Patitos / laboratori... Published at Sep 19, 2022 Private View source El-Mendez

- Orlando Cabrera #19943
- Diana Zaray Corado #191025
- Pablo Méndez #19195

Laboratorio 5 - Análisis de Sentimientos

Utilice el dataset Natural Language Processing with Disaster Tweets de Kaggle. Debe hacer un análisis exploratorio para entender mejor los datos, sabiendo que el objetivo final clasificar si un tweet se refiere a un desastre real no. Genere un informe en pdf con las explicaciones de los pasos que llevó a cabo y los resultados obtenidos. Recuerde que la investigación debe ser reproducible por lo que debe guardar el código que ha utilizado para resolver los ejercicios y/o cada uno de los pasos llevados a cabo si utiliza una herramienta visual. Incluya una nube de palabras que le ayude a detectar las que más se repiten.

```
from textblob import TextBlob
import pandas as pd
import re
import nltk
import seaborn as sns
sns.set_theme(style="whitegrid", palette='Set2')
from collections import defaultdict
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
True
```

```
data = pd.read_csv("train.csv")
data
```

	id int64	keyword object	location object	text object	target int64

0	1	nan	nan	Our Deeds are the Reason of…	1
1	4	nan	nan	Forest fire near La Ronge…	1
2	5	nan	nan	All residents asked to…	1
3	6	nan	nan	13,000 people receive…	1
4	7	nan	nan	Just got sent this photo fro…	1
5	8	nan	nan	#RockyFire Update =>…	1
6	10	nan	nan	#flood #disaster Heav…	1
7	13	nan	nan	I'm on top of the hill and I…	1
8	14	nan	nan	There's an emergency…	1
9	15	nan	nan	I'm afraid that the tornado is…	1

Descripción de los datos

El conjunto de datos se llama Natural Language Processing with Disaster Tweets, el cual fue obtenido gracias a la página Kaggle. Posee un total de 7613 observaciones. Cuenta con 5 variables categóricas, las cuales son:

- id
- keyword
- location
- text
- target

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7613 entries, 0 to 7612
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
```

```
 0   id        7613 non-null   int64
 1   keyword   7552 non-null   object
 2   location  5080 non-null   object
 3   text      7613 non-null   object
 4   target    7613 non-null   int64
dtypes: int64(2), object(3)
memory usage: 297.5+ KB
```

Preprocesamiento

```
data.head()
```

	id int64	keyword object	location object	text object	target int64
0	1	nan	nan	Our Deeds are the Reason of…	1
1	4	nan	nan	Forest fire near La Ronge…	1
2	5	nan	nan	All residents asked to…	1
3	6	nan	nan	13,000 people receive…	1
4	7	nan	nan	Just got sent this photo fro…	1

Convertir el Texto a Minúsculas

```
data['text'] = data['text'].str.lower()
data.head()
```

	id int64	keyword object	location object	text object	target int64
0	1	nan	nan	our deeds are the reason of…	1
1	4	nan	nan	forest fire near la ronge…	1
2	5	nan	nan	all residents asked to…	1
3	6	nan	nan	13,000 people	1

4	7	nan	nan	just got sent this photo fro…	1

Quitar URLs

```
expression = r"(?i)\b((?:https?://|www\d{0,3}[.]|[a-z0-9.\-]+[.][a-z]{2,4}/)(?:[^\s()<>]+|
```

```
data['text'] = data['text'].apply(lambda s: re.sub(expression, "", s))
data.head()
```

	id int64	keyword object	location object	text object	target int64
0	1	nan	nan	our deeds are the reason of…	1
1	4	nan	nan	forest fire near la ronge…	1
2	5	nan	nan	all residents asked to…	1
3	6	nan	nan	13,000 people receive…	1
4	7	nan	nan	just got sent this photo fro…	1

Remover caracteres especiales

Se eliminarán todas las palabras que empiezan con '@', ya que hacen referencia a un usuario. Esto es para evitar que la red neuronal sea afectada por nombres de usuarios complejos, como *@wildifires_are_bad*.

```
expression = r"\B@\w*"
data['text'] = data['text'].apply(lambda s: re.sub(expression, "", s))
data.head()
```

	id int64	keyword object	location object	text object	target int64
0	1	nan	nan	our deeds are the reason of…	1
1	4	nan	nan	forest fire	1

2	5	nan	nan	all residents asked to…	1
3	6	nan	nan	13,000 people receive…	1
4	7	nan	nan	just got sent this photo fro…	1

Para facilitar el aprendizaje del modelo, se removerán todos los caracteres que no son letras.

```
expression = "[^a-z ]"
data['text'] = data['text'].apply(lambda s: re.sub(expression, "", s))
data.head()
```

	id int64	keyword object	location object	text object	target int64
0	1	nan	nan	our deeds are the reason of…	1
1	4	nan	nan	forest fire near la ronge…	1
2	5	nan	nan	all residents asked to…	1
3	6	nan	nan	people receive wildfires…	1
4	7	nan	nan	just got sent this photo fro…	1

Quitar las "stopwords"

Los *stopwords* son palabras que no proporcionan mucho significado a una frase desde el punto de vista de ML. La librería nltk proporciona una lista de stopwords en diferentes idiomas.

```
stopwords = nltk.corpus.stopwords.words('english')

def remove_stopwords(s):
    words = [word for word in s.split(" ") if not word in stopwords]
    return " ".join(words)

data['text'] = data['text'].apply(remove_stopwords)
data.head()
```

	id int64	keyword object	location object	text object	target int64

0	1	nan	nan	deeds reason earthquake may…	1
1	4	nan	nan	forest fire near la ronge…	1
2	5	nan	nan	residents asked shelter place…	1
3	6	nan	nan	people receive wildfires…	1
4	7	nan	nan	got sent photo ruby alaska…	1

Unigramas y Bigramas

```
real_disaster = data[data.target == 1].text
fake_disaster = data[data.target == 0].text
real_disaster
```

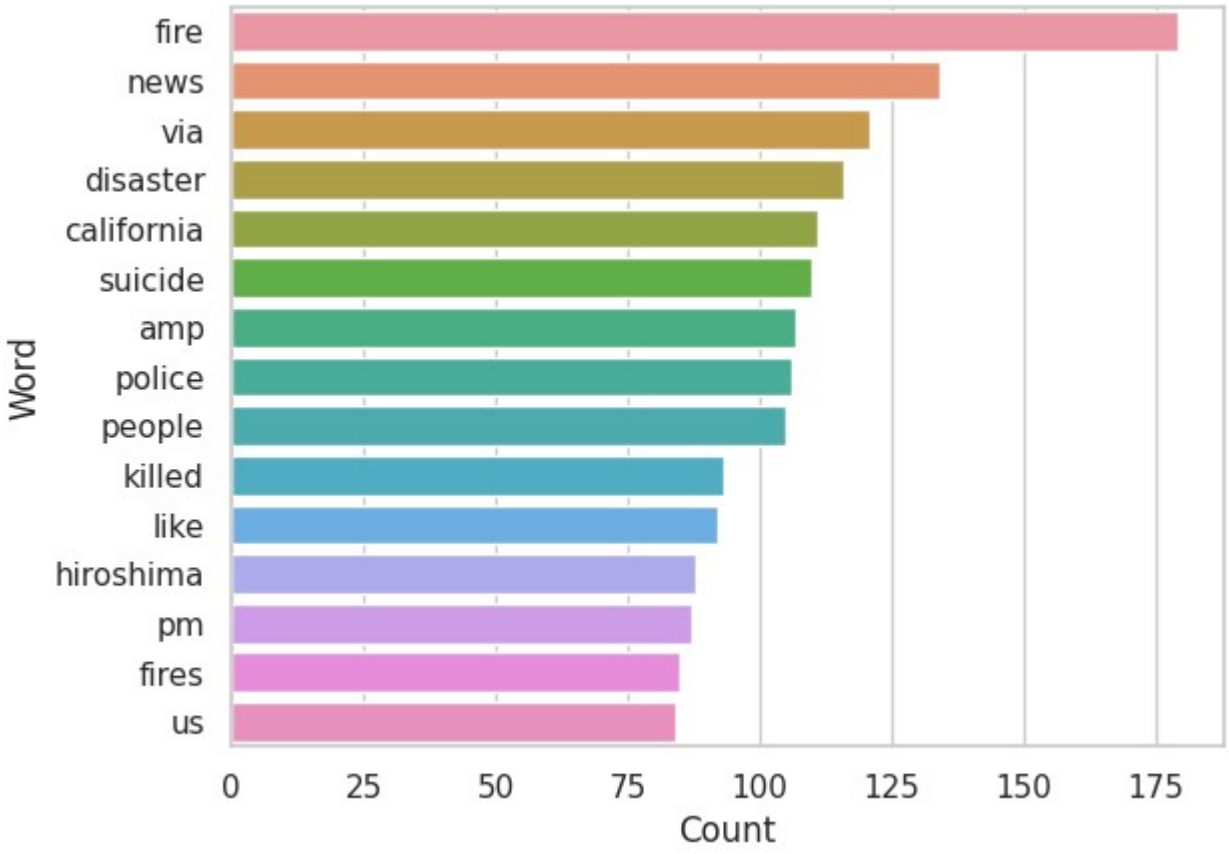
```
0            deeds reason earthquake may allah forgive us
1                   forest fire near la ronge sask canada
2       residents asked shelter place notified officer...
3        people receive wildfires evacuation orders ca...
4       got sent photo ruby alaska smoke wildfires pou...
                              ...
7608    two giant cranes holding bridge collapse nearb...
7609      control wild fires california even northern ...
7610                               utckm volcano hawaii
7611    police investigating ebike collided car little...
7612    latest homes razed northern california wildfir...
Name: text, Length: 3271, dtype: object
```

```
real_unigrams = real_disaster.apply(lambda s: TextBlob(s).ngrams(n=1))
fake_unigrams = fake_disaster.apply(lambda s: TextBlob(s).ngrams(n=1))
real_bigrams = real_disaster.apply(lambda s: TextBlob(s).ngrams(n=2))
fake_bigrams = fake_disaster.apply(lambda s: TextBlob(s).ngrams(n=2))
# Frecuencia de palabras en un desastre real
unireal = defaultdict(int)
unifake = defaultdict(int)
bireal = defaultdict(int)
bifake = defaultdict(int)
for gram in real_unigrams:
    for word in gram:
        unireal[word[0]] += 1
for gram in fake_unigrams:
    for word in gram:
```

```
        unifake[word[0]] += 1
for word in real_bigrams:
    for gram in word:
        bireal[' '.join(gram)] += 1
for word in fake_bigrams:
    for gram in word:
        bifake[' '.join(gram)] += 1
# visualizacion de frecuencias
df_unireal = pd.DataFrame([[key, unireal[key]] for key in unireal.keys()], columns=['Word'
df_unifake = pd.DataFrame([[key, unifake[key]] for key in unifake.keys()], columns=['Word'
df_bireal = pd.DataFrame([[key, bireal[key]] for key in bireal.keys()], columns=['Word', '
df_bifake = pd.DataFrame([[key, bifake[key]] for key in bifake.keys()], columns=['Word', '
```

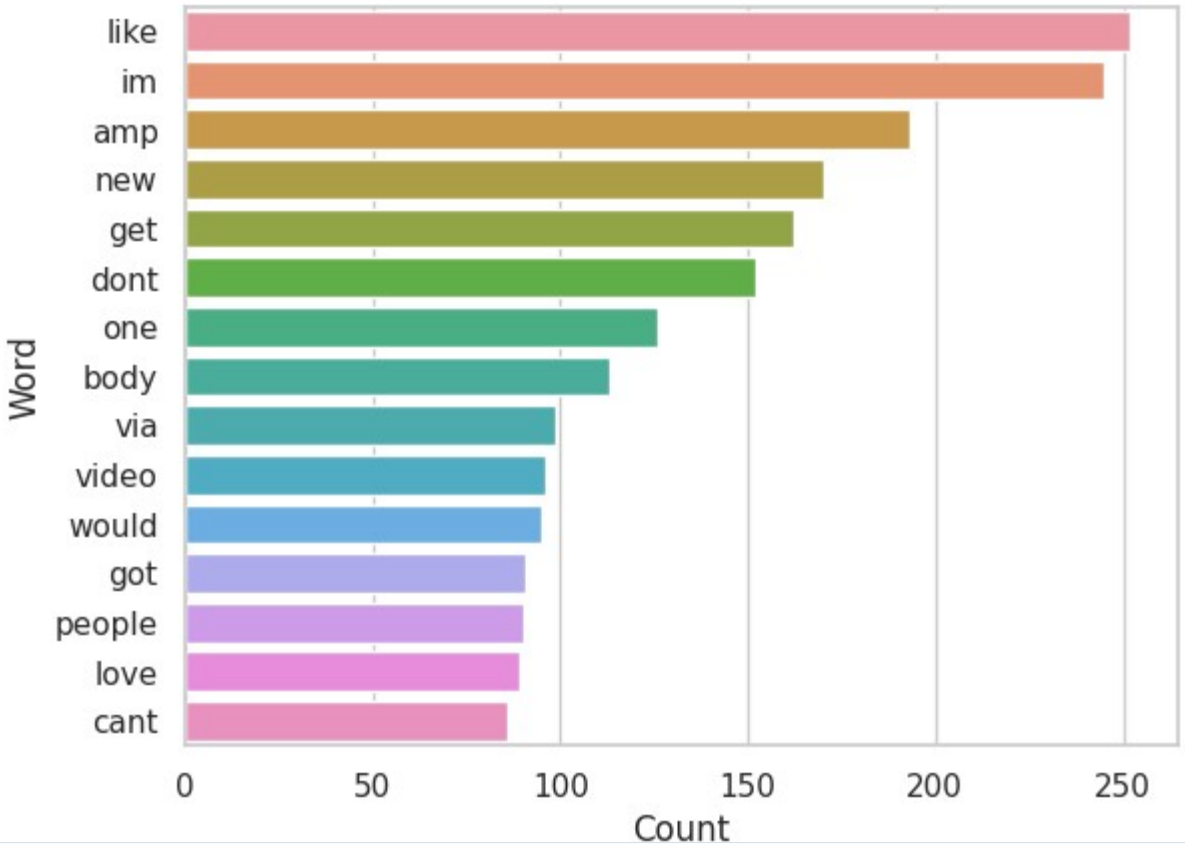
```
sns.barplot(y=df_unireal.Word.head(15), x=df_unireal.Count.head(15))
```

```
<AxesSubplot:xlabel='Count', ylabel='Word'>
```



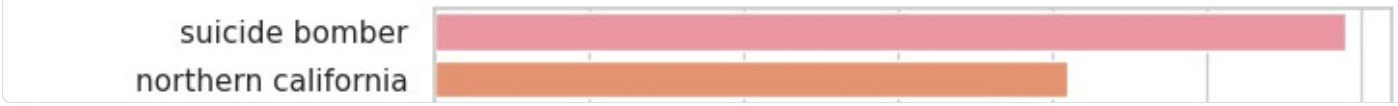

```
sns.barplot(y=df_unifake.Word.head(15), x=df_unifake.Count.head(15))
```

```
<AxesSubplot:xlabel='Count', ylabel='Word'>
```



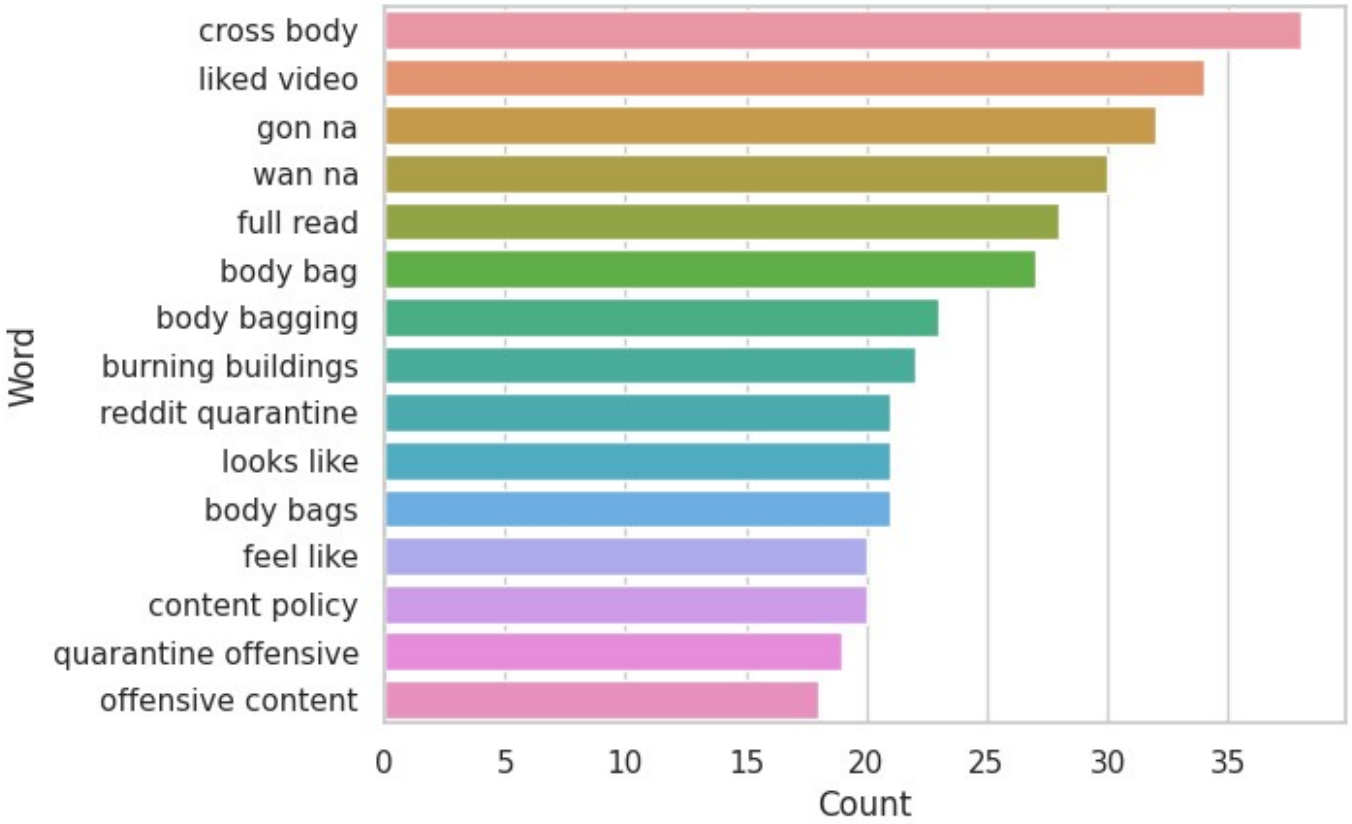

```
sns.barplot(y=df_bireal.Word.head(15), x=df_bireal.Count.head(15))
```

```
<AxesSubplot:xlabel='Count', ylabel='Word'>
```




```
sns.barplot(y=df_bifake.Word.head(15), x=df_bifake.Count.head(15))
```

```
<AxesSubplot:xlabel='Count', ylabel='Word'>
```



Algoritmo de Clasificación en Tweets

```
data['positivity'] = data['text'].apply(lambda s: TextBlob(s).sentiment.polarity)
data.head()
```

	id int64	keyword object	location object	text object	target int64	p
1	4	nan	nan	forest fire near la ronge…	1	

3	6	nan	nan	people receive wildfires…	1	
4	7	nan	nan	got sent photo ruby alaska…	1	
2	5	nan	nan	residents asked shelter place…	1	