En esta práctica, se plantea la construcción de un modelo de aprendizaje automático (con RNA) que realice predicciones sobre qué Tweets tratan de desastres reales y cuáles no (ejemplo de detección de noticias *fake*).

https://www.kaggle.com/datasets/vstepanenko/disaster-tweets

1.- Carga el conjunto de datos
A veces en Kaggel no etiquetan el subconjunto de datos de test para que ellos puedan evaluarlo después. En este caso es así, nuestro subconjunto de test está sin etiquetar.

2.- Visualiza el conjunto de datos (análisis exploratorio)

Siempre es conveniente realizar un análisis exploratorio de la distribución de los datos para determinar la mejor manera de resolver el problema. Por ejemplo, visualiza el tamaño de los subconjuntos y observa cuántos tenemos de cada tipo en el caso de train, y represéntalo mediante un histograma. En concreto, deberás explorar el campo text (número de palabras por tweet, distribución en una gráfica para los etiquetados como desastre y los que no, etcétera), También puedes extraer el número de palabras únicas (en vez de totales como hicimos antes).

Más cosas interesantes que puedes extraer: longitud media de las palabras por Tweet, número de caracteres por Tweet
Otras posibles: Número de urls, hashtags o '@ 'por Tweet, etc.

Obtén las stopwords más utilizadas (*from nltk.corpus import stopwords*)

Estas palabras no tienen un significado por sí solas, sino que modifican o acompañan a otras, este grupo suele estar conformado por artículos, pronombres, preposiciones, adverbios e incluso algunos verbos. En el procesamiento de datos en lenguaje natural son filtradas antes o después del proceso en sí, no se consideran por su nulo significado, en el caso de los buscadores como Google no lo consideran al momento de posicionar, pero sí en el momento de mostrar los resultados de búsqueda.

Compara en gráficas las stopwords en los tweets etiquetados como desastres y como no.

Análisis de signos de puntuación (- | : ? + @ / = %, …)

Al igual que con las stopwords, haz lo mismo con los signos de puntuación.

Análisis de Ngramas (from slearn.feature_extraction.text import CountVectorizer)

Fundamentalmente, es un análisis que consiste en identificar qué palabras aparecen juntas entre ellas. Por ejemplo, si N=2, voy a hacer un análisis de frecuencia de aquellas dos palabras que aparecen siempre juntas.

3.- Limpieza del conjunto de datos

En base a lo anterior, elimina lo que consideres del texto que pudiera ser ruido para que nuestra red neuronal funcione mejor. Por ejemplo, puedes eliminar urls, texto html, stopwords, signos de puntuación, …

Puedes ayudarte con *re* que sirve para generar expresiones regulares (*import re*), y para los tags html puedes usar HTMLParser (from html.parser import HTMLParser).

También puedes eliminar emoticonos (no son otra cosa que una serie de caracteres).

Os recomiendo que uséis la función apply de pandas para que aplique cada una de vuestras funciones de limpieza sobre el campo text de los conjuntos de datos de train y test.

Comprueba de nuevo los Ngramas habiendo realizado ya la limpieza.

4.- Vectorización del conjunto de datos

Obtén las etiquetas y codifica el texto (vectorización basada en frecuencias), por ejemplo con (*from sklearn.feature_extraction.text import TfidfVectorizer)*

Nota: Para el conjunto de datos de test, usa solo .transform que es lo que usaremos para ejemplos nuevos. (el .fit_transform solo se aplica a train)

5.- División del conjunto de datos

Divide el conjunto de datos *train* en *train* y *val* (porque el conjunto de datos *test* dado no tiene etiquetas)

6.- Construye el modelo

Pruebas con distintas configuraciones de hiperparámtros, explicando por escrito el por qué de cada hiperparámetro, hasta obtener la óptima.

Obtén la gráfica de loss y val_loss, y explícala.

¿se produce overfitting? ¿cómo puedes reducirlo? Aplica alguna técnica de reducción de overfitting. Describe los resultados (Deberías obtener un accuracy y precision de, al menos, 0.85)

7.- Evalúa el modelo

Como no tenemos las etiquetas en *test*, ejecuta la predicción igualmente y representa el tweet y su predicción para poder tener una intuición visual. Descríbelo brevemente.