Procesamiento del Lenguaje Natural

Rodrigo S. Cortez Madrigal

TD-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) es una técnica utilizada en el procesamiento del lenguaje natural (NLP) y la recuperación de información para evaluar la importancia de una palabra en un documento dentro de un conjunto de documentos (corpus). Combina dos métricas principales:

TF (Term Frequency)

Mide la frecuencia de una palabra en un documento. Se calcula como el número de veces que una palabra aparece en un documento dividido por el número total de palabras en ese documento. Representa qué tan relevante es una palabra en un documento específico.

Fórmula: TF(t) = (Número de veces que aparece el término t en el documento) / (Número total de términos en el documento)

IDF (Inverse Document Frequency)

Mide la rareza de una palabra en el corpus. Si una palabra aparece en muchos documentos, su IDF será bajo, ya que no aporta mucha información. Si aparece en pocos documentos, su IDF será alto.

Fórmula: IDF(t) = log_e(Total de documentos / Número de documentos que contienen el término t)

TF-IDF

Es el producto de TF e IDF. Da un peso mayor a las palabras que son frecuentes en un documento pero raras en el resto del corpus, ayudando a identificar términos relevantes.

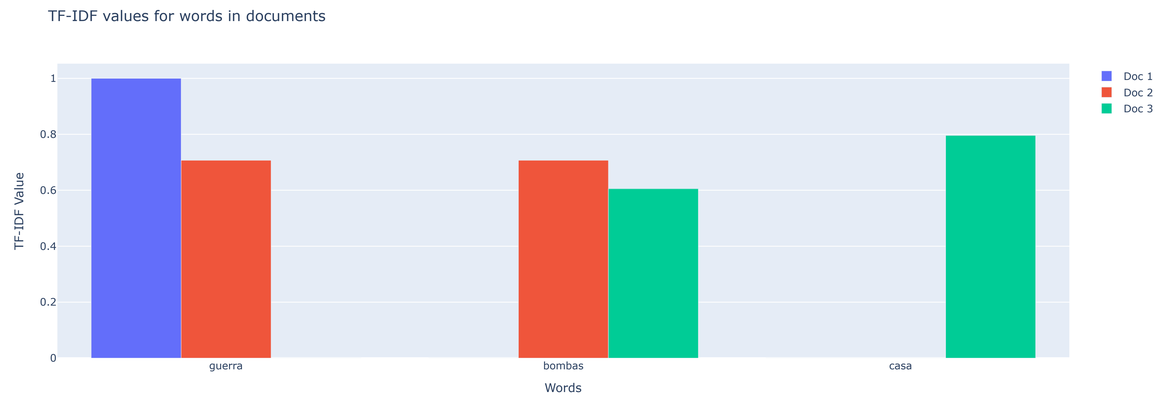
Fórmula: TF-IDF(t) = TF(t) * IDF(t)

https://www.youtube.com/watch?v=OkSZZ0F7ToA

In [8]:
```
import numpy as np
import pandas as pd
import re
import spacy
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB

import plotly
from plotly import graph_objs as go
from plotly import express as px
from plotly.subplots import make_subplots

import spacy

plotly.offline.init_notebook_mode(connected=True)
```

In [9]:
```
docs = ["No sé con qué armas se peleará la tercera guerra mundial, pero la cuarta se peleará con palos y piedras",
"El fin de la segunda guerra mundial llegó con las bombas atómicas lanzadas en Japón.",
"La casa se está incendiando porque le cayeron bombas."]

words = ["guerra", "bombas", "casa"]
```

In [10]:
```
# Para cada documento calcular el valor de tfidf de "guerra", "bombas" y "casa".

nlp = spacy.load("es_core_news_sm")

def preprocess_docs(docs, nlp):

    processed_docs = []
    for doc in docs:
        spacy_doc = nlp(doc)
        tokens = [token.text for token in spacy_doc if not token.is_stop and not token.is_punct]
        processed_docs.append(" ".join(tokens))
    return processed_docs

processed_docs = preprocess_docs(docs, nlp)

def get_tfidf(docs, words):
    vectorizer = TfidfVectorizer(vocabulary=words)
    X = vectorizer.fit_transform(docs)
    df = pd.DataFrame(X.toarray(), columns=vectorizer.get_feature_names_out())
    return df

# Calcular TF-IDF con los documentos procesados
df = get_tfidf(processed_docs, words)

fig = make_subplots(rows=1, cols=1)
for i in range(len(df)):
    fig.add_trace(go.Bar(x=df.columns, y=df.iloc[i], name=f'Doc {i+1}'))
fig.update_layout(title='TF-IDF values for words in documents', xaxis_title='Words', yaxis_title='TF-IDF Value')
fig.show()
```

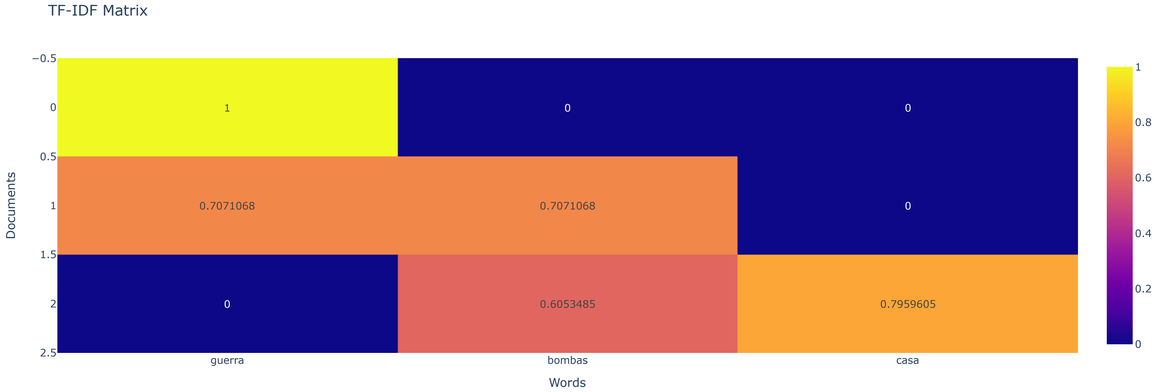


In [11]:
```
df
```

Out[11]:

	guerra	bombas	casa
0	1.000000	0.000000	0.000000
1	0.707107	0.707107	0.000000
2	0.000000	0.605349	0.795961

In [12]:
```
# Plot df matrix

fig = px.imshow(df, text_auto=True, aspect="auto", title="TF-IDF Matrix")
fig.update_xaxes(title_text="Words")
fig.update_yaxes(title_text="Documents")
fig.show()
```



In [ ]: