THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

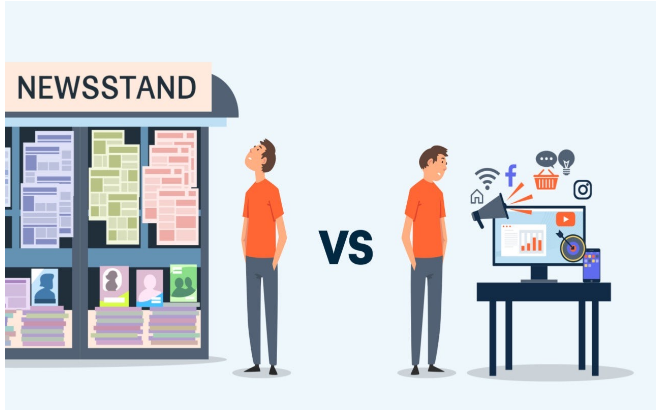
NLP Project Presentation Fall 2023

Identifying Genuine Disaster-Related Tweets for Urgency Detection

Group 5:
Rose Hemans
Pranav Chandaliya
Vaishnavi Nagarajaiah

How would you respond in the face of an emergency or disaster situation?

Shift from Traditional Media to Online Platforms

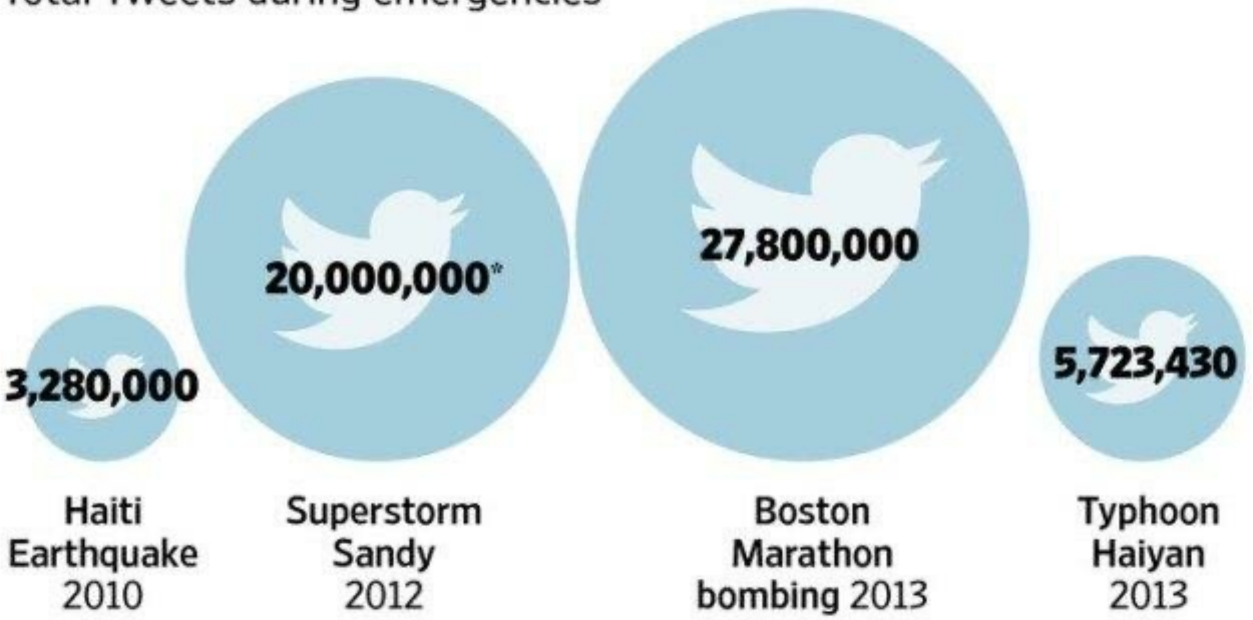


- Technological advancements have liberated societies from sole reliance on traditional media (newspapers, radio, TV), fostering a diverse communication landscape.
- With rapidly evolving smartphone technologies, societies are just an 'app' away from being able to deliver or receive information within milliseconds.
- Popular social media platforms such as Twitter, Facebook, and YouTube, have supplanted the traditional media outlets for accessing and responding to information.
- Daily, millions of users globally stay connected and obtain their news through online social networks

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Measuring the Twitter Storms

Total Tweets during emergencies



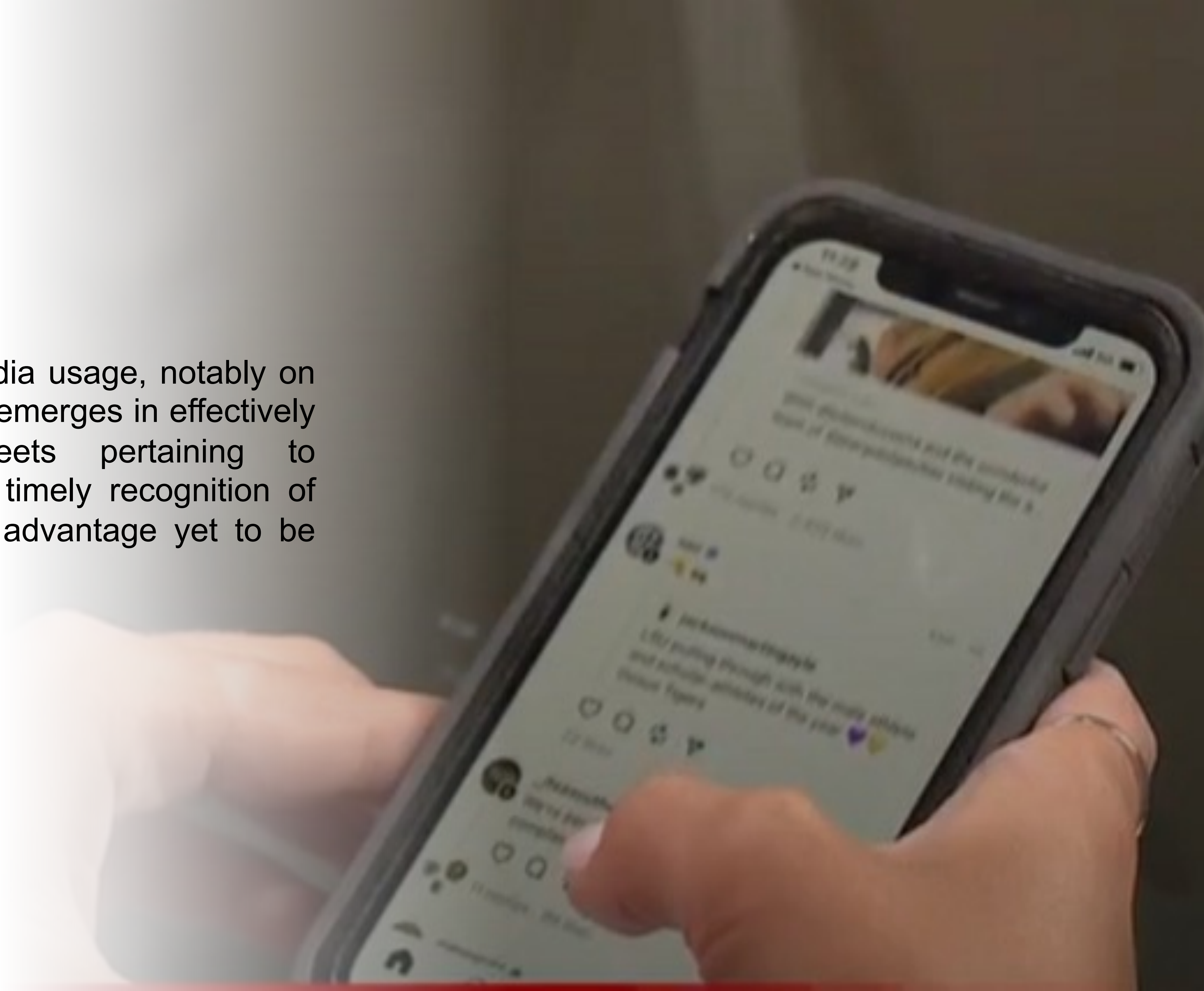
*Includes five-day period covering the approach and aftermath of the storm.

Sources: International Journal of Information Management; Pew Research Center

THE WALL STREET JOURNAL.

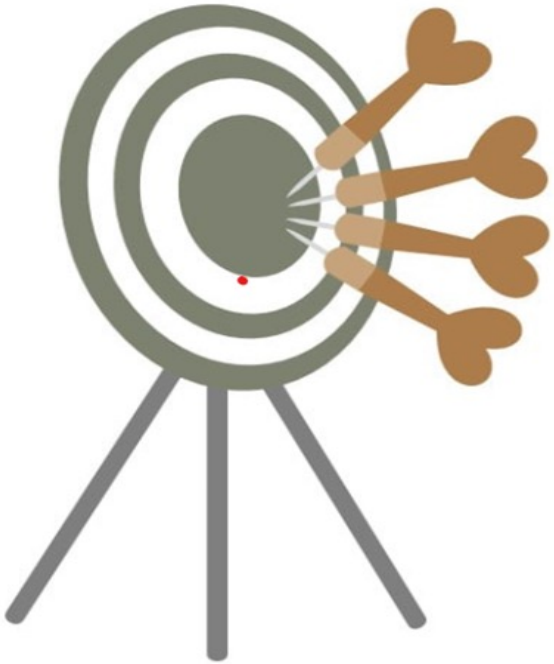
Problem Statement

In the age of prevalent social media usage, notably on platforms like Twitter, a challenge emerges in effectively detecting and identifying tweets pertaining to emergencies and disasters. The timely recognition of such tweets stands as a critical advantage yet to be fully realized.



Objectives

- The aim is to use machine learning and NLP to create a model identifying genuine disaster-related tweets, aiding accurate responses by relief teams to real disaster and not fake ones.
- This project intends to utilize labeled Twitter data sourced from Kaggle, employing advanced NLP techniques such as tokenization, stop word removal, and punctuation removal, followed by the utilization of a deep learning model for classification
- It also addresses the pressing need for automated systems that can quickly and accurately identify and prioritize relevant information during critical situations.



About the Data

- Twitter Dataset
- Data Source: Kaggle
- Data Size: 7K+ tweets

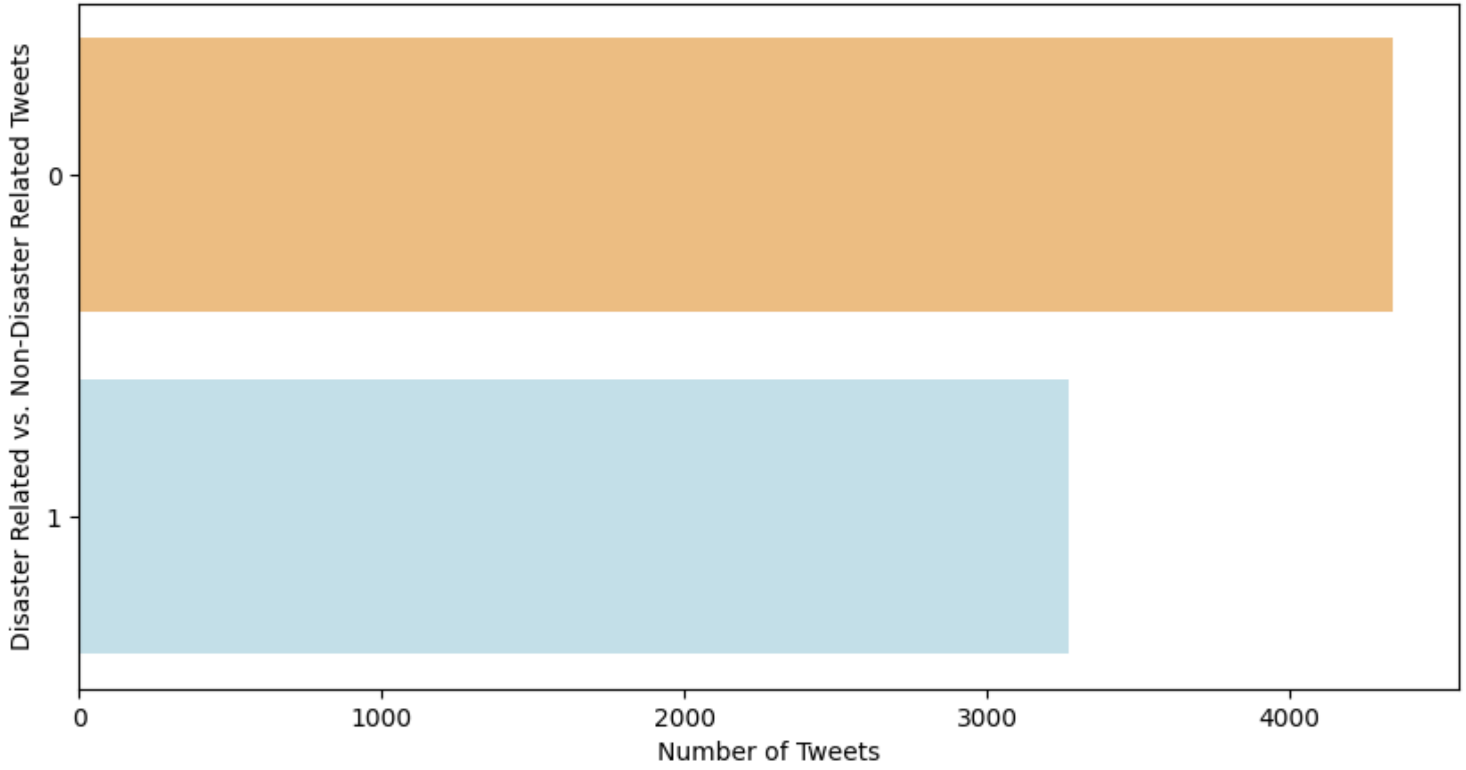
Important Attributes	Definition
id	Unique identifier
Text	Tweets
Target	Class of the tweet

id	text	target
1	Our Deeds are the Reason of this #earthquake May ALLAH Forgive us all	1
4	Forest fire near La Ronge Sask. Canada	1
5	All residents asked to 'shelter in place' are being notified by officers. No other evacuation or shelter in place orders are expected	1
6	13,000 people receive #wildfires evacuation orders in California	1
7	Just got sent this photo from Ruby #Alaska as smoke from #wildfires pours into a school	1


EDA

Class Distribution

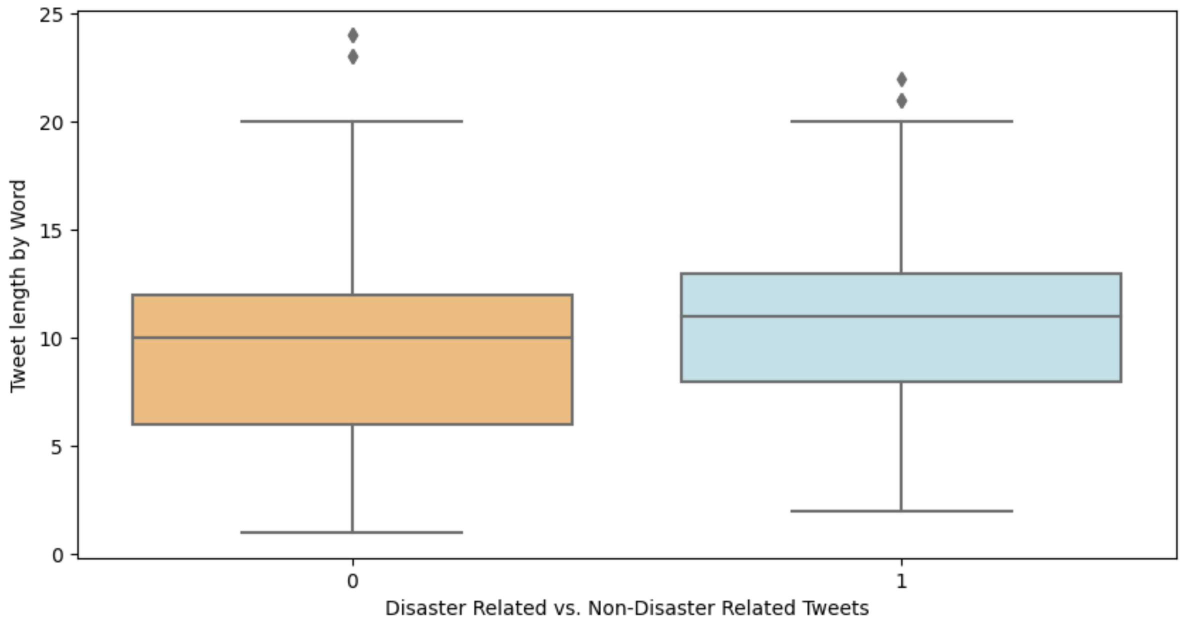
- The dataset is not highly imbalanced
- There are more normal tweets than emergency or disaster-related ones.

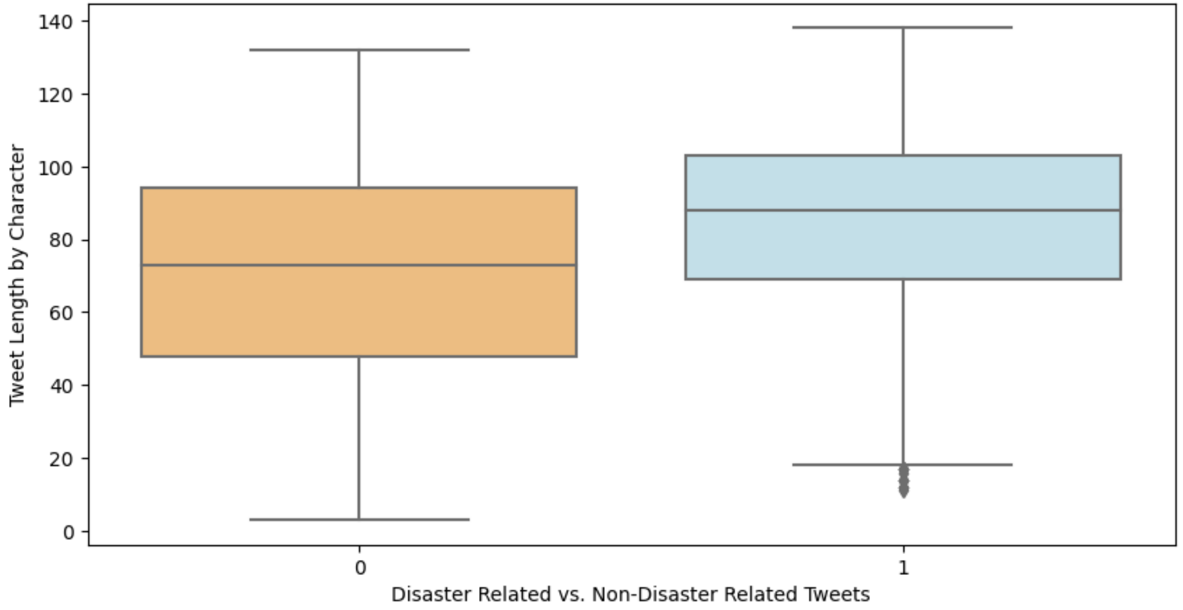


Tweet Character Length and Word Length

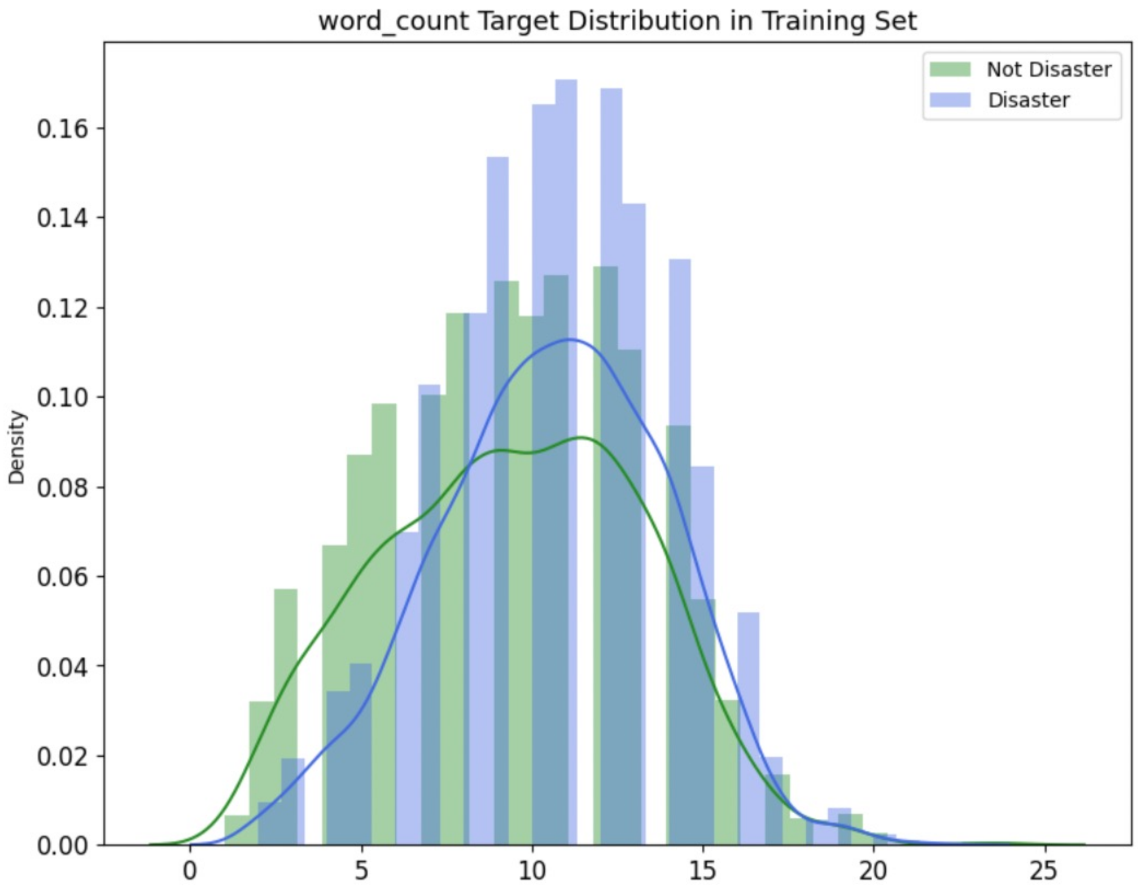
- The median character length and word length of tweets are greater for disaster/emergency tweets than for normal tweets

- Some emergency tweets might be very short in character length, hence creating outliers

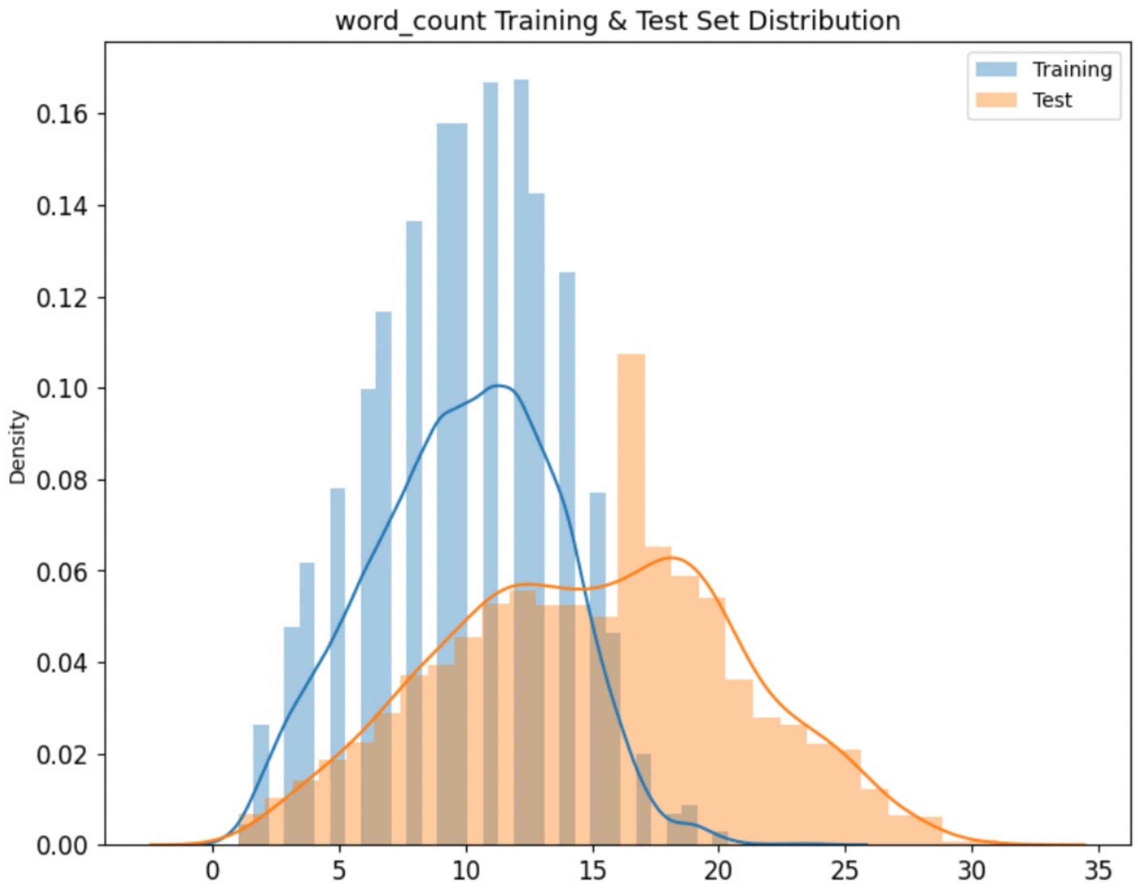




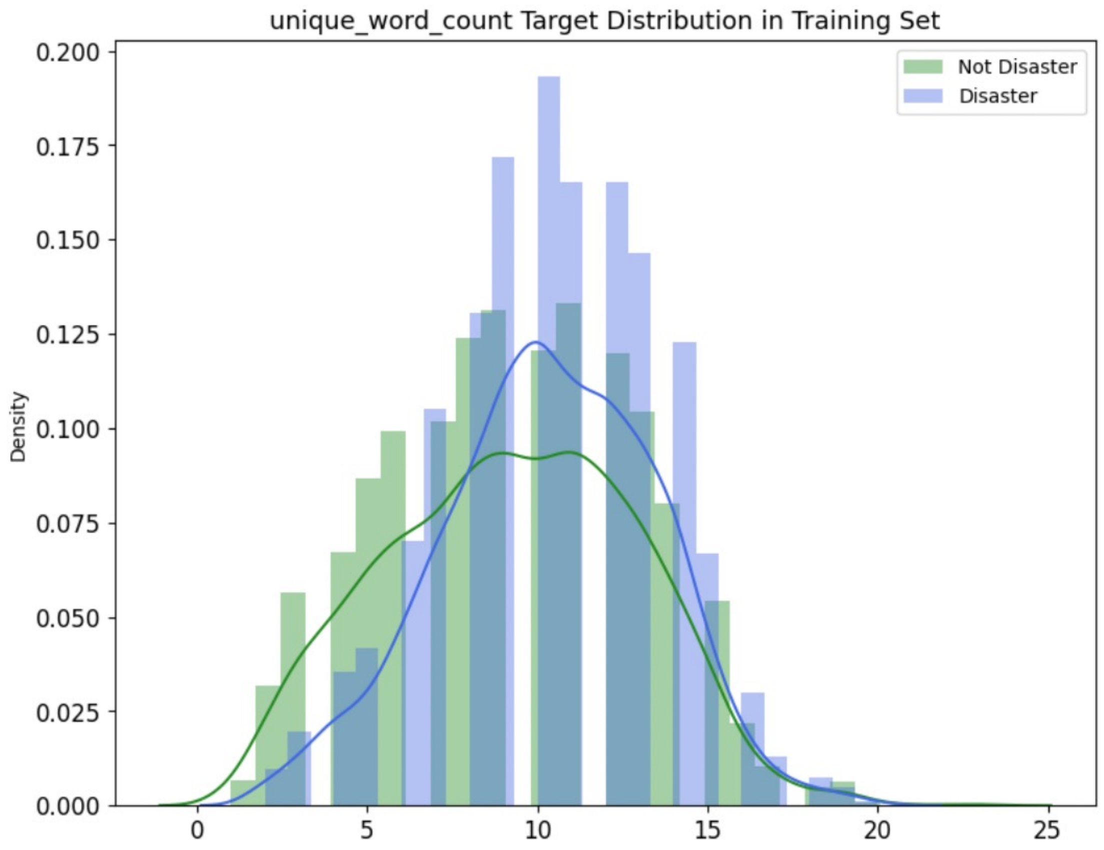
Word Count for Disaster vs Non-Disaster Tweets



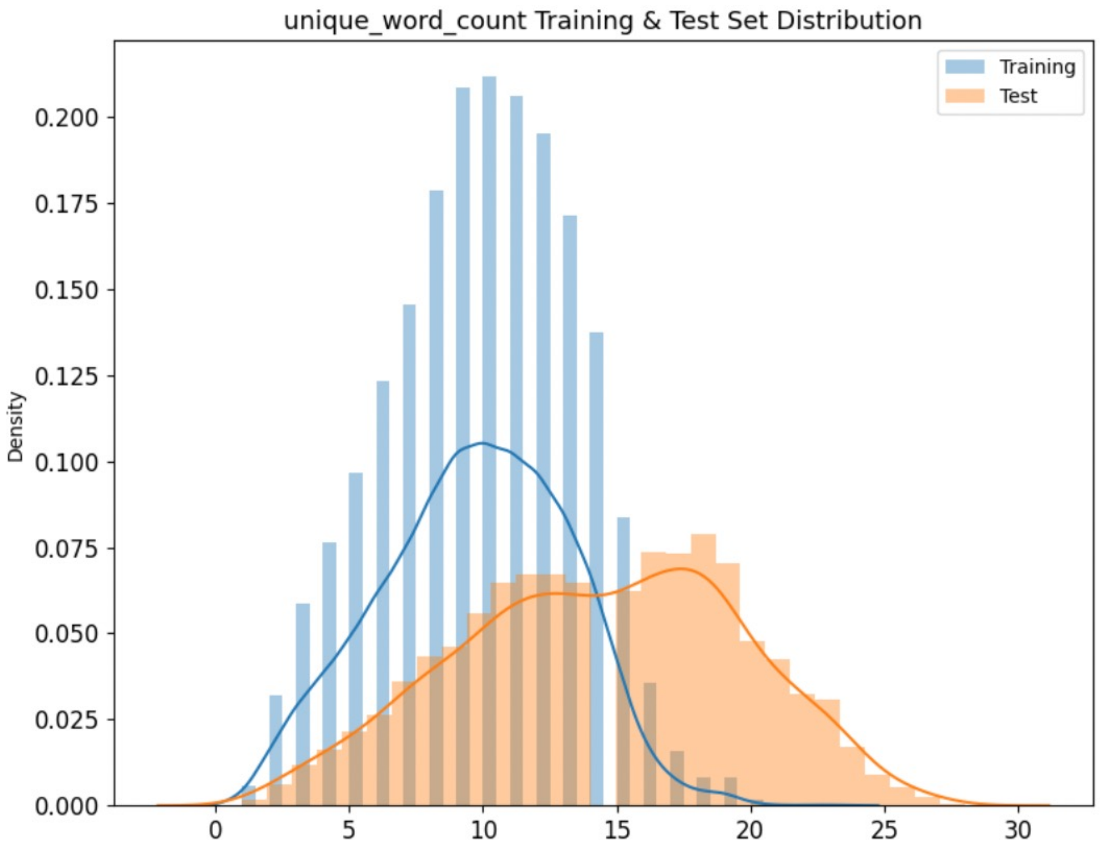
Word Count for Training vs Test Set



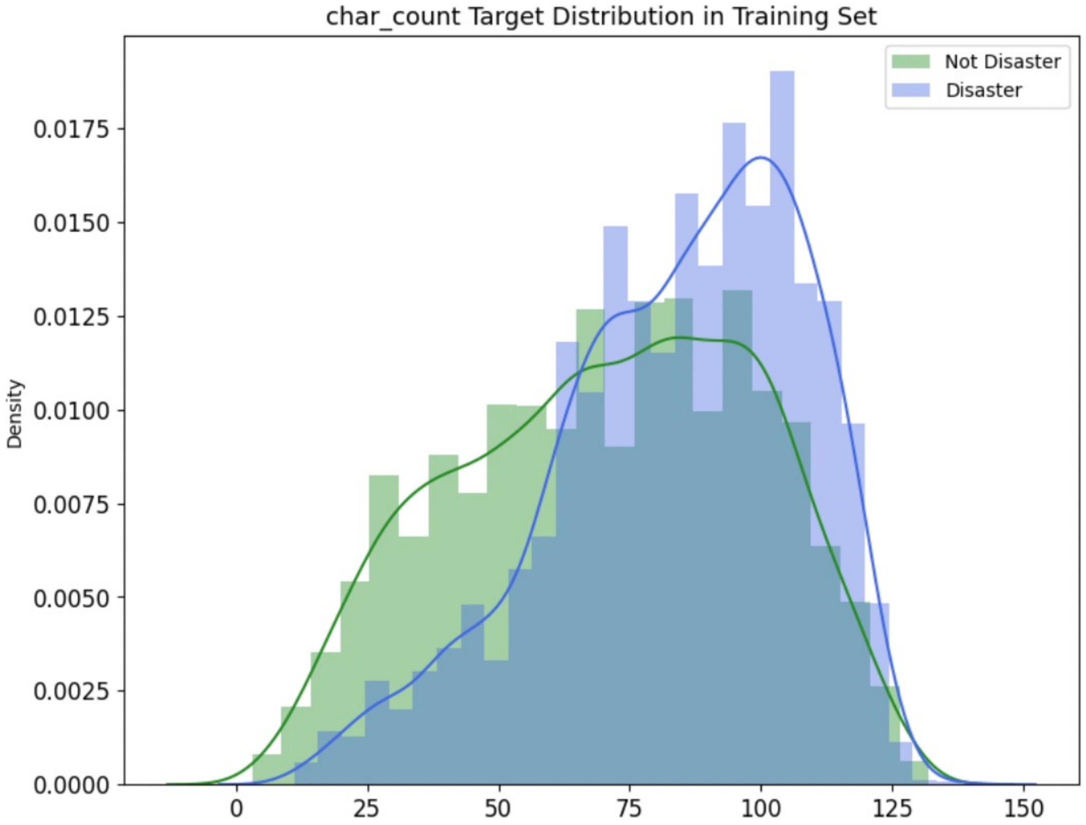
UNIQUE Word Count for Disaster vs Non-Disaster Tweets



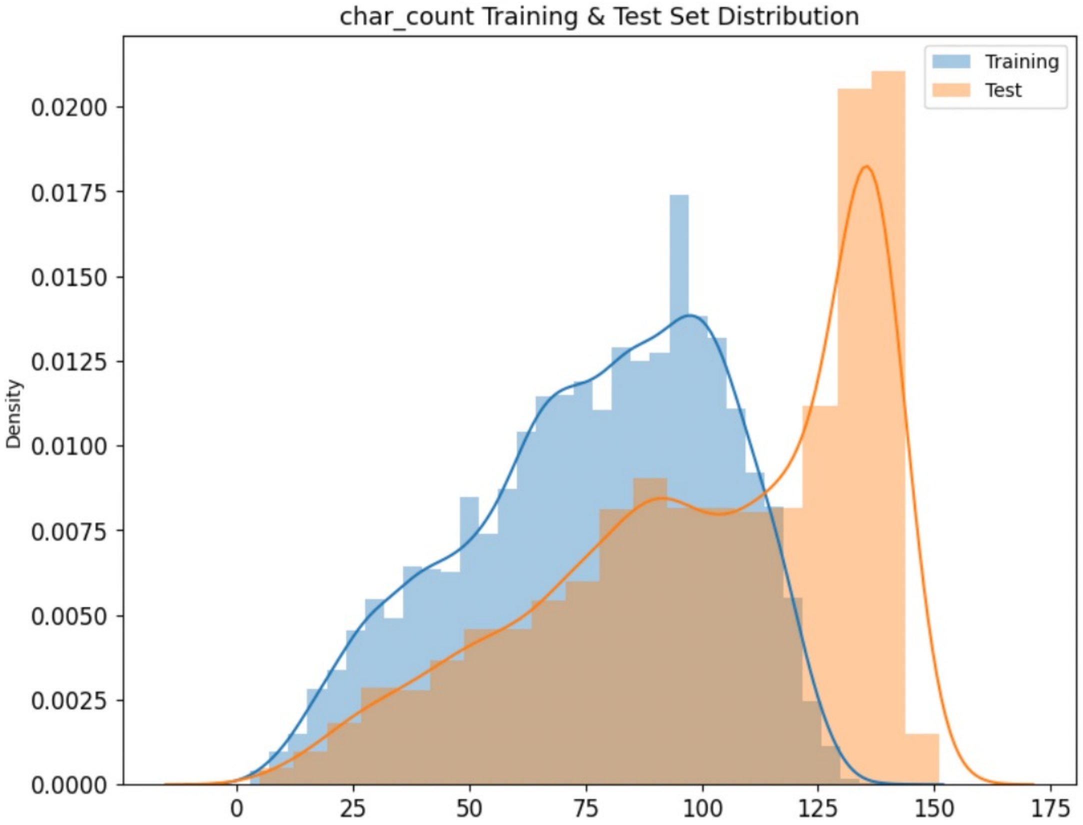
UNIQUE Word Count for Training vs Test Set



Character Count for Disaster vs Non-Disaster Tweets



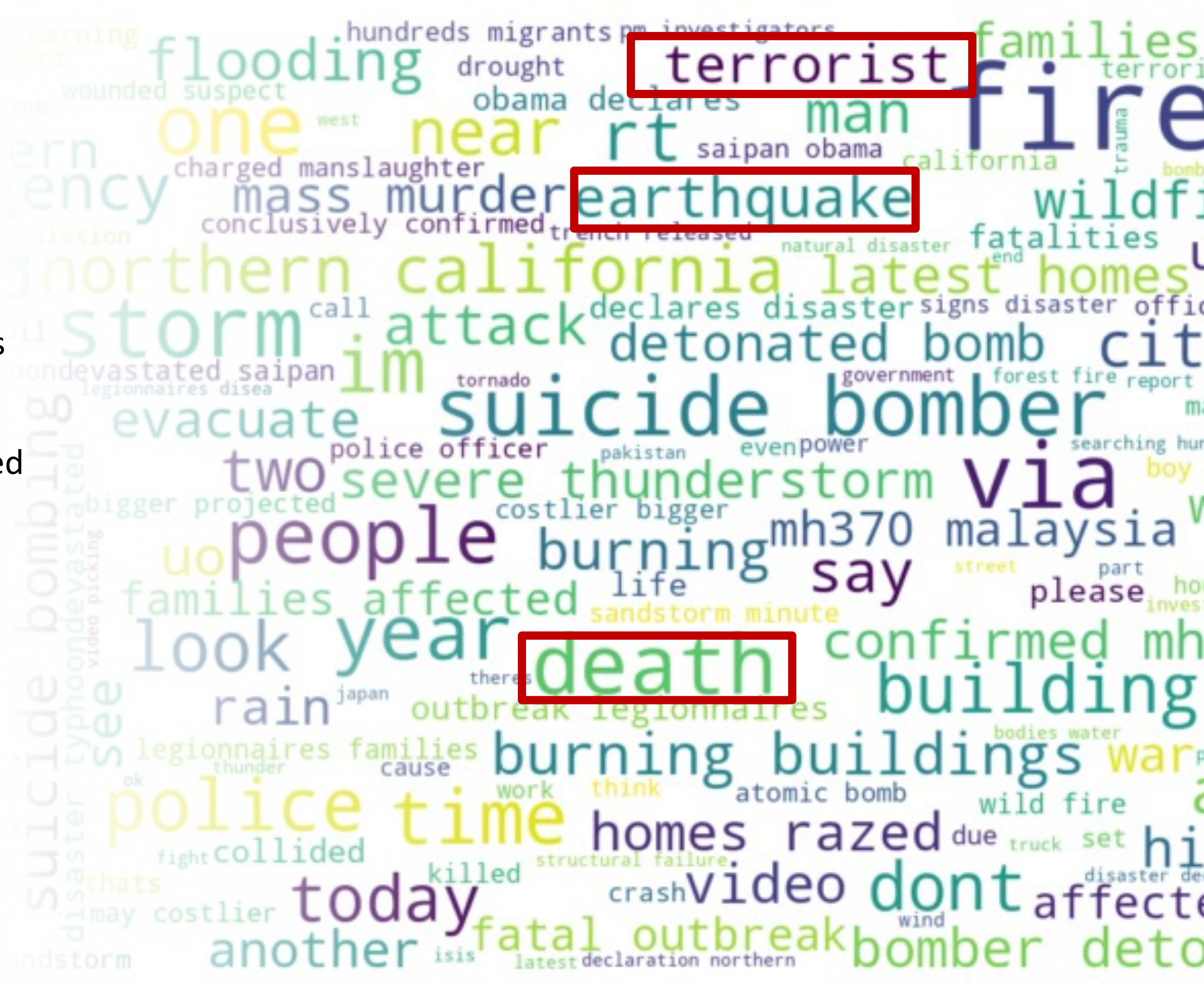
Character Count for Training vs Test Set



Word Cloud – Class1

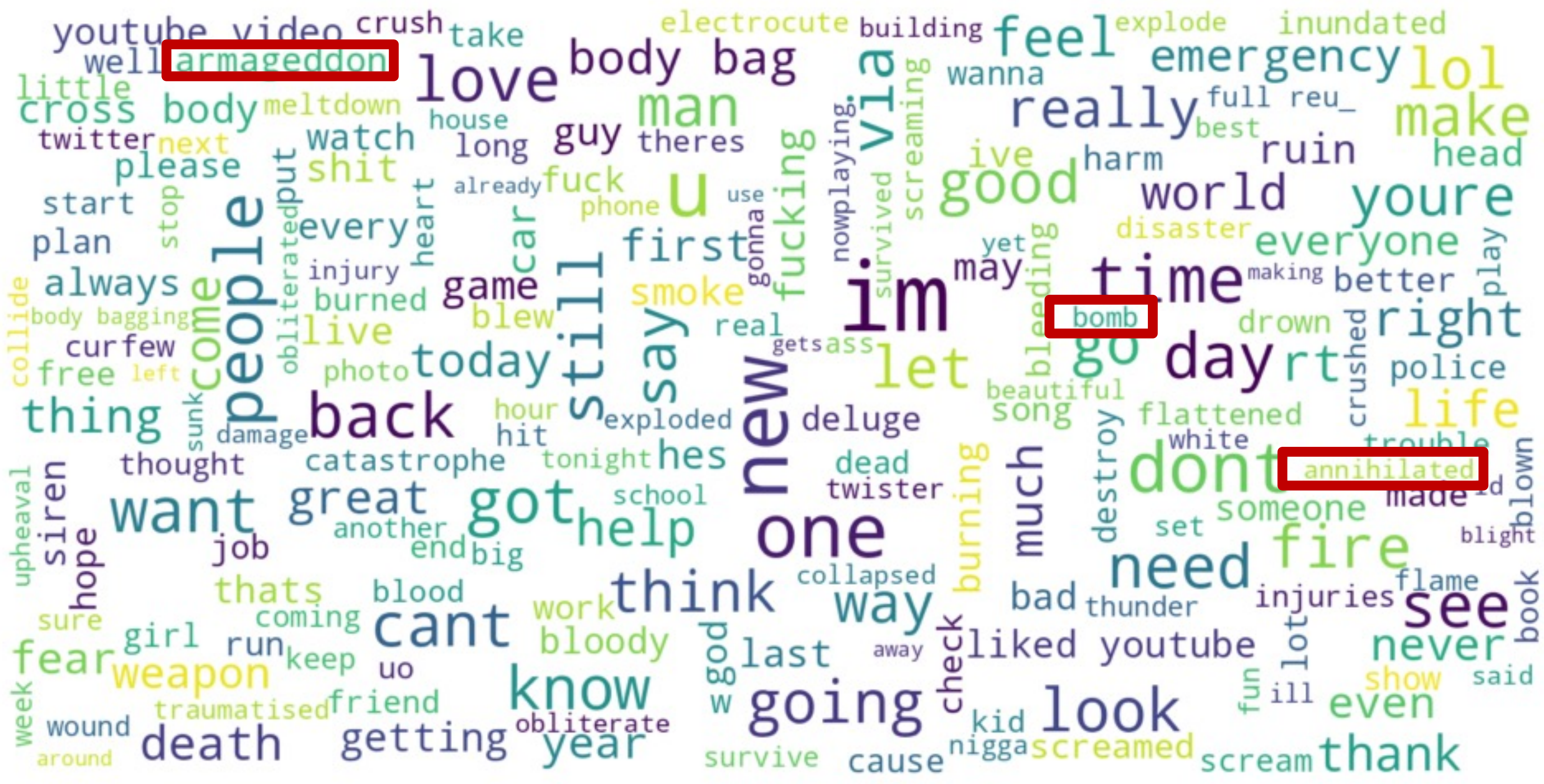
- Word cloud visually represents Class 1 Disaster/Emergency tweets, revealing a prominent cluster of terms closely associated with emergencies and disasters

- This visualization provides a clear insight into the prevalent themes and topics within this category



Word Cloud – Class2

- ➢ Word cloud pertains to Class 0, representing normal tweets.
- ➢ Occasionally we see words like "armageddon", "annihilated", and "bomb"



Investigating "armageddon", "annihilated", "bomb" References

- "armageddon" tweets related to movies and gaming
- "annihilated" tweets related to sports e.g., baseball and college football
- "bomb" tweets relate to gaming and the phrase "time bomb"

```
[ ]  formal_tweets[formal_tweets['text'].str.contains('armageddon')]['text'].head(5).apply(print)

pbban temporary300 avysss armageddon kill flags fast xp reason
pbban temporary300 russaky89 armageddon kill flags fast xp reason
official vid doublecups httpstcolfkmtzaekk trubgme prod thisizbwright armageddon
ouvindo peace love armageddon
best movie youve ever seen armageddon httptcoqouxigdtbz
304    None
305    None
306    None
307    None
308    None
Name: text, dtype: object
```

```
[ ]  formal_tweets[formal_tweets['text'].str.contains('annihilated')]['text'].head(5).apply(print)

episode trunks annihilated freiza cleanest shit ever showed [redacted] mercy
shall annihilated petebests dessicated laid bare shall kneel
uribe annihilated baseball mets
marksmaponyane heysundowns annihilated previous meeting celticindeed improvement
volfan326 tneazzy mizzou annihilated florida past 2 seasons even ended muschamps career cant compete bama
209    None
210    None
211    None
212    None
213    None
Name: text, dtype: object
```

```
[ ]  formal_tweets[formal_tweets['text'].str.contains('bomb')]['text'].head(5).apply(print)

mfalcon21 go look blew w atomic bomb
listen hit song summer bomb full positive energy youthdid like ithttpstco2liwkjybe9 norge2040
beforeitsnews global derivatives 15 quadrillion time bomb httptcoghmmuj7gbe vu_ httptcou9lvvlzhye httptcolj
bomb ass firework picture httptcolr4btvueom
namjoons fantastic bomb bye omg
748     None
1059    None
1060    None
1062    None
1065    None
Name: text, dtype: object
```

Text Preprocessing

- Tokenization
- Stop word removal
- Punctuation removal
- Normalization

```
tokenizer = AutoTokenizer.from_pretrained('bert-large-uncased')
bert = TFBertModel.from_pretrained('bert-large-uncased')
```

```
tokenizer_config.json: 100% 28.0/28.0 [00:00<00:00, 931B/s]
config.json: 100% 571/571 [00:00<00:00, 11.7kB/s]
vocab.txt: 100% 232k/232k [00:00<00:00, 1.76MB/s]
tokenizer.json: 100% 466k/466k [00:00<00:00, 2.45MB/s]
model.safetensors: 100% 1.34G/1.34G [00:09<00:00, 225MB/s]
```

```
x_train = tokenizer(
    text=train_data.text.tolist(),
    add_special_tokens=True,
    max_length=max_length,
    truncation=True,
    padding=True,
    return_tensors='tf',
    return_token_type_ids = False,
    return_attention_mask = True,
    verbose = True)
```

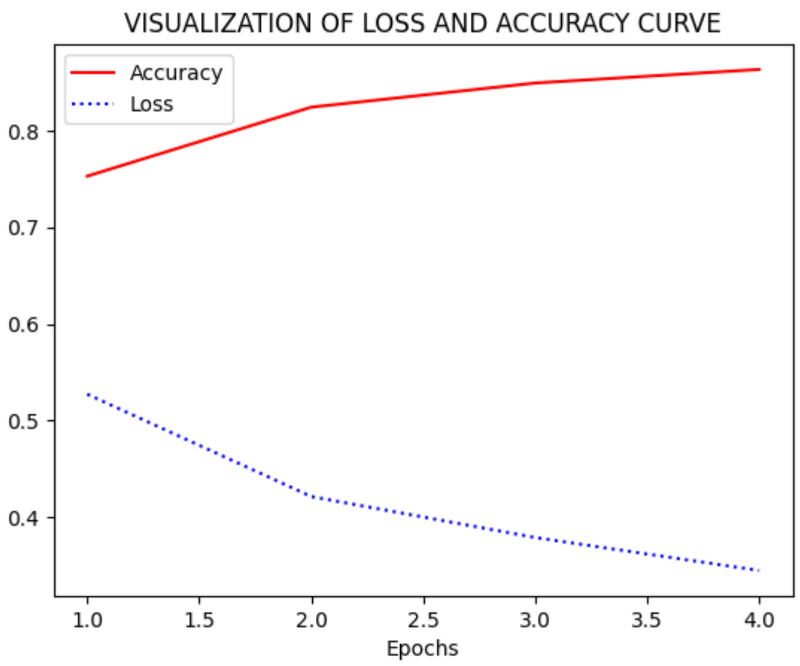
BERT Model

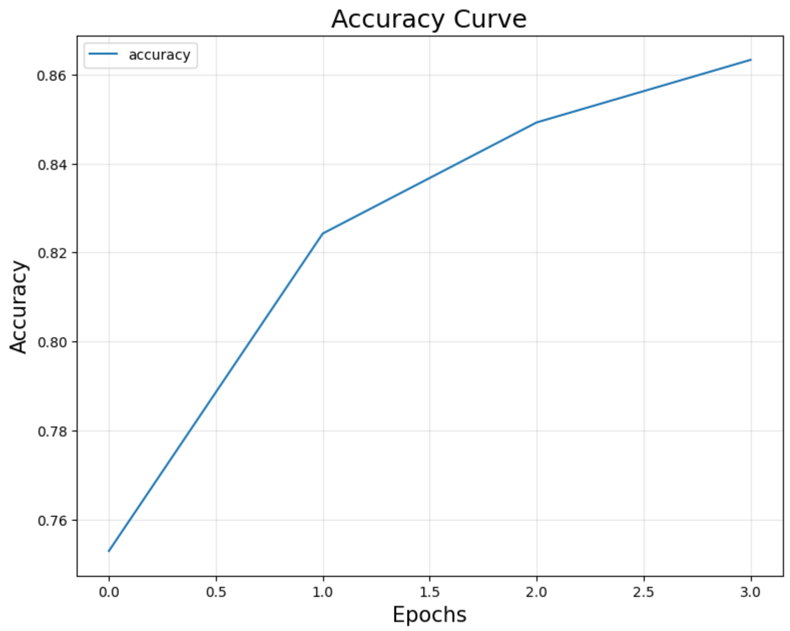
```
input_ids = Input(shape=(max_len,), dtype=tf.int32, name="input_ids")
input_mask = Input(shape=(max_len,), dtype=tf.int32, name="attention_mask")


embeddings = bert(input_ids,attention_mask = input_mask)[1]
out = tf.keras.layers.Dropout(0.1)(embeddings)

out = Dense(128, activation='relu')(out)
out = tf.keras.layers.Dropout(0.1)(out)
out = Dense(32,activation = 'relu')(out)

y = Dense(1,activation = 'sigmoid')(out)

model = tf.keras.Model(inputs=[input_ids, input_mask], outputs=y)
model.layers[2].trainable = True
```

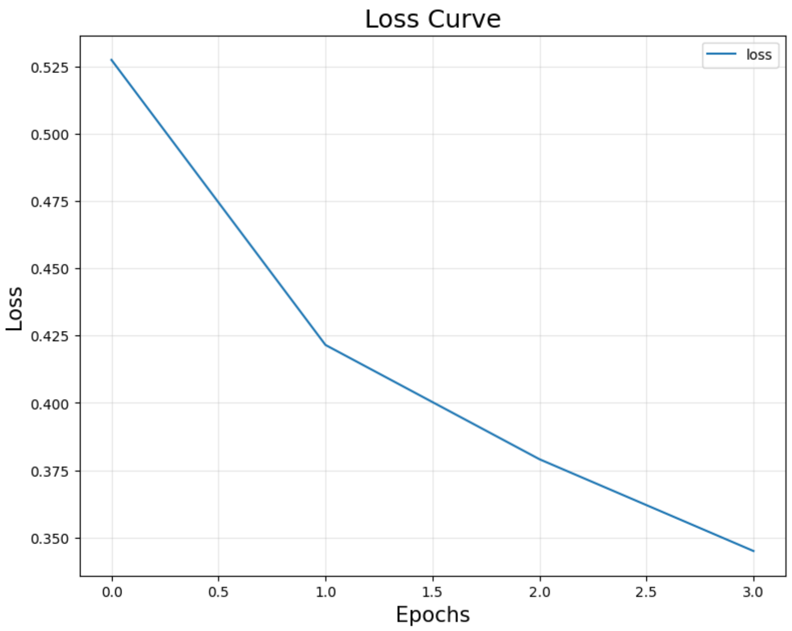
```
model.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape                     Param #    Connected to
==================================================================================================
 input_ids (InputLayer)         [(None, 36)]                     0          []

 attention_mask (InputLayer     [(None, 36)]                     0          []
 )

 tf_bert_model (TFBertModel     TFBaseModelOutputWithPooli       3351418    ['input_ids[0][0]',
 )                              ngAndCrossAttentions(last_       88          'attention_mask[0][0]']
                                hidden_state=(None, 36, 10
                                24),
                                 pooler_output=(None, 1024
                                ),
                                 past_key_values=None, hid
                                den_states=None, attention
                                s=None, cross_attentions=N
                                one)

 dropout_73 (Dropout)           (None, 1024)                     0          ['tf_bert_model[0][1]']

 dense (Dense)                  (None, 128)                      131200     ['dropout_73[0][0]']

 dropout_74 (Dropout)           (None, 128)                      0          ['dense[0][0]']

 dense_1 (Dense)                (None, 32)                       4128       ['dropout_74[0][0]']

 dense_2 (Dense)                (None, 1)                        33         ['dense_1[0][0]']

==================================================================================================
Total params: 335277249 (1.25 GB)
Trainable params: 335277249 (1.25 GB)
Non-trainable params: 0 (0.00 Byte)
__________________________________________________________________________________________________
```

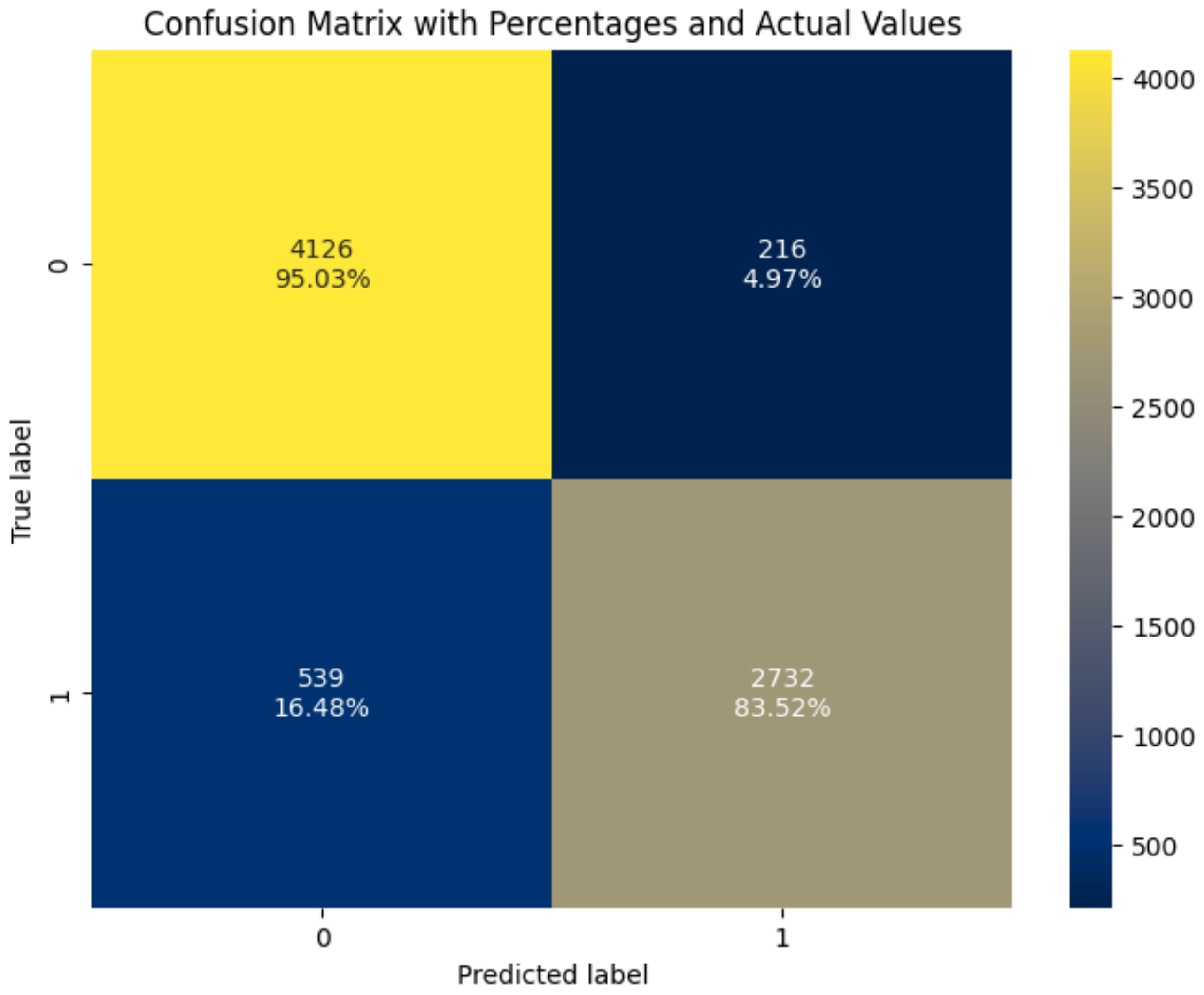
Model


VISUALIZATION OF LOSS AND ACCURACY CURVE
Accuracy
Loss
0.8
0.7
0.6
0.5
0.4
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Epochs


Accuracy Curve
accuracy
0.86
0.84
0.82
0.80
0.78
0.76
Accuracy
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Epochs

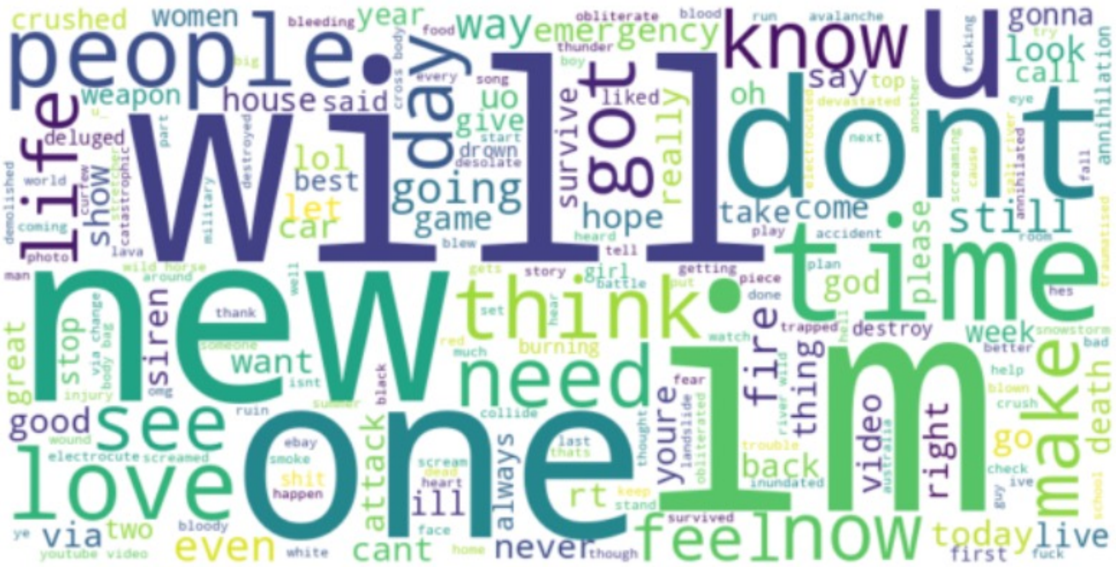

Loss Curve
loss
0.525
0.500
0.475
0.450
0.425
0.400
0.375
0.350
Loss
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Epochs

Evaluation

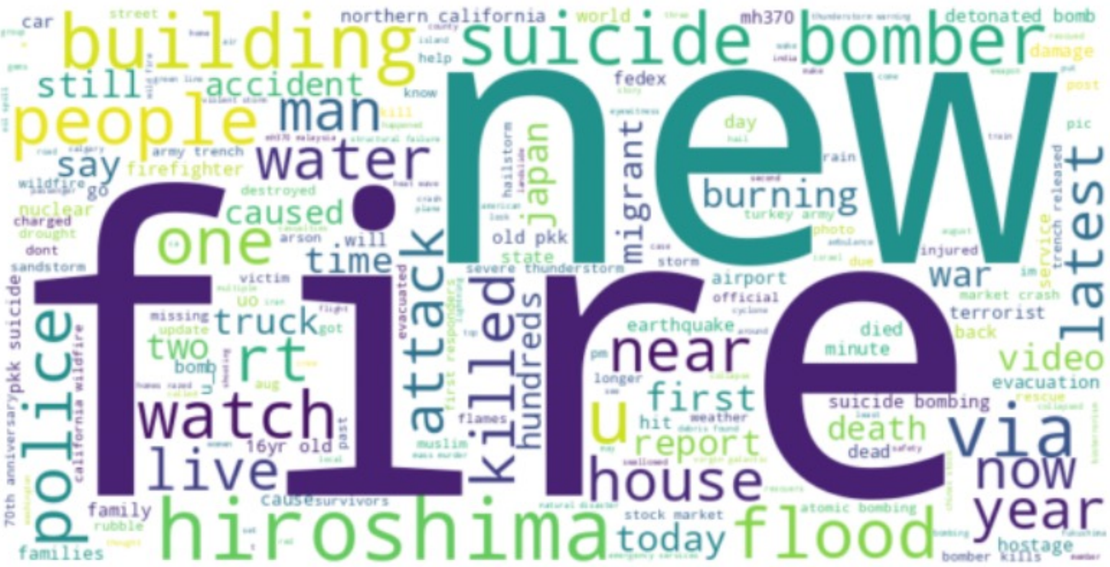

Confusion Matrix with Percentages and Actual Values
4126
95.03%
216
4.97%
539
16.48%
2732
83.52%
True label
Predicted label
0
1
0
1
4000
3500
3000
2500
2000
1500
1000
500

Evaluation – Predictions Word Cloud

Non-Disaster Predicted Tweets



Disaster Predicted Tweets



Evaluation – t-SNE Visualization