Web Science Assessment (M) – Geolocalisation

Bhuvaneshwari Ramakrishnan (2647486R)

1. Distribution of London Tweet Data:

To analyse the 2000-5000 locations of the tweet data being given, a grid of 1kmX1km was created and the tweets were allocated to the grids to see the distribution. The analysis of the geo co-ordinates is then done by producing a heat map and bar graph containing the spread of tweets across the grid

1.1 Grid creation and distance calculation:

To calculate the number of rows and columns of the grid, the distance between the grid boundaries was calculated. This provides the total number of rows (latitude) and total number of columns (longitude) that is present inside the grid.

Pseudo code for Grid points:

num_rows = int (np.ceil (distance_points (bottomLeft, topLeft)))
num_cols = int (np.ceil (distance_points (bottomLeft, bottomRight)))

This gives **48 rows** and **59 columns**.

To get the latitude longitude points use linspace:

Cols = np.linspace (bottomLeft [0], bottomRight [0], num=num_cols)
Rows = np.linspace (bottomLeft [1], topLeft [1], num=num_rows)

The distance between the two points (A, B) can be calculated using the **Haversine formula** which is defined in the **distance_points** function.

Pseudo code for Distance calculation:

LongA, LatA – Points A

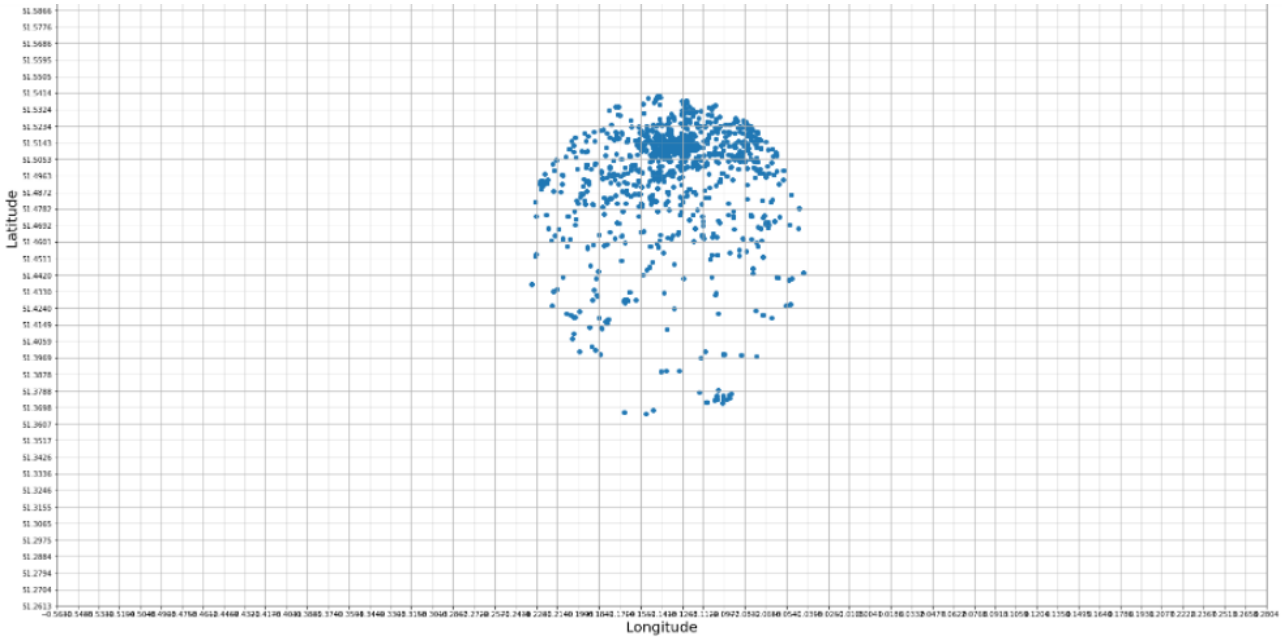
LongB, LatB – Points B

delta_lat = (LatA-LatB)
delta_long = (LongA-LongB)
Radius = 6371

a = sin (delta_lat/2) **2 + (cos (LatA)*cos (LatB)* sin (delta_long/2)**2)
c = 2*atan2 (sqrt (a), sqrt (1-a))
d = Radius*c

1.2 Tweet Distribution and allocation:

Upon plotting the tweets using the coordinates to the grid created, it can be seen that most of the tweets are concentrated to certain latitude and longitude.



Allocation of tweets to the grid to calculate the number of number of tweets inside each grid is achieved by first creating an empty grid of the size 48X59 (number of rows and cols). Then the distance between the boundary co-ordinates (bottomLeft) and each tweet is calculated to get the row and column index positions. Once that it received, the grid location is incremented by 1 to store the count of tweet.

Pseudo code:

Tweet_grid = np.zeros in shape rows, cols

For each tweet:

Row index = ceil (distance (bottomLeft of grid, latitude of tweet coordinate))

Col index = ceil (distance (bottomLeft of grid, Longitude of tweet coordinate))

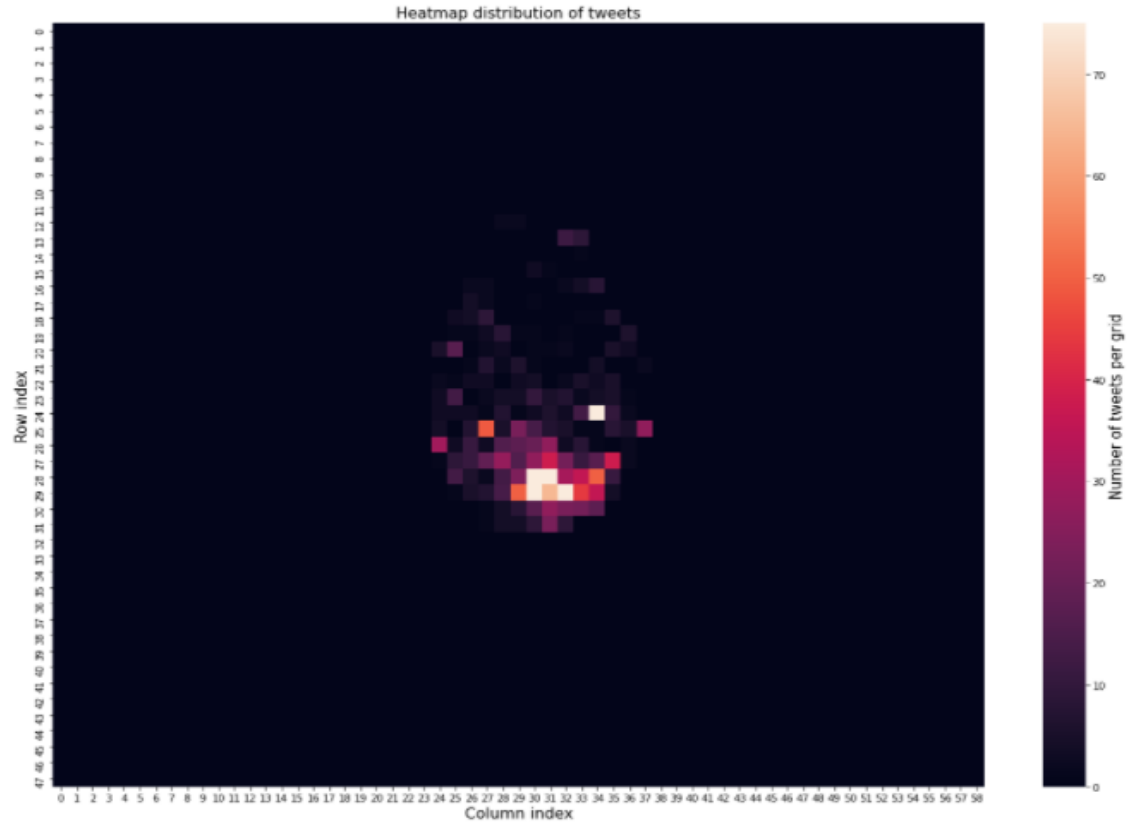
Tweet_grid [row index, col index] increment by 1

1.3 Data Interpretation:

1.3.1 Heat Map:

When plotting the Tweet_grid as a heat map (**sns**), it can be seen that a large number of tweets are concentrated at certain latitude longitude locations. The maximum number of tweets at a given grid sums up to: **1338** at location Tweet_grid **[29, 32]**. Most of the grids contain zero tweets with an increase in the number of tweets at row index range from [12, 32) and column index range from [23, 39).

This Heat Map helps to decipher where the information is being gathered from. This sort of information is really useful in case of disasters or political/social events or emergencies, where people can be in lookout for up-to date information.

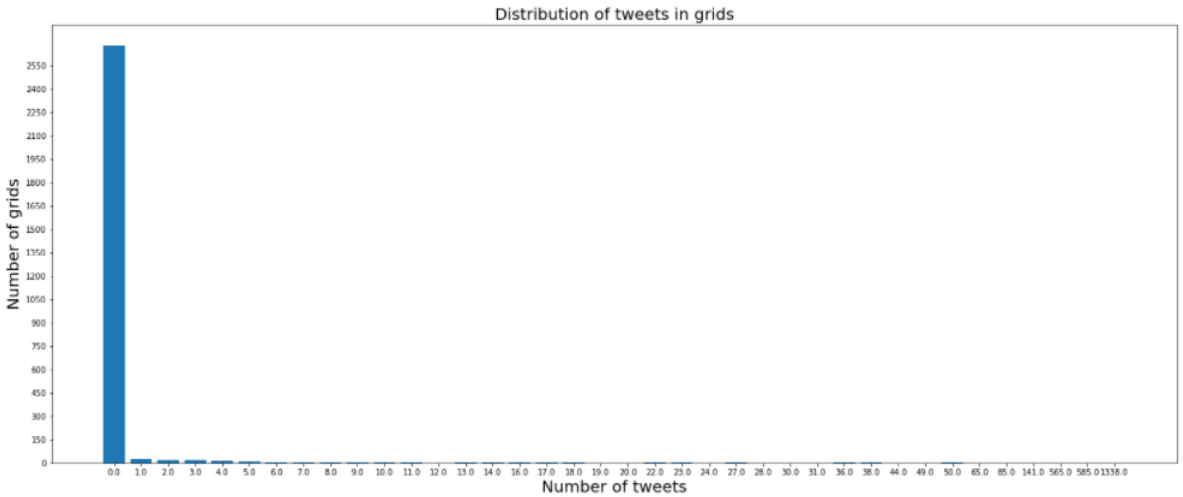


1.3.2 Bar Graph:

Displaying the number of tweets vs. number of grids gives a clear picture of how the tweets are distributed across the coordinates. This can be achieved by doing **np.unique** to the Tweet_grid that already contains the counts of all tweets at each grid index.

Around **2678** grids have a tweet count of **0** and a steep decrease in the number of grids to tweet ratio. Higher numbers of tweets are concentrated in less number of grids. The maximum tweets: **1338** is located at **1** grid followed by **585** tweets again concentrated at a single grid location.

Once the grid index is known, it is easier to get the co-ordinates at which the tweets are originated, hence helping at the time of emergencies or crisis.



2. Create a scoring model to calculate newsworthiness:

Utilizing the High Quality data file and the low Quality data file, a newsworthy scoring model was created which scores each term from each file with a set score based on threshold. This newsworthy model can then be used further to classify other data for their newsworthiness.

This is particularly helpful to filter unnecessary information so that emphasis can be given to data that is of most use.

2.1 Data Clean-up:

The high quality data files and the low quality data files are tokenised, so as to remove stopwords, other words that do not necessarily contribute to the newsworthiness. This has been achieved using SpaCy and nltk for stopwords.

A **tokenize_text** function has been created that takes in a text and provides tokens after processing it.

Pseudo Code:

Create doc with nlp(text)
For every term in doc:
    Check if it not a **stopword**, not **punctuation**, not **space** and is an **alphanumeric** (remove special characters)
    If true, add it to a list of tokens
Return the list of tokens

2.2 Creation of Background Document:

A Background document is a collection of both high quality and low quality content. This document is used to calculate the likelihood ratio of a term and can be used as training data for the model.

Background Document = High Quality tokens + Low Quality tokens

2.3 Creating the Scoring Model:

2.3.1 Term Frequency:

To calculate the term frequency of the words that occur in the high quality files, low quality files and the background document, Counter is used. Counter function gives the count of each text in the document.

Pseudo code:
HQ term frequencies = Counter (High Quality Tokens)
LQ term frequencies = Counter (Low Quality Tokens)
BG term frequencies = Counter (Background Doc Tokens)

2.3.2 Raw Frequency:

The raw frequency denotes the sum of the counts of the terms. This can be calculated by taking the length of the tokens in each data file

Pseudo:

Raw HQ = length (High Quality Tokens)
Raw LQ = length (Low Quality Tokens)
Raw BG = length (Background docs Tokens)

2.3.3 Model Creation:

The model is created in the function score_calculate that takes in a threshold value to categorise the tokens as high priority or low priority. This is important so as to eliminate any tokens which have scores in middle range and do not provide much importance in newsworthiness.

The model is created using the Background document as the training data.

Pseudo:

For every term in Background document tokens:
    If the term frequency in **HQ** tokens and **BG** doc is not 0, then
        Calculate **RHQ** (term) = TF_HQ (term)/Raw HQ / TF_BG (term)/Raw BG
    Else, RHQ (term) = 0
    If the term frequency in **LQ** tokens and **BG** doc is not 0, then
        Calculate **RLQ** (term) = TF_LQ (term)/Raw LQ / TF_BG (term)/Raw BG
    Else, RLQ (term) = 0
    If RHQ and RLQ are **greater** than **threshold**:
        SHQ and **SLQ** of term = RHQ and RLQ respectively
    Else, SHQ and SLQ of term = 0
Return the **SHQ** and **SLQ** of the entire set.

2.3.4 Data Frame of SHQ and SLQ:

The SHQ and SLQ that is calculated for each term in the Background document is stored as a Dataframe that displays the term and their corresponding SHQ and SLQ values.

Terms	SHQ	SLQ
new	1.079263	0.000000
york	0.000000	1.094022
times	0.000000	1.076376
bought	0.000000	1.435168
wordle	0.000000	1.291652
...	...	...
concentrate	0.000000	2.152753
confiscating	0.000000	2.152753
putins	0.000000	2.152753
cronies	0.000000	2.152753
saskatchewan	0.000000	2.152753

7932 rows × 2 columns

3. Newsworthiness of Tweets:

Using the scoring model created in the previous question, the Geo London tweets provided is tested for their newsworthiness. The column used to judge the newsworthiness is the text column of the tweets that contain the content of the tweets.

3.1 Tokenise the Tweets Text:

Similar to the high quality and low quality files, the Tweets text column is tokenised so that the newsworthiness can be calculated for text that contributes to the importance of the content.

Pseudo:
For each tweet in the Geo London Tweet set:
    Calling the tokenize_text function
Storing the tokens of each tweet in separate column of the Tweets Data Frame for ease of access.

3.2 Newsworthiness Calculation:

The newsworthiness of the tweet is called by using the SHQ and SLQ Dataframe created using the Background document previously.

Pseudo:
For each tweet in Geo London Tweet:
    For each token in the tweet:
        Sum of SHQ values and sum of SLQ values
    Calculate Newsworthiness value N_d :
    $\mathbf{N_d} = \log_2$ (1+sum of SHQ / 1+sum of SLQ)
    If N_d is **greater** than **0**, then
        Tweet is **newsworthy**
    Else, Tweet is **not newsworthy**
Storing the outcome of the newsworthiness in separate column in Tweets data Frame

3.2.1 Threshold calculation:

The threshold to calculate the SHQ and SLQ values has been chosen by checking the Newsworthiness values being displayed for the tweets. Based on the text and the content it can be seen that some of the tweets provide useful information whereas some provide more generic information.

With the values being displayed for the high quality tweets the threshold can be enhanced in such a way that the model is able to predict these high quality tweets properly.

The current **threshold** being used for the **SHQ** and **SLQ** are **1.0.**

3.3 Data Analysis of Newsworthy tweets:

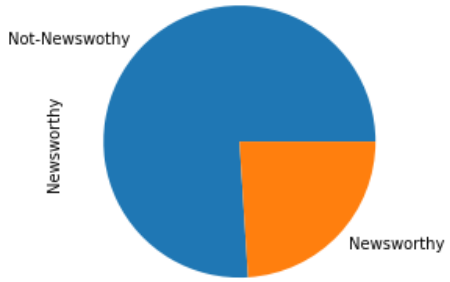
3.3.1 Newsworthy content:

After passing the tweets through the model and categorising them into newsworthy or not newsworthy, it can be seen that 994 tweets are newsworthy and 3148 are not.

This helps us to focus on tweets that are more important and disregard the more generic tweets and collect information from the ones that are newsworthy.

```
Not-Newswothy    3148
Newsworthy        994
Name: Newsworthy, dtype: int64

<AxesSubplot:ylabel='Newsworthy'>
```



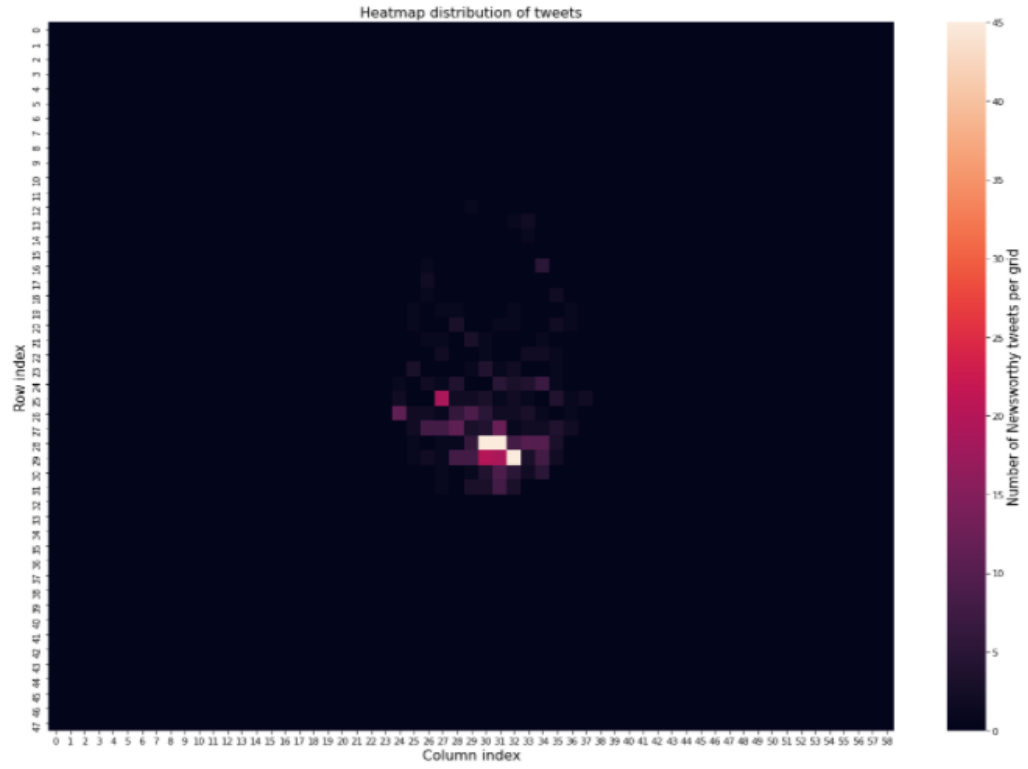
3.3.2 Tweet allocation into grids:

Similar to the normal tweet set, the newsworthy tweets are allocated to the grid of 48X59. This helps to better understand the distribution of newsworthy tweets according to their geo-location.

Pseudo:
Newsworthy_Tweet_grid = np.zeros in shape rows, cols
For each tweet:
    Row index = ceil (distance (bottomLeft of grid, latitude of tweet coordinate))
    Col index = ceil (distance (bottomLeft of grid, Longitude of tweet coordinate))
    Newsworthy_Tweet_grid [row index, col index] increment by 1

The Newsworthy_Tweet_grid now contains the count of tweets at each row, column index pair. This information is helpful to visualise how the tweets are spread across the co-ordinates with the help of heat map and the number of tweets vs. number of grids chart.

Heat Map representation of the Newsworthy Tweets:



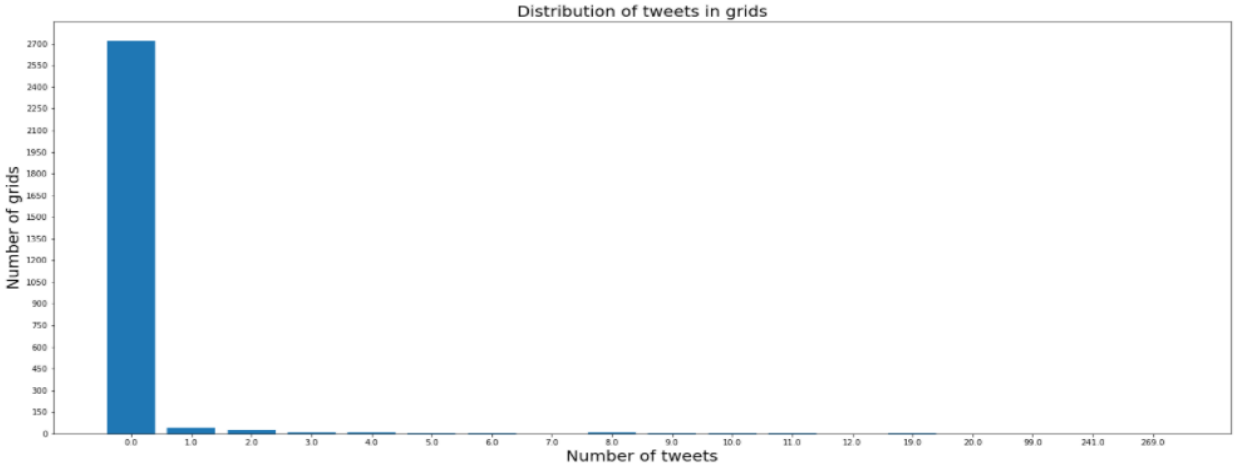
Similar to the full tweet data set, we can see that the newsworthy tweets are also concentrated to a particular grid index. But with the removal of majority number of tweets as not newsworthy, the intensity in the grids have drastically reduced.

The grid with maximum number of newsworthy tweets is: **[28, 30]** with **269** tweets.

This grid index is very much closer to the full tweet dataset index with maximum tweets, which further illustrates that more tweets originating from a given location, there is higher chance of finding good quality information.

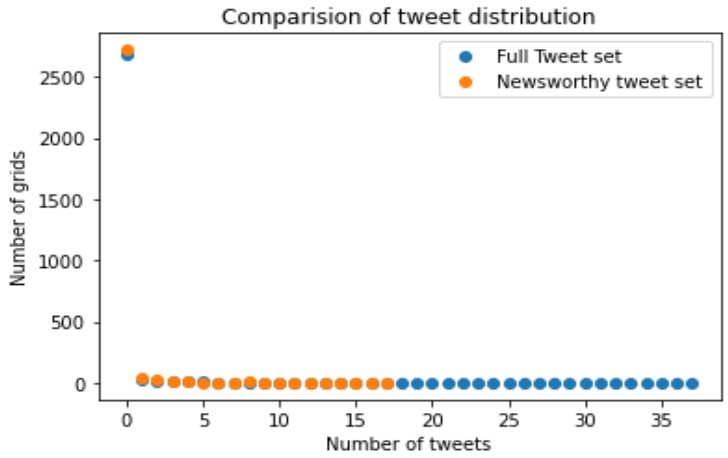
Using the grid index, the co-ordinates of the tweets can be easily captured, in turn helping to process information regarding that location faster.

Bar Graph of number of tweets vs. number of grids:



The distribution of number of tweets to number of grids is similar to the full tweet dataset where majority of the grids have tweet count of 0. The maximum number of tweets **269** is concentrated to one grid as mentioned above, followed by **241** tweets that is again concentrated in a single grid.

Comparison of the Tweet Distribution with Full Tweet Dataset and Newsworthy Tweet Dataset:



When compared both the datasets for tweet distribution, there is not much change in the patterns except for reduce in its intensity after eliminating the non-newsworthy content. This can be seen from the below graph.