

# Informe Proyecto

## IAA PNL

Adrián Fleitas de la Rosa <alu0101024363>

## Preprocesamiento:

Para el preprocesamiento lo que ha dado mejores resultados ha sido eliminar emoticonos, eliminar enlaces, eliminar los números, los caracteres especiales y los dobles espacios y tabuladores(sustituídos por un solo espacio). Este proceso se hace por cada tweet luego se tokeniza y si no es una stop word se trunca. Mantener las mayúsculas en el vocabulario mejoraba ligeramente la precisión así que decidí no pasar todo a minúsculas. He probado partiendo de la lista propuesta de preprocesamiento todas las combinaciones excepto usar truncamiento y ninguna me ha mejorado la precisión por encima de la que uso actualmente.

```
def parse(line):
    parsed = ""
    # Borrar emojis
    aux = deEmojify(line)
    #emojis = demoji.findall(line)
    #print(emojis)
    # Borrar html tags
    aux2 = deletehtml(aux)
    parsed = parsed + aux2
    # Borrar urls
    finalElement = re.sub(r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[*]
    # Borrar numeros
    final = re.sub(r'\d+', '', finalElement)
    final = re.sub(r'^\w\s', '', final)
    final = final.replace('_', '')
    " ".join(final.split())
    return final
```

## Librerías utilizadas:

- nltk para la tokenización y las stop words.
- pandas para el manejo de archivos excel y dataframes
- re para el preprocesamiento (regular expressions)
- json para crear más fácilmente los diccionarios
- math para logaritmos y demás cálculos
- Algunas librerías para probar pre procesamientos distintos como demoji, string, etc

## Implementación

El proceso es el siguiente:

1. Usando la función printVocab(generateVocab()) del fichero vocabfromxlsx generamos el vocabulario global del corpus. Esta función usa generateTokens la cual genera los tokens de una línea parseado con la ayuda de la función parse. La misma usa a su vez métodos auxiliares para parsear un tweet. Además este archivo nos proporciona la opción de generar un diccionario de vocabulario para un archivo dado.

2. En `classprobabilities.py` llamamos a `separate` para separar el excel en dos excels según su categoría.
3. Seguidamente llamamos a `generateJsons` que nos crea un json por cada clase con sus palabras y su frecuencia.
4. Luego creamos los corpus teniendo en cuenta el suavizado y las palabras `unknow` ( $k = 3$  daba el mejor resultado) para ello usamos `generateCorpus`. Este método de nuevo nos crea un json pero esta vez teniendo en cuenta `unknow`,  $k$ , etc.
5. Llamando a `languageModel` generamos los ficheros pedidos `modelo_lenguaje_N.txt` y `modelo_lenguaje_P.txt`. Usando el `negative_corpus.json` y `positive_corpus.json` respectivamente.
6. Finalmente llamando a `printResult()` en `test.py` crearemos los ficheros finales `resumen_alu..` y `clasificacion_alu..`. Este método utiliza un generador de tokens por tweet basado en el explicado anteriormente, el método que cálculo la probabilidad del tweet dado (tras tokenizar) teniendo en cuenta los language dictionaries creados con la función `languageDicts`.
7. Para el caso particular del fichero de entrenamiento existe un método llamado `checkResult` el cuál nos dice la precisión de la clasificación para el fichero dado.  
\*Se ha usado ficheros json por la facilidad de exportar a ellos e importar de ellos como diccionarios.

## Acierto sobre train

Acierto sobre el conjunto de entrenamiento de un 59.4%.

```
[Running] python -u "d:\Vs proyectos\IAA-PLN\test.py"  
Negative aciertos: 7920 Positive aciertos: 11945  
Aciertos: 19865 Precision: 59.4
```

## Programas y ficheros

Programas:

- `vocabfromxlsx.py` generador vocabulario + función auxiliar para crear tokens
- `classprobabilities.py` generador de probabilidades de palabras para negative y positive. Usa la función auxiliar de `vocabfromxlsx.py`
- `test.py` clasifica el conjunto de entrenamiento (o el establecido como `df/corpus`). Usa el parser de `vocabfromxlsx.py`

Ficheros:

- El programa genera ficheros intermedios pero solo necesita que exista un `COV_train.xlsx`. (y en su defecto `COV_test...` para testing)
- Cada programa genera el fichero o ficheros pedidos en cada parte del proyecto con el formato y el nombre correspondiente.