Importing Libraries

In [1]:

```
import nltk
import numpy as np
import pandas as pd
```

In [2]:

```
nltk.download("punkt")
nltk.download("stopwords")
nltk.download("wordnet")
nltk.download('omw-1.4')
from nltk.tokenize import word_tokenize
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
```

Initialising Preprocessing Functions

In [3]:

```
stopwords = nltk.corpus.stopwords.words('english')
stemmer = nltk.stem.SnowballStemmer('english')
lemmatizer = nltk.stem.WordNetLemmatizer()
```

Loading Data

In [4]:

```
df_train = pd.read_csv("../data/disaster/train.csv")
df_test = pd.read_csv("../data/disaster/test.csv")
df_train.drop(columns=["target"], inplace=True)
df = pd.concat([df_train, df_test])
```

In [5]:

```
print(df.shape)
df.head()
```

```
(10876, 4)
```

Out[5]:

	id	keyword	location	text
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...

In [6]:

```
corpus = df["text"].values
```

Week 1 - 21/01/2022

Tokenizing Corpus

In [7]:

```
def tokenize_corpus(tweets):
    return list(map(word_tokenize, tweets))
```

In [8]:

```
tokenized_corpus = tokenize_corpus(corpus)
```

In [9]:

```
for i in range(3):
    print(f"Tweet {i}: {tokenized_corpus[i]}")
```

```
Tweet 0: ['Our', 'Deeds', 'are', 'the', 'Reason', 'of', 'this', '#', 'earthq
uake', 'May', 'ALLAH', 'Forgive', 'us', 'all']
Tweet 1: ['Forest', 'fire', 'near', 'La', 'Ronge', 'Sask', '.', 'Canada']
Tweet 2: ['All', 'residents', 'asked', 'to', "'shelter", 'in', 'place', "'",
'are', 'being', 'notified', 'by', 'officers', '.', 'No', 'other', 'evacuatio
n', 'or', 'shelter', 'in', 'place', 'orders', 'are', 'expected']
```

Removing Stopwords

In [10]:

```
def remove_stopwords(tokenized_tweets):
    _function = lambda tweet: [word for word in tweet if word not in stopwords]
    return list(map(_function, tokenized_tweets))
```

In [11]:

```
tokenized_nostopw_corpus = remove_stopwords(tokenized_corpus)
```

In [12]:

```
for i in range(3):
    print(f"Tweet {i}: {tokenized_nostopw_corpus[i]}")
```

```
Tweet 0: ['Our', 'Deeds', 'Reason', '#', 'earthquake', 'May', 'ALLAH', 'Forg
ive', 'us']
Tweet 1: ['Forest', 'fire', 'near', 'La', 'Ronge', 'Sask', '.', 'Canada']
Tweet 2: ['All', 'residents', 'asked', "'shelter", 'place', "'", 'notified',
'officers', '.', 'No', 'evacuation', 'shelter', 'place', 'orders', 'expecte
d']
```

Creating Inverted Index

In [13]:

```
def create_inverted_index(tokenized_tweets):
    inverted_index = dict()
    for document_idx in range(len(tokenized_tweets)):
        for word in tokenized_tweets[document_idx]:
            if word not in inverted_index:
                inverted_index[word] = list()
            inverted_index[word].append(document_idx)

    for key in inverted_index:
        inverted_index[key] = list(set(inverted_index[key]))

    return inverted_index
```

In [14]:

```
inverted_index = create_inverted_index(tokenized_nostopw_corpus)
```

In [15]:

```
for key in list(inverted_index.keys())[:3]:
    print(f"Word: {key}\nTweet Indices: {inverted_index[key]}\n")
```

```
Word: Our
Tweet Indices: [0, 9095, 7816, 1673, 1550, 8080, 3345, 8979, 4630, 3099, 297
6, 3618, 3235, 2855, 6567, 3369, 2220, 4659, 3124, 4024, 4281, 3786, 10829,
1371, 9570, 3172, 10603, 1645, 4209, 7157, 4987, 2431]

Word: Deeds
Tweet Indices: [0, 4985]

Word: Reason
Tweet Indices: [0, 8610, 304, 305, 317, 319]
```

Week 2 - 28/01/2022

Case Folding

In [16]:

```
def case_folding(tokenized_tweets):
    _function = lambda tweet: [word.lower() for word in tweet]
    return list(map(_function, tokenized_tweets))
```

In [17]:

```
tokenized_nostopw_case_corpus = case_folding(tokenized_nostopw_corpus)
```

In [18]:

```
for i in range(3):
    print(f"Tweet {i}: {tokenized_nostopw_case_corpus[i]}")
```

```
Tweet 0: ['our', 'deeds', 'reason', '#', 'earthquake', 'may', 'allah', 'forg
ive', 'us']
Tweet 1: ['forest', 'fire', 'near', 'la', 'ronge', 'sask', '.', 'canada']
Tweet 2: ['all', 'residents', 'asked', "'shelter", 'place', "'", 'notified',
'officers', '.', 'no', 'evacuation', 'shelter', 'place', 'orders', 'expecte
d']
```

Lemmatizing Words

In [19]:

```
def lemmatize_words(tokenized_tweets):
    _function = lambda tweet: [lemmatizer.lemmatize(word) for word in tweet]
    return list(map(_function, tokenized_tweets))
```

In [20]:

```
tokenized_nostopw_case_lemm_corpus = lemmatize_words(tokenized_nostopw_case_corpus)
```

In [21]:

```
for i in range(3):
    print(f"Tweet {i}: {tokenized_nostopw_case_lemm_corpus[i]}")
```

```
Tweet 0: ['our', 'deed', 'reason', '#', 'earthquake', 'may', 'allah', 'forgi
ve', 'u']
Tweet 1: ['forest', 'fire', 'near', 'la', 'ronge', 'sask', '.', 'canada']
Tweet 2: ['all', 'resident', 'asked', "'shelter", 'place', "'", 'notified',
'officer', '.', 'no', 'evacuation', 'shelter', 'place', 'order', 'expected']
```

Stemming

In [22]:

```
def stem_words(tokenized_tweets):
    _function = lambda tweet: [stemmer.stem(word) for word in tweet]
    return list(map(_function, tokenized_tweets))
```

In [23]:

```
tokenized_nostopw_case_stem_corpus = lemmatize_words(tokenized_nostopw_case_corpus)
```

In [24]:

```
for i in range(3):
    print(f"Tweet {i}: {tokenized_nostopw_case_stem_corpus[i]}")
```

```
Tweet 0: ['our', 'deed', 'reason', '#', 'earthquake', 'may', 'allah', 'forgi
ve', 'u']
Tweet 1: ['forest', 'fire', 'near', 'la', 'ronge', 'sask', '.', 'canada']
Tweet 2: ['all', 'resident', 'asked', "'shelter", 'place', "'", 'notified',
'officer', '.', 'no', 'evacuation', 'shelter', 'place', 'order', 'expected']
```

Created new Inverted Index

In [25]:

```
inverted_index_processed = create_inverted_index(tokenized_nostopw_case_lemm_corpus) # toke
```

In [26]:

```
for key in list(inverted_index_processed.keys())[:3]:
    print(f"Word: {key}\nTweet Indices: {inverted_index_processed[key]}\n")
```

```
Word: our
Tweet Indices: [0, 9095, 7816, 1673, 1550, 8080, 3345, 8979, 4630, 3099, 297
6, 3618, 3235, 2855, 6567, 3369, 2220, 4659, 3124, 4024, 4281, 3786, 10829,
1371, 9570, 3172, 10603, 1645, 4209, 7157, 4987, 2431]

Word: deed
Tweet Indices: [0, 4985]

Word: reason
Tweet Indices: [0, 1920, 10496, 4997, 5000, 5001, 9737, 9739, 781, 8593, 537
2, 6166, 7961, 10526, 8224, 8610, 10670, 304, 305, 7218, 6453, 9911, 2747, 6
459, 4669, 317, 319, 2112, 7740, 8001, 8260, 2252, 6991, 6232, 6242, 746, 48
43, 4333, 6898, 4084, 763, 3452, 894, 4991]
```