Luciana Diaz Kralj, Roberto José Catalán, Federico Kaplun

Trigger Warnings: Topic Modelling en reseñas de libros

82.18 - Procesamiento del Lenguaje Natural

01

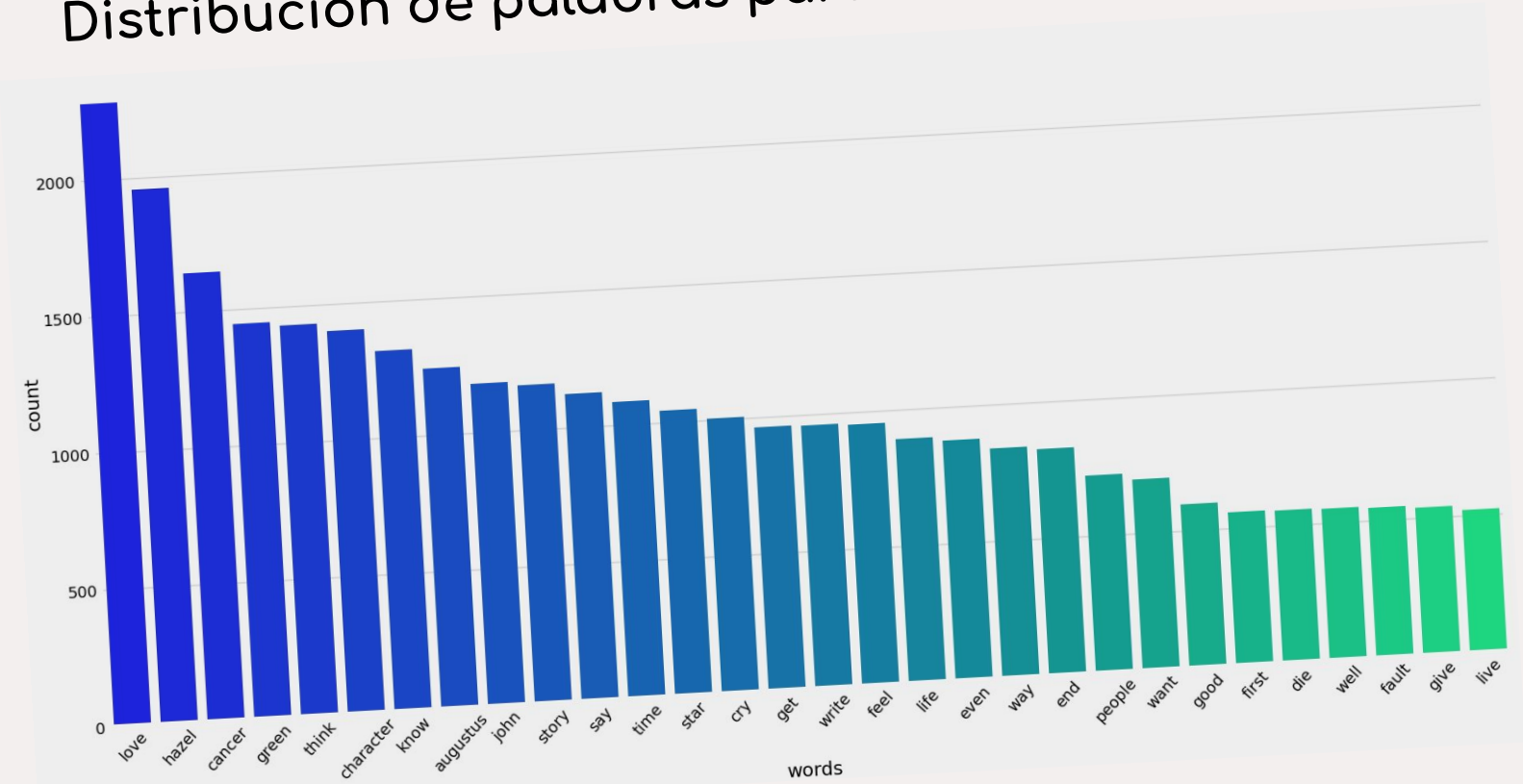
Resultados previos

Nuestro objetivo

Hacer **topic modelling** sobre **reseñas de libros**, sacando los temas principales y comparando dichos tópicos contra un diccionario de **trigger warnings**, no sólo textualmente sino también por su cercanía en temática.

assault, animal abuse

Distribución de palabras para un libro en particular



02

Las mejoras

Obtención de datos

Utilizamos el dataset de Kaggle goodreads-books-reviews-290312, el cual está compuesto por reseñas (reviews) para libros que contienen en su mayoría spoilers sobre la trama. Cuenta con dos datasets, uno para training y otro para testing.

Para el armado del corpus, combinamos ambos datasets y nos quedamos con los campos de interés:

	book_id	review_id	rating	review_text
0	18245960	dfdbb7b0eb5a7e4c26d59a937e2e5feb	5.0	This is a special book. It started slow for about the first third, then in t...
1	16981	a5d2c3628987712d0e05c4f90798eb67	3.0	Recommended by Don Katz. Avail for free in December: http://www.audible.com/...
2	28684704	2ede853b14dc4583f96cf5d120af636f	3.0	A fun, fast paced science fiction thriller. I read it in 2 nights and couldn...
3	27161156	ced5675e55cd9d38a524743f5c40996e	0.0	Recommended reading to understand what is going on in middle america, and po...
4	25884323	332732725863131279a8e345b63ac33e	4.0	I really enjoyed this book, and there is a lot to recommend it. It did drag ...

Técnicas de preprocesamiento

Anteriormente aplicamos técnicas de **tokenización** y **lematización** haciendo uso de la librería **Natural Language Toolkit** (NLTK), pero para esta entrega utilizamos librerías especiales de **Gensim** para poder procesar también **bigramas** y t**rigramas**:

```
from gensim.models import Phrases

# Build the bigram and trigram models
bigram = Phrases(data_words, min_count=5, threshold=100) # higher threshold fewer phrases.
trigram = Phrases(bigram[data_words], threshold=100)
bigram_mod = gensim.models.phrases.Phraser(bigram)
trigram_mod = gensim.models.phrases.Phraser(trigram)
```

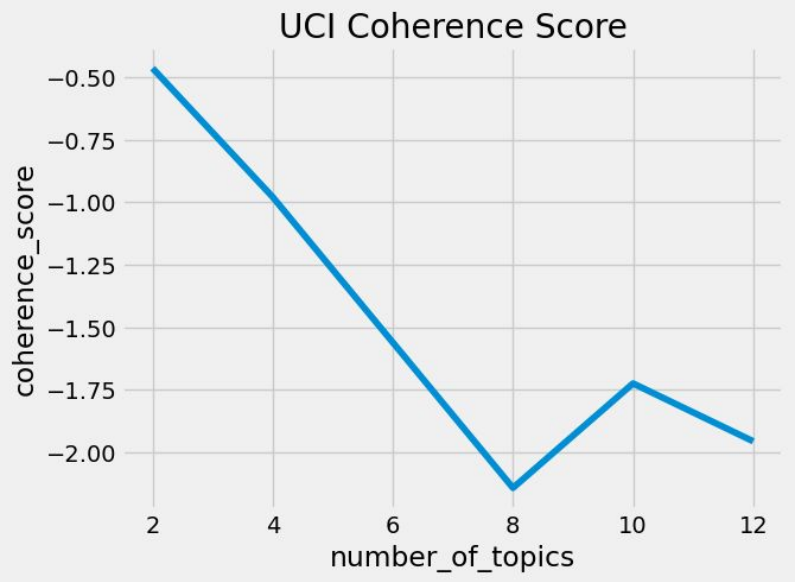
Técnicas de preprocesamiento

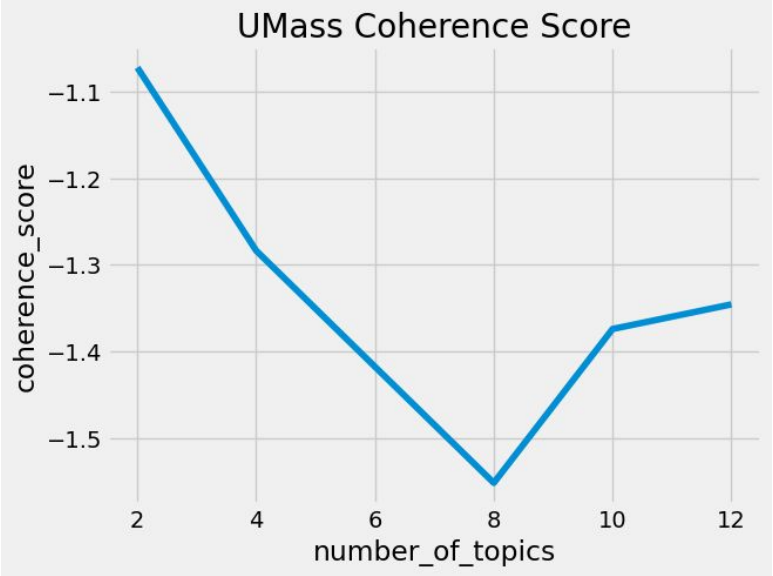
```
from gensim.models import simple_preprocess

def process_words(texts, stop_words=stop_words, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV']):
    texts = [[word for word in simple_preprocess(str(doc)) if word not in stop_words] for doc in texts]
    texts = [bigram_mod[doc] for doc in texts]
    texts = [trigram_mod[bigram_mod[doc]]for doc in texts]
    texts_out = []
    nlp = spacy.load('en_core_web_sm', disable=['parser', 'ner'])
    for sent in texts:
        doc = nlp(" ".join(sent))
        texts_out.append([token.lemma_for token in doc if token.pos_ in allowed_postags])
    texts_out = [[word for word in simple_preprocess(str(doc)) if word not in stop_words] for doc in texts_out]
    return texts_out
```

Armado de clusters

Aplicamos el modelo de LDA de **Gensim** variando el número de clusters, obteniendo el mejor resultado con 8.





Dado que los textos son cortos, luego de los 8 clústers empieza a repetir

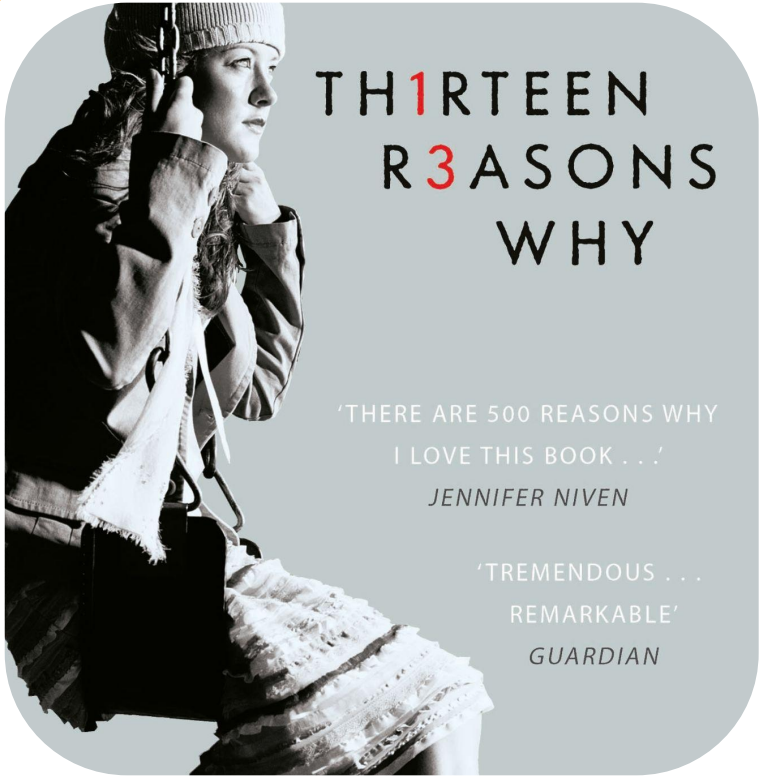
Aplicación de LDA

```
data_ready = process_words(data_words)
id2word = corpora.Dictionary(data_ready)
corpus = [id2word.doc2bow(text) for text in data_ready]

lda_model = gensim.models.ldamodel.LdaModel(corpus = corpus,
                                            id2word = id2word,
                                            num_topics = 8,
                                            random_state = 100,
                                            update_every = 1,
                                            chunksize = 10,
                                            passes = 10,
                                            alpha = 'symmetric',
                                            iterations = 100,
                                            per_word_topics = True)
```

03

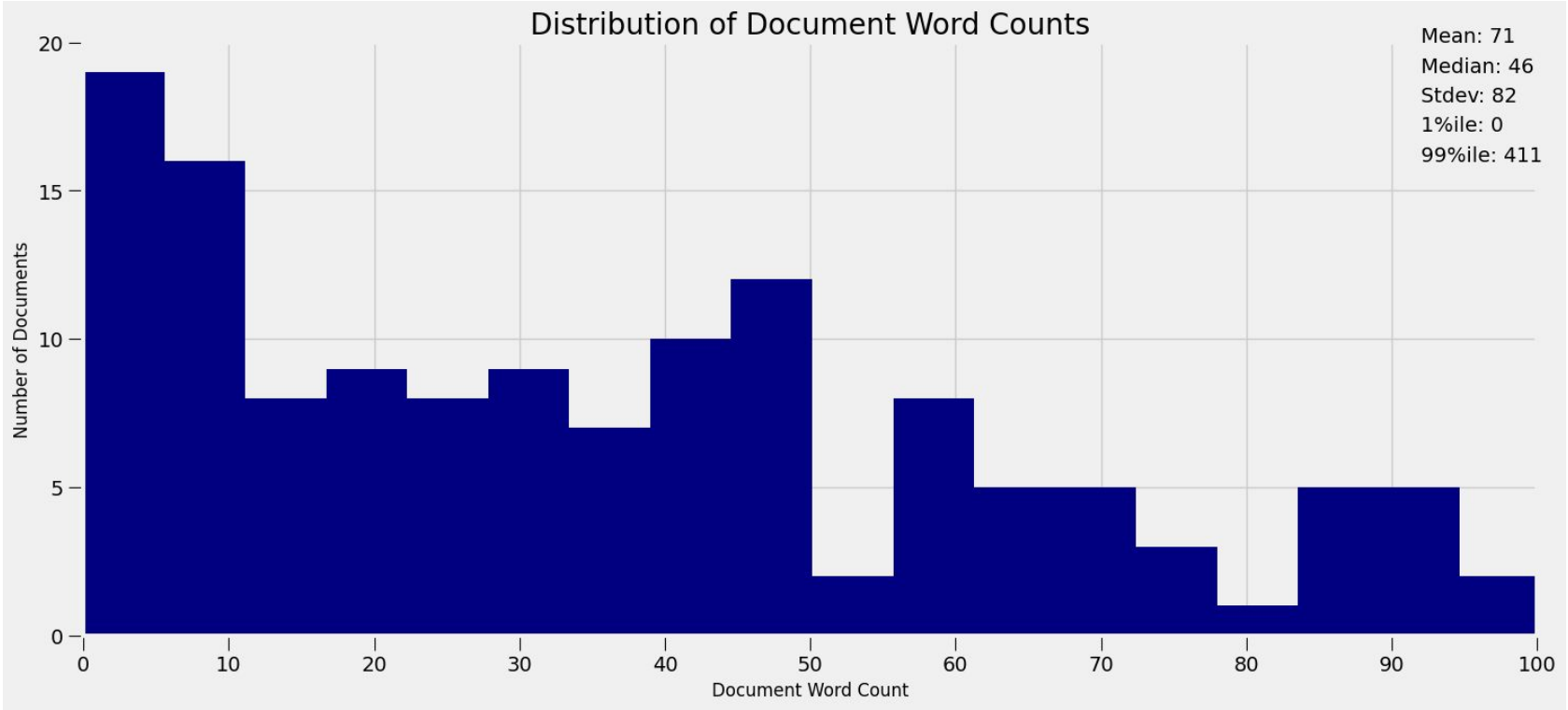
Resultados obtenidos



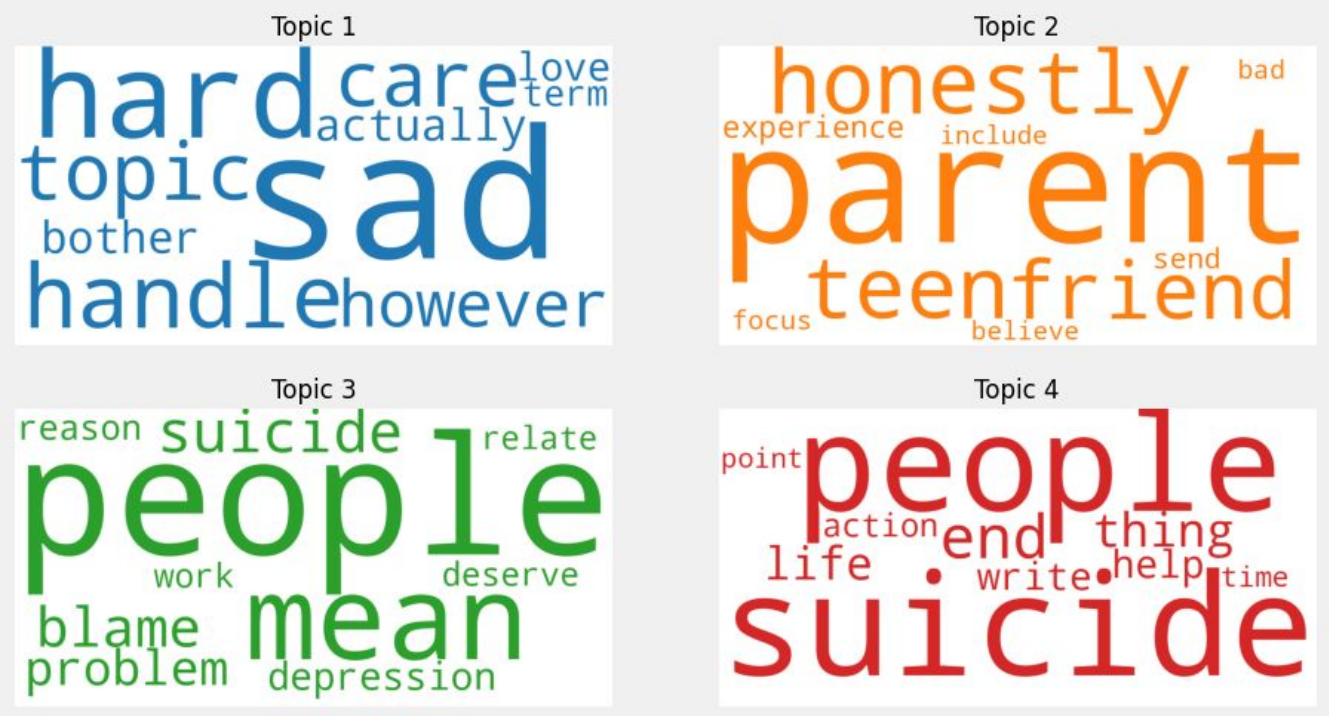
13 Reasons Why

- En base a varios intentos determinamos que era uno de los que mejores resultados aportaba.
- Contiene una gran cantidad de comentarios relevantes para el análisis.
- Contiene Trigger Warnings claros y relevantes a la historia

Distribución de la cantidad de palabras por review



Wordcloud de cada tópico



Wordcloud de cada tópico

Trigger Warnings encontrados

Rape

Topic 5

Death

Topic 5

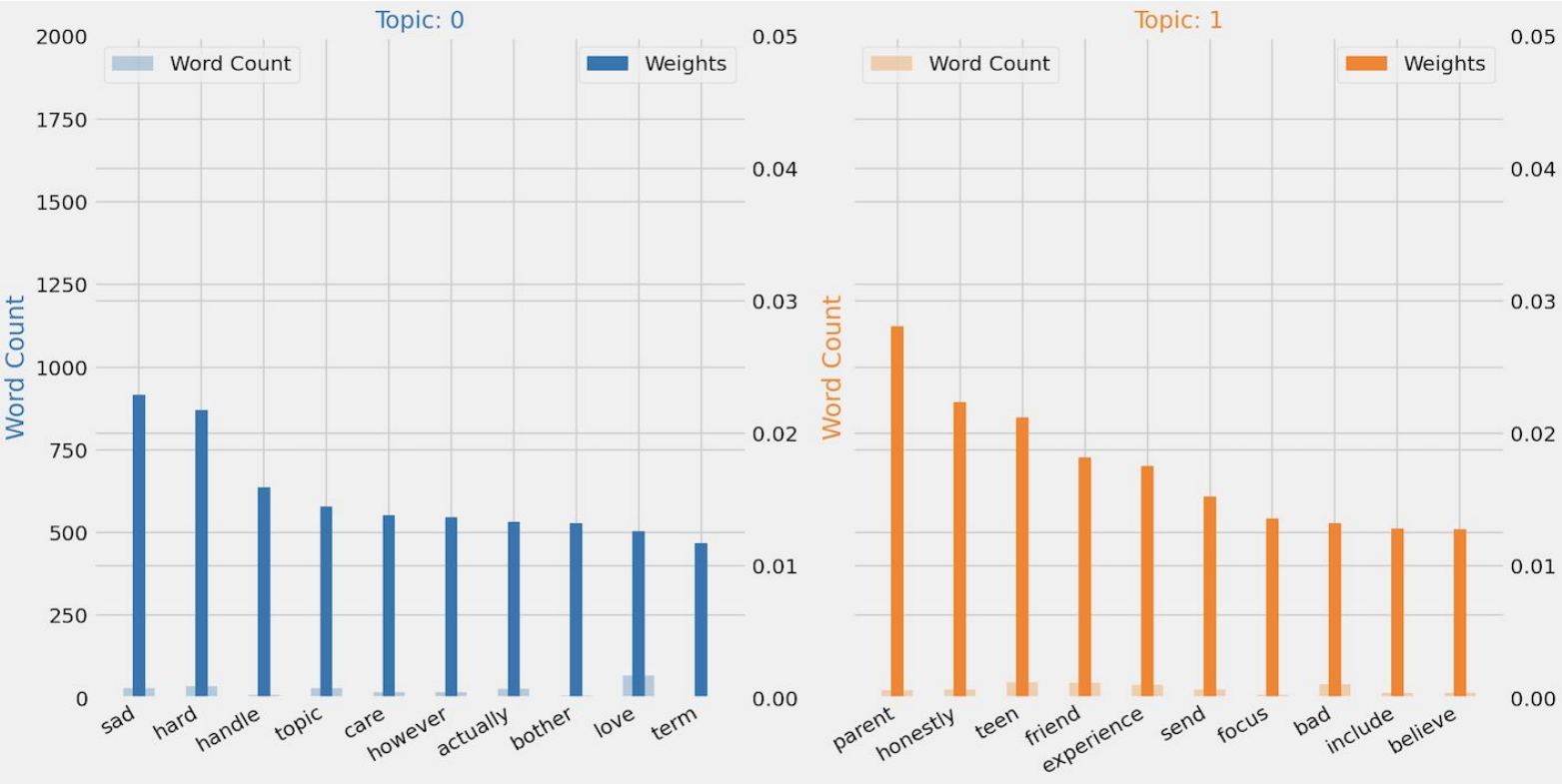
Suicide

Topic 4

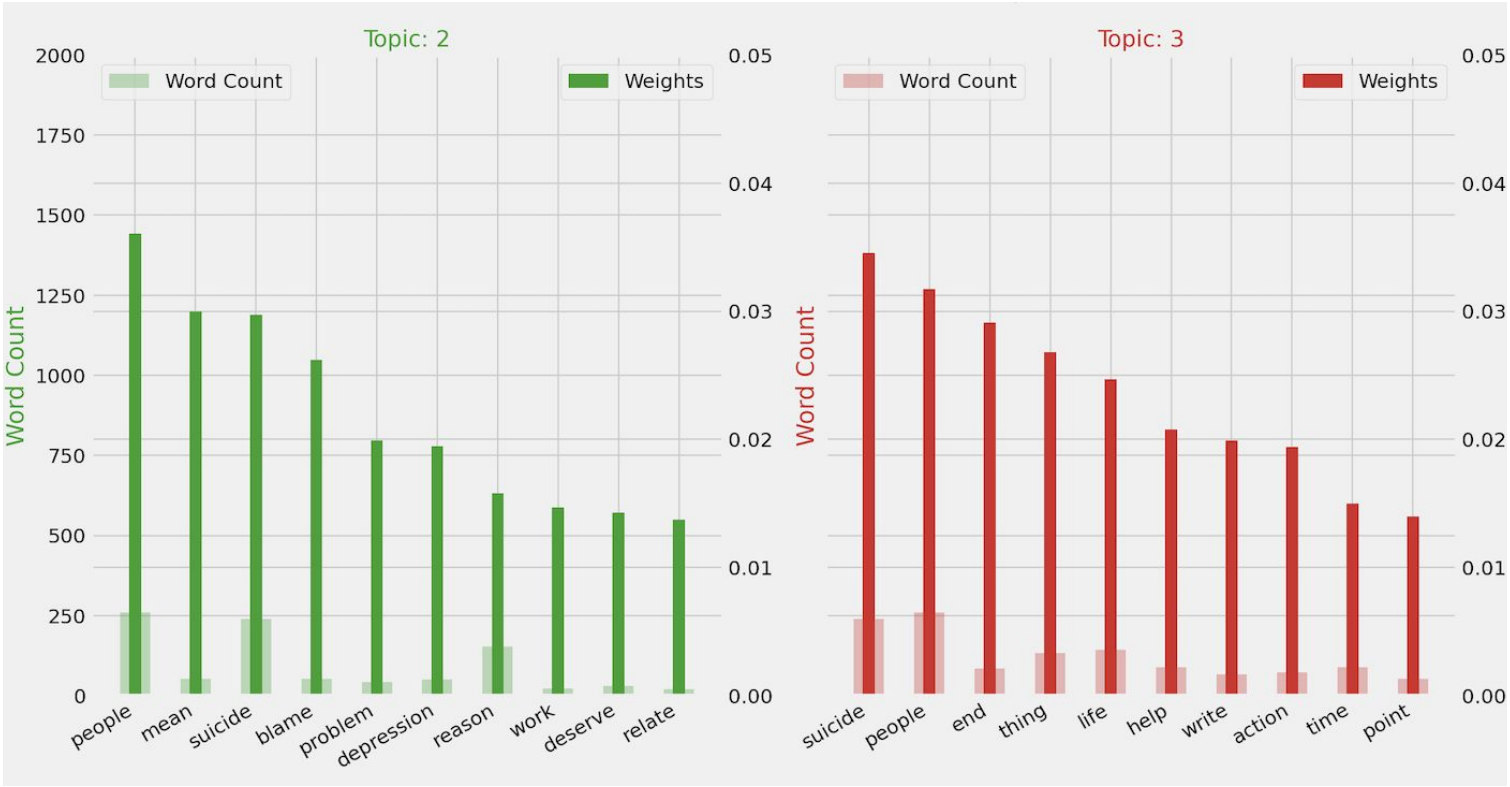
Depression

Topic 3

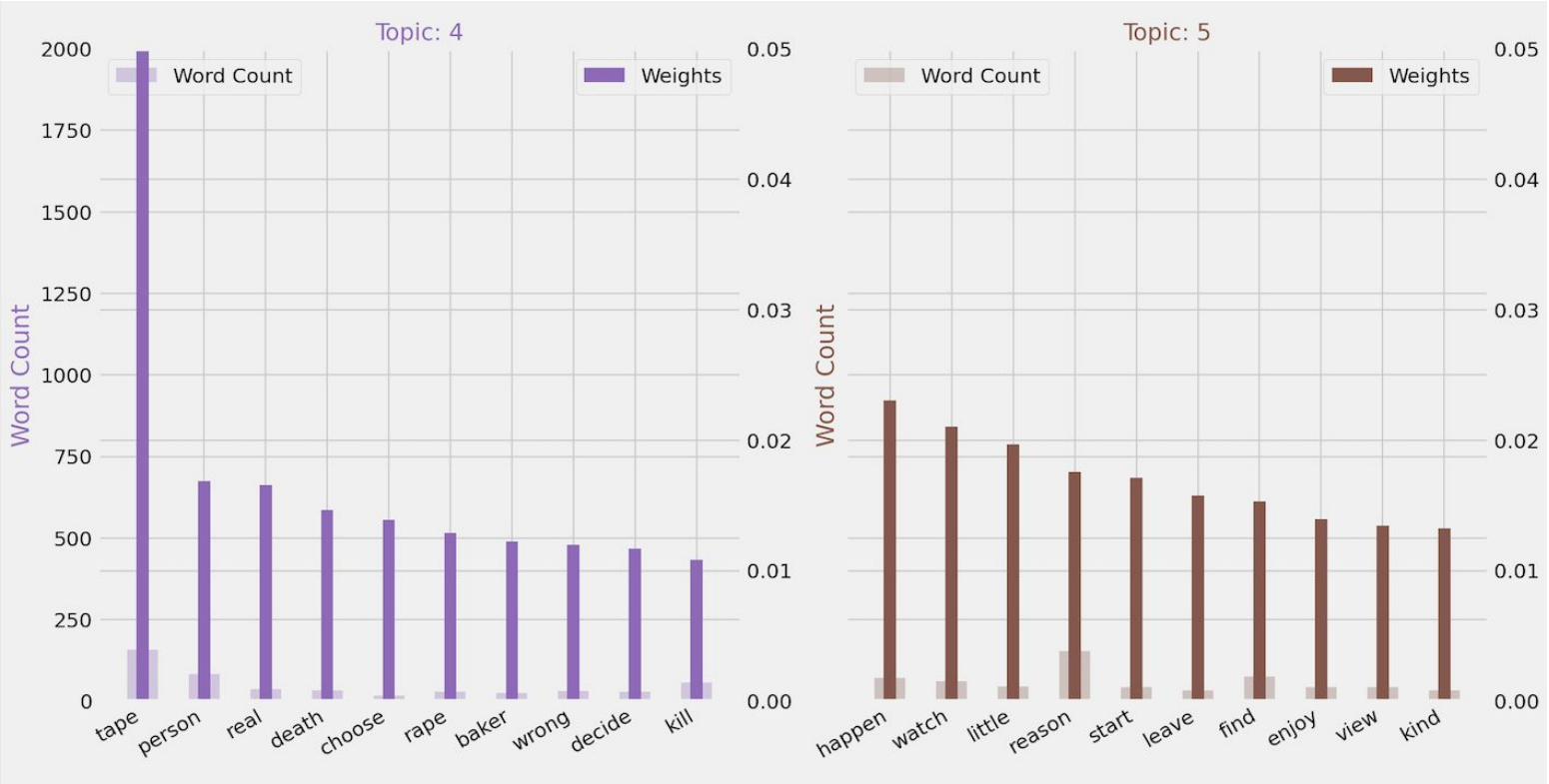
Cantidad de palabras y peso en el tópico



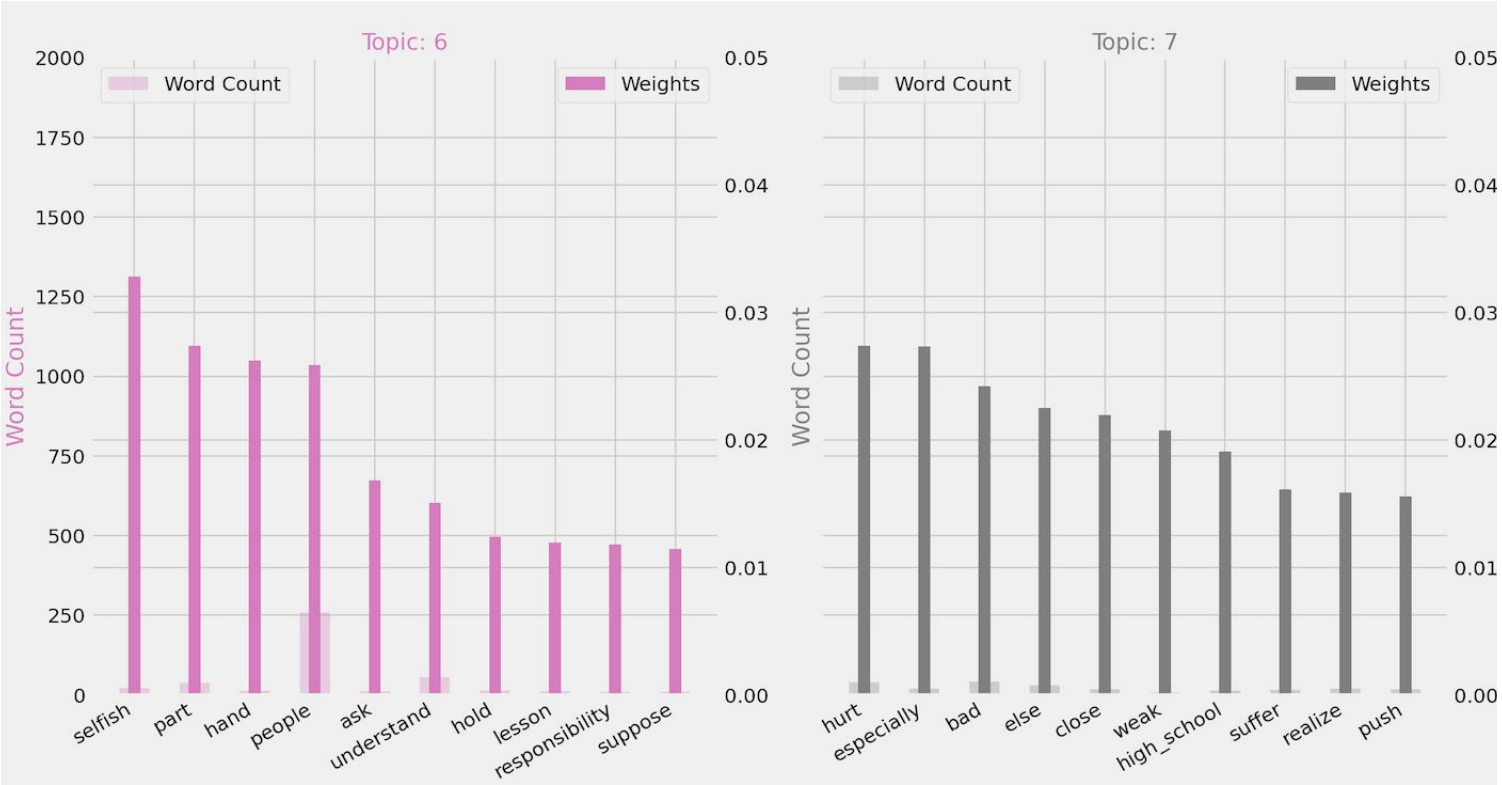
Cantidad de palabras y peso en el tópico



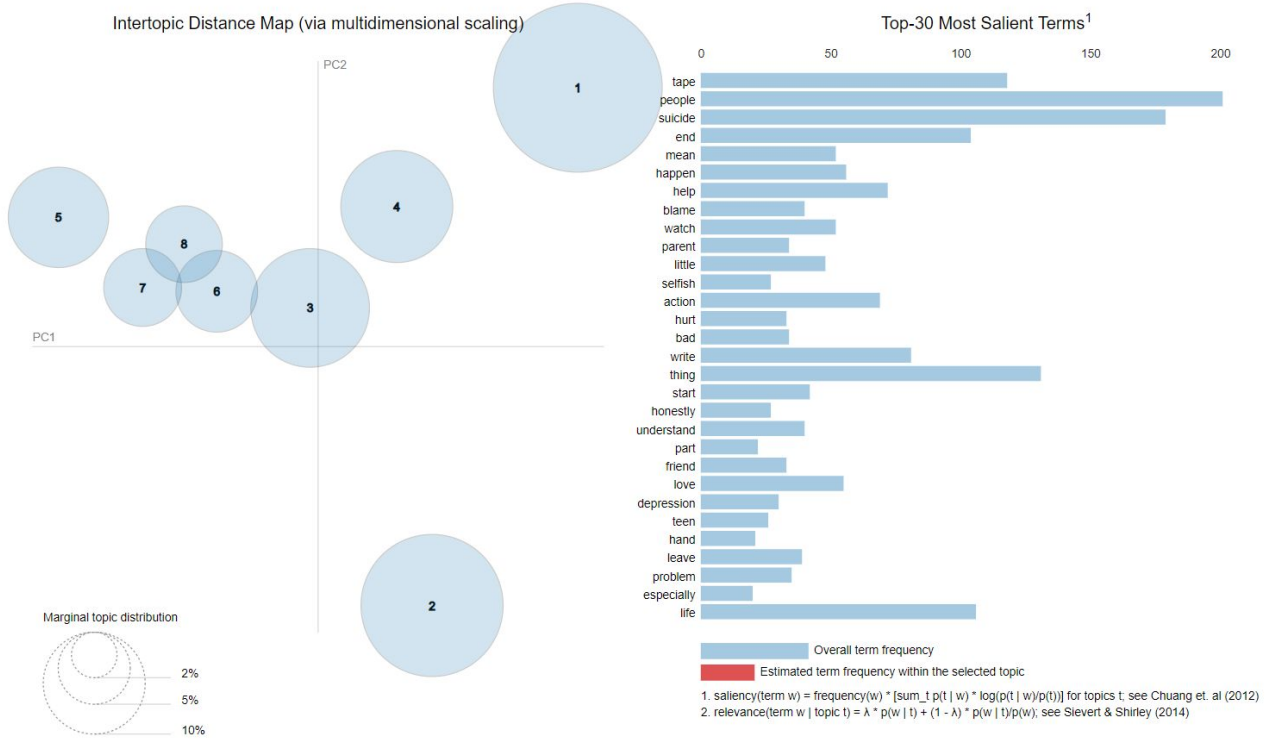
Cantidad de palabras y peso en el tópico



Cantidad de palabras y peso en el tópico



Visualizador pyLDAVis



Mejoras a futuro

Topics <> TW

Automatizar la relación de TW con Topics

Modelo Propio

Entrenar una red neuronal propia

Análisis

Conclusiones para mejorar el preprocesamiento

Producto

Ofrecer acceso a la herramienta a través de una API o WebApp

¡Muchas gracias!

Integrantes del grupo:
Luciana Diaz Kralj, Roberto José Catalán, Federico Kaplun