

23 DE MAYO DE 2024

HateSpeechDetector



Manuel Estévez Simonet
Carla Murillo Fernández
Anel Hizaguirre Valle
María Isabel Prieto Guerrero

Indice

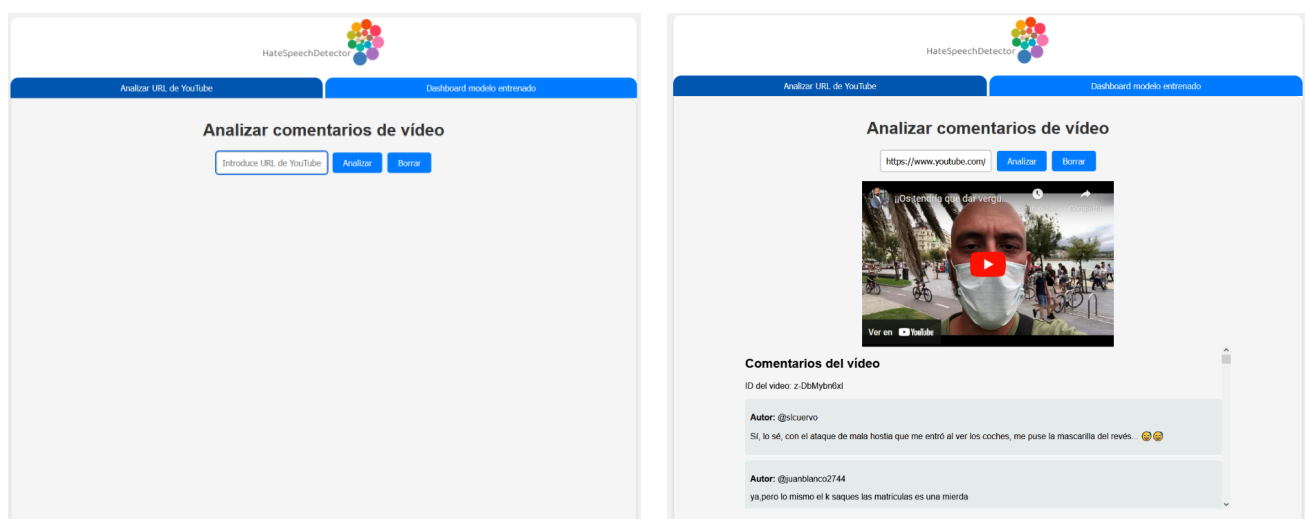
Hate Speech Detector	3
Metodologia utilizada	4
• Arquitectura	4
• Análisis exploratorio	8
• Procesado	12
• Modelado	14
Predicciones datos YouTube	18
Conclusiones	19

Hate Speech Detector

Hate Speech Detector surge como una respuesta a la creciente preocupación por la proliferación de comentarios de odio en las plataformas digitales. En la era de la información, el acceso a internet y las redes sociales ha facilitado la comunicación global, pero también ha dado lugar a la diseminación de discursos nocivos que pueden tener consecuencias graves en la salud mental y el bienestar de las personas. La necesidad de herramientas que puedan identificar y mitigar estos comentarios se ha vuelto imprescindible para fomentar un entorno en línea más seguro y respetuoso.

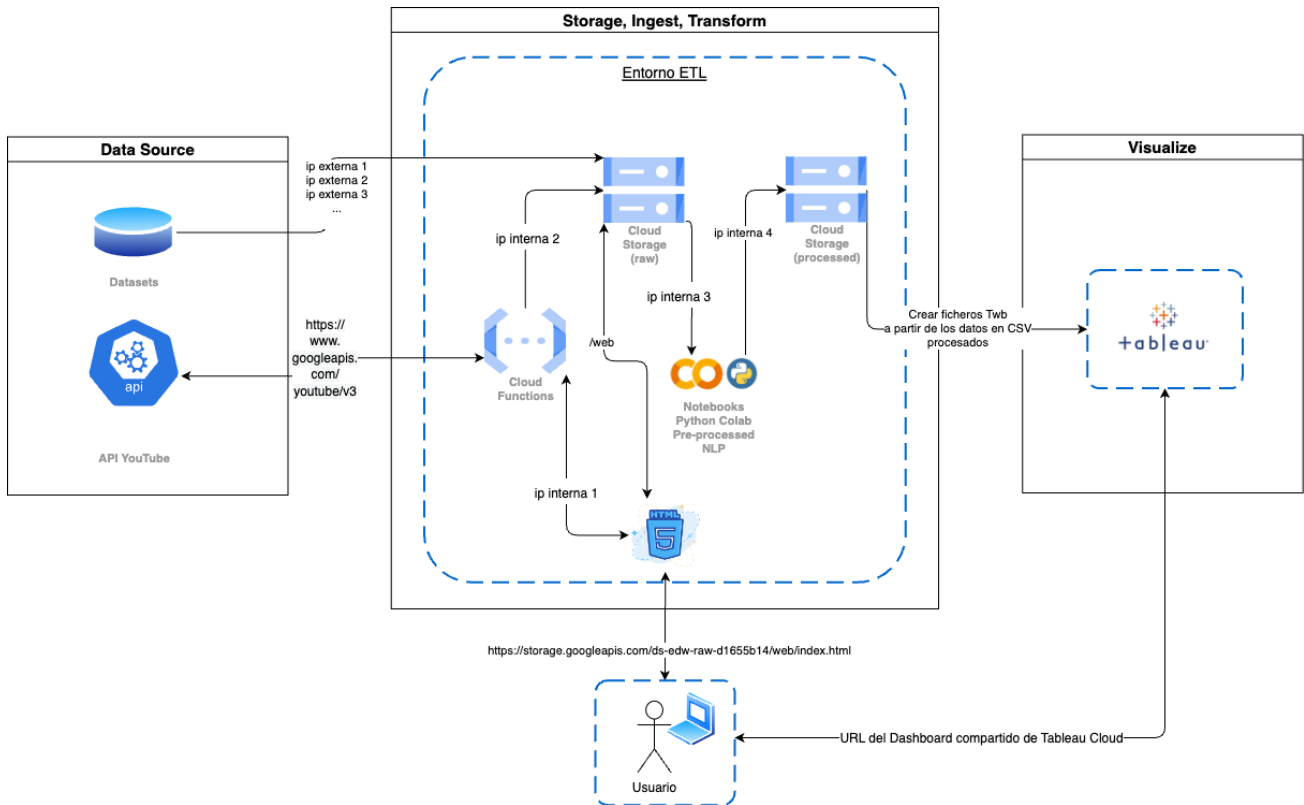
Pero... ¿Discurso de odio que significa? Naciones Unidas define el discurso de odio como "cualquier tipo de comunicación ya sea oral o escrita, —o también comportamiento—, que ataca o utiliza un lenguaje peyorativo o discriminatorio en referencia a una persona o grupo en función de lo que son, en otras palabras, basándose en su religión, etnia, nacionalidad, raza, color, ascendencia, género u otras formas de identidad".

Hate Speech Detector emplea avanzados algoritmos de inteligencia artificial y procesamiento del lenguaje natural para detectar automáticamente el lenguaje ofensivo y los comentarios de odio en tiempo real en la plataforma YouTube, como se ejemplifica en el apartado Predicciones datos YouTube.



Metodología utilizada

1. Arquitectura



La creación de un entorno en Google Cloud para analizar y procesar datos relacionados con mensajes de odio en YouTube. Este proyecto implica varios pasos críticos, desde la recopilación de datasets hasta el entrenamiento de modelos de aprendizaje automático capaces de identificar comentarios tóxicos.

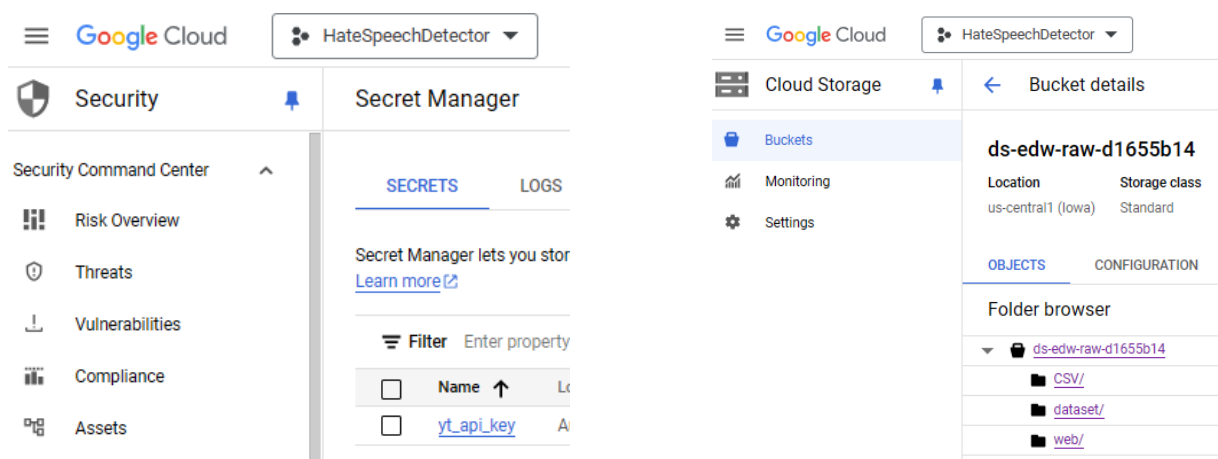
En primer lugar, se recopilan datasets que contengan ejemplos de mensajes de odio y otros tipos de comentarios. Estos datasets provienen de fuentes públicas y están debidamente etiquetados para facilitar el entrenamiento del sistema. Una vez obtenidos los datos, se almacenan en Google Cloud Storage, una solución escalable que permite administrar grandes volúmenes de información de manera eficiente.

1. Estrategia del DAaaS

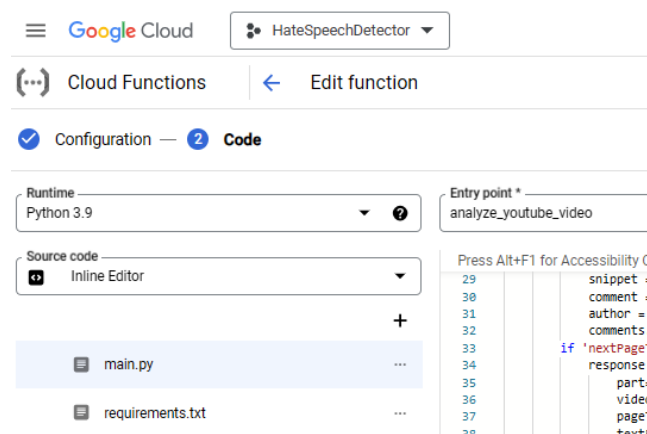
- Una web y un software que, mediante Google Cloud Functions, se conecta a la API de YouTube (marco Data Source), y descarga en "raw", sin tratar, todos los ficheros siempre vienen en formato CSV y se almacenan en un bucket de Google Cloud Storage, y dicho Google Cloud Storage está dentro de un marco SIT (Storage, Ingest, Transform).

Para ello se hace uso de las API de YouTube:

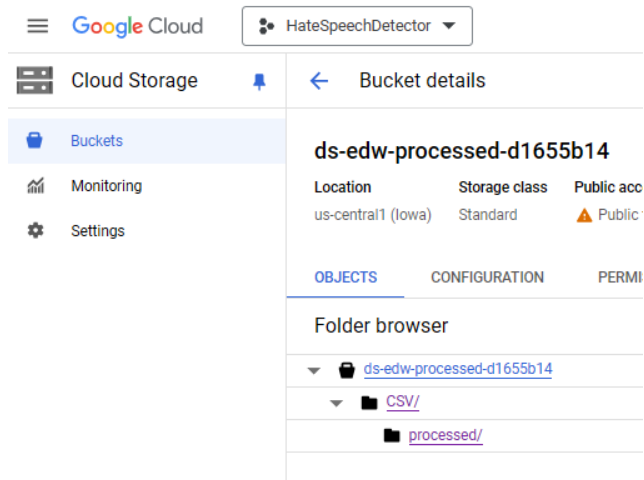
API de YouTube (<https://www.googleapis.com/youtube/v3>)



- Desde la web, el usuario introduce una URL válida de YouTube y solicita "analizar los comentarios del vídeo", esta petición lanzará el software de la Google Cloud Functions. Una vez se ejecute dicho software y ya los datos estén en el bucket raw del Google Cloud Storage, mediante herramientas como Google Colab, con Notebooks en lenguaje Python, se realizan todas las funciones del proceso ETL para tratar los datos "raw".



- Cuando ya se han tratado los datos, se vuelven a llevar a un nuevo bucket processed de Google Cloud Storage.



- Por otro lado, y aun dentro del mismo marco SIT, se procede a la carga de dichos comentarios en la web para poder visualizarlos, a la espera de los resultados.
- Finalmente, y ya en un nuevo marco Visualize, mediante la herramienta Tableau, se construyen dashboards "Ad-Hoc" para que los puedan utilizar y visualizar usuarios, viendo datos, gráficas, e incluso los conocidos emojis, muy usados para comentar.

2. Arquitectura DAaaS

- Fuentes de datos:
 - Datasets de terceros
 - API de YouTube
- Componentes:
 - Google Cloud Functions
 - Google Cloud Storage
 - Notebooks Python en Google Colab
 - Tableau

3. DAaaS Operating Model Design and Rollout

- Delimitar que se van a tener 3 marcos
 - El primero, con las fuentes de origen, marco Data Source.
 - A continuación, donde estará la ingesta, tratamiento y almacenamiento de los datos, marco SIT (Storage, Ingest, Transform)
 - Finalmente, para poder visualizar todos esos datos de forma, limpia, ordenada y personalizada, tendremos el marco Visualize, que dará uno de los resultados a visualizar
- Crear y preparar el entorno cloud con todas las instancias necesarias, como:
 - Google Cloud Functions
 - Google Cloud Storage
 - Notebooks Python en Google Colab
 - Tableau
- El software desarrollado, consultará las API en busca de los datos solicitados.
- Se descargarán los ficheros CSV generados en la consulta a la API y se compararán con los ya previamente entrenados de los dataset.

2. Análisis exploratorio

El análisis exploratorio se centra en un conjunto de datos compuesto por múltiples archivos CSV que contienen comentarios de redes sociales clasificados como discursos de odio o no odio. Se realizan diversas etapas de preprocesamiento simple y análisis para obtener una comprensión detallada de la presencia de emoticonos y la naturaleza del contenido textual en estos comentarios.

1. Preparación del entorno

- Se utilizan librerías de **Python** como ``pandas``, ``numpy``, ``regex``, ``nltk``, ``matplotlib``, ``emoji``, ``unidecode``, ``num2words`` y ``wordcloud``.
- Se desarrollan **funciones** para realizar diversas tareas como, contar emoticonos, eliminar nombres de usuario, eliminar signos de puntuación, convertir números a palabras, etc.

2. Carga y exploración de los datos

- Los datos se cargan desde cuatro archivos:
 - `hascosva_2022.csv`
 - `hsd_merge_cleaned_lowered.csv`
 - `Spain_test.csv`
 - `Hate_Speech_in_Spain.csv`
- Se extraen muestras aleatorias de cada archivo para una rápida inspección.
- Se concatenan todos los DataFrames en uno solo, eliminando columnas innecesarias y convirtiendo la columna ``label`` a tipo ``int64``.

	text	label
0	Avión venezolano: ocho tripulantes coinciden e...	0
1	Aquí pidiéndole consejos al amigo veneco pa so...	0
2	@DianaDvilaNarv1 Cínicos desinforman con guión...	0
3	@DianAngel01 No se chama dime tu 🤔🤔🤔🤔🤔🤔	0

3. Análisis exploratorio de datos (EDA)

- Conteo de emoticonos:
 - ▶ 4.288 emoticonos en total.
 - ▶ 2.072 registros contenían al menos un emoticono.
 - ▶ 11.28% de los registros contenían emoticonos.
- Filtrado y limpieza:
 - ▶ Eliminación:
 - Registros que contenían emoticonos.
 - Nombres de Usuario.
 - Signos de puntuación.
 - ▶ Conversión de números a palabras y mayúsculas a minúsculas.

	text	label
0	avion venezolano ocho tripulantes coinciden en...	0
1	aqui pidiendole consejos al amigo veneco pa so...	0
2	cinicos desinforman con guion preparado ecena ...	0
4	gobierno de belgica ustedes que albergan a raf...	0
6	y esto es lo que pasa cuando dejas a autentico...	0

- Cálculo la cardinalidad del vocabulario para cada comentario, proporcionando una medida de la diversidad léxica de cada comentario.

	text	label	cardinalidad_vocabulario
0	avion venezolano ocho tripulantes coinciden en...	0	13
1	aqui pidiendole consejos al amigo veneco pa so...	0	9
2	cinicos desinforman con guion preparado ecena ...	0	10
4	gobierno de belgica ustedes que albergan a raf...	0	32
6	y esto es lo que pasa cuando dejas a autentico...	0	29

4. Análisis de distribución de comentarios

- Distribución:
 - ▶ No Odio (0): 16.804 comentarios.
 - ▶ Odio (1): 11.080 comentarios.

- Se añaden columnas adicionales para la longitud del comentario y el recuento de palabras, proporcionando información sobre la extensión de los textos.

	cardinalidad_vocabulario	Longitud del Comentario	Recuento de Palabras
0	13	99	13
1	9	68	10
2	10	98	10
4	32	261	38
6	29	203	33

5. N-grams y nubes de palabras

- Se identificaron los **n-grams** más frecuentes (pares de palabras) en los comentarios, lo cual ayuda a entender las combinaciones de palabras más comunes.
- Nubes de Palabras. Se genera una nube de palabras para los comentarios clasificados.
 - ▶ No Odio (0): 'Ma', 'Si', 'Inmigrante', 'Mujeres', 'Mujer', etc.
 - ▶ Odio (1): 'Inmigrante', 'Comunista', 'Fascista', 'Subnormal', 'mierda', etc.

Las nubes de palabras permiten visualizar las palabras más frecuentes en cada categoría, facilitando la identificación de temas recurrentes.

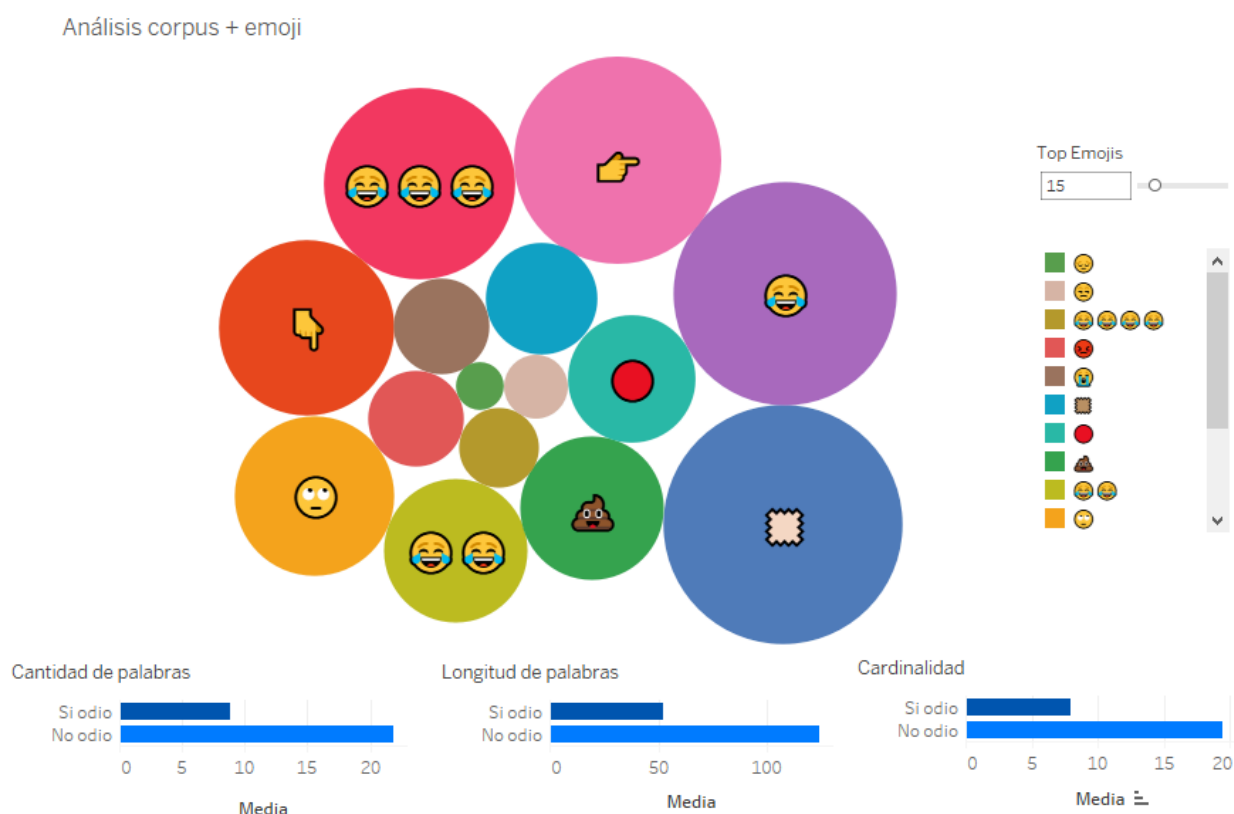


*Oodio(1)

Una minoría significativa de los comentarios contiene emoticonos, lo cual podría ser relevante para el análisis de sentimientos o para detectar patrones en los comentarios de odio.

La eliminación de emoticonos y la limpieza de los textos fueron cruciales para obtener un conjunto de datos adecuado para análisis textuales más profundos.

Tras el análisis exploratorio se representan los puntos analizados con Tableau.



3. Procesado de los datos

El objetivo principal es limpiar y balancear el conjunto de los datos textuales y emoticonos, seguido de la carga del archivo csv. resultante al bucket de Google Cloud Storage. Con el fin de conseguir los mejores resultados en los modelos predictivos.

1. Preparación del entorno

- Se utilizan librerías de Python como `pandas`, `numpy`, `regex`, `nltk`, `matplotlib`, `emoji`, `unidecode`, `num2words` y `wordcloud`.
- Se desarrollan 2 funciones:
 - ▶ Procesar datos de texto.
 - ▶ Procesar emoticonos.

2. Carga de los datos

- Los datos se cargan desde el archivo generado en el análisis de datos `df_concat.csv`.
- Se eliminan los valores nulos para asegurar la calidad de los datos.



3. Balanceo de clases en datos de texto

Los datos que tratamos están ya categorizados con el label (0) y (1), pero pueden ser datos desbalanceados. En un conjunto de datos desbalanceado, una clase (la mayoritaria) tiene muchas más instancias que otra (la minoritaria). Este desajuste puede llevar al modelo de aprendizaje automático a sesgarse hacia la clase mayoritaria, disminuyendo la precisión y utilidad del modelo para predecir la clase minoritaria.

Se realiza un submuestreo aleatorio de la clase mayoritaria para que su tamaño sea proporcional al de la clase minoritaria, basándose en un ratio de balanceo predefinido.

4. Extracción de emoticonos

- Se crea una nueva columna que contiene listas de emoticonos extraídos para cada fila. Se recopilan todos los emoticonos en una lista única para facilitar el conteo y análisis.
- Se cuenta la frecuencia de cada emoji en la lista única. Se calcula el porcentaje de aparición de cada emoji respecto al total de emojis extraídos, proporcionando una medida relativa de su prevalencia en el conjunto de datos.

	emoji	% del total
0		5.99
1		5.34
2		2.77
3		2.54
4		1.95

5. Limpieza y normalización del texto balanceado

- Eliminación:
 - Emoticonos.
 - Nombres de Usuario.
 - Signos de puntuación.
- Conversión de números a palabras y mayúsculas a minúsculas.

```

0  arco forum recibe este viernes la medalla de o...
1  las reacciones a la muerte de chacon de todo s...
2  millones de chilenos no votan nunca otros tant...
3  dime a mi y yo te hago reir com osea para que ...
4  se compraron esas aireadoras culias de papas f...
5  si yo entrego un goya y el que lo gana no esta...
6  es cultural de nosotros los ecuatorianos ser m...
7                                     malditos
8      y educarlos tu que son tus hijos te imaginas
9  y se espera que nuevos flujos de inmigrantes d...
    
```

4. Modelado

1. Preparación del entorno

Se utilizan librerías de Python como `pandas`, `numpy`, `matplotlib`, `sklearn`.

2. Carga de los datos

Los datos se cargan desde el archivo generado en el procesamiento de los datos `df_processed.csv`.

3. División de datos

Se dividen los datos en Train y Test comprobando que ambos conjuntos de train tienen los datos que han de tener.

9480	choco un video aficionado registro un angustio...	9480	0
22396	toneladas de inmigrantes musulmanes africanos	22396	1
20199	gilipollez progre macho	20199	1
1373	con justa razon esas canciones que mujer mas b...	1373	0
14364	habla como chileno antes de irte culiao weon l...	14364	1
14669	recambio en equipo de chanta expertos ingresa ...	14669	1
9406	dices que sois la solucion para todos los cata...	9406	0
1135	un poquito de por favor los indigenas al igual...	1135	0
22136	mentirosos populares	22136	1
426	en cataluna ha ganado la derecha la independen...	426	0

Name: processed_text, dtype: object

Name: label, dtype: int64

4. Entrenamiento del modelo

- Se aplica `TfidfVectorizer` de la biblioteca `scikit-learn` para transformar el conjunto de datos de texto en una representación numérica basada en la técnica TF-IDF (Term Frequency-Inverse Document Frequency). Durante el proceso, el vectorizador aprende el vocabulario de los datos de entrenamiento y calcula las estadísticas necesarias para calcular las puntuaciones TF-IDF.

```
TfidfVectorizer
TfidfVectorizer(max_df=0.95, max_features=2500, min_df=3)
```

- ▶ Descarte de las palabras que aparecen en mas del 95% de los comentarios ya que son términos comunes que no son muy informativos.
- ▶ Se ignoran las palabras que aparecen en menos de 3 comentarios.
- ▶ Límite en el número de características, 2.500 términos más importantes según la puntuación TF-IDF.
- ▶ Solo se consideran unigramas (palabras individuales).

El proceso de transformación es crucial ya que garantiza que tanto los datos de entrenamiento y de prueba estén en el mismo formato y escala.

● Modelo Regresión logística.

Se implementa un modelo de regresión logística utilizando la técnica de búsqueda de hiperparámetros `GridSearchCV` para encontrar el mejor valor de regularización C . Se obtienen un mejor valor de C de 5.

```

Accuracy en train: 0.8879
Accuracy en test:  0.8428
Confussion matrix:
[[7712 801]
 [1110 7418]]

Classification report:

```

	precision	recall	f1-score	support
0	0.87	0.91	0.89	8513
1	0.90	0.87	0.89	8528
accuracy			0.89	17041
macro avg	0.89	0.89	0.89	17041
weighted avg	0.89	0.89	0.89	17041

El modelo de regresión logística muestra un buen desempeño, con una precisión del 88.79% en el conjunto de entrenamiento (train) y del 84.28% en el conjunto de prueba (test), indicando una adecuada generalización a datos nuevos. La matriz de confusión revela un manejo equilibrado de verdaderos positivos y negativos, con F1-scores de 89% para ambas clases.

● Modelo Gradiente Boosting.

Se implementa un modelo de Gradiente Boosting utilizando la técnica de búsqueda de hiperparámetros `GridSearchCV` para encontrar el mejor número de estimadores que en este caso es 1.000.

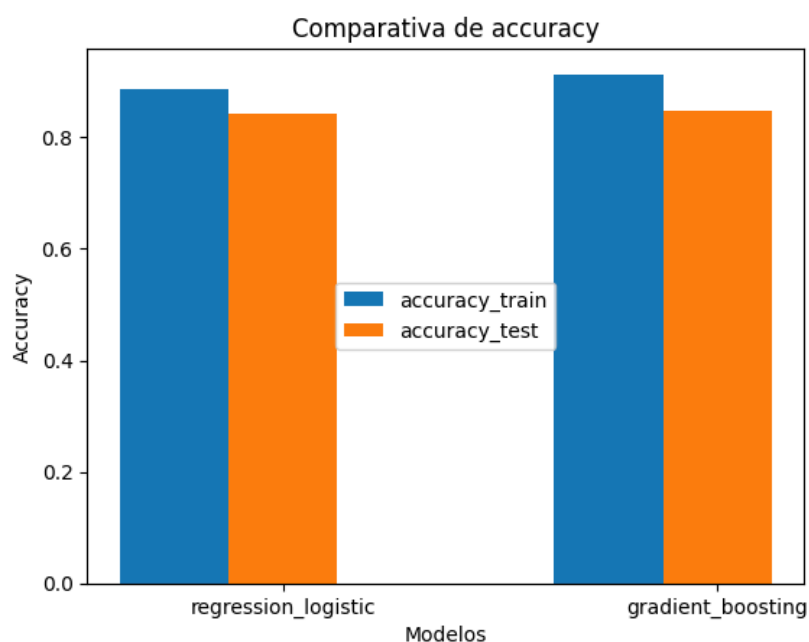
Accuracy en train: 0.9139
Accuracy en test: 0.8481

Confussion matrix:
[[7957 556]
[912 7616]]

Classification report:

	precision	recall	f1-score	support
0	0.90	0.93	0.92	8513
1	0.93	0.89	0.91	8528
accuracy			0.91	17041
macro avg	0.91	0.91	0.91	17041
weighted avg	0.91	0.91	0.91	17041

La precisión en el conjunto de entrenamiento (train) es del 91,39%, mientras que en el conjunto de prueba (test) es del 84,81%. A pesar de la diferencia ambas son relativamente altas. Esto indica que el modelo es capaz de realizar predicciones con un alto grado de exactitud en ambos conjuntos de datos y todavía puede generalizar adecuadamente nuevos datos.



Se deduce que el modelo de Gradient Boosting exhibe un desempeño superior, evidenciado por su mayor precisión (91% en contraste con 88% del modelo de Regresión Logística).

Mediante una función se compara la etiqueta del comentario, con la predicción obtenida por el modelo.

Regresión logística

Review no. 1955
Actual sentiment: 0
Prediction: [0]

Review no. 7492
Actual sentiment: 0
Prediction: [0]

Review no. 5217
Actual sentiment: 0
Prediction: [0]

Review no. 7566
Actual sentiment: 0
Prediction: [1]

Review no. 6769
Actual sentiment: 0
Prediction: [0]

Gradient Boosting

Review no. 12163
Actual sentiment: 1
Prediction: [1]

Review no. 4551
Actual sentiment: 0
Prediction: [0]

Review no. 22247
Actual sentiment: 1
Prediction: [1]

Review no. 14959
Actual sentiment: 1
Prediction: [0]

Review no. 21279
Actual sentiment: 1
Prediction: [1]

Predicción datos Youtube

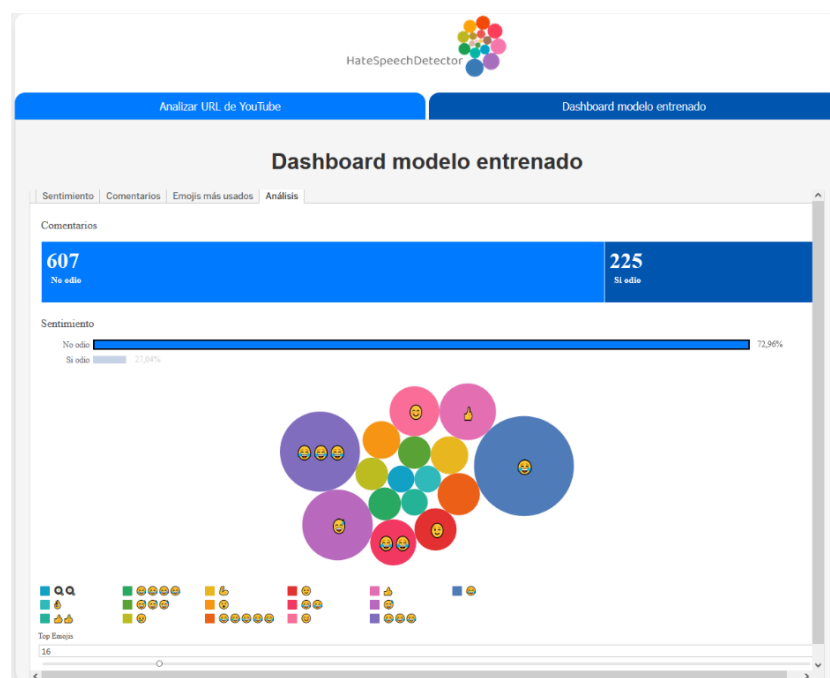
- Lectura de los nuevos datos ya procesados. Anteriormente se procesan los datos a partir de un código mejorado y adaptado.

	Autor	new_processed
0	@slcuervo	si lo se con el ataque de mala hostia que me e...
1	@juanblanco2744	yapero lo mismo el k saques las matriculas es ...
2	@crikso1980	pero eso pasa en todos los sitiosno es nada es...
3	@ramoncillerofernandez6706	senor si la tarifa en el punto de recarga fue...
4	@josefasuarezsanchez8420	saul esto es espana no hay verguenza que valga

- Se vectorizan los nuevos comentarios para poder predecir su etiqueta.
- Se calculan las predicciones con ambos modelos.

	comments	Logistic Regression	Gradient Boosting
0	si lo se con el ataque de mala hostia que me e...	0	0
1	yapero lo mismo el k saques las matriculas es ...	1	0
2	pero eso pasa en todos los sitiosno es nada es...	0	0
3	senor si la tarifa en el punto de recarga fue...	0	0
4	saul esto es espana no hay verguenza que valga	1	0

- Los resultados del modelo de Gradient Boosting se visualizan en la plataforma web mediante Tableau, destacando su elección debido a su superior rendimiento en comparación con el modelo de Regresión Logística.



Conclusiones

● Suposiciones iniciales. Cuales han demostrado ser válidas y cuáles no. ¿Por qué?

La primera suposición fue que los comentarios de odio abundan principalmente en las redes sociales, especialmente en la app TikTok. Se pretende investigar los comentarios en distintas redes, pero nos encontramos con la imposibilidad de acceder a esos comentarios, ya que solo YouTube tiene una API pública y viable para llevar a cabo esta investigación. Aun así, en YouTube, observamos que en un video aparentemente polémico, los comentarios de odio son menos numerosos que los demás comentarios. Sin embargo, este resultado puede variar según el video que se investigue.

● Métricas seleccionadas: ¿han sido las correctas o no? ¿por qué?

Las métricas son adecuadas para el modelo de predicción de mensajes de odio, ya que:

- ▶ **Accuracy:** La precisión mide la proporción de predicciones correctas sobre el total de predicciones. Es útil para obtener una visión general del rendimiento del modelo.
- ▶ **Precision:** Es vital en la detección de mensajes de odio, ya que minimiza los falsos positivos, evitando etiquetar mensajes no ofensivos como ofensivos.
- ▶ **Recall:** Crucial para asegurarse de que el modelo identifique la mayoría de los mensajes de odio, reduciendo los falsos negativos. Esto es esencial para no pasar por alto mensajes que realmente son ofensivos.
- ▶ **F1-Score:** Equilibra precisión y recall, proporcionando una medida que refleja tanto la exactitud de las predicciones positivas como la capacidad del modelo para encontrar todas las instancias de mensajes de odio.
- ▶ **Matriz de confusión:** La matriz de confusión proporciona una visión detallada de los verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos. Esto ayuda a identificar dónde y cómo el modelo está cometiendo errores.

- ▶ **Macro Average:** Calcula la media de las métricas para cada clase por separado y luego las promedia, tratando todas las clases por igual. Esto es útil cuando se quiere asegurar que el modelo tenga un buen rendimiento en cada clase, sin importar su frecuencia.
- ▶ **Weighted Average:** El promedio ponderado toma en cuenta la frecuencia de cada clase, dando más peso a las clases más comunes. Esto proporciona una medida más representativa del rendimiento general del modelo.

● **Arquitectura elegida: ¿ha sido la correcta o no? ¿por qué?**

Se elige crear una arquitectura DAaaS (Data Analysis as a Service). Se considera que es la correcta puesto que es una buena forma que podemos hacer uso de herramientas pensadas para tal propósito, desde la de almacenamiento, ingesta y tratamiento de los datos, hasta su visualización. Nos da la facilidad y la centralización el tener montado todo en el ecosistema Google Cloud, haciendo uso de los bucket como almacenamiento de los datos, para interactuar con las diferentes piezas, usamos Cloud Functions, Secrets para consumir APIs con las API Key obtener los datos, herramientas de Google como Google Colab.

● **Métodos de ML utilizados, ¿cuales han sido los mejores?**

Aunque ambos modelos han obtenido una accuracy bastante alta, el mejor modelo ha sido Gradient Boosting. Era de esperar ya que este tipo de modelos tiene un mejor rendimiento predictivo y puede manejar diferentes tipos de datos y relaciones complejas. Aunque su tiempo de entrenamiento ha sido superior al modelo de regresión logística, se apuesta por el modelo generado por Gradient Boosting.

● **Teniendo en cuenta lo aprendido ¿Qué cosas se harían igual y cuales se harían de otra forma? ¿Por qué?**

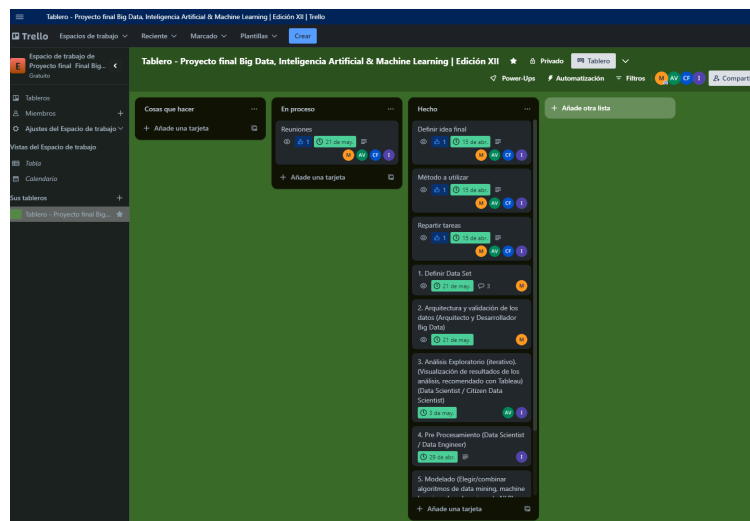
En la parte de la Arquitectura, seguiríamos manteniendo la arquitectura DAaaS, pero se añadirían más piezas propias del Google Cloud, mediante instancias internas, por no consumir recursos de externos como Tableau y Google Colab.

● Información obtenida del data set

Se ha utilizado dataset de terceros para poder entrenar nuestro sistema, en base a nuestro objetivo: "Detectar mensajes de odio".

● Trello

En nuestro proyecto, hemos empleado la plataforma Trello como una herramienta clave para la gestión y organización de tareas. Trello nos ha permitido visualizar el flujo de trabajo de manera clara y colaborativa, lo que ha facilitado la asignación de tareas, el seguimiento del progreso y la coordinación entre los miembros del equipo.



● Documentación

<https://github.com/PFBKCXII/petTeam>