**DOCUMENTACIÓN GITHUB HATEMEDIA**

**Importación y uso del modelo para realizar predicciones:**

Los diferentes modelos empleados tienen como fin analizar datos extraídos de diferentes redes sociales como por ejemplo Twitter o Facebook de la cual se extraen los comentarios referentes a los periódicos (La Vanguardia, El Mundo, ABC, El País y 20 Minutos) y páginas web de los mismos para su posterior clasificación en función de si estos contienen o no odio.

Para llevar a cabo esta clasificación, se han entrenado distintos modelos como *Gradient Boosting*, SVM (*Support Vector Machines*), RF (*Random Forest*) o CART (*Classification And Regression Tree*).

Para el uso de los diferentes modelos de predicción se adjuntan los scripts Python “ejemplo.py” y “obtencion\_caracteristicas.py” y la carpeta “recursos” en la que se encuentran las distintas bolsas de palabras utilizadas para la obtención de características en el preprocesado de los comentarios a predecir.

Antes de ejecutar estos scripts es necesario instalar las distintas librerías importadas en ellos.

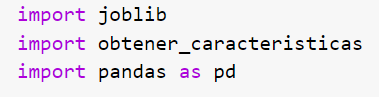
**OBTENCION\_CARACTERISTICAS.PY:**

Las funciones presentes en este script son necesarias para el preprocesado del corpus. Los distintos modelos necesitan las características extraídas mediante este script para realizar sus predicciones.

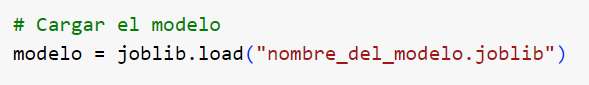
**EJEMPLO.PY:**

Se detalla a continuación mediante el código de este script el procedimiento a seguir para realizar las predicciones.

Importamos la librería joblib para poder cargar los distintos modelos, así como importamos el otro script adjunto y la librería pandas para trabajar con dataframes:

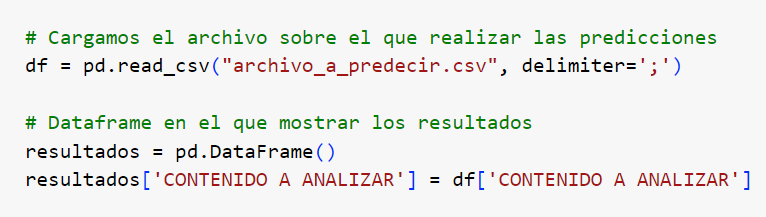


Cargamos el modelo mediante el uso de la librería joblib:

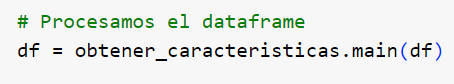


Leemos el archivo csv en el que están presentes los distintos comentarios sobre los que queremos realizar las predicciones. Para que el preprocesado se lleve a cabo correctamente necesitamos nombrar como “CONTENIDO A ANALIZAR” la columna que contiene los comentarios.

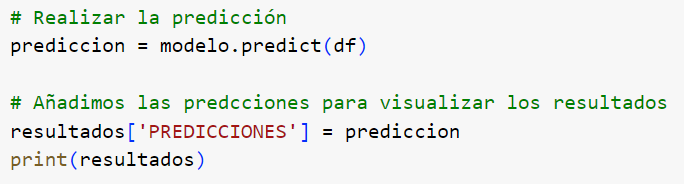
Se crea un dataframe que contendrá los comentarios, al que más tarde se añadirán las predicciones.



Se obtienen las características que utilizará el modelo para realizar la predicción de los comentarios:



Obtenemos las predicciones y mostramos los resultados:



**Uso del código de entrenamiento:**

Se adjunta el notebook de Google Colab “Modelo\_binario\_.ipynb”, siendo aquí donde encontramos el código a ejecutar para el entrenamiento de los modelos.

Colab es un producto de Google Research que permite a cualquier usuario escribir y ejecutar código arbitrario de Python en el navegador; es especialmente adecuado para tareas de aprendizaje automático, análisis de datos y educación. Con esta herramienta veremos el código en distintas celdas que podremos ejecutar de manera independiente.

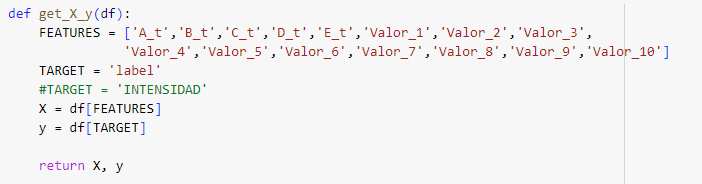
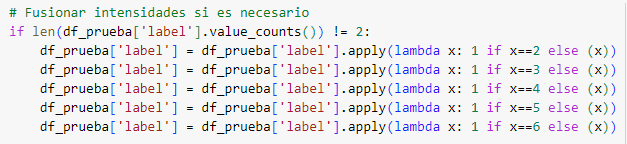
Al lanzar la siguiente celda se importan las distintas librerías necesarias para la ejecución de todo el código:



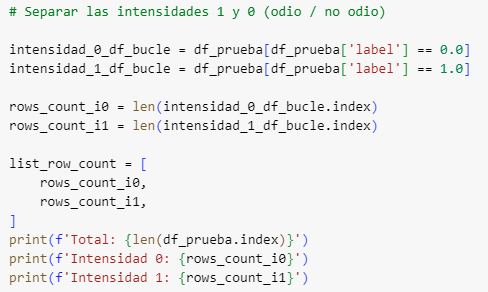
Si es la primera vez que ejecutamos esta celda, antes debemos instalar las distintas librerías en el entorno de ejecución que proporciona Google Colab, de la siguiente forma:

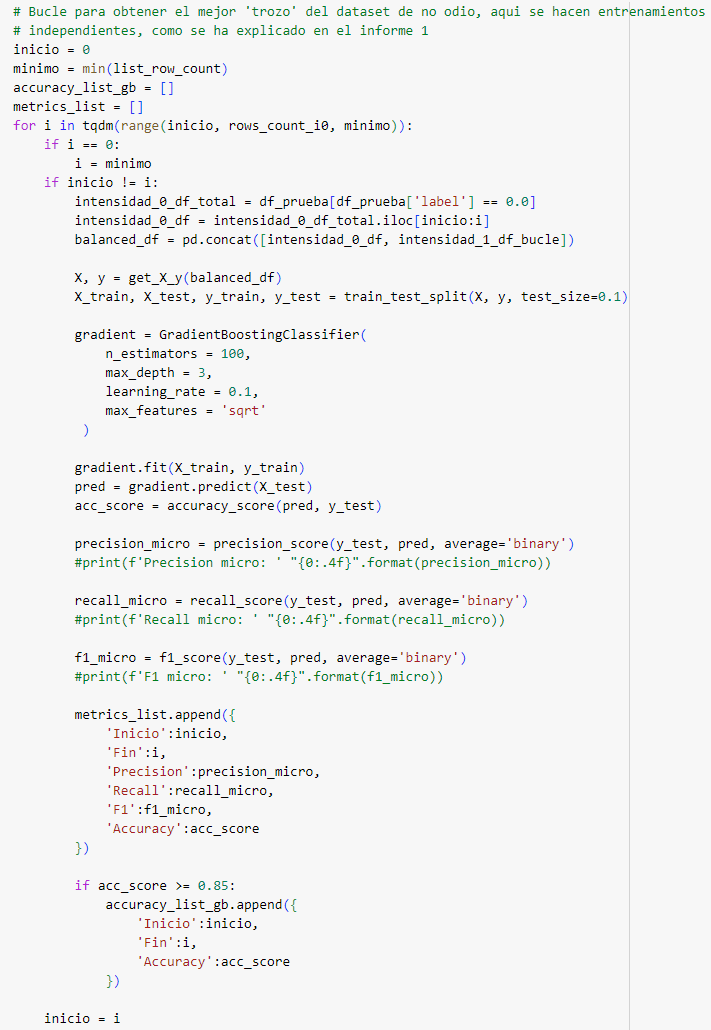
! pip install nombre\_de\_la\_libreria

En las siguientes celdas se realiza lo siguiente:

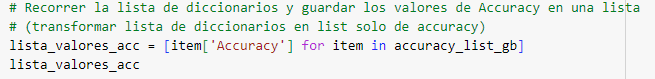
1. La función get\_X\_y(df) retorna por separado las características y la etiqueta asociada (odio = 1, no odio = 0) con las que se entrena y valida el modelo.  
   
2. Leemos el archivo .csv a un DataFrame de pandas.  
   
3. Fusionar intensidades ejecutando  
   

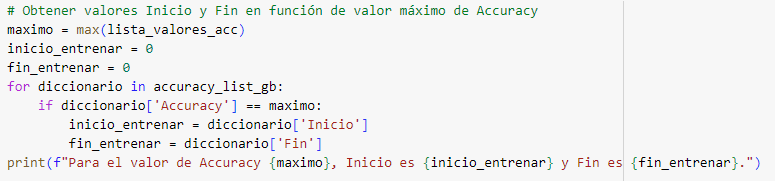
Empezamos preparando el dataset separando las entradas que contienen odio y las que no.

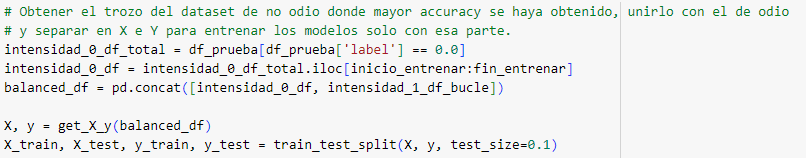


Ejecutamos para obtener las secciones del dataset que consigan un mayor accuracy para posteriormente realizar el entrenamiento de los modelos.

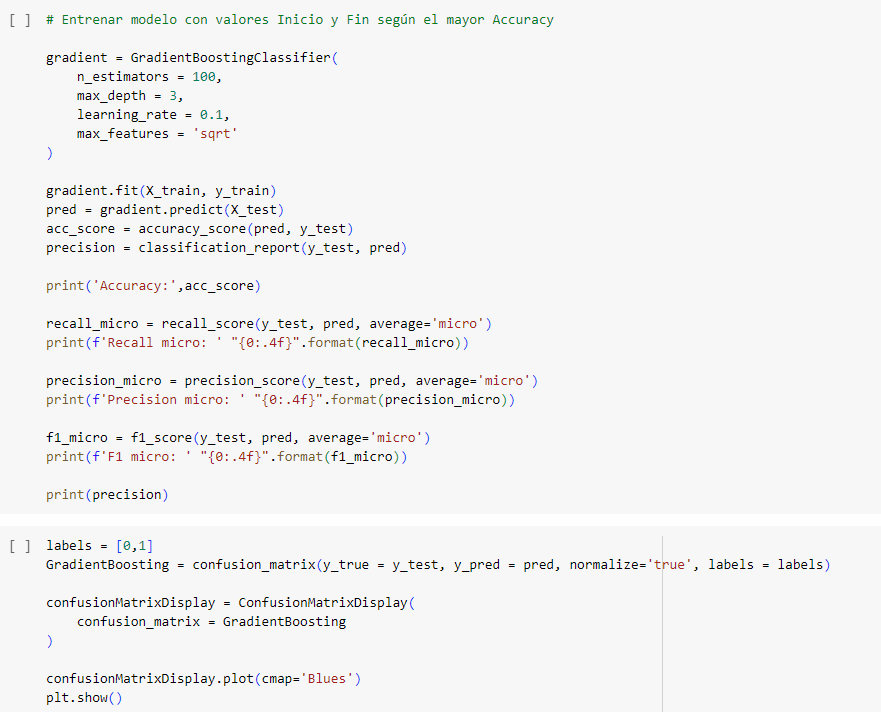
Obtenemos la sección con la que se ha conseguido el mayor accuracy para conformar el cuerpo de entrenamiento y validación de los distintos modelos.







Las siguientes celdas servirán para lanzar el entrenamiento, en este caso, del modelo *Gradient Boosting* (a continuación aparecen las celdas con el código correspondiente al resto de modelos); y visualizar los resultados en el reporte de clasificación y la matriz de confusión pintada con la librería matplotlib.



En una última celda podemos descomentar la segunda línea si queremos exportar un modelo ya entrenado.

