Proyecto2

September 10, 2021

0.1 Proyecto 2 - Ciencia de datos

Integrentes:

- Karina Valladares, 18005
- Alexa Bravo, 18831
- José Eduardo López, 181045

[52]:
```
#Importamos las Librerias a utilizar
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import neattext as nt
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import textdistance
import re
from collections import Counter
from sklearn.cluster import AgglomerativeClustering
```

[2]:
```
# leemos la data
data = pd.read_csv("train.csv")

data
```

[2]:
```
          id keyword location  \
0          1     NaN      NaN
1          4     NaN      NaN
2          5     NaN      NaN
3          6     NaN      NaN
4          7     NaN      NaN
…        …       …        …
7608   10869     NaN      NaN
7609   10870     NaN      NaN
7610   10871     NaN      NaN
7611   10872     NaN      NaN
7612   10873     NaN      NaN

                                                  text  target
```

```
0     Our Deeds are the Reason of this #earthquake M…       1
1                Forest fire near La Ronge Sask. Canada       1
2     All residents asked to 'shelter in place' are …       1
3     13,000 people receive #wildfires evacuation or…       1
4     Just got sent this photo from Ruby #Alaska as …       1
…                                                   …     …
7608  Two giant cranes holding a bridge collapse int…       1
7609  @aria_ahrary @TheTawniest The out of control w…       1
7610  M1.94 [01:04 UTC]?5km S of Volcano Hawaii. htt…       1
7611  Police investigating after an e-bike collided …       1
7612  The Latest: More Homes Razed by Northern Calif…       1

[7613 rows x 5 columns]
```

[3]:
```
# revisamos cuántos NA hay
data.isna().sum().to_frame()
```

[3]:
```
              0
id            0
keyword      61
location   2533
text          0
target        0
```

A partir del resumen anterior, es importante señalar que la cantidad de datos faltantes es sustancial para "location", por lo que esto necesitará un tratamiento especial.

0.1.1 Variables

- **id**: hace referencia a un número único para cada registro, es de tipo numérico (no necesita limpieza). Tenemos 0 datos faltantes.
- **Keyword**: es una palabra que caracteriza al tweet, no todos los tweets lo tienen pero no escencial para el análisis de texto, aunque puede resultar de gran ayuda para aquellos tweets que si estén vinculados a una. Es de tipo String y solo necesitamos remover los signos de puntuación y pasar todo a mayúsuculas. Tenemos 61 datos faltantes.
- **Location**: hace referencia al país desde el que se publicó el tweet, no todos los registros tienen esta variables pero puede ser de utilidad para los que si la tienen. Para la limpieza solo es necesario estandarizar los nombres y colocar todo en mayúsculas. Tenemos 2,533 datos faltantes.
- **text**: es el mensaje de texto adjunto al tweet, esta es la variable más importante y la quiere mayor limpieza. Como es una variable que contiene texto es necesaria procesar como lenguaje natural, quitar caracteres especiales, correos, números de teléfono, stopwrods, signos de puntuación, pasar todo a mayúsculas, remover URL, entre otros. Tenemos 0 datos faltantes
- **target**: es una variable binaria que clasifica si el tweet es sobre un desastre real: 1, 0 en caso contrario, no requiere limpieza. Tenemos 0 datos faltantes.

0.1.2 Análisis Exploratorio

WordClouds

Keyword

[4]:
```
# convertimos todo a un string
final_string = ""
for i in data["keyword"]:
    final_string += str(i) + " "
final_string = final_string[:-1]
```

[5]:
```
#WordCloud "keyword"
wordcloud = WordCloud().generate(str(final_string))

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

Podemos observar que aparecen valores repetidos y varios “NaN” por ello es necesario hacer limpieza y estandarizar los datos.

Location

[6]:
```
# convertimos todo a un string
final_string = ""
for i in data["location"]:
    final_string += str(i) + " "
final_string = final_string[:-1]
```

[7]:
```
#WordCloud "location"
wordcloud = WordCloud().generate(str(final_string))

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

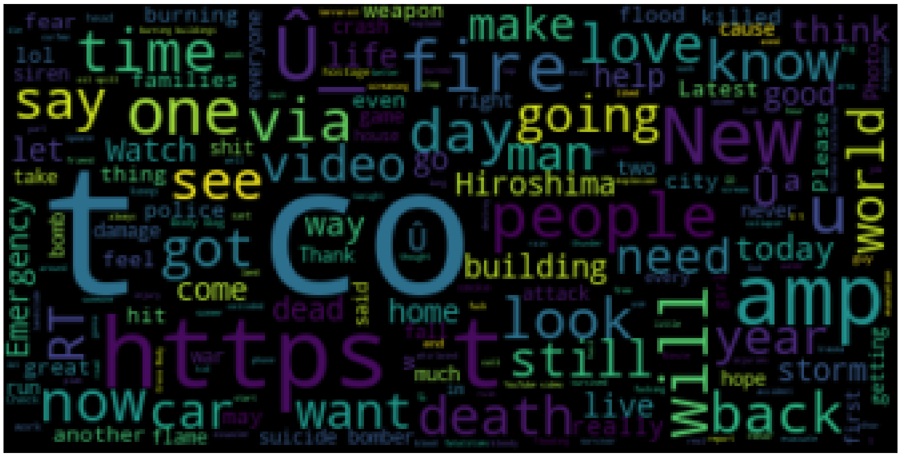
```
plt.show()
```

Podemos observar que aparecen valores repetidos y varios “NaN” por ello es necesario hacer limpieza y estandarizar los datos.

Text

[8]:
```
# convertimos todo a un string
final_string = ""
for i in data["text"]:
    final_string += str(i) + " "
final_string = final_string[:-1]
```

[9]:
```
#WordCloud "text"
wordcloud = WordCloud().generate(str(final_string))

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

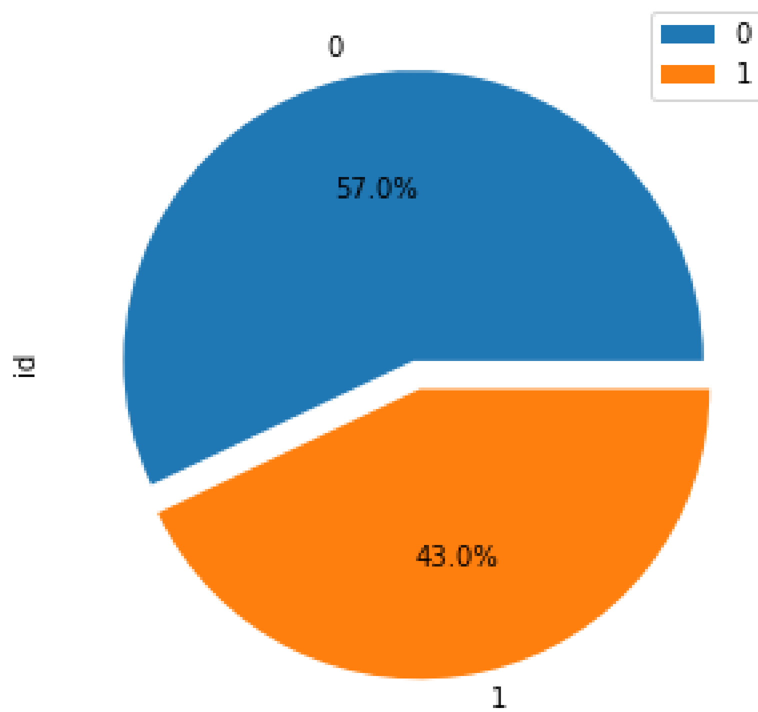


Target

```
[10]: pie = data.groupby("target").count()
```

```
[11]: pie.plot.pie(y='id', figsize=(5, 5), autopct='%1.1f%%', explode=(0, 0.1))
```

```
[11]: <AxesSubplot:ylabel='id'>
```



Podemos observar que un poco mas de la mitad de tweets no son sobre un desastre real.

0.1.3 Funciones

[12]:
```
#Función para convertir todo a Mayúsculas
def convertir_mayus(X):

    return X.str.upper()

#Función para remover stopwords, emails, números, números de telefono, url's,␣
 ↪emojis, tags, caracteres especiales, signos de puntuación, ets.
def limpiar(X):
    for i in range(len(X)):
        X[i] = nt.TextFrame(str(X[i])).remove_stopwords(lang = "en")
        X[i] = nt.TextFrame(str(X[i])).remove_emails()
        X[i] = nt.TextFrame(str(X[i])).remove_numbers()
        X[i] = nt.TextFrame(str(X[i])).remove_phone_numbers()
        X[i] = nt.TextFrame(str(X[i])).remove_btc_address()
        X[i] = nt.TextFrame(str(X[i])).remove_urls()
        X[i] = nt.TextFrame(str(X[i])).remove_emojis()
        X[i] = nt.TextFrame(str(X[i])).remove_html_tags()
        X[i] = nt.TextFrame(str(X[i])).remove_puncts()
        X[i] = nt.TextFrame(str(X[i])).remove_special_characters()
        X[i] = nt.TextFrame(str(X[i])).remove_stopwords(lang = "es")

    return X

#Funcion para normalizar los textos
def normalizar(text):
    return re.sub('[^a-z0-9]+', ' ', text.lower())

def preNormText(texts, threshold=0.4):

    normTexts = np.array([normalizar(text) for text in texts])
    distancia = 1 - np.array([[textdistance.jaro_winkler(one, another) for one␣
 ↪in normTexts] for another in normTexts])
    clustering =␣
 ↪AgglomerativeClustering(distance_threshold=threshold,affinity="precomputed",␣
 ↪linkage="complete", n_clusters=None).fit(distancia)
    center = dict()
    for clusterId in set(clustering.labels_):
        index = clustering.labels_ == clusterId
        central = distancia[:, index][index].sum(axis=1)
        center[clusterId] = normTexts[index][central.argmin()]
    return [center[i] for i in clustering.labels_]
```

[13]:
```
#Funcion para limpiar NA
def limpiar_na(X):
    resultado = []
    for i in range(len(X)):
        if str(X[i]) == "NAN":
            resultado.append("NO DATA")
        else:
            resultado.append(X[i])
    return resultado
```

0.1.4 Limplieza de Datos

Keyword

[14]:
```
keyword = data["keyword"]
```

Pasamos todo a mayúsculas

[15]:
```
keyword = convertir_mayus(keyword)
```

Removemos Stopwords, emails, números, números de teléfono, direcciones. URL, emojis, HTML tags

[16]:
```
keyword = limpiar(keyword)
```

[17]:
```
data["keyword"] = keyword.astype(str)
```

[18]:
```
data.groupby("keyword").count()
```

[18]:

keyword	id	location	text	target
ABLAZE	36	29	36	36
ACCIDENT	35	28	35	35
AFTERSHOCK	34	25	34	34
AIRPLANEACCIDENT	35	29	35	35
AMBULANCE	38	26	38	38
…	..	…	…	…
WOUNDS	33	27	33	33
WRECK	37	27	37	37
WRECKAGE	39	28	39	39
WRECKED	39	22	39	39
nan	61	0	61	61

[222 rows x 4 columns]

Normalización

[19]:
```
keyword = data["keyword"]
```

[20]:
```
objetivo = preNormText(keyword, 0.6)
```

[21]:
```
# sobreescribimos la columna en el dataframe
data["keyword"] = objetivo
```

[22]:
```
# convertimos el texto a mayúsculas
data["keyword"] = data["keyword"].str.upper()
```

Limpieza de NA

[23]:
```
# extraemos en una variable para trabajar sobre ella
Y = data["keyword"]
```

[24]:
```
# limpiamos los na
Y = limpiar_na(Y)
```

[25]:
```
# reemplazamos los espacios por "_"
for i in range(len(Y)):
    word = Y[i]
    new = word.replace(" ", "_")
    Y[i] = new
```

[26]:
```
# sobreescribimos en el dataframe
data["keyword"] = Y
```

[27]:
```
# agrupamos según la ubicación, para ver el resultado y sus cambios
data.groupby("keyword").count()
```

[27]:

keyword	id	location	text	target
ARSONIST	292	161	292	292
BOMBING	549	367	549	549
COLLAPSE	256	197	256	256
CRASHED	381	247	381	381
DEATH	451	314	451	451
DEMOLISHED	439	297	439	439
DROWN	245	168	245	245
ELECTROCUTE	243	155	243	243
EMERGENCY	281	194	281	281
EVACUATE	363	246	363	363
FAMINE	332	231	332	332
FATALITIES	464	324	464	464
FLOOD	359	233	359	359
HAIL	240	172	240	240
HAZARD	213	137	213	213
OBLITERATE	395	260	395	395
RAZED	385	257	385	385
RESCUE	294	184	294	294
SURVIVE	347	208	347	347
TORNADO	344	217	344	344

```
WILDFIRE     198       129   198     198
WINDSTORM    327       245   327     327
WRECK        215       137   215     215
```

Location

[28]:
```
location = data["location"]
```

Pasamos todo a mayúsculas

[29]:
```
location = convertir_mayus(location)
```

Removemos Stopwords, emails, números, números de teléfono, direcciones. URL, emojis, HTML tags

[30]:
```
location = limpiar(location)
```

[31]:
```
data["location"] = location.astype(str)
```

[32]:
```
data.groupby("location").count()
```

[32]:
```
                          id  keyword  text  target
location
                         302      302   302     302
A PROPERTY UNIVERSE        1        1     1       1
AARHUS CENTRAL JUTLAND     1        1     1       1
AAS AZTEC PRINCESS         1        1     1       1
AB CANADA                  1        1     1       1
…                        …        …     …       …
ZERO BRANCO                1        1     1       1
ZIAM AF                    1        1     1       1
ZIMBABWE                   1        1     1       1
ZONE LAYER                 1        1     1       1
nan                     2533     2533  2533    2533

[2805 rows x 4 columns]
```

Normalización

[33]:
```
location = data["location"]
```

[34]:
```
# limpiamos con la función definida
resultado = preNormText(location, 0.6)
```

[35]:
```
# sobreescribimos la columna en el dataframe
data["location"] = resultado
```

[36]:
```
# convertimos el texto a mayúsculas
data["location"] = data["location"].str.upper()
```

Limpieza de NA

[37]:
```
# extraemos en una variable para trabajar sobre ella
X = data["location"]
```

[38]:
```
# limpiamos los na
X = limpiar_na(X)
```

[39]:
```
# reemplazamos los espacios por "_"
for i in range(len(X)):
    word = X[i]
    new = word.replace(" ", "_")
    X[i] = new
```

[40]:
```
# sobreescribimos en el dataframe
data["location"] = X
```

[41]:
```
# agrupamos según la ubicación, para ver el resultado y sus cambios
data.groupby("location").count()
```

[41]:

location	id	keyword	text	target
	302	302	302	302
AIRES_ARGENTINA	32	32	32	32
ALBERTA	56	56	56	56
AMERICA	20	20	20	20
AMERICAN_WASTELAND_MV	20	20	20	20
…	…	…	…	…
WELLINGTON	19	19	19	19
WEST_COAST	23	23	23	23
WILLIAMSBURG	9	9	9	9
WINTER_PARK_COLORADO	19	19	19	19
WORLDWIDE	89	89	89	89

[179 rows x 4 columns]

Text

[42]:
```
text = data["text"]
```

Pasamos todo a mayúsculas

[43]:
```
text = convertir_mayus(text)
```

Removemos Stopwords, emails, números, números de teléfono, direcciones. URL, emojis, HTML tags

[44]:
```
text = limpiar(text)
```

[45]:
```
data["text"] = text.astype(str)
```

[46]:
```
data.groupby("text").count()
```

[46]:
```
                                                    id  keyword  location  \
text
A NIGHTMARE ELM STREET GETTING REMADE                 1        1         1
A RADAR UPDATE WIDESPREAD SHOWERSSTORMS MOVING …      1        1         1
A SLAMMING DOOR LESSON LEARNED LET LOVER CRASH …      1        1         1
A STARTED WRITING TALK TRAUMA THERAPY WAY COMMU…      1        1         1
A VOLUNTARY EVACUATION RECOMMENDED TIME PICKERE…      1        1         1
…                                                    ..       …         …
ZOUMA ABSOLUTELY FLATTENED GUY                        1        1         1
ZOUMA FLATTENED CFC                                   1        1         1
ZOUMA FLATTENED GUY                                   1        1         1
ZOURRYART FORGOT ADD BURNING BUILDINGS SCREAMIN…      1        1         1
ZXATHETIS ARE OKAY ELECTROCUTE BADLY RIGHT            1        1         1

                                                    target
text
A NIGHTMARE ELM STREET GETTING REMADE                    1
A RADAR UPDATE WIDESPREAD SHOWERSSTORMS MOVING …         1
A SLAMMING DOOR LESSON LEARNED LET LOVER CRASH …         1
A STARTED WRITING TALK TRAUMA THERAPY WAY COMMU…         1
A VOLUNTARY EVACUATION RECOMMENDED TIME PICKERE…         1
…                                                       …
ZOUMA ABSOLUTELY FLATTENED GUY                           1
ZOUMA FLATTENED CFC                                      1
ZOUMA FLATTENED GUY                                      1
ZOURRYART FORGOT ADD BURNING BUILDINGS SCREAMIN…         1
ZXATHETIS ARE OKAY ELECTROCUTE BADLY RIGHT               1

[6854 rows x 4 columns]
```

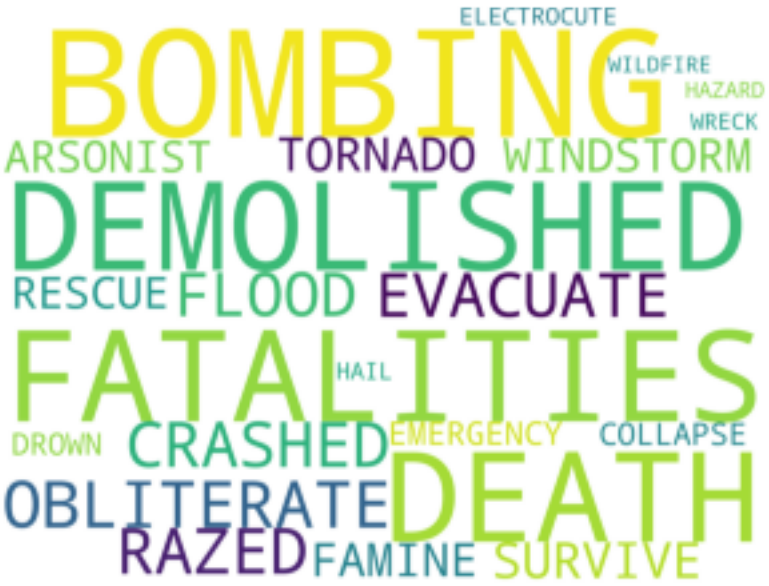
0.1.5 Análisis exploratorio de los datos más limpios

WordClouds

[47]:
```
kwText = " ".join(review for review in data.keyword.astype(str))
```

[48]:
```
wordcloud = WordCloud(background_color="white", width=4200,␣
↪height=3200,collocations = False,
                          stopwords=STOPWORDS).generate(kwText)
plt.axis("off")
plt.imshow(wordcloud)
```

[48]:
```
<matplotlib.image.AxesImage at 0x2ae96a10040>
```



[49]:
```
# convertimos todo a un string
lText = " ".join(review for review in data.location.astype(str))
```

[50]:
```
wordcloud = WordCloud(background_color="white", width=4200,␣
 ↪height=3200,collocations = False,
                          stopwords=STOPWORDS).generate(lText)
plt.axis("off")
plt.imshow(wordcloud)
```

[50]:
```
<matplotlib.image.AxesImage at 0x2ae9c1343a0>
```

Frecuencia de Palabras

Keyword

[79]:
```
kwFrequencies = Counter(kwText.split())
```

[80]:
```
kwFrequencies
```

[80]:
```
Counter({'ARSONIST': 292,
         'RAZED': 385,
         'FAMINE': 332,
         'EMERGENCY': 281,
         'ELECTROCUTE': 243,
         'DEMOLISHED': 439,
         'FATALITIES': 464,
         'OBLITERATE': 395,
         'BOMBING': 549,
         'HAIL': 240,
         'HAZARD': 213,
         'FLOOD': 359,
         'WILDFIRE': 198,
         'CRASHED': 381,
         'COLLAPSE': 256,
         'DEATH': 451,
         'RESCUE': 294,
         'TORNADO': 344,
         'DROWN': 245,
```

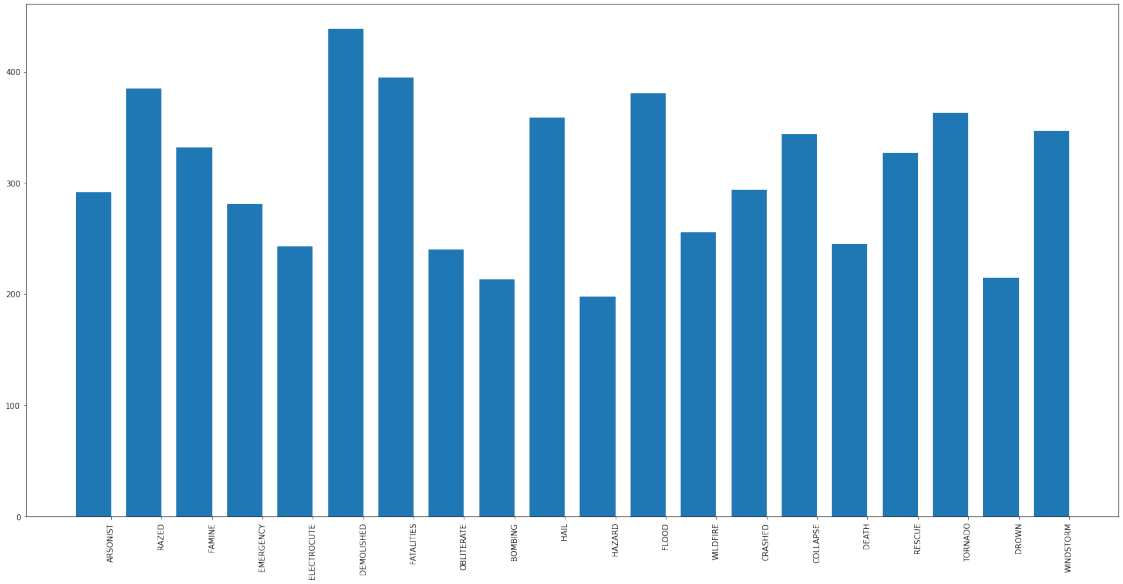
```
            'WINDSTORM': 327,
            'EVACUATE': 363,
            'WRECK': 215,
            'SURVIVE': 347})
```

[81]:
```
palabras = []
frecuencias = []
for i in kwFrequencies.keys():
    palabras.append(i)

for k in kwFrequencies.values():
    frecuencias.append(k)
```

[82]:
```
index1 = frecuencias.index(max(frecuencias))
pa1 = palabras[index1]
frecuencias.pop(index1)

index2 = frecuencias.index(max(frecuencias))
pa2 = palabras[index2]
frecuencias.pop(index2)

index3 = frecuencias.index(max(frecuencias))
pa3 = palabras[index3]
frecuencias.pop(index3)

print("La palabra más frecuente es: ", pa1)
print("La segunda palabra más frecuente es:", pa2)
print("La tercera palabra más frecuente es:", pa3)
```

```
La palabra más frecuente es:  BOMBING
La segunda palabra más frecuente es: FATALITIES
La tercera palabra más frecuente es: CRASHED
```

[87]:
```
n = 20
words = palabras[0:n]
freq = frecuencias[0:n]

# Plot histogram using matplotlib bar().
indexes = np.arange(len(words))
width = 0.7

fig = plt.figure(figsize = (25, 12 ))
plt.bar(indexes, freq, width)
plt.xticks(indexes + width * 0.5, words)
plt.xticks(rotation = 90)
plt.show()
```



Location

[73]:
```
lFrequencies = Counter(lText.split())
```

[74]:
```
lFrequencies
```

[74]:
```
Counter({'NO_DATA': 2548,
         'BIRMINGHAM': 23,
         'HERE': 32,
         'AMERICA': 20,
         'PHILADELPHIA_PA': 22,
         'LONDON': 115,
         'RHODE_ISLAND': 22,
         'WORLDWIDE': 89,
         'ALBERTA': 56,
         'LIVE_MS': 13,
         'MIDWEST': 42,
         'OREGON': 25,
         'ANGELES': 70,
         'GAINESVILLE_FL': 25,
         'INDIANA': 82,
         'CANADA': 70,
         'SOUTH_ASIA': 56,
         'COLUMBUS_OHIO': 28,
         'H': 20,
         'TRINIDAD_TOBAGO': 16,
         'CALGARY_CANADA': 98,
         'TWITTER': 6,
         'CAPE_TOWN': 25,
```

```
'SAN_FRANCISCO': 112,
'IRELAND': 41,
'NASHVILLE_TN': 20,
'UK': 42,
'WALKER_COUNTY_ALABAMA': 11,
'PORTLAND_ORE': 59,
'SAN_ANTONIO_TX': 21,
'BROOKLYN_NY': 23,
'LIVERPOOL': 30,
'LIVINGSTON_IL': 33,
'NEW_YORK': 170,
'LOUISIANA': 19,
'MANCHESTER': 32,
'WELLINGTON': 19,
'GLOBAL': 19,
'CHICAGO_IL': 77,
'BANGALORE_INDIA': 23,
'GARRETT': 12,
'LEICESTER': 23,
'NAIROBI_KENYA': 31,
'SINGAPORE': 29,
'KNOW': 10,
'MAD_HELL': 14,
'MORIOH_JAPAN': 21,
'EARTH': 31,
'UNITED_STATES': 66,
'MUMBAI': 58,
'RALEIGH_DURHAM_NC': 32,
'PENNSYLVANIA': 41,
'DETROIT': 22,
'PALO_ALTO_CALIFORNIA': 24,
'AIRES_ARGENTINA': 32,
'HEAD': 20,
'PERTH_WESTERN_AUSTRALIA': 19,
'FINANCIAL_NEWS_VIEWS': 13,
'SILICON_VALLEY': 11,
'POTTERS_HANDS': 15,
'BRISBANE': 30,
'T': 29,
'NYC': 41,
'JAPON': 25,
'WEST_COAST': 23,
'HAPPILY_MARRIED_KIDS': 21,
'RIO': 26,
'ONDO': 13,
'WILLIAMSBURG': 9,
'ATLANTA': 100,
```

```
'WINTER_PARK_COLORADO': 19,
'ONA_BLOCK_W_BOY': 4,
'CT': 10,
'LAGOS': 45,
'JOHANNESBURG': 20,
'PITTSBURGH': 13,
'ROAD_BILLIONAIRES_CLUB': 46,
'HOUSTON': 56,
'AMERICAN_WASTELAND_MV': 20,
'NIGERIA': 49,
'TORRANCE': 19,
'VIRGINIA_UNITED_STATES': 23,
'CALIFORNIA': 60,
'VICTORIA_BC': 19,
'SYDNEY_AUSTRALIA': 30,
'OK': 13,
'REPUBLIC_TEXAS': 17,
'PHOENIX_AZ': 27,
'RIGHT_YOU': 17,
'HUNTSVILLE': 11,
'ORLANDO': 33,
'TEXAS': 28,
'ILLINOIS': 22,
'ASHEVILLE_NC': 14,
'AUSTIN_TX': 22,
'BRUM': 10,
'OKLAHOMA_CITY': 29,
'VEGAS_NEVADA': 30,
'LONDON_ENGLAND': 38,
'MACCLESFIELD': 9,
'FLORIDA': 32,
'TAYLOR_SWIFT': 17,
'NEWCASTLE_OK': 37,
'MAGNOLIA': 22,
'KENYA': 36,
'PERTHSHIRE': 16,
'MEMPHIS_TN': 28,
'PARIS': 29,
'WASHINGTON_DC': 73,
'UK_GERMANY': 15,
'INTERNET': 30,
'NORTH': 32,
'SS': 13,
'BLEAK_HOUSE': 13,
'KAMA_FRANCE': 13,
'DALLAS_TX': 28,
'PORT_JERVIS_NY': 20,
```

```
'DUBLIN': 19,
'SEATTLE_WA': 62,
'DELI': 11,
'LYTHAM_ST_ANNES': 15,
'MADISON_WI': 28,
'GOLD_COAST_AUSTRALIA': 18,
'HTX': 8,
'CENTRAL_COAST_CALIFORNIA': 26,
'BUY_MONEY': 18,
'DENVER_COLORADO': 31,
'BRASIL': 22,
'PHILIPPINES': 15,
'METHVILLE': 17,
'PAIGNTON': 27,
'QUEENS': 16,
'UTAH': 15,
'MACON_GA': 23,
'MELBOURNE_AUSTRALIA': 30,
'MOON': 19,
'FRESNO': 13,
'DC': 7,
'ROCKY_MOUNTAINS': 16,
'GOTHAM_CITYUSA': 17,
'CONTAC_BFE': 8,
'SECRET': 16,
'CHINA': 23,
'BALTIMORE_MD': 17,
'TENNESSEE': 24,
'SOCHI_KDA_RU': 6,
'NORWAY': 9,
'VANCOUVER_BC': 25,
'PLANET_EARTH': 20,
'ONTARIO_CANADA': 17,
'MPP': 6,
'EWA_BEACH_HI': 15,
'IRAQAFGHANISTAN_RSA_BAGHDAD': 9,
'NEW_ENGLAND': 29,
'MINA_CITY': 18,
'THAILAND': 24,
'KISUMU': 4,
'VENTURA': 14,
'UNITED_KINGDOM': 26,
'PATERSON_NEW_JERSEY': 20,
'NOTTINGHAM_ENGLAND': 12,
'NJ': 11,
'ISTANBUL': 7,
'WACO_TEXAS': 15,
```

```
'ENGLAND_UNITED_KINGDOM': 13,
'TULSA_OK': 12,
'CHEVY_CHASE_MD': 11,
'LOWELL': 4,
'REDDING_CALIFORNIA': 15,
'PORTO': 11,
'UNTMDOUTDOORS_TORK': 7,
'KUALA_LUMPUR_MALAYSIA': 8,
'IM_SENT': 9,
'MICHIGAN': 15,
'US': 15,
'COVENTRY': 19,
'HONG_KONG': 10,
'SCOTTSDALE_AZ': 12})
```

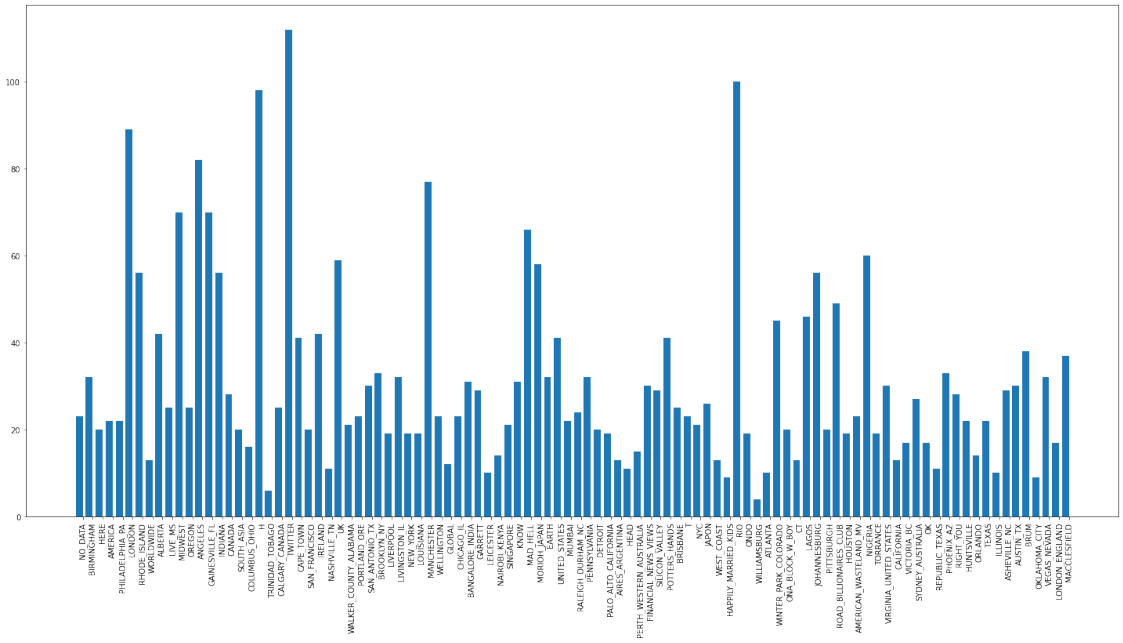
[76]:
```
palabras = []
frecuencias = []
for i in lFrequencies.keys():
    palabras.append(i)

for k in lFrequencies.values():
    frecuencias.append(k)
```

[77]:
```
index1 = frecuencias.index(max(frecuencias))
pa1 = palabras[index1]
frecuencias.pop(index1)

index2 = frecuencias.index(max(frecuencias))
pa2 = palabras[index2]
frecuencias.pop(index2)

index3 = frecuencias.index(max(frecuencias))
pa3 = palabras[index3]
frecuencias.pop(index3)

print("La palabra más frecuente es: ", pa1)
print("La segunda palabra más frecuente es:", pa2)
print("La tercera palabra más frecuente es:", pa3)
```

```
La palabra más frecuente es:  NO_DATA
La segunda palabra más frecuente es: LIVINGSTON_IL
La tercera palabra más frecuente es: PHILADELPHIA_PA
```

[78]:
```
n = 100
words = palabras[0:n]
freq = frecuencias[0:n]

# Plot histogram using matplotlib bar().
```

```
indexes = np.arange(len(words))
width = 0.7

fig = plt.figure(figsize = (25, 12 ))
plt.bar(indexes, freq, width)
plt.xticks(indexes + width * 0.5, words)
plt.xticks(rotation = 90)
plt.show()
```



Text

[88]:
```
# convertimos todo a un string
Text = " ".join(review for review in data.text.astype(str))
```

[91]:
```
Frequencies = Counter(Text.split())
```

[92]:
```
Frequencies
```

[92]:
```
Counter({'DEEDS': 2,
         'REASON': 20,
         'EARTHQUAKE': 50,
         'ALLAH': 9,
         'FORGIVE': 2,
         'FOREST': 65,
         'FIRE': 252,
         'NEAR': 54,
         'RONGE': 1,
```

```
'SASK': 1,
'CANADA': 11,
'RESIDENTS': 8,
'ASKED': 9,
'SHELTER': 6,
'PLACE': 26,
'NOTIFIED': 1,
'OFFICERS': 8,
'EVACUATION': 50,
'ORDERS': 11,
'EXPECTED': 15,
'PEOPLE': 196,
'RECEIVE': 2,
'WILDFIRES': 11,
'CALIFORNIA': 117,
'GOT': 112,
'SENT': 13,
'PHOTO': 41,
'RUBY': 1,
'ALASKA': 6,
'SMOKE': 48,
'POURS': 1,
'SCHOOL': 66,
'ROCKYFIRE': 4,
'UPDATE': 37,
'HWY': 10,
'CLOSED': 20,
'DIRECTIONS': 1,
'LAKE': 14,
'COUNTY': 38,
'CAFIRE': 2,
'FLOOD': 56,
'DISASTER': 153,
'HEAVY': 20,
'RAIN': 44,
'CAUSES': 13,
'FLASH': 21,
'FLOODING': 50,
'STREETS': 8,
'MANITOU': 1,
'COLORADO': 16,
'SPRINGS': 5,
'AREAS': 9,
'IM': 306,
'HILL': 7,
'WOODS': 2,
'THERES': 45,
```

```
'EMERGENCY': 157,
'HAPPENING': 12,
'BUILDING': 30,
'STREET': 24,
'AFRAID': 5,
'TORNADO': 31,
'COMING': 51,
'AREA': 41,
'DIED': 28,
'HEAT': 46,
'WAVE': 35,
'FAR': 28,
'HAHA': 20,
'SOUTH': 28,
'TAMPA': 4,
'GETTING': 56,
'FLOODED': 4,
'HAH': 3,
'WAIT': 26,
'SECOND': 28,
'LIVE': 60,
'GONNA': 43,
'FVCK': 1,
'RAINING': 4,
'FLORIDA': 8,
'TAMPABAY': 1,
'DAYS': 29,
'IVE': 47,
'LOST': 23,
'COUNT': 3,
'BAGO': 2,
'MYANMAR': 18,
'WE': 15,
'ARRIVED': 8,
'DAMAGE': 54,
'BUS': 37,
'MULTI': 2,
'CAR': 91,
'CRASH': 119,
'BREAKING': 41,
'WHATS': 29,
'MAN': 110,
'LOVE': 101,
'FRUITS': 2,
'SUMMER': 42,
'LOVELY': 8,
'FAST': 22,
```

```
'GOOOOOOOAAAAAAL': 1,
'RIDICULOUS': 4,
'LONDON': 16,
'COOL': 31,
'SKIING': 1,
'WONDERFUL': 5,
'DAY': 113,
'LOOOOOOL': 1,
'WAYI': 1,
'CANT': 108,
'EAT': 7,
'SHIT': 56,
'NYC': 12,
'WEEK': 37,
'GIRLFRIEND': 6,
'COOOOL': 1,
'LIKE': 345,
'PASTA': 2,
'END': 44,
'BBCMTD': 1,
'WHOLESALE': 4,
'MARKETS': 7,
'ABLAZE': 28,
'TRY': 19,
'BRING': 17,
'METAL': 13,
'RT': 107,
'AFRICANBAZE': 1,
'NEWSNIGERIA': 1,
'FLAG': 21,
'SET': 48,
'ABA': 14,
'CRYING': 9,
'MORE': 17,
'PLUS': 8,
'LOOK': 73,
'SKY': 15,
'NIGHT': 50,
'PHDSQUARES': 1,
'MUFC': 2,
'THEYVE': 5,
'BUILT': 6,
'HYPE': 3,
'NEW': 226,
'ACQUISITIONS': 2,
'DOUBT': 5,
'EPL': 1,
```

```
'SEASON': 14,
'INEC': 2,
'OFFICE': 12,
'ABIA': 2,
'BARBADOS': 1,
'BRIDGETOWN': 1,
'JAMAICA': 5,
'CARS': 21,
'SANTA': 6,
'CRUZ': 10,
'HEAD': 42,
'ST': 46,
'ELIZABETH': 2,
'POLICE': 141,
'SUPERINTENDE': 1,
'LORD': 20,
'D': 23,
'CHECK': 46,
'OUT': 31,
'NSFW': 3,
'OUTSIDE': 25,
'ALIVE': 11,
'DEAD': 96,
'INSIDE': 28,
'AWESOME': 16,
'TIME': 127,
'VISITING': 1,
'CFC': 2,
'ANCOP': 1,
'SITE': 27,
'THANKS': 30,
'TITA': 1,
'VIDA': 2,
'TAKING': 17,
'CARE': 31,
'SOOOO': 2,
'PUMPED': 1,
'SOUTHRIDGELIFE': 1,
'WANTED': 18,
'CHICAGO': 12,
'PREACHING': 1,
'HOTEL': 6,
'GAINED': 6,
'FOLLOWERS': 8,
'YOU': 51,
'KNOW': 112,
'STATS': 3,
```

```
'GROW': 8,
'WEST': 24,
'BURNED': 39,
'THOUSANDS': 19,
'PERFECT': 10,
'TRACKLIST': 1,
'LIFE': 87,
'LEAVE': 28,
'RETAINERS': 1,
'IN': 30,
'WEIRD': 9,
'BETTER': 36,
'IT': 65,
'WEAR': 3,
'SINGLE': 11,
'YEAR': 70,
'LEAST': 2,
'DEPUTIES': 4,
'SHOT': 27,
'BRIGHTON': 2,
'HOME': 77,
'WIFE': 12,
'YEARS': 80,
'JAIL': 3,
'SETTING': 11,
'NIECE': 3,
'SUPERINTENDENT': 1,
'LANFORD': 1,
'SALMON': 3,
'R': 19,
'ARSONIST': 18,
'DELIBERATELY': 1,
'BLACK': 66,
'CHURCH': 7,
'NORTH': 27,
'CAROLINAABLAZE': 1,
'NOCHES': 1,
'ELBESTIA': 1,
'ALEXISSANCHEZ': 1,
'HAPPY': 23,
'TEAMMATES': 1,
'TRAINING': 12,
'HARD': 20,
'GOODNIGHT': 1,
'GUNNERS': 1,
'KURDS': 1,
'TRAMPLING': 1,
```

```
'TURKMEN': 2,
'LATER': 11,
'VANDALIZED': 1,
'OFFICES': 3,
'DIYALA': 1,
'TRUCK': 50,
'VOORTREKKER': 1,
'AVE': 15,
'TAMBO': 1,
'INTL': 1,
'CARGO': 3,
'SECTION': 4,
'HEARTS': 5,
'CITY': 60,
'GIFT': 4,
'SKYLINE': 2,
'KISS': 2,
'LIPS': 1,
'TONIGHT': 38,
'ANGELES': 4,
'EXPECTING': 4,
'IG': 4,
'FB': 4,
'FILLED': 3,
'SUNSET': 6,
'SHOTS': 15,
'PEEPS': 3,
'CLIMATE': 19,
'ENERGY': 7,
'REVEL': 1,
'WMV': 2,
'VIDEOS': 9,
'MEANS': 14,
'MAC': 6,
'FAREWELL': 1,
'ROUTE': 10,
'DVD': 5,
'GTXRWM': 1,
'PROGRESSIVE': 1,
'GREETINGS': 1,
'MONTH': 12,
'STUDENTS': 13,
'PENS': 1,
'TORCH': 4,
'PUBLICATIONS': 1,
'RENE': 2,
'AMP': 300,
```

```
'JACINTA': 1,
'SECRET': 15,
'K': 15,
'FALLEN': 4,
'SKIES': 3,
'EDIT': 2,
'MAR': 1,
'NAVISTA': 1,
'STEVE': 8,
'FIRES': 100,
'ELSE': 3,
'TINDERBOX': 1,
'CLOWN': 1,
'HOOD': 2,
'NEWS': 198,
'NOWPLAYING': 26,
'IAN': 4,
'BUFF': 1,
'MAGNITUDE': 1,
'EDM': 11,
'NXWESTMIDLANDS': 1,
'HUGE': 21,
'TALK': 19,
'UNTIL': 1,
'WORK': 74,
'I': 68,
'KIDS': 30,
'CUZ': 6,
'BICYCLE': 2,
'ACCIDENT': 87,
'SPLIT': 3,
'TESTICLES': 1,
'IMPOSSIBLE': 4,
'MICHAEL': 12,
'FATHER': 5,
'W': 63,
'NASHVILLETRAFFIC': 1,
'TRAFFIC': 31,
'MOVING': 11,
'M': 31,
'SLOWER': 3,
'USUAL': 5,
'CENTER': 23,
'LANE': 8,
'BLOCKED': 11,
'SANTACLARA': 1,
'US': 71,
```

```
'NB': 5,
'GREAT': 62,
'AMERICA': 19,
'PKWY': 2,
'BAYAREA': 1,
'PERSONALINJURY': 1,
'READ': 64,
'ADVICE': 2,
'SOLICITOR': 2,
'HELP': 72,
'OTLEYHOUR': 1,
'STLOUIS': 1,
'CARACCIDENTLAWYER': 1,
'SPEEDING': 1,
'TEEN': 13,
'ACCIDENTS': 9,
'TEE': 2,
'REPORTED': 16,
'MOTOR': 5,
'VEHICLE': 19,
'CURRY': 1,
'HERMAN': 1,
'RD': 30,
'STEPHENSON': 1,
'INVOLVING': 13,
'OVERTURNED': 1,
'USE': 31,
'BIGRIGRADIO': 1,
'AWARENESS': 2,
'MILE': 8,
'MARKER': 3,
'MOORESVILLE': 2,
'IREDELL': 2,
'RAMP': 1,
'PM': 105,
'SLEEPJUNKIES': 1,
'SLEEPING': 10,
'PILLS': 1,
'DOUBLE': 13,
'RISK': 16,
'BY': 11,
'KNEW': 8,
'GON': 4,
'HAPPEN': 13,
'N': 29,
'CABRILLO': 1,
'HWYMAGELLAN': 1,
```

```
'AV': 5,
'MIR': 1,
'CONGESTION': 1,
'PASTOR': 1,
'SCENE': 15,
'ACCIDENTWHO': 1,
'OWNER': 11,
'RANGE': 4,
'ROVER': 2,
'MOM': 21,
'WISHED': 1,
'WHY': 4,
'THAT': 24,
'THERE': 15,
'SPILT': 1,
'MAYONNAISE': 1,
'HORRIBLE': 35,
'PAST': 35,
'SUNDAY': 10,
'FINALLY': 23,
'ABLE': 10,
'AROUND': 3,
'THANK': 32,
'GOD': 57,
'PISSED': 2,
'DONNIE': 1,
'TELL': 25,
'TRUCKCRASH': 1,
'OVERTURNS': 1,
'FORTWORTH': 1,
'INTERSTATE': 2,
'CLICK': 6,
'CRASHGT': 1,
'ASHVILLE': 1,
'SB': 4,
'SR': 5,
'CAROLINA': 1,
'MOTORCYCLIST': 9,
'DIES': 14,
'CROSSED': 6,
'MEDIAN': 1,
'MOTORCYCLE': 5,
'RIDER': 5,
'TRAVELING': 2,
'FYI': 3,
'CADFYI': 2,
'PROPERTY': 18,
```

```
'DAMAGENHS': 1,
'PINER': 2,
'RDHORNDALE': 2,
'DR': 13,
'NAAYF': 1,
'TURNING': 3,
'CHANDANEE': 1,
'MAGU': 1,
'MMA': 6,
'TAXI': 1,
'RAMMED': 2,
'HALFWAY': 1,
'TURNED': 16,
'CONF': 1,
'LEFT': 32,
'MANCHESTER': 9,
'EDDY': 1,
'STOP': 50,
'NHA': 1,
'DELAY': 3,
'MINS': 4,
'DAMAGEWPD': 1,
'TH': 51,
'INJURY': 37,
'WILLIS': 1,
'FOREMAN': 1,
'AASHIQUI': 1,
'ACTRESS': 2,
'ANU': 1,
'AGGARWAL': 1,
'NEARFATAL': 1,
'SUFFIELD': 1,
'ALBERTA': 4,
'BACKUP': 2,
'SOUTHACCIDENT': 1,
'BLOCKING': 5,
'RIGHT': 68,
'LANES': 5,
'EXIT': 8,
'LANGTREE': 1,
'RDCONSIDER': 1,
'NC': 10,
'ALTERNATE': 1,
'CHANGED': 4,
'DETERMINE': 1,
'OPTIONS': 4,
'FINANCIALLY': 1,
```

```
'SUPPORT': 24,
'PLANS': 27,
'ONGOING': 1,
'TREATMENT': 5,
'DEADLY': 9,
'HAPPENED': 19,
'HAGERSTOWN': 1,
'TODAY': 87,
'ILL': 39,
'DETAILS': 6,
'YOURSTATE': 1,
'WHAG': 1,
'FLOWRI': 1,
'MARINADING': 1,
'FUCKING': 46,
'MFS': 3,
'DRIVE': 18,
'BAHRAIN': 1,
'PREVIOUSLY': 4,
'ROAD': 38,
'KILLED': 96,
'EXPLOSION': 41,
'HEARD': 35,
'LEADERS': 4,
'KENYA': 4,
'FORWARD': 4,
'COMMENT': 13,
'ISSUE': 10,
'DISCIPLINARY': 1,
'MEASURESARRESTPASTORNGANGA': 1,
'AFTERSHOCKDELO': 2,
'SCUF': 2,
'PS': 10,
'GAME': 37,
'CYA': 1,
'THE': 65,
'EFFORT': 6,
'GETS': 26,
'PAINFUL': 1,
'WIN': 21,
'ROGER': 2,
'BANNISTER': 1,
'ICEMOON': 7,
'AFTERSHOCK': 19,
'DJICEMOON': 7,
'DUBSTEP': 8,
'TRAPMUSIC': 7,
```

```
'DNB': 8,
'DANCE': 14,
'ICES': 7,
'VICTORY': 5,
'BARGAIN': 5,
'BASEMENT': 3,
'PRICES': 5,
'DWIGHT': 1,
'DAVID': 9,
'EISENHOWER': 1,
'NOBODY': 3,
'REMEMBERS': 1,
'CAME': 36,
'CHARLES': 4,
'SCHULZ': 1,
'SPEAKING': 4,
'XB': 2,
'ALSO': 3,
'HARDER': 1,
'CONFLICT': 5,
'GLORIOUS': 3,
'TRIUMPH': 1,
'THOMAS': 7,
'PAINE': 1,
'GROWINGUPSPOILED': 1,
'GOING': 103,
'CLAY': 1,
'PIGEON': 1,
'SHOOTING': 29,
'GUESS': 15,
'ACTUALLY': 28,
'WANTS': 15,
'FREE': 42,
'TC': 2,
'TERRIFYING': 3,
'BEST': 73,
'ROLLER': 3,
'COASTER': 2,
'ON': 21,
'DISCLAIMER': 1,
'FEW': 2,
'KJFORDAYS': 1,
'SEEING': 16,
'ISSUES': 30,
'WISDOMWED': 1,
'BONUS': 1,
'MINUTE': 31,
```

```
'DAILY': 21,
'HABITS': 2,
'IMPROVE': 2,
'DO': 7,
'LIFEHACKS': 1,
'PROTECT': 7,
'PROFIT': 4,
'GLOBAL': 18,
'FINANCIAL': 13,
'MELTDOWN': 32,
'WIEDEMER': 1,
'HTTP': 1,
'MOMENT': 17,
'SCARY': 6,
'GUY': 19,
'SCREAMING': 43,
'BLOODY': 44,
'MURDER': 42,
'SILVERWOOD': 1,
'FULL': 2,
'STREAMING': 2,
'YOUTUBE': 98,
'GTGT': 13,
'BOOK': 23,
'ESQUIREATTIRE': 1,
'FACE': 46,
'DIFFICULTIES': 1,
'WRONG': 21,
'JOEL': 3,
'OSTEEN': 1,
'THING': 36,
'STANDS': 4,
'DREAM': 9,
'BELIEF': 2,
'POSSIBLE': 32,
'BROWN': 26,
'PRAISE': 1,
'MINISTRY': 1,
'TELLS': 5,
'IS': 14,
'NOW': 45,
'WDYOUTH': 1,
'BIBLESTUDY': 1,
'REMEMBERING': 6,
'DIE': 24,
'WAY': 77,
'AVOID': 10,
```

```
'TRAP': 5,
'THINKING': 16,
'LOSE': 10,
'JOBS': 21,
'TRIED': 15,
'ORANGE': 4,
'ONFIREANDERS': 1,
'BB': 7,
'KICK': 9,
'WANT': 80,
'MAKING': 29,
'INTERRUPT': 1,
'GEORGE': 6,
'BERNARD': 1,
'SHAW': 2,
'OYSTER': 1,
'SHELL': 4,
'ANDREW': 2,
'CARNEGIE': 1,
'NEED': 72,
'PU': 1,
'PLAY': 26,
'HYBRID': 3,
'SLAYER': 3,
'EU': 3,
'HMU': 2,
'CODSANDSCRIMS': 1,
'EMPIRIKGAMING': 1,
'CODAWSCRIMS': 1,
'TPKOTC': 1,
'TPFA': 1,
'AFTERSHOCKORG': 1,
'EXPERTS': 18,
'FRANCE': 16,
'BEGIN': 12,
'EXAMINING': 10,
'AIRPLANE': 37,
'DEBRIS': 50,
'FOUND': 52,
'REUNION': 33,
'ISLAND': 38,
'FRENCH': 10,
'AIR': 41,
'O': 14,
'STRICT': 2,
'LIABILITY': 2,
'CONTEXT': 3,
```

```
'PILOT': 8,
'ERROR': 5,
'COMMON': 10,
'COMPONENT': 1,
'AVIATION': 2,
'CR': 4,
'CROBSCARLA': 1,
'LIFETIME': 3,
'ODDS': 3,
'DYING': 9,
'WEDN': 2,
'ALEXALLTIMELOW': 1,
'AWWWW': 2,
'THEYRE': 23,
'CUTIES': 1,
'GOOD': 89,
'JOB': 25,
'FAMILY': 44,
'MEMBERS': 15,
'OSAMA': 1,
'BIN': 10,
'LADEN': 9,
'IRONIC': 2,
'MHMMM': 1,
'GOV': 9,
'SUSPECT': 36,
'GOES': 31,
'ENGINE': 4,
'WINGS': 6,
'CESSNA': 1,
'OCAMPO': 1,
'COAHUILA': 2,
'MEXICO': 7,
'JULY': 7,
'MEN': 27,
'INCLUDING': 8,
'STATE': 51,
'GOVERNMENT': 31,
'OFFICIAL': 20,
'WATCHTHEVIDEO': 1,
'WEDNESDAY': 15,
'BEGAN': 9,
'T': 18,
'KCA': 4,
'VOTEJKTID': 4,
'MBATAWEEL': 1,
'RIP': 9,
```

```
'BINLADEN': 1,
'AIRPLANES': 1,
'COWORKER': 2,
'NUDES': 1,
'MODE': 16,
'MICKINYMAN': 1,
'THEATLANTIC': 1,
'WRECK': 61,
'POLITICS': 10,
'MLB': 5,
'UNBELIEVABLY': 1,
'INSANE': 6,
'AIRPORT': 30,
'AIRCRAFT': 25,
'AEROPLANE': 1,
'RUNWAY': 7,
'FREAKY': 1,
'USAMA': 1,
'LADINS': 1,
'NATURALLY': 1,
'PLANE': 33,
'FESTIVAL': 7,
'DEATH': 71,
'CARFEST': 1,
'DTN': 4,
'BRAZIL': 7,
'EXP': 6,
'WTF': 12,
'BELIEVE': 28,
'EYES': 23,
'NICOLE': 1,
'FLETCHER': 1,
'VICTIM': 12,
'CRASHED': 36,
'TIMES': 48,
'AGO': 26,
'LITTLE': 49,
'BIT': 13,
'TRAUMA': 39,
'HER': 10,
'OMG': 21,
'THIS': 15,
'BRO': 8,
'JETENGINE': 1,
'TURBOJET': 1,
'BOING': 1,
'G': 17,
```

```
'PHONE': 37,
'LOOKS': 46,
'SHIP': 21,
'TERRIBLE': 7,
'STATISTICALLY': 1,
'COP': 8,
'CRASHES': 5,
'HOUSE': 43,
'COLOMBIA': 1,
'DRONE': 6,
'CAUSE': 48,
'PILOTS': 6,
'WORRIED': 5,
'DRONES': 5,
'ESP': 3,
'CLOSE': 11,
'VICINITY': 2,
'AIRPORTS': 1,
'EARLY': 16,
'WAKE': 28,
'SISTER': 8,
'BEGGING': 2,
'COME': 56,
'RIDE': 4,
'WHER': 1,
'AMBULANCE': 39,
'HOSPITAL': 10,
'RODKIAI': 1,
'FEARED': 22,
'PAKISTANI': 16,
'HELICOPTER': 25,
'AMBULANCES': 2,
'LORRY': 1,
'EMSNE': 1,
'REUTERS': 24,
'YUGVANI': 3,
'LEADING': 9,
'SERVICES': 44,
'BOSS': 5,
'WELCOMES': 1,
'CHARITY': 8,
'TRAVELLING': 2,
'ABERYSTWYTHSHREWSBURY': 1,
'INCIDENT': 10,
'HALT': 2,
'SHREWS': 1,
'SPRINTER': 4,
```

```
'AUTOMATIC': 7,
'FRONTLINE': 4,
'CHOICE': 7,
'LEZ': 4,
'COMPLIANT': 4,
'EBAY': 33,
'NANOTECH': 1,
'DEVICE': 4,
'TARGET': 9,
'DESTROY': 41,
'BLOOD': 44,
'CLOTS': 1,
'SKYHAWKMM': 1,
'TRAPLORD': 1,
'FREDOSANTANA': 1,
'LILREESE': 1,
'HELLA': 2,
'CRAZY': 21,
'FIGHTS': 5,
'COUPLE': 19,
'MOSH': 1,
'PITS': 2,
'RUN': 42,
'LUCKY': 11,
'JUSTSAYING': 1,
'RANDOMTHOUGHT': 1,
'TILNOW': 3,
'DNA': 1,
'TANSLASH': 1,
'WAITING': 10,
'FOUSEYTUBE': 2,
'OK': 36,
'HAHAHAH': 2,
'PAKISTAN': 21,
'KILLS': 35,
'THENISSONIAN': 1,
'REJECTDCARTOONS': 1,
'NISSAN': 1,
'MEDICAL': 6,
'ASSISTANCE': 4,
'EMS': 7,
'NY': 10,
'EMTS': 1,
'PETITION': 12,
'HOUR': 13,
'MINIMUM': 1,
'WAGE': 1,
```

```
'PARAMEDICS': 1,
'KIWIKARYN': 1,
'PARKING': 4,
'LOT': 33,
'SAID': 57,
'JOHNS': 3,
'LTLT': 2,
'DOG': 17,
'THINKS': 3,
'HES': 45,
'HATZOLAH': 1,
'RESPONDING': 2,
'DUAL': 3,
'SIRENS': 30,
'AND': 15,
'WORLDNEWS': 9,
'NUMBER': 14,
'LESOTHO': 1,
'BODY': 125,
'MEDIC': 2,
'AACEORG': 1,
'SURPRISED': 5,
'STANDARDISED': 1,
'CLINICAL': 1,
'PRACTICE': 8,
'NHS': 2,
'TRUST': 18,
'JWALK': 1,
'PASSING': 7,
'HATE': 21,
'EPISODE': 12,
'TRUNKS': 1,
'ANNIHILATED': 31,
'FREIZA': 1,
'CLEANEST': 1,
'EVER': 5,
'SHOWED': 3,
'NIGGA': 13,
'MERCY': 4,
'SHALL': 10,
'PETEBESTS': 1,
'DESSICATED': 1,
'LAID': 3,
'BARE': 4,
'KNEEL': 1,
'URIBE': 4,
'BASEBALL': 7,
```

```
'METS': 8,
'MARKSMAPONYANE': 1,
'HEYSUNDOWNS': 1,
'PREVIOUS': 4,
'MEETING': 9,
'CELTICINDEED': 1,
'IMPROVEMENT': 1,
'VOLFAN': 1,
'TNEAZZY': 1,
'MIZZOU': 1,
'SEASONS': 3,
'ENDED': 7,
'MUSCHAMPS': 1,
'CAREER': 13,
'COMPETE': 2,
'BAMA': 1,
'ABS': 1,
'STATUS': 5,
'EDUCATION': 5,
'MBA': 1,
'BEHALF': 3,
'EASY': 8,
'CAREEN': 1,
'EOVM': 1,
'TO': 12,
'LUKA': 2,
'THEM': 28,
'ALOIS': 2,
'TRANCY': 2,
'ACAREWORNHEART': 1,
'FELLA': 2,
'SORRY': 21,
'PULLS': 5,
'DRUNK': 11,
'DRIVER': 17,
'SAFETY': 15,
'SECONDS': 8,
'HIT': 52,
'TRAIN': 93,
'VIRALSPELL': 5,
'BOOM': 3,
'COUNTRY': 22,
'ENTIRELY': 1,
'H': 9,
'BRITAIN': 2,
'AMIRKINGKHAN': 1,
'FLOYDMAYWEATHER': 1,
```

```
'SUREGOD': 1,
'PROMISED': 1,
'ISRAEL': 17,
'BUTTHE': 1,
'HORROR': 26,
'IRAN': 35,
'WNUKES': 1,
'VIOLENTFEMINAZI': 1,
'THATS': 61,
'ARMENIANS': 1,
'WEVE': 8,
'SPENT': 8,
'HISTORY': 25,
'INSTANTLY': 2,
'AWARE': 5,
'ABILITY': 4,
'ANNIHILATE': 1,
'HUMANITY': 9,
'TRYOUTS': 2,
'WENT': 33,
'MINUS': 1,
'FACT': 12,
'STOPPED': 9,
'QUICKLY': 9,
'SHORT': 12,
'BALL': 21,
'TOENAIL': 1,
'S': 43,
'ORYX': 1,
'SYMBOL': 1,
'ARABIAN': 4,
'PENINSULA': 1,
'HUNTERS': 7,
'THEY': 1,
'READY': 18,
'BUCS': 1,
'PHILIPDUNCAN': 1,
'BREAKFASTONE': 1,
'NIGHTS': 2,
'WEATHER': 51,
'PHILIP': 1,
'THOUGHT': 31,
'FORECAST': 9,
'DOMAIN': 1,
'SOPHISTICATION': 1,
'CLOSELY': 1,
'UPTOTHEMINUTE': 1,
```

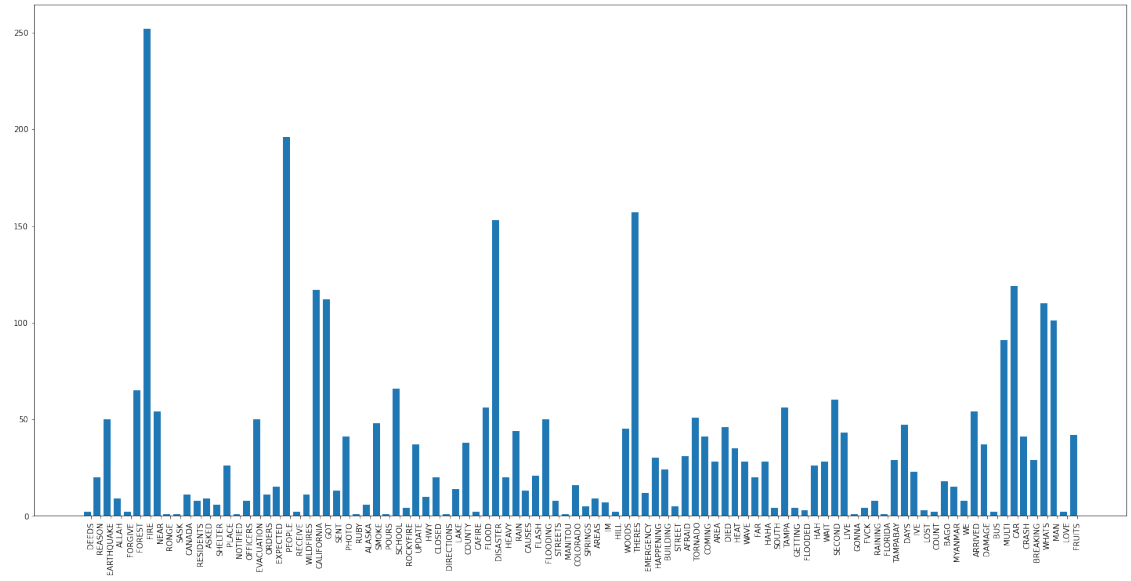
```
 'FEAT': 12,
 'ZRNF': 1,
 'STORMBEARD': 1,
 'STEELLORD': 1,
 …})
```

[93]:
```
palabras = []
frecuencias = []
for i in Frequencies.keys():
    palabras.append(i)

for k in Frequencies.values():
    frecuencias.append(k)
```

[94]:
```
index1 = frecuencias.index(max(frecuencias))
pa1 = palabras[index1]
frecuencias.pop(index1)

index2 = frecuencias.index(max(frecuencias))
pa2 = palabras[index2]
frecuencias.pop(index2)

index3 = frecuencias.index(max(frecuencias))
pa3 = palabras[index3]
frecuencias.pop(index3)

print("La palabra más frecuente es: ", pa1)
print("La segunda palabra más frecuente es:", pa2)
print("La tercera palabra más frecuente es:", pa3)
```

```
La palabra más frecuente es:  LIKE
La segunda palabra más frecuente es: IM
La tercera palabra más frecuente es: PUBLICATIONS
```

[95]:
```
n = 100
words = palabras[0:n]
freq = frecuencias[0:n]

# Plot histogram using matplotlib bar().
indexes = np.arange(len(words))
width = 0.7

fig = plt.figure(figsize = (25, 12 ))
plt.bar(indexes, freq, width)
plt.xticks(indexes + width * 0.5, words)
plt.xticks(rotation = 90)
plt.show()
```


250
200
150
100
50
0
DEEDS
REASON
EARTHQUAKE
ALLAH
FORGIVE
FOREST
FIRE
NEAR
RONGE
SASK
CANADA
RESIDENTS
ASKED
SHELTER
PLACE
NOTIFIED
OFFICERS
EVACUATION
SHELTERS
EXPECTED
PEOPLE
RESIDENT
WILDFIRES
CALIFORNIA
EVERY
SENT
PHOTO
RUBY
ALASKA
SMOKE
POURS
SCHOOL
ROCKYFIRE
UPDATE
HWY
CLOSED
DIRECTIONS
LAKE
COUNTY
CAFIRE
FLOOD
DISASTER
HEAVY
RAIN
CAUSES
FLASH
FLOODING
STREETS
MANITOU
COLORADO
SPRINGS
AREAS
M
HILL
WOODS
TWEETS
EMERGENCY
HAPPENING
BUILDING
STREET
AFRAID
TORNADO
COMING
AREA
DIED
HEAT
WAVE
FAR
HAHA
SOUTH
TAMPA
GETTING
FLOODED
HAHA
WAIT
SECOND
LIVE
GONNA
FUCK
RAINING
FLORIDA
TAMPABAY
DAYS
I'VE
LOST
COUNT
DAGO
MYANMAR
WE
ARRIVED
BANGKOK
BUS
MULI
CAN
CRASH
BREAKING
WHATS
WAR
LOVE
FRUITS