fundamental requirement would be that they serve as communication between humans and not for the sole purpose of analysis such as database entries.
- “different levels of linguistic analysis” they are multiple elements of language processing when humans produce or comprehend language. However, NLP systems only encompass certain levels or combination of levels. Levels included are phonology, morphology, lexical, syntactic, sematic, discourse and pragmatic.

- “range of tasks” NLP is not the end goal itself, however its use in a particular task such as information retrieval systems which result in applications such as question answering.

  [14]

Limitations of Text mining

Text mining is a newly found discipline which does not have any one correct way or methodology. Obstacles can occur when trying to get the required data. This may hinder researches in their work when they’re collecting and analysing the data.

Legal protections and tools which limit the practice of TDM. The first is that activities conducted during TDM could be considered copyright infringements or violation of the database SUI generis law. The second being contracts which limit or disallow TDM processes. Technology protection measures and can subject intellectual property rights and contractual obligations to hinder TDM. Lastly the use of personal data which could identify a person would be a violation of data protection laws. [15]

An example of this would be if a sales company was to mine data from disasters and to see what was damaged in order to promote affected or damaged items. This would be unethical as it would essentially give a list of what people needed.

Cost is an important factor in our case as social media providers charge for use of archived data. So, the section of what type of data and its quantity is a factor which has to be taken into account…

Data gathering and pre-processing.

Key words

In order to see what types of tweets were recovered via the process a manual search was done to see the brief content of the search via twitters search features.

Search criteria	Good tweets	Bad tweets
Yorkshire flood until:2016-01-01 since:2015-12-01	Elland Bridge in West **Yorkshire** closed after **floods** cause road to collapse http://itv.com/news/story/2015-12-29/storm-frank-more-heavy-rain-expected-in-flood-hit-areas/	Saturday Afternoon Headlines: Red **flood** warnings for Lancashire & **Yorkshire**, China mine collapse & Australia fires http://snpy.tv/1YIeIY1
	In **Yorkshire** & Lancashire, whole community is united in **#flood** response. Heartening to see.	#yorkfloods In 2012 #coalition 30% cut in **#Yorkshire** **flood** defence funding. **#floods** #COBRA #CutOffTheHeadOfTheSnake
	Thoughts are with Mark Warriner's business in Sowerby West Bridge, West **Yorkshire** #boxing day **#flood**	What could possibly go wrong with fracking in a **flood** zone? #fracking #flooding **#yorkshire** #keepitintheground

Table 2, search criteria for tweets

Search criteria	Good tweets	Bad tweets

Bradford flood until:2016-01-01 since:2015-12-01	Hanfia Mosque in **Bradford** have been packing and delivering food to those affected by **#Floods**.	The **Bradford floods** of 1947. #BradfordHistory @BradfordLoverUK @bradfordmdc @hiddenbradford
	#Bradford bikers protect **#floods** victims' empty homes #bigupBradford Charity night at The Northern	**Flood** Warning, 9:43AM 26/12/2015, Batley Beck at **Bradford** Road through central Batley, http://floodalerts.com/?id=94560

Table 3, search criteria for tweets

Search criteria	Good tweets	Bad tweets
calder river flood until:2016-01-01 since:2015-12-01	A giant section of riverbank taken out by **River Calder flood** on boxing day beneath Whalley Viaduct	**River Calder** level in Hebden Bridge 26/12. Please sign and share: Dredge the **river Calder** https://petition.parliament.uk/petitions/115970
	Short video I just took of **River Calder** in **flood** between Hebden Bridge & Mytholmroyd. So sorry for those affected.	Severer **flood** warnings on **Rivers Calder**, Ribble and Wyre …https://flood-warning-information.service.gov.uk/warnings

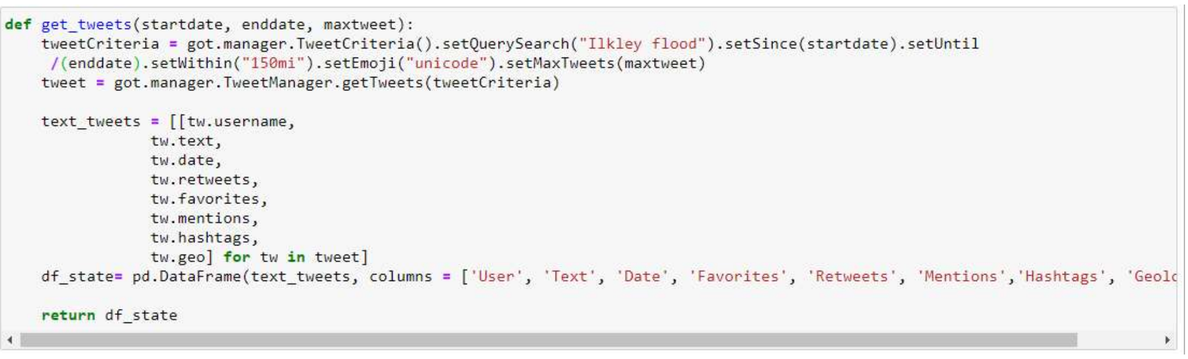
Table 4, search criteria for tweets

By this process we could see that good tweets talk about the flood directly and people or property which are affected. However bad tweets usually contain links to media which would be unusable as there would be no way to verify the legitimacy of it. Next bad tweets contain automated reposes by weather bots which isn't people talking about the events.

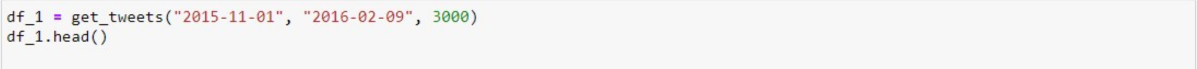
Gathering tweets

Twitter own API allows users to buy tweets at a specific rate, however due to financial restraints tweets could not be purchased. Its API allows us to search via various parameters such as key words, date frame and location.

In order to get tweets the use of Get Old tweets was used, this is a python library which allowed us to bypass the need to purchase the tweets, which for research can be costly and this is a trial and error process.

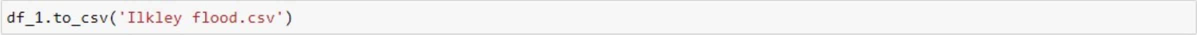


```
def get_tweets(startdate, enddate, maxtweet):
    tweetCriteria = got.manager.TweetCriteria().setQuerySearch("Ilkley flood").setSince(startdate).setUntil
     /(enddate).setWithin("150mi").setEmoji("unicode").setMaxTweets(maxtweet)
    tweet = got.manager.TweetManager.getTweets(tweetCriteria)

    text_tweets = [[tw.username,
                  tw.text,
                  tw.date,
                  tw.retweets,
                  tw.favorites,
                  tw.mentions,
                  tw.hashtags,
                  tw.geo] for tw in tweet]
    df_state= pd.DataFrame(text_tweets, columns = ['User', 'Text', 'Date', 'Favorites', 'Retweets', 'Mentions','Hashtags', 'Geolc

    return df_state
```



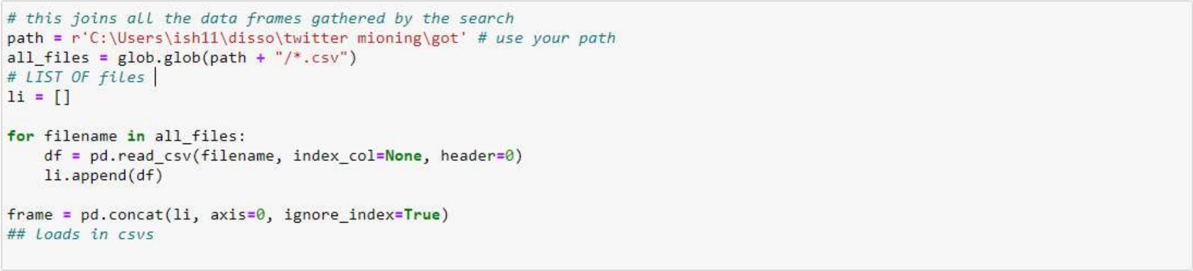
```
df_1 = get_tweets("2015-11-01", "2016-02-09", 3000)
df_1.head()
```

The function above allowed us to get tweets which matched our given parameters. these could be changed when executing the function. To facilitate for different searches and various regions of affected areas.



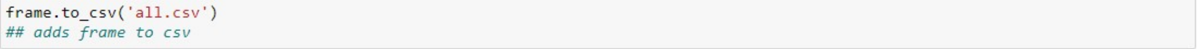
```
df_1.to_csv('Ilkley flood.csv')
```

This was then composed into a data frame which was later exported as a CSV file. Due to the pandas data frame being reset on each use. Once the data sets were obtained, we complied it into one file for future processing.

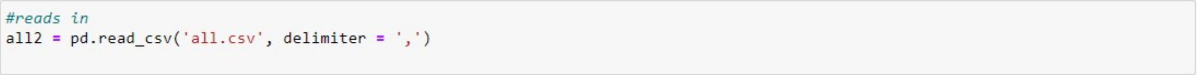


```
# this joins all the data frames gathered by the search
path = r'C:\Users\ish11\disso\twitter mioning\got' # use your path
all_files = glob.glob(path + "/*.csv")
# LIST OF files |
li = []

for filename in all_files:
    df = pd.read_csv(filename, index_col=None, header=0)
    li.append(df)

frame = pd.concat(li, axis=0, ignore_index=True)
## loads in csvs
```

The above allows to place all the files together in one file.

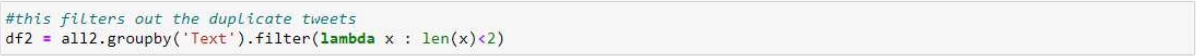


```
frame.to_csv('all.csv')
## adds frame to csv
```

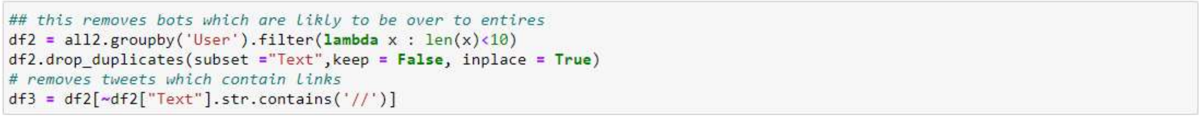


```
#reads in
all2 = pd.read_csv('all.csv', delimiter = ',')
```

Next the data fame is saved and read back in into a new df for future use.



```
#this filters out the duplicate tweets
df2 = all2.groupby('Text').filter(lambda x : len(x)<2)
```



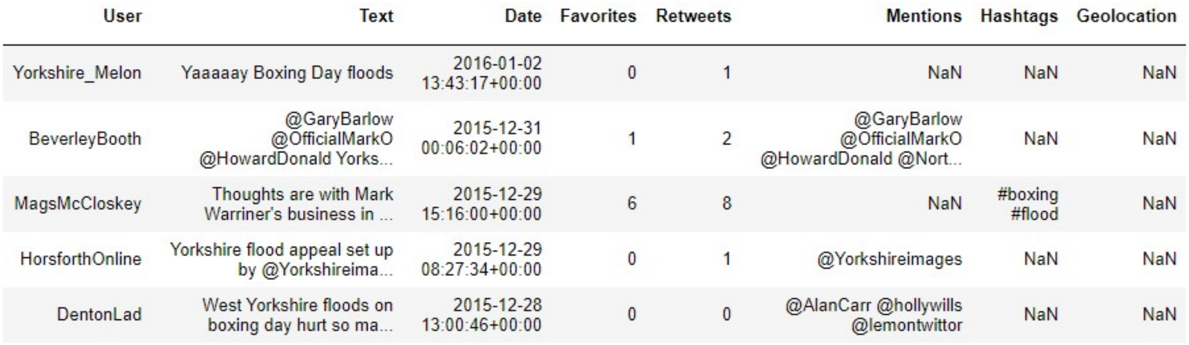
```
## this removes bots which are likly to be over to entires
df2 = all2.groupby('User').filter(lambda x : len(x)<10)
df2.drop_duplicates(subset ="Text",keep = False, inplace = True)
# removes tweets which contain links
df3 = df2[~df2["Text"].str.contains('//')]
```

Next duplicates rows are removed from the Text column.

Then it removed records which have more than 10 values which are the same in the username column. This is to remove weather bots and auto tweeters. As this would greatly impact our natural language analysis. Lastly it removes any records from the text column which contain “//” as they are most likely to be links to media or articles.

Lastly this is saved into a CSV file which we open and remove any indexing overlaps created by multiple data frames being stitched together.

The following figure shows what the data frame looks like in terms of retrieving the raw tweets.

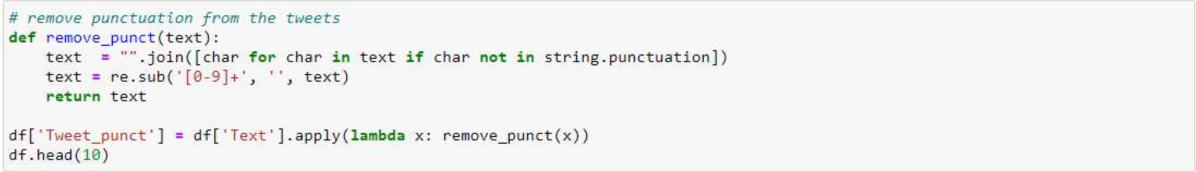


User	Text	Date	Favorites	Retweets	Mentions	Hashtags	Geolocation
Yorkshire_Melon	Yaaaaay Boxing Day floods	2016-01-02 13:43:17+00:00	0	1	NaN	NaN	NaN
BeverleyBooth	@GaryBarlow @OfficialMarkO @HowardDonald Yorks...	2015-12-31 00:06:02+00:00	1	2	@GaryBarlow @OfficialMarkO @HowardDonald @Nort...	NaN	NaN
MagsMcCloskey	Thoughts are with Mark Warriner's business in ...	2015-12-29 15:16:00+00:00	6	8	NaN	#boxing #flood	NaN
HorsforthOnline	Yorkshire flood appeal set up by @Yorkshireima...	2015-12-29 08:27:34+00:00	0	1	@Yorkshireimages	NaN	NaN
DentonLad	West Yorkshire floods on boxing day hurt so ma...	2015-12-28 13:00:46+00:00	0	0	@AlanCarr @hollywills @lemontwittor	NaN	NaN

Figure 2 data frame of tweets collected

Cleaning

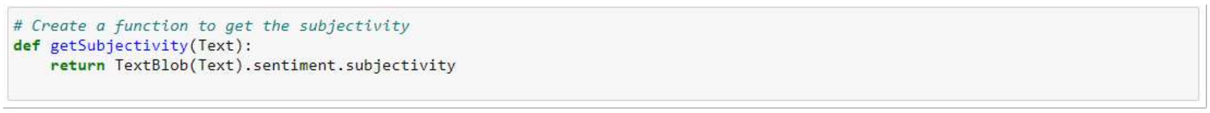
From the complete data only the tweet and user data were needed. So, a separate DF was created to start working on.



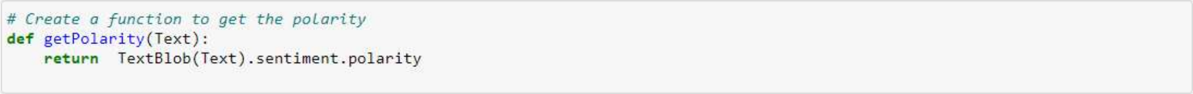
```
# remove punctuation from the tweets
def remove_punct(text):
    text  = "".join([char for char in text if char not in string.punctuation])
    text = re.sub('[0-9]+', '', text)
    return text

df['Tweet_punct'] = df['Text'].apply(lambda x: remove_punct(x))
df.head(10)
```

The following functions create sentiment subjectivity and polarity from the data without emojis.

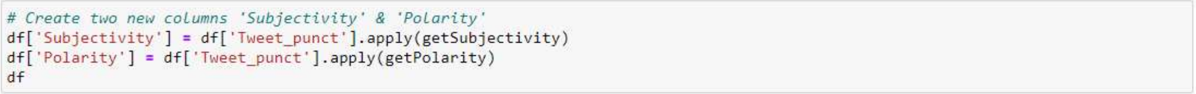
They are then placed in a different column.



```
# Create a function to get the subjectivity
def getSubjectivity(Text):
    return TextBlob(Text).sentiment.subjectivity
```

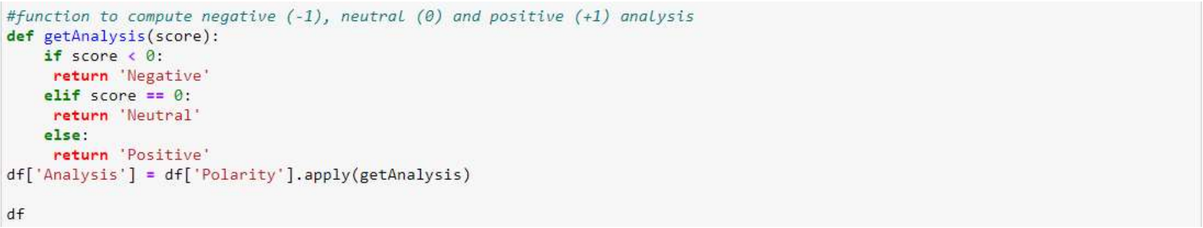


```
# Create a function to get the polarity
def getPolarity(Text):
    return  TextBlob(Text).sentiment.polarity
```

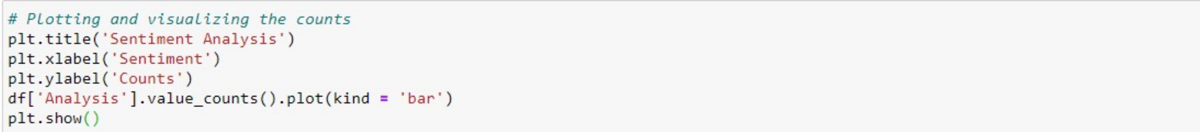


```
# Create two new columns 'Subjectivity' & 'Polarity'
df['Subjectivity'] = df['Tweet_punct'].apply(getSubjectivity)
df['Polarity'] = df['Tweet_punct'].apply(getPolarity)
df
```

This function attaches a score to the tweets and gives them either a positive, neutral or negative rating based on the previous polarity. The block below produces a bar graph of the 3 values.

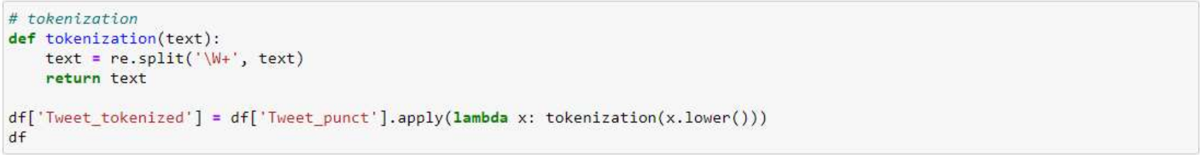


```
#function to compute negative (-1), neutral (0) and positive (+1) analysis
def getAnalysis(score):
    if score < 0:
     return 'Negative'
    elif score == 0:
     return 'Neutral'
    else:
     return 'Positive'
df['Analysis'] = df['Polarity'].apply(getAnalysis)

df
```



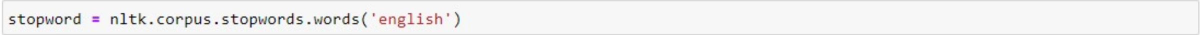
```
# Plotting and visualizing the counts
plt.title('Sentiment Analysis')
plt.xlabel('Sentiment')
plt.ylabel('Counts')
df['Analysis'].value_counts().plot(kind = 'bar')
plt.show()
```

Next, we tokenize the clean text. Which means to get individual tokens form the tweets.



```
# tokenization
def tokenization(text):
    text = re.split('\W+', text)
    return text

df['Tweet_tokenized'] = df['Tweet_punct'].apply(lambda x: tokenization(x.lower()))
df
```

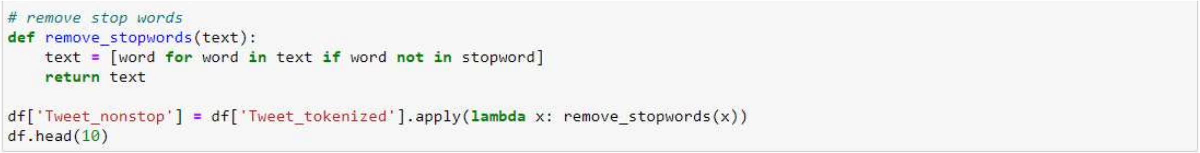
Stop words is a variable which contains all stop words in the English language



```
stopword = nltk.corpus.stopwords.words('english')
```

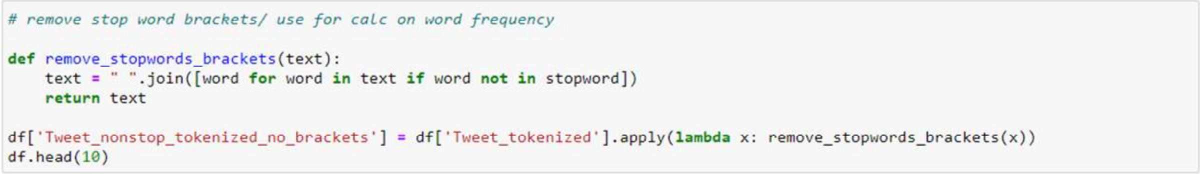
Removing stop words from the tokenised list.

Stop words are word used by human ho help easy the meaning of a statement. However, if the stop words are removed it would not hinder the meaning. The essence of the meaning can still be obtained. These are removed as they don’t provide any meaningful contribution. Example of such is:“ we will go to the shop” into “we will go shop”



```
# remove stop words
def remove_stopwords(text):
    text = [word for word in text if word not in stopword]
    return text

df['Tweet_nonstop'] = df['Tweet_tokenized'].apply(lambda x: remove_stopwords(x))
df.head(10)
```

Separate tokenized column used to calculate word frequency. Along with the code to rung a bar graph.

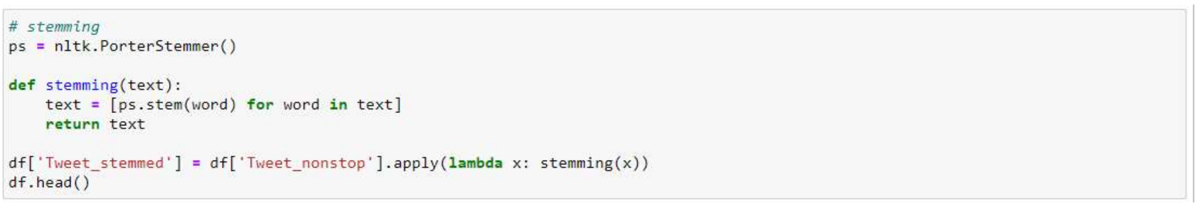


```
# remove stop word brackets/ use for calc on word frequency

def remove_stopwords_brackets(text):
    text = " ".join([word for word in text if word not in stopword])
    return text

df['Tweet_nonstop_tokenized_no_brackets'] = df['Tweet_tokenized'].apply(lambda x: remove_stopwords_brackets(x))
df.head(10)
```

Stemming in a process to reducing a word into its root words by removing its suffix. Such as “eating” and “ate ”to “eat”. Its main use is to reduce inflection Form of different words in order to get a single word deprived of any tense or attribute. This method is not perfect as words can be over stemmed or under stemmed.

Over stemming is when much of the word is removed depriving it of its actual meaning. Such as university or universe. Over stemming could reduce these words to “univers”

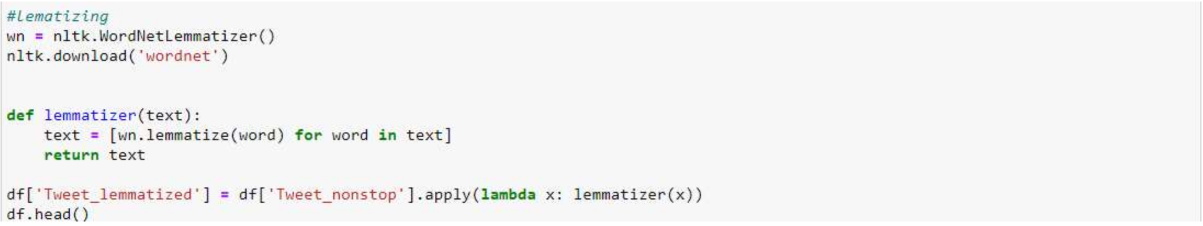
Under-stemming when a word is not stemmed enough resulting in the same work having different forms amongst the corpus. In this case we used porter stemmer as the data set was not large furthermore its focus is based around removing common ending of words. Others such as snowball stemmer and Lancaster stemmer were considered but for a small corpus comprising of 8127 tokens, they weren't necessary.



```
# stemming
ps = nltk.PorterStemmer()

def stemming(text):
    text = [ps.stem(word) for word in text]
    return text

df['Tweet_stemmed'] = df['Tweet_nonstop'].apply(lambda x: stemming(x))
df.head()
```

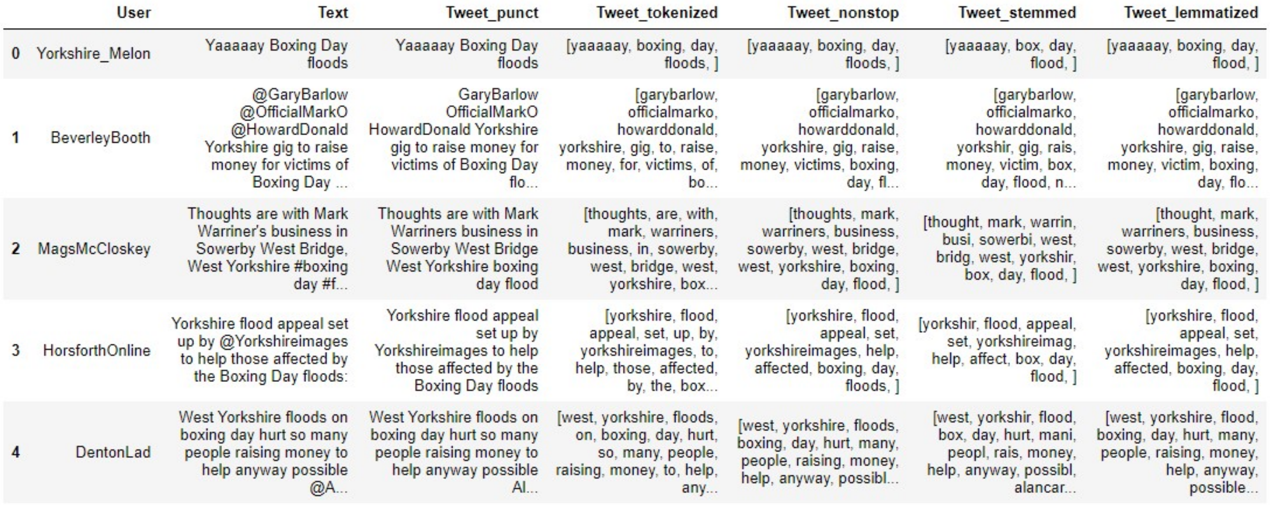
Lemmatizing

This is a more calculated process of defining where to start the drop off. It converts the words into their dictionary form. This uses the words part of speech. It uses the "word net" database decipher synonyms.



```
#Lematizing
wn = nltk.WordNetLemmatizer()
nltk.download('wordnet')


def lemmatizer(text):
    text = [wn.lemmatize(word) for word in text]
    return text

df['Tweet_lemmatized'] = df['Tweet_nonstop'].apply(lambda x: lemmatizer(x))
df.head()
```

The following figure shows the steps of cleaning in actual dataset. As we can see from each column the initial tweet changes to facilitate for future work.



	User	Text	Tweet_punct	Tweet_tokenized	Tweet_nonstop	Tweet_stemmed	Tweet_lemmatized
0	Yorkshire_Melon	Yaaaaay Boxing Day floods	Yaaaaay Boxing Day floods	[yaaaaay, boxing, day, floods,]	[yaaaaay, boxing, day, floods,]	[yaaaaay, box, day, flood,]	[yaaaaay, boxing, day, flood,]
1	BeverleyBooth	@GaryBarlow @OfficialMarkO @HowardDonald Yorkshire gig to raise money for victims of Boxing Day ...	GaryBarlow OfficialMarkO HowardDonald Yorkshire gig to raise money for victims of Boxing Day flo...	[garybarlow, officialmarko, howarddonald, yorkshire, gig, to, raise, money, for, victims, of, bo...	[garybarlow, officialmarko, howarddonald, yorkshire, gig, raise, money, victims, boxing, day, fl...	[garybarlow, officialmarko, howarddonald, yorkshir, gig, rais, money, victim, box, day, flood, n...	[garybarlow, officialmarko, howarddonald, yorkshire, gig, raise, money, victim, boxing, day, flo...
2	MagsMcCloskey	Thoughts are with Mark Warriner's business in Sowerby West Bridge, West Yorkshire #boxing day #f...	Thoughts are with Mark Warriners business in Sowerby West Bridge West Yorkshire boxing day flood	[thoughts, are, with, mark, warriners, business, in, sowerby, west, bridge, west, yorkshire, box...	[thoughts, mark, warriners, business, sowerby, west, bridge, west, yorkshire, boxing, day, flood,]	[thought, mark, warrin, busi, sowerbi, west, bridg, west, yorkshir, box, day, flood,]	[thought, mark, warriners, business, sowerby, west, bridge, west, yorkshire, boxing, day, flood,]
3	HorsforthOnline	Yorkshire flood appeal set up by @Yorkshireimages to help those affected by the Boxing Day floods:	Yorkshire flood appeal set up by Yorkshireimages to help those affected by the Boxing Day floods	[yorkshire, flood, appeal, set, up, by, yorkshireimages, to, help, those, affected, by, the, box...	[yorkshire, flood, appeal, set, yorkshireimages, help, affected, boxing, day, floods,]	[yorkshir, flood, appeal, set, yorkshireimag, help, affect, box, day, flood,]	[yorkshire, flood, appeal, set, yorkshireimages, help, affected, boxing, day, flood,]
4	DentonLad	West Yorkshire floods on boxing day hurt so many people raising money to help anyway possible @A...	West Yorkshire floods on boxing day hurt so many people raising money to help anyway possible Al...	[west, yorkshire, floods, on, boxing, day, hurt, so, many, people, raising, money, to, help, any...	[west, yorkshire, floods, boxing, day, hurt, many, people, raising, money, help, anyway, possibl...	[west, yorkshir, flood, box, day, hurt, mani, peopl, rais, money, help, anyway, possibl, alancar...	[west, yorkshire, flood, boxing, day, hurt, many, people, raising, money, help, anyway, possible...

Figure 3data frame of processed tweets

Embedding

Since the tweets we chose have been processed accordingly, we implement embedding, this is a numerical representation of a word.

Word embedding plays a major part in building word vectors based on their context in large corpora. It captures both semantic and syntactic information of the word. This can then be used to produce word similarities which are key components in natural language processing. In this each word resides in a space alongside similar words with the same context.

For our application we sort to use multiple embedding techniques to evaluate different their uses and contribution to the final product.

Word 2 vec.

Word 2vec is the first of our embedding techniques we used. Word 2 vec has two modelling systems, being CBOW and Skip gram. These decipher how the words are processed in relation to each other.

Word2vec's focuses on the sematic relation between words, this method is improved via our removal of stop words during pre-processing. This means that the order of words does not affect prediction. For our experiment we used the default which was CBOW. This model predicts a centre given an instance of surrounding words. However, this means that the syntactic relation of the words is cast aside. In relation to our corpus we needed it to provide adequate word positionings in relation to each other. [16]

CBOW and SKIP gram

Cbow is similar to a feedforward network, it predicts the output word from other near word vectors. in essence wit tries to predict a words place based on its neighbouring words. Skip gram aims to produce vectors of similar words based on one word. Its basis is to predict words surrounding the given word.[17]

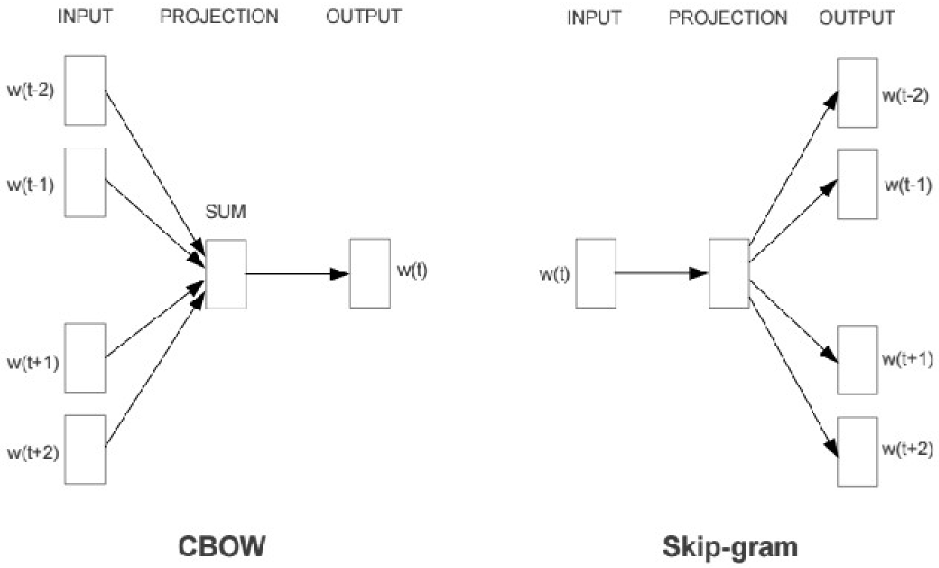
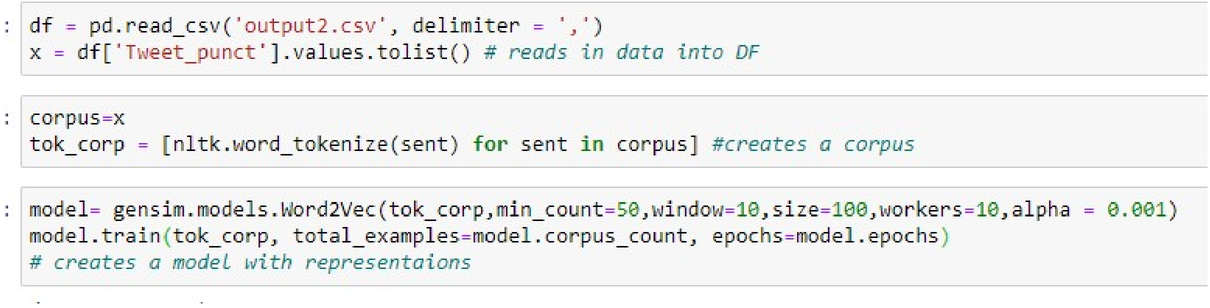


Figure 4 cbow and Skip gram architecture

[18] [De Gruyter Open, 2018]

Implementation



```
: df = pd.read_csv('output2.csv', delimiter = ',')
  x = df['Tweet_punct'].values.tolist() # reads in data into DF
```

```
: corpus=x
  tok_corp = [nltk.word_tokenize(sent) for sent in corpus] #creates a corpus
```

```
: model= gensim.models.Word2Vec(tok_corp,min_count=50,window=10,size=100,workers=10,alpha = 0.001)
  model.train(tok_corp, total_examples=model.corpus_count, epochs=model.epochs)
  # creates a model with representaions
```

The above code allows us to create representations of the words in a numerical format based on sematic similarity using the NLTK library. This also allowed us to change the parameters to better optimise the performance such as window size.

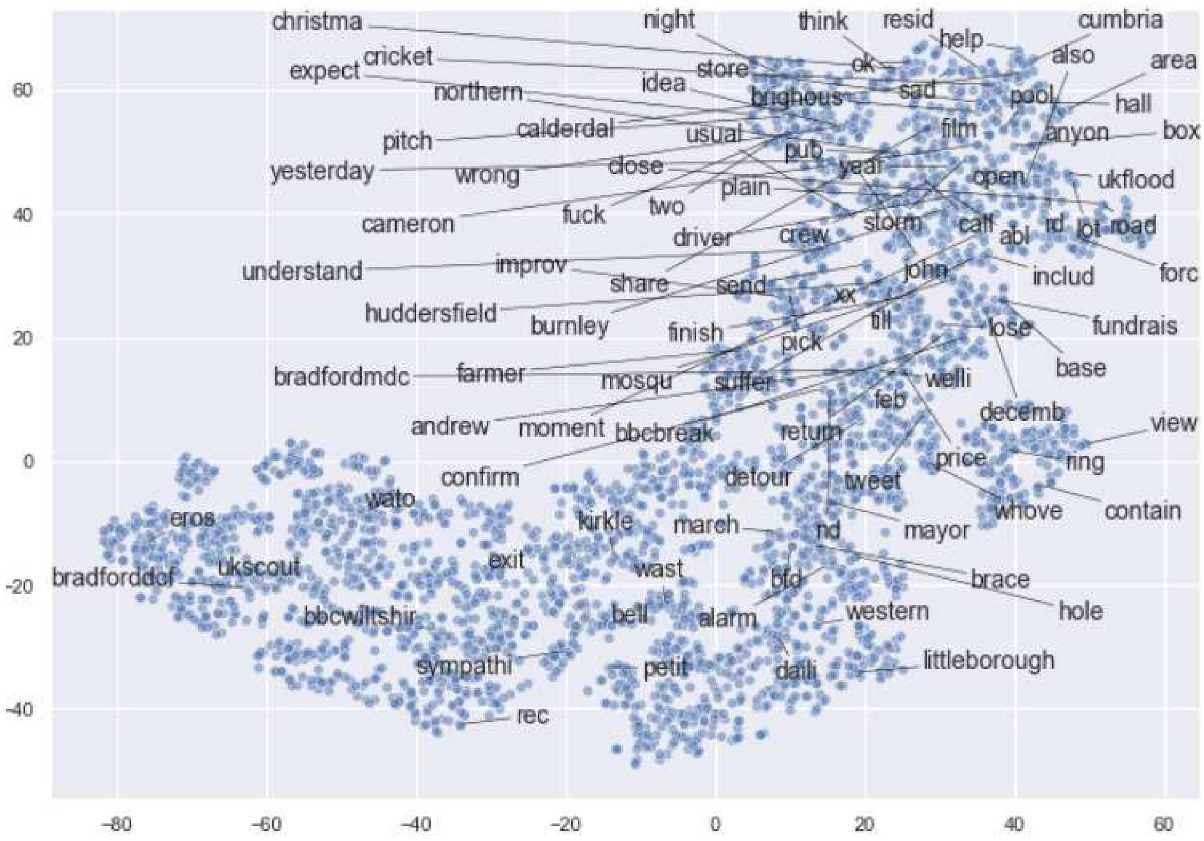


Figure 5 word2vec distribution of words based on sematic similarity.

The above figure show where words are placed in relation to each other during embedding. From this there is no clear indication of which words are close to each other. For this we performed a similarity test which took one word and found others most similar to it.

word	Similarity measure
air	0.9786539077758789
bank	0.9369361996650696
wharf	0.9165114164352417
level	0.910872757434845
instant	0.9074547290802002
alert	0.9053256511688232
high	0.8963362574577332
water	0.8902625441551208
saltair	0.8850008249282837
near	0.8784326314926147

Table 5, closest words based on similarity

The result showed were valid as all words could be associate with the test word. Next, we performed a similarly measure to compare two words in our embedding model.

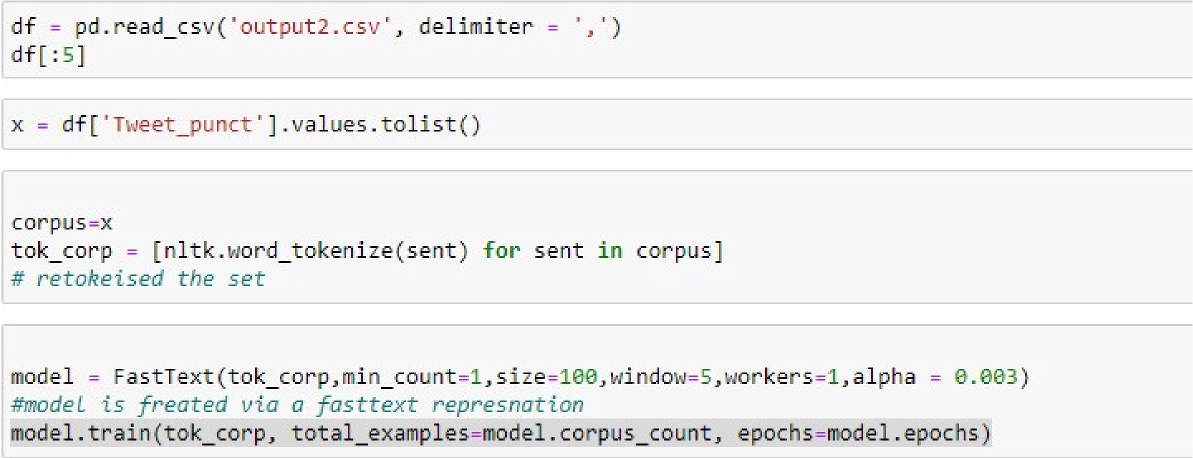
Word 1	Word 2	Similarity
Calder	Wet	0.05205367
Calder	safe	0.13359392
Calder	rain	0.15048048
Calder	leed	0.18625787
Calder	run	-0.009935208
Calder	walk	0.3638655
Calder	engin	0.41367233
Calder	calder	0.99999994

Table 6, word comparisons based on similarity.

The above shows words and how similar they are based on cosine similarity. As we can see the results were not what we expected in some cases such as the first comparison of the word “wet” to the test word. Which only gave a similarity measure of 0.05. naturally one would assume these two were related to each other. Furthermore, the word “engin” was somehow similar to the test word.

Fast text.

Fastext allows each word to be modelled by a sum of vectors. With each vector representing an N-gram. This is done so strength may be shared across the training process for words made of common roots. Words such as “bystander” which maybe rare in a corpus, however the words “by” “stand” may not be. This helps model rare words in a dictionary. Furthermore, character embedding is useful in representing slang or misspelt words such as “catz”.[19]



```
df = pd.read_csv('output2.csv', delimiter = ',')
df[:5]
```

```
x = df['Tweet_punct'].values.tolist()
```

```
corpus=x
tok_corp = [nltk.word_tokenize(sent) for sent in corpus]
# retokeised the set
```

```
model = FastText(tok_corp,min_count=1,size=100,window=5,workers=1,alpha = 0.003)
#model is freated via a fasttext represnation
model.train(tok_corp, total_examples=model.corpus_count, epochs=model.epochs)
```

The above code shows implementation of fast text. This code shares the same format of word2vec as it built upon the NLTK library. We can also change the parameters accordingly.

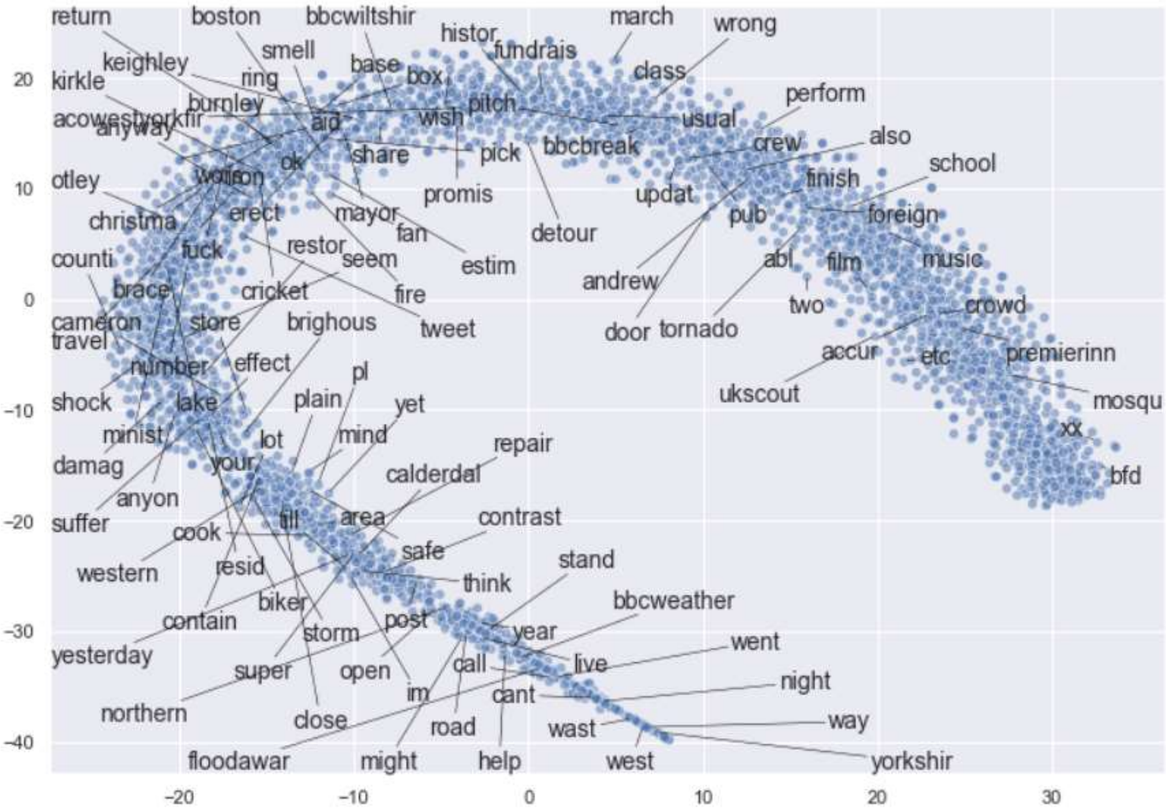


Figure 6 fast text distribution of words based on sematic similarity.

The above figure show where words are placed in relation to each other during embedding.as with word2vec we used cosine similarity measure to see which words were closest to our test word.

word	Similarity measure
caldervalley	0.9517279863357544
calm	0.9202991724014282
older	0.9149298667907715

bank	0.9136507511138916
calderdalemkt	0.907793402671814
calderdalelif	0.9028406739234924
builder	0.9013144373893738
calderdal	0.8987247347831726
calderdalecol	0.8958585858345032)
valley	0.8887931108474731

Table 7, closest words based on similarity

The above shows the closest word to the test word. This was different from the previous model of word2vec as the relation between words was not apparent.

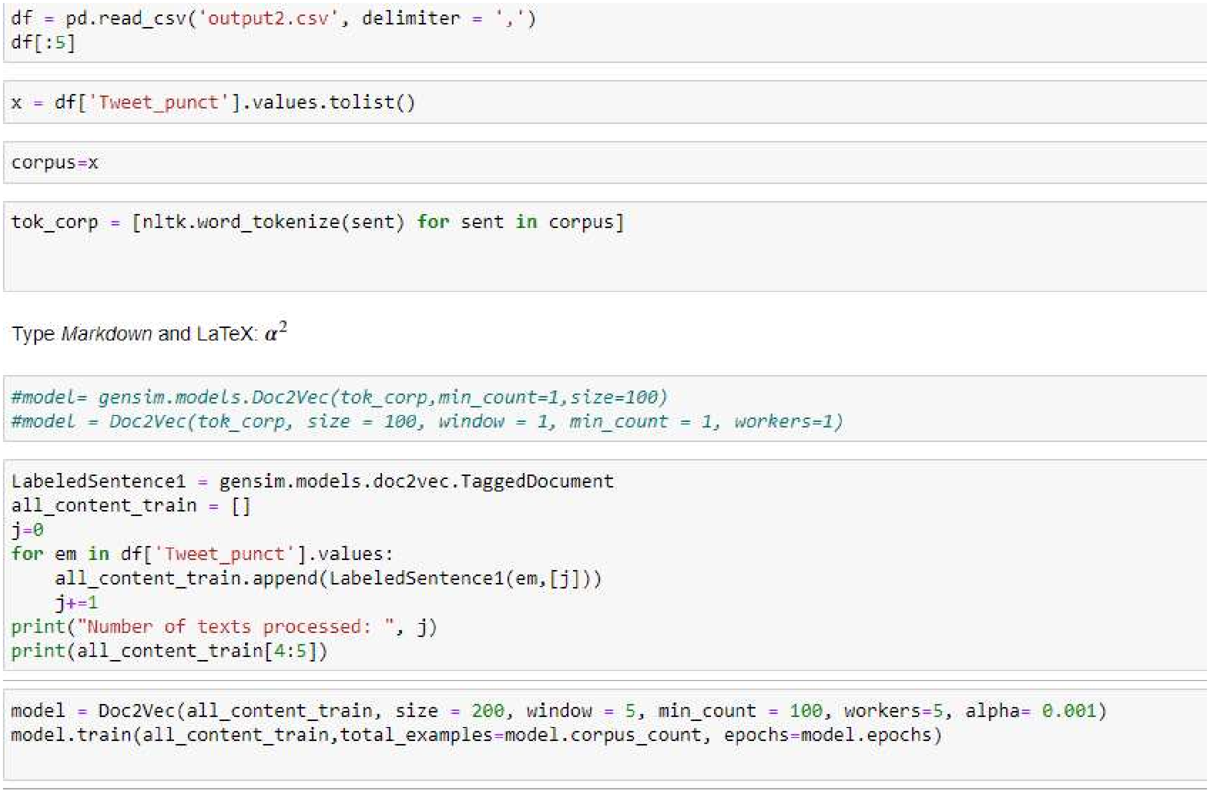
Word 1	Word 2	Similarity
Calder	Wet	-0.37158227
Calder	money	0.43689817
Calder	pig	0.6996176
Calder	leed	-0.13279098
Calder	rain	-0.18025687
Calder	engin	0.4004713
Calder	rescu	0.5223977
Calder	calder	1.0

Table 8, word comparisons based on similarity.

This figure shows the cosine distance between the two words the first being a test and the second being a semi-random word used to assess the distance.

Doc 2 vec.

Doc2vec is an extension of word 2 vec in the way embedding is learnt, from words to word sequences. It is unbiased regarding the way the word sequence is structured. Doc2vec does not represent a word in a vector rather a collection of words. This could be in a sentence or a paragraph. Doc2 vec has two models being DBOW and DMPV. [20] For our version we chose to implement DBOW(distributed bag of words). This method uses the concatenation of the paragraph vector with the word vector to predict the next word in a text window. [21]



```
df = pd.read_csv('output2.csv', delimiter = ',')
df[:5]
```

```
x = df['Tweet_punct'].values.tolist()
```

```
corpus=x
```

```
tok_corp = [nltk.word_tokenize(sent) for sent in corpus]
```

Type *Markdown* and LaTeX: α^2

```
#model= gensim.models.Doc2Vec(tok_corp,min_count=1,size=100)
#model = Doc2Vec(tok_corp, size = 100, window = 1, min_count = 1, workers=1)
```

```
LabeledSentence1 = gensim.models.doc2vec.TaggedDocument
all_content_train = []
j=0
for em in df['Tweet_punct'].values:
    all_content_train.append(LabeledSentence1(em,[j]))
    j+=1
print("Number of texts processed: ", j)
print(all_content_train[4:5])
```

```
model = Doc2Vec(all_content_train, size = 200, window = 5, min_count = 100, workers=5, alpha= 0.001)
model.train(all_content_train,total_examples=model.corpus_count, epochs=model.epochs)
```

The above shows implementation of doc 2 vec which is similar to the above 2 implementations as they all use NLTK library.

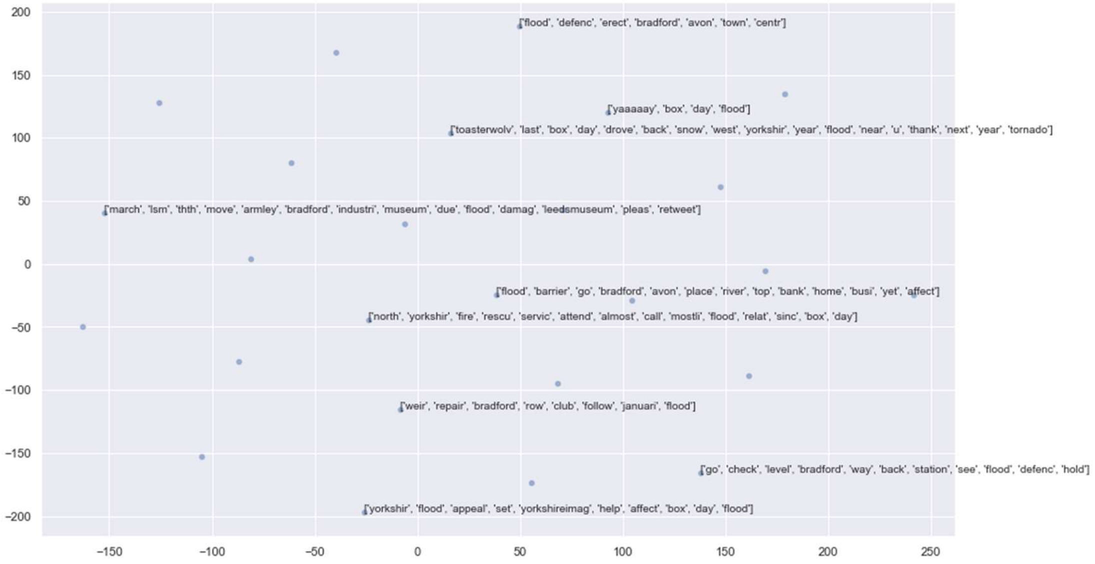


Figure 7 doc2vec distribution of documents based on sematic similarity

The above figure show where documents are placed in relation to each other during embedding.

For doc2vec we did not apply similarity measures as they were primarily based on word comparison not document comparisons.

Clustering

clustering by definition is to organise objects which are similar to each other by some way. For example, it could be to distinguish fruits based on shape. In data science It is mostly used when there are no apparent labels on a data set. Which clustering will help by association data which is similar in some way. Furthermore, it helps to generate a shape or an idea of what the data is about or discover similarities when no prior knowledge is known.

As we have no labels to our data and we would like to know the structure between words/documents, for this we chose hierarchical agglomerative clustering. Which allows us to generate a cluster of each element and as a given parameter changes, they merge into a larger cluster. So eventually all words will end up as one cluster from individual leaf nodes. The primary reason of this method is so we can wee where the most convergences happen and where they don't. so, we can select a cut-off point for out clusters.

In the paper" comparative study on text clustering methods" a comparison was made between frequent itemset based hierarchical clustering and k-means in-order to see the effectiveness of them via 3 validation criteria. From this FIHC was deemed better as it uses a larger search space and is able to generate nested clusters. With the downside of using more processing power and slowing down as document numbers increase. Furthermore, for K-means a number of clusters has to be chosen.[22]

In the paper "An Unsupervised Fuzzy Clustering Method for Twitter Sentiment Analysis" clustering was done using 3 methods. K-means, EM clustering and a modified fuzzy clustering algorithm. This was done to cut out manual processing and training time. This method of fuzzy clustering yielded high quality results with better accuracy than the other 2. With the additional bonus of speed and accuracy.[23]

Implemented clustering

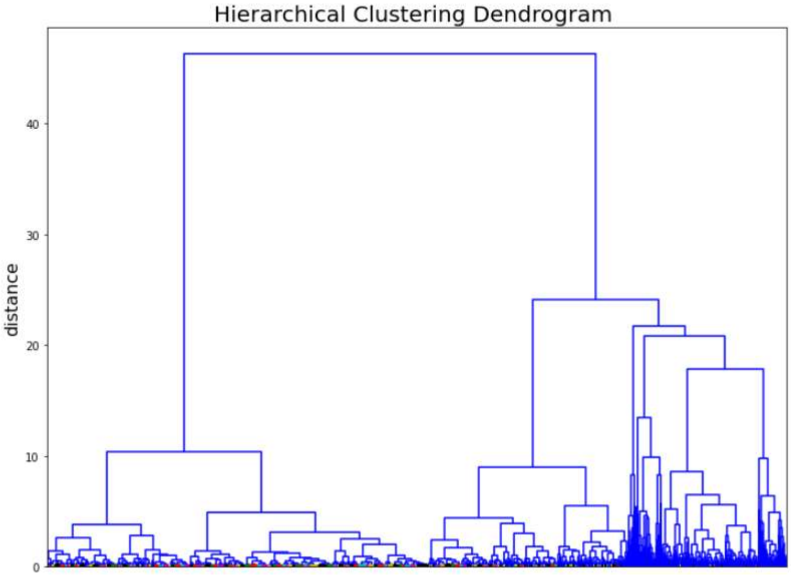


Figure 8 hierarchal cluster dendrogram based on distance from word 2 vec

We chose this method over more popular clustering types such as k means as it allowed us to select the number of clusters based on their distance. Whereas k-means would allow a set number. Furthermore, we can visualise where distance is most intrusive.

The above figure shows what our corpus produces in terms of clusters. From this we can see that they are 3 major clusters with the far right one being densely populated. However, the far one does have major discrepancies in distancing as they converge upon a higher distance. For this we decided that 19 would be out cut off as it provides suitable clusters alongside representation of words. This provided us with 5 clusters, the we used a word cloud to display words by their count and to what was actually inside them.

For fast text we chose a similar approach, of using wards variance to see where the words would converge into clusters. As we can see form the following dendrogram they are 4 clear clusters in which the words are allocated into. With addition to an even spread of words. As in the previous model there were some clusters bigger than others. The embedding technique used was the same as word 2 vec's in-terms parameters. The minimum frequency was 2, size was 100 and alpha was 0.05.

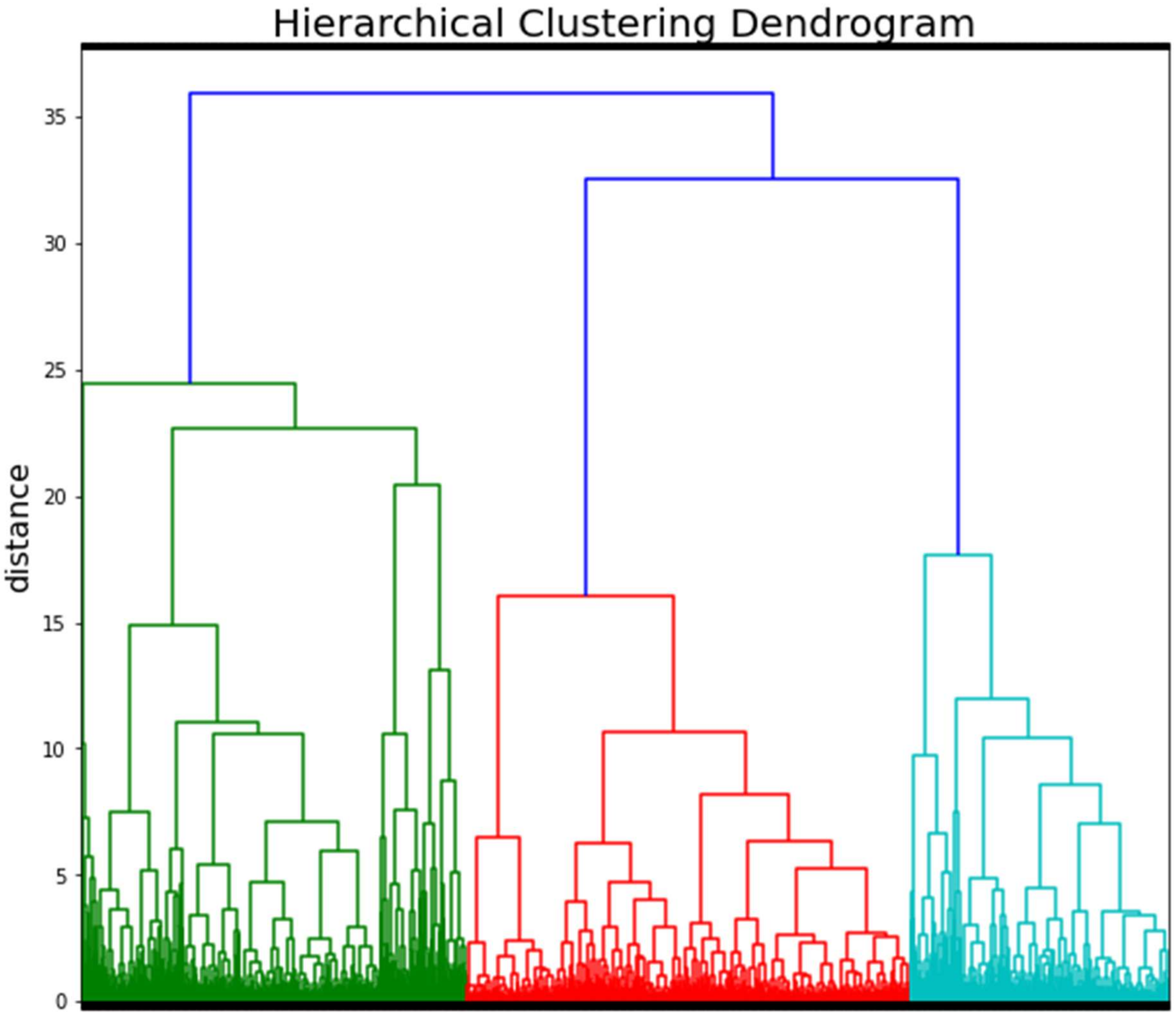


Figure 9 hierarchal cluster dendrogram based on distance from fast text

latent Dirichlet allocation

Now we had our distinct clusters we needed to see what topics could be derived from them as the main point was to associate tweets with topics in order to judge potential risk. For this we used latent Dirichlet allocation is probabilistic model of a corpus, its aim is that documents are represented as random mixtures over latent topics. Where each topic is defined by a distribution over words. [24]

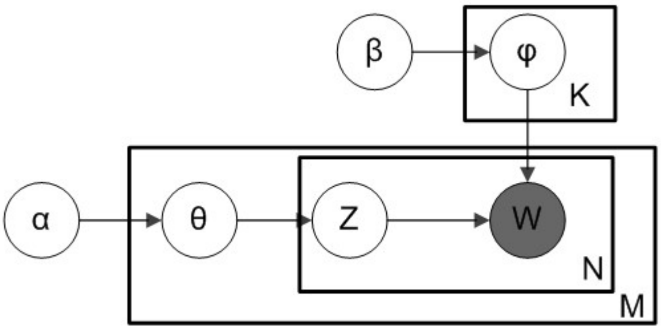


Figure 10 architecture of latent Dirichlet allocation

The figure above describes the LDA model where:

α is pre document distribution,

β is per topic word distribution,

θ is topic distribution for document m,

ϕ is the word distribution for topic k,

Z is the topic for the n-th word in document m.

W is the specific word

Topic models provide a probabilistic framework which can be used to organise, structure and understand a piece of text. A document is seen as a mixture of topics with modelling helping to discover hidden themes.

Examples of LDA in research

In the paper" A TEXT MINING RESEARCH BASED ON LDA TOPIC MODELLING" LDA topic modelling is used in order to gain research and analysis over twitter uses and interests alongside Wikipedia articles. Their pipeline is similar to the one we implemented save the use of hierarchal clustering. In doing so they found that number of topics is less than ones done of regular documentation such as Wikipedia mining due to it being formed with natural language. Next the boundaries were not as clear as Wikipedia mining as per the use of concise natural; language in a tweet. [25]

In the paper "Real Time Road Traffic Monitoring Alert based on Incremental Learning from Tweets" they aim to create a system which can alert them on traffic related problems. For the tweet dataset they used both official tweets and user generated tweets. In this they use a novel approach for classifying tweeting using LDA. In this work they did a comparison of their approach with LDA and SVM of which LDA provided a better solution. [26]

Now the clusters were established we need to derive topics from them. First, we need the optimum number of topics per cluster

Chapter 4 computational experiments and result analysis

For the measurement of our model we used 2 metrics. The first being coherence score and perplexity.

A coherent statement is one which makes sense and revolves around the same topic. So, by default words with similar meanings should occur in similar context.

Topic coherence is a measurement of the degree of semantic similarity between words in a topic. They help differentiate topics which are semantically interpretable and ones which are there by statistical inference.

Perplexity is common quality measure for language models. Its purpose is to evaluate how well a model is able to fit unseen text. Low perplexity means the model isn't surprised of the introduced text.

Next, we had to find the optimum number of topics for each cluster as this would affect the modelling and the classification to come. For this

word 2 vec

cluster 1

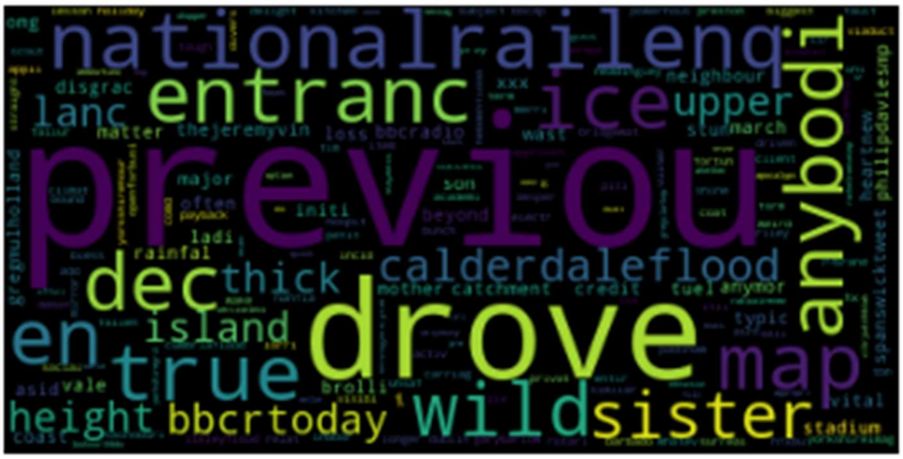
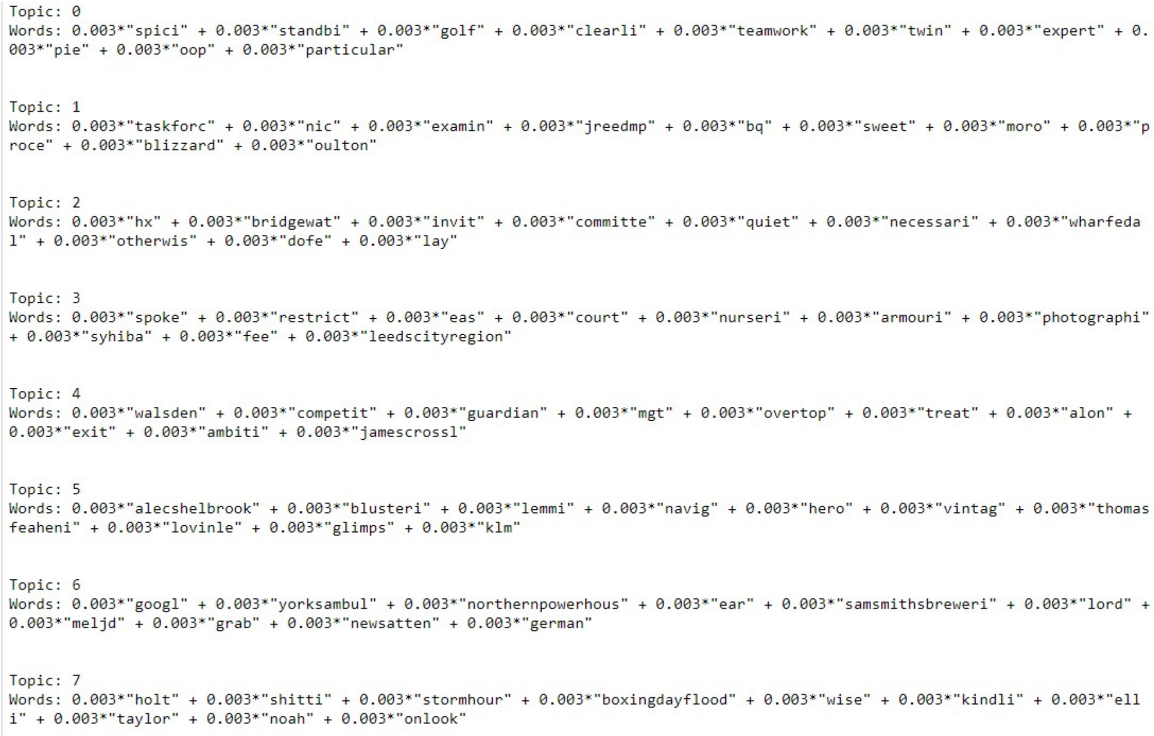


Figure 11 cluster 1 derived from dendrogram

This cluster was the biggest as it 1702 tokens. Furthermore, it contained non relatable words as seen by the word cloud such as "Scotland" and "Calderdaleflood" which via the naked eye they proposed no correlation. However as these are tweets this could have been a comparison of various places or tweets said as passing comments.

Next, we had to find the optimum number of topics based on coherence score. Our testing gave 2 or 8 with a full corpus. For this we chose 8 as the initial cluster contained 1702 tokens.

The topics we derived as such were.



```
Topic: 0
Words: 0.003*"spici" + 0.003*"standbi" + 0.003*"golf" + 0.003*"clearli" + 0.003*"teamwork" + 0.003*"twin" + 0.003*"expert" + 0.
003*"pie" + 0.003*"oop" + 0.003*"particular"


Topic: 1
Words: 0.003*"taskforc" + 0.003*"nic" + 0.003*"examin" + 0.003*"jreedmp" + 0.003*"bq" + 0.003*"sweet" + 0.003*"moro" + 0.003*"p
roce" + 0.003*"blizzard" + 0.003*"oulton"


Topic: 2
Words: 0.003*"hx" + 0.003*"bridgewat" + 0.003*"invit" + 0.003*"committe" + 0.003*"quiet" + 0.003*"necessari" + 0.003*"wharfeda
l" + 0.003*"otherwis" + 0.003*"dofe" + 0.003*"lay"


Topic: 3
Words: 0.003*"spoke" + 0.003*"restrict" + 0.003*"eas" + 0.003*"court" + 0.003*"nurseri" + 0.003*"armouri" + 0.003*"photographi"
+ 0.003*"syhiba" + 0.003*"fee" + 0.003*"leedscityregion"


Topic: 4
Words: 0.003*"walsden" + 0.003*"competit" + 0.003*"guardian" + 0.003*"mgt" + 0.003*"overtop" + 0.003*"treat" + 0.003*"alon" +
0.003*"exit" + 0.003*"ambiti" + 0.003*"jamescrossl"


Topic: 5
Words: 0.003*"alecshelbrook" + 0.003*"blusteri" + 0.003*"lemmi" + 0.003*"navig" + 0.003*"hero" + 0.003*"vintag" + 0.003*"thomas
feaheni" + 0.003*"lovinle" + 0.003*"glimps" + 0.003*"klm"


Topic: 6
Words: 0.003*"googl" + 0.003*"yorksambul" + 0.003*"northernpowerhous" + 0.003*"ear" + 0.003*"samsmithsbreweri" + 0.003*"lord" +
0.003*"meljd" + 0.003*"grab" + 0.003*"newsatten" + 0.003*"german"


Topic: 7
Words: 0.003*"holt" + 0.003*"shitti" + 0.003*"stormhour" + 0.003*"boxingdayflood" + 0.003*"wise" + 0.003*"kindli" + 0.003*"ell
i" + 0.003*"taylor" + 0.003*"noah" + 0.003*"onlook"
```

Figure 12 results from LDA topic modelling on cluster 1

From the topic analysis no, coherent topics could be made out as it displayed words which would seem semi random if it wasn't for the modelling. However, topic 3 had some coherence to it as it could be related to travel into the Leeds region as per the events/ names mentioned.

However, the coherence score we generated for this was 0.84 which would lead to a contrary belief to the above. This cluster was large which may have been the cause of us not establishing viable topics. The perplexity of this cluster was -9.90 which stated that it had a high chance of encountering words which were unfamiliar.

Cluster 2

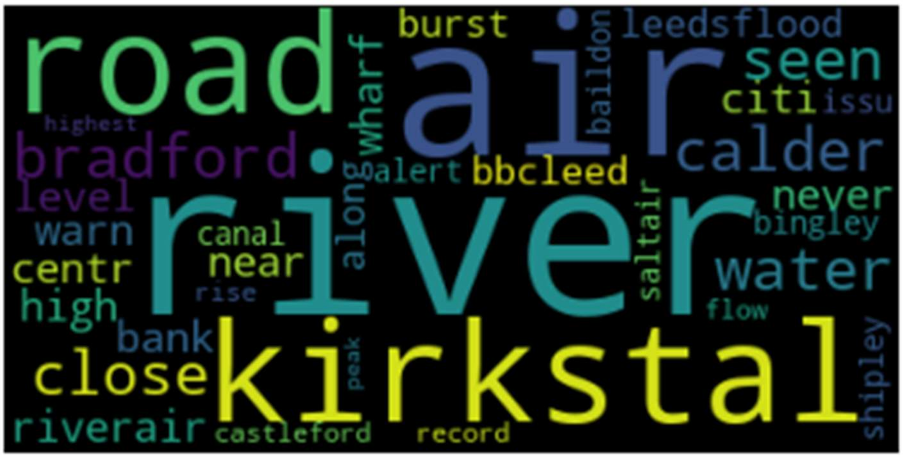
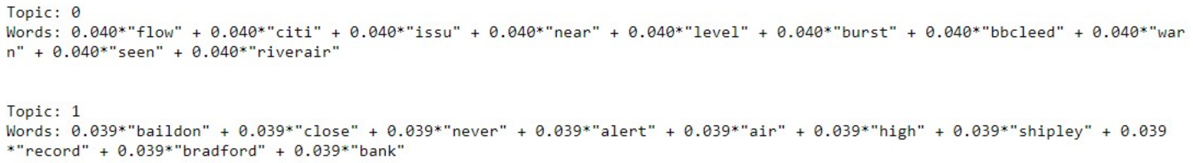


Figure 13 cluster 2 derived from dendrogram

For this cluster 37 unique tokens were found based on its distant cut-off. As seen by the word cloud, the words are highly similar to the search terms used.

For this our method gave us the same co hence score for topics 2 till 10. As this cluster only contained 37 tokens, we chose 2.



```
Topic: 0
Words: 0.040*"flow" + 0.040*"citi" + 0.040*"issu" + 0.040*"near" + 0.040*"level" + 0.040*"burst" + 0.040*"bbcleed" + 0.040*"war
n" + 0.040*"seen" + 0.040*"riverair"


Topic: 1
Words: 0.039*"baildon" + 0.039*"close" + 0.039*"never" + 0.039*"alert" + 0.039*"air" + 0.039*"high" + 0.039*"shipley" + 0.039
*"record" + 0.039*"bradford" + 0.039*"bank"
```

Figure 14 results from LDA topic modelling on cluster 2

From this we made out that both were talking about flooding however one was more orientated in Leeds and with its river system. The second could be the surrounding areas and rivers as “air” is mentioned, which may have been a stemmed version of “aire” the river. This is further supported by the names which were proposed such as “bradford” and “balidon”

- Coherence Score: 0.43
- Perplexity: -4.65

Cluster 3

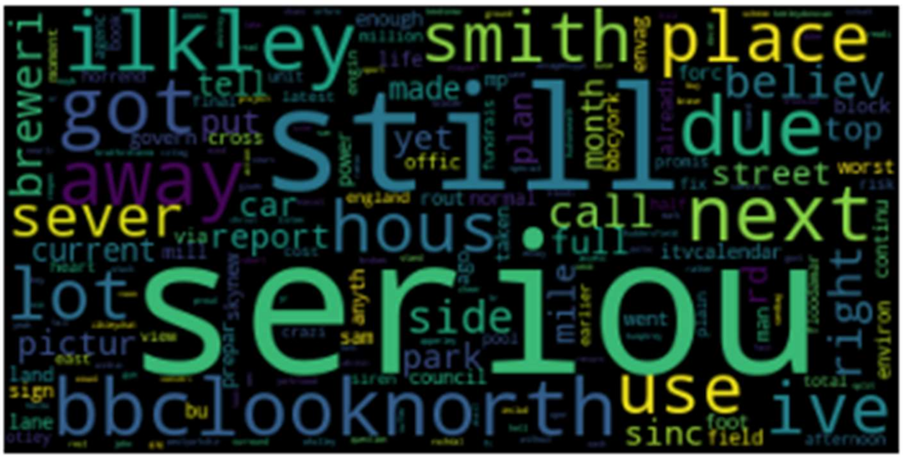
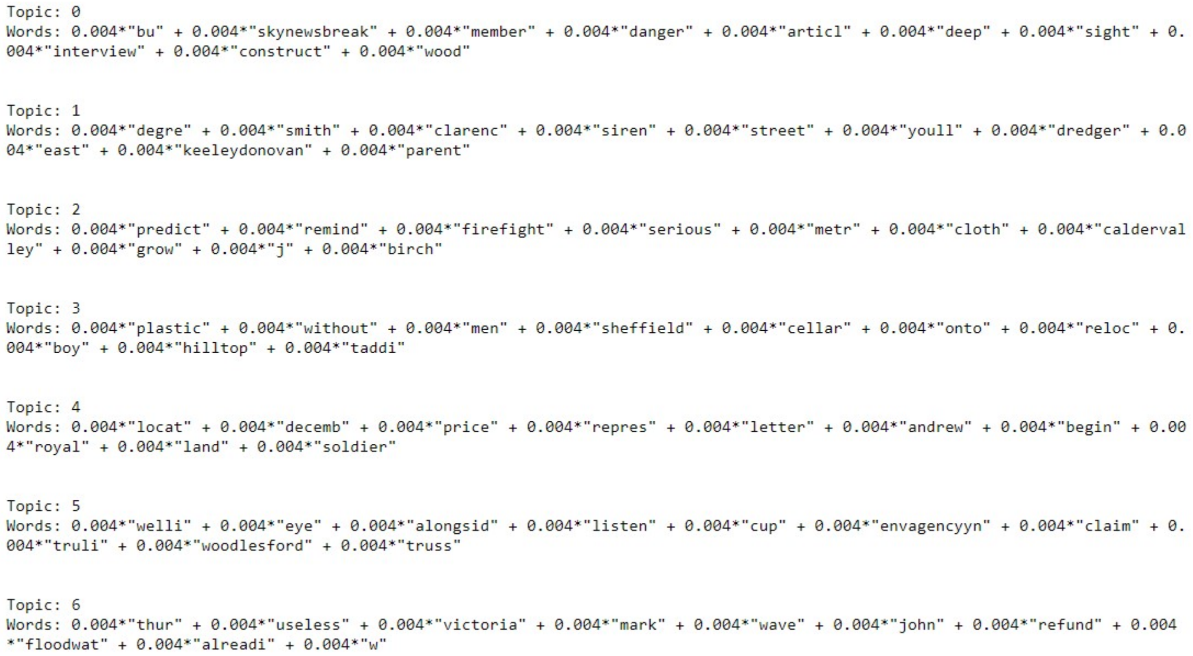


Figure 15 cluster 3 derived from dendrogram

This cluster consisted of 1026 tokens.

For this cluster, results were slightly better as produced 7 topics for this cluster. However, there was minimal difference between 6 and 10 topics



```
Topic: 0
Words: 0.004*"bu" + 0.004*"skynewsbreak" + 0.004*"member" + 0.004*"danger" + 0.004*"articl" + 0.004*"deep" + 0.004*"sight" + 0.
004*"interview" + 0.004*"construct" + 0.004*"wood"


Topic: 1
Words: 0.004*"degre" + 0.004*"smith" + 0.004*"clarenc" + 0.004*"siren" + 0.004*"street" + 0.004*"youll" + 0.004*"dredger" + 0.0
04*"east" + 0.004*"keeleydonovan" + 0.004*"parent"


Topic: 2
Words: 0.004*"predict" + 0.004*"remind" + 0.004*"firefight" + 0.004*"serious" + 0.004*"metr" + 0.004*"cloth" + 0.004*"calderval
ley" + 0.004*"grow" + 0.004*"j" + 0.004*"birch"


Topic: 3
Words: 0.004*"plastic" + 0.004*"without" + 0.004*"men" + 0.004*"sheffield" + 0.004*"cellar" + 0.004*"onto" + 0.004*"reloc" + 0.
004*"boy" + 0.004*"hilltop" + 0.004*"taddi"


Topic: 4
Words: 0.004*"locat" + 0.004*"decemb" + 0.004*"price" + 0.004*"repres" + 0.004*"letter" + 0.004*"andrew" + 0.004*"begin" + 0.00
4*"royal" + 0.004*"land" + 0.004*"soldier"


Topic: 5
Words: 0.004*"welli" + 0.004*"eye" + 0.004*"alongsid" + 0.004*"listen" + 0.004*"cup" + 0.004*"envagencyyn" + 0.004*"claim" + 0.
004*"truli" + 0.004*"woodlesford" + 0.004*"truss"


Topic: 6
Words: 0.004*"thur" + 0.004*"useless" + 0.004*"victoria" + 0.004*"mark" + 0.004*"wave" + 0.004*"john" + 0.004*"refund" + 0.004
*"floodwat" + 0.004*"alreadi" + 0.004*"w"
```

Figure 16 results from LDA topic modelling on cluster 3

Topic 0 has some instances which may indicate that it is about the news and news which is important. As seen by the key words used such as “skynewsbreak” “interview”.

Next topic 4 may represent a royal visit from price Andrew.[27] The rest are uninterpretable.

- Coherence Score: 0.81
- Perplexity: -9.26

Cluster 4

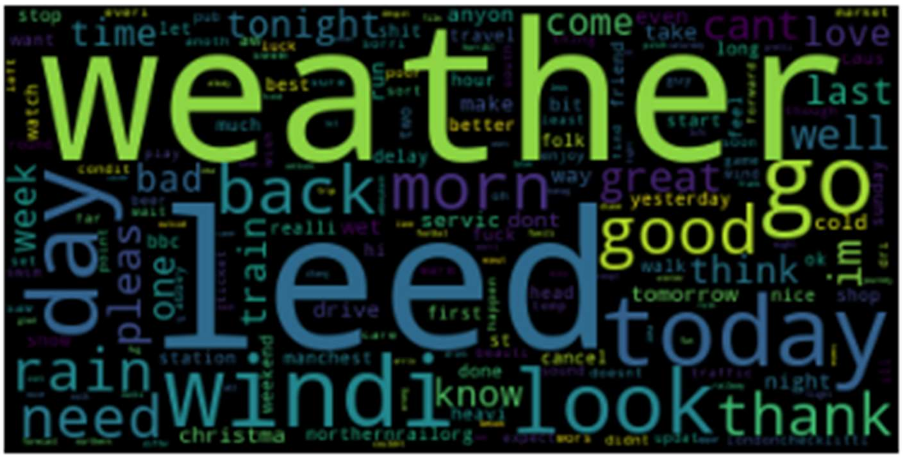
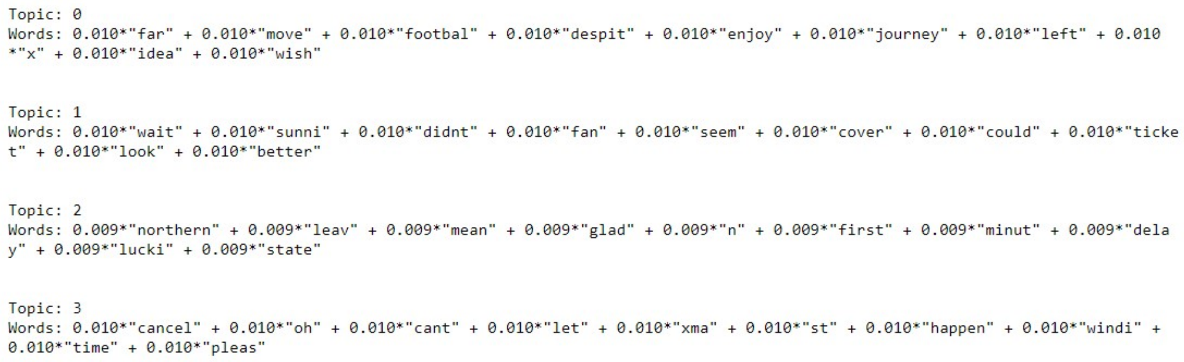


Figure 17 cluster 4 derived from dendrogram

This cluster consisted of 260 tokens of which the most frequent are shown.

This cluster was problematic as the coherence score was identical. So, the method used to derive number of topics wasn’t applicable for this cluster. However, as the size of the actual cluster was only 260 tokens, we opted for 4 topics.



```
Topic: 0
Words: 0.010*"far" + 0.010*"move" + 0.010*"footbal" + 0.010*"despit" + 0.010*"enjoy" + 0.010*"journey" + 0.010*"left" + 0.010
*"x" + 0.010*"idea" + 0.010*"wish"


Topic: 1
Words: 0.010*"wait" + 0.010*"sunni" + 0.010*"didnt" + 0.010*"fan" + 0.010*"seem" + 0.010*"cover" + 0.010*"could" + 0.010*"ticke
t" + 0.010*"look" + 0.010*"better"


Topic: 2
Words: 0.009*"northern" + 0.009*"leav" + 0.009*"mean" + 0.009*"glad" + 0.009*"n" + 0.009*"first" + 0.009*"minut" + 0.009*"dela
y" + 0.009*"lucki" + 0.009*"state"


Topic: 3
Words: 0.010*"cancel" + 0.010*"oh" + 0.010*"cant" + 0.010*"let" + 0.010*"xma" + 0.010*"st" + 0.010*"happen" + 0.010*"windi" +
0.010*"time" + 0.010*"pleas"
```

Figure 18 results from LDA topic modelling on cluster 4

The topics we found were as following

Topic 0 infers travel to football or events surrounding it due to the words such as “far”, “journey” and “footbal”

Topic 2 may infer train travel as “northen” and “first” are train operators with words such as “delay” and “leav” to support it.

The rest are uninterpretable.

- Coherence Score: 0.68
- Perplexity: -7.33

Cluster 5

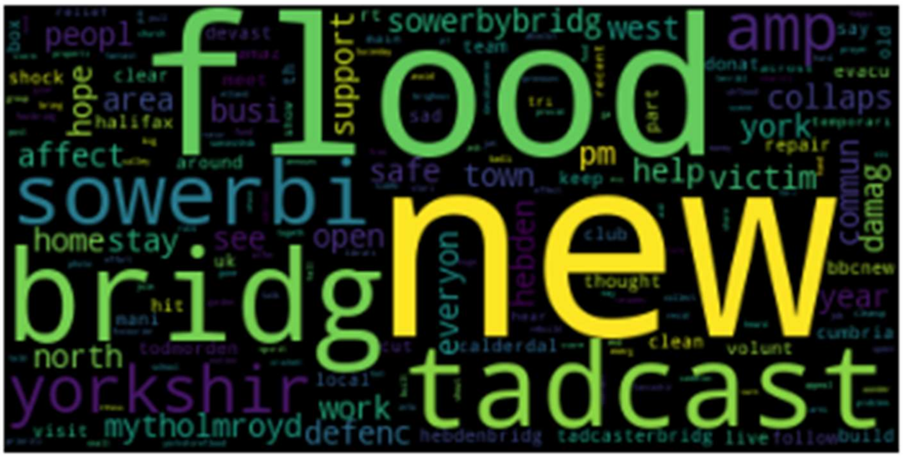
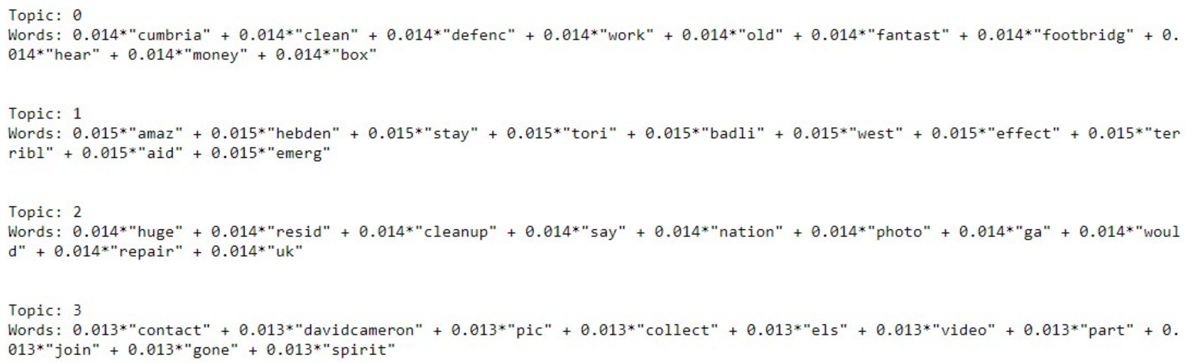


Figure 19 cluster 5 derived from dendrogram

This cluster had 182 tokens

This cluster like cluster 4 didn’t have an optimum number of topics as the coherence score was same for all of the them. So, we chose 4 topics like before.



```
Topic: 0
Words: 0.014*"cumbria" + 0.014*"clean" + 0.014*"defenc" + 0.014*"work" + 0.014*"old" + 0.014*"fantast" + 0.014*"footbridg" + 0.
014*"hear" + 0.014*"money" + 0.014*"box"


Topic: 1
Words: 0.015*"amaz" + 0.015*"hebden" + 0.015*"stay" + 0.015*"tori" + 0.015*"badli" + 0.015*"west" + 0.015*"effect" + 0.015*"ter
ribl" + 0.015*"aid" + 0.015*"emerg"


Topic: 2
Words: 0.014*"huge" + 0.014*"resid" + 0.014*"cleanup" + 0.014*"say" + 0.014*"nation" + 0.014*"photo" + 0.014*"ga" + 0.014*"woul
d" + 0.014*"repair" + 0.014*"uk"


Topic: 3
Words: 0.013*"contact" + 0.013*"davidcameron" + 0.013*"pic" + 0.013*"collect" + 0.013*"els" + 0.013*"video" + 0.013*"part" + 0.
013*"join" + 0.013*"gone" + 0.013*"spirit"
```

Figure 20 ,results from LDA topic modelling on cluster 5

Topic 0 may be aid which was received in the area due the majority of words in the topic which are prevalent.

Topic 1 Is unclear as what it is about.

Topic 2 could be about damage and damage control as they include words such as “cleanup” and “repair” with words such as “resid” and “nation” to support as they involve a place or community Topic 3 has clear connotations of a political view as it mentions the PM at the time alongside media elements such as “video” and “pic” which is short for picture in slang.

- Coherence Score: 0.66
- Perplexity: -6.96

In the process we did manage to find topics which are associated with the clusters. However, in some cases these topics were not visible by the words alone. Which does not necessarily mean that they don’t exist as seen in cluster 3 topic 4. With some research in the words chosen we found that there was an event which occurred during the floods.

Evaluation

Overall, in terms of coherence and perplexity the more words a cluster has the better coherence score. This is shown as the cluster with 37 tokens has the lowest score and the cluster with 1702 has the highest. Perplexity follows the same rules as more tokens provide less chance of model predicting the words or being surprised by them.

Fast text

Cluster 1

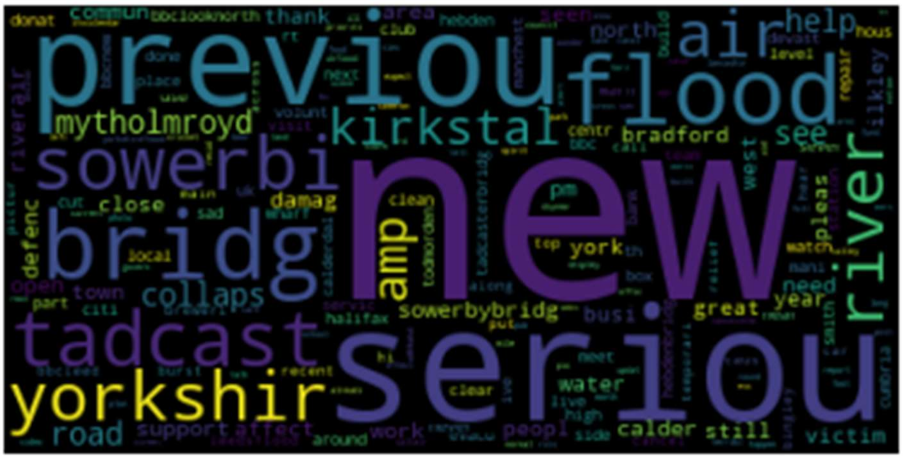


Figure 21 cluster 1 of fast text

This cluster had 1137 tokens, the coherence score was 0.79 and perplexity was Perplexity: -9.21

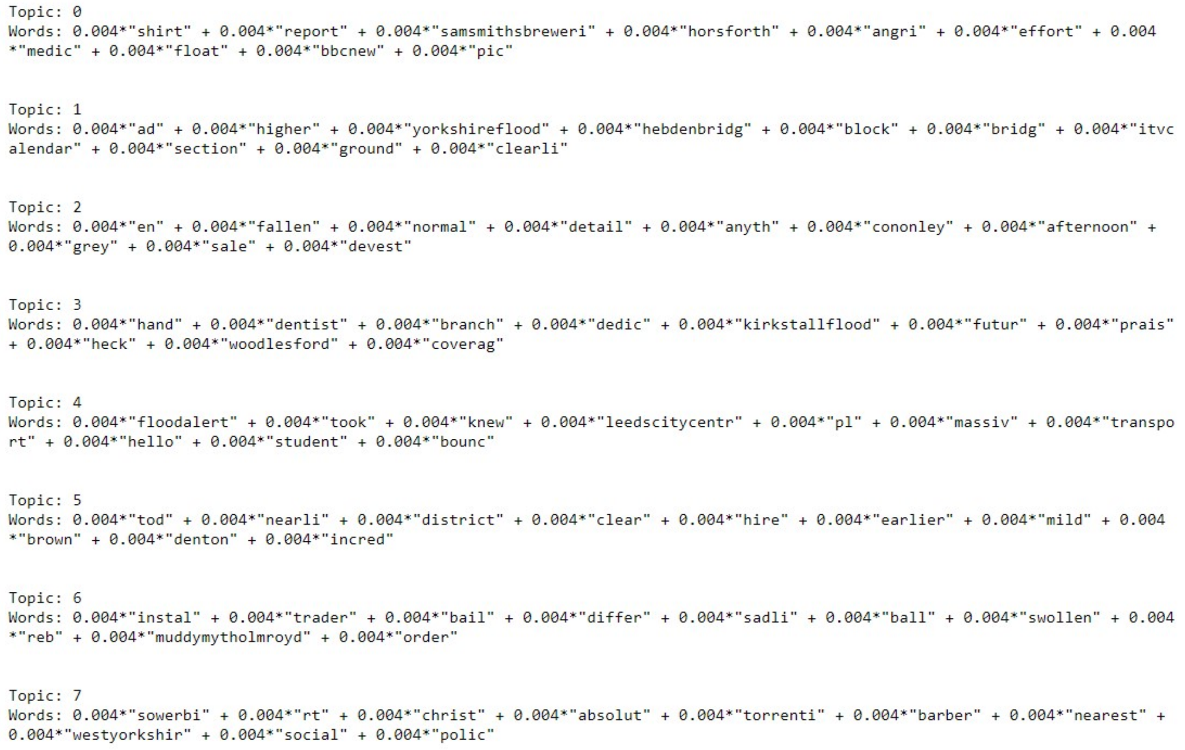

```
Topic: 0
Words: 0.004*"shirt" + 0.004*"report" + 0.004*"samsmithsbreweri" + 0.004*"horsforth" + 0.004*"angri" + 0.004*"effort" + 0.004
*"medic" + 0.004*"float" + 0.004*"bbcnew" + 0.004*"pic"


Topic: 1
Words: 0.004*"ad" + 0.004*"higher" + 0.004*"yorkshireflood" + 0.004*"hebdenbridg" + 0.004*"block" + 0.004*"bridg" + 0.004*"itvc
alendar" + 0.004*"section" + 0.004*"ground" + 0.004*"clearli"


Topic: 2
Words: 0.004*"en" + 0.004*"fallen" + 0.004*"normal" + 0.004*"detail" + 0.004*"anyth" + 0.004*"cononley" + 0.004*"afternoon" +
0.004*"grey" + 0.004*"sale" + 0.004*"devest"


Topic: 3
Words: 0.004*"hand" + 0.004*"dentist" + 0.004*"branch" + 0.004*"dedic" + 0.004*"kirkstallflood" + 0.004*"futur" + 0.004*"prais"
+ 0.004*"heck" + 0.004*"woodlesford" + 0.004*"coverag"


Topic: 4
Words: 0.004*"floodalert" + 0.004*"took" + 0.004*"knew" + 0.004*"leedscitycentr" + 0.004*"pl" + 0.004*"massiv" + 0.004*"transpo
rt" + 0.004*"hello" + 0.004*"student" + 0.004*"bounc"


Topic: 5
Words: 0.004*"tod" + 0.004*"nearli" + 0.004*"district" + 0.004*"clear" + 0.004*"hire" + 0.004*"earlier" + 0.004*"mild" + 0.004
*"brown" + 0.004*"denton" + 0.004*"incred"


Topic: 6
Words: 0.004*"instal" + 0.004*"trader" + 0.004*"bail" + 0.004*"differ" + 0.004*"sadli" + 0.004*"ball" + 0.004*"swollen" + 0.004
*"reb" + 0.004*"muddymytholmroyd" + 0.004*"order"


Topic: 7
Words: 0.004*"sowerbi" + 0.004*"rt" + 0.004*"christ" + 0.004*"absolut" + 0.004*"torrenti" + 0.004*"barber" + 0.004*"nearest" +
0.004*"westyorkshir" + 0.004*"social" + 0.004*"polic"
```

Figure 22 results from LDA topic modelling on cluster 1

Looking at the content of the topics which the LDA model produced we can see that there is no clear understanding or what they represent in terms of a topic. As there is no coherent structure to them.

Cluster 2

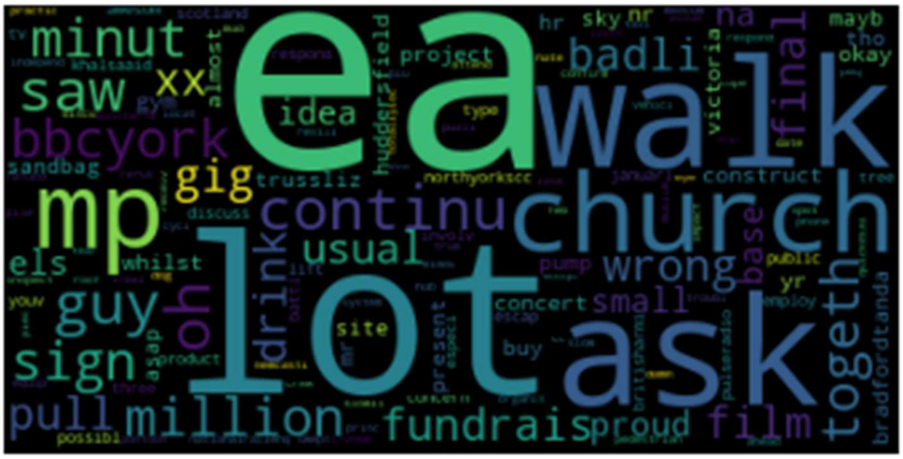
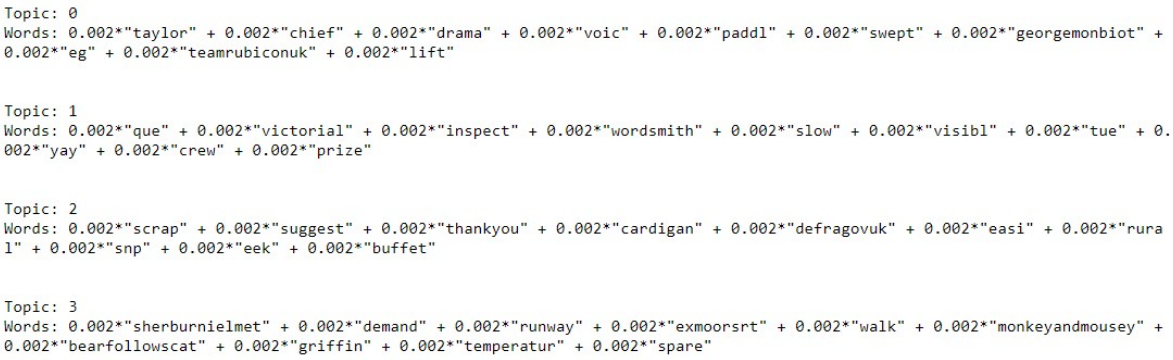


Figure 23 cluster 2 of fast text

This cluster had 1310 tokens the coherence was 0.83 and the perplexity was -9.64



```
Topic: 0
Words: 0.002*"taylor" + 0.002*"chief" + 0.002*"drama" + 0.002*"voic" + 0.002*"paddl" + 0.002*"swept" + 0.002*"georgemonbiot" +
0.002*"eg" + 0.002*"teamrubiconuk" + 0.002*"lift"


Topic: 1
Words: 0.002*"que" + 0.002*"victorial" + 0.002*"inspect" + 0.002*"wordsmith" + 0.002*"slow" + 0.002*"visibl" + 0.002*"tue" + 0.
002*"yay" + 0.002*"crew" + 0.002*"prize"


Topic: 2
Words: 0.002*"scrap" + 0.002*"suggest" + 0.002*"thankyou" + 0.002*"cardigan" + 0.002*"defragovuk" + 0.002*"easi" + 0.002*"rura
l" + 0.002*"snp" + 0.002*"eek" + 0.002*"buffet"


Topic: 3
Words: 0.002*"sherburnielmet" + 0.002*"demand" + 0.002*"runway" + 0.002*"exmoorsrt" + 0.002*"walk" + 0.002*"monkeyandmousey" +
0.002*"bearfollowscat" + 0.002*"griffin" + 0.002*"temperatur" + 0.002*"spare"
```

Figure 24 results from LDA topic modelling on cluster 2

For this cluster we found out the optimum number of topics was 4. As before no logical topics could be derived from the above allocation.

Cluster 3

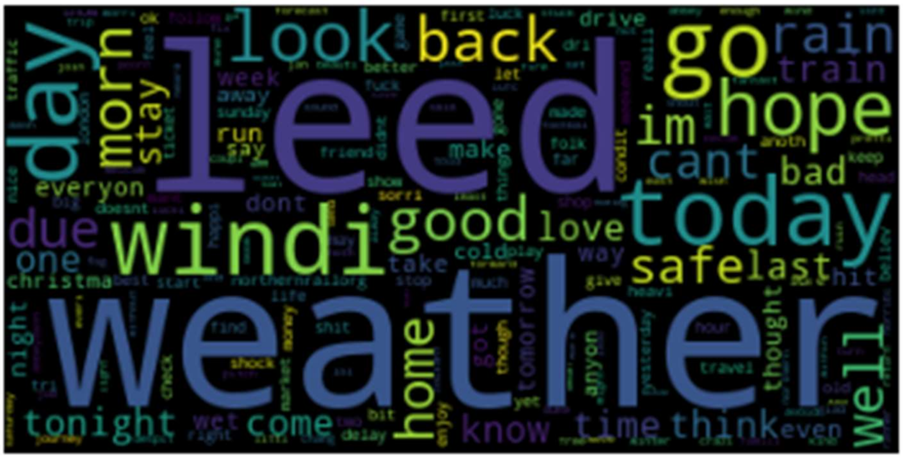
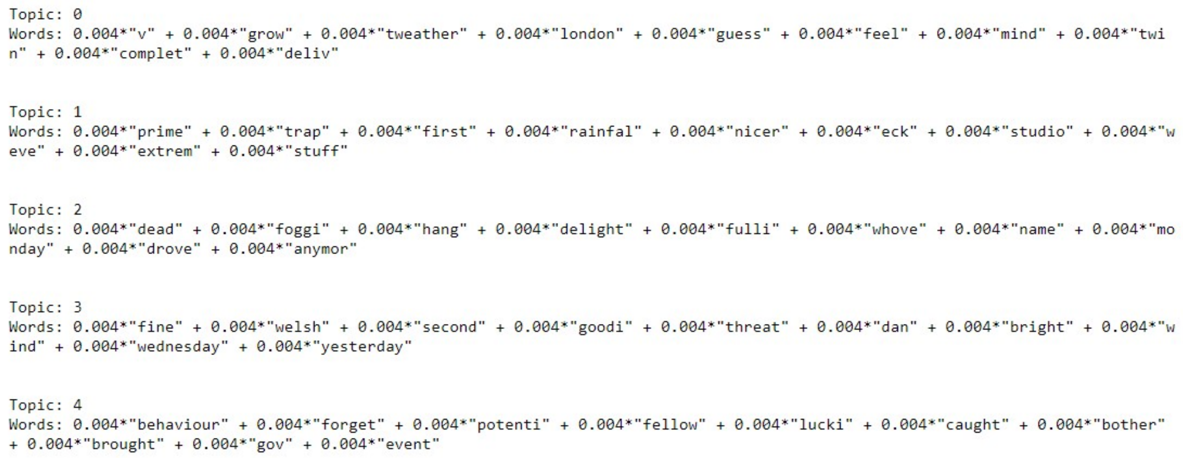


Figure 25 cluster 3 of fast text

This cluster had 760 tokens the coherence value was 0.79and the perplexity was -8.62



```
Topic: 0
Words: 0.004*"v" + 0.004*"grow" + 0.004*"tweather" + 0.004*"london" + 0.004*"guess" + 0.004*"feel" + 0.004*"mind" + 0.004*"twi
n" + 0.004*"complet" + 0.004*"deliv"


Topic: 1
Words: 0.004*"prime" + 0.004*"trap" + 0.004*"first" + 0.004*"rainfal" + 0.004*"nicer" + 0.004*"eck" + 0.004*"studio" + 0.004*"w
eve" + 0.004*"extrem" + 0.004*"stuff"


Topic: 2
Words: 0.004*"dead" + 0.004*"foggi" + 0.004*"hang" + 0.004*"delight" + 0.004*"fulli" + 0.004*"whove" + 0.004*"name" + 0.004*"mo
nday" + 0.004*"drove" + 0.004*"anymor"


Topic: 3
Words: 0.004*"fine" + 0.004*"welsh" + 0.004*"second" + 0.004*"goodi" + 0.004*"threat" + 0.004*"dan" + 0.004*"bright" + 0.004*"w
ind" + 0.004*"wednesday" + 0.004*"yesterday"


Topic: 4
Words: 0.004*"behaviour" + 0.004*"forget" + 0.004*"potenti" + 0.004*"fellow" + 0.004*"lucki" + 0.004*"caught" + 0.004*"bother"
+ 0.004*"brought" + 0.004*"gov" + 0.004*"event"
```

Figure 26 results from LDA topic modelling on cluster 3

For this cluster we could not make out any meaning to these topics.

Testing and analysis

Now we have established our clusters and their topics we can use the LDA model to associate a tweet to a topic. In doing so we can see what the tweet represents and what that tweet is most similar to. In order to do this the tweet must be processing in the same way as the data set.

Next as there was no way of classifying a tweet to one of the original clusters, we were forced to use all of the clusters in the test. This was rather problematic as increased similarity score does not mean that the tweet belongs in that cluster or that topic, however we can measure the success by seeing the topics we manually created during topic naming.

The number of topics per cluster were allocated by traversing through the alpha, beta and topic number to find the highest coherence per cluster based on topic count.

The results of word 2 vec are seen in the appendix

Score

Score is the numerical value which represents to which topic the given testing document is assigned to, based on the LDA model. The higher the score the more likely the document is classed with that topic Based on the conjunction of words from the derived topic and the new text.

Word 2 vec

Test 1

The original was "What a beautiful contrast to the rain and floods of Boxing Day. A great run above Carleton.#Yorkshire #Skipton"

The processed version was ['beauti', 'contrast', 'rain', 'flood', 'box', 'day', 'great', 'run', 'carletonyorkshir', 'skipton']

Cluster	1	2	3	4	5
Results of topic allocation	0.37297362 -1 0.089590244 -7	0.5- 1 0.5- 1	0.4100149- 6 0.09834336-5	0.037987113- 2 0.03774897- 1	0.71153885- 2 09616617- 3

For cluster 1 as no coherent topics were formed, we could not assess the success. cluster 2 was split evenly, this means that the tweet could belong in both topics. In cluster 3 there is a clear association to topic 6. In cluster 4 the topic the tweet is most assigned to is topic 2, which by out extrapolation of potential label would make sense as the tweet regards travel within the given bad conditions. However, on cluster 5 the tweet is most associated with topic 2 with our analysis we concluded that this topic was about repairing, this is difficult to say if this was related or not.

Test 2

The original tweet was 'Live in Leeds, need to get a train to Bradford to get to Leeds #floods']

The processed tweet was ['live','leed', 'need', 'get', 'train', 'bradford', 'get', 'leed', 'flood']

Cluster	1	2	3	4	5
Results of topic allocation	0.125-0 0.125-1 0.125-2 0.125-3 0.125-4 0.125-5 0.125-6 0.125-7	0.80775726-1 0.1922427-0	0.14285713-0 0.14285713-1 0.14285713-2 0.14285713-3 0.14285713-4 0.14285713-5 0.14285713-6	0.6087078-3 0.3048989-2	0.40524343-1 0.40125823-0

In cluster 1 the test proved score test proved inconclusive as all were the same. In cluster 2 topic 1 was most prevalent which according to our analysis this topic was related to flooding around Leeds. Judging by the test tweet it could belong in both topics. However, the use of language such as "bradford" may have skewed the classification. Cluster 3 was inconclusive. Cluster 4 gave preference to topic 4 however this was uninterpretable from the given words. Cluster 5 gave classification to topic 1 however this topic was uninterpretable. However, topic 0 was second by 0.004 and this topic we assessed as" aid" to the region. Which was not what the tweet was proposing.

Test 3

The original tweet was 'Sad to report that the traffic chaos in #Kirkstall has returned to pre #flood levels'

The processed version of the tweet was ['sad', 'report', 'traffic', 'chao', 'kirkstal','return', 'pre', 'flood', 'level']

Cluster	1	2	3	4	5
Results of topic allocation	0.37293267-2 0.08958894-1	0.88087076-0 0.11912923-1	0.25371733-4 0.25195345-2 0.25075743-3 0.060917296-0	0.5826326-2 0.13915674-0	0.7115365-1 0.0961769-3

This test tweet was different from the previous as it provided a positive light on the floods. Never the less it was tested and in cluster 1 topic 2 was what the tweet was associated with and without topic naming we could not evaluate this. Next for cluster 2 topic 0 was the best. In this there is a relation as this topic is about flood events outside the city of Leeds including Kirkstall. However, this is an observation. In cluster 3 it was topic 4 with 2&3 within 0.004 of it. However, topic 4 is about a royal visit not traffic problems without having to speculate. For cluster 4 it was topic 2 which was about travel however a connection could be made with the two as they both inferred travel problems. For cluster 5 it was topic 1 which didn't have a topic assigned as we could not interpret it.

Fast text

Test 1

The same set of tweets were used in testing the potential allocation.

	Cluster 1	Cluster 2	Cluster 3
Results of topic allocation	0.2812548 5 0.2812445 2 0.28122044 6 0.03125606 7 0.031256057 4 0.031256054 0 0.031256054 3 0.03125605 1	0.25 0 0.25 1 0.25 2 0.25 3	0.4610268 2 0.31423876 4 0.16754885 3 0.028592985 1 0.028592601 0

From test 1 we could see that there isn't much different in score from regarding the allocation of the tweet in cluster 1. Furthermore, with no substance to each of the produce topics there isn't a clear topic allocation. For cluster 2 there wasn't a clear choice as all 4 topics returned the same score.

Test 2

	Cluster 1	Cluster 2	Cluster 3
Results of topic allocation	0.4252113 0 0.22489591 7 0.2248697 2 0.02500462 5 0.025004618 4 0.025004614 3 0.025004614 6 0.02500461 1	0.25 0 0.25 1 0.25 2 0.25 3	0.86663777 3 0.033340637 2 0.03334058 1 0.033340562 4 0.033340454 0

From test 2 we can see in cluster 1 this tweet would be allocated topic 0. For cluster 2 there wasn't a clear allocation as they returned the same values.

Test 3

Cluster	Cluster 1	Cluster 2	Cluster 3
Results of topic allocation	0.30375168 0 0.30374125 3 0.16055673 2 0.16050327 1 0.017861767 7 0.017861763 4	0.62445325 3 0.12518322 2 0.12518261 1 0.12518093 0	0.7332671 4 0.06668378 3 0.066683225 2 0.06668309 1 0.06668279 0

	0.017861763 5 0.017861761 6		

From test 3 we can see there is the tweet could be allocated to either topic 0 or 3. For cluster 2 the tweet would be most likely to appear in topic 3.

Discussion

During development up to this stage there was no way to see which was the correct cluster for the test tweet and subsequently which was the correct topic. For this a secondary layer of classification would be necessary to associate tweet with cluster in order to thin the process. Currently we have many possible topic allocations for the tweet and this layer would help narrow them down to a topic set produced by one cluster. Thus, assigning a label of content to the tweet through inference.

The aim of this project was to assign a label to a tweet which gave information about the content of the tweet and what category it could be placed in based on context. For this we designed a system from the bottom up which allowed us to take a tweet and process it until we could associate a "topic" with it.

The first part was selecting tweets. In this only roughly 5244 tweets were found after processing them the number of useable tokens were roughly 3207 which was a huge loss as the total processed tokens produced were roughly 8100.

The reason for this was to ensure that correct words were being used during embedding and misspelt words did not affect where the words lie in relation to each other. For this section an improvement would be to increase the dataset size so that any loss of tokens would be minor.

Next was the embedding systems. For this we used 3 distinct during the development we implemented all three but as we moved down the pipeline doc2vec could not be used as out method diverted into a word-based system. The end of development on this method came at the hierarchal clustering as not all of the documents were being displayed. Even though the same libraries were used across the 3. Our method was not suitable for sentences rather words by themselves.

However fast text and word2vec were both completed to the end as they used a word system instead of a document system.

In terms of development as the fast text system used the same methods as the word2vec one. we could interchange code blocks which made development faster.

The use of LDA determined what topics existed in clusters formed was driven by the paper

" Combining Latent Dirichlet Allocation and K-Means for Documents Clustering: Effect of Probabilistic Based Distance Measures" in which they use both LDA and K-Means for better results than LDA with naïve bayers and SVM. [28] From this we gained assurance that this method could work. Naturally we substituted K-means for our hierarchal approach.

Conclusion

Twitter is a microblogging website which allows users to state whatever they want via their account. Having said this, we researched usage of this feature during crisis and public related events. During this we found many works which used tweets as a monitoring system or extraction system.

We also found out that the publics tweets are more likely to be genuine during these events. This led to the motivation to develop our algorithms. we aimed to create a program which would enable us to retrieve tweets which were then processed in order to be suitable for NLP vectorisation. We initially started with 3 embedding models but only 2 of them made it to our current stage in pipeline.

Furthermore, the system of classification we implemented judging by our results worked but had to be improved in regards to topic mapping and pre LDA classification. We found it difficult to assign topics in some clusters and some topics were overlapping in certain clusters.

Overall, we came shy of our goal due to missing a classification layer but reached a model with high potential.

Further work.

The first and foremost thing would be to implement the tweet classification layer at the clustering stage. With this having been completed a first complete model would be done. Along with testing the fast-text model was created but not yet tested due to time constraints.

Second, but it is not vital to this study is to implement a better testing system than our current one of manually testing tweets. Additional testing data would provide us with a better estimate on the success of this process along with more tests as we did not conduct sufficient tests along each stage. Moreover, a larger dataset would be better in order to account for more range inn tweets.

Next would be an automatic allocation system, which once a tweet has been given a topic it would place both in a data set. This would create a dataset of tweets with a viable label which would be extremally useful as supervised learning algorithms require a label in order to function.

References

[1] Dr. Paola Sakai (2016). *Economic Impact Assessment of the Boxing Day Floods (2015) on SMEs in the Borough of Calderdale Final report*. [online] Available at: https://www.see.leeds.ac.uk/uploads/media/Economic_Impact_Assessment_of_Boxing_Day_floods.pdf.

[2] BBC News. 2020. *Storms Flooded 16,000 Homes In England*. [online] Available at: <https://www.bbc.co.uk/news/uk-35235502> [Accessed 25 August 2020].

[3] Dr. Paola Sakai (2016). *Economic Impact Assessment of the Boxing Day Floods (2015) on SMEs in the Borough of Calderdale Final report*. [online] Available at: <https://www.see.leeds.ac.uk/uploads/media/Economic_Impact_Assessment_of_Boxing_Day_floods.pdf.> [Accessed 25 Aug. 2020].

[4] Neoh Siew Ping, Uta Wehn, Zevenbergen, C. and van (2016). *Towards two-way flood risk communication: Current practice in a community in the UK*. [online] ResearchGate. Available at: <https://www.researchgate.net/publication/303737420_Towards_two-way_flood_risk_communication_Current_practice_in_a_community_in_the_UK >[Accessed 25 Aug. 2020].

[5] Forrest, S., Trell, E. and Woltjer, J. (2018). Civil society contributions to local level flood resilience: Before, during and after the 2015 Boxing Day floods in the Upper Calder Valley. *Transactions of the Institute of British Geographers*, [online] 44(2), pp.422–436. Available at: <https://rgs-ibg.onlinelibrary.wiley.com/doi/pdf/10.1111/tran.12279 >[Accessed 25 Aug. 2020].

[6] Lindsay, B.R. (2011). Social Media and Disasters: Current Uses, Future Options, and Policy Considerations. *UNT Digital Library*. [online] Available at: < https://digital.library.unt.edu/ark:/67531/metadc93902/ >[Accessed 25 Aug. 2020].

[7] Bairong Wang and Jun Zhuang (2017). *Crisis information distribution on Twitter: a content analysis of tweets during Hurricane Sandy*. [online] ResearchGate. Available at: <https://www.researchgate.net/publication/318154420_Crisis_information_distribution_on_Twitter_a_content_analysis_of_tweets_during_Hurricane_Sandy >[Accessed 25 Aug. 2020].

[8] Zahra Ashktorab, C. Brown, Manojit Nandi and A. Culotta (2014). *Tweedr: Mining twitter to inform disaster response*. [online] undefined. Available at: https://www.semanticscholar.org/paper/Tweedr%3A-Mining-twitter-to-inform-disaster-response-Ashktorab-Brown/a92eb50c300c77431d69be6a6644280bb4e2e63d [Accessed 25 Aug. 2020].

[9] D'Andrea, E., Pietro Ducange, Lazzerini, B. and Marcelloni, F. (2015). *Real-Time Detection of Traffic From Twitter Stream Analysis*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/273912602_Real-Time_Detection_of_Traffic_From_Twitter_Stream_Analysis [Accessed 25 Aug. 2020].

[10] Bian, J., Topaloglu, U. and Yu, F. (2012). Towards large-scale twitter mining for drug-related adverse events. *Proceedings of the 2012 international workshop on Smart health and wellbeing - SHB '12*. [online] Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5619871/ [Accessed 25 Aug. 2020].

[11] Crooks, A., Croitoru, A., Stefanidis, A. and Radzikowski, J. (2012). #Earthquake: Twitter as a Distributed Sensor System. *Transactions in GIS*, [online] 17(1), pp.124–147. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9671.2012.01359.x [Accessed 25 Aug. 2020].

[12] Nayomi Kankanamge, Tan Yigitcanlar, Ashantha Goonetilleke and Md. Kamruzzaman (2019). *Determining disaster severity through social media analysis: Testing the methodology with South East...* [online] ResearchGate. Available at: https://www.researchgate.net/publication/336585513_Determining_disaster_severity_through_social_media_analysis_Testing_the_methodology_with_South_East_Queensland_Flood_tweets [Accessed 17 Sep. 2020].

[13] Hearst, M. (2003). *What Is Text Mining? What is text mining? What are its potential applications and limitations? The difference between regular data mining and text mining is that in text mining the patterns are extracted from natural language text rather than*. [online] Available at: https://www.jaist.ac.jp/~bao/MOT-Ishikawa/FurtherReadingNo1.pdf.

[14] Liddy, E. (n.d.). *Natural Language Processing Recommended Citation*. [online] *Center for Natural Language Processing School of Information Studies (iSchool)*, p.2001. Available at: https://surface.syr.edu/cgi/viewcontent.cgi?referer=https://scholar.google.co.uk/&httpsredir=1&article=1019&context=cnlp [Accessed 25 Aug. 2020].

[15] Ducato, R. and Strowel, A. (2019). Limitations to Text and Data Mining and Consumer Empowerment: Making the Case for a Right to "Machine Legibility." *IIC - International Review of Intellectual Property and Competition Law*, [online] 50(6), pp.649–684. Available at: https://link.springer.com/article/10.1007/s40319-019-00833-w [Accessed 25 Aug. 2020].

[16] Ling, W., Dyer, C., Black, A. and Trancoso, I. (2015). *Two/Too Simple Adaptations of Word2Vec for Syntax Problems*. [online] Association for Computational Linguistics, pp.1299–1304. Available at: https://www.aclweb.org/anthology/N15-1142.pdf [Accessed 20 Aug. 2020].

[17] Jang, B., Kim, I. and Kim, J.W. (2019). Word2vec convolutional neural networks for classification of news articles and tweets. *PLOS ONE*, [online] 14(8), p.e0220976. Available at: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0220976 [Accessed 20 Aug. 2020].

[18] De Gruyter Open (2018). *Fig. 4. The CBOW and Skip-gram architectures (from [15])*. [online] ResearchGate. Available at: https://www.researchgate.net/figure/The-CBOW-and-Skip-gram-architectures-from-15_fig4_324014399 [Accessed 20 Aug. 2020].

[19] Athiwaratkun, B., Wilson, A. and Anandkumar, A. (n.d.). *Probabilistic FastText for Multi-Sense Word Embeddings*. [online] Available at: https://arxiv.org/pdf/1806.02901.pdf.

[20] Lau, J. and Baldwin, T. (n.d.). *An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation*. [online] Available at: https://arxiv.org/pdf/1607.05368.pdf?source=post_page---------------------------.

[21] Le, Q. and Mikolov, T. (2014). *Distributed Representations of Sentences and Documents*. [online] Available at: https://arxiv.org/pdf/1405.4053.pdf.

[22] Zheng, Y., Cheng, X., Huang, R. and Man, Y. (2006). A Comparative Study on Text Clustering Methods. *Advanced Data Mining and Applications*, [online] pp.644–651. Available at: https://link.springer.com/chapter/10.1007/11811305_71 [Accessed 3 Sep. 2020].

[23] Suresh, H. and Gladston Raj S. (2016). An unsupervised fuzzy clustering method for twitter sentiment analysis. *2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*. [online] Available at: https://ieeexplore.ieee.org/abstract/document/7779444 [Accessed 3 Sep. 2020].

[24] Blei, D., Edu, B., Ng, A., Jordan, M. and Edu, J. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, [online] 3, pp.993–1022. Available at: http://www.cse.cuhk.edu.hk/irwin.king/_media/presentations/latent_dirichlet_allocation.pdf.

[25] Zhou Tong and Haiyi Zhang (2016). *A Text Mining Research Based on LDA Topic Modelling*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/303563965_A_Text_Mining_Research_Based_on_LDA_Topic_Modelling [Accessed 6 Sep. 2020].

[26] Wang, D., Al-Rubaie, A., Davies, J. and Clarke, S.S. (2014). Real time road traffic monitoring alert based on incremental learning from tweets. *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*. [online] Available at: https://ieeexplore.ieee.org/abstract/document/7009503 [Accessed 6 Sep. 2020].

[27] BBC News (2016). *Duke of York “saddened” by flood-hit Tadcaster bridge*. [online] BBC News. Available at: https://www.bbc.co.uk/news/uk-england-york-north-yorkshire-35258420 [Accessed 8 Sep. 2020].

[28] Bui, Q.V., Sayadi, K., Amor, S.B. and Bui, M. (2017). Combining Latent Dirichlet Allocation and K-Means for Documents Clustering: Effect of Probabilistic Based Distance Measures. *Intelligent Information and Database Systems*, [online] pp.248–257. Available at: https://link.springer.com/chapter/10.1007/978-3-319-54472-4_24 [Accessed 9 Sep. 2020].