MEMORIA DETOXIS IberLEF 2021d

Nombres: Álvaro Mazcuñán Herreros - Miquel Marín Colomé
Nombre del equipo registrado: Dembo

En nuestro caso, se decidió elegir el concurso de **DETOXIS** con el fin de entrenar modelos de Machine Learning aprendidos durante estos cursos y, de esta forma, poder analizar los diferentes tweets tóxicos que se encontraban en dicha base de datos.

En esta tarea se disponía de dos conjuntos de datos. El primero de ellos era el de train, con un total de 3958 tweets. Por otra parte, para poder validar la calidad del modelo, se disponía del conjunto de test el cual estaba compuesto de 891 tweets.

En conjunto, este dataset disponía de una gran variedad de variables como por ejemplo:

- Constructiveness
- Positive / Negative stance
- Stereotype
- Sarcasm
- Aggresiveness

Sin embargo, para esta competición solamente se utilizó la variable **comment**, es decir, aquella variable donde figuraban los diferentes tweets de varios periódicos españoles como ABC, elDiario.es, El Mundo, etc.

El objetivo de este concurso era doble. Por una parte se requería etiquetar los tweets del conjunto de test con 0 o 1, es decir, si dichos tweets eran no tóxicos o tóxicos, respectivamente. Para ello, en el conjunto de entrenamiento, y de esta forma poder evaluar los modelos de Machine Learning correspondientes, se disponía de la variable **toxicity**.

Además, en la segunda subtarea, el objetivo se complicaba un poco más, debido a que en este caso se pedía etiquetar el mismo conjunto de test pero, en este caso, añadiendo distintos niveles de toxicidad:

- 0 → Not toxic
- 1 → Mildly toxic
- 2 → Toxic
- 3 → Very toxic

Tal y como se ha comentado anteriormente para la primera subtarea, con tal de evaluar nuestros modelos, en el conjunto de los 3958 tweets, se disponía de la variable **toxicity_level**.

Antes de realizar ninguna separación en los datos con tal de poder entrenar los modelos correspondientes, se decidió llevar a cabo una pequeña **limpieza/preprocesamiento** de los datos.

Para ello, lo primero que se hizo fue eliminar de los mensajes, aquellos caracteres que eran emojis, hashtags (#), URLs y otros especiales.

Eliminadas dichas partes de los tweets, se empezó **tokenizando** el texto, es decir, separar el tweet en palabras y, con esto, obtener una lista con dichas palabras.

Con esta lista de palabras, el siguiente paso era la eliminación de las **stopwords**, mejor dicho, aquellas palabras que de por sí no tienen significado. Este grupo de palabras suele estar formado por artículos, pronombres, preposiciones, adverbios y algunos verbos en especial.

Lo siguiente sería aplicar los algoritmos de **stemming** y **lemmatization**. El primero funciona cortando el final o el comienzo de la palabra, teniendo en cuenta una lista de prefijos y sufijos comunes que se pueden encontrar en una palabra flexionada. Hay que decir que este enfoque puede tener éxito en algunos ocasiones pero no siempre.

A continuación se muestra un ejemplo de cómo funciona el algoritmo de stemming:

studies → -es → studi
studying → -ing → study

Por otra parte, el algoritmo de lemmatization tiene en cuenta el análisis morfológico de las palabras. Para hacerlo es necesario tener diccionarios detallados en los que el algoritmo puede consultar para vincular el formulario con su lema. Se pasa a mostrar un ejemplo como en el caso anterior:

studies → third person, present tense of the verb study → study
studying → gerund of the verb study → study

Una vez realizada toda la parte de preprocesamiento de los tweets, se puede pasar ya a la parte de **representación de textos** que se ha utilizado para este concurso.

No obstante, antes de entrar con esto, se tiene que realizar una partición de entrenamiento y test. Concretamente, en el dataset de entrenamiento (el que contiene 3958 mensajes) se hará una partición de entrenamiento y validación (alrededor del 10-20% dependiendo del algoritmo que se utilice).

Realizado esto, con los 891 tweets restantes del otro conjunto, se podrán etiquetar los mensajes deseados.

Además de realizar las particiones correspondientes, se tiene que observar si las clases están balanceadas o no, ya que esto puede conllevar problemas a la hora de evaluar los modelos posteriores.

En la variable que contiene clasificación binaria, **toxicity,** se pueden observar 2316 tweets con la clase 0, es decir, no tóxicos y el resto (1147) con la clase 1, queriendo decir que estos sí que son tóxicos.

Según el criterio que consideró el equipo, se decidió no llevar a cabo ninguna tarea de balanceo de clases.

Sin embargo, la situación varía en la variable **toxicity_level**. Dicha variable contiene 2317 tweets con la clase 0, 808 con la clase 1, 269 con la clase 2 y, finalmente, 69 con la clase más tóxica. En este caso sí que se decidió llevar a cabo una tarea de balanceo (se explicará brevemente la solución adoptada en la parte de la evaluación de modelos).

Una vez considerado el tema del balanceo de clases, se pasan ya a explicar las técnicas de representación de textos utilizadas.

La primera de ellas se trata de la **bolsa de palabras**. Es una manera de representar el vocabulario que utilizaremos en nuestros modelos y consiste en crear una matriz en la que cada columna es un token y se contabilizará la cantidad de veces que aparece ese token en cada oración.

A continuación se muestra un pequeño ejemplo de cómo funciona dicha técnica:

Se dispone de 3 oraciones que son las siguientes:

- *This movie is very scary and long*
- *This movie is not scary and is slow*
- *This movie is spooky and good*

Con estas 3 frases se obtiene un vocabulario con todas las palabras únicas que es el siguiente: "This", "movie", "is", "very", "scary", "and", "long", "not", "slow", "spooky", "good".

Aplicando bag-of-words nos quedaría la siguiente matriz:

	1 This	2 movie	3 is	4 very	5 scary	6 and	7 long	8 not	9 slow	10 spooky	11 good	Length of the review(in words)
Review 1	1	1	1	1	1	1	1	0	0	0	0	7
Review 2	1	1	2	0	0	1	1	0	1	0	0	8
Review 3	1	1	1	0	0	0	1	0	0	1	1	6

El problema de esta técnica anterior es que solamente considera unigramas. Por dicho motivo, también se utilizó la técnica de los **n-gramas** ya que, de esta forma, se podría considerar orden de las palabras. Se decidió utilizar entre 2 y 3-gramas (bigramas-trigramas). El procedimiento sería el mismo que antes pero teniendo en cuenta este último enfoque.

La tercera y última técnica de representación de textos que se utilizó fue la de **Term-Frequency - Inverse Document Frequency (TF-IDF)**. Dicha técnica consiste en medir cuánto de importante es una palabra dentro de un texto, es decir, cada palabra va a tener un peso asociado, dependiendo de su importancia.

Explicadas las técnicas de representación de los tweets, se pasa ya a mencionar los distintos **modelos** de Machine Learning que se utilizaron:

- SVC
- Decision Tree
- Logistic Regression
- MLP (Multilayer-Perceptron)
- Random Forest
- Stacking
- BETO

Hay que decir que el modelo de Stacking es una combinación de algunos de los modelos anteriores, concretamente utilizamos como modelos base SVC, Decision Tree, Random Forest y MLP y, como modelo metalearner, la regresión logística.

Además, en algunos de los métodos anteriores, con tal de poder obtener los parámetros ideales para cada uno de ellos, se utilizó **fine tuning**, en concreto, la técnica de **Grid search**.

Tal y como se ha comentado previamente, el tema del desbalanceo es un problema a tener en cuenta a la hora de obtener las etiquetas para los distintos tipos de toxicidad (**toxicity_level**).

Por lo tanto, para todos los modelos anteriores, exceptuando el modelo BETO, se utilizó, en el momento de entrenar dichos modelos, un parámetro extra llamado *stratify* que permitía balancear las clases mientras se realizaba dicha tarea de entrenamiento.

Finalmente, también se utilizó la técnica de BERT, en concreto el modelo dccuchile/bert-base-spanish-wwm-uncased de Hugging Face para tweets en castellano.

BERT utiliza Transformers, un mecanismo de atención que aprende las relaciones contextuales entre palabras en un texto. En su forma básica, Transformer incluye dos mecanismos separados: un codificar que lee la entrada de texto y un decodificador que produce una predicción de la tarea. Dado que el objetivo de BERT es generar un modelo de lenguaje, solo es necesario el mecanismo del codificador.

Debido a que esta técnica era muy novedosa para todos nosotros, requirió más tiempo con tal de investigar cómo funcionaba y, de alguna forma, poder implementarlo en Python con la ayuda de gente experta que ya había tratado el tema con anterioridad.

Una vez se ha hecho una breve descripción de los modelos que ha utilizado este equipo, se pasa a la parte de los resultados.

Sin embargo, antes se tiene que comentar que para poder evaluar la calidad de dichos modelos, se han utilizado dos medidas de evaluación:

- F1-Score
- Accuracy

Para la competición de DETOXIS, debido a que solo se podían enviar un máximo de 5 runs, se decidieron enviar los modelos de Stacking y BETO (aunque en este caso los resultados no fueron del todo buenos pero se decidió enviarlo igualmente ya que había sido una nueva técnica que nunca habíamos aprendido y queríamos probar)

Los resultados para el modelo BETO fueron los siguientes:

- Toxicity:
    - Accuracy: 0.7839
    - Loss: 0.9899
- Toxicity Level:
    - Accuracy: 0.265
    - Loss: 1.406

Por otra parte, los resultados para el modelo híbrido de Stacking fueron estos:

- Toxicity:
    - Accuracy: 0.7716
    - F1-Score: 0.7568
- Toxicity Level:
    - Accuracy: 0.6949
    - F1-Score: 0.6347

A lo largo de este proyecto se han podido adoptar distintos enfoques con tal de poder obtener los mejores resultados posibles a la hora de etiquetar los tweets con sus correspondientes valores de toxicidad. Sin embargo, por falta de tiempo, no se han podido aplicar algunas mejoras que se tenían en mente.

Por lo tanto, sabiendo esto, el legado / posibles mejores de este proyecto es el siguiente:

- Debido que en el caso de BETO el accuracy deja mucho que desear, lo que se podría realizar es lo siguiente: debido a que dicha clase está muy desbalanceada, alrededor de 2000 muestras y, en este caso, se quiere predecir el nivel de toxicidad de un tweet según si el modelo previamente lo ha predecido como si ese tweet es tóxico o no. Por lo tanto, primero se podría hacer una predicción sobre la variable **toxicity** y, una vez obtenidas las predicciones, se obtendría una nueva columna *new_predictions* y con ésta solamente se trabajará con aquellos tweets que el modelo haya predecido como tóxicos.
- Realizar tareas de balanceo antes de entrenar los modelos
- Utilizar más variables tales como *sarcasm, aggressiveness,* etc y no solamente quedarnos con la variables del comentario del tweet en sí.