ECMM443 Introduction to Data Science :

June 2022 Twitter Dataset Analysis

Introduction:

In this coursework, handling Twitter data for the month of June 2022 is required. The offered dataset is a zip file of approx. 9GB before extraction which is huge. The goal of this coursework is to effectively use Python libraries to handle this enormous data volume using the optimal strategy.

Data Read:

In [1]:
```
import json
import os
import pandas as pd
import zipfile

parent_directory = os.fsencode('S:\MSc Data Science with AI\IDS\Coursework\TwitterJune2022\TwitterJune2022')
file_count = 0
total_tweet = 0
tweet_dict = {"id": [], "text": [], "tweet_created_at": [], "timestamp_ms": []}

for folder in os.listdir(parent_directory):
    folder_name = os.fsdecode(folder)
    with zipfile.ZipFile(parent_directory.decode("utf-8") + '/' + folder_name) as zip_archive:
        for item in zip_archive.filelist:
            print(item.filename)
            json_file_data = zip_archive.read(item.filename)
            json_file_data_string = json_file_data.decode("utf-8")
            json_file_data_list = json_file_data_string.split("\n")

            for tweet in json_file_data_list:
                if tweet:
                    total_tweet += 1
                    json_tweet = json.loads(tweet)
                    if 'id_str' in json_tweet and 'created_at' and 'timestamp_ms' in json_tweet :
                        tweet_dict['id'].append(json_tweet['id_str'])
                        tweet_dict['text'].append(json_tweet['text'])
                        tweet_dict['tweet_created_at'].append(json_tweet['created_at'])
                        tweet_dict['timestamp_ms'].append(json_tweet['timestamp_ms'])

            # Capturing high level tweet statistics
            file_count += 1
```

```
geoEurope/geoEurope_2022060100.json
geoEurope/geoEurope_2022060101.json
geoEurope/geoEurope_2022060102.json
geoEurope/geoEurope_2022060103.json
geoEurope/geoEurope_2022060104.json
geoEurope/geoEurope_2022060105.json
geoEurope/geoEurope_2022060106.json
geoEurope/geoEurope_2022060107.json
geoEurope/geoEurope_2022060108.json
geoEurope/geoEurope_2022060109.json
geoEurope/geoEurope_2022060110.json
geoEurope/geoEurope_2022060111.json
geoEurope/geoEurope_2022060112.json
geoEurope/geoEurope_2022060113.json
geoEurope/geoEurope_2022060114.json
```

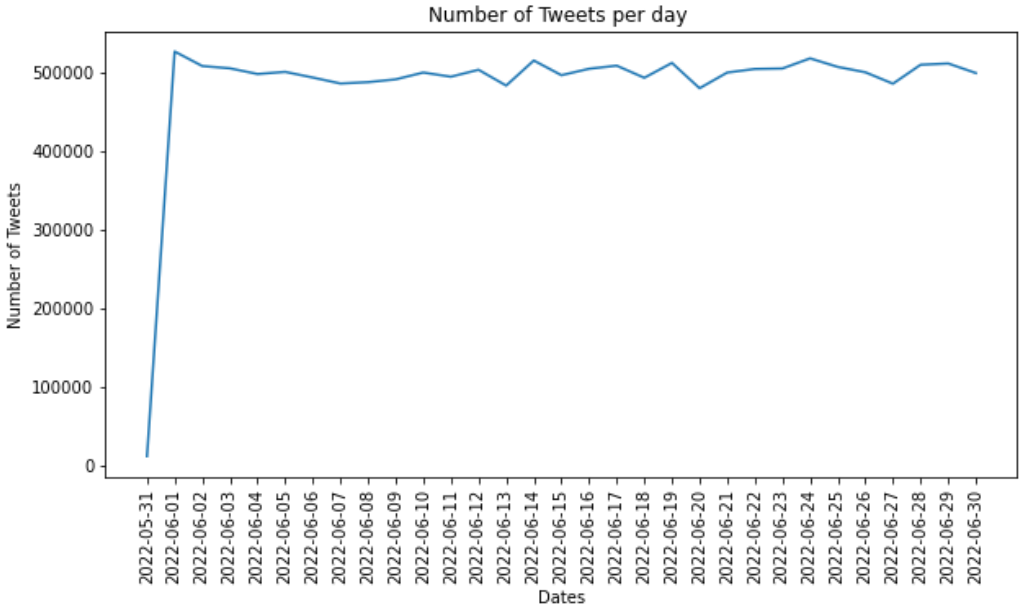
Part 1:

1. There were 15040386 tweets in all. This is the output obtained by simply reading the files, ignoring duplicates and NAN values which have no impact on our conclusions. After removing the NAN values using the unique function, there were 15033548 unique tweets in total. Duplicates that were eliminated (6838 tweets) using the "id_str" argument is specific to each tweet. Thus, that line was skipped whenever the word "id_str" appeared more than once.

2. From the given dataset, after plotting a basic time-series graph between date and number of tweets, the following are the conclusions:

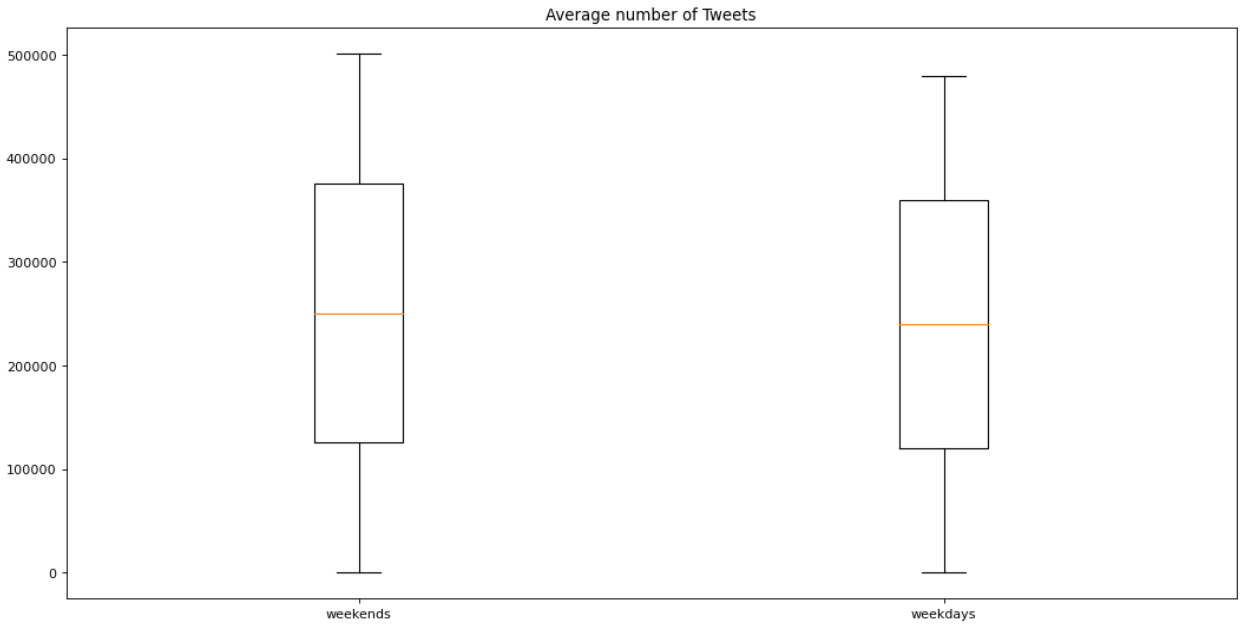
- Average tweets for the month of June 2022 were 480000.
- There were a couple of spikes going up resulting in the fact that something important happened on June 14th, 18th, 24th and 29th.
- The least number of tweets for the month of June 2022 was observed on 19th of the same month.

```
dates = []
daily_tweets = []
for a,b in tweets_per_day.items():
    dates.append(a)
    daily_tweets.append(b)

figure = plt.figure(figsize=(21, 8))
plt.plot(dates, daily_tweets)
plt.xlabel("Dates")
plt.ylabel("Number of Tweets")
plt.xticks(dates, rotation ="vertical")
plt.title("Number of Tweets per day ")
plt.show()
```



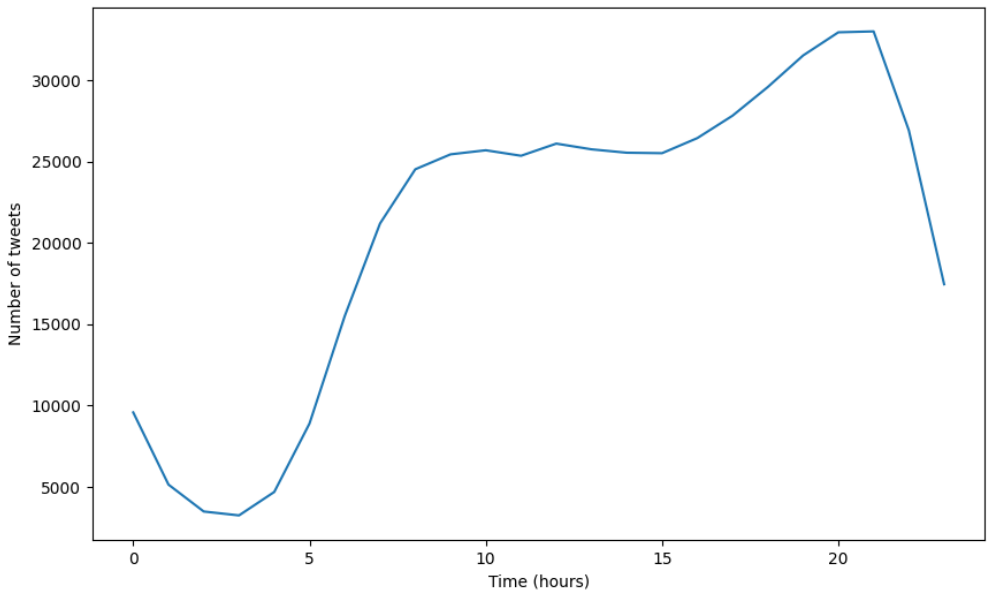
3. The average number of tweets observed on weekdays was 501135.125 and on weekends were 479622. Looking at the plots itself, we can say that 25th, 50th (median) and 75th percentiles were similar. One conclusion we can obtain from the above statement is that the number of tweets (irrespective of the day) were more or less constant. After looking at the tails of the box plots, we can conclude that average number of tweets over the weekends (50000) when compared to the weekday (48000) were statistically different.

```
weekdays = 0
weekend_days = 0
avg_val = []

for i in Days:
    if i.weekday() > 4:
        weekend_days += 1
    else:
        weekdays += 1

labels, data = no_tweets.keys(), no_tweets.values()
count = 0
for val in data:
    if count == 0:
        avg_val.append(val/weekend_days)
    else:
        avg_val.append(val/weekdays)
    count = 1

print(weekdays)
print(weekend_days)

print(avg_val)

values = [[weekend_days, avg_val[0]], [weekdays, avg_val[1]]]
plt.figure(figsize = (16, 8), dpi = 80)
plt.boxplot(values)
plt.xticks(range(1, len(labels) + 1), labels)
plt.show()
```



4. The below graph does not show any single trend but varies according to the time of the day. The least number of tweets during the month is between 12am and 5am. We can see a gradual increase in the number of tweets after 5am till 10am with the peak for this interval being around 9:30am-10am. During the mid-day, the number of tweets is usually constant with no sudden increase or decreases. The most active time is usually during the evening after 4pm with the peak (roughly 35000 tweets) being at 8pm. We can see an obvious decrease after around 9pm which is usually the average bedtime for the European region.

```
no_tweets = {'weekends': 0, 'weekdays': 0}
weekdays = []

df['tweet_created_at'] = pd.to_datetime(df['tweet_created_at'])
df['tweet_created_at'] = df['tweet_created_at'].dt.date
for i in df['tweet_created_at']:
    if i.weekday() > 4:
        no_tweets['weekends'] += 1  #weekend
    else:
        weekdays.append(i)
        no_tweets['weekdays'] += 1  #weekday

for i in range(len(df)):
    for j in range(len(weekdays)):
        if df['tweet_created_at'][i] == weekdays[j]:
            new_df = df.iloc[i]

def f(x):
    df_temp = x['id'].count()
    return pd.Series(dict(no_tweets = x['id'].count()))

hourly_count = df.groupby(df.index.hour).apply(f)
print (len(hourly_count))

hourly_plot = hourly_count['Number_of_tweets'].plot(kind='line')
hours = list(range(1,25))
xticks(np.arange(24), hours, rotation = 0, fontsize = 9)
xticks(fontsize = 9)
yticks(fontsize = 9)
hourly_plot.set_xlabel('Time (Hours)', weight = 'bold', labelpad = 15)
hourly_plot.set_ylabel('Number of Tweets', weight = 'bold', labelpad = 15)
daily_plot.tick_params(axis = 'x', pad = 5)
```

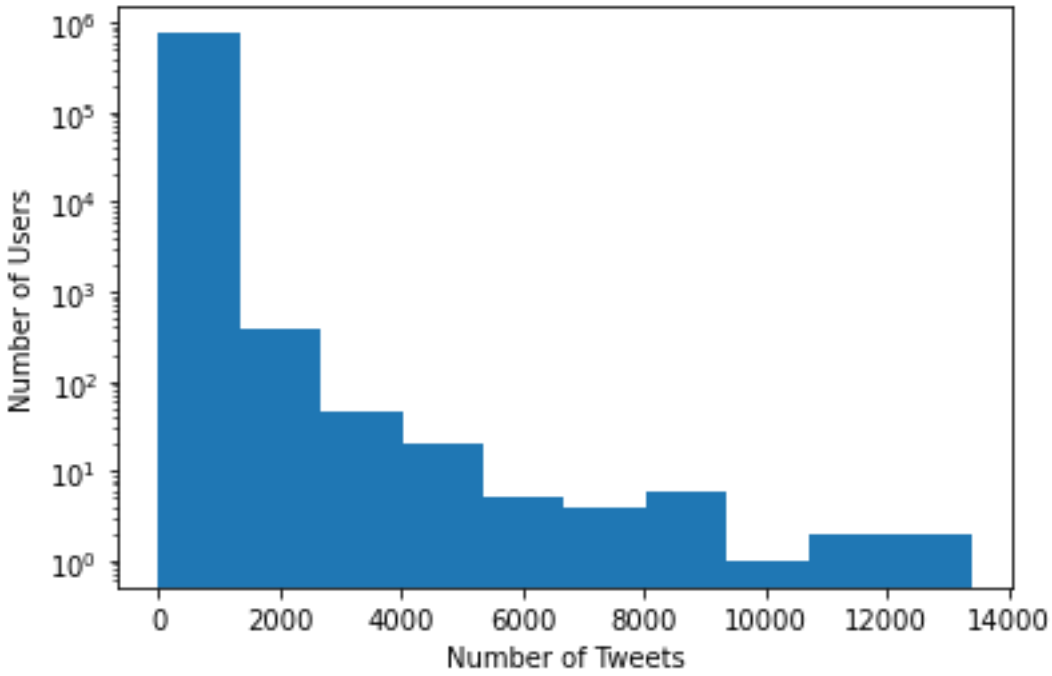


Part 2:

1. I did the log transformation for the number of users per tweet as there was a single bar with a large number of outliers in the bar plot. After doing the log transformation for the variable "user", we obtain the below histogram. There are few issues with the histogram itself:

- The graph is right skewed (mean > median)
- There are a large number of outliers even after log transforming the variable.
- The percentage of people who were less active, thereby, the number of tweets being low is greater than percentage of people who were most active on the platform during the month of June.

```
user_tweets = pd.DataFrame(df['user'].value_counts().reset_index())
user_tweets.columns = ['user', 'tweet_count']
plt.hist(user_tweets['tweet_count'], log = True)
plt.xlabel('Number of Tweets')
plt.ylabel('Number of Users')
```



2. The below output shows the top 5 users for the month of June 2022 based on the number of tweets each user made. According to my research (Reference attached at the bottom of the report), the average number of tweets by a human user is 800. Below we can see that the top 5 users are ranging between 10000 and 13500 tweets per month. This is a red flag when compared to a human user. Assuming each tweet takes about 2 minutes, the number of hours required to send out 13500 tweets is about 445 hours (given that June only has 720 hours). Also, from the above histogram and the time series graph we saw that the most number of users are active between 4pm-8pm. It is practically impossible to send out 14000 tweets a month when a user is only active during the evening for 4 hours. Hence, I fail to reject the hypothesis that the top 5 use rs are bots.

```
top5 = Counter(user).most_common(5)
top5
```

```
[(1402775770481713157, 13382),
 (1491316688549253120, 12533),
 (954732087235563520, 11632),
 (1384110594, 11305),
 (1069583615762288642, 10087)]
```

3. The top 5 people who received the most mentions had a diverse range of mentions, which could indicate that bots generated tweets get more mentions than tweets from real users. Accounts controlled by bots may be found using this method, and their data could be analysed.

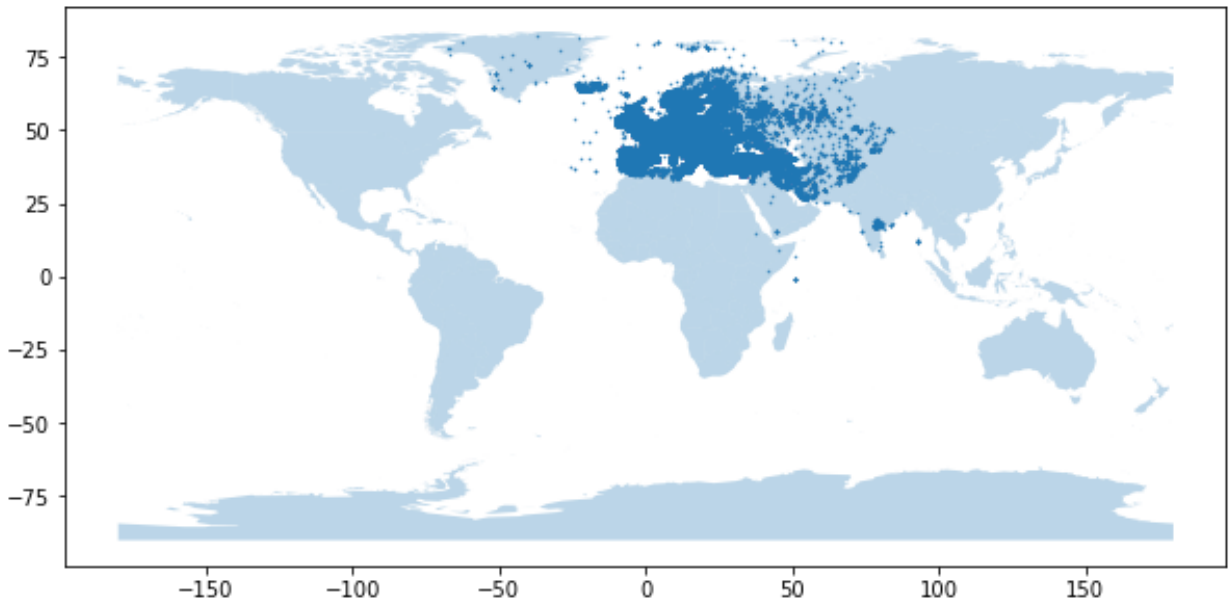
```
mention = Counter(mentioned_user)
mention.most_common(5)
```

```
[(10228272, 15542),
 (68034431, 6615),
 (2866804900, 5460),
 (1503799593405800449, 5413),
 (3131144855, 5385)]
```

Part 3:

1. This is a world map showing all the tweets, which are primarily dispersed around Europe with a few tweets around the world.

```
In [56]: map = gpd.read_file('TM_WORLD_BORDERS_SIMPL-0.3.shp')

In [57]: geometry = [Point(xy) for xy in zip(coor_data['longitude'], coor_data['latitude'])]

In [58]: geo_df = gpd.GeoDataFrame(coor_data, geometry = geometry)

In [59]: fig, ax = plt.subplots(figsize = (20, 10))
         map.plot(ax= ax, alpha = 0.5)
         geo_df.plot(ax = ax, markersize = 0.2)
```

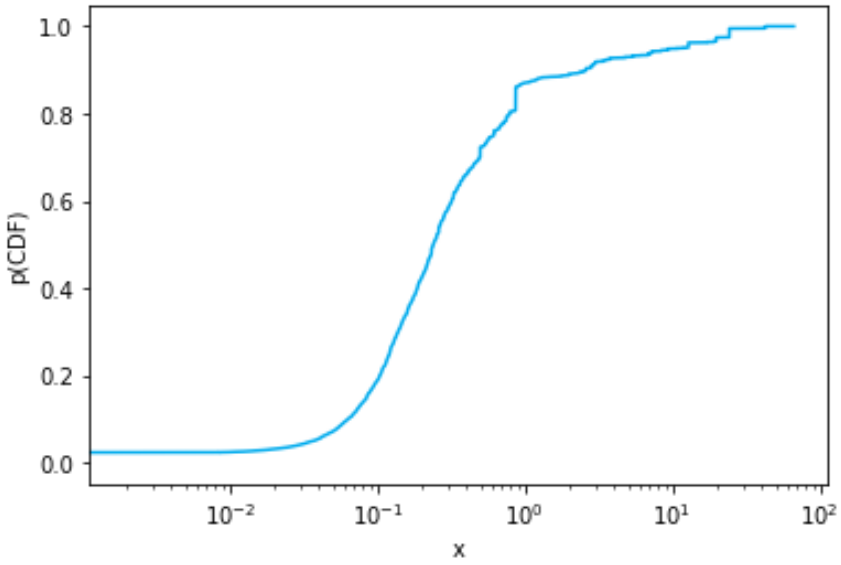


2. The correlation seen in the map indicates that the tweets decrease as the plot becomes more dispersed. Dark blue highlights the entire continent of Europe to show how many tweets were received from there. Compared to eastern European countries, central European countries tweet more frequently. The western section of Russia, however, has considerably less tweets than its counterparts in Europe. Small section of India also used the twitter during the June 2022 month. The surrounding island near the UK has a lesser average number of tweets due to their population.

3. The Cumulative Distribution Function was calculated using the bounding box element and the 2d coordinates to determine the Euclidean distance between the diagonal coordinates from the place tag's bounding box. The CDF for the number of matching tweets as the box size increases is shown in this graph. We took the place tag's tweets and utilised them as the tweets in the bounding box.

```
bound_box_diag = []
for index, tweet in tweets_df.iterrows():
    json_str = ast.literal_eval(tweet['place_bounding_box'])
    top_left = json_str['coordinates'][0][1]
    bottom_right = json_str['coordinates'][0][3]
    Euclid_dist = math.dist(top_left, bottom_right)
    bound_box_diag.append(Euclid_dist)
```

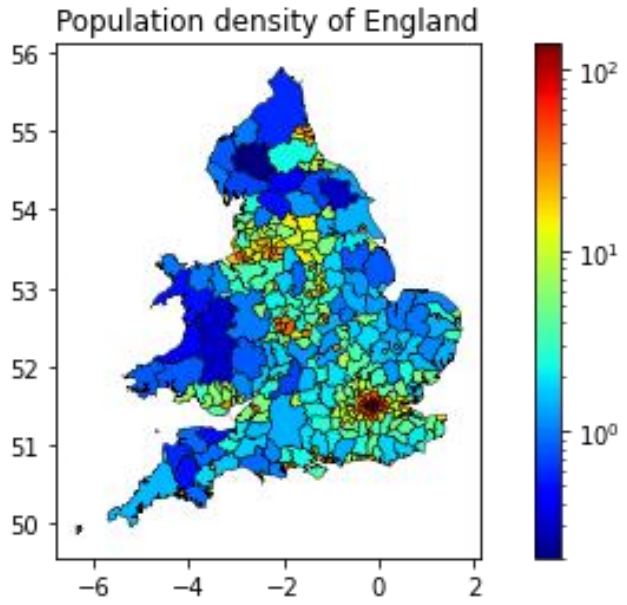
```
diag_data = bound_box_diag
p = 1. * np.arange(len(diag_data)) / (len(diag_data) - 1)
plt.plot(sorted(diag_data), p, color = '#00acee')
plt.xscale('log')
plt.xlabel('x')
plt.ylabel('p(CDF)')
```



4. I took England's twitter dataset as my additional spatial dataset and have plotted a population density/heatmap for the number of active users on twitter for the month of December 2015. London, capital of England has the most number of active twitter users (about 100 million). This is expected because it is the most populated area of England and the most visited city in England. After London, we have cities like Birmingham and Manchester in the race for most number of twitter users. Surrounding cities from London, Brighton, Oxford and Cambridge also have about 10 million active twitter users. The least number of active users on the platform for England are the north and southwest part (including Wales), resulting about 1 million active users. One of the reasons for the drop in number of users in compared to big cities like London and Manchester could be, the settlement of retired professionals in the north and southwest part of England. These numbers could have changed from 2015 till date.

```
place = {'code':[], 'name':[], 'density':[]}
with open('KS101EW_LA.csv','r') as csvfile:
    read = csv.reader(csvfile, delimiter = ',', quotechar = '"')
    next(read)
    for row in read:
        place['code'].append(row[2])
        place['name'].append(row[1])
        place['density'].append(float(row[-1]))
place_df = pd.DataFrame(place)
```

```
with open("S:\MSc Data Science with AI\IDS\Coursework\Local_Authority_Districts__December_2015__Boundaries.kml.xml",'r',
          encoding='utf-8') as myfile:
    doc = myfile.read().encode('utf-8')

k = kml.KML()
k.from_string(doc)
document = list(k.features())
folder = list(document[0].features())
places = list(folder[0].features())
code_to_polygon = {}
for p in places:
    poly = p.geometry
    for d in p.extended_data.elements[0].data:
        if d['name'] == 'lad15cd':
            id_code = d['value']
        if d['name'] == 'lad15nm':
            name = d['value']
    code_to_polygon[id_code] = poly

geometry = [code_to_polygon[c] for c in place_df['code']]
gdf = gpd.GeoDataFrame(place_df, crs = 'EPSG:4326', geometry=geometry)
gdf.plot('density', legend=True, cmap = 'jet', edgecolor = 'k', linewidth = 0.4,
         norm = LogNorm(vmin = gdf['density'].min(), vmax = gdf['density'].max()), figsize=(8,4))
plt.title('Population density of England')
plt.show()
```



Part 4:

1. I extracted "tweet_created_at", "country" and "text" from the given json documents to a csv file. The countries selected are Ukraine, Norway and Germany. I used Pandas library to create dataframes for these countries. After this, I followed the process of Evolutionary Data Analysis and formatted the created dataframe. I used the max() function on the variable "text" to find out the unusual day with the maximum number of tweets. The use of idxmax() function helped me to find out the exact date of the unusual number of tweets.

UKRAINE:

```
Ukraine_df = pd.DataFrame(country_data, columns=['text' , 'country' , 'tweet_created_at'])
Ukraine_data = Ukraine_df.loc[Ukraine_df['country'] == 'Ukraine']
Ukraine_data
```

```
max_Ukraine = Ukraine_df['text'].max()
max_Ukraine
```

```
2332
```

```
Ukraine_df['text'].idxmax(skipna = True)
```

```
datetime.date(2022, 6, 27)
```

NORWAY:

```
Norway_df = pd.DataFrame(country_data, columns=['text' , 'country' , 'tweet_created_at'])
Norway_data = Norway_df.loc[Norway_df['country'] == 'Norway']
Norway_data
```

```
Norway_max = Norway_df['text'].max()
Norway_max
```

```
1459
```

```
Norway_df['text'].idxmax(skipna = True)
```

```
datetime.date(2022, 6, 25)
```

GERMANY:

```
Germany_cdf = pd.DataFrame(country_data, columns=['text' , 'country' , 'tweet_created_at'])
Germany_data = Germany_cdf.loc[Germany_cdf['country'] == 'Germany']
Germany_data
```

```
max_Germany = Germany_df['text'].max()
max_Germany
```

```
7818
```

```
Germany_df['text'].idxmax(skipna = True)
```

```
datetime.date(2022, 6, 24)
```

2.a. I extracted “tweet_created_at”, “country” and “text” from the given json documents to a csv file. I followed the process of Evolutionary Data Analysis, extracted the clean text and appended it to a list and used it in the word cloud function to draw the word cloud.

UKRAINE:

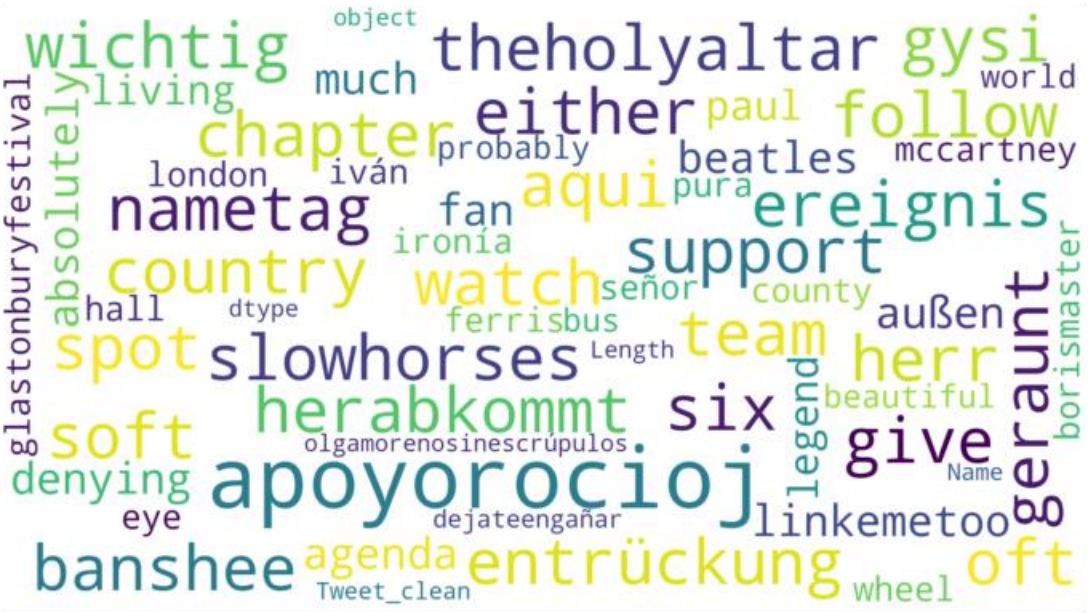
```
# Store the data in a list for text visualization
total_tweet = []
for x in cleaned_tweet_list:
    for y in split_name(x):
        total_tweet.append(y)
```

```
# Plot the word cloud
plt.figure(figsize = (20, 5))
words = ' '.join([twts for twts in total_tweet])
wordCloud = WordCloud(background_color = 'white', width=2000, height=1000, random_state=21, max_font_size=200).generate(words)

plt.imshow(wordCloud, interpolation = "bilinear")
plt.axis('off')
plt.show()
```

NORWAY:

```
# Store the data in a list for text visualization
total_tweet = []
for x in cleaned_tweet_list:
    for y in split_name(x):
        total_tweet.append(y)
```

```
# Plot the word cloud
plt.figure(figsize = (20, 5))
words = ' '.join([twts for twts in total_tweet])
wordCloud = WordCloud(background_color = 'white', width=2000, height=1000, random_state=21, max_font_size=200).generate(words)

plt.imshow(wordCloud, interpolation = "bilinear")
plt.axis('off')
plt.show()
```



GERMANY:

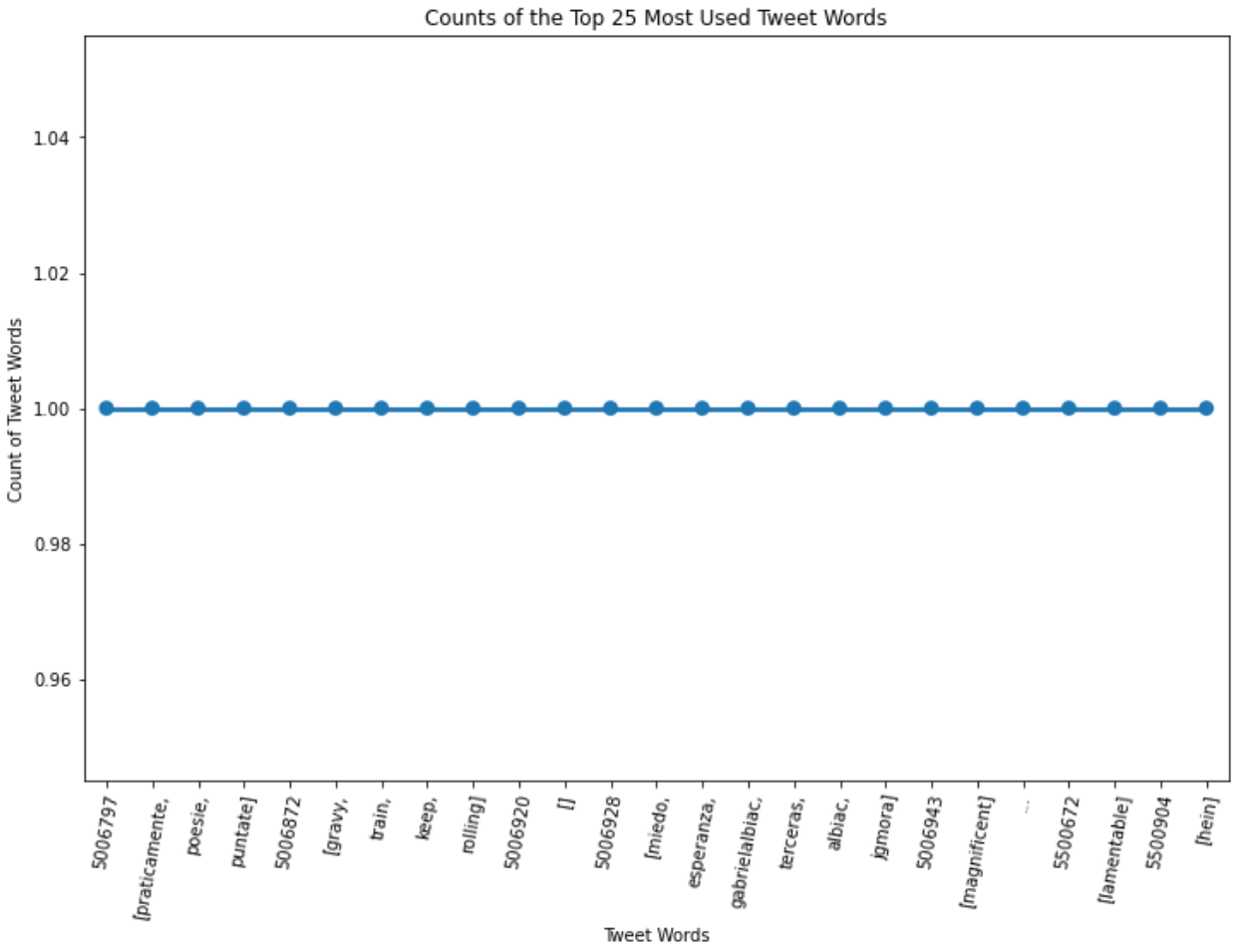
```
# Store the data in a list for text visualization
total_tweet = []
for x in cleaned_tweet_list:
    for y in split_name(x):
        total_tweet.append(y)
```

```
# Plot the word cloud
plt.figure(figsize = (20, 5))
words = ' '.join([twts for twts in total_tweet])
wordCloud = WordCloud(background_color = 'white', width=2000, height=1000, random_state=21, max_font_size=200).generate(words)

plt.imshow(wordCloud, interpolation = "bilinear")
plt.axis('off')
plt.show()
```

2.b. The other method for plotting I used is the Point plot from the seaborn library. I used the most_common () function to extract the most common words from the tweets and appended them to a list. To make my program run faster and for better analysis purposes, I restricted my list to top 25 most used words. It is a straightforward process after this where I converted my list to a pandas dataframe and used the point plot function alongside the counter function to plot the point graph of top 25 most used words in tweets.

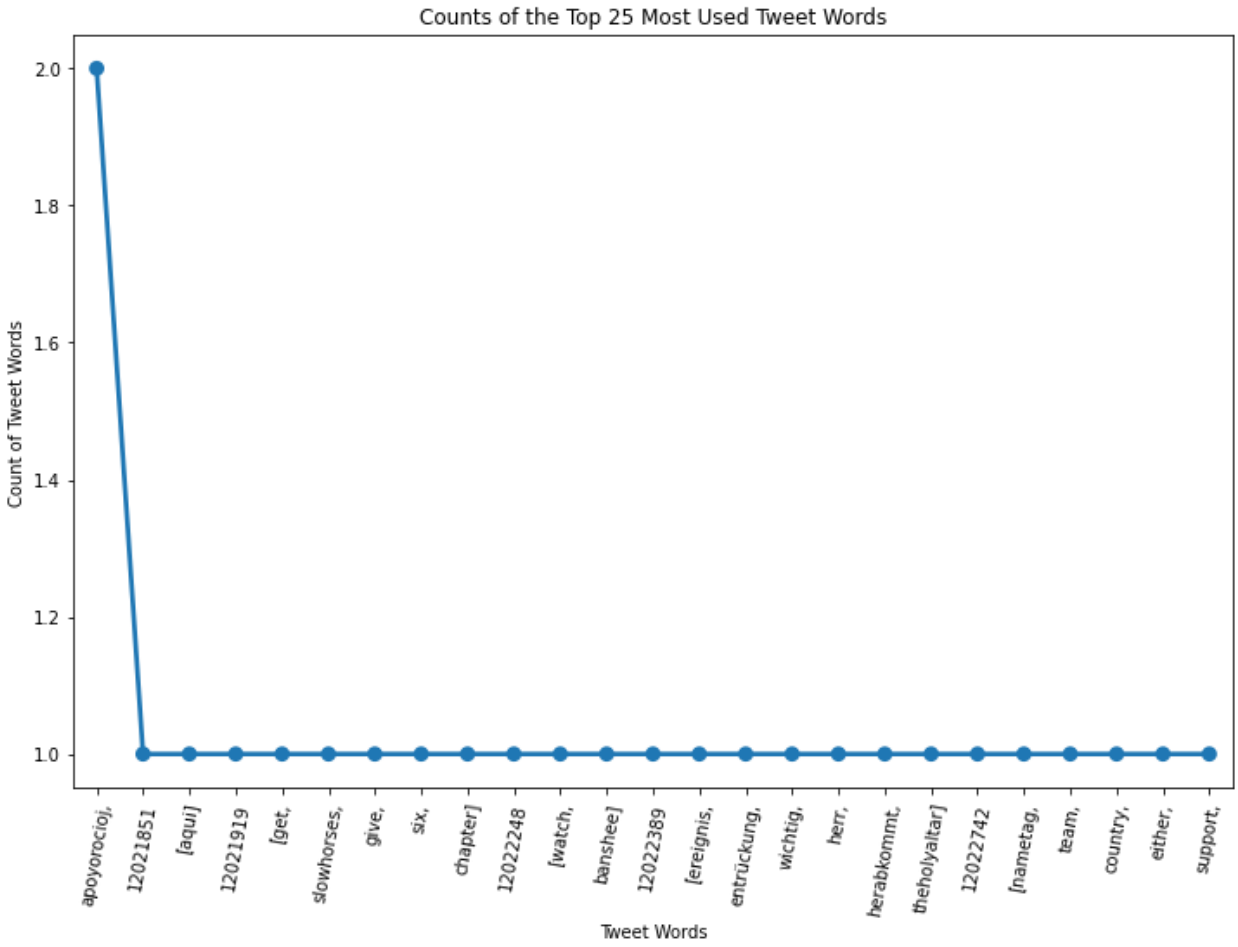
UKRAINE:

```
top25 = Counter(total_tweet).most_common()
top25 = top25[0:25]

bar_plot = pd.DataFrame(top25)
bar_plot.rename(columns = {0:'Tweet Words', 1:'Count'}, inplace=True)

plt.figure(figsize=(12,8))
plot = sns.barplot(x = 'Tweet Words', y = 'Count', data = bar_plot)
plot.set_title('Counts of the Top 25 Most Used Tweet Words')
plot.set_ylabel('Count of Tweet Words')
plot.set_xlabel('Tweet Words')
plot.set_xticklabels(plot.get_xticklabels(), rotation=80)
```



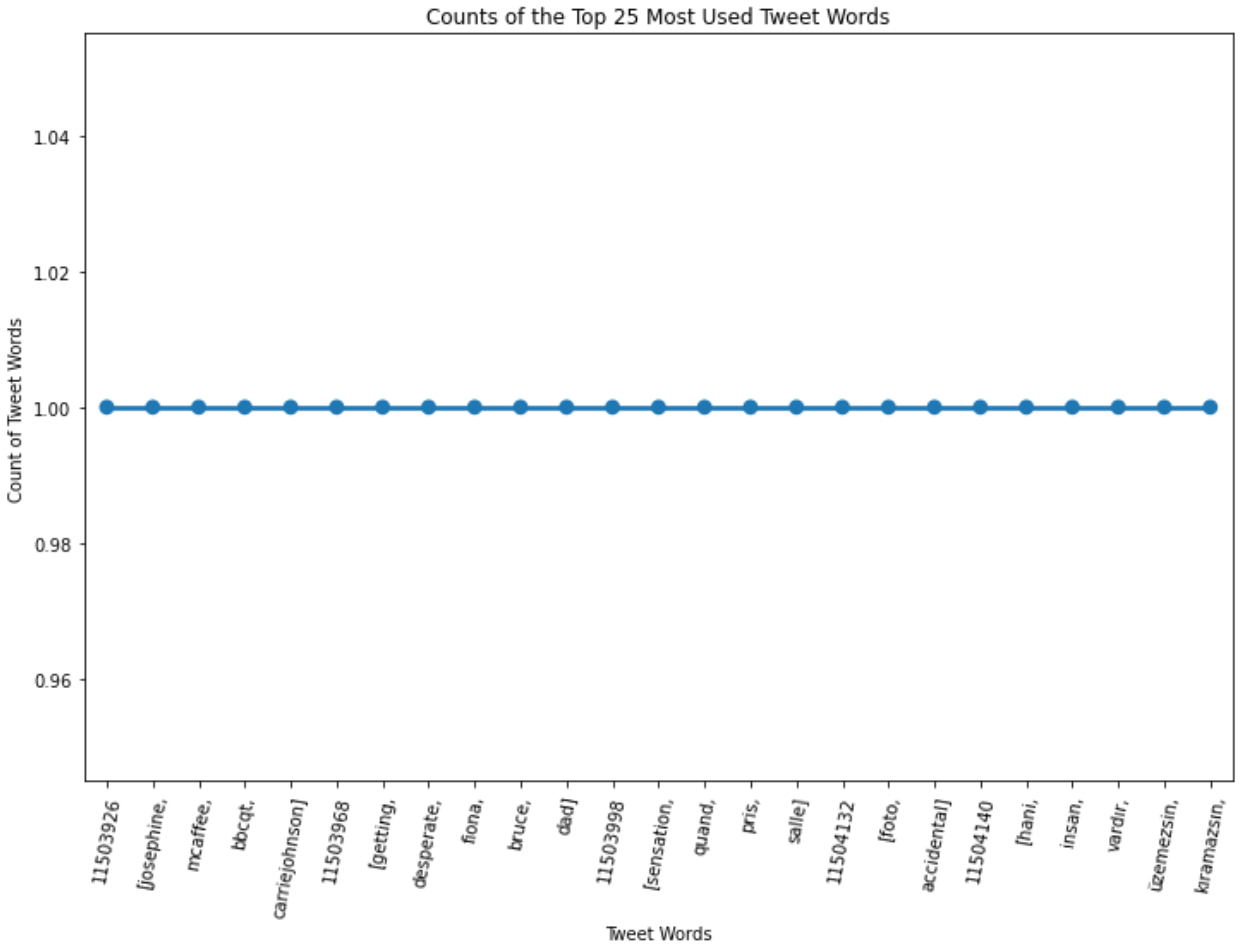
NORWAY:

```
top25 = Counter(total_tweet).most_common()
top25 = top25[0:25]

bar_plot = pd.DataFrame(top25)
bar_plot.rename(columns = {0:'Tweet Words', 1:'Count'}, inplace=True)

plt.figure(figsize=(12,8))
plot = sns.barplot(x = 'Tweet Words', y = 'Count', data = bar_plot)
plot.set_title('Counts of the Top 25 Most Used Tweet Words')
plot.set_ylabel('Count of Tweet Words')
plot.set_xlabel('Tweet Words')
plot.set_xticklabels(plot.get_xticklabels(), rotation=80)
```



GERMANY:

```
top25 = Counter(total_tweet).most_common()
top25 = top25[0:25]

bar_plot = pd.DataFrame(top25)
bar_plot.rename(columns = {0:'Tweet Words', 1:'Count'}, inplace=True)

plt.figure(figsize=(12,8))
plot = sns.barplot(x = 'Tweet Words', y = 'Count', data = bar_plot)
plot.set_title('Counts of the Top 25 Most Used Tweet Words')
plot.set_ylabel('Count of Tweet Words')
plot.set_xlabel('Tweet Words')
plot.set_xticklabels(plot.get_xticklabels(), rotation=80)
```



3. <u>UKRAINE:</u>

The highest number of tweets observed in Ukraine was 2332 on 27th June 2022. According to many articles and the tweets, we can confirm that on this day Russia did shelling of Kyiv, capital of Ukraine, thereby, starting the war between 2 countries.

<u>NORWAY:</u>

The highest number of tweets observed in Norway was 1459 on 25th June 2022. According to many articles and the tweets, we can confirm that on this day 2 people were killed and 21 people were wounded as a result of a mass shooting in Oslo, Norway. These killings took place at the Oslo LGBTQ pride event.

<u>GERMANY:</u>

The highest number of tweets observed in Germany was 7818 on 24th June 2022. According to many articles and the tweets, we can confirm that this day was the last of The Special Olympics National Games held in Berlin.

Part 5:

Use & Drawbacks: Twitter is being consumed every day in a multitude of ways by different people in different ways to address everyone's curiosity. For example, I use it to watch sports news and articles and follow the latest in EPL. One of the top things about Twitter is it's easy to consume and have a conversation with other people on what they think about an event, an article, day-to-day happening of something even so trivial as a meme or so important as elections. One other useful way especially in this age of no cable tv has been getting the latest information on natural disaster or traffic updates and news. With people who are reporting information on these events in almost real-time, makes things more transparent and more broadcasted from different people's mindset, something which could easily be missed in mainstream news.
However, with all the sharing comes the problem of misinformation at times. Or the fact that a lot of people on Twitter are from recent generation. It's hard to find people of our grandfather's time hence limiting the demographics and also the views of people are skewed from a particular generation which could make people draw erroneous conclusions. Another problem along similar lines are the fact that most tweets are done by a subset of people and others are more of a consumer and less contributor. This adds to views being skewed too. I think it's not an unknown at this point that there are a lot of bots on Twitter so data returned from bot tweets can be significant.

Ethical Concerns: Twitter has come out and mentioned that any tweet can be used in any context anywhere and is usually visible by the whole world. This creates a lack of private space for people to share their thoughts, possibly with people they want to share it with. Twitter API can access all the data of all tweets around the world, leaving it for organizations/researchers or even individuals to analyses tweets without users realizing that's happening. The organizations don't have consent of people they analyze tweets of. Before 2022, I had no idea this was happening either. Given the fact that tweets are not anonymized, it's not hard at all to find who tweeted something, their geo-location and possibly all history of the person. Do elders or children realize their tweet will be analyzed by hundreds of people? Most likely not. One more point here is people don't have an option to revoke or withdraw their data even if the user doesn't exist on Twitter anymore.

References:

1. (Feb 14, 2022) Python's Zipfile: Manipulate your ZIP Files Efficiently by Leodanis Pozo Ramos.
   https://realpython.com/python-zipfile/

2. (August 2, 2022) Twitter testing 'tweets per month' counter to display on profiles by Vishwam Sankaran.
   https://finance.yahoo.com/news/twitter-testing-tweets-per-month-073959753.html?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAKLt1CN6Qit4Xx6VRb4WVL1KK-JtR7fmjsHUZEsajE7gwk5c1vVAPWCEXqVKY07GtheHCsS6YBuMiL9F7XqbqbmHiHGJFPpNAWz7CHeo-Yq7GtxtOK1EZ4B80dceY_bd9Q714ED3FCXGoY1oXbubnGErUiA9lSfvuNs7yc_PE83m

3. Analyze Geospatial Data in Python: GeoPandas and Shapely by Loannis Prapas.
   https://www.learndatasci.com/tutorials/geospatial-data-python-geopandas-shapely/

4. Local Authority Districts (December 2015) Boundaries
   https://geoportal.statistics.gov.uk/maps/ons::local-authority-districts-december-2015-full-clipped-boundaries-in-great-britain/explore?location=55.197919%2C-2.950000%2C6.92

5. (Nov 2019) Generating WordClouds in Python Tutorial by Duong Vu.
   https://www.datacamp.com/tutorial/wordcloud-python

6. Ahmed, W., Bath, P. and Demartini, G. (2017) Chapter 4 Using Twitter as a Data Source: An Overview of Ethical, Legal, and Methodological Challenges. In: Woodfield, K., (ed.) The Ethics of Online Research. Advances in Research Ethics and Integrity (2). Emerald , pp. 79-107. ISBN 978-1-78714-486-6.
   https://eprints.whiterose.ac.uk/126729/8/Normal_-_Ethics_Book_Chapter_WA_PB_GD_Peer_Review_comments_implemented__1_.pdf