

Universidad del Valle de Guatemala
Data Science - CC3066 - Sección 20
Catedrático: LUIS ROBERTO FURLAN COLLVER

Integrantes:

JULIO ROBERTO HERRERA SABAN

OSCAR RENE SARAVIA DONIS

DIEGO DE JESUS ARREDONDO TURCIOS

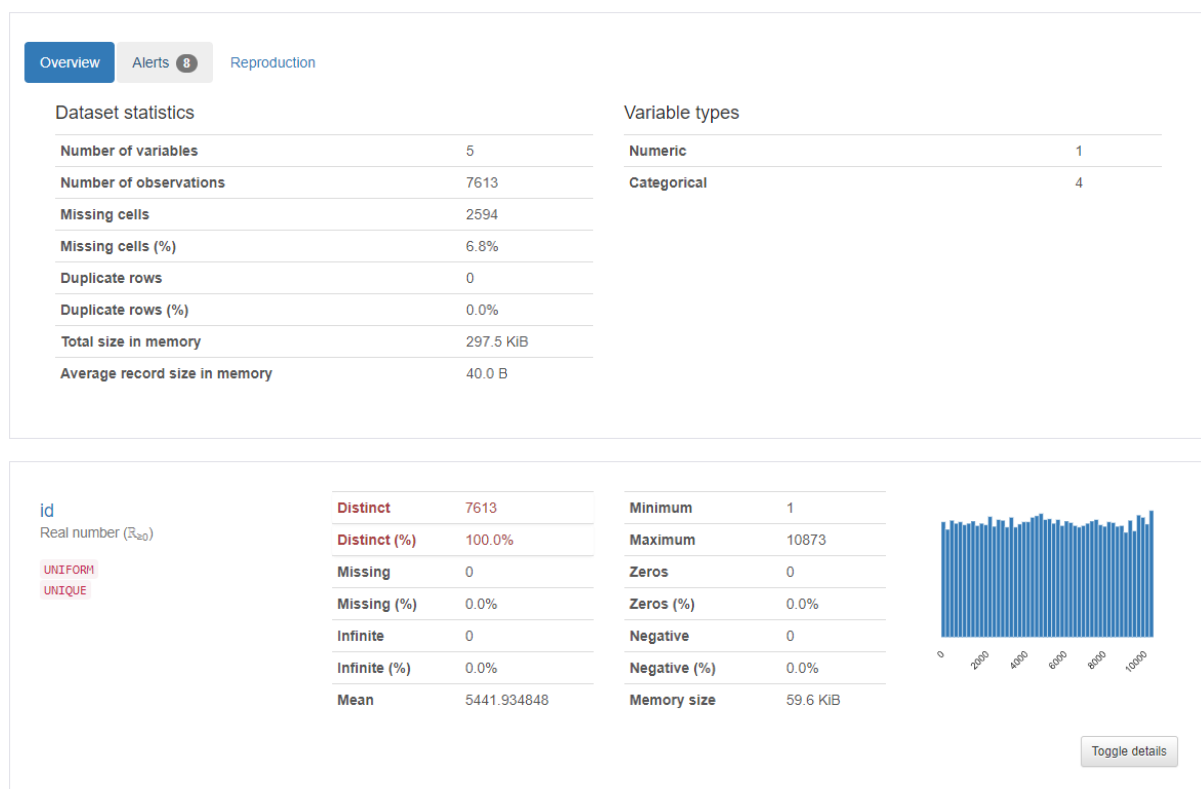
Laboratorio 5 Análisis de Sentimientos

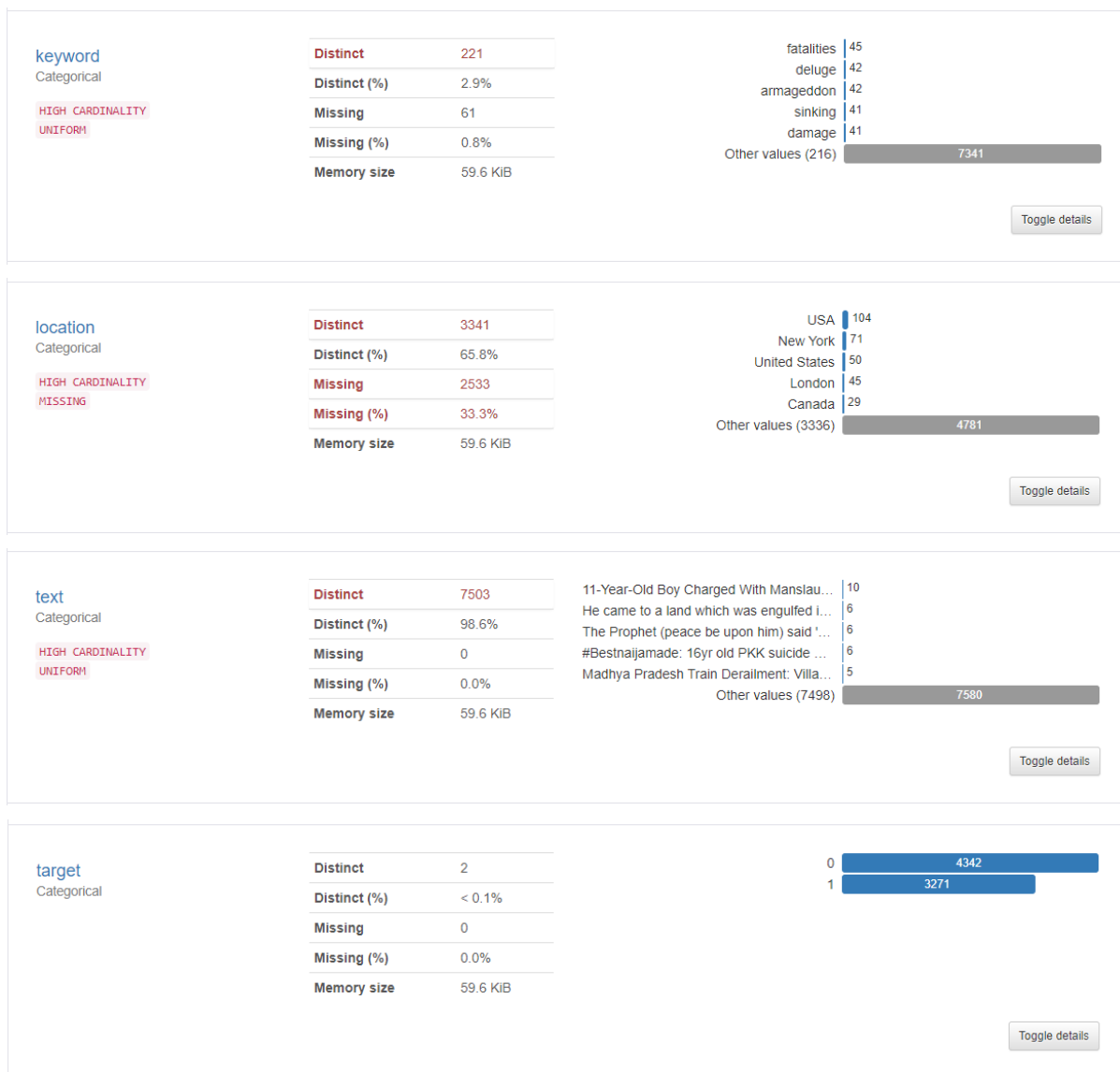
Link del Repositorio: <https://github.com/arr19422/Lab5-DS>

Análisis exploratorio

El *dataset* utilizado fue adquirido de Kaggle: [Natural Language Processing with Disaster Tweets](#) y se generó un reporte al *dataset* sin procesamiento.

Reporte:





Las variables del dataset representan lo siguiente:

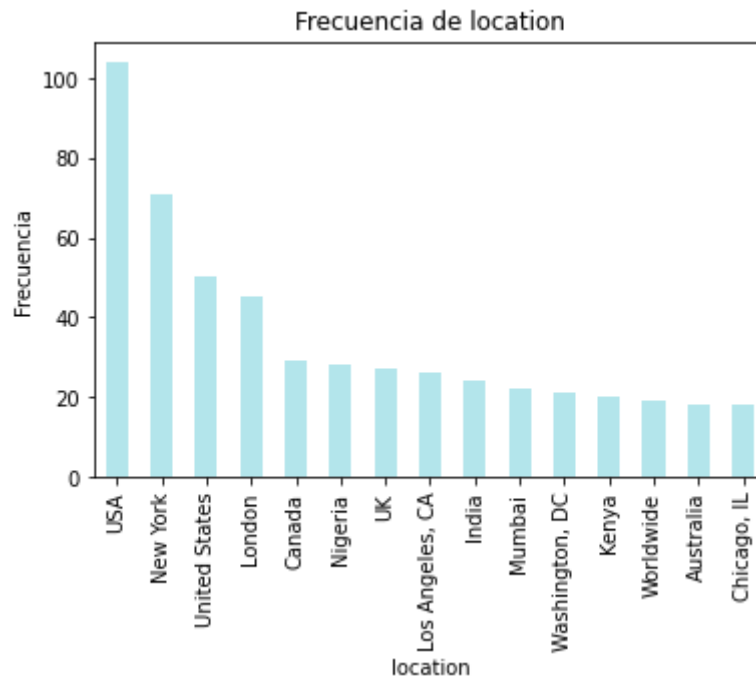
- id: Identificador del tweet.
- keyword: Palabra clave del tweet.
- location: Ubicación desde donde se envió el tweet.
- text: Cuerpo del tweet.
- target: Variable boolean que es verdadero si el tweet habla de un desastre real y falso si no.

Como podemos ver, las variables *keyword* y *location* tienen 0.8% y 33.3% de valores vacíos sin embargo, no se removerán estas observaciones ya que si bien podemos obtener conclusiones de ellas, es la variable *text* la que nos importa para realizar la clasificación.

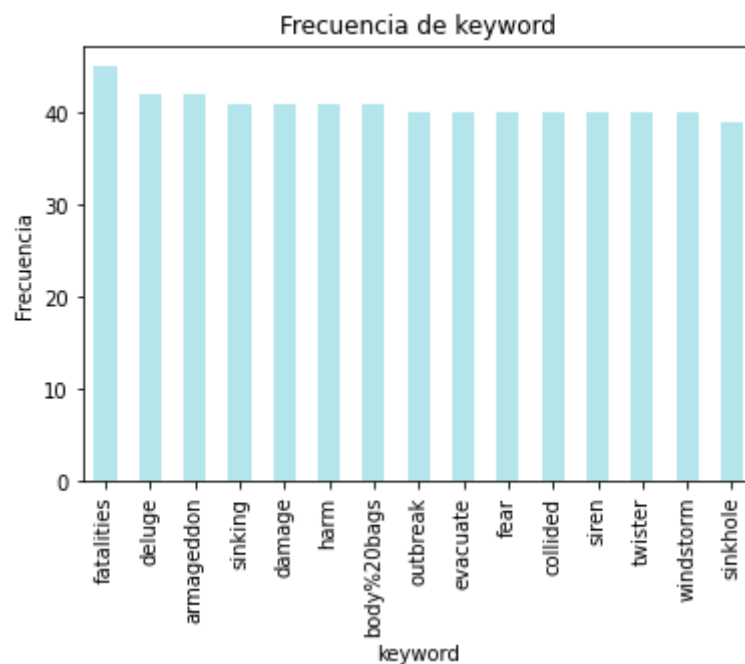
Hallazgos:

Algunas hallazgos encontrados por medio de gráficas a las variables son:

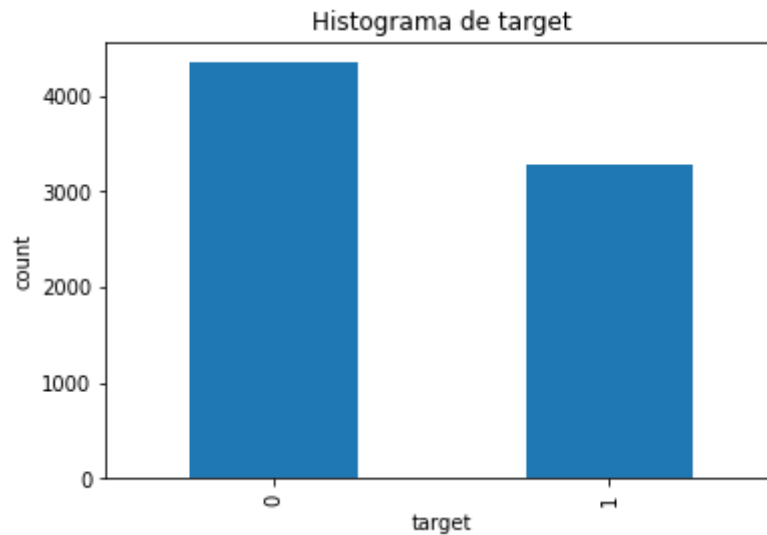
La ubicación desde donde más se tuiteó fue USA, sin embargo hay distintos valores que representan a este mismo país ya que también se encuentra como United States o como todos los estados específicos del país como New York, Los Angeles, CA y Washington, DC que se encuentran también entre el top 11 de ubicaciones más repetidas.



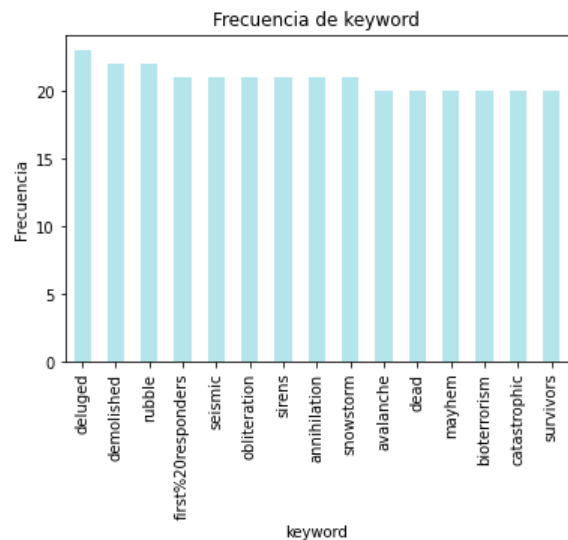
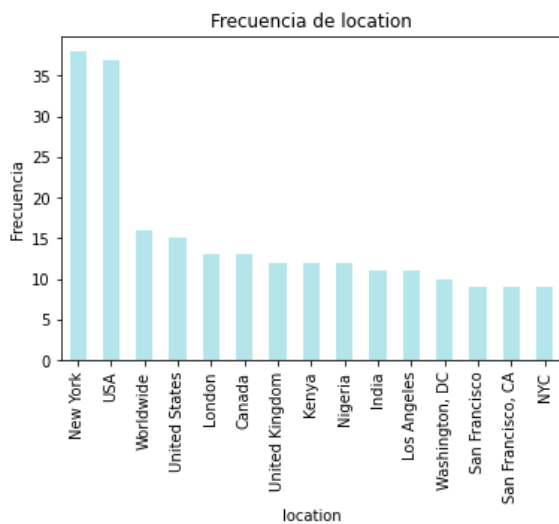
Las palabras clave (*keywords*) tienen una distribución más uniforme, repitiéndose más de la mitad de las observaciones entre 30 y 45 veces. Se puede observar que entre el top 15 de palabras más frecuentes como *keywords* están relacionadas con algún hecho trágico. También se observa que algunos valores pueden estar conformados realmente por más de una palabra pero siendo el espacio representado por "%20" normalmente utilizado para hacer *encoding* de URLs.



La variable *target* es la que nos ayudará a entrenar el modelo y clasificar si se habla de un desastre real o no. Esta es la variable que no se encuentra en el *dataset* de *testeo* y en este de entrenamiento muestra que existe mayoría de tweets que no hablan de un desastre real, sin embargo es una distribución bastante buena.



En cuanto al *dataset* de *testeo* se observó un comportamiento parecido en las variables *location* y *keywords* con algunas diferencias en las posiciones de los valores con más frecuencia.



Limpieza y procesamiento de los datos

Primero se convirtieron todos los datos a minúsculas:

Convirtiendo a minúsculas

```
for i in range(len(train)):
    train.loc[i, 'text'] = train.loc[i, 'text'].lower()

for i in range(len(test)):
    test.loc[i, 'text'] = test.loc[i, 'text'].lower()
```

✓ 2.1s

```
train.head()
```

✓ 0.6s

	id	keyword	location	text	target
0	1	NaN	NaN	our deeds are the reason of this #earthquake m...	1
1	4	NaN	NaN	forest fire near la longe sask. canada	1
2	5	NaN	NaN	all residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	just got sent this photo from ruby #alaska as ...	1

```
test.head()
```

✓ 0.4s

	id	keyword	location	text
0	0	NaN	NaN	just happened a terrible car crash
1	2	NaN	NaN	heard about #earthquake is different cities, s...
2	3	NaN	NaN	there is a forest fire at spot pond, geese are...
3	9	NaN	NaN	apocalypse lighting. #spokane #wildfires
4	11	NaN	NaN	typhoon soudelor kills 28 in china and taiwan

Luego se quitaron los URLs:

Quitando urls

```
#Quitar las urls
for i in range(len(train)):
    train.loc[i, 'text'] = re.sub(r'((http|https)\:\/\/)?[a-zA-Z0-9\.\/\?\:@\-\_=#]+\.[a-zA-Z]{2,6}([a-zA-Z0-9\.\&\/\?\:@\-\_=#])*', '', train.loc[i, 'text'])

for i in range(len(test)):
    test.loc[i, 'text'] = re.sub(r'((http|https)\:\/\/)?[a-zA-Z0-9\.\/\?\:@\-\_=#]+\.[a-zA-Z]{2,6}([a-zA-Z0-9\.\&\/\?\:@\-\_=#])*', '', test.loc[i, 'text'])
```

✓ 2.2s

```
train.head()
```

✓ 0.5s

	id	keyword	location	text	target
0	1	NaN	NaN	our deeds are the reason of this #earthquake m...	1
1	4	NaN	NaN	forest fire near la ronge sask. canada	1
2	5	NaN	NaN	all residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	just got sent this photo from ruby #alaska as ...	1

```
test.head()
```

✓ 0.4s

	id	keyword	location	text
0	0	NaN	NaN	just happened a terrible car crash
1	2	NaN	NaN	heard about #earthquake is different cities. s...
2	3	NaN	NaN	there is a forest fire at spot pond. geese are...
3	9	NaN	NaN	apocalypse lighting. #spokane #wildfires
4	11	NaN	NaN	typhoon soudelor kills 28 in china and taiwan

Seguidamente se quitaron signos de puntuación y los caracteres especiales:

Eliminando signos de puntuación y caracteres especiales (@, #, \$, %, etc.)

```
#Se eliminan los signos de puntuacion
for i in range(len(train)):
    train.loc[i, 'text'] = train.loc[i, 'text'].translate(str.maketrans(string.punctuation, ' ' * len(string.punctuation)))

for i in range(len(test)):
    test.loc[i, 'text'] = test.loc[i, 'text'].translate(str.maketrans(string.punctuation, ' ' * len(string.punctuation)))
```

✓ 2.2s

```
train.head()
```

✓ 0.4s

	id	keyword	location	text	target
0	1	NaN	NaN	our deeds are the reason of this earthquake m...	1
1	4	NaN	NaN	forest fire near la ronge sask canada	1
2	5	NaN	NaN	all residents asked to shelter in place are ...	1
3	6	NaN	NaN	13 000 people receive wildfires evacuation or...	1
4	7	NaN	NaN	just got sent this photo from ruby alaska as ...	1

```
test.head()
```

✓ 0.6s

	id	keyword	location	text
0	0	NaN	NaN	just happened a terrible car crash
1	2	NaN	NaN	heard about earthquake is different cities s...
2	3	NaN	NaN	there is a forest fire at spot pond geese are...
3	9	NaN	NaN	apocalypse lighting spokane wildfires
4	11	NaN	NaN	typhoon soudelor kills 28 in china and taiwan

Luego se quitaron los stopwords, definiendo el idioma inglés.

Definiendo stopwords en idioma inglés Quitando las stopwords (the, a, an, etc.)

```
#Ref: https://www.delftstack.com/howto/python/remove-stop-words-in-python/#:~:text=The%20nltk%20(Natural%20Language%20Processing,the%20list%20from%20this%20library.
#Quitar las stopwords
stopWords = stopwords.words('english')

for i in range(len(train)):
    train.loc[i,'text'] = ' '.join(word for word in train.loc[i,'text'].split() if word not in stopWords)

for i in range(len(test)):
    test.loc[i,'text'] = ' '.join(word for word in test.loc[i,'text'].split() if word not in stopWords)
```

```
train.head()
```

	id	keyword	location	text	target
0	1	NaN	NaN	deeds reason earthquake may allah forgive us	1
1	4	NaN	NaN	forest fire near la ronge sask canada	1
2	5	NaN	NaN	residents asked shelter place notified officer...	1
3	6	NaN	NaN	13 000 people receive wildfires evacuation ord...	1
4	7	NaN	NaN	got sent photo ruby alaska smoke wildfires pou...	1

```
test.head()
```

	id	keyword	location	text
0	0	NaN	NaN	happened terrible car crash
1	2	NaN	NaN	heard earthquake different cities stay safe ev...
2	3	NaN	NaN	forest fire spot pond geese fleeing across str...
3	9	NaN	NaN	apocalypse lighting spokane wildfires
4	11	NaN	NaN	typhoon soudelor kills 28 china taiwan

Posteriormente se quitan los emojis:

```
train.head()
```

	id	keyword	location	text	target
0	1	NaN	NaN	deeds reason earthquake may allah forgive us	1
1	4	NaN	NaN	forest fire near la ronge sask canada	1
2	5	NaN	NaN	residents asked shelter place notified officer...	1
3	6	NaN	NaN	13 000 people receive wildfires evacuation ord...	1
4	7	NaN	NaN	got sent photo ruby alaska smoke wildfires pou...	1

```
test.head()
```

	id	keyword	location	text
0	0	NaN	NaN	happened terrible car crash
1	2	NaN	NaN	heard earthquake different cities stay safe ev...
2	3	NaN	NaN	forest fire spot pond geese fleeing across str...
3	9	NaN	NaN	apocalypse lighting spokane wildfires
4	11	NaN	NaN	typhoon soudelor kills 28 china taiwan

Y por último, se quitan los números, pero se decidió dejar el 911 por futura utilidad al momento de analizar los datos

Quitando números excepto el 911

```
Quitar numeros, se considera dejar 1 911 ya que es un numero representativo en EEUU y puede ser de utilidad
for i in range(len(train)):
    train.loc[i,'text'] = re.sub(r'#\S+|\d+', lambda match: match.group(0) if match.group(0).startswith('911') else '', train.loc[i,'text'])

for i in range(len(test)):
    test.loc[i,'text'] = re.sub(r'#\S+|\d+', lambda match: match.group(0) if match.group(0).startswith('911') else '', test.loc[i,'text'])

2.2s
```

```
train.head()
```

✓ 0.4s

	id	keyword	location	text	target
0	1	NaN	NaN	deeds reason earthquake may allah forgive us	1
1	4	NaN	NaN	forest fire near la ronge sask canada	1
2	5	NaN	NaN	residents asked shelter place notified officer...	1
3	6	NaN	NaN	people received wildfire evacuation orders c...	1
4	7	NaN	NaN	got sent photo ruby alaska smoke wildfires pou...	1

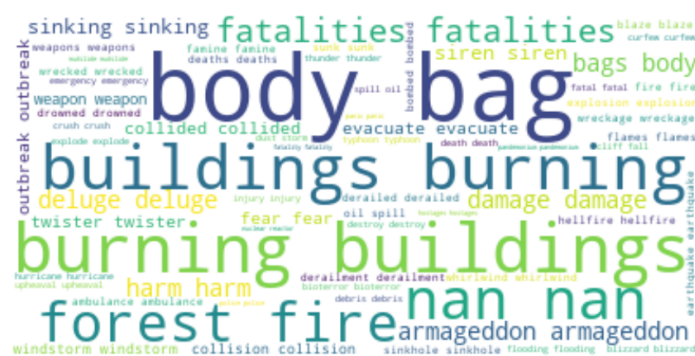
```
test.head()
```

✓ 0.4s

	id	keyword	location	text
0	0	NaN	NaN	happened terrible car crash
1	2	NaN	NaN	heard earthquake different cities stay safe ev...
2	3	NaN	NaN	forest fire spot pond geese fleeing across str...
3	9	NaN	NaN	apocalypse lighting spokane wildfires
4	11	NaN	NaN	typhoon soudelor kills china taiwan

Durante todo este proceso pudimos ir viendo cómo cambiaban - al menos - las primeras observaciones en su columna *text*, por lo que comprobamos que los valores en ambos datasets fueron arreglados según lo indicado.

Se generó una nube de palabras (*wordcloud*) para visualizar las palabras más repetidas segun los *keywords* de todos los tweets.



Clasificación de Palabras

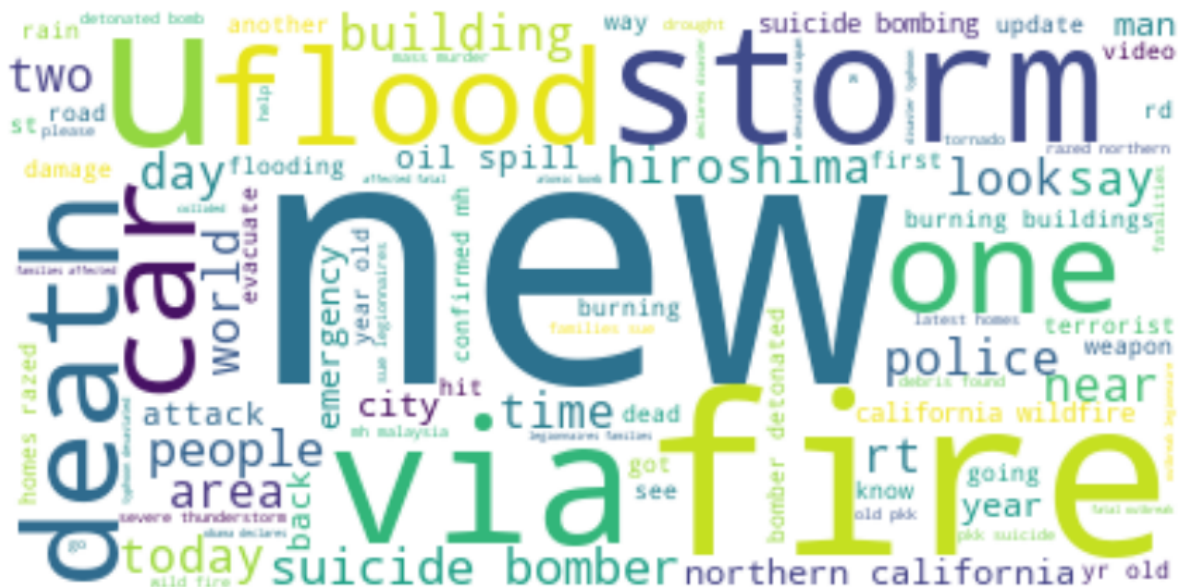
Primero separamos los tweets que son de desastres reales y los que no a partir del valor de la variable *target*. También generamos nuevamente un *wordcloud* para cada una de estas clasificaciones y contamos cuántas observaciones quedaron en cada clase.

Frecuencia de tweets de desastres

```
disaster_freq = train[train['target'] == 1]
disaster_words = []
for i in range(len(disaster_freq)):
    text = disaster_freq.iloc[i]['text'].split()
    for word in text:
        disaster_words.append(word)

keywords = " ".join(review for review in disaster_words)
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(keywords)
plt.figure(figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

✓ 0.6s



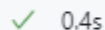
```
# count de observaciones en disaster_freq
disaster_freq['target'].value_counts()
```

✓ 0.3s

1 3271

Name: target, dtype: int64

✓ 0.9s



```
0    4342
Name: target, dtype: int64
```

Analizando bigramas

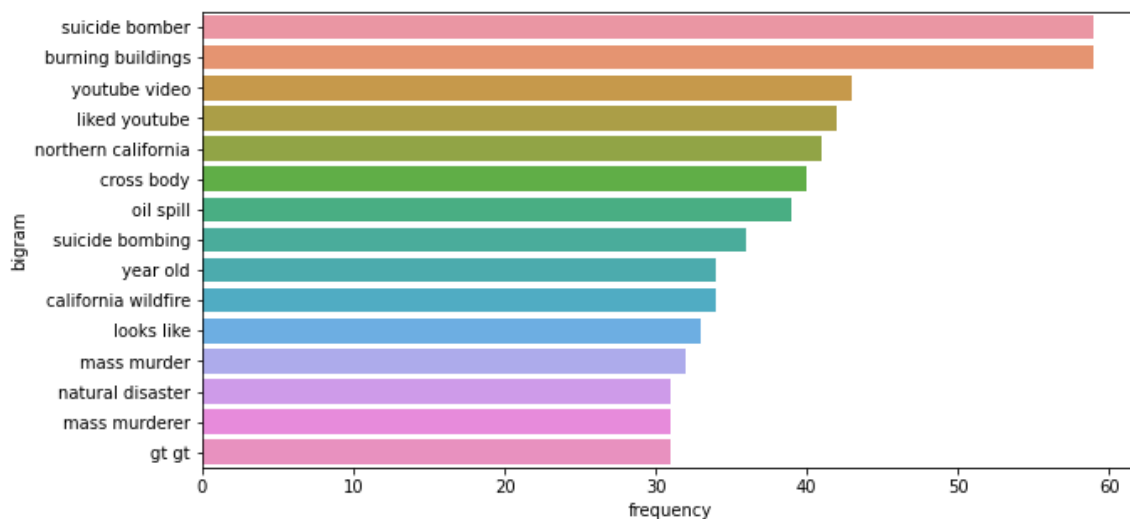
[+ Code](#)

```
# https://towardsdatascience.com/text-analysis-basics-in-python-443282942ec5
def get_bigrams(text):
    count_vectorizer = CountVectorizer(ngram_range=(2, 2))
    ngrams = count_vectorizer.fit_transform(text)
    count_values = ngrams.toarray().sum(axis=0)
    vocab = count_vectorizer.vocabulary_
    df_ngram = pd.DataFrame(sorted([(count_values[i], k) for k, i in vocab.items()],
                                   reverse=True)).rename(columns={0: 'frequency', 1: 'bigram'})
    return df_ngram

plt.figure(figsize=(10, 5))
bigrams = get_bigrams(train['text'])
x = bigrams['frequency'][:15]
y = bigrams['bigram'][:15]
sns.barplot(x=x, y=y)
```

✓ 1.3s

<AxesSubplot:xlabel='frequency', ylabel='bigram'>



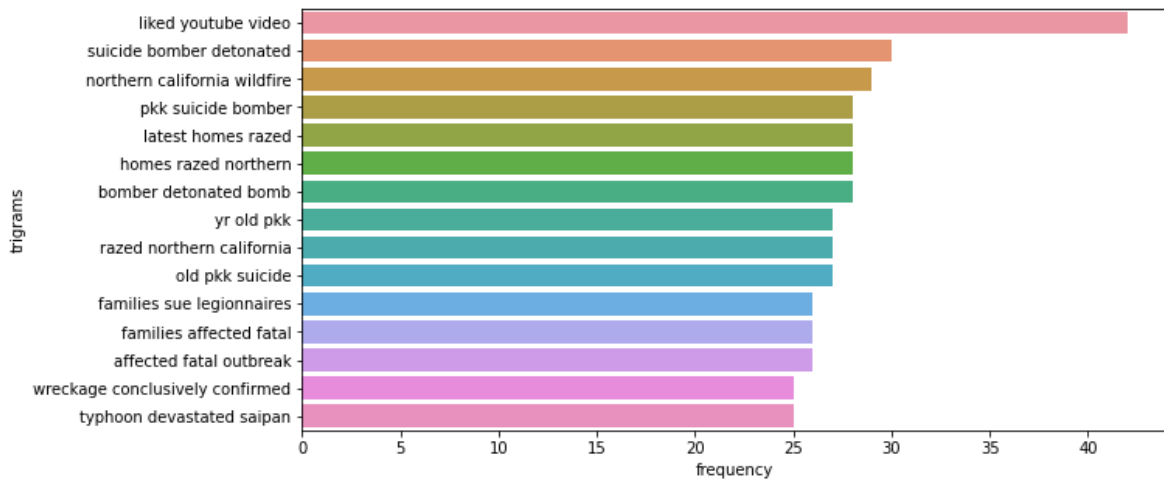
Analizando trigramas

```
# https://towardsdatascience.com/text-analysis-basics-in-python-443282942ec5
def get_trigrams(text):
    count_vectorizer = CountVectorizer(gram_range=(3, 3))
    ngrams = count_vectorizer.fit_transform(text)
    count_values = ngrams.toarray().sum(axis=0)
    vocab = count_vectorizer.vocabulary_
    df_ngram = pd.DataFrame(sorted([(count_values[i], k) for k, i in vocab.items()]), reverse=True).rename(columns={0: 'frequency', 1: 'trigrams'})
    return df_ngram

plt.figure(figsize=(10, 5))
trigrams = get_trigrams(train['text'])
x = trigrams['frequency'][:15]
y = trigrams['trigrams'][:15]
sns.barplot(x=x, y=y)
```

✓ 1.3s

<AxesSubplot: xlabel='frequency', ylabel='trigrams'>



Utilizando el corpus Vader incluido en la librería NLTK de Python y el analizador de sentimientos se obtiene un score que va de valores de -1 a 1 e indica según el rango de valor de retorno, la negatividad, positividad o neutralidad a partir del texto de cada tweet. Posteriormente se agrega dicha clasificación como una columna al dataset.

```
train.head()
```

✓ 0.9s

	id	keyword	location	text	target	score	sentiment
0	1	NaN	NaN	deeds reason earthquake may allah forgive us	1	0.2732	neutral
1	4	NaN	NaN	forest fire near la ronge sask canada	1	-0.3400	neutral
2	5	NaN	NaN	residents asked shelter place notified officer...	1	0.0000	neutral
3	6	NaN	NaN	people receive wildfires evacuation orders c...	1	0.0000	neutral
4	7	NaN	NaN	got sent photo ruby alaska smoke wildfires pou...	1	0.0000	neutral

```
test.head()
```

✓ 0.4s

	id	keyword	location	text	score	sentiment
0	0	NaN	NaN	happened terrible car crash	-0.7003	negative
1	2	NaN	NaN	heard earthquake different cities stay safe ev...	0.4404	neutral
2	3	NaN	NaN	forest fire spot pond geese fleeing across str...	-0.6159	negative
3	9	NaN	NaN	apocalypse lighting spokane wildfires	0.0000	neutral
4	11	NaN	NaN	typhoon soudelor kills china taiwan	-0.5423	negative

Clasificación de tweets en positivo, negativo y neutro. Variable de “negatividad” del tweet.

Tener el compuesto de los scores de todas las palabras de cada tweet nos permite clasificar el tweet en sí como positivo, negativo o neutro.

A partir del *score* retornado por el modelo de analizador de sentimientos, se verifica si este tiene un valor mayor a 0.5 entonces es un tweet positivo, si va entre -0.5 y 0.5 es un tweet neutral y si va menor a -0.5 entonces es un tweet negativo.

Resultados.

Con esto podemos obtener *insights* sobre los tweets a partir de su negatividad y positividad. Cómo obtener los tweets más negativos de todos o los más positivos.

¿Cuáles son los 10 tweets más negativos? ¿En qué categoría están?

La mayoría de los 10 tweets más negativos están dentro de la categoría de catástrofes reales, por ejemplo atentados con bombas y situaciones donde hubieron múltiples personas heridas. Por otro lado, una pequeña parte de los tweets se clasifican dentro de la categoría de cosas que no ocurrieron pero hablan acerca del miedo y el terror.

<pre># Los 10 tweets mas negativos train[train['sentiment'] == 'negative'].sort_values(by=['score'])[:10]</pre>							
✓ 0.7s							
	id	keyword	location	text	target	score	sentimen
7472	10689	wreck	NaN	wreck wreck wreck wreck wreck wreck wreck wrec...	0	-0.9879	negativ
2844	4089	displaced	NaN	peterjukes crime killed displaced millions sys...	1	-0.9652	negativ
6411	9166	suicide bomber	NaN	suicide bomber kills saudi security site mosq...	1	-0.9623	negativ
6393	9137	suicide bomb	Worldwide	th day since jul nigeria suicide bomb attack...	1	-0.9595	negativ
6407	9159	suicide bomber	Worldwide	killed sarabia mosque suicide bombing suicide...	1	-0.9552	negativ
5240	7493	obliteration	23 countries and counting!	fear mind killer fear little death brings tota...	0	-0.9552	negativ
472	682	attack	portland, oregon	illegal alien released obama dhs times charge...	1	-0.9538	negativ
1540	2225	chemical emergency	Las Vegas, Nevada	bomb crash loot riot emergency pipe bomb nucle...	1	-0.9524	negativ
6930	9940	trouble	NaN	cspan prez mr president biggest terrorist trou...	1	-0.9493	negativ
2932	4213	drowned	Pembroke NH	lake sees dead fish poor little guy wonder hap...	0	-0.9477	negativ

¿Cuáles son los 10 tweets más positivos? ¿En qué categoría están?

En cuanto a los 10 tweets más positivos, podemos encontrar que las categorías hablan sobre situaciones reales y sobre deseos de las personas

<pre># Los 10 tweets mas positivos train[train['sentiment'] == 'positive'].sort_values(by=['score'], ascending=False)[:10]</pre>							
✓ 0.6s							
	id	keyword	location	text	target	score	sentiment
6992	10028	twister	NaN	check want twister tickets vip experience see ...	0	0.9682	positive
3382	4844	evacuation	Renfrew, Scotland	batfanuk enjoyed show today great fun emergenc...	0	0.9423	positive
3163	4541	emergency	Renfrew, Scotland	batfanuk enjoyed show today great fun emergenc...	0	0.9423	positive
6292	8989	storm	NaN	todays storm pass let tomorrows light greet ki...	1	0.9403	positive
7182	10291	weapon	NaN	roguewatson nothing wrong lethal weapon series...	0	0.9365	positive
344	493	armageddon	#FLIGHTCITY UK	official vid thereal gt gt gt gt gt gt trubgme...	0	0.9313	positive
2238	3198	deluge	NaN	meditationbysmg ppl got method meditation u p...	0	0.9287	positive
6295	8994	stretcher	NaN	free ebay sniping rt lumbar extender back stre...	0	0.9260	positive
6560	9386	survived	Puerto Rico	duchovbutt starbuck scully madmakny davidducho...	0	0.9217	positive
4154	5903	harm	Gotham City	never escape bullets harm nothing harms know p...	1	0.9196	positive

¿Son los tweets de la categoría que indica que habla de un desastre real más negativos que los de la otra categoría?

Se puede ver que predominan los tweets neutros pero que en los tweets de desastres no reales hay más predominancia de tweets negativos sobre los positivos. Los tweets que hablan más acerca de desastres reales si tienen una puntuación más negativa que aquellos tweets que no describen una situación real. Como se puede observar en la imagen de los primeros 5 tweets, 3 de ellos son reales, como lo indica la variable target.

```
# Cantidad de 'sentiment' entre los tweets con target 0
disaster_tweets = train[train['target'] == 0]
disaster_tweets['sentiment'].value_counts()
✓ 0.3s
```

neutral	2901
negative	862
positive	579

Name: sentiment, dtype: int64

```
# Cantidad de 'sentiment' entre los tweets con target 0
not_disaster_tweets = train[train['target'] == 1]
not_disaster_tweets['sentiment'].value_counts()
✓ 0.3s
```

neutral	1966
negative	1153
positive	152

Name: sentiment, dtype: int64

Cree una variable que contenga la “negatividad” de cada tweet. Inclúyala en el conjunto de datos y entrene nuevamente el modelo de clasificación de la hoja pasada.

¿La inclusión de esta variable mejoró los resultados del modelo de clasificación?

Se agregó la variable score a cada uno de las filas en el conjunto de datos, mejoró los resultados ya que nos permitió medir cuantitativamente que tan positivos o negativos son los tweets, lo cual nos permite poder identificar con mayor facilidad dentro de qué clasificación entra cada uno de ellos y también poder determinar los principales temas de los tweets positivos y negativos.

Referencias

Akladyous, Boula. (2021). sentiment Analysis using VADER. Medium. Extraído de: <https://akladyous.medium.com/sentiment-analysis-using-vader-c56bcffe6f24>

Yang, Sophia. (2020). Text analysis basics in Python. Towards Data Science. Extraído de: <https://towardsdatascience.com/text-analysis-basics-in-python-443282942ec5>