Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Inteligencia Artificial Avanzada

Proyecto

Daniel Hernández de León
(alu0101331720@ull.edu.es)

1. Preprocesamiento.

Para el preprocesamiento los mejores valores utilizados han sido:

- No quitar números.
- No quitar palabras de más de 20 carácteres.
- Poner todo en minúsculas.
- Quitar signos de puntuación.
- No quitar stopwords.
- No quitar los emojis.
- Quitar url html y hashtags.
- Utilizar correción ortográfica.
- Utilizar truncamiento pero no lematización.

Para la verificación de estos parámetros dividí el conjunto de entrenamiento en dos conjuntos aleatorios, entrené el modelo con uno y validé con el otro.
Están todas las pruebas realizadas en un fichero *report.md*.
Probar a cambiar el valor de **k** (k es el valor el cuál tomamos un token como unknown según su frecuencia) empeora el resultado, en menor a 2 es lo mejor posible y cambiar el suavizado laplaciano también.

2. Librerías utilizadas.

Utilizo NLTK para el truncamiento y la lematización con PorterStemmer y WordNetLemmatizer. https://www.nltk.org/
SymspellPy para la corrección ortográfica. https://pypi.org/project/symspellpy/
Pandas para la lectura de los ficheros excel. https://pandas.pydata.org/
Y emoji para poder pasar a palabras los emojis. https://pypi.org/project/emoji/

3. Implementación.

Se han creados 3 ficheros en python, uno llamado *src/vocabulary/vocabulary.py* para la creación del vocabulario con una clase Vocabulary que tokeniza todo el input y genera un fichero vocabulario.txt y un json de configuración de los parámetros de preprocesado utilizados.
Otro llamado *src/language_model.py* para la creación de los modelos de lenguajes positivos y negativos que utiliza el vocabulario generado anteriormente y los parámetros con las probabilidades logarítmicas de cada token y su frecuencia.

Por último un clasificador llamado *src/clasificator.py* que dado un input de testeo lo preprocesa y luego lo clasifica como positivo o negativo según las probabilidades de sus tokens y de la clase positivo o negativo.

4. Error.

He probado con bastantes opciones de configuración y el mejor porcentaje de acierto que he logrado ha sido de 52.34% sobre un conjunto de entrenamiento y validación independientes.