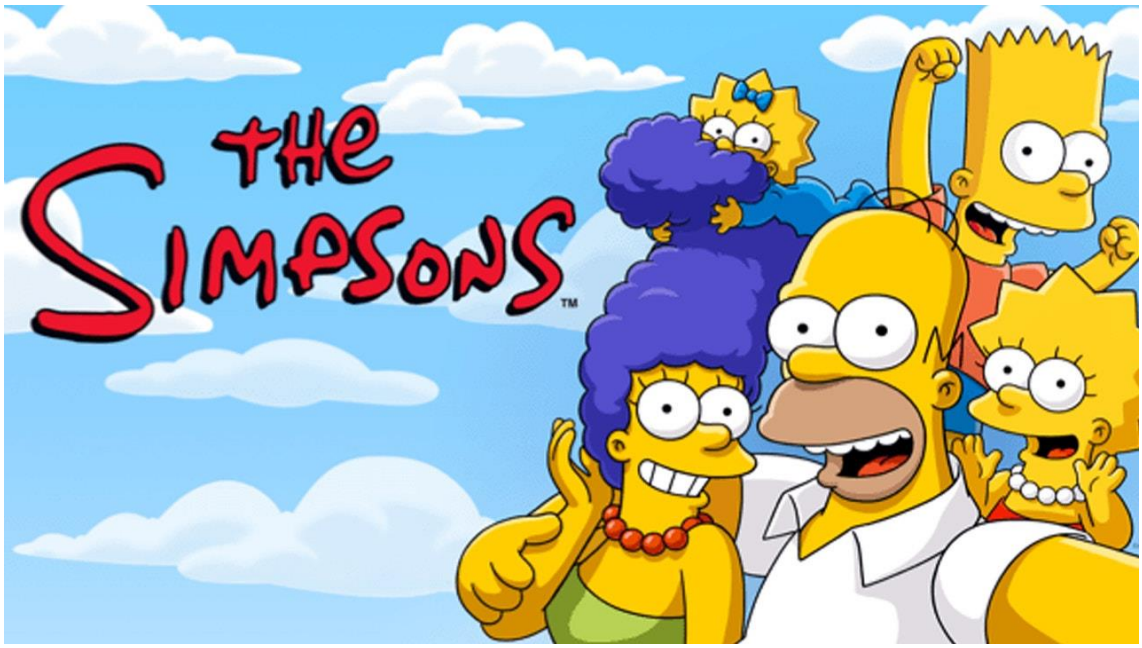
Ejercicio de Minería de Texto con NLP

Introducción

El ejercicio intentara usar muchas de las técnicas que hemos aprendido, haciendo minería de texto desde un dataset de conversaciones extraídas de la serie de televisión Los Simpsons, que se puede obtener en el siguiente enlace
https://drive.google.com/file/d/1j5DhcKjN5IrHnpcl8RG0LQD24_fCvAPB/view?usp=sharing



El contenido del fichero csv, básicamente son dos columnas, el personaje de los Simpsons que pronuncia la frase, y la frase en cuestión.

	character	text
0	raw_character_text	spoken_words
1	Miss Hoover	No, actually, it was a little of both. Sometim...
2	Lisa Simpson	Where's Mr. Bergstrom?
3	Miss Hoover	I don't know. Although I'd sure like to talk t...
4	Lisa Simpson	That life is worth living.

El objetivo final, será dada una frase, intentar predecir quien la dijo.

En este caso, no tiene demasiada importancia la calidad de la predicción, sino implementar los pasos necesarios para obtenerla.

Como punto de partida, se facilita el notebook **Ejercicio mineria de Texto Incompleto.ipynb**, en el siguiente enlace: https://colab.research.google.com/drive/104HqRfuW7G9vsK-WVeXSBLYy6uf3PrzI?usp=sharing

También se recomienda tener en cuenta dos notebooks explicados en clase:

- Word Embedding, en su parte final , trata con el mismo dataset, y puede ser de ayuda: https://colab.research.google.com/drive/1vIXaRSDYHasIL7G7633rMQ072wSznKv5?usp=sharing
- Análisis de texto sobre la biblia: hace una clasificación con un modelo Bayesiano, que puede ser muy útil, para implementar la clasificación que se pide en el ejercicio: https://colab.research.google.com/drive/1re0ocFvpMGERpv_9GRuaPMT3gLusKvEC?usp=sharing

Pasos necesarios

Lectura del Dataset

En este caso, siendo un csv, se recomienda usar la libraría Pandas, que facilita mucho el proceso de lectura desde un fichero, con la función **read_csv**

Exploración y limpieza de datos

En este dataset hay valores nulos, y es conveniente eliminarlos

También hay muchos personajes que tienen muy poquitas frases, y por lo tanto, seria muy difícil, predecirlos. Por eso, se recomienda filtrar el dataset, a los personajes que digan al menos 10000 frases.

Flujo de proceso de diálogos

Para simplificar el problema conviene realizar algunas tareas previas.

Se recomienda crear funciones y aplicarlas secuencialmente

Sera necesario implementar las siguientes funciones, que realicen las siguientes tareas:

- **borrar_signos_puntuacion(texto):** Esta función, recibirá el texto en formato cadena de texto, y pasara el **texto a minúsculas**, y eliminara los **caracteres de puntuación** (‘,’,’.’,’?’ ,‘!’,’;’,’-’)… y retornara la misma cadena sin caracteres de puntuación
- **eliminar_numeros(texto):** Quitara los números de una cadena de texto y retornara la misma cadena sin números
- **eliminar_stop_words(texto):** Quitara los las **stop words** de una cadena de texto y retornara la misma cadena sin stop words
- **stemming(texto):** Recibirá una cadena de texto y por cada palabra realizara stemming y retornara la misma cadena tras aplicar Stemming

- **Limpiar_texto(texto):** Recibirá una cadena de texto, y aplicara todas las transformaciones anteriores, y retornara una nueva cadena de texto transformada.

Crear el vector de palabras para entrenamiento

Ahora intentaremos obtener una matriz donde el eje x, serán las filas del dataframe, es decir las frases, y el eje y, cada una de las palabras del diccionario, con la frecuencia en la que aparecen en cada frase.

	aa	aaa	aaahhh	aah	aannnnd	aappl	aargh	aaron	aaronson	aarp	...	zombi	zone	zoo	zoot	zorrinid	zu	zuckerberg	zuh	zum	zzebra
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 12336 columns

Para ello se recomienda usar CountVectorizer de sklearn

```
from sklearn.feature_extraction.text import CountVectorizer
```

Esta matriz se almacenara en la **variable X**

En la **variable y,** almacenaremos los datos a predecir, es decir el personaje que pronuncia la frase, que podemos obtener del dataframe.

Entrenamiento del modelo Bayesiano

Aquí tenemos que entrenar un modelo bayesiano para saber cual es el personaje a la que pertenece cada frase.

Se recomienda seguir la implementación de el Análisis de texto sobre la biblia: https://colab.research.google.com/drive/1re0ocFvpMGERpv_9GRuaPMT3gLusKvEC?usp=sharing

Básicamente los pasos a seguir con:

- Separar los conjuntos de entrenamiento y test usando la función **train_test_split** de **sklearn.model_selection**
- Entrenar un modelo bayesiano a partir de los datos en entrenamiento: Se recomienda usar la clase **MultinomialNB** del paquete **sklearn.naive_bayes**