Tarea 2 - Investigación de TF-IDF y Laplace Smoothing

1. Investigue la estrategia de vectorización TF-IDF, ¿Cómo se calcula? ¿En qué situaciones es más efectivo usar TF-IDF para tareas de clasificación de texto? ¿Con qué bibliotecas se puede implementar?

TF-IDF vectorizer es una medida de feature weighting que expresa lo relevante que es una palabra en un documento, el cual forma parte de un corpus.

La estrategia tiene en cuenta el número de veces que aparece la palabra (o token) en dicho documento, pero también el total de veces que aparece en el corpus.

- Los tokens muy frecuentes a nivel de documento y de corpus (posibles stopwords) obtendrán un valor de TF-IDF Vectorizer bajo.
- Los tokens que aparecen solo en ciertos documentos del corpus tendrán un IDF mayor que aquellos que aparecen en mayor número de documentos.

De modo que:

$$W_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF → término x sin documento y.

$tf_{x,y}$ → frecuencia de x en y

df_x → Número de documentos que contienen x.

N → Número total de documentos.

Juan Pablo Bernal Lafarga A01742342

El TF-IDF resulta ser efectivo en casos donde buscas distinguir la importancia relativa de las palabras dentro de un corpus más grande y evitar que los términos más comunes dominen la representación de los documentos.

El algoritmo viene incluido en la librería de Scikit Learn como:

```
from sklearn.feature_extraction.text import TfidTransformer
```

2. ¿Qué problema de los N-gram resuelve el "Laplace Smoothing"? ¿Cómo trabaja? ¿y qué pasa con un modelo de NLP cuando se emplea esta técnica?

El Laplace Smoothing es una técnica estadística que resuelve el problema de la probabilidad cero en los modelos estadísticos añadiendo una pequeña constante a cada recuento en una distribución de frecuencia.

Para un modelo de N-gram sin suavizado, la probabilidad condicional de una palabra dada una secuencia previa se calcula como:

$$P(w_i | w_{i-1}, w_{i-2}, \ldots, w_{i-(n-1)}) = \frac{Cuenta(w_{i-(n-1)}, \ldots, w_i)}{Cuenta(w_{i-(n-1)}, \ldots, w_{i-1})}$$

Donde:

- w_i es la palabra que se predice
- $w_{i-(n-1)}, \ldots, w_{i-1}$ es la secuencia previa de palabras.

Con el Laplace Smoothing cambia a:

$$P(w_i | w_{i-1}, w_{i-2}, \ldots, w_{i-(n-1)}) = \frac{Cuenta(w_{i-(n-1)}, \ldots, w_i) + 1}{Cuenta(w_{i-(n-1)}, \ldots, w_{i-1}) + V}$$

Donde V es el tamaño del vocabulario, y se agrega 1 a cada cuenta para evitar probabilidades de cero.

Juan Pablo Berral Lafarga

AD1742310

El empleo del Laplace Smoothing en modelos de procesamiento de lenguaje natural mejora la generalización del modelo, pero reduce la precisión en N-gramas comunes.
Es útil en situaciones donde el corpus de entrenamiento es limitado y se espera que haya muchas secuencias de palabras nuevas en los datos de prueba.

3. ¿Qué pasa cuando una palabra en el test set no se encuentra en el vocabulario del modelo de los N-gram? ¿Cómo se puede modelar la probabilidad de palabras out-of-vocabulary? (OOV)

Cuando una palabra en el conjunto de prueba no está en el vocabulario del modelo de N-gramas, el modelo enfrenta un problema de probabilidad cero para cualquier N-grama que contenga esa palabra, lo que termina por atrofiar el rendimiento del modelo.
No obstante, existen técnicas que lo hacen frente a las palabras out-of-vocabulary tales como el Laplace Smoothing que evita la probabilidad cero, el Good-Turing Smoothing que estima la probabilidad de las palabras OOV utilizando la información sobre las palabras que aparecieron una sola vez en el conjunto de entrenamiento, los Word Embeddings que pueden proporcionar un vector para la palabra OOV basándose en su similitud con otras palabras conocidas, o los Transformers que utilizan embeddings para capturar el contexto de la palabra dentro de una oración.

Juan Pablo Bernal Lafarga A01742342