

CLASIFICADOR DE NOTICIAS CON PYTHON

Autor: Joël Plambek

Autor: Angel Escobar Anchuelo

Profesor: Borja Monsalve Piqueras

Asignatura: Proyecto de Computacion I

Universidad Europea de Madrid

1 – Resumen

El sistema que hemos desarrollado es una interfaz gráfica en el lenguaje de programación Python que a partir de un conjunto de noticias categorizadas de “odio” y “no odio” es capaz de distinguir a que categorías pertenece usando un entrenador y un clasificador.

Se disponen de un total de tres algoritmos de aprendizaje procedentes de la actividad previa de RapidMiner, que extraerán de las carpetas preseleccionadas las noticias de odio y no odio, posteriormente, el clasificador obtiene noticias de una carpeta donde hay noticias sin clasificar, lo cual tiene que averiguar de que tipo son dichas noticias.

Una vez clasificadas se almacena el resultado en un archivo .csv donde se indica el nombre de la noticia y la categoría a la que pertenece.

El objetivo principal es que el porcentaje de falsos positivos y de falsos negativos sea lo más reducido posible.

2 - Índice

1 – Resumen	2
2 - Índice	3
3 - Introducción.....	4
3.1 - Contextualización	4
3.2 - Descripción del problema	4
3.3 - Como se organiza el resto del documento	4
4 – Estado del problema	4
5 - Solución.....	5
5.1 - Objetivo general	5
5.2 - Descripción de la solución propuesta.....	5
5.2.1 - ¿Qué hace y que no hace?.....	5
5.2.2 - Tecnologías / librerías empleadas.	5
5.2.3 - Descripción de los datos de entrada	5
5.2.4 - Datos de salida / Resultados	6
5.2.5 - Tratamiento del texto.....	6
5.2.6 - Proceso de entrenamiento	6
6 - Diseño	7
6.1 Fuente de datos y dataset generado	7
7 - Metodología de trabajo.....	9
7.1. Plan de trabajo.....	9
8 - Presupuesto	9
9 - Diseño de pruebas y resultado de las mismas	10
10. Manual de instalación.....	11
10.1 Requeriments.txt	11
11. Manual de usuario	11
12. Conclusiones	11
13. Trabajos futuros.....	12
14. Bibliografía	12

3 - Introducción

3.1 - Contextualización

Nos situamos ante un problema en el que existen cientos de miles de noticias que aumentan día a día, se origina porque estamos en un mundo de la información, donde se extraen datos de casi cualquier actividad por cotidiana que sea, y el análisis y procesamiento de esa información es una ardua tarea. Ese es el motivo de este proyecto, la creación de un sistema que es capaz de a partir de un gran conjunto de noticias, aprender a distinguir si las noticias son de odio, o no lo son.

3.2 - Descripción del problema

El problema que tenemos y para el que necesitamos la solución, es que no tenemos medios fiables por el cual podamos saber de qué categoría es una noticia, por eso, con este proyecto de clasificador en Python solucionamos en un momento un problema facilitando el acceso de información procedente de las noticias por parte de profesionales analíticos que se dedican a extraer datos concretos y poder garantizar una información concreta y actualizada.

3.3 - Como se organiza el resto del documento

En el resto del documento, hablaremos del estado del problema que nos trae entre manos, también hablaremos de la solución propuesta enfocando en las librerías, la arquitectura, tratamiento de los datos, en resumen, hablaremos de todo lo referente al proceso por el cual se genera la solución.

Posteriormente detallare el plan de trabajo, el cómo nos hemos organizado, los tiempos empleados y las tecnologías utilizadas.

Después de detallar los tiempos, estableceremos un presupuesto para la realización del proyecto, manuales de instalación y de usuario para la correcta utilización del software.

Finalmente hablaremos de posibles trabajos a futuro, unas conclusiones y una lista de la bibliografía utilizada durante el proyecto.

4 – Estado del problema

El estado del problema que nos aborda es que actualmente y tras la implementación del software solo se distinguen noticias y se categorizan en “odio” y “no odio”, para un mejor tratamiento de la información sería conveniente amplificar los algoritmos de tal forma de que se puedan llegar a clasificar un mayor número de noticias, con una mayor cantidad de categorías y reducir el tiempo total que tarda en realizar el proceso.

Para ello necesitaremos mejores máquinas y algoritmos, esos detalles los explicaremos más adelante en el apartado de presupuestación.

5 - Solución

5.1 - Objetivo general

En concreto y sintetizadamente, el objetivo del software es la creación de un entrenador a partir de un algoritmo que permita diferenciar entre noticias de odio y de no odio. Y posteriormente probarlo con un conjunto de noticias de categoría desconocida para probar si el modelo funciona o no funciona correctamente.

5.2 - Descripción de la solución propuesta

5.2.1 - ¿Qué hace y que no hace?

Esta aplicación desarrollada genera un modelo entrenado que distingue noticias de odio y de no odio, eso significa no distingue el resto de categorías como política, deporte. Un posible objetivo futuro sería ampliar el espectro de categorías que se pueden identificar

5.2.2 - Tecnologías / librerías empleadas.

Para la realización del proyecto hemos usado:

- Visual Studio Code: Es un entorno de programación en el cual podemos programar en distintos lenguajes, para este proyecto, la interfaz gráfica hemos usado Python y distintas librerías.
- Python 3.10: Es un lenguaje de programación interpretado, multiparadigma, ya que soporta orientación a objetos.
- GitHub: Es una forja para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador.
- Librería Tkinter: Es una librería de Python orientada a facilitar la creación y uso de interfaces de usuario.
- Librería NLTK: Son un conjunto de herramientas dedicado al tratamiento y procesamiento del lenguaje natural (PLN)
- Librería Scikit-Learn: Es una librería de aprendizaje automático de software libre para el lenguaje de programación Python. Incluye varios algoritmos de clasificación, regresión y análisis de grupos.

5.2.3 - Descripción de los datos de entrada

Tenemos dos tipos de entrada:

- A): La entrada que son noticias categorizadas de odio y no odio, la cual es un fichero “.txt” con el contenido de una noticia mostrando titular, entrada y contenido de la noticia
- B): La segunda entrada de datos es una carpeta con noticias de categorías desconocidas y un apartado para importar el modelo entrenado para clasificar estas noticias

5.2.4 - Datos de salida / Resultados

Tenemos dos apartados de salida

- A): La salida en la que se genera el modelo entrenado habiendo hecho uso del algoritmo y un vector
- B): La salida del segundo apartado es un fichero (.csv) donde muestra de forma binaria si la noticia es de odio o de no odio junto con el nombre de la noticia.

5.2.5 - Tratamiento del texto

Para tratar el texto procedente de las noticias hacemos uso del procesamiento del lenguaje natural con la librería NLKT. Este procesamiento consta de varias fases

1 - Tokenización: El proceso de tokenización coge todo el texto y lo separa en tokens, que son unidades de palabras. De esta forma es más fácil hacer el resto de operaciones.

2 – Lowercase: Una vez todo tokenizado, hay que transformar todo a minúsculas y eliminar los signos de puntuación

3 – Stopwords: Aplicamos una lista de parada, consta de eliminar aquellas palabras del lenguaje castellano más comunes, así hacemos que las palabras a analizar sean concretas y relevantes

4 – Stemming: Este último procesamiento consta de sacar la raíz de la palabra y agrupar aquellas que tengan la misma

5.2.6 - Proceso de entrenamiento

Para hacer el proceso de entrenamiento se usan distintos algoritmos que funcionan de distinta forma

1 – Decision Tree: Es un algoritmo de decisión que en base a atributos va decidiendo sobre el tema principal, va rama por rama y según donde acabe saldrá si es de odio o no, si en alguno de esos atributos el algoritmo comete un error, pues la noticia es posible que sea categorizada incorrectamente

2 – Naive Bayes: Este algoritmo dictamina que la presencia o ausencia de una característica particular no está relacionada con la presencia o ausencia de cualquier otra característica, de este modo según las características del documento es capaz de realizar una predicción

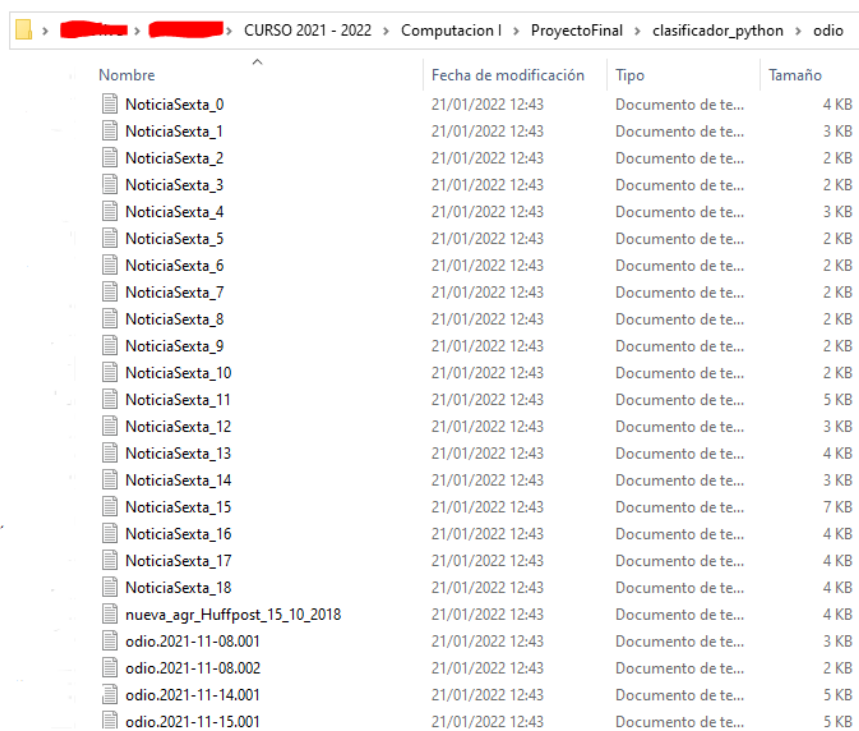
3 – Gradient Boosted Tree: Es una técnica de aprendizaje automático utilizado para el análisis de la regresión y para problemas de clasificación estadística, el cual produce un modelo predictivo en forma de un conjunto de modelos de predicción débiles

6 - Diseño

6.1 Fuente de datos y dataset generado

La fuente de los datos de los que se nutre la aplicación, son dos carpetas que contienen alrededor de 1000 noticias en total extraídas de distintos medios de comunicación, posteriormente mostrare la fotografía del resultado obtenido tras el proceso de clasificación

El resultado obtenido es un documento csv donde se indica el nombre del documento y de forma binaria (1/0) si es de odio o no odio

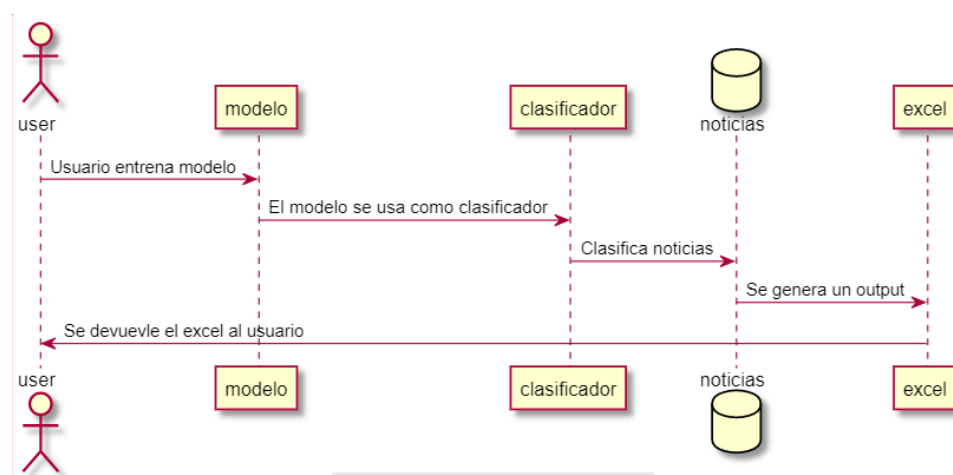


Nombre	Fecha de modificación	Tipo	Tamaño
NoticiaSexta_0	21/01/2022 12:43	Documento de te...	4 KB
NoticiaSexta_1	21/01/2022 12:43	Documento de te...	3 KB
NoticiaSexta_2	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_3	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_4	21/01/2022 12:43	Documento de te...	3 KB
NoticiaSexta_5	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_6	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_7	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_8	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_9	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_10	21/01/2022 12:43	Documento de te...	2 KB
NoticiaSexta_11	21/01/2022 12:43	Documento de te...	5 KB
NoticiaSexta_12	21/01/2022 12:43	Documento de te...	3 KB
NoticiaSexta_13	21/01/2022 12:43	Documento de te...	4 KB
NoticiaSexta_14	21/01/2022 12:43	Documento de te...	3 KB
NoticiaSexta_15	21/01/2022 12:43	Documento de te...	7 KB
NoticiaSexta_16	21/01/2022 12:43	Documento de te...	4 KB
NoticiaSexta_17	21/01/2022 12:43	Documento de te...	4 KB
NoticiaSexta_18	21/01/2022 12:43	Documento de te...	4 KB
nueva_agr_Huffpost_15_10_2018	21/01/2022 12:43	Documento de te...	4 KB
odio.2021-11-08.001	21/01/2022 12:43	Documento de te...	3 KB
odio.2021-11-08.002	21/01/2022 12:43	Documento de te...	2 KB
odio.2021-11-14.001	21/01/2022 12:43	Documento de te...	5 KB
odio.2021-11-15.001	21/01/2022 12:43	Documento de te...	5 KB

Nombre	Fecha de modificación	Tipo	Tamaño
20min_ciencia.2022-01-06.001	21/01/2022 12:42	Documento de te...	3 KB
20min_ciencia.2022-01-06.002	21/01/2022 12:42	Documento de te...	4 KB
20min_ciencia.2022-01-07.001	21/01/2022 12:42	Documento de te...	7 KB
20min_ciencia.2022-01-07.002	21/01/2022 12:42	Documento de te...	4 KB
20min_ciencia.2022-01-07.003	21/01/2022 12:42	Documento de te...	4 KB
20min_ciencia.2022-01-07.004	21/01/2022 12:42	Documento de te...	6 KB
20min_ciencia.2022-01-07.005	21/01/2022 12:42	Documento de te...	6 KB
20min_ciencia.2022-01-07.006	21/01/2022 12:42	Documento de te...	5 KB
20min_ciencia.2022-01-07.007	21/01/2022 12:42	Documento de te...	3 KB
20min_ciencia.2022-01-07.008	21/01/2022 12:42	Documento de te...	3 KB
20min_ciencia.2022-01-07.009	21/01/2022 12:42	Documento de te...	2 KB
20min_ciencia.2022-01-08.001	21/01/2022 12:42	Documento de te...	2 KB
20min_ciencia.2022-01-08.002	21/01/2022 12:42	Documento de te...	3 KB
20min_ciencia.2022-01-08.003	21/01/2022 12:42	Documento de te...	3 KB
20min_ciencia.2022-01-08.004	21/01/2022 12:42	Documento de te...	7 KB
20min_ciencia.2022-01-09.001	21/01/2022 12:42	Documento de te...	2 KB
20min_ciencia.2022-01-09.002	21/01/2022 12:42	Documento de te...	2 KB
20min_ciencia.2022-01-09.003	21/01/2022 12:42	Documento de te...	2 KB
20min_ciencia.2022-01-09.004	21/01/2022 12:42	Documento de te...	2 KB
20min_ciencia.2022-01-09.005	21/01/2022 12:42	Documento de te...	5 KB
20min_ciencia.2022-01-09.006	21/01/2022 12:42	Documento de te...	7 KB
20min_ciencia.2022-01-09.007	21/01/2022 12:42	Documento de te...	10 KB
20min_ciencia.2022-01-10.001	21/01/2022 12:42	Documento de te...	3 KB
20min_ciencia.2022-01-10.002	21/01/2022 12:42	Documento de te...	4 KB

	A	B	C
1	name,prediction		
2	salud.2022-01-15.001.txt,0.0		
3	salud.2022-01-14.004.txt,0.0		
4	salud.2022-01-14.005.txt,0.0		
5	odio.2021-10-08.001.txt,1.0		
6	odio.2021-09-27.001.txt,1.0		
7	salud.2022-01-14.002.txt,0.0		
8	salud.2022-01-14.003.txt,0.0		
9	odio.2021-09-27.002.txt,0.0		
10	salud.2022-01-14.001.txt,0.0		
11	salud.2022-01-20.001.txt,0.0		
12	salud.2022-01-20.003.txt,0.0		
13	salud.2022-01-16.001.txt,0.0		
14	salud.2022-01-20.002.txt,0.0		
15	salud.2022-01-17.001.txt,0.0		
16	salud.2022-01-20.006.txt,0.0		
17	salud.2022-01-20.007.txt,0.0		
18	salud.2022-01-20.005.txt,0.0		
19	odio.2021-10-25.001.txt,0.0		
20	salud.2022-01-20.004.txt,0.0		
21	salud.2022-01-20.008.txt,0.0		

6.2 Diagrama UML de clases



7 - Metodología de trabajo

7.1. Plan de trabajo

Los dos autores de este proyecto haciendo uso de GitHub han colaborado en la elaboración del código y en el desarrollo de esta documentación

Haciendo un sistema de metodología ágil basado en Sprint, estableciendo unas metas a cumplir para una determinada fecha.

También hemos hecho un tablero Kamban para visualizar de forma más óptima el trabajo a realizar

Empezamos con el desarrollo en Diciembre y terminamos de picar código el 14 de Enero de 2022

Uno se ha dedicado más en el tema de la interfaz y otro en el del entrenamiento según preferencias personales pero los dos hemos colaborado en todo

8 - Presupuesto

Para la correcta evaluación de un posible presupuesto se han de tener en cuenta los posibles ingresos generados a partir de la venta de este software

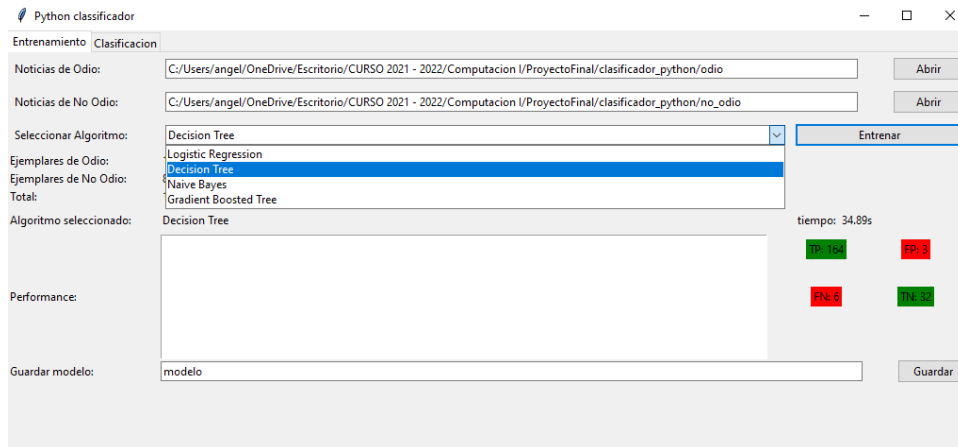
En adición, para mejorar este software necesitaríamos más poder de cómputo para hacer frente a más noticias y a más algoritmos más sofisticados para poder realizar entrenamientos más efectivos que generen menor porcentaje de error.

Un presupuesto de 15.000€ sería necesario para empezar el desarrollo y obtener unos resultados gratamente superiores

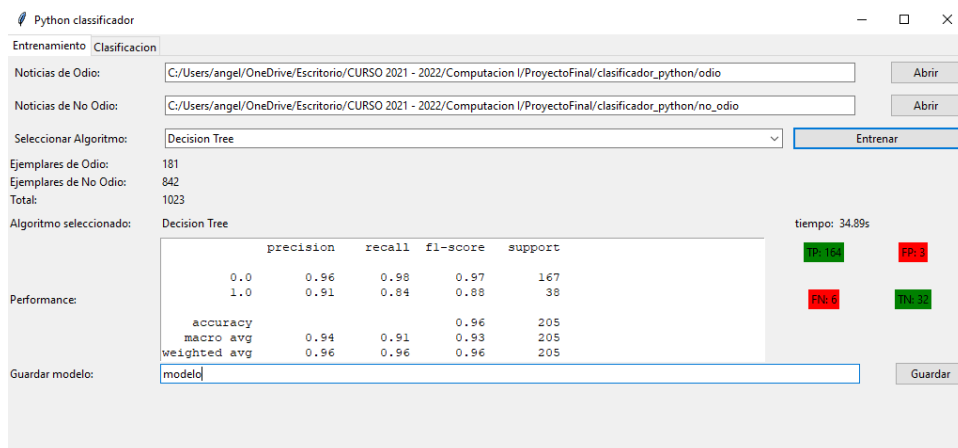
9 - Diseño de pruebas y resultado de las mismas

En este apartado mostrare la interfaz y como funciona, haciendo un pequeño ensayo de la aplicación

Fotografía #1: Interfaz de entrenamiento para seleccionar el algoritmo, una vez seleccionado junto con la ruta de donde se van a extraer las noticias, se puede dar al boton para entrenar

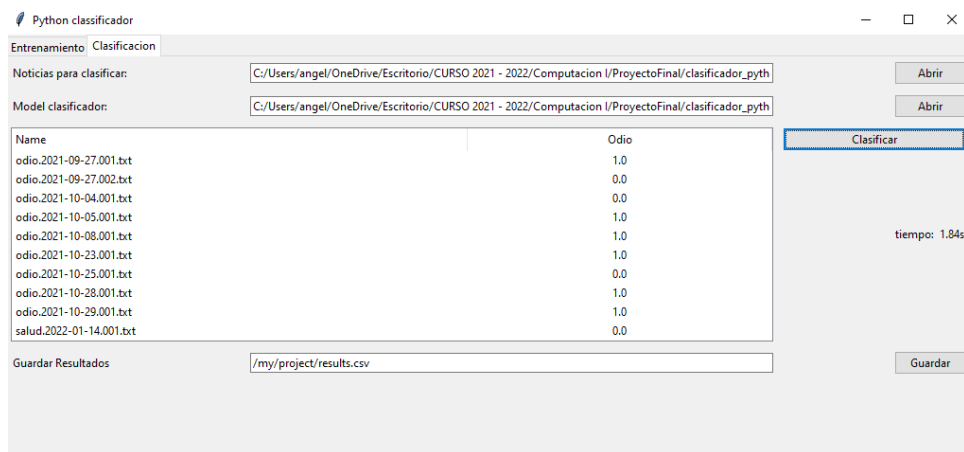


Fotografía #2: En esta imagen muestro el resultado de un entrenamiento, se aprecia como el porcentaje de falsos positivos/negativos es muy pequeño, solo queda guardar el modelo generado



Fotografía #3: Aquí referenciamos la ruta donde están las noticias de categoría desconocida y el modelo generado previamente, podemos ver como analiza y establece si son de odio o no lo son

Luego permite el guardado del csv con los resultados obtenidos



10. Manual de instalación

10.1 Requeriments.txt

Para visualizar este software se necesitan una serie de librerías: “nltk”, “tkinter”, “sklearn” y “IPython”

Una vez instaladas ya se podrá ejecutar el código y visualizar la interfaz gráfica

11. Manual de usuario

Para hacer uso de esta aplicación, primero se deben seleccionar las carpetas donde estarán contenidas las noticias de “odio” y “no odio” abriendo el explorador de archivos e introduciendo la ruta.

El siguiente paso es seleccionar un algoritmo y darle a entrenar, tras un breve lapso de tiempo se mostrará en un campo de texto el resultado del entrenamiento, dándole un nombre a un modelo podrá guardar el modelo generado

A continuación, en clasificador, se debe seleccionar la carpeta contenedora de noticias que queremos analizar, así como un modelo clasificador, dándole al botón de clasificar, mostrará la clasificación de noticias en “odio” y “no odio”.

Finalmente, solo quedará guardar el archivo generado con los resultados obtenidos.

12. Conclusiones

En conclusión, a este proyecto final, se ha mostrado la forma de a partir de un conjunto prefijado de noticias obtenidas mediante técnicas ETL, el proceso de entrenamiento de un modelo para clasificación

Se quiere comprobar cómo es posible realizar un trabajo a “priori” simple, convendría llevarlo a gran escala para facilitar el acceso a información de muchísima gente

Sería una gran opción seguir trabajando en este aspecto

13. Trabajos futuros

Poniendo la vista en el futuro, habría que mejorar el proyecto para que aparte de noticias de “odio” y “no odio” permita reconocer otro tipo de categorías, aparte de incrementar el número de noticias a analizar, para ello necesitaríamos un computador mucho más potente

14. Bibliografía

Para la realización de este trabajo se han utilizado

Stack overflow -> Se buscaban soluciones a los errores de compilación que iban surgiendo

<https://likegeeks.com/es/tutorial-de-nlp-con-python-nltk/>

<https://www.master-data-scientist.com/scikit-learn-data-science/>

<https://docs.python.org/es/3/library/tk.html>