Descripción detallada

Consiste en el estudio, diseño y desarrollo de un sistema de detección de ofensividad que reciba como entrada un texto y que devuelva como salida la clase que el sistema haya obtenido de acuerdo a una escala basada en dos valores: ofensivo y no ofensivo.

Para probar la validez del sistema se utilizará el corpus OffendES, que está formado por comentarios escritos en español extraídos de tres redes sociales: Twitter, Youtube e Instagram.

Este corpus se ha dividido en dos conjuntos de datos: train (para entrenar el sistema) y test (para evaluar el sistema).

A la hora de realizar el clasificador de comentarios en cuanto a lenguaje ofensivo, lo he hecho mediante aprendizaje supervisado.

Para ello he entrenado el algoritmo con los comentarios como documentos a clasificar y la columna de label como los tags (etiquetas) del fichero train.tsv., y usando este modelo, he clasificado los comentarios del fichero test.tsv.

El resultado de esta clasificación se ve reflejada en el fichero prediction_test.tsv.

Explicación del código

1.- Librerías

En este espacio de código se hallan todas las librerías usadas durante el código:

- `from google.colab import drive`

Para conectarnos a nuestra cuenta de Drive

- `import os`

Para poder situarnos en un directorio concreto donde trabajar

- `import pandas as pd`:

Para poder leer los ficheros .tsv

- `from nltk import WordPunctTokenizer`:

Para tokenizar los comentarios

- `from nltk.corpus import stopwords`

Para poder eliminar las palabras vacías o stopwords

- `from nltk.stem.snowball import SnowballStemmer`

Para poder reducir las palabras a su raíz

- `from sklearn import svm`

Para crear más tarde el modelo para entrenar el algoritmo

- `from sklearn.feature_extraction.text import TfidfVectorizer`

Para extraer las características usando TF-IDF

2.- Funciones

Este apartado lo he dividido en las distintas funciones que uso a lo largo del código principal:

- **Función para leer ficheros .tsv**

Este método se usa para leer tanto el fichero train.tsv como test.tsv, dado que ambos tienen el mismo formato.

Primero usamos el método read_csv() de la librería pandas para poder abrir los fichero mencionados. A este método le pasamos como parámetros: el nombre del fichero (facilitado por la cabecera de la función); el delimeter, que es el carácter por el cual se va a guiar el método para separar las comunas, en este caso '\t' (usado para indicar la tabulación); y un encoding para poder leer la codificación del texto.

Para acceder a cada columna usamos df.iloc[:, c] (sustituyendo 'c' por el número de la columna) y luego accediendo al numero de fila con el numero de la fila entre corchetes quedando así "df.iloc[:, 0][5]" para acceder a la primera columna y sexta fila

A la hora de leer lo comentarios he tenido el problema de que por más encoding diferentes que usase (utf-8, Unicode_escape, utf-16, latin1, ...), no llegaba a a leer correctamente todos los caracteres como los acentos y la ñ. Para solucionar esto, lo he hecho una función específicamente para esto llamada correcciónCodificacin().

El contenido de todas las columnas lo amaceno en un diccionario de listas, siendo cada columna una lista.

- **Función para procesar el texto**

Función que uso para procesar los comentarios de los ficheros leídos anteriormente:

1º Tokenizamos el texto para separarlo en palabras separadas, usando WordPunctTokenizer por la separación que hace de la puntuación

2º Pasamos todos los tokens a minúsculas

3º Filtramos los tokens quitando las palabras vacías, ya que son palabras que no nos interesan

4º Reducimos las palabras a su raíz o lexema

5º Por último devolvemos los tokens resultantes

- **Función para corregir la mala lectura de los ficheros**

Función comentada anteriormente usada en la lectura de los ficheros .tsv para corregir los fallos de codificación.

Esta función se basa en reemplazar los caracteres extraños por su letra real, por ejemplo sustituir 'Ãi' por 'á' entre otras.

Recibe el comentario como cadena de texto y devuelve otra cadena corregida.

3.- Código principal

Este apartado es donde se desarrolla el código principal de la práctica. Para ello lo he dividido también en varias partes:

- **Lectura y procesamiento del fichero de entrenamiento**

Esta parte es donde se lee el fichero que usamos para el entrenamiento del aprendizaje supervisado, el fichero train.tsv. También procesamos los comentarios obtenidos de la lectura. Toda la información de este fichero se guarda en el diccionario llamado 'doc'

- **Lectura y procesamiento del fichero de evaluación**

Tiene el mismo propósito que el apartado anterior solo que en este leemos el fichero test.tsv, fichero para evaluar el algoritmo. Toda la información de este fichero es guardada en el diccionario 'fichero_test'

- **Entrenamiento del algoritmo**

En esta parte del código creamos el modelo y lo entrenamos usando los comentarios del primer fichero leído y su columna label como tags o etiquetas.

- **Creación del fichero de salida**

Creamos un fichero tsv llamado 'prediction_test.tst' donde irán almacenado el resultado de la predicción del algoritmo en formato de dos columnas, la primera el id del comentario y en la segunda con OFF si resulta ser un comentario ofensivo y NON si no lo es.

- ### Predicción y anotación de resultados

En este punto es cuando recorremos los comentarios del fichero de evaluación para predecir su ofensividad. Este resultado es anotado a medida que realizamos el recorrido.

- ### Evaluación del sistema

Por último, ejecutamos el script ‘score.py’ para evaluar el resultado obtenido por el algoritmo de aprendizaje supervisado desarrollado durante la práctica. Este script crea el fichero ‘resuts.txt’ donde se almacena el resultado de esta evaluación [adjunto con los demás ficheros].