UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA MECÁNICA ELÉCTRICA, ELECTRÓNICA Y SISTEMAS

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

MODELO NAIVE BAYES MULTINOMIAL PARA LA CLASIFICACIÓN DE COMENTARIOS SPAM EN ESPAÑOL DE VIDEOS SOBRE FINANZAS DE LA PLATAFORMA YOUTUBE

TESIS

PRESENTADA POR:

DEYVIS MAMANI LACUTA

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO DE SISTEMAS

PUNO – PERÚ

2024

Reporte de similitud

NOMBRE DEL TRABAJO

MODELO NAIVE BAYES MULTINOMIAL PARA LA CLASIFICACIÓN DE COMENTARIOS SPAM EN ESPAÑOL DE VIDEOS SOBRE FINANZAS DE LA PLATAFORMA YOUTUBE

AUTOR

DEYVIS MAMANI LACUTA

RECUENTO DE PALABRAS

29138 Words

RECUENTO DE CARACTERES

163559 Characters

RECUENTO DE PÁGINAS

160 Pages

TAMAÑO DEL ARCHIVO

7.3MB

FECHA DE ENTREGA

Jul 24, 2024 9:48 PM GMT-5

FECHA DEL INFORME

Jul 24, 2024 9:50 PM GMT-5

6% de similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base de datos.

- 4% Base de datos de Internet
- 1% Base de datos de publicaciones
- Base de datos de Crossref
- Base de datos de contenido publicado de Crossref
- 5% Base de datos de trabajos entregados

Excluir del Reporte de Similitud

- Material bibliográfico
- Material citado
- Material citado
- Coincidencia baja (menos de 12 palabras)

V°B°

Firmado digitalmente por SOTOMAYOR ALZAMORA Guina Guadalupe FAU 20145496170 hard
Motivo: Doy V° B°
Fecha: 24.07.2024 22:34:27 -05:00

FIRMA DIGITAL

Firmado digitalmente por: CONDORI ALEJO Henry Ivan FIR 01325355 hard
Motivo: Doy V° B°
Fecha: 24/07/2024 21:57:16-0500

Resumen

DEDICATORIA

A mi madre, Victoria, por su amor incondicional, su sacrificio y su apoyo constante. Sin ti, este logro no habría sido posible.

A mis hermanas Rosa Luz, Nancy Paola y Mariela, por estar siempre a mi lado, brindándome su cariño y comprensión en cada paso de este camino.

A mi pareja, Jaqueline, por su paciencia, apoyo y motivación, siendo una fuente de fortaleza en los momentos difíciles.

A mis amigos, por su compañerismo y por los momentos de alegría que hicieron este viaje más llevadero.

Deyvis Mamani Lacuta

AGRADECIMIENTOS

Primeramente, quiero expresar mi más sincero agradecimiento a mi Escuela Profesional de Ingeniería de Sistemas, por la formación integral que recibí durante mis años de estudio.

Agradezco profundamente a los docentes de la escuela, y en especial a los siguientes:

A la D.Sc. Donia Alizandra Ruelas Acero, por su invaluable orientación y sabios consejos que han enriquecido no solo mi formación académica, sino también mi desarrollo personal y profesional.

Al Dr. Henry Iván Condori Alejo, por su dirección y orientación en la realización de este trabajo.

Quiero también reconocer el apoyo incondicional del Mtr. Víctor Hugo Bejar Gonzales, así como el vasto conocimiento del D.Sc. Adolfo Carlos Jiménez Chura, quienes han sido pilares fundamentales en mi desarrollo académico.

Finalmente, agradezco al D.Sc. Elmer Coyla Idme, por su compromiso constante con la escuela, y al M.Sc. Pedro Feder Ponce Cordero, por su genuino interés en el bienestar de los estudiantes.

A todos ustedes, gracias por su dedicación y por ser una fuente de inspiración en mi vida académica.

Deyvis Mamani Lacuta

ÍNDICE GENERAL

Pág.

CAPÍTULO II

REVISIÓN DE LITERATURA

Área: Inteligencia Artificial y Sistemas Bio-Inspirados

Tema: Machine Learning

FECHA DE SUSTENTACIÓN: 02 de agosto del 2024

ÍNDICE DE TABLAS

Pág.

ÍNDICE DE FIGURAS

Pág.

ÍNDICE DE ANEXOS

Pág.

ACRÓNIMOS

AI:	Artificial Intelligence
API:	Application Programming Interface
CRISP-DM:	Cross Industry Standard Process for Data Mining
JSON:	JavaScript Object Notation
ML:	Machine Learning
NLP:	Natural Language Processing

RESUMEN

En la actualidad, la sección de comentarios de YouTube está siendo utilizada por cibercriminales como un medio para estafar a las personas mediante la ejecución de campañas spam sobre recomendaciones de supuestos expertos en inversión. Esto ocurre con mayor frecuencia en videos con temática de finanzas, donde se pueden encontrar usuarios más interesados en el tema de las inversiones. Esta situación tiene un efecto directo sobre los creadores de contenido, ya que perjudica la experiencia de su público en la sección de comentarios y además posibilita que estos sean estafados. Es por ello que el presente trabajo tuvo como objetivo determinar en qué medida un modelo Naive Bayes Multinomial clasifica correctamente los comentarios spam en español en videos sobre finanzas de YouTube. El enfoque de la investigación es cuantitativo, de tipo experimental y con un diseño preexperimental. La muestra está conformada por más de 25,000 comentarios pertenecientes a 30 videos sobre finanzas de YouTube. La metodología empleada es la de Cross Industry Standard Process for Data Mining (CRISP-DM) que, a través de su aplicación, permitió la elaboración del conjunto de datos, y la generación y evaluación del modelo de clasificación de comentarios spam Naive Bayes Multinomial. Como principales resultados, se determinó que el modelo Naive Bayes Multinomial generado logra clasificar correctamente una cantidad correspondiente al 98% de comentarios spam en español en videos sobre finanzas de la plataforma YouTube, lo cual indica que el modelo presenta un rendimiento adecuado en la identificación de comentarios spam, esto en función de la métrica de evaluación Recall.

Palabras Clave: Comentarios spam, CRISP-DM, Naive Bayes Multinomial, NLP, Plataforma YouTube.

ABSTRACT

Currently, YouTube's comment section is being used by cybercriminals as a means to scam people through spam campaigns featuring recommendations from supposed investment experts. This occurs more frequently in videos with financial themes, where users are more interested in investment topics. This situation has a direct effect on content creators, as it harms their audience's experience in the comment section and also makes it possible for them to be scammed. Therefore, the present work aimed to determine the extent to which a Multinomial Naive Bayes model correctly classifies spam comments in Spanish on YouTube finance videos. The research approach is quantitative, experimental, and with a pre-experimental design. The sample consists of more than 25,000 comments from 30 finance-related YouTube videos. The methodology used is the Cross Industry Standard Process for Data Mining (CRISP-DM), which, through its application, allowed for the creation of the dataset and the generation and evaluation of the Multinomial Naive Bayes spam comment classification model. As main results, it was determined that the generated Multinomial Naive Bayes model correctly classifies 98% of spam comments in Spanish on YouTube finance videos, indicating that the model performs adequately in identifying spam comments, based on the Recall evaluation metric.

Keywords: Spam comments, CRISP-DM, Multinomial Naive Bayes, NLP, YouTube Platform.

CAPÍTULO I

INTRODUCCIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA

YouTube es una de las plataformas con el mayor número de usuarios a nivel mundial, teniendo aproximadamente 1.5 billones de usuarios por mes, también se conoce que las personas pasan viendo videos de YouTube en promedio 1 hora al día, esto solo considerando a los usuarios de dispositivos móviles (Abdullah et al., 2018).

En el año 2015 YouTube adoptó un sistema de monetización, con el objetivo de recompensar a los creadores de contenido por su trabajo y además incentivar el uso de la plataforma (Alberto et al., 2015). Para lograr monetizar un canal es necesario contar con más de 1,000 suscriptores y 4,00 horas de visualización en los últimos 12 meses, esto generó en su momento, que muchos canales pequeños realicen comentarios spam publicitandose en videos con un gran número de visualizaciones (Oh, 2021).

En la actualidad esta situación se ha convertido en un problema crítico, debido a que cibercriminales vieron en la sección de comentarios de YouTube un medio por el cual era posible entrar en contacto con potenciales víctimas, mediante la aplicación de técnicas cada vez más avanzadas como los bots para la ejecución de campañas spam a gran escala (Aiyar & Shetty, 2018), con el principal objetivo de estafar a las personas, tratando en muchos casos de hacerse pasar por el creador del video en cuestión, haciendo uso de su nombre de usuario y foto de perfil, esto para aumentar sus posibilidades (Vincent, 2022).

Estas campañas de spam se manifiestan con mayor frecuencia en videos con una temática de “finanzas”, ya que es en este tipo de contenido donde se pueden encontrar a

personas más interesadas y dispuestas en invertir o aumentar sus ingresos (Nesbo, 2021). La ejecución de estas compañas consiste en la publicación de una gran cantidad de comentarios spam recomendando a personas "expertas" en inversiones y finanzas haciendo uso de diferentes cuentas de usuarios con perfiles falsos, incitando a los usuarios a contactar con estos "expertos" por medios externos como WhatsApp, Instagram y Telegram, en donde finalmente la estafa será consumada (Brown, 2022).

Los comentarios spam tienen un efecto directo sobre los creadores de contenido, ya que su existencia perjudica la experiencia de usuario de su público y seguidores, viéndose obligados en algunos casos a deshabilitar la sección de comentarios (Alias et al., 2019). Pero al realizar esta acción el creador de contenido pierde información valiosa sobre el video compartido, como las críticas constructivas o ideas para próximos videos por parte de su público (Abdullah et al., 2018).

El proceso de eliminación de comentarios spam es una tarea laboriosa, que incluye pasos como la búsqueda, identificación y eliminación de los comentarios, esto puede tomarle al creador de contenido bastante trabajo y tiempo, ello principalmente en función del número de comentarios y visualizaciones, estamos hablando en promedio de veinte minutos al día por video, con un número 200,000 visualizaciones (ThioJoe, 2021).

A pesar que la plataforma YouTube tiene su propio sistema para el filtrado de spam, este parece no ser muy eficiente (Oh, 2021), ya que aún existen una gran cantidad de comentarios spam con el que los usuarios deben de lidiar día tras día. Gandra (2014) como se citó en Samsudin et al. (2019) afirma que en promedio uno de cada cien comentarios son spam. Es por ello que en el presente trabajo se busca generar

un modelo Naive Bayes Multinomial para la clasificación de comentarios spam en español de videos sobre finanzas de la plataforma YouTube.

1.2. FORMULACIÓN DEL PROBLEMA

¿En qué medida un modelo Naive Bayes Multinomial clasificará correctamente los comentarios spam en español en videos sobre finanzas de la plataforma YouTube?

1.3. JUSTIFICACIÓN DEL PROBLEMA

En la plataforma YouTube la sección de comentarios es una de las funcionalidades más importantes para los creadores de contenido, ya que se trata del medio que posibilita la interacción con el público (Abdullah et al., 2018), la presencia de comentarios spam genera una mala experiencia de usuario y además posibilita a que las personas sean estafadas , esto ocasiona que los creadores de contenido necesiten revisar los comentarios y realicen la eliminación del spam de forma ineficiente (ThioJoe, 2021), en otros casos viéndose obligados a deshabilitar la sección de comentarios, situación aún más desfavorable para el creador de contenido, ya que de esa forma se deshace del medio principal que tiene para recibir la retroalimentación sobre el video por parte del público (Alias et al., 2019).

Como aporte práctico, el presente trabajo busca la creación de un modelo Naive Bayes Multinomial enfocado en la web para la clasificación de comentarios spam en español en videos sobre finanzas de la plataforma YouTube, el cual en un futuro pueda ser incorporado de forma directa en una herramienta web sin la necesidad de hacer uso de servicios de terceros, como las VPS dedicadas.

Por el lado teórico, tenemos el aporte de un dataset de actualidad de comentarios spam en español de videos sobre finanzas de la plataforma YouTube, etiquetado

manualmente, que contará con más de 20,000 registros, y surgen a partir de treinta videos sobre finanzas seleccionados dentro de la plataforma YouTube publicados entre los años 2022 y 2023, este conjunto de datos podrá ser utilizado en estudios futuros.

1.4. OBJETIVOS DE LA INVESTIGACIÓN

1.4.1. Objetivo General

Determinar en qué medida un modelo Naive Bayes Multinomial clasifica correctamente los comentarios spam en español en videos sobre finanzas de la plataforma YouTube.

1.4.2. Objetivos Específicos

- Definir un conjunto de reglas para la determinación de comentarios spam dentro de videos sobre finanzas de la plataforma YouTube.
- Elaborar un dataset de comentarios spam en español de videos sobre finanzas de la plataforma YouTube.
- Generar un modelo de clasificación de comentarios spam haciendo uso del algoritmo Naive Bayes Multinomial.
- Evaluar el rendimiento del modelo de clasificación Naive Bayes Multinomial generado.

1.5. HIPÓTESIS DE LA INVESTIGACIÓN

1.5.1. Hipótesis General

El modelo Naive Bayes Multinomial generado clasifica correctamente una cantidad significativa de comentarios spam en español en videos sobre finanzas de la plataforma YouTube.

1.6. VARIABLES

1.6.1. Variable Independiente

- Modelo Naive Bayes Multinomial.

1.6.2. Variable Dependiente

- Comentarios spam en español en videos sobre finanzas de la plataforma YouTube.

1.7. OPERACIONALIZACIÓN DE VARIABLES

Tabla 1

Operacionalización de Variables

TIPO	VARIABLE	DEFINICIÓN	DIMENSIÓN	INDICADOR	ESCALA
Independiente	Modelo Naive Bayes Multinomial	Modelo generado a través del algoritmo Naive Bayes Multinomial, que permite clasificar comentarios en dos clases: spam o no spam.	Estado	Sin modelo (0) Con modelo (1)	Dicotómica (0/1)
Dependiente	Comentarios spam en español en videos sobre finanzas de la plataforma YouTube.	Comentarios spam en español presentes en videos con una categoría de finanzas dentro de la plataforma YouTube.	Cantidad de comentarios spam	Número de comentarios spam	Razón (núm.)

Nota: Elaboración propia.

CAPÍTULO II

REVISIÓN DE LITERATURA

2.1. ANTECEDENTES

2.1.1. Antecedentes Nacionales

Torres et al. (2022) proponen el desarrollo de un etiquetador de videos de YouTube considerando 3 clases: video útil, inútil y neutro, a partir de clasificar los comentarios del video en cuestión, para ello emplean el modelo de aprendizaje automático BERT, que es utilizado ampliamente en el procesamiento del lenguaje natural, a mismo hacen uso de un dataset público sobre reseñas de aplicaciones de la Google Play Store, que cuenta con más de 12,000 registros de reseñas hechas por usuarios sobre más de 82 billones de aplicaciones que corresponden a una clasificación basada en números del 1 al 5 (menor a mejor puntaje), el mismo que consta de las siguientes características: reviewID, UserName, UserImage, Content, thumbsUpCount, at, replyContent y repliedAt. El modelo bidireccional de aprendizaje BERT es utilizado como un clasificador que pueda determinar a cuál clase (video útil, inútil, neutro) corresponde un comentario, para que de esta forma se pueda realizar la clasificación del video en cuestión, además se hace uso de la YouTube Data API para la obtención de los comentarios de los videos de YouTube, en el estudio se determina que el modelo BERT aplicado en esta tarea de clasificación, presenta mejores resultados en comparación a técnicas como Term Frequency-Inverse Document Frequency (TFIDF) y vectores de documentos (Doc2vec), alcanzando una precisión del 73%.

Torres & Cárdenas (2021) realizan un análisis comparativo de métodos de machine learning para clasificar opiniones sobre el servicio de restaurantes peruanos en Facebook, considerando cuatro modelos de clasificación: Naive Bayes, Random Forest y Máquinas de Soporte Vectorial con kernel lineal y RBF, para lo cual elaboran dos datasets (primario y secundario) de comentarios que pertenecen a publicaciones de las cadenas de restaurantes Roky's, Pardos, Don Tito y Norky's, el dataset primario consta de un total de 1494 comentarios, y el dataset secundario de 100 comentarios, se realiza el preprocesamiento aplicando stemming, eliminando los signos de puntuación y tokenizando la oración en unigramas, así mismo se realiza la vectorización aplicando el vectorizador TF-IDF, esto con la finalidad de que los modelos de machine learning puedan recibir a los ejemplares ya procesados como entrada. Del dataset primario se emplea el 75% y 25% para la parte del entrenamiento y testing respectivamente, se utiliza la técnica de validación cruzada en la parte de entrenamiento para cada uno de los modelos con un valor de kfold igual a 10, como resultados se obtuvo que los modelos de Máquina de Soporte Vectorial (SVM) lineal y RBF obtuvieron la mayor exactitud tanto para la parte de testing del dataset primario y del dataset secundario con 91% y 94% respectivamente, por otro lado el modelo Naive Bayes también alcanzó un valor elevado de Accuracy de 90% para la parte de testing del dataset primario y 93% con el dataset secundario. Además, se llega a determinar que la existencia de falsos positivos y negativos ocurren debido al enfoque de independencia de características con la cual trabajan estos algoritmos de machine learning, en donde el contexto de una oración queda en segundo plano.

Tristán Gomez (2019) propone un modelo predictivo del índice Net Promoter Score (NPS) basado en información textual de percepción del servicio al cliente de una administradora de fondo de pensiones (AFP), considerando las clases: promotor, neutro y detractor, para lo cual hace uso de los algoritmos de clasificación de Maquinas de Soporte Vectorial (SVM) y Naive Bayes, el dataset se encuentra conformado por los comentarios de los clientes que son transcritos a partir de encuestas telefónicas y su clase correspondiente (promotor, neutro, detractor), dentro de los cuales se tienen 18,466 comentarios con clase "promotor", 8132 comentarios con clase "neutro" y 5005 comentarios con clase "detractor", el preprocesamiento del dataset consiste en la tokenización, lematización, y eliminación de Stop Words, para la parte de entrenamiento se realiza la representación vectorial del dataset haciendo uso del estadístico tf-idf, posteriormente se divide el dataset en 80% y 20% para la parte de entrenamiento y pruebas respectivamente, una aclaración importante se presenta en la parte del entrenamiento, en donde no se hace uso del indicador Accuracy debido a la naturaleza del conjunto de datos (desbalanceados), y en su lugar se utiliza Precision, Recall y F1 Score, asi mismo se hace uso de la técnica k-fold cross validation para la obtención de mejores resultados en cuanto a la medición de la calidad con una valor de k igual a 10, se realiza el entrenamiento de los modelos de clasificación, en donde el modelo Naive Bayes obtiene un F1 Score de 34%, 57% y 86% para las clases "detractor", "neutro" y "promotor" respectivamente, mientras que el modelo SVM obtiene valores de F1 Score de 14%, 58% y 84% respectivamente para las mismas clases, por otro lado, entre los aspectos que afectan la obtención de un mejor resultado se identificaron los problemas de redacción, la calidad de la data, la ortografía, la semántica e inclusive la existen

de comentarios repetidos, otro factor está relacionado con el número de clases con el que se trabaja, en este caso 3, indicando que se podrían obtener mejores resultados si este número es reducido a sólo dos clases “promotor” y “no promotor”.

Jimenez Palomino (2018) realiza la implementación de un servicio web para la clasificación de comentarios con contenido textual agresivo con la finalidad de contrarrestar el Cyberbulling, para esto hace uso de la metodología KDD (Knowledge Discovery in Databases), la herramienta Weka y la metodología en cascada para el desarrollo del servicio web. En el estudio se trabaja con tres conjuntos de datos (que representan tres escenarios) con comentarios con un contexto peruano extraídos de las redes sociales Facebook, Twitter y YouTube, estos datasets comparten las siguientes cuatro características (categorías nominales): Muy agresivo, Agresivo, Poco agresivo y Neutro, los mismos que difieren en la cantidad de registros, teniendo 800, 1200 y 1561 comentarios en los datasets respectivamente. Seguido de ello se realizó la limpieza manual de cada uno de los comentarios y su preparación para poder ser utilizados dentro de la herramienta de minería de datos Weka, en el software Weka se seleccionaron los algoritmos Naive Bayes, BayesNet, DecisiónStump, J48, OneR, Part, SMO, IBK y KStart, posterior a ello se realiza una comparativa del desempeño y rendimiento proporcionados por los algoritmos, en donde se observa que el algoritmo que obtiene mejores resultados en los tres casos es el de IBK, con una precisión del 89% en relación al tercer dataset (1561 comentarios), el cual luego es integrado en un servicio web desarrollado mediante la metodología en cascada, para que pueda ser utilizado por otros personas. En relación al rendimiento de los otros modelos de clasificación, se

puede apreciar que el algoritmo con el menor tiempo de generación del modelo corresponde al de Naive Bayes, demorando seis segundos para el caso tres (dataset con el mayor número de comentarios), así mismo este viene a ser el que presenta el segundo menor valor de error absoluto medio para los tres datasets solo por detrás de IBK, así mismo el algoritmo Naive Bayes obtenido un valor por encima del 70% respecto a la métrica de evaluación precisión en los tres casos.

Cuzcano Chavez & Ayma Quirita (2020) comparan cuatro modelos de clasificación para la detección del Cyberbulling dentro de contenido textual en publicaciones escritas en el idioma español peruano dentro de la plataforma Twitter, para lo cual emplean las técnicas de Machine Learning (algoritmos de clasificación) y el Procesamiento del Lenguaje Natural (preprocesamiento y extracción de características), el conjunto de datos consiste en 10,096 tweets en español que fueron distribuidos en las siguientes cuatro clases: no acoso (5,122), acoso directo (2127), incitación al odio (1,000) y acoso sexual (1,847), esta categorización fue llevada a cabo por personas relacionadas al área, como psicólogos y comunicadores. Con el dataset ya elaborado se procedió a realizar un preprocesamiento corriente, pero en la parte de extracción de características se aplicaron dos técnicas, el análisis semántico y sintáctico, por parte del análisis semántico se realizó tanto el stemming y la lematización de forma separada, y por el lado del análisis sintáctico se utilizarón bi-gramas y tri-gramas del mismo modo de forma independiente, y a cada uno de estos se aplicó la representación numérica de TF-IDF, la separación del conjunto de datos corresponde a un 70% y 30% para el training y testing respectivamente, los algoritmos seleccionados para la comparación son: Naive Bayes (NB), Multinomial Logistic Regression

(MLR), Random Forest (RF) y Support Vector Machine (SVM), la comparación fue realiza en dos escenarios, el primero con un dataset balanceado y el segundo con un datasets desbalanceado, llegándose a determinar que el algoritmo que obtuvo los resultados en ambos escenarios corresponde al SVM con un valor de exactitud promedio por encima del 80%, lo siguiente que se puede mencionar es en relación a los esquemas de extracción de características, en donde se observó que con la extracción de características aplicando el análisis o esquema semántico se obtenían mejores resultados y dentro de esta el stemming proporciona mejores resultados que la lematización, considerando las distribuciones del dataset, se pudo observar que el algoritmo Naive Bayes muestra un mejor desempeño cuando trabaja con conjuntos de datos balanceados.

2.1.2. Antecedentes Internacionales

Samsudin et al. (2019) proponen un framework para la detección de spam en YouTube que consiste en 5 fases; colección de datos, preprocesamiento, selección y extracción de características, clasificación y detección. Para lo cual hacen uso del conjunto de datos de recopilación de spam de YouTube del Repositorio de Machine Learning UCI, que consta de 1005 y 951 comentarios spam y no spam respectivamente, extraídos de 5 videos. A partir de ello se realizaron 3 tipos de experimentos que consisten en la consideración de un número distinto de características, siendo 7, 8 y 13 características para cada caso, posteriormente realizaron la clasificación en donde se emplearon las técnicas de Naive Bayes y Regresión logística de las herramienta de minería de datos Weka y RapidMiner, obteniendo como resultados un Accuracy del 87,21% y 85,7% para Naive Bayes y Regresión logística respectivamente, llegándose a

determinar que mientras más características sean consideradas en este framework, mayor podrá ser la exactitud alcanzada.

Sinhal & Maheshwari (2022) realizan una revisión de varias técnicas de clasificación de Machine Learning como Naïve Bayes, Random Forest, Máquinas de Soporte Vectorial, entre otros, para determinar si un comentario es spam o no dentro de YouTube, además muestran diferentes técnicas y formas de llevar a cabo el preprocesamiento de texto, la extracción de características y técnicas para la clasificación de spam, para la revisión consideran 17 estudios, a partir de los cuales se determina que las técnicas de clasificación Naive Bayes y Support Vector Machine son las que se utilizan con mayor frecuencia, habiendo sido incluidas ambas en 12 estudios del total.

Oh (2021) plantea un esquema de detección de comentarios spam de YouTube utilizando el modelo de aprendizaje automático de conjuntos en cascada, para lo cual emplea 6 técnicas de clasificación de machine learning: Árboles de decisión, Regresión logística, Bernoulli Naïve Bayes, Random forest, Support Vector Machine con kernel lineal y gaussiano, y dos modelos de conjunto: Ensemble con hard voting y Ensemble con soft voting. Todas estas técnicas son utilizadas en el dataset de comentarios de 5 videos populares de música; Psy, Katy Perry, LMFAO, Eminem y Shakira, este dataset es de acceso libre proporcionado por el repositorio de machine learning UCI, que contiene un total de 1,005 comentarios spam y 978 no spam, de los cuales el 70% fue empleado para el entrenamiento y el 30% para las pruebas, para el preprocesamiento del data se empleó la función CountVectorizer y la técnica de bolsa de palabras (BoW), en relación a la metodología propuesta sobre conjuntos, el modelo ESM-H realiza la clasificación a partir del resultado de las

otras técnicas de clasificación consideradas, es decir, tomará el resultado de la clase que haya sido predicha por la mayoría de las otras técnicas, algo similar ocurre para el modelo ESM-S, en este caso se realiza un promedio de las probabilidades de las predicciones de clase realizadas por cada una de las otras técnicas de clasificación. Para la evaluación se hace uso 5 indicadores: Accuracy rate, Spam caught rate, Blocked ham rate, F1-score, y el MCC (coeficiente de correlación de Matthews), en donde el modelo por conjuntos ESM-S obtiene el mejor resultado en la mayoría de indicadores, con un valor sobre el 90% en mayor parte, por otro lado, Bernoulli Naïve Bayes viene a ser la técnica que obtiene el menor índice de Blocked ham rate, lo cual demuestra que esta técnica de clasificación tiene el valor más bajo de falsos positivos, aspecto útil cuando se trata de tareas de clasificación de spam.

Ifriza & Sam'an (2021) realizan una comparación del rendimiento de los clasificadores Support Vector Machine y Gaussian Naive Bayes en la detección de comentarios spam de YouTube, para los cual hacen uso de un dataset de acceso libre de comentarios spam de un video musical de Shakira, proporcionado por el repositorio de Machine Learning UCI, en la parte de evaluación de los clasificadores se emplea el cross-approval y el método k-fold, procedimiento que consiste en separar el dataset en cinco secciones iguales, dentro de las cuales una parte es empleada para la prueba y las demás son usadas como conjunto de entrenamiento en cada overlay, de las cuales posteriormente se saca un promedio de los resultados obtenidos en cada overlay, como parte de los resultados se obtiene que el modelo Support Vector Machine presenta un mejor rendimiento alcanzando un accuracy del 91%, por otro lado, el modelo Gaussian Naive Bayes obtuvo una exactitud del 86%, así mismo se observa que

el modelo Gaussian Naive Bayes presenta un Blocked Ham Rate mejor que el clasificador Support Vector Machine.

Narváez-Albuja (2022) propone el desarrollo de una aplicación que permita reportar y eliminar comentarios spam de YouTube automáticamente, haciendo uso de un clasificador Naive Bayes Multinomial, para ello elabora un dataset empleando la YouTube Data API y un script de Python, logrando recolectar 255,109 comentarios en total, a partir de ello realiza el etiquetado de forma manual, obteniendo 28,804 comentarios spam, cada registro tiene el nombre del usuario, el comentario y la clase (1 spam y 0 no spam), algo diferente a un enteramiento común se realiza en este estudio, ya que se considera entrenar 2 modelos, uno para los nombres de usuario y otro para los comentarios, en donde un comentario será etiquetado como spam siempre y cuando al menos uno de los dos clasificadores lo etiquete como tal, en la evaluación de los modelos entrenados de obtiene que el clasificador de spam por comentarios obtiene un accuracy del 99% y el clasificador de spam por nombres de usuario obtiene un accuracy del 100%, asi mismo se realiza una comparación de rendimiento con la programa de código abierto YT Spammer Purge que tiene un propósito similar; la identificación y eliminación de comentarios spam, y que a diferencia del modelo planteado, esta emplea filtros de patrones de texto para la clasificación de los comentarios, en donde se llega a determinar que el modelo de clasificación Naive Bayes Multinomial propuesto realiza una identificación ligeramente mejor de comentarios spam que el YT Spammer Purge.

2.2. MARCO TEÓRICO

2.2.1. YouTube

YouTube es una de las plataformas de redes sociales de mayor alcance, considerada el segundo motor de búsqueda más grande del mundo, YouTube permite compartir videos, y cuenta con más del 80% de usuarios localizados fuera de los Estados Unidos. Así mismo cada día aproximadamente 1 millón de horas de video son reproducidas. Por otra parte, YouTube es considerada en muchos estudios como una plataforma, una librería, un laboratorio y un narrador de la era digital (Djerf-Pierre et al., 2019).

Cho & Suh (2021) definen YouTube como una plataforma en la cual los creadores de contenido pueden interactuar con los espectadores en tiempo real, YouTube permite que los espectadores puedan valorar un video mediante los botones de “like” o “dislike”, como también el poder comunicarse con otros espectadores, por otro lado, en YouTube los creadores de contenido pueden recibir retroalimentación de sus espectadores sobre un video en la sección de comentarios.

2.2.1.1. La categoría Finanzas en la plataforma YouTube

Dentro de la plataforma YouTube no existe como tal una categoría con el nombre de “Finanzas”, la más cercana vendría a ser la categoría de “Educación”, por lo mismo que al mencionar la expresión videos sobre finanzas, se busca hacer referencia a los videos cuya temática o público objetivo sean personas que buscan conocimientos financieros, estrategias de inversión o maneras para la administración adecuada del dinero (Nguyen, 2023).

2.2.1.2. Los comentarios Spam en videos sobre Finanzas

Aiyar & Shetty (2018) indican que los comentarios spam vine a ser un factor importante que influye en la impresión del usuario respecto al video, ya que muchos usuarios inclusive antes de reproducir el video van directamente a la sección de comentarios. Los comentarios spam por lo general son contenidos e información irrelevante en relación al video, que en la mayoría de casos es generada por bots que se hacen pasar por personas reales (Vincent, 2022).

Los comentarios spam dentro de videos que tratan sobre temas relacionados a las finanzas, vienen a ser aquellos que buscan en su mayoría estafar a los usuarios. Se trata de comentarios recomendando a personas "expertas" en inversiones y finanzas haciendo uso de diferentes cuentas de usuarios con perfiles falsos, incitando a los usuarios a contactar con estos "expertos" por medios externos como WhatsApp, Instagram y Telegram, en donde finalmente la estafa será consumada (Brown, 2022; Nesbo, 2021).

2.2.2. YouTube Data API

La YouTube Data API v3 permite la incorporación de funcionalidades que normalmente se encuentran presentes dentro de la plataforma YouTube en otros sitios web o entornos, esta trabaja con una gran variedad de "recursos" tales como: Channels, Comments, y otros, por otro lado, la API (Application Programming Interface) soporta métodos de inserción, recuperación, actualización y eliminación para la mayoría de estos recursos (*API Reference | YouTube Data API*, s. f.).

2.2.2.1. Requerimientos

Los requerimientos básicos para poder hacer uso de la API son los siguientes:

- Se debe de contar con una API Key que se genera dentro de la consola de Google o proporcionar un token OAuth 2.0.
- Se debe de enviar un token de autorización en peticiones de inserción, actualización y eliminación, así mismo cuando se desea obtener información privada del usuario autentificado.

2.2.2.2. Tipos de Recursos

En la siguiente tabla se proporciona una descripción de todos los recursos con los que se pueden interactuar al hacer uso de la YouTube Data API.

Tabla 2

Tipos de Recursos YouTube Data API

RECURSO	DESCRIPCIÓN
activity	Contiene información sobre una acción que un usuario en particular ha realizado en el sitio de YouTube. Las acciones del usuario que se informan en las fuentes de actividad incluyen calificar un video, compartir un video, marcar un video como favorito y publicar un boletín de canal, entre otras.
caption	Representa una pista de subtítulos de YouTube
channelBanner	Identifica la URL que se utilizará para configurar una imagen recién cargada como imagen de banner para un canal.
channelSection	Contiene información sobre un conjunto de vídeos que un canal ha elegido presentar. Por ejemplo, una sección podría incluir las últimas cargas de un canal, las cargas más populares o videos de una o más listas de reproducción.
channel	Contiene información sobre un único canal de YouTube.

RECURSO	DESCRIPCIÓN
commentThread	Contiene información sobre un hilo de comentarios de YouTube, que comprende un comentario de nivel superior y respuestas, si existen, a ese comentario
comment	Contiene información sobre un único comentario de YouTube. Un recurso comment puede representar un comentario sobre un vídeo o un canal. Además, el comentario podría ser un comentario de nivel superior o una respuesta a un comentario de nivel superior.
guideCategory	Identifica una categoría que YouTube asocia con canales según su contenido u otros indicadores, como la popularidad. Las categorías de guía buscan organizar los canales de una manera que facilite a los usuarios de YouTube encontrar el contenido que buscan. Si bien los canales pueden estar asociados con una o más categorías de guías, no se garantiza que estén en ninguna categoría de guías.
i18nLanguage	Identifica un idioma de aplicación compatible con el sitio web de YouTube. El idioma de la aplicación también puede denominarse idioma de la interfaz de usuario.
i18nRegion	Identifica un área geográfica que un usuario de YouTube puede seleccionar como región de contenido preferida. La región de contenido también puede denominarse ubicación de contenido.
member	Representa a un miembro de un canal de YouTube.
membershipsLevel	Identifica un nivel de precio para el creador que autorizó la solicitud de API.
playlistItem	Identifica un recurso, como un vídeo, que forma parte de una lista de reproducción. El recurso playlistItem también contiene detalles que explican cómo se utiliza el recurso incluido en la lista de reproducción.
playlist	Representa una única lista de reproducción de YouTube. Una lista de reproducción es una colección de videos que se pueden ver secuencialmente y compartir con otros usuarios.
search result	Contiene información sobre un vídeo, canal o lista de reproducción de YouTube que coincide con los parámetros de búsqueda especificados en una solicitud de API. Si bien un resultado de búsqueda apunta a un recurso identificable de forma única, como un vídeo, no tiene sus propios datos persistentes.
subscription	Contiene información sobre la suscripción de un usuario de YouTube. Una suscripción notifica a un usuario cuando se agregan nuevos videos a un canal o cuando otro usuario realiza una de varias acciones en YouTube, como cargar un video, calificar un video o comentar un video.
thumbnail	Identifica imágenes en miniatura asociadas con un recurso.
videoAbuseReportReason	Contiene información sobre el motivo por el cual un vídeo

RECURSO	DESCRIPCIÓN
	se marcaría por contener contenido abusivo
videoCategory	Identifica una categoría que ha estado o podría estar asociada con videos subidos.
video	Representa un único vídeo de YouTube.
watermark	Identifica una imagen que se muestra durante la reproducción de los videos de un canal específico

Nota: (*API Reference | YouTube Data API*, s. f.)

2.2.2.3. Tipos de Operaciones

En la siguiente tabla se proporciona una descripción de cuales viene a ser las operaciones o peticiones más comunes que admite la API, así mismo es importante mencionar que algunos recursos pueden admitir métodos más específicos.

Tabla 3

Tipos de Operaciones YouTube Data API

OPERACIONES	DESCRIPCIÓN
list	Recupera (GET) una lista de cero o más recursos.
insert	Crea (POST) un nuevo recurso.
update	Modifica (PUT) un recurso existente para reflejar los datos de su solicitud.
delete	Elimina (DELETE) un recurso específico.

Nota: (*API Reference | YouTube Data API*, s. f.)

2.2.2.4. Operaciones Soportadas

A continuación, se muestra la información sobre los tipos de consultas que son soportados para los diferentes tipos de recursos, cabe aclarar que las operaciones que inserten, actualicen o eliminen recursos requieren necesariamente la autorización del usuario.

Tabla 4

Operaciones Soportadas por Recurso

RECURSO	OPERACIÓN			
	LIST	INSERT	UPDATE	DELETE
activity	✓	✓	✗	✗
caption	✓	✓	✓	✓
channel	✓	✗	✗	✗
channelBanner	✗	✓	✗	✗
channelSection	✓	✓	✓	✓
comment	✓	✓	✓	✓
commentThread	✓	✓	✓	✗
guideCategory	✓	✗	✗	✗
i18nLanguage	✓	✗	✗	✗
i18nRegion	✓	✗	✗	✗
playlist	✓	✓	✓	✓
playlistItem	✓	✓	✓	✓
search result	✓	✗	✗	✗
subscription	✓	✗	✗	✗
thumbnail	✗	✗	✗	✗
video	✓	✓	✓	✓
videoCategory	✓	✗	✗	✗
watermark	✗	✗	✗	✗

Nota: (*API Reference | YouTube Data API*, s. f.)

2.2.2.5. Cuotas

La YouTube Data API utiliza una cuota, por lo mismo que todas las solicitudes de API generan como mínimo un costo de cuota de un punto (incluidas las solicitudes no válidas), los proyectos que habilitan la API desde el Google API Console tienen asignados una cuota de 10,000

unidades por día, el uso de cuota se puede observar dentro de la consola de la API.

A continuación, se muestra una tabla con el costo de cuota que genera el llamar a los métodos de los principales recursos de la YouTube Data API.

Tabla 5

Costo de Cuota por tipo de Método

RECURSO	MÉTODO	COSTO
channels	list	1
	update	50
comments	list	1
	insert	50
	update	50
	markAsSpam	50
	setModerationStatus	50
	delete	50
commentThreads	list	1
	insert	50
	update	50
videos	list	1
	insert	1,600
	update	50
	rate	50
	getRating	1
	reportAbuse	50
	delete	50

Nota: (*API Reference | YouTube Data API*, s. f.)

Si bien el aspecto de la cuota puede representar un factor limitante en la creación de aplicativos, es importante mencionar que dentro de Google API Console se pueden crear varios proyectos, y debido a que cada proyecto maneja su cuota que se le es asignado a diario de forma independiente, es posible contrarrestar la limitante de la cuota mediante la creación de varios proyectos a la vez.

2.2.3. Procesamiento del Lenguaje Natural

El procesamiento de lenguaje natural, o por sus siglas en inglés NLP se encarga de analizar data textual haciendo uso de herramientas estadísticas y de algoritmos de machine learning, con ello se busca extraer características como sentimientos, partes de una oración, y resúmenes de datos, entre otros. También se puede entender al procesamiento del lenguaje natural como un proceso computacional que busca comprender el lenguaje natural de los humanos, las principales aplicaciones del NLP son la clasificación y categorización de texto, el cual puede ser dividido en distintas áreas como la opinión de usuarios, los filtros de spam, etc (Vangara & Vangara, 2020).

Por su parte Prakash (2019) indica que el Procesamiento del Lenguaje Natural es un subcampo de la Inteligencia Artificial que hace uso de algoritmos para representar y procesar automáticamente varias formas del lenguaje humano (natural), y que además recientemente se ha hecho uso de algoritmos de Machine Learning para procesar el lenguaje natural mediante el estudio de millones de ejemplares de texto escritos por humanos, en este proceso los modelos entrenados obtienen un entendimiento del contexto del habla humana, escritura y otras formas de comunicación.

Los términos más utilizados en el procesamiento de lenguaje natural son (Madnani, 2007):

- Token: Unidades lingüísticas, como palabras, números y alfanuméricos.
- Tokenización: Proceso que consiste en separar una oración en sus componentes tokens.
- Corpus: Un cuerpo de texto, que por lo general contiene una gran cantidad de oraciones.
- Part-of-speech (POS) Tag: Símbolo para una categoría lexical, como NN(Nombre), VB(Verbo), JJ(Adjetivo), AT(Articulo).
- Parse Tree: Describe la estructura sintáctica de una oración.

2.2.3.1. El Procesamiento del Lenguaje Natural en Machine Learning

Behzadi (2015) indica en relación a Machine Learning, que el procesamiento del lenguaje natural se ocupa de preprocesar el texto real, y poder convertirlo en una entrada válida para los algoritmos de Machine Learning. Los sistemas de Machine Learning son capaces de aprender automáticamente de la data, algunos métodos de Machine Learning más empleados son los siguientes:

- Clasificadores: Concerniente a tareas de clasificación de documentos.
- Modelos estructurados: Concerniente a tareas de etiquetado, análisis y extracción.
- Aprendizaje no supervisado: Concerniente a tareas de generalización e inducción de estructura.

Las aplicaciones de Machine Learning conjunto al procesamiento del lenguaje natural están en constante crecimiento, como tareas más representativas se tienen a la detección y filtrado de spam, y por otro lado a modelos que permitan realizar traducciones de texto.

2.2.3.2. Stemming

El stemming consiste en convertir una palabra en su forma base, tomando como ejemplo a las siguientes palabras: presentación, presentado y presentando, todos estos se convertirían en presentar después de aplicar stemming, el principal propósito de este método es reducir la cantidad de palabras mediante la eliminación de varios sufijos, además esto ayuda enormemente a que el modelo no cometa errores en el proceso de entrenamiento, existen una variedad de algoritmos disponibles sobre stemming, como Porter stemming, snowball stemming y Lancaster stemming (Deshpande & Kumar, 2018).

2.2.3.2.1. Porter Stemmer

Este viene a ser una forma de los algoritmos de stemming que elimina los prefijos de las palabras, el propósito general de Porter stemmer es el de mejorar el rendimiento del proceso de entrenamiento del modelo, y lo realiza a través de la eliminación de sufijos de una palabra para que quede convertida en su forma base, de esta forma, el número de términos reduce drásticamente. Porter stemmer no está basado en diccionarios, utiliza un conjunto de reglas genéricas, algunas personas ven esto como un inconveniente ya que su funcionamiento es muy sencillo, y no se ocupa de los detalles contextuales de nivel interior de las

palabras, este algoritmo es utilizado por su simplicidad y velocidad (Deshpande & Kumar, 2018).

2.2.4. Clasificador Naive Bayes Multinomial

El clasificador Naive Bayes Multinomial también conocido como el modelo NB multinomial, realiza una suposición simplificada (ingenua) sobre cómo interactúan las características (independencia mutua), esto consiste en ignorar el orden de las palabras y darle más importancia a la frecuencia de cada palabra dentro del vocabulario, para lo cual hace uso de la representación de bolsa de palabras (bag of words) (Jurafsky & Martin, 2023).

2.2.4.1. Componentes de la Regla de Bayes

2.2.4.1.1. Probabilidad Posterior

El clasificador NB Multinomial hace uso de la regla de Bayes o también conocida como "la inferencia Bayesiana", formulado por Thomas Bayes en 1763, su representación formal se presenta a continuación (Jurafsky & Martin, 2023):

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \tag{1}$$

Como se muestra, la ecuación de la regla de Bayes está compuesta por varias partes, que se pueden representar con palabras en la siguiente manera (Raschka, 2017):

$$probabilidad\ posterior = \frac{probabilidad\ condicional * probabilidad\ previa}{evidencia} \tag{2}$$

La probabilidad posterior, en el contexto de un problema de clasificación se puede interpretar de la siguiente forma: "¿Cuál es la

probabilidad de que un objeto (ejemplar) pertenezca a una clase i dado los valores de sus características observadas?". Para entender de mejor forma esta pregunta se representa la regla de Bayes con simbología relacionada a la clasificación de documentos (Raschka, 2017):

$$P(c_j|d_i) = \frac{P(d_i|c_j) * P(c_j)}{P(d_i)} \quad (3)$$

En donde:

- c_j: Notación de la clase j, $j \in \{1,2,\dots,m\}$ (m: número total de clases)
- d_i: Notación del documento i, $i \in \{1,2,\dots,n\}$ (n: número total de ejemplares)

Esta ecuación representa la probabilidad de que dado un documento d_i, este pertenezca a la clase c_j. En este punto es importante recordar que, al estar trabajando en una tarea de clasificación, se tendrá una cierta cantidad de clases m, entonces partiendo de ese punto, para determinar a qué clase pertenece (predicción) un documento d_i, lo que se realizará es determinar cuál es la mayor probabilidad posterior alcanzada para todas las m clases, esto se puede representar de la siguiente forma (Jurafsky & Martin, 2023):

$$Predicción = \underset{j=1\dots,m}{\operatorname{argmax}} P(c_j|d_i) = \underset{j=1\dots,m}{\operatorname{argmax}} \frac{P(d_i|c_j) * P(c_j)}{P(d_i)} \quad (4)$$

La ecuación anterior, ya puede ser empleada para la clasificación de nuevos ejemplares, algo que es importante mencionar es el hecho de que puede ser mejorada, principalmente para evitar el problema de

"underflow" y aumentar la velocidad de cómputo (Jurafsky & Martin, 2023).

2.2.4.1.2. Probabilidad Condicional (Likelihood)

En este punto es importante recordar la suposición de *independencia de características* del clasificador NB Multinomial. Entonces, si empezamos asumiendo que las características $f_1, f_2, \dots, f_n$ del ejemplar d_i solo representan la identidad de la palabra y no su posición (son independientes), podemos realizar la siguiente igualdad (Raschka, 2017):

$$P(d_i|c_j) = P(f_1, f_2, \dots, f_n|c_j) = P(f_1|c_j) * P(f_2|c_j) * \dots * P(f_n|c_j) = \prod_{k=1}^{n} P(f_k|c_j) \quad (5)$$

Ahora bien, en la parte final de la ecuación anterior se tienen a los likelihoods "individuales" $P(f_k|c_j)$ para cada característica del ejemplar d_i, estos se calculan a través de la estimación del likelihood máximo, cuya ecuación es la siguiente (Raschka, 2017):

$$P(f_k|c_j) = \frac{N_{f_k,c_j}}{N_{c_j}} \quad (6)$$

En donde:

- N_{f_k,c_j}: Número de veces que la característica f_k aparece en ejemplares de la clase c_j.
- N_{c_j}: Número total de características de la clase c_j.

Como se puede observar en estas dos últimas ecuaciones, al final, para calcular la probabilidad condicional, se debe de realizar la

multiplicación de cada uno de los likelihoods individuales $P(f_k|c_j)$ $(k = 1 \dots, n)$, y se sabe que estos dependen completamente del número de veces que la característica f_k aparece en ejemplares de la clase c_j (N_{f_k,c_j}) y del número total de características de la clase c_j (N_{c_j}). Aquí es importante considerar el caso en donde para alguna característica f_k el valor de N_{f_k,c_j} sea cero (la característica f_k no aparece en ningún ejemplar de la clase c_j), lo que generaría que la probabilidad condicional para la clase c_j sea cero, sin importar las demás n características ya que se trata de una multiplicación (Jurafsky & Martin, 2023).

Laplace smoothing

Con la finalidad de evitar este problema de obtener una probabilidad condicional cero, se puede hacer uso de Laplace smoothing, que consiste en agregar un suavizado de una unidad a la ecuación de los likelihoods "individuales", obteniéndose lo siguiente (Raschka, 2017):

$$P(f_k|c_j) = \frac{N_{f_k,c_j} + 1}{N_{c_j} + 1 * |V|} \tag{7}$$

En donde:

- N_{f_k,c_j}: Número de veces que la característica f_k aparece en ejemplares de la clase c_j.
- N_{c_j}: Número total de características de la clase c_j.
- V: Vocabulario (conjunto de palabras de todas las clases).

2.2.4.1.3. Probabilidad Previa

La probabilidad previa se puede interpretar como la creencia previa o el conocimiento previo, que representa "la probabilidad general de encontrar una clase particular", este mismo puede ser establecido a través de consultar a un experto en el tema o la otra forma es que puede ser estimada de la data de entrenamiento, para esto último, la data debe contener ejemplares con características independientes y además la data debe de representar a toda la población, la ecuación es la siguiente (Jurafsky & Martin, 2023; Raschka, 2017):

$$P(c_j) = \frac{N_{c_j}}{N_c} \tag{8}$$

En donde:

- N_{c_j}: Número de ejemplares de la clase c_j.
- N_c: Número total de ejemplares.

De la anterior ecuación, se pueden mencionar lo siguiente, tomando de referencia a la primera ecuación base, podemos indicar que si el número de ejemplares para cada una de las m clases es el mismo (distribución uniforme), la probabilidad previa estimada será la misma para todas las clases, y en dicho caso la probabilidad posterior será determinada completamente por la probabilidad condicional (likelihood), y la "evidencia". Así mismo, debido a que la "evidencia" (como se verá a continuación) es una constante, la "probabilidad posterior" dependerá completamente de la "probabilidad condicional" (Raschka, 2017).

2.2.4.1.4. Evidencia

La evidencia $P(d_i)$ se puede entender como la probabilidad de encontrar un documento d_i independiente de la etiqueta de la clase, esta se puede calcular de la siguiente forma (Raschka, 2017):

$$P(d_i) = P(d_i|c_j) * P(c_j) + P(d_i|c_j^C) * P(c_j^C) \quad (9)$$

En donde c_j^C representa el "complemento" y se puede interpretar en palabras como "que no sea la clase c_j". Se sabe que la "evidencia" está presente en la ecuación base y además permite calcular con exactitud la "probabilidad posterior", sin embargo, esta puede ser removida de la ecuación sin alterar el resultado final, esto es posible ya que para la predicción lo que se realiza es computar la "probabilidad posterior" $(\frac{P(d_i|c_j)*P(c_j)}{P(d_i)})$ para cada una de las m clases, pero aquí nos podemos dar cuenta que $P(d_i)$ no cambia para cada clase, como si lo hacen las demás partes de la ecuación, por lo mismo es que se puede realizar la siguiente equivalencia (Jurafsky & Martin, 2023; Raschka, 2017):

$$Predicción = \underset{j=1...,m}{\operatorname{argmax}} \frac{P(d_i|c_j) * P(c_j)}{P(d_i)} \propto \underset{j=1...,m}{\operatorname{argmax}} P(d_i|c_j) * P(c_j) \quad (10)$$

En este punto, si reemplazamos la ecuación calculada de la "probabilidad condicional" (likelihood), obtenemos lo siguiente:

$$Predicción \propto \underset{j=1...,m}{\operatorname{argmax}} P(c_j) \prod_{k=1}^{n} P(f_k|c_j) \quad (11)$$

2.2.4.2. Clasificador NB Multinomial en un espacio Logarítmico

Como podemos apreciar en la ecuación anterior, muchos likelihoods individuales $P(f_k|c_j)$ son multiplicados, y como esto será calculado por un computador, puede terminar en un subdesbordamiento del punto flotante (Jurafsky & Martin, 2023), para evitar este subdesbordamiento y además mejorar la velocidad, es recomendable realizar el cálculo de la probabilidad sumando logaritmos de probabilidades en lugar de multiplicar probabilidades. Esto no afecta en nada al resultado final, ya que la clase con la mayor puntuación de probabilidad posterior logarítmica será el resultado de la predicción (Jurafsky & Martin, 2023).

Para poder convertir la ecuación a un espacio logarítmico se hace uso de la siguiente propiedad de los logaritmos de la multiplicación: $\log(xy) = \log(x) + \log(y)$, con ello se logra obtener lo siguiente (Manning et al., 2008):

$$Predicción \propto \underset{j=1\ldots,m}{\operatorname{argmax}} \log\left(P(c_j)\prod_{k=1}^{n} P(f_k|c_j)\right) \tag{12}$$

$$Predicción \propto \underset{j=1\ldots,m}{\operatorname{argmax}}[\log\left(P(c_j)\right) + \log\left(\prod_{k=1}^{n} P(f_k|c_j)\right)] \tag{13}$$

$$Predicción \propto \underset{j=1\ldots,m}{\operatorname{argmax}}[\log\left(P(c_j)\right) + \sum_{k=1}^{n} \log\left(P(f_k|c_j)\right)] \tag{14}$$

Esta ecuación calculada viene a ser la que es utilizada en una gran mayoría de implementaciones de clasificadores de NB multinomial, y puede interpretarse de la siguiente forma, cada probabilidad condicional

(likelihood) individual logarítmica $\log\left(P\left(f_k|c_j\right)\right)$ es una medida que indica que tan buena es una característica f_k para la clase c_j. De forma similar la probabilidad previa logarítmica $\log\left(P\left(c_j\right)\right)$ es una medida que indica la frecuencia relativa de la clase c_j, Algo importante en este punto, es notar que las clases que tengan una mayor frecuencia (mayor número de ejemplares) tienen más probabilidad a ser determinadas como la clase predicha (predicción) en comparación a las clases infrecuentes (menor número de ejemplares) (Manning et al., 2008).

2.2.4.3. Algoritmo Naive Bayes Multinomial

El algoritmo Naive Bayes Multinomial viene a ser uno de los métodos de clasificación de texto más utilizados (Samsudin et al., 2019), esto principalmente por que presenta un buen rendimiento y bajo tiempo de ejecución (Jimenez Palomino, 2018), estamos hablando de una complejidad lineal, en relación a lo que le toma realizar un escaneo de los datos tanto para el training y testing (Manning et al., 2008). A continuación, en la Figura 1 se presenta el algoritmo Naive Bayes Multinomial en un espacio logarítmico.

Figura 1

Naive Bayes Multinomial (Training y Testing)

Algorithm 1 TrainMultinomialNB(C, D)

$V \leftarrow \text{ExtractVocabulary}(D)$
$N \leftarrow \text{CountDocs}(D)$
for each $c \in C$ **do**
  $N_c \leftarrow \text{CountDocsInClass}(D, c)$
  $\text{prior}[c] \leftarrow \frac{N_c}{N}$
  $\text{text}_c \leftarrow \text{ConcatenateTextOfAllDocsInClass}(D, c)$
  for each $t \in V$ **do**
    $T_{ct} \leftarrow \text{CountTokensOfTerm}(\text{text}_c, t)$
  end for
  for each $t \in V$ **do**
    $\text{condprob}[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
  end for
end for
return $V, \text{prior}, \text{condprob}$

Algorithm 2 ApplyMultinomialNB$(C, V, \text{prior}, \text{condprob}, d)$

$W \leftarrow \text{ExtractTokensFromDoc}(V, d)$
for each $c \in C$ **do**
  $\text{score}[c] \leftarrow \log \text{prior}[c]$
  for each $t \in W$ **do**
    $\text{score}[c] \mathrel{+}= \log \text{condprob}[t][c]$
  end for
end for
return $\arg\max_{c \in C} \text{score}[c]$

Nota: (Manning et al., 2008)

Es importante mencionar que el clasificador Naive Bayes presenta un índice bajo de Block Ham Rate (falsos positivos), esto quiere decir que un modelo Naive Bayes muy pocas veces clasifica como spam a un comentario que realmente no lo es, esto es muy importante cuando se trata de tareas relacionadas a la clasificación de spam (Ifriza & Sam'an, 2021; Oh, 2021)

Otro aspecto a considerar viene a ser la parte del dataset, ya que se puede observar que este algoritmo proporciona mejores resultados cuando se trabajan con datasets previamente balanceados (Cuzcano

Chavez & Ayma Quirita, 2020). Al emplear un dataset desbalanceado, la probabilidad previa (prior probability) toma un valor distinto para cada una de las clases, esto afecta directamente al resultado de predicción final, lo que genera que no se pueda construir un modelo apropiado (Tong et al., 2011).

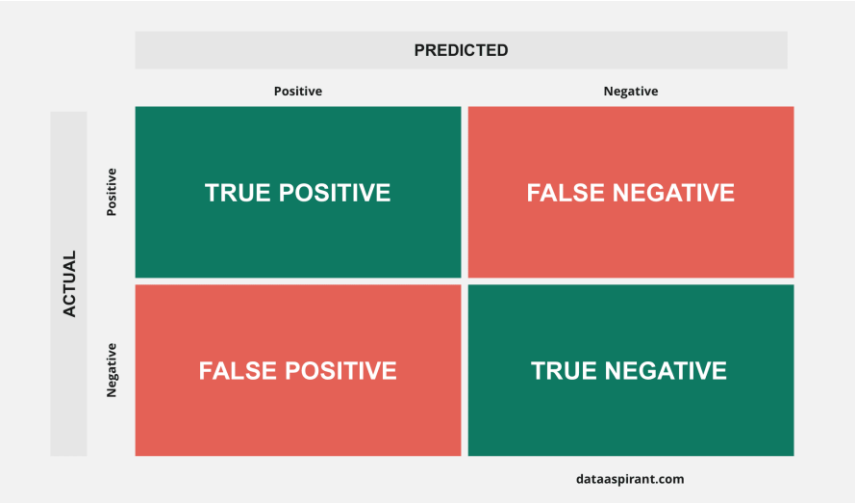
2.2.5. Matriz de Confusión

Es una matriz de números que permite observar el desempeño de un algoritmo de aprendizaje supervisado, en donde cada columna de la matriz representa el número de predicciones por cada clase, y cada fila viene a representar a las instancias en la clase real, en términos generales la matriz de confusión nos permite observar el número de aciertos y errores que está presentando un modelo a la hora de realizar el testing (Arce, 2019).

A continuación, se presenta de forma gráfica la estructura de la matriz de confusión.

Figura 2

Matriz de Confusión para Clasificación Binaria



Nota: Adaptado de Polamuri (2020)

En la Figura 2, podemos observar que la matriz de confusión está conformado por cuatro cuadrantes, cada uno de estos términos se describen a continuación (Bhandari, 2020):

- **True Positive (TP)**

  El valor real era positivo, y el modelo predijo un valor positivo.

- **True Negative (TN)**

  El valor real era negativo, y el modelo predijo un valor negativo.

- **False Positive (FP) – Error tipo I**

  El valor real era negativo, y el modelo predijo un valor positivo.

- **False Negative (FN) – Error tipo II**

  El valor real era positivo, y el modelo predijo un valor negativo.

2.2.5.1. Métricas de Rendimiento

Cuando se necesita medir el rendimiento o eficacia de un modelo de clasificación, podemos hacer uso de métricas como Precision, Recall, F1 Score, las cuales se basan en la matriz de confusión, a continuación, se presenta una descripción de cada una de estas (Chakraborty et al., 2021):

2.2.5.1.1. Precision

Es la relación entre las observaciones positivas predichas correctamente y el total de observaciones positivas predichas, una alta precisión se relaciona con una tasa de falsos positivos (FP) baja, se define de la siguiente forma:

$$Precision = \frac{true\ positive}{actual\ results} = \frac{TP}{TP + FP} \tag{15}$$

2.2.5.1.2. Recall

Es la relación de observaciones positivas predichas correctamente con respecto a todas las observaciones de la clase real, los valores altos de Recall están relacionados con tasas bajas de falsos negativos (FN), se define de la siguiente forma:

$$Recall = \frac{true\ positive}{predicted\ results} = \frac{TP}{TP + FN} \tag{16}$$

2.2.5.1.3. F1 Score

Es la media armónica (promedio) de Precision y Recall, por lo tanto, esta puntuación tiene en cuenta tanto a los falsos positivos (FP) como los falsos negativos (FN), esta métrica otorga mayor peso a los FN y FP, sin importar que un gran número de verdaderos negativos (TN) influya en la puntuación final, se define de la siguiente manera:

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{17}$$

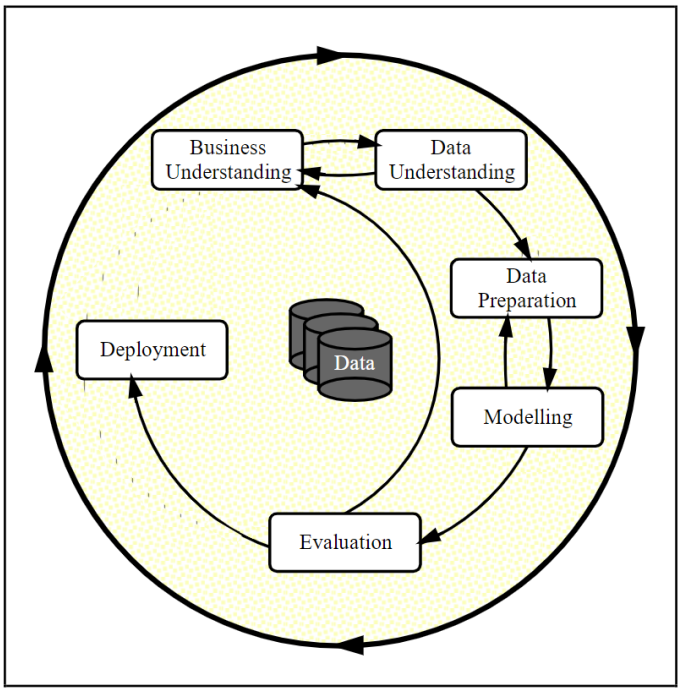
2.2.6. Metodología CRISP-DM

La metodología CRISP-DM (Cross Industry Standard Process for Data Mining) o también conocida como el modelo de referencia CRISP-DM, proporciona una visión genérica del ciclo de vida de un proyecto de minería de datos, el cual es dividido en seis fases iterativas que se muestran en la figura a continuación, no existe una restricción respecto al orden o secuencia de las fases, las flechas sólo indican las dependencias más importantes y frecuentes entre fases, pero dentro de un proyecto particular, la fase que debe realizarse a

continuación, dependerá principalmente del resultado obtenido en la fase previa (Wirth & Hipp, 2000).

Figura 3

Fases de la Metodología CRISP-DM



Nota: Adaptado de Wirth & Hipp (2000)

En la Figura 3, el círculo exterior representa la naturaleza cíclica de la minería de datos. Esto dado el hecho que la minería de datos no finaliza cuando una solución es desplegada, más al contrario, de las lecciones aprendidas en el transcurso de cada fase, incluyendo el despliegue de la solución, se pueden generar nuevas preguntas de negocio mucho más enfocadas.

A continuación se presenta una descripción de cada una de las fases de la metodología CRISP-DM (Azevedo & Santos, 2008; Schröer et al., 2021; Wirth & Hipp, 2000):

2.2.6.1. Comprensión del Negocio

Esta fase inicial se centra en entender los objetivos y requerimientos del proyecto desde una perspectiva de negocios, para posteriormente convertir este conocimiento en la definición de un problema de minería de datos, y el diseño de un plan preliminar para lograr los objetivos.

Se debe de evaluar la situación del negocio para obtener un panorama de los recursos disponibles y requeridos. El aspecto más importante en esta fase es la determinación del objetivo de la minería de datos. Se debe describir el tipo de minería de datos a emplear y el criterio de éxito que será empleado.

2.2.6.2. Comprensión de los Datos

Esta fase inicia con una colección de datos inicial, y se realizan actividades con la finalidad de familiarizarse con los datos, para que de esta forma se puedan lograr identificar problemas de calidad de datos o descubrir los primeros conocimientos respecto a los datos. Como se puede observar en la Figura 3, existe un enlace o relación cercana entre esta fase y la fase de compresión del negocio, esto es debido a que para la formulación del problema de minería de datos y del plan de proyecto, se requiere por lo menos algo de conocimiento sobre los datos disponibles.

2.2.6.3. Preparación de los Datos

Esta fase se encarga de todas las actividades que deben de ser realizadas para lograr la construcción del dataset final (conjunto de datos

que será utilizado en el modelo) a partir de los datos iniciales. Las tareas de preparación que incluye esta fase, son probables que deben ser realizadas múltiples veces, y no existe un orden prescrito, algunas de estas tareas son la selección de tablas, registros y atributos, la limpieza de datos, la construcción de nuevos atributos, y la transformación de los datos para su posterior uso en el modelo.

La selección del conjunto de datos debe ser llevada a cabo considerando un criterio de inclusión y exclusión. La mala calidad de los datos se puede manejar haciendo una limpieza de datos, dependiendo del modelo que será empleado, es posible que se requiera la creación de atributos derivados. Existen diferentes métodos que pueden ser utilizados, pero su aplicación dependerá principalmente del modelo definido.

2.2.6.4. Modelado

En esta fase, distintas técnicas de modelamiento son seleccionadas y aplicadas, y sus parámetros son calibrados a valores óptimos. Comúnmente, existen muchas técnicas para el mismo tipo de problema de minería de datos, algunas de estas requieren formatos de datos específicos. Aquí también existe un enlace cercano con la fase anterior de preparación de datos, esto debido a que realizando el modelado es posible identificar problemas con los datos y realizar los cambios necesarios en la fase de preparación de datos.

2.2.6.5. Evaluación

Antes de proceder con el despliegue final del modelo, en esta fase ya se debe de tener construido uno o más modelos que parezcan tener una calidad alta, desde una perspectiva de análisis de datos. Es importante evaluar el modelo, y revisar los pasos que fueron ejecutados para la construcción del modelo, esto para tener certeza que los objetivos de negocio definidos en un inicio fueron propiamente alcanzados.

2.2.6.6. Despliegue

La finalización de un proyecto de minería de datos, generalmente no culmina con la creación de un modelo, es común que los conocimientos obtenidos necesiten ser organizados y presentados de tal forma que el usuario pueda hacer uso de estos. Dependiendo de los requerimientos, la fase de despliegue puede consistir en algo sencillo como un reporte final o un componente de software, pero es importante entender que las acciones que se lleven a cabo dentro de esta fase tienen el objetivo de hacer que el modelo pueda ser empleado dentro del negocio.

2.3. GLOSARIO DE TÉRMINOS BÁSICOS

2.3.1. Algoritmo

Un algoritmo es un procedimiento finito, que incluye un conjunto finito de instrucciones, con la finalidad de resolver un problema, y como tal puede o no requerir de una entrada, un algoritmo debe ser inequívoco, por lo que todos sus instrucciones deben ser claras y bien definidas, un algoritmo puede ser

representado en un lenguaje natural o mediante un programa de computador que implemente dicho procedimiento (Al-Fedaghi, 2020).

2.3.2. Application Program Interface (API)

Una API es un conjunto de reglas y protocolos que permiten a un programa externo tener acceso a funcionalidades de otra aplicación, también considerado como un intermediario que permite a dos servidores de aplicaciones poder comunicarse entre sí, las APIs tiene como objetivo principal fomentar la reutilización de funcionalidades ya existentes (Gor & Upadhyay, 2019).

2.3.3. Aprendizaje Supervisado

También conocido como aprendizaje automático supervisado, viene a ser una subcategoría del Machine Learning y la Inteligencia Artificial, su característica principal es el hecho de trabajar con conjuntos de datos etiquetados para entrenar algoritmos que clasifiquen datos o predigan resultados con cierta fiabilidad, a diferencia del aprendizaje no supervisado este no puede clasificar o agrupar datos por sí mismo (IBM, s. f.).

2.3.4. Dataset

Un dataset es una colección de observaciones relacionadas y formateadas para un propósito en particular, puede tratarse de un conjunto de imágenes, gráficos, o textos (Chapman et al., 2020). Además puede contener uno o más archivos/tablas que contienen los registros de datos, estos registros están formados por atributos (Lin et al., 2006).

2.3.5. Framework

Un Framework viene a ser un conjunto definido de cosas por realizar, se centra en responder la pregunta de "¿Cómo hacerlo?", este puede incluir elementos que están relacionados unos con otros. Un Framework describe con claridad los pasos en secuencia que se deben de realizar para el cumplimiento de un propósito, describe qué actividades están involucradas, las cuales permiten conectar varios elementos dentro del marco de trabajo (Soni & Kodali, 2013).

2.3.6. Machine Learning

Es un tipo de AI (Inteligencia Artificial), en donde se desarrollan modelos de predicción haciendo uso de conjuntos de datos, para que posteriormente puedan ser empleados en realizar predicciones sobre nuevos datos (Koyuncugil & Ozgulbas, 2019). Así mismo se puede indicar que Machine Learning viene a ser un conjunto de métodos algorítmicos utilizados para la resolución de problemas del mundo real (Hirt et al., 2023).

2.3.7. Modelo de clasificación

Un modelo de clasificación es obtenido después del entrenamiento de un algoritmo de clasificación con cierta data (Yan & Xu, 2009), y se puede representar matemáticamente de la siguiente forma: $f: X \rightarrow Y$, en donde X corresponde a una entrada o ejemplar y Y a la clase, etiqueta, label o target que viene a ser el resultado de la clasificación (Vadillo & Santana, 2019).

2.3.8. Naive Bayes Multinomial

Naive Bayes Multinomial es la implementación del algoritmo de Naive Bayes para datos distribuidos multinomialmente (frecuencia de ocurrencias), se

utiliza con frecuencia para la clasificación de texto, el método utilizado por el clasificador Naive Bayes Multinomial corresponde al aprendizaje supervisado, también es considerado un algoritmo de aprendizaje probabilístico el cual requiere de características discretas como entrada (Damanik & Setyohadi, 2021; Kadam et al., 2018).

2.3.9. Procesamiento del Lenguaje Natural

El NLP es un área de las ciencias de la computación enfocada en el desarrollo de sistemas de software, que hacen uso de funcionalidades de análisis del lenguaje para resolver problemas reales (Vlachidis, 2012). También puede entenderse como la habilidad de utilizar computadoras para procesar texto y poder realizar tareas como la identificación de oraciones (Runyon et al., 2023).

2.3.10. Spam

El spam es cualquier contenido que no es solicitado ni requerido por el usuario, y no es de actual interés para para el receptor, contenido que además es enviado indiscriminadamente (Iqbal et al., 2022), así mismo el spam es una problema masivo en las redes sociales (Prabhu Kavin et al., 2022), por lo mismo que la detección de spam sea haya convertido en un tema que es estudiado continuamente en el área del procesamiento del lenguaje natural (NLP) en los últimos años (Chrismanto et al., 2022).

2.3.11. Stop words

Son palabras que tienen una aparición frecuente en las oraciones del conjunto de datos, por lo mismo que tiene una baja importancia para el análisis, estas palabras deben de ser excluidas de la entrada, mantener stop words en la

data hace que el algoritmo se confunda debido a que estas palabras no tienen un significado contextual, algunos ejemplos de stop words con los artículos, números, símbolos y nombres propios, una forma de eliminar los stop words es teniendo una lista pre compilada de stop words y luego eliminando estas palabras de la data (Deshpande & Kumar, 2018).

2.3.12. Testing

Es el proceso en el cual se evalúa el rendimiento de un modelo completamente entrenado con un conjunto de datos de prueba, permitiendo identificar cualquier desviación del comportamiento del modelo respecto a las expectativas, como también el poder evaluar la calidad de los datos o parámetros empleados, así mismo, este procedimiento posibilita calcular el rendimiento del modelo (Wang & Zheng, 2013).

2.3.13. Training

Es el proceso en el cual un algoritmo de ML (Machine Learning) es "alimentado" con un conjunto de datos de entrenamiento de los cuales este puede llegar a aprender, y se obtiene como resultado un modelo de Machine Learning, el cual posteriormente tendrá la posibilidad de procesar de forma rápida grandes volúmenes de datos, identificar patrones o encontrar anomalías, que podrían resultar en tareas tediosas para las personas (Weedmark, 2021).

2.3.14. YouTube

YouTube es una plataforma popular que permite a los usuarios, subir videos, compartir videos con otros usuarios, y poder realizar comentarios en videos publicados. La característica más prominente de YouTube es que

contiene una gran cantidad de videos de temáticas diferentes, y se encuentra disponible en más de 60 idiomas y muchos países alrededor del mundo (Binmahboob, 2020).

CAPÍTULO III

MATERIALES Y MÉTODOS

3.1. DISEÑO METODOLÓGICO DE LA INVESTIGACIÓN

El enfoque de esta investigación es cuantitativo, debido a la importancia crucial que tiene el proceso de medición o manipulación experimental del conjunto de variables, para de esa forma poder responder las preguntas e hipótesis de investigación que son resultado de la teoría (Creswell & Creswell, 2018).

3.1.1. Tipo de Investigación

La investigación es de tipo experimental debido a que se manipulará deliberadamente una variable experimental o independiente no comprobada, para poder determinar cuáles son las variaciones o efectos que esta pueda causar sobre la variable dependiente (Tamayo, 2003).

3.1.2. Diseño de Investigación

El diseño de la investigación es preexperimental de estudio de caso con una sola medición, debido a que se tiene la certeza de que el grupo experimental no será afectado por algún factor externo que pueda generar cambios sobre este, además de permitir que la conformación del grupo experimental no se deba de realizar estrictamente de forma aleatoria (Hernández Sampieri et al., 2014).

Teniendo para esta investigación el siguiente esquema:

$$G: \quad X \quad O$$

Donde:

- G : Representa al grupo experimental, conformado por el 20% de los comentarios de la muestra.
- X : Representa la aplicación del modelo de clasificación Naive Bayes Multinomial.
- O : Representa la medición del número de comentarios categorizados como spam de forma correcta después de la aplicación del modelo Naive Bayes Multinomial.

3.2. POBLACIÓN Y MUESTRA DE INVESTIGACIÓN

3.2.1. Población

La población de este estudio corresponde a todos los comentarios de videos sobre finanzas en español de la plataforma YouTube, publicados entre los años 2022 y 2023 (Hernández Sampieri et al., 2014).

Se determina esta población con la finalidad de poder crear un modelo de clasificación de utilidad y uso en la actualidad, ya que será entrenado con comentarios actuales de spam y no spam sobre finanzas, esto a su vez permite alargar la vigencia del modelo.

3.2.2. Muestra

El tipo de muestreo empleado corresponde al no probabilístico, con técnica de muestreo según criterio de forma discrecional, en donde la selección de los elementos no depende de la probabilidad sino de algunas características de investigación, como el planteamiento del estudio o diseño de investigación, así mismo la cantidad y los elementos que conformarán la muestra se determinan

utilizando únicamente el criterio del investigador (Hernández Sampieri et al., 2014; Supo, 2014).

Esta elección del muestreo no probabilístico responde y está influenciada por aspectos del planteamiento del estudio, más específicamente el cumplimiento del objetivo general, ya que se busca determinar en qué medida el modelo Naive Bayes Multinomial clasifica correctamente los comentarios spam, para ello es indispensable que una cantidad significativa de los comentarios que formen parte de la muestra sean de tipo spam, esto permitirá la creación de un modelo adecuado.

Por lo mismo es que la muestra de esta investigación estará conformada por un total de 25,672 comentarios en español pertenecientes a 30 videos sobre finanzas de la plataforma YouTube publicados entre los años 2022 y 2023 que contengan necesariamente comentarios spam.

3.3. MATERIALES EMPLEADOS

3.3.1. Recursos de Hardware

Para esta investigación el recurso de hardware corresponde a un portátil, todas las características de interés se presentan en la siguiente tabla.

Tabla 6

Recurso de Hardware

CARACTERÍSTICA	VALOR
Marca	ASUS
Modelo	FX504GE
Procesador	Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz

CARACTERÍSTICA	VALOR
Memoria RAM	8.00GB
Capacidad de almacenamiento	931GB Seagate ST1000LM035-1RK172 (SATA)
Sistema Operativo	Windows 10 Pro 64-bit

Nota: Elaboración propia.

De la tabla anterior, podemos mencionar que se estará trabajando con un solo computador, este corresponde a una con características comunes en la actualidad.

3.3.2. Recursos de Software

Los recursos de software a disposición que se emplearán se describen a continuación en la siguiente tabla:

Tabla 7

Recurso de Software

NOMBRE	VERSIÓN	ICONO
Visual Studio Code	1.80	
Node JS	18.12.0 (LTS)	
Excel	Microsoft 365	
Brave Browser	1.52.130	

Nota: Elaboración propia.

En la Tabla 7, si bien muchos de los programas cuentan con una versión de paga, para el desarrollo del presente trabajo es suficiente trabajar con el acceso que se brinda la capa gratuita.

Las herramientas en línea que se emplearon, son significativamente menores en cantidad a las herramientas de software, las cuales se presentan a continuación.

Tabla 8

Recurso Online

NOMBRE	VERSIÓN	ICONO
Google Colaboratory	Empleando versión de Python 3.10.12.	
Drive	Correspondiente a septiembre de 2023.	

Nota: Elaboración propia.

Sobre la tabla anterior, al tratarse de herramientas en línea no cuentan con una versión específica, por lo mismo que en el apartado de versión solo se menciona una referencia de fecha o de otro componente que permita aproximar una "versión", esto con la finalidad de determinar las características y especificaciones con las que contaban estas herramientas en el momento que fueron empleadas.

En relación a recursos de datos, solo se estará empleando la YouTube Data API en su versión 3, que tendrá un rol fundamental a lo largo del desarrollo del trabajo, esta API es de acceso gratuito, funciona mediante tokens por petición, no cuenta con un plan de paga, y se gestiona a través de la consola de Google.

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

En los siguientes apartados, se procederá a desarrollar las tareas correspondientes a cada uno de los objetivos específicos definidos anteriormente. Este proceso es esencial para asegurar que cada objetivo específico se cumpla de manera detallada y exhaustiva, permitiendo así alcanzar el objetivo general de esta investigación de manera efectiva.

4.1.1. Definición de reglas para la Clasificación

Reglas para la Determinación de Comentarios Spam

Considerando el trabajo de Brown (2022) se define el siguiente conjunto de reglas que permite determinar cuándo un comentario es spam en un video sobre finanzas de la plataforma YouTube, entonces, se tiene que, un comentario será categorizado como spam cuando dentro de este:

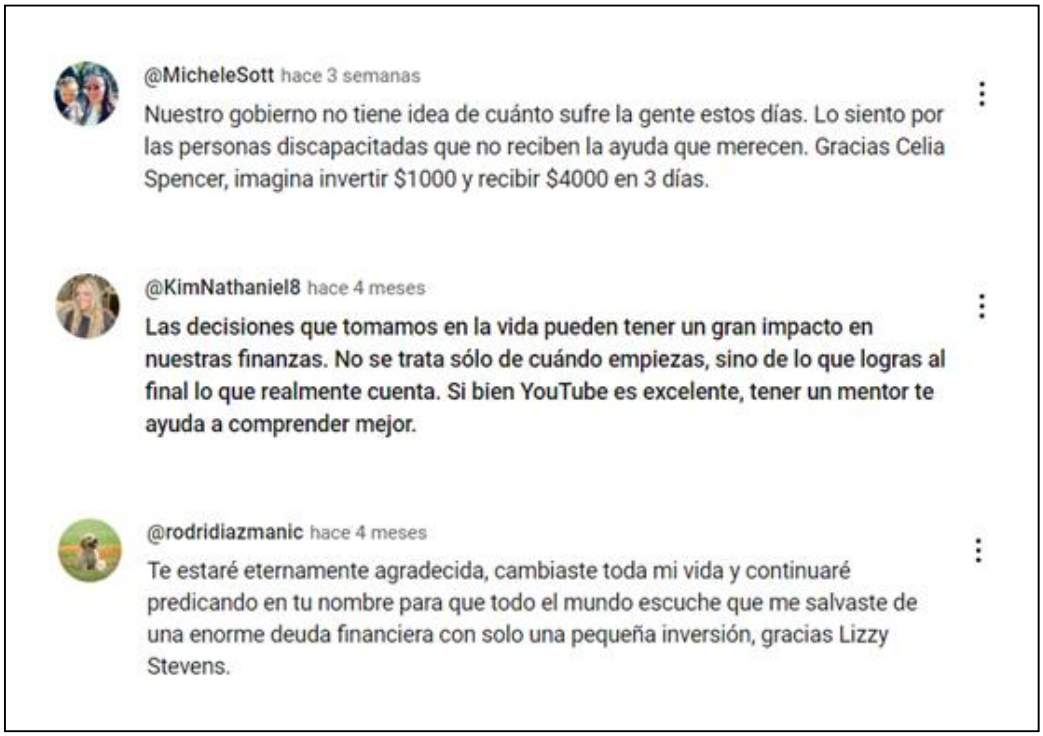
- Se recomienda a una persona experta en inversión, trading y criptomonedas.
- Se habla sobre resultados o ganancias obtenidas gracias a una persona experta.
- Se menciona que es necesario la orientación de una persona experta en inversión.
- Se comparten las redes sociales como WhatsApp, Telegram e Instagram de una persona experta en inversiones.

- Se pide la ayuda de una persona experta en inversiones.
- Se pide el contacto de una persona experta en inversiones.

Algunos ejemplos de comentarios spam de este tipo se muestran en la siguiente figura.

Figura 4

Ejemplos de Comentarios Spam



Nota: Elaboración propia.

De la Figura 4, como se puede observar, en la actualidad la plataforma YouTube trabaja con nombres de usuario único, por lo mismo que los cibercriminales ya no pueden hacer uso de nombres de canales “conocidos”, y por esta razón no se realiza ningún tipo de análisis sobre los nombres de usuario y solo se considera el contenido textual del comentario.

4.1.2. Elaboración del Dataset de Comentarios

4.1.2.1. Determinación de los Objetivos del Negocio

Contexto

YouTube es una de las plataformas con el mayor número de usuarios a nivel mundial, teniendo aproximadamente 1.5 billones de usuarios por mes, también se conoce que las personas pasan viendo videos de YouTube en promedio 1 hora al día, y esto solo considerando a los usuarios de dispositivos móviles (Abdullah et al., 2018).

La sección de comentarios de YouTube es una de las funcionalidades más importantes para los creadores de contenido, ya que se trata del medio por el cual pueden obtener una retroalimentación directa sobre sus videos, pero en la actualidad, esta es utilizada por cibercriminales como un medio para estafar a las personas mediante la ejecución de campañas spam.

A pesar que YouTube tiene su propio sistema para el filtrado de spam, este parece no ser muy eficiente (Oh, 2021), ya que aún existen una gran cantidad de comentarios spam con el que los usuarios deben de lidiar día tras día. Gandra (2014) cómo se citó en Samsudin et al. (2019) afirma que en promedio uno de cada cien comentarios son spam dentro de YouTube.

Objetivos del negocio

- Identificar comentarios spam en español de videos sobre Finanzas de la plataforma YouTube.

Criterios de éxito del negocio

Desde una perspectiva del negocio, el criterio de éxito corresponde a la identificación de una cantidad por encima del 90% de comentarios spam de videos sobre Finanzas de YouTube

4.1.2.2. Evaluación de la Situación

Inventario de recursos

El inventario de recursos ya fue desarrollado en el apartado 3 (Materiales y Métodos).

Requerimientos, supuestos y restricciones

Los requerimientos del proyecto son los siguientes:

- Acceso a una cuenta de Google, para poder hacer uso de los servicios de Google Drive, Colaboratory y la consola de Google.
- Acceso a una credencial de la YouTube Data API, para poder realizar la extracción de los datos a través de peticiones HTTP.
- Acceso a la documentación de la YouTube Data API para la obtención de los distintos endpoints y poder definir los parámetros necesarios de las peticiones HTTP.
- Los supuestos del proyecto son los siguientes:
- Será posible elaborar un conjunto de datos de comentarios mediante la YouTube Data API con el número de tokens que se nos asigna por día.
- Será posible extraer todos los comentarios y respuestas de un video sin presentar problemas de incompletitud.

- Será posible reportar y eliminar comentarios mediante su ID a través de la YouTube Data API.
- No se requerirá contratar ningún tipo de plan de pago (“premium”) de algún software para poder completar el proyecto.
- Las restricciones del proyecto son las siguientes:
- El alcance del proyecto compete a los comentarios en español de videos sobre finanzas dentro de la plataforma YouTube.
- El tamaño del conjunto de datos con el que se estará trabajando estará influenciado por el tiempo que toma trabajar en su elaboración.
- Sobre el conjunto de datos de comentarios de videos sobre finanzas de la plataforma YouTube, los videos serán seleccionados según criterio del investigador en beneficio del proyecto.

Riesgos y contingencias

En la siguiente tabla se mencionan los riesgos existentes por categoría, como también un plan de contingencia correspondiente.

Tabla 9

Riesgos y Contingencias

CATEGORÍA	RIESGO	CONTINGENCIA
Financiero	Necesidad de contratar un plan de pago de algún software o herramienta online.	Buscar alternativas ya sean de código abierto, y en última instancia realizar el pago del software.

CATEGORÍA	RIESGO	CONTINGENCIA
Datos y fuentes de datos	Baja calidad del conjunto de datos, imposibilidad para encontrar patrones diferenciales.	Llevar a cabo una limpieza y preprocesamiento adecuado del conjunto de datos. Averiguar sobre técnicas que puedan mejorar su calidad.

Nota: Elaboración propia.

Respecto a la anterior tabla, es probable la existencia de una mayor cantidad de riesgos, pero los mencionados logran agrupar a una gran cantidad de estos. No es posible considerar otras categorías como la de Negocios y Organizacional, ya que no se cuenta con un control para poder definir planes de contingencia.

Terminología

La compilación del glosario con terminología sobre el negocio y la minería de datos que son de importancia para el proyecto, ya ha sido desarrollada en la sección 2.3.

Costos y beneficios

Los costos identificados para el proyecto son los siguientes:

- Etiquetado del conjunto de datos: Esta no viene a ser una actividad común, ya que para determinar su costo debemos considerar la cantidad en vez del tiempo, entonces podemos indicar que el costo por etiquetar un conjunto de en promedio 20,000 comentarios corresponde a S/. 900.00 (Cloudfactory, 2024).

- Desarrollo e implementación de la solución: Considerando la complejidad del proyecto, podemos indicar que su finalización tomará alrededor de dos meses, y usando de referencia el sueldo mínimo del Perú (S/ 1025.00), estamos hablando entonces de S/ 2050.00 (Ontop, 2024).
- Los beneficios identificados por el proyecto se mencionan a continuación:
- Reducción de la cantidad de comentarios spam existentes.
- Disminución del tiempo que toma reportar y eliminar comentarios spam.
- Mayor control para los creadores de contenido sobre los comentarios spam.
- Mejor experiencia de usuario en la sección de comentarios.

4.1.2.3. Determinación de los Objetivos de la Minería de Datos

Objetivos de la minería de datos

- Elaborar un dataset de comentarios spam en español de videos sobre finanzas de la plataforma YouTube.
- Generar un modelo de clasificación de comentarios spam haciendo uso del algoritmo Naive Bayes Multinomial.

Criterio de éxito de la minería de datos

- El dataset elaborado es robusto y diversificado, además cuenta con una cantidad significativa de comentarios spam y no spam.

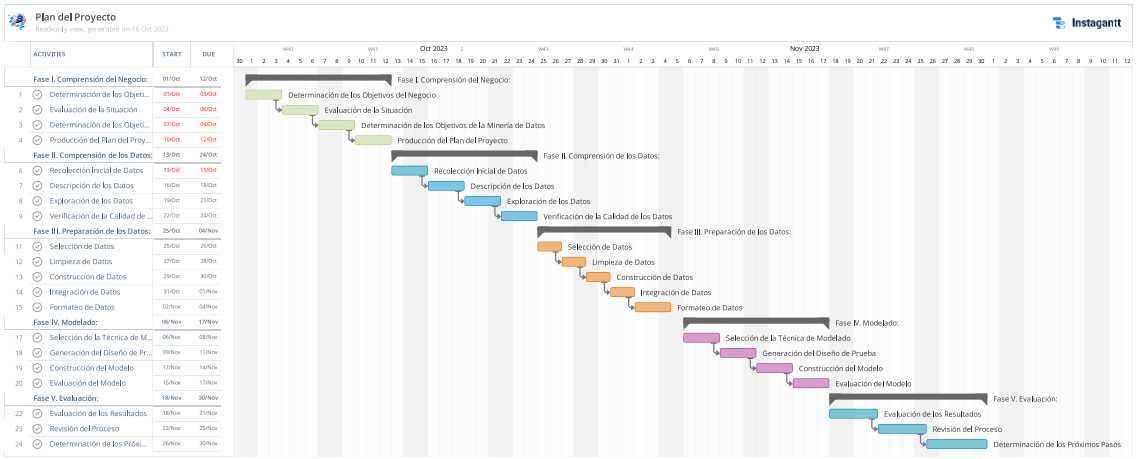
- El modelo de clasificación logra alcanzar un puntaje por encima del 85% en la métrica de evaluación Accuracy y Recall.

4.1.2.4. Producción del Plan del Proyecto

Plan del proyecto

Figura 5

Diagrama Gantt del Plan del Proyecto



Nota: Elaboración propia.

Respecto al diagrama Gantt de la Figura 5, es importante mencionar que cada una de las secciones corresponden a las seis fases de la metodología CRISP-DM y sus actividades correspondientes, las cuales, a través de su desarrollo conducirán al logro de los objetivos de la minería de datos definidos en la presente fase y con esto el cumplimiento de los objetivos del negocio. El tiempo estimado para la culminación de todas las fases es de 61 días calendario, correspondiente a los meses de noviembre y diciembre del 2023.

Evaluación inicial de herramientas y técnicas

A continuación, se realiza una descripción de las herramientas con las que se estará trabajando durante el proyecto:

- **Google Colaboratory**: Viene a ser un Jupyter Notebook pero en la nube, para su uso es necesario contar con una cuenta de Google, los archivos que se creen son almacenados dentro de Google Drive, este entorno es ampliamente utilizado en tareas de minería de datos, ya que cuenta con un gran número de librerías, y además proporciona acceso a GPU con limitantes en el plan gratuito para una mayor velocidad (Chng, 2022).
- **Node JS**: Es un entorno de ejecución de JavaScript enfocado en el lado del servidor, que permita la ejecución de código JavaScript fuera del navegador, emplear Node JS para tareas relacionadas a la minería de datos permite la ejecución de modelos de forma local en el cliente, sin la necesidad de la nube, y con esto posibilita la creación de herramientas mucho más interactivas (Barbosa et al., 2020).
- **Python**: Es un lenguaje de programación, ampliamente utilizado en tareas de minería de datos como la clasificación y el agrupamiento, esto debido a que cuenta con una gran cantidad de librerías con distintos propósitos como Numpy, Matplotlib y Scipy (Rundell, 2016), en lo que respecta al presente trabajo, será empleado en todo el proceso de la elaboración del dataset y la evaluación del modelo.

- **JavaScript**: Es un lenguaje de programación, que puede ser utilizado en cualquier dispositivo que tenga el motor de JavaScript, como los navegadores y servidores (Mozilla, 2023). En el proyecto lo estaremos empleando para la generación del modelo de clasificación en formato JSON (JavaScript Object Notation), esto mediante la implementación del algoritmo Naive Bayes Multinomial usando JavaScript vanilla.

En la parte de técnicas, tenemos a las siguientes:

- **Random Oversampling**: Es una técnica que involucra el duplicado aleatorizado de ejemplares pertenecientes a la clase minoritaria y agregándolas al conjunto de datos de entrenamiento, la técnica de Random Oversampling ha presentado un buen rendimiento en estudios empíricos, incluso en comparación con otros métodos más complejos de oversampling (Stahlbock et al., 2009).
- **Stopwords removal**: Los stop words son generalmente las palabras más comunes en un lenguaje, son palabras simples que tienen un pequeño significado, y son mayormente usadas como parte de la estructura gramáticas de una oración, palabras como “el”, “un”, “en”, etc., son consideradas stop words, la eliminación de los stop words debe ser realiza por que su existencia dentro de la data podría presentar complejidad (Jacob et al., 2021).
- **Stemming**: Es un proceso para obtener la forma base o raíz de una palabra mediante la eliminación de su sufijo o parte final, en el procesamiento del lenguaje natural existen una gran variedad

de algoritmos para el stemming tales como PorterStemmer, SnowballStemmer, y LancasterStemmer, de entre todas estas el algoritmo de PorterStemmer es el que proporciona los mejores resultados en la obtención de la forma base de una palabra (Jacob et al., 2021).

- **Naive Bayes Multinomial**: Es una versión especializada de Naive Bayes que está diseñada para texto o características discretas, realiza una suposición simplificada (ingenua) sobre cómo interactúan las características (independencia mutua), esto consiste en ignorar el orden de las palabras y darle más importancia a la frecuencia de cada palabra dentro del vocabulario, para lo cual hace uso de la representación de bolsa de palabras (bag of words) (Jurafsky & Martin, 2023).

4.1.2.5. Recolección Inicial de Datos

Para la obtención de los datos se empleó el script del Anexo 1 haciendo uso de la YouTube Data API en el entorno de Google Colaboratory, los comentarios que forman parte del conjunto de datos pertenecen a 30 videos sobre finanzas de la plataforma YouTube, a continuación, se estará desarrollando cada una de las tareas llevadas a cabo.

Obtención de una clave API para la YouTube Data API

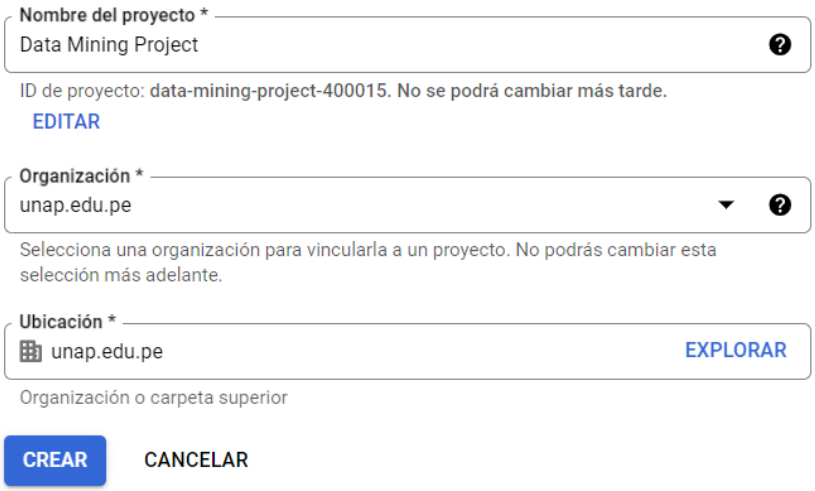
Para obtener los comentarios de los videos a través de la YouTube Data API, se necesita una clave de API o “API Key”, no se requiere de un “Client ID” ya que no se necesitará de ningún token OAuth 2.0,

debido a que las solicitudes a los métodos de la API commentThreads y comments no requieren de una autorización.

Entonces, para la obtención de esta API Key lo primero que se debe realizar es acceder a la Google API Console (https://console.developers.google.com/project) con una cuenta de Gmail y luego crear un proyecto, en la siguiente figura se muestra la creación de un proyecto.

Figura 6

Creación de un Proyecto en Google API Console



Nota: Elaboración propia.

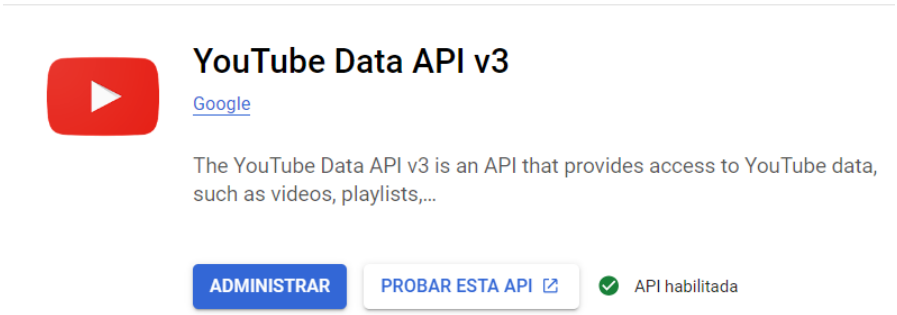
Como se muestra en la Figura 6, para crear un proyecto, solo necesitamos ingresar un nombre, y si es posible seleccionar una organización (para el caso de cuentas organizacionales).

Luego de esto necesitamos habilitar la YouTube Data API en su versión 3, para lo cual dentro del nuevo proyecto nos dirigimos al apartado de “APIs y servicios habilitados” que se encuentra en la barra lateral (lado izquierdo), dentro de ese apartado encontraremos la opción

de "+ HABILITAR APIS Y SERVICIOS", ingresamos y buscamos "YouTube Data API v3", luego de eso obtendremos un único resultado, lo seleccionamos y le damos a "HABILITAR", en la siguiente figura se muestra la API ya habilitada.

Figura 7

YouTube Data API habilitada

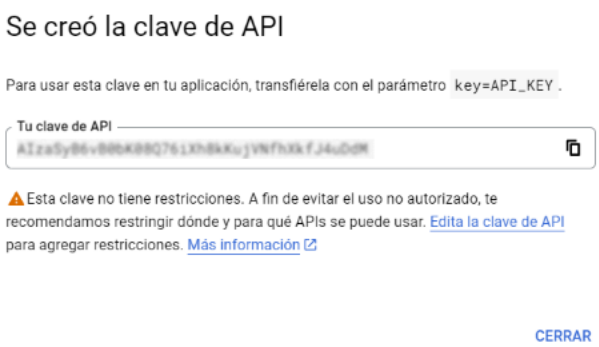


Nota: Elaboración propia.

En este punto, ya es posible obtener una API Key, para ello, se debe ingresar al apartado de "Credenciales" de la barra lateral, posteriormente seleccionar la opción "+ CREAR CREDENCIALES", seguidamente seleccionar la opción "Clave de API", empezará a cargar y después de unos segundos ya se contará con la API Key.

Figura 8

API Key creada para YouTube Data API



Nota: Elaboración propia.

En este punto, es importante mencionar que la necesidad de crear esta clave, está relacionada con el manejo de las cuotas que se le es asignado a los proyectos de forma independiente, con el objetivo de que los usuarios hagan un uso correcto de la API, en la siguiente figura podemos observar las cuotas iniciales asignadas al proyecto creado.

Figura 9

Cuotas Iniciales Asignadas para la YouTube Data API

Cuota	Límite	Porcentaje de uso actual ↓	Uso actual
Queries per day	10,000	0 %	0
Queries per minute	1,800,000	0 %	0
Queries per minute per user	180,000	–	

Nota: Elaboración propia.

Sobre la figura anterior, es importante notar que existen tres "tipos" de cuotas, pero como tal siempre se estará trabajando en función de la primera, ya que es poco probable realizar un consumo sobre las cuotas dos y tres, esto debido al tipo de proyecto en el que se está trabajando, en donde no se necesita realizar un uso extremo de la API con una cantidad excesiva de consultas por minuto.

Respecto a la primera cuota, podemos observar que contamos con 10,000 consultas por día, es importante mencionar que cada método de la API tiene un costo de cuota diferente, esto se muestra en la Tabla 5 correspondiente al marco teórico, en donde se presenta una descripción detallada del costo de cuota de cada uno de los métodos de la YouTube Data API.

Selección de Videos dentro de la plataforma YouTube

Para la selección de videos, de los cuales posteriormente se extraerán los comentarios, lo que primeramente se realizó es una búsqueda común, usando la expresión: "videos sobre finanzas", seguidamente se empezó a revisar cada uno de los resultados obtenidos de forma aleatoria, esto buscando evaluar que el video cumpla con los siguientes puntos:

- El video ha sido publicado entre los años 2022 y 2023.
- El video trata sobre "finanzas".
- Los comentarios del video se encuentran en español.
- Dentro de los comentarios, existen indicios de que una cierta parte son comentarios spam.

Considerando los puntos anteriores, se seleccionaron un total de 30 videos, los cuales se presentan en la siguiente tabla.

Tabla 10

Videos Seleccionados

Nº	TÍTULO	ID	FECHA	URL
01	La gente que tiene estos 4 hábitos NO cae en deudas	-KnxAh2yMWM	16/10/2022	link
02	4 Trucos que usa la gente que SIEMPRE tiene DINERO	1bZuVWBN1jY	02/05/2023	link
03	APRENDE A INVERTIR desde cero - Nivel Básico (Curso completo)	2ibqfxEAESo	06/07/2022	link
04	¿Cómo INVERTIR siendo MENOR de EDAD? | 5 Formas	3-4g-jBN1p4	19/12/2022	link
05	Cómo Invertir Tus Primeros 1000 en 2022 - Paso a Paso (Curso gratuito)	3ybL7JLBqlA	08/04/2022	link
06	¡Empieza a Invertir con $0! Como Invertir Sin Dinero Robert Kiyosaki	45-jr2jA43E	20/04/2023	link
07	💰 Tu LIBERTAD FINANCIERA	68sXnXBxSgw		link

Nº	TÍTULO	ID	FECHA	URL
	inicia en ESTE VIDEO 💰 Aprende lo Básico: Interés Compuesto, Activos y MÁS🚀		29/06/2022	
08	7 IDEAS PARA GANAR INGRESOS DESDE CASA	6orj7xbeBRc	27/01/2023	link
09	💥 ALERTA: “DESPLOME INMOBILIARIO INMINENTE” | Crisis Inmobiliaria 2022: Bajarán los Precios?	CG3Om0f9jiw	15/06/2022	link
10	📊 NO INVIERTAS EN ETFs ANTES DE VER ESTE VIDEO 👉 ¿CÓMO ESCOGER LOS MEJORES ETFs PARA TU PORTAFOLIO?	CQDhVL_lV3g	25/01/2022	link
11	Cuánto se tiene que INVERTIR para recibir $10,000 MXN mensuales de CETES - Paso a Paso 2023	D2wGBfh4ZaM	28/05/2023	link
12	⚠️¿POR QUÉ va a haber una GRAN CRISIS económica en 2023 y cómo prepararte para ella?⚠️	Gy3nOLFIQRA	04/01/2023	link
13	🚀 Eleva tus INGRESOS al SIGUIENTE NIVEL 🚀 8 IDEAS CONCRETAS para generar INGRESOS este 2023 🔥	JmCRMt7lldQ	16/11/2022	link
14	Los 4 Pilares de un EMPRENDEDOR | Podcast de Negocios y Emprendimiento	KgGboHohoqg	11/09/2022	link
15	Las 10 Mejores Lecciones de Dinero Que Cambiaron Mi Vida	NkMQpa2a2Ec	23/04/2023	link
16	💰 5 COSAS que hacen los RICOS y los POBRES no!	ObqT6a7LZXc	02/06/2022	link
17	¿En qué INVERTIR en 2023?	SzKnKC5BvDw	23/01/2023	link
18	Cómo lograr vivir de tus inversiones (3 pasos)	TYfQZA4ZaXs	03/06/2023	link
19	7 REGLAS del Dinero que los RICOS No Quieren que Sepas (La Última es la Mejor)	VOf1Dc_nxCA	15/02/2023	link
20	Cómo INVERTIR tu dinero (para principiantes) : La guía Definitiva	X_h7chiQJ4g	14/09/2022	link
21	"Tenia una DEUDA de $800,000 Dólares" Cómo Pagar tus Deudas | Robert Kiyosaki En Español	baJ5O0Kwaco	12/05/2023	link
22	COMO hacer TRADING siendo PRINCIPIANTES - Curso Gratis de Trading	e5xaGh4Rtqo	22/06/2022	link
23	6 IDEAS para GENERAR INGRESOS en 2023	iHcW5WfnmnU	10/01/2023	link

Nº	TÍTULO	ID	FECHA	URL
24	TUTORIAL 💰 ¿Cómo invertir en CETES desde CERO? : La guía Definitiva 💰 #cetes	k-nQV_QkNl4	09/07/2022	link
25	Cómo generar ingresos pasivos en 2023	lugETBcmV28	02/02/2023	link
26	El mejor TUTORIAL para INVERTIR en acciones para PRINCIPIANTES	mDeSYUoUTXM	07/03/2023	link
27	7 Trucos para AHORRAR mucho DINERO RÁPIDO	o1FCkAj6U7c	21/03/2023	link
28	¿Cómo INVERTIR en CETES? | TUTORIAL desde 0	qF0gQBKg2os	01/11/2022	link
29	Los peores bancos en México para 2023	slkvRuMmgAk	22/12/2022	link
30	Cómo generar 7 fuentes de ingreso en 2022	t1NxNInQW9Q	08/12/2021	link

Nota: Elaboración propia.

En la tabla anterior, se muestran las características básicas sobres los videos seleccionados, un detalle importante a mencionar es que la información presentada corresponde a la fecha del 25/09/2023 (fecha de consulta).

Es conveniente aclarar que el campo ID, corresponde al identificador del video, y se obtiene a partir de la URL del video, en donde seguido de la primera parte “https://www.youtube.com/watch?v=” se nos proporciona una cadena la cual corresponde al ID del video, este identificador será el único valor de utilidad a la hora de la obtención de los comentarios en el siguiente paso.

Obtención de los Comentarios

Con la clave de API creada y los identificadores de los videos listos, se procedió a la obtención de los comentarios pertenecientes a los 30 videos, para lo cual se elaboró un script en Python (Anexo 1), que nos permite obtener todos los comentarios (de primer orden y sus respuestas),

estableciendo para ello una API Key y la lista con los identificadores de los videos.

El script fue programado para que proporcioné como salida un archivo “data.xlsx”, conformado por las siguientes columnas o características:

- Identificador del video (video_id)
- Identificador del comentario (comment_id)
- Nombre del autor (author)
- Contenido del comentario (comment)

A través del script de Python, que emplea en el fondo la YouTube Data API, es posible obtener muchas más características, como la fecha de publicación, el número de likes, entre otros, para propósitos del proyecto, solo son utilidad las características anteriormente enlistadas.

De la ejecución del script, se obtuvo un archivo en formato “.xlsx”, los detalles del archivo se muestran en la tabla a continuación.

Tabla 11

Detalles del Archivo “data.xlsx”

ICONO	CARACTERÍSTICA	VALOR
	Nombre	“data.xlsx”
	Tamaño	2.60 MB
	Número de columnas	4
	Número de registros	25,672 (sin cabeceras)

Nota: Elaboración propia.

Como se puede observar en la tabla anterior, se logró obtener una cantidad significativa de comentarios con los que se estarán trabajando en adelante, cabe aclarar que todos los comentarios de los treinta videos, se encuentran incluidos en este único archivo, y es posible su distinción gracias a la columna “video_id”.

Etiquetado de los Comentarios

Con los comentarios ya obtenidos y las reglas para la determinación de comentarios spam claramente establecidas, se procedió a realizar el etiquetado manual de cada uno de los comentarios de la data, para lo cual dentro del archivo “data.xlsx” generado, se creó una nueva columna con el nombre de “class”, que hace referencia a la etiqueta, los valores o nomenclatura utilizada dentro de esta columna se muestra a continuación:

- 0 : Para un comentario no spam.
- 1 : Para un comentario spam.

4.1.3. Generación del Modelo de Clasificación

4.1.3.1. Descripción de los Datos

A continuación, se presenta la descripción textual de las características o columnas que conforman el archivo “data.xlsx”.

Tabla 12

Descripción de las Características de la Data

CARACTERÍSTICA	TIPO	DESCRIPCIÓN
video_id	Alfanumérico	Corresponde al identificador del video dentro de la plataforma YouTube.
comment_id	Alfanumérico	Corresponde al identificador del comentario dentro de la plataforma YouTube.
author	Alfanumérico	Corresponde al nombre de la persona que realizó el comentario.
comment	Alfanumérico	Corresponde al contenido del comentario.
class	Numérico	Corresponde a la clasificación del comentario, ya sea spam (1) o no spam (0).

Nota: Elaboración propia.

Sobre la tabla anterior, es importante notar que las características que representan un identificador, como “video_id” y “comment_id”, no serán utilizadas en la fase de modelamiento, pero son de utilidad para ciertas tareas que se necesiten realizar, como por ejemplo, si se desea saber a qué video corresponde un comentario entonces a través de “video_id” es posible su identificación, o en el caso que se desee obtener más características del comentario, como su fecha de publicación, para esa situación la forma más fácil sería utilizando el “comment_id”.

Campos Vacíos en la Data

Seguidamente se procedió a evaluar la presencia de campos vacíos para las diferentes características, esta información se presenta en la tabla a continuación.

Tabla 13

Registros Vacíos por Característica

CARACTERÍSTICA	NRO. CAMPOS VACÍOS	NRO. TOTAL DE CAMPOS
video_id	0	25,672
comment_id	0	25,672
author	0	25,672
comment	3	25,672
class	0	25,672

Nota: Elaboración propia.

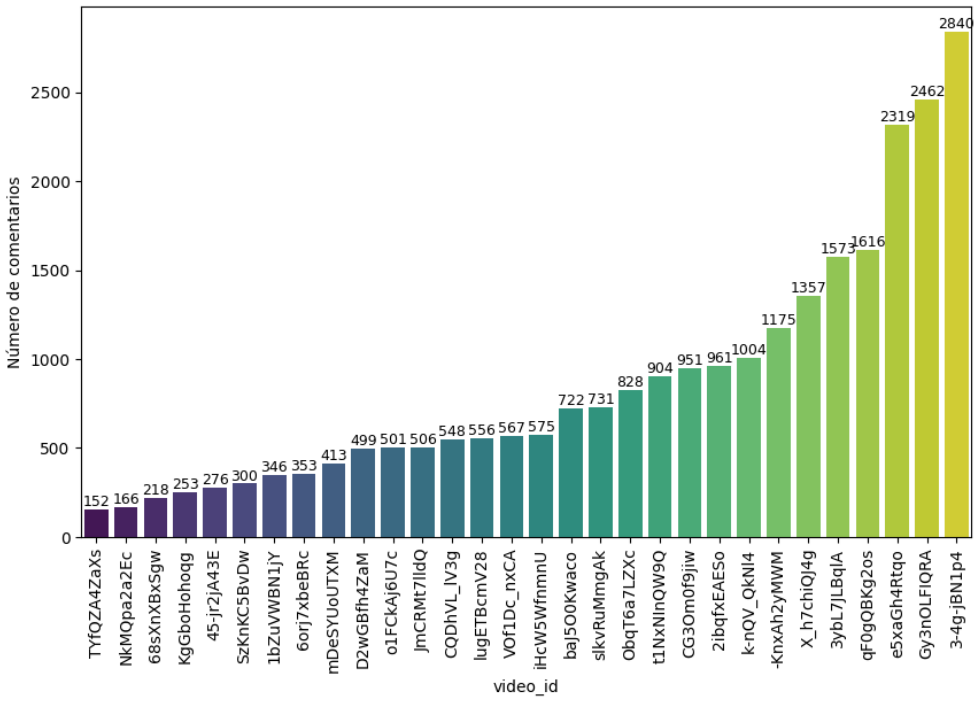
En la tabla anterior, se puede observar que en su mayoría todos los registros de la data se encuentran con contenido, y solo existe una cantidad minúscula de campos vacíos para la característica de "comment", las mismas que serán depuradas en la Fase predecesora.

Comentarios por Video

A continuación, se presenta un gráfico de barras con el número de comentarios correspondiente a cada de cada uno de los 30 videos que forman parte de la data, se presentan en orden ascendente respecto al número de comentarios por video.

Figura 10

Cantidad de Comentarios por Video



Nota: Elaboración propia.

En la figura anterior, podemos observar que la mayor parte de los videos seleccionados contienen un número de comentarios menor a 1,000, y solo una tercera parte supera esta cantidad, así mismo se tienen las cantidades de 152 y 2,840 comentarios para los videos con el menor y mayor número de comentarios respectivamente.

Así mismo, aunque ya fue mencionado, la cantidad total de comentarios correspondientes a los 30 videos, que viene a ser la sumatoria del valor numérico de cada una de las barras de la Figura 10 es de 25,672 comentarios.

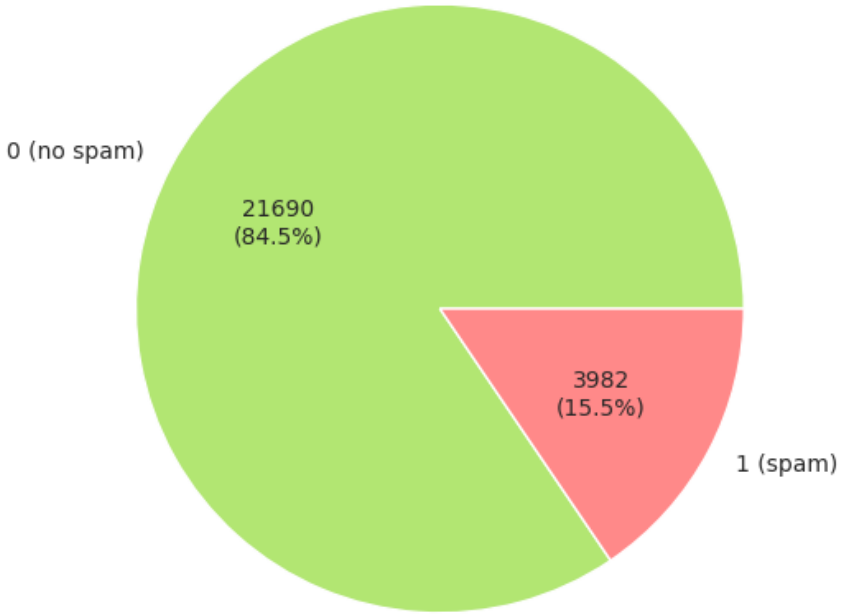
Comentarios por Clase

Primeramente, empezaremos mostrando de forma general el número de comentarios por clase, es decir ya sean spam (1) o no spam

(0), esto mediante un diagrama circular, en donde además del porcentaje correspondiente a cada parte fraccional, también se muestra la cantidad numérica respectiva.

Figura 11

Cantidad de Comentarios por Clase



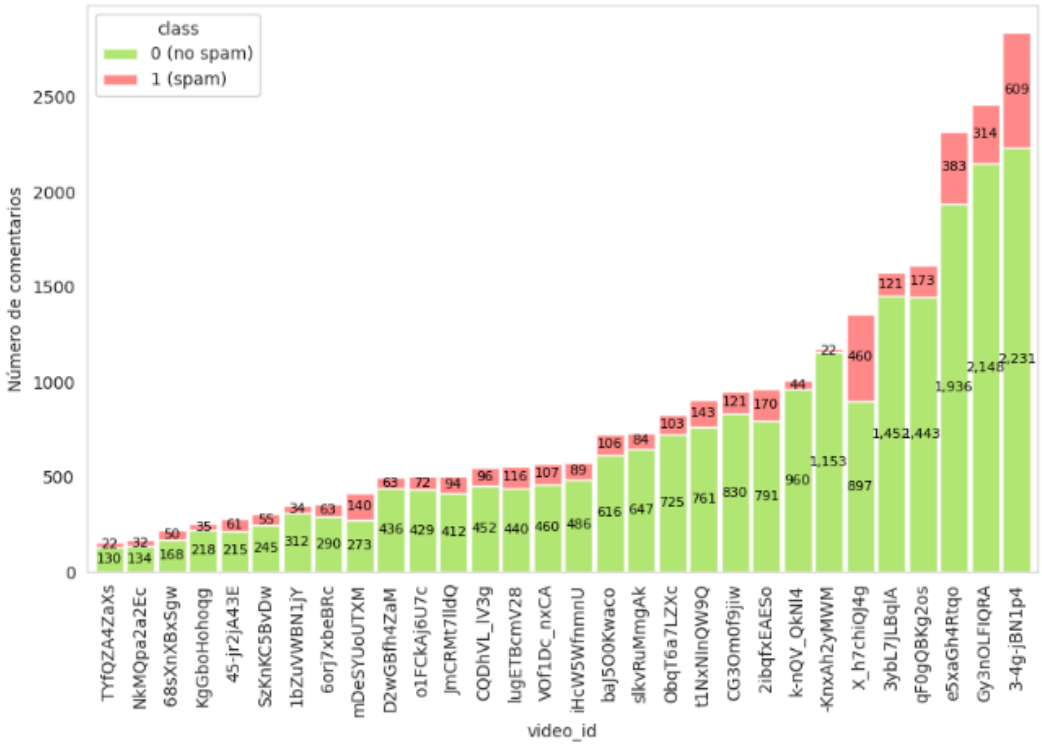
Nota: Elaboración propia.

En la figura anterior, podemos observar que la cantidad total de comentarios spam (con clase 1), corresponde a una proporción menor que la quinta parte respecto al número de comentarios no spam (0), es por ello que, es bastante probable que más adelante sea necesario balancear la data.

Luego se procedió a realizar un análisis a nivel de video, a continuación, se presenta una gráfica de barras apiladas que muestra el número de comentarios spam y no spam, presentes en cada uno de los videos.

Figura 12

Cantidad de Comentarios por Video y Clase



Nota: Elaboración propia.

En la figura anterior, se puede observar que la mayoría de los videos tienen un número de comentarios spam (1) que corresponde como máximo a la cuarta parte del número de comentarios no spam (0), los videos que no cumplen con esta regla viene a ser el de la posición 25 y el ultimo, los cuales además corresponden a los videos con el mayor número de comentario spam en comparación con los otros videos.

4.1.3.2. Exploración de los Datos

Luego de haber realizado la descripción general de la data, esta fase estará enfocada en realizar un análisis, pero únicamente sobre las características de interés, en este caso estamos hablando de la característica “comment”.

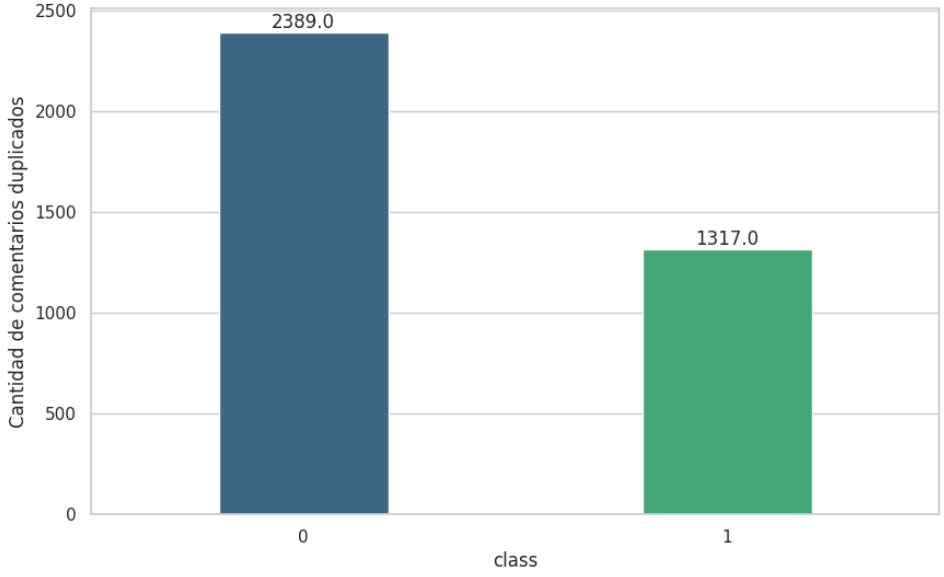
Es importante mencionar que las gráficas de nube de palabras como las tablas de frecuencia correspondientes a esta sección, no consideran a letras individuales, símbolos, ni números, ya que corresponden a “stopwords” naturales al poseer una significancia nula.

Comentarios Duplicados

Empezaremos mostrando una gráfica de barras, que representa a la cantidad de comentarios duplicados presentes en cada clase, es importante mencionar que en este conteo de duplicados no se están considerando a los comentarios que se encuentren vacíos, ya que estos ya fueron revisados previamente, también se excluyen del conteo a las primeras apariciones de los comentarios duplicados.

Figura 13

Cantidad de Comentarios Duplicados por Clase



Nota: Elaboración propia.

En la figura anterior, se tienen un total de 3,706 comentarios duplicados, correspondientes en mayor proporción a la clase no spam (0),

todos estos comentarios estarán siendo eliminados en la Fase de preparación de los datos, ya que su presencia simboliza un sesgo en los resultados finales y su interpretación.

Palabras con Mayor Frecuencia en los Comentarios con Clase No Spam

Empezaremos viendo a un alto nivel, cuales vienen a ser las palabras que mayormente se repiten en todos los comentarios con la clase no spam (0), esto se muestra en la figura a continuación.

Figura 14

Palabras con Mayor Frecuencia en los Comentarios con clase No Spam

Nota: Elaboración propia.

En la figura anterior, algo interesante es el hecho de que en su mayoría las palabras con la mayor frecuencia vienen a ser artículos como "la", "lo", "los", que corresponden a palabras con una poca significancia, así mismo podemos avizorar la presencia de palabras relacionadas al tema de las finanzas como "dinero", "invertir", "comprar", "trading"

entre otras, está en comparación con las anteriores tienen una mayor utilidad.

Seguidamente se procede a mostrar una tabla con las 10 palabras con el valor de frecuencia más alto.

Tabla 14

Las 10 palabras con la Mayor Frecuencia en los Comentarios no Spam

Nº	PALABRA	FRECUENCIA
1	de	15,941
2	que	14,618
3	en	9,925
4	la	8,031
5	el	7,642
6	no	7,413
7	es	6,221
8	para	5,207
9	un	4,869
10	me	4,826

Nota: Elaboración propia.

En la tabla anterior se puede observar que, las cuatro primeras palabras superan las 8,000 unidades de frecuencia, esto se debe a que si vienen estas palabras tienen una aparición más frecuente dentro de los comentarios, el análisis se realizó sobre los comentarios no spam, los cuales ya de por sí representan la mayor parte del volumen total.

Palabras con Mayor Frecuencia en los Comentarios con Clase Spam

De forma similar se procedió a realizar un análisis de frecuencia sobre las palabras de los comentarios, pero en este caso para los que

tienen como clase el valor de spam (1), a continuación, comenzaremos mostrando la nube de palabras correspondiente a este segundo caso.

Figura 15

Palabras con Mayor Frecuencia en los Comentarios con clase Spam

Nota: Elaboración propia.

En la figura anterior, el primer detalle que podemos observar es que existen palabras que también se encuentran dentro de las palabras de comentarios con clase no spam (0) como por ejemplo “de”, “que”, “la”, entre otras, así mismo, también podemos observar palabras relacionados al contexto del spam como “sra”, “ella”, “cartera”, “ganancias”, “inversión” entre otras.

Ahora se continuará mostrando la tabla con las 10 palabras con el mayor valor de frecuencia correspondientes a comentarios con la clase spam (1).

Tabla 15

Las 10 palabras con la Mayor Frecuencia en los Comentarios Spam

Nº	PALABRA	FRECUENCIA
1	de	4,909
2	la	2,916
3	con	2,826
4	que	2,656
5	en	2,370
6	es	1,826
7	el	1,577
8	ella	1,380
9	mi	1,272
10	un	1,247

Nota: Elaboración propia.

En la tabla anterior, en comparación a los valores de frecuencias mostrados para la clase no spam, podemos observar que el mayor valor de frecuencia alcanzado viene a ser aproximadamente la tercera parte de la palabra con el mayor valor de frecuencia que se alcanzó en la clase no spam, esto como tal se debe a que la cantidad total de comentarios no spam y spam no es proporcional.

Palabras con Mayor Frecuencia de los Comentarios para ambas Clases

Para poder determinar este conjunto de palabras, lo primero que se necesita definir es el tamaño inicial de palabras con el que se trabajará, para este caso la cantidad será de 300, con las 300 palabras por grupo no spam y spam se procede a realizar la intersección, para con ello

determinar cuáles son las palabras con la mayor frecuencia presentes en ambas clases.

La cantidad obtenida de la intersección corresponde a 145 palabras, a continuación, se presenta una tabla que muestra a las primeras 10 palabras con el mayor valor de frecuencia de la intersección, la tabla completa con las 145 palabras se puede encontrar en el Anexo 2.

Tabla 16

Las 10 palabras con Mayor Frecuencia de los Comentarios para ambas Clases

Nº	PALABRA	FRECUENCIA	
		NO SPAM (0)	SPAM (1)
1	de	15,941	4,909
2	que	14,618	2,656
3	en	9,925	2,370
4	la	8,031	2,916
5	el	7,642	1,577
6	no	7,413	798
7	es	6,221	1,826
8	para	5,207	974
9	un	4,869	1,247
10	me	4,826	1,197

Nota: Elaboración propia.

En la tabla anterior, aunque en un principio parezca que se trata únicamente de las palabras pertenecientes al grupo de no spam, ya que vienen a ser las mismas palabras que se presentaron en la Tabla 14, es necesario notar que, al ser las 10 palabras con la frecuencia más alta, estas también tienen presencia en los comentarios spam, es por ello que los valores de frecuencia presentados no guardan una relación clara.

Estás 145 palabras, resultado de la intersección, es lo más importante que se obtuvo de este análisis, ya que como tal formarán parte de la lista de "stopwords".

Nube de palabras de Comentarios con Clase No Spam sin Stopwords

Con el listado de Stopwords generado, se procede a realizar el filtrado de estos en los comentarios con clase no spam para observar cuales son las palabras más representativas de esta clase.

Figura 16

Palabras más representativas de los Comentarios con clase No Spam

Nota: Elaboración propia.

En la figura anterior, podemos observar las palabras que tienen las frecuencias más altas dentro de los comentarios con clase No Spam, estas palabras son muy importantes ya que al hacer el cálculo de las probabilidades que un comentario pertenezca o no a la clase No Spam

está involucrada directamente la frecuencia de las palabras que lo conforman.

Nube de palabras de Comentarios con Clase Spam sin Stopwords

De forma similar al anterior punto, en esta parte se continúa mostrando la nube de palabras ahora para los comentarios pertenecientes a la clase Spam, la figura se muestra a continuación.

Figura 17

Palabras más representativas de los Comentarios con clase Spam

Nota: Elaboración propia.

En la figura anterior, podemos observar que las palabras con una mayor frecuencia corresponden en algunos casos a artículos como ella y él, esto podría interpretarse como un problema al momento de realizar la construcción de Stopwords, pero en este escenario no es así, ya que al realizar la evaluación de los comentarios de la clase Spam, se puede ver el patrón de referirse a las “personas expertas” usando artículos en vez de

su nombre directo en cada una de las respuestas que realizan las cuentas bots a los comentarios principales.

4.1.3.3. Verificación de la Calidad de los Datos

Después de haber realizado la descripción y exploración de la data, podemos decir lo siguiente:

- La data está completa, esto significa que todas las características con las que se cuentan son suficientes para el cumplimiento del objetivo del proyecto, esto se debe en gran parte al hecho de estar trabajando con una data elaborada específicamente para este estudio, ya que desde un inicio se consideraron todas las necesidades y requerimientos futuros de forma responsable, algo que probablemente no habría ocurrido si es que se hubiese tomado una data existente cualquiera.
- La data es correcta y no contiene errores de alto riesgo, esto significa que no existirá una complejidad alta a la hora de realizar las correcciones de errores. Problemas que se encontraron respecto a este punto tiene que ver con campos vacíos presentes en la característica “comment” que son un total de tres, lo cual no corresponde a un problema de complejidad, ya que serán depurados de la data en la siguiente fase.

4.1.3.4. Selección de Datos

En este estudio se está trabajando con una sola data, es por ello que no es posible realizar una selección a ese nivel, así mismo todos los

registros de la data serán utilizados, pero si existe la necesidad de realizar una selección a nivel de características o variables, a continuación, se presentan las características que fueron seleccionadas con sus razones, y del mismo modo para las características que no fueron seleccionadas.

Características Seleccionadas

Las características seleccionadas vienen a ser:

- comment
- class

Se seleccionaron estas características ya que vienen a ser las únicas que serán de utilidad en la parte del modelado, la característica "comment" almacena la información textual de un comentario, que será una entrada del modelo (previo a un preprocesamiento) tanto en el proceso de entrenamiento y prueba, de forma similar la característica "class" que almacena si un comentario es de tipo spam (1) o no (0), también será una entrada del modelo, en este caso sin la necesidad de un preprocesamiento previo.

Características No Seleccionadas

Como características no seleccionadas tenemos:

- video_id
- comment_id
- author

Estas características no fueron seleccionadas por lo siguiente, en relación a las dos primeras, corresponden a identificadores, como tal no tienen una utilidad al momento de realizar el modelado, pero como se pudo apreciar, si tuvieron un papel importante en las tareas de descripción y exploración de la data, y en relación a la última característica, “author”, no es seleccionada, ya que en cambios recientes de la plataforma YouTube, este ahora trabaja con nombres de usuarios únicos para cada cuenta, esto imposibilita a que cuentas falsas puedan usar el nombre de usuario de creadores de contenido reales, por lo mismo que esté característica ya no es relevante.

4.1.3.5. Limpieza de Datos

Como ya se había anunciado en la fase anterior, es momento de realiza una limpieza a los datos, en este caso mediante la eliminación de los registros con comentarios sin contenido (vacíos), y de los que se encuentren duplicados, todo esto se realizó haciendo uso del script del Anexo 3.

Eliminación de Registros con Comentarios Vacíos

La cantidad de comentarios vacíos ya fue identificada en la Tabla 13, que corresponde a un total de tres, estos se presentan en la siguiente figura.

Figura 18

Comentarios vacíos dentro de la Data

	video_id	comment_id	author	comment	class
0	VOf1Dc_nxCA	UgxD5ss0hdjB4Qqb-wl4AaABAg	SAMUEL GUEVARA ARIAS	NaN	0
1	3ybL7JLBqlA	UgzMgJ-S5_8KE9E-LtV4AaABAg	Alejandro Rico Méndez	NaN	0
2	3-4g-jBN1p4	Ugw5HIvh24mzVXeU73F4AaABAg	Jenniffer Nacipucha	NaN	0

Nota: Elaboración propia.

En la figura anterior, se puede observar que los tres comentarios vacíos fueron etiquetados como no spam (0), y además pertenecen a videos diferentes, luego de la eliminación de los registros de estos comentarios, dentro de la data se tiene un total de 25,669 registros de una cantidad inicial de 25,672.

Eliminación de Registros con Comentarios Duplicados

De forma similar se procede a realizar la eliminación de los registros con comentarios duplicados, estos comentarios ya fueron presentados de forma explícita en la Figura 13, en donde se observó que son una cantidad total de 3,706 comentarios duplicados, esto sin incluir a las primeras apariciones, en la figura a continuación se presentan cinco ejemplares de estos comentarios.

Figura 19

Cinco ejemplares de Comentarios Duplicados dentro de la Data

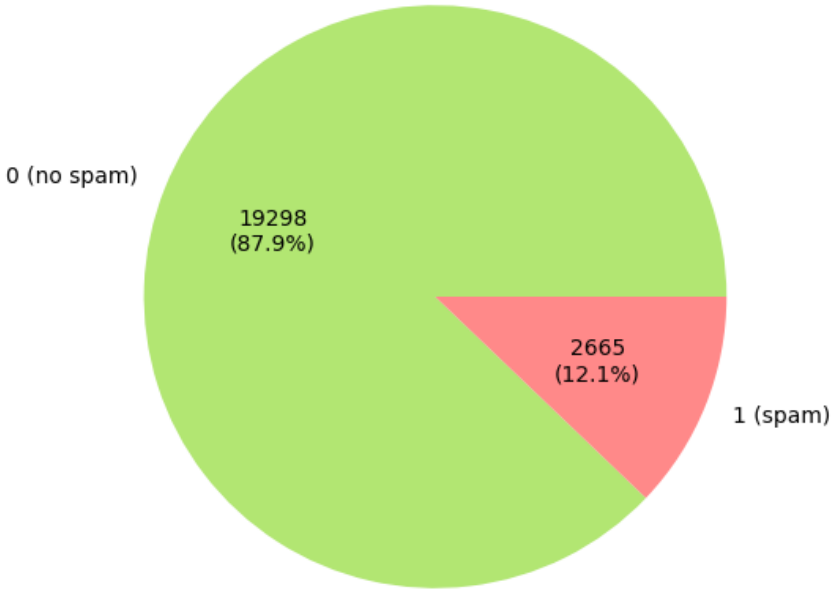
	video_id	comment_id	author	comment	class
0	3-4g-jBN1p4	Ugytj0_s76ZGDy4sCpR4AaABAg.9uf50zzaG5Y9ufR_zsOFV4	Kate fernando	Al principio era escéptico hasta que decidí in...	1
1	mDeSYUoUTXM	UgzNgR-xO52poKE5MdV4AaABAg.9uKQ_1boXSE9uOINSJZtxO	Laura	Perspicaz, pude comunicarme con ella gracias p...	1
2	lugETBcmV28	UgysqEexAGE6_VaRRjB4AaABAg.9uPGg4pc_jQ9uPHLlQjU1V	Sophia	Puedes enviar un ella mensaje	1
3	JmCRMt7lldQ	Ugwj3nTMMvRGoazcDFx4AaABAg.9u__0SExk_59u_a07K_NOt	Cesaria M Ramírez	¡Guau, eso es impresionante! La Sra. Maureen M...	1
4	JmCRMt7lldQ	Ugwj3nTMMvRGoazcDFx4AaABAg.9u__0SExk_59u__yx6-3yH	Katharina Saldivar	Parece tener una buena educación. Busqué su no...	1

Nota: Elaboración propia.

Después de la eliminación de los 3,706 registros con comentarios duplicados, dentro de la data quedaron un total de 21,963 registros, en este punto es importante mostrar cual es el estado de la data respecto al número de registros por clase, es por ello que en la siguiente figura se presenta una gráfica circular que representa dicha información.

Figura 20

Cantidad de Registros por Clase después de la Limpieza de Datos



Nota: Elaboración propia.

En la figura anterior, se puede observar la desproporcionalidad existente entre los registros con clase no spam (0) y spam (1), y ya con la limpieza de datos realizada, es el momento adecuado para poder balancear la data, con la finalidad de contar con el mismo número de registros por clase.

4.1.3.6. Construcción de Datos

Considerando las características de la data, como el tipo de información que almacena cada característica, no fue necesario realizar la

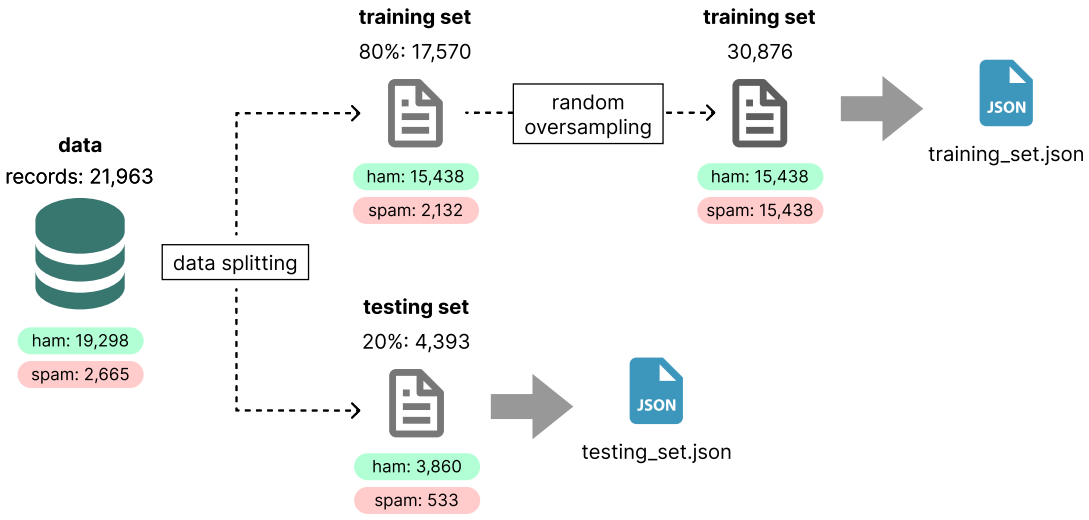
producción de atributos derivados o la creación de registros completamente nuevos, es por ello que en esta tarea en específico no es necesario realizar ninguna actividad, y con esto la data sigue manteniendo un total de 21,963 registros.

4.1.3.7. Integración de Datos

En esta parte se estarán realizando distintas actividades, que son de relevancia, es por ello que a continuación se presenta una figura que esquematiza el flujo de las actividades que se llevaron a cabo, con la finalidad de que todo el procedimiento pueda ser comprendido de forma clara.

Figura 21

Procedimiento para la Integración de Datos



Nota: Elaboración propia.

Sobre la figura anterior, primeramente, indicar que todo el procedimiento mostrado fue realizado utilizando el script del Anexo 4, ahora bien, como se puede apreciar, el primer procedimiento realizado consistió en la separación de la data (“data splitting”), en dos grupos;

"training set" y "testing set", con la cantidad de registros correspondientes al 80% y 20% respectivamente.

Posterior al "data splitting", se realizó un "random oversampling", sobre el "training set", esta es una de las tareas de mayor importancia, ya que, el "training set" será el conjunto de datos con los que se entrenará el modelo, y como se indicó en la parte del marco teórico, el algoritmo Naive Bayes Multinomial, que será utilizado en el modelado, necesita obligatoriamente que el dataset con el que será entrenado se encuentre balanceado.

Como última actividad de la Figura 21, se procedió a guardar tanto el "training set" como el "testing set" en archivos con formato JSON, ambos archivos contienen las claves "comments" y "classes", los conjuntos de datos fueron almacenados en formato JSON debido a que las tareas que aún faltan realizar sobre estos ya no se estarán llevando a cabo en el entorno de Google Colaboratory, a partir de este punto se empezará a trabajar en el entorno de Node JS.

4.1.3.8. Formateo de Datos

Implementación del Preprocesador de Texto

Se implementó el preprocesador de texto (TextPreprocessor) en JavaScript usando el entorno de Node JS, el código fuente correspondiente se encuentre en el Anexo 5, hablando de forma general, el script tiene como objetivo recibir una cadena de texto (comentario), y devolver un arreglo de tokens, que puedan ser utilizados posteriormente por el modelo de clasificación Naive Bayes Multinomial.

Cada uno de los métodos pertenecientes a la clase TextPreprocessor se definen a continuación:

- **cleaner**

  Este método recibe una cadena de texto, y retorna una cadena de texto limpia, para esto primeramente convierte toda la cadena a minúscula, elimina distintos tipos de caracteres como: arrobas, comillas, signos de puntuación, números, espacios, entre otros.

- **tokenizer**

  Este método recibe una cadena de texto, y devuelve un arreglo de palabras (tokens), realiza la separación de la cadena considerando los espacios en blanco intermedios entre palabras.

- **stopWordsRemover**

  Este método recibe un arreglo de tokens (palabras), evalúa cada uno de los tokens, y desecha los tokens que se encuentren dentro de la lista de stopwords (Anexo 2).

- **stemmer**

  Este método recibe un arreglo de tokens, y devuelve otro arreglo con el resultado de haber pasado los tokens de entrada por el stemmer PorterStemmer, en forma general lo que busca PorterStemmer es convertir una palabra a su forma base. La implementación de PorterStemmer utilizada como base pertenece al paquete de código abierto Natural (*NaturalNode/natural: general natural language facilities for node*, s. f.).

- **preprocess**

  Este es el método principal, utiliza cada uno de los métodos descritos anteriormente, recibe como entrada una cadena de texto, y la pasa a través de los métodos cleaner, tokenizer, stopWordsRemover y stemmer de forma secuencial, devolviendo como resultado un arreglo de tokens.

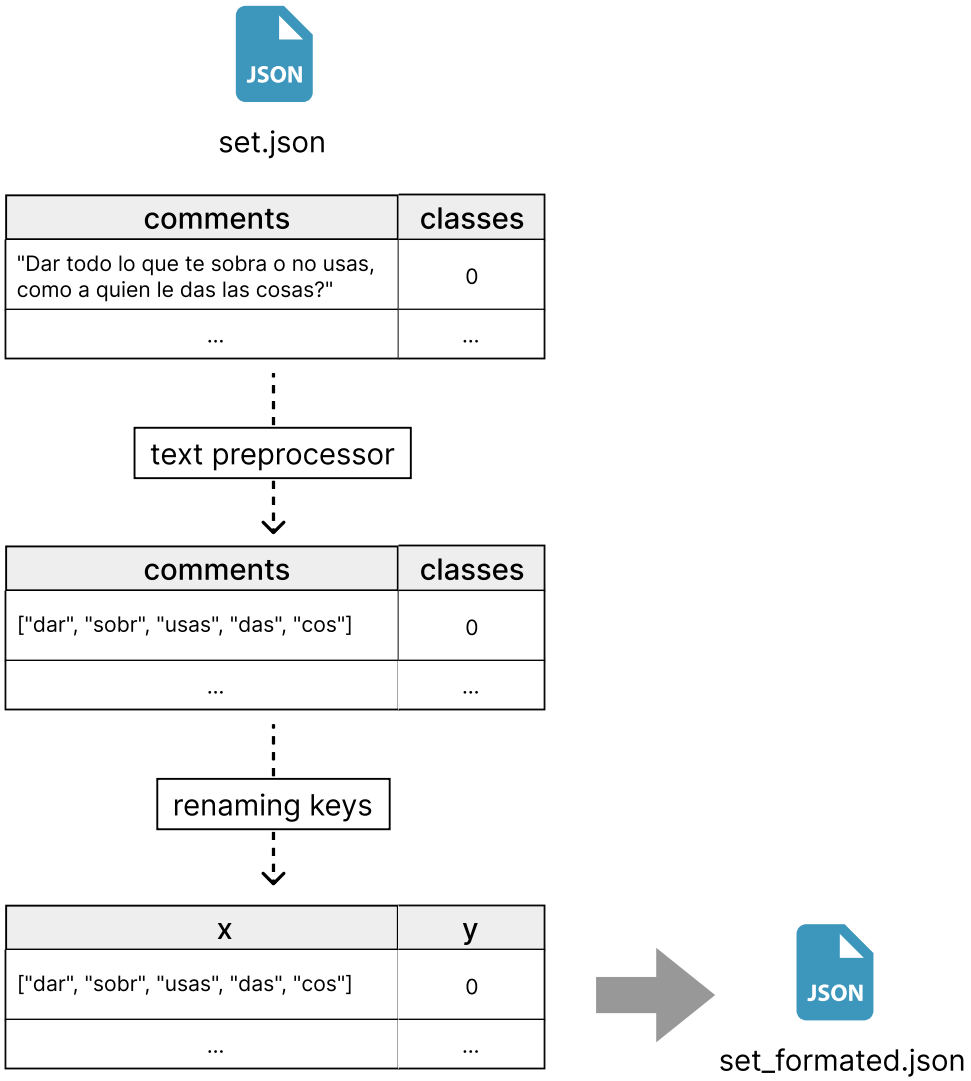
Formateo de conjunto de datos de Entrenamiento y Prueba

Con la clase TextPreprocessor implementada, lo que se realizó seguidamente fue formatear el “training set” y “testing set”, buscando con esto que ambos conjuntos de datos tengan el formato de entrada esperado por el modelo de clasificación Naive Bayes Multinomial, para esto se utilizó el script del Anexo 6.

El proceso del formateo de los conjuntos de datos se presenta de forma generalizada a continuación en la siguiente figura.

Figura 22

Procedimiento del Formateo de los Conjuntos de Datos



Nota: Elaboración propia.

En la figura anterior, es importante mencionar que se trata de una representación gráfica generalizada del procedimiento del formateo de los conjuntos de datos, esto debido a que en ambos datasets se realizaron las mismas acciones (text preprocessor y renaming keys), en ese sentido "set.json" representa tanto al "training set" como al "testing set", y de forma similar la salida "set_formated.json" representa a "training_set_formated" y "testing_set_formated" a la vez.

De la Figura 22, la principal tarea llevada a cabo es el “text preprocessor”, ya que es en donde se emplean métodos de la clase “TextPreprocessor”, que permiten la obtención de arreglos de tokens que formarán parte de la entrada tanto en la etapa de training y testing, también se tiene a la tarea de “renaming keys”, esta tiene una menor relevancia, ya que solo consiste en cambiar los nombres de las llaves de los archivos JSON (conjuntos de datos) con fines de utilizar una nomenclatura convencional.

4.1.3.9. Selección de la Técnica de Modelado

Para la selección de la técnica de modelo, se consideró los objetivos del proyecto de minería de datos establecidos en un principio, en donde en uno de los puntos se menciona la generación de un modelo de clasificación, para este caso se seleccionó el algoritmo Naive Bayes Multinomial, esto considerando las distintas características a favor del algoritmo reflejados en el apartado 2.2.4.3 (Algoritmo Naive Bayes Multinomial).

Los algoritmos de clasificación como Naive Bayes Multinomial pueden ser encontrados en distintas herramientas de minería de datos, como en librerías de algunos lenguajes de programación, en este caso, al buscar la generación de un modelo que posteriormente pueda ser utilizado en la web de forma independiente, es necesario trabajar con un algoritmo deslindado de dependencias, ya que esto podría condicionar el uso del modelo en el futuro en un entorno más restrictivo como las extensiones web.

Implementación del Algoritmo Naive Bayes Multinomial

Después de haber realizado la evaluación de algunas opciones de herramientas disponibles para este propósito, es que se optó por realizar una implementación propia del algoritmo Naive Bayes Multinomial usando JavaScript en el entorno de Node JS, para esto, el algoritmo base que se tomó como referencia, corresponde al de Manning et al. (2008), que fue presentado en la Figura 1.

La implementación del algoritmo estructurado en una clase se puede encontrar en el Anexo 7, en este punto, es importante reportar las principales modificaciones realizadas a partir del algoritmo base de Manning et al. (2008), la primera está relacionada a la puntuación final por clase (score), en donde ya no se considera el valor de la probabilidad previa de las clases, esta modificación se presenta en la siguiente figura:

Figura 23

Modificación Probabilidad previa Algoritmo Naive Bayes Multinomial

```
//scores[class_name] = this.log_prior[class_name];
scores[class_name] = 0;
```

Nota: Elaboración propia.

En la figura anterior, es importante notar que el cambio realizado no representará una variación significativa en los resultados, ya que, al estar trabajando con un conjunto de datos balanceados, que es nuestro caso, el valor de la probabilidad previa de las clases es el mismo, esto sería pericialmente igual a trabajar con un valor de probabilidad previa de cero.

El algoritmo Naive Bayes Multinomial al momento de realizar una predicción siempre devolverá el valor de una clase, en este caso 0 (no spam) o 1 (spam), para esto simplemente se selecciona a la clase con el mayor puntaje obtenido y la devuelve como resultado de predicción, aun cuando la diferencia con la otra clase sea casi nula, esto hace que el índice de falsos positivos se incremente notoriamente (se predicen como spam cuando no son spam).

Para solucionar este problema, el segundo cambio realizado está relacionado con la determinación del resultado de la predicción, en donde el resultado final se determinará a través de un umbral de proporción, esta modificación se representa en la siguiente figura.

Figura 24

Modificación determinación de Predicción Algoritmo Naive Bayes Multinomial

```
// const max_probability = Math.max(...Object.values(scores));
// pred = Object.keys(scores).find((key) => scores[key] === max_probability);

let threshold = 1.4;
pred = scores["1"] / scores["0"] >= threshold ? "1" : "0";
```

Nota: Elaboración propia.

En la figura anterior, ahora con la modificación realizada, para que el resultado de una predicción sea 1 (spam) el puntaje alcanzado en la clase “spam” sobre el alcanzado en la clase “no spam” debe de ser mayor o igual al valor del umbral establecido, el valor adecuado de umbral de proporción corresponde a 2.5, aunque a partir de 1 ya se pueden apreciar mejoras significativas.

4.1.3.10. Generación del Diseño de Prueba

Al tratarse de una tarea de aprendizaje supervisado, en este caso de clasificación, es conveniente hacer uso de las distintas métricas derivadas de la matriz de confusión, como Precision, Recall y F1 score, estas métricas nos permitan medir el error y del mismo determinar la calidad del modelo de clasificación generado, además que con esto se podrá dar alcance a nuestros objetivos de minería de datos planteados en este proyecto.

En relación a la preparación de la data para la ejecución de la prueba, es importante mencionar que esto ya fue realizado en el apartado 4.1.3.4 (Integración de datos), ya que era necesario que antes de realizar cualquier tipo de integración, el training y set ya se encuentren separados, para que las modificaciones (el balanceado) solo se realicen sobre el conjunto de datos de entrenamiento, y de esa forma poder mantener la pureza del conjunto de datos de prueba, ya que sobre ese conjunto se realizarán las distintas mediciones.

4.1.3.11. Construcción del Modelo

Ajustes de Parámetros

Primeramente, se empezó ajustando el único parámetro a disposición, se trata del umbral de proporción mínimo (threshold), que surge a partir de la segunda modificación del algoritmo Naive Bayes Multinomial, como ya se mencionó, el valor adecuado es el de 2.5. Se establece este valor como resultado de haber realizado múltiples pruebas, y como se mostrará en el apartado de evaluación del modelo, se puede

apreciar que este valor de threshold es el que proporciona los mejores resultados.

Modelo

Ya teniendo los parámetros establecidos, se procedió a la generación del modelo de clasificación, para lo cual se hizo uso del script del Anexo 8, que emplea la implementación realizada del algoritmo Naive Bayes Multinomial en el entorno de Node JS, para la generación del modelo se emplea el conjunto de datos de entrenamiento.

Es importante mencionar que la generación del modelo demora muy poco tiempo, estamos hablando de menos de un minuto, esto se debe a características propias del algoritmo y también al hecho de estar trabajando con un conjunto de datos que ya había sido procesado previamente, como salida, el script guarda el modelo generado en formato “json” con el nombre “model.json”.

Descripción del Modelo

El modelo generado y guardado en formato JSON tiene un tamaño de 1.945 KB, y su estructura interna se presenta a continuación en la siguiente figura.

Figura 25

Estructura interna del Modelo en formato JSON

Nota: Elaboración propia.

En la figura anterior, se puede observar que el modelo en formato JSON está conformado por un total de 52,394 líneas, y contiene seis atributos o llaves, de entre todos estos, el atributo de mayor utilidad viene a ser el de “log_likelihood”, ya que como se puede observar dentro del algoritmo propuesto por Manning et al. (2008), en el método de predicción (predict) se utilizarán en mayor parte los valores de este objeto en particular para determinar los puntajes finales de predicción.

Rescatando otro atributo, podemos mencionar a “class_names”, que permite identificar que se están trabajando con dos clases usando la nomenclatura de 0 y 1, otro punto llamativo es el valor de la llave “log_prior”, en donde para cada clase (0 y 1), se tiene el mismo valor, como ya se explicó anteriormente, este valor de “log_prior” fue descartado por esa misma razón, su valor no influye en el resultado final en este escenario.

4.1.4. Evaluación del Rendimiento del Modelo

4.1.4.1. Evaluación del Modelo

Para esta etapa de la evaluación, debemos de seguir el diseño de prueba planteado anteriormente, considerando ello, se hizo uso del script del Anexo 9, en donde primeramente se carga el modelo que acaba de ser generado, y se realizan las predicciones sobre el conjunto de datos de prueba (20%), seguidamente se hizo uso de métricas de evaluación que derivan a partir de los resultados de la matriz de confusión que se presenta a continuación en la Tabla 16.

Tabla 17

Resultados Matriz de Confusión

		CLASES PREDICHAS		TOTAL
		SPAM (1)	**NO SPAM (0)**	
CLASES REALES	**SPAM (1)**	TP = 437	FN = 96	**533**
	NO SPAM (0)	FP = 122	TN = 3,738	**3,860**
TOTAL		**559**	**3,834**	**4,393**

Nota: Elaboración propia.

En la tabla anterior, es importante recordar que los resultados mostrados corresponden para el valor de umbral de proporción mínimo de 2.5, empezando con la clase spam (1), podemos observar que el modelo realiza una predicción correcta el 78% de las veces, y respecto a la clase no spam (0), el modelo muestra aún un mejor desempeño, en donde su porcentaje de aciertos es del 97%.

Así mismo se puede apreciar que las celdas con los menores valores corresponden a la de los falsos negativos (FN) y falsos positivos

(FP), lo cual indica que el modelo realiza predicciones erróneas con menor frecuencia, estos valores dependen del umbral, por lo mismo que si se trabaja con un valor menor, la cantidad de falsos negativos puede disminuir considerablemente, pero esto también hará que la cantidad de falsos positivos incremente, es por ello que lo más conveniente es buscar un equilibrio entre ambos.

A partir de los resultados de la Tabla 17, es posible determinar el puntaje de las métricas Precision, Recall y F1 Score, estos valores se presentan a continuación en la Tabla 18.

Tabla 18

Resultados de las Métricas de Evaluación por Umbral

MÉTRICA	VALOR DE UMBRAL						
	1.0	**1.3**	**1.6**	**1.9**	**2.2**	**2.5**	**2.8**
Precision	0.49	0.59	0.66	0.71	0.74	0.78	**0.80**
Recall	**0.98**	0.96	0.93	0.89	0.85	0.81	0.77
F1 Score	0.65	0.73	0.77	0.79	0.79	**0.80**	0.78

Nota: Elaboración propia.

En la tabla anterior, los valores de umbral utilizados van en paso de 0.3, no es posible determinar a primeras cuál es el valor de umbral que proporcione el mejor rendimiento, ya que las métricas no coinciden en un solo valor, pero, como se conoce, la métrica F1 Score utiliza y mide a la vez el Precision y Recall, se puede decir entonces que, el mejor valor de umbral de proporción mínima corresponde al de 2.5.

Individualmente hablando, la métrica Precision mide que tan preciso es el modelo para la identificación de comentarios spam, se

puede observar que su valor incrementa a medida que el valor del "threshold" es cada vez mayor, esto se debe a de forma general Precision tiene una relación inversa a la cantidad de falsos positivos (FP), y estos a su vez tienen otra relación inversa con el valor del umbral.

Recall es la métrica de mayor importancia para este estudio, ya que permite lograr el objetivo general, con esto dar respuesta a la pregunta de investigación y a su vez evaluar la hipótesis planteada, debido a que esta métrica mide el porcentaje de comentarios que el modelo clasifica correctamente como spam entre todos los comentarios que realmente eran spam, y como se puede observar en la Tabla 18, el modelo obtuvo el mayor puntaje correspondiente 98% para un umbral de 1.0.

4.2. DISCUSIÓN

A partir de los resultados obtenidos, se da por aceptada a la hipótesis general, que indica que un modelo Naive Bayes Multinomial clasifica correctamente una cantidad significativa de comentarios spam en español en videos sobre finanzas de la plataforma YouTube, ya que en la métrica de rendimiento Recall, que permite la medición del enunciado de la hipótesis, se obtuvo un puntaje del 98%.

Tomando en consideración el tipo de conjunto de datos empleado, tenemos primeramente al dataset de comentarios spam de videos musicales de la plataforma YouTube perteneciente a la UCI Machine Learning, utilizado en los estudios de Oh (2021), Ifriza & Sam'an (2021) y Samsudin et al. (2019), en donde en relación al primero, los resultados son bastantes similares a los alcanzados en este estudio, debido a

que en el estudio de Oh (2021) se alcanzó un puntaje del 81% para el algoritmo Bernoulli Naive Bayes respecto a la métrica F1-score.

En el estudio de Ifriza & Sam'an (2021) los resultados obtenidos son parcialmente superiores, ya que se alcanzó un puntaje del 84% respecto a la métrica F1-score en este caso para el algoritmo Gaussian Naive Bayes, este incremento en el puntaje puede deberse en gran parte al número de ejemplares con el que se trabajó, ya que en el estudio en mención solo se utilizaron los comentarios de un video musical, lo cual representa una cantidad pequeña de ejemplares, esto además se sustenta con la explicación realizado por Oh (2021) en donde demuestra que cuando se trabaja con una mayor cantidad de ejemplares, el rendimiento del modelo tiende a disminuir.

De forma similar, en el estudio de Samsudin et al. (2019), el puntaje alcanzado corresponde al 87% para el algoritmo Naive Bayes, pero esto respecto a la métrica Precision, lo cual representa un puntaje superior al alcanzado en este estudio, este resultado puede ser algo engañoso ya que el conjunto de datos con el que se trabajó, similar al estudio de Ifriza & Sam'an (2021), está conformado por menos de 2,000 ejemplares, lo cual, como se explicó en el anterior párrafo, influye directamente en la obtención de un puntaje más alto en las métricas de rendimiento, , y además que emplea software especializado de minería de datos, Weka y RapidMiner, que proporciona herramientas de preprocesamiento de texto y algoritmos más potentes en comparación a los utilizados en el presente estudio.

El estudio de Narváez-Albuja (2022) es el que presenta una mayor semejanza al presente trabajo, tanto en el algoritmo utilizado como los datos, en donde se empleó un conjunto de datos sobre comentario spam de videos sobre finanzas de la plataforma YouTube, en donde el total de ejemplares es de 255,109, el puntaje obtenido para el

algoritmo Naive Bayes Multinomial corresponde a 74% respecto a la métrica F1-score, lo cual representa un puntaje inferior al obtenido en el presente estudio (80%), esto se debe en gran parte a que no se realizó un buen preprocesamiento de la data, como la eliminación de stopwords en base a una lista personalizada, ya que respecto al algoritmo, no es posible decir mucho, debido a que se utilizó una implementación de Naive Bayes bastante sólida perteneciente a la librería Scikit-learn de Python.

V. CONCLUSIONES

PRIMERA: Se determinó que el modelo Naive Bayes Multinomial desarrollado clasifica correctamente el 98% de los comentarios spam en español de videos sobre finanzas en la plataforma YouTube. Este resultado indica que el modelo tiene un rendimiento adecuado, especialmente considerando la métrica de evaluación Recall.

SEGUNDA: Se estableció un conjunto de seis reglas para identificar comentarios spam en videos sobre finanzas de YouTube. Estas reglas se derivaron de un análisis exhaustivo del conjunto de datos, donde se identificaron patrones recurrentes en las intenciones, mensajes y palabras de los comentarios spam. Este análisis reveló que una característica común en estos comentarios es la recomendación de un "experto" en finanzas. Las reglas formuladas permiten una identificación preliminar eficaz de comentarios potencialmente spam, facilitando así la etapa de etiquetado y procesamiento de datos.

TERCERA: Se elaboró un conjunto de datos de comentarios spam en español de videos sobre finanzas de la plataforma YouTube, haciendo uso de la YouTube Data API y el conjunto de reglas establecidas, la tarea con el mayor trabajo involucrado corresponde al etiquetado manual del conjunto de datos, que en este caso asciende a los 25,000 ejemplares, que vienen a ser comentarios pertenecientes a 30 videos sobre finanzas de actualidad, publicados entre los años 2022 y 2023, en donde se logró recabar un total de 3,982 y 21,690 comentarios spam y no spam

respectivamente.

CUARTA: Se desarrolló un modelo de clasificación de comentarios spam utilizando el algoritmo Naive Bayes Multinomial. Este desarrollo incluyó la implementación del algoritmo en el entorno de Node.js con JavaScript. Además, se creó una clase de preprocesamiento de texto (TextPreprocessor) para limpiar y preparar adecuadamente los comentarios antes de su análisis. El proceso de preprocesamiento incluyó la normalización del texto, eliminación de caracteres especiales y tokenización. La combinación de estas implementaciones permitió generar un modelo robusto y eficiente para la clasificación de comentarios spam a través del entrenamiento del algoritmo Naive Bayes Multinomial.

QUINTA: Se evaluó el rendimiento del modelo de clasificación Naive Bayes Multinomial, haciendo uso de tres métricas de rendimiento, en este caso Precision, Recall y F1 Score, las evaluaciones fueron realizadas para diferentes valores de umbral de proporción mínima, en donde se observó que el mejor valor de umbral corresponde al de 2.5, esto debido a que respecto a la métrica F1 score, que considera tanto a Precision y Recall en su medición, se alcanzó una puntuación del 80%, por otro lado la métrica Recall alcanzó su mejor puntaje (80%) para el mayor valor de umbral probado (2.8), caso contrario ocurre para la métrica Recall, la cual alcanzó su mejor puntaje (98%) para el menor valor de umbral (1).

VI. RECOMENDACIONES

PRIMERA: Se recomienda a YouTube mejorar su actual herramienta de filtrado de spam, integrando algoritmos más potentes de Machine Learning como el de Naive Bayes Multinomial para realizar un mejor filtrado de spam y darle la importancia necesaria al tema de los comentarios spam.

SEGUNDA: Se recomienda tomar un enfoque de neutralidad en la definición de las reglas que determinen si un comentario es spam o no, evitando centrarse en particularidades y viéndolo de manera holística, buscando que cada una de las reglas esté definida de forma clara y que no entre en conflicto con las demás.

TERCERA: Se recomienda realizar una evaluación exhaustiva para determinar cuáles son las características que necesariamente deben formar parte del conjunto de datos. Así mismo, buscar la forma más óptima de extracción de datos, valorando a métodos de consumo de APIs sobre otras técnicas.

CUARTA: En la parte de modelamiento, se sugiere realizar un preprocesamiento personalizado del contenido textual del conjunto de datos, empleando una lista específica de Stopwords adaptada al contexto del estudio y utilizando técnicas avanzadas como la Lematización si fuese posible.

QUINTA: Se recomienda emplear la métrica o técnica de evaluación más precisa y pertinente, tomando en consideración la tarea de Machine Learning que se lleva a cabo y la naturaleza del conjunto de datos, pudiendo ser balanceado o desbalanceado.

VII. REFERENCIAS BIBLIOGRÁFICAS

Abdullah, A. O., Ali, M. A., Karabatak, M., & Sengur, A. (2018). A comparative analysis of common YouTube comment spam filtering techniques. *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, 1-5. https://doi.org/10.1109/ISDFS.2018.8355315

Aiyar, S., & Shetty, N. P. (2018). *N-Gram Assisted Youtube Spam Comment Detection*. *132*, 174-182. https://doi.org/10.1016/j.procs.2018.05.181

Alberto, T. C., Lochter, J. V., & Almeida, T. A. (2015). TubeSpam: Comment Spam Filtering on YouTube. *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, 138-143. https://doi.org/10.1109/ICMLA.2015.37

Al-Fedaghi, S. (2020). Exploration in Algorithm Engineering: Modeling Algorithms. *arXiv preprint arXiv:2012.01908*.

Alias, N., Foozy, C. F. M., & Ramli, S. N. (2019). Video spam comment features selection using machine learning techniques. *Indonesian Journal of Electrical Engineering and Computer Science*, *15*(2), 1046-1053. https://doi.org/10.11591/ijeecs.v15.i2.pp1046-1053

API Reference | YouTube Data API. (s. f.). Google for Developers. Recuperado 1 de septiembre de 2023, de https://developers.google.com/youtube/v3/docs

Arce, J. I. B. (2019, julio 26). *La matriz de confusión y sus métricas – Inteligencia Artificial –*. Juan Barrios. https://www.juanbarrios.com/la-matriz-de-confusion-y-sus-metricas/

Azevedo, A., & Santos, M. F. (2008). KDD, SEMMA and CRISP-DM: A parallel overview. *IADS-DM*.

Barbosa, V., Ngo, T. A., Eck, P. V., & Wang, Y.-H. (2020). *An introduction to AI in Node.js*. IBM Developer. https://developer.ibm.com/tutorials/an-introduction-to-ai-in-nodejs/

Behzadi, F. (2015, septiembre 9). *Natural Language Processing and Machine Learning: A Review*. (IJCSIS) International Journal of Computer Science and Information Security. https://civilica.com/doc/522657/

Bhandari, A. (2020, abril 17). Understanding & Interpreting Confusion Matrix for Machine Learning (Updated 2023). *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/

Binmahboob, T. (2020). YouTube as a Learning Tool to Improve Students' Speaking Skills as Perceived by EFL Teachers in Secondary School. *International Journal of Applied Linguistics and English Literature*, *9*(6), 13-22. https://doi.org/10.7575/aiac.ijalel.v.9n.6p.13

Brown, A. (2022, abril 4). *YouTube superstars issue new scam warning, how to avoid the trick*. Express.Co.Uk. https://www.express.co.uk/life-style/science-technology/1590429/youtube-comment-spam-scam-warning-trick-trying-to-steal-your-money

Chakraborty, R., Ghosh, A., & Mandal, J. K. (2021). *Machine Learning Techniques and Analytics for Cloud Security*. John Wiley & Sons.

Chapman, A., Simperl, E., Koesten, L., Konstantinidis, G., Ibáñez, L.-D., Kacprzak, E., & Groth, P. (2020). Dataset search: A survey. *The VLDB Journal*, *29*(1), 251-272. https://doi.org/10.1007/s00778-019-00564-x

Chng, Z. M. (2022, abril 27). Google Colab for Machine Learning Projects. *MachineLearningMastery.Com*. https://machinelearningmastery.com/google-colab-for-machine-learning-projects/

Cho, S.-E., & Suh, J. (2021). Translating Korean Beauty YouTube Channels for a Global Audience. En R. Desjardins, C. Larsonneur, & P. Lacour (Eds.), *When Translation Goes Digital: Case Studies and Critical Reflections* (pp. 173-197). Springer International Publishing. https://doi.org/10.1007/978-3-030-51761-8_8

Chrismanto, A. R., Sari, A. K., & Suyanto, Y. (2022). SPAMID-PAIR: A Novel Indonesian Post–Comment Pairs Dataset Containing Emoji. *International Journal of Advanced Computer Science and Applications*, *13*(11).

Cloudfactory. (2024). *Data Labeling for ML in 2024: A Comprehensive Guide*. https://www.cloudfactory.com/data-labeling-guide

Creswell, J. W., & Creswell, J. D. (2018). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. SAGE Publications.

Cuzcano Chavez, X. M., & Ayma Quirita, V. H. (2020). *A comparison of classification models to detect cyberbullying in the Peruvian Spanish language on twitter*. https://repositorio.ulima.edu.pe/handle/20.500.12724/12718

Damanik, F. J., & Setyohadi, D. B. (2021). Analysis of public sentiment about COVID-19 in Indonesia on Twitter using multinomial naive bayes and support vector machine. *IOP Conference Series: Earth and Environmental Science*, *704*(1), 012027.

Deshpande, A., & Kumar, M. (2018). *Artificial Intelligence for Big Data: Complete guide to automating Big Data solutions using Artificial Intelligence techniques*. Packt Publishing Ltd.

Djerf-Pierre, M., Lindgren, M., & Budinski, M. A. (2019). The role of journalism on YouTube: Audience engagement with «Superbug» reporting. *Media and Communication*, *7*(1), 235-247. https://doi.org/10.17645/mac.v7i1.1758

Gor, B., & Upadhyay, D. (2019). *Enhancing Security: Vulnerability Detection of an API.*

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2014). *Metodología de la investigación*. McGraw Hill España. https://dialnet.unirioja.es/servlet/libro?codigo=775008

Hirt, R., Kühl, N., Martin, D., & Satzger, G. (2023). Enabling inter-organizational analytics in business networks through meta machine learning. *Information Technology and Management*. https://doi.org/10.1007/s10799-023-00399-7

IBM. (s. f.). *What is Supervised Learning? | IBM*. Recuperado 4 de septiembre de 2023, de https://www.ibm.com/topics/supervised-learning

Ifriza, Y. N., & Sam'an, M. (2021). Performance comparison of support vector machine and gaussian naive bayes classifier for youtube spam comment detection.

Journal of Soft Computing Exploration, *2*(2), Article 2. https://doi.org/10.52465/joscex.v2i2.42

Iqbal, H., Khan, U. M., Khan, H. A., & Shahzad, M. (2022). Left or Right: A Peek into the Political Biases in Email Spam Filtering Algorithms During US Election 2020. *Proceedings of the ACM Web Conference 2022*, 2491-2500. https://doi.org/10.1145/3485447.3512121

Jacob, I. J., Shanmugam, S. K., Piramuthu, S., & Falkowski-Gilski, P. (2021). *Data Intelligence and Cognitive Informatics: Proceedings of ICDICI 2020*. Springer Nature.

Jimenez Palomino, R. (2018). *Influencia del Aprendizaje Computacional Basado en Técnicas de Minería de Textos en la Clasificación de Comentarios de Textos Agresivos*. https://repositorio.unajma.edu.pe/handle/20.500.14168/323

Jurafsky, D., & Martin, J. H. (2023). Naive Bayes and Sentiment Classification. En *Speech and Language Processing* (3.ª ed., p. 23). https://web.stanford.edu/~jurafsky/slp3/

Kadam, S., Gala, A., Gehlot, P., Kurup, A., & Ghag, K. (2018). Word embedding based multinomial naive bayes algorithm for spam filtering. *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, 1-5.

Koyuncugil, A. S., & Ozgulbas, N. (2019). Statistical roots of machine learning, deep learning, artificial intelligence, big data analytics and data mining. *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science*, 22-24.

Lin, P. J., Samadi, B., Cipolone, A., Jeske, D. R., Cox, S., Rendón, C., Holt, D., & Xiao, R. (2006). Development of a synthetic data set generator for building and testing information discovery systems. *Third International Conference on Information Technology: New Generations (ITNG'06)*, 707-712.

Madnani, N. (2007). Getting started on natural language processing with Python. *XRDS: Crossroads, The ACM Magazine for Students*, *13*(4), 5. https://doi.org/10.1145/1315325.1315330

Manning, C. D., Raghavan, P., & Schütze, H. (2008). Text classification and Naive Bayes. En *Introduction to Information Retrieval* (p. 35). https://nlp.stanford.edu/IR-book/pdf/13bayes.pdf

Mozilla. (2023, septiembre 16). *What is JavaScript? - Learn web development | MDN*. https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript

Narváez-Albuja, A. (2022). *Aplicación antispam de comentarios para prevenir delitos informáticos en la plataforma de YouTube* [masterThesis]. https://reunir.unir.net/handle/123456789/14020

NaturalNode/natural: General natural language facilities for node. (s. f.). Recuperado 4 de octubre de 2023, de https://github.com/NaturalNode/natural

Nesbo, E. (2021, septiembre 4). *What Is a YouTube Spambot? How Can You Stop Them?* MUO. https://www.makeuseof.com/what-is-a-youtube-spambot/

Nguyen, A. (2023, mayo 17). *What is the RPM for Finance Category in YouTube*. ClashPanda. https://clashpanda.com/what-is-the-rpm-for-finance-category-in-youtube/

Oh, H. (2021). A YouTube Spam Comments Detection Scheme Using Cascaded Ensemble Machine Learning Model. *IEEE Access*, *9*, 144121-144128. IEEE Access. https://doi.org/10.1109/ACCESS.2021.3121508

Ontop. (2024). *What is the minimum wage in Peru?* https://www.getontop.com/blog/what-is-the-minimum-wage-in-peru-in-2024

Polamuri, S. (2020, agosto 3). *Best Confusion Matrix Guide With Sklearn Python—Dataaspirant*. https://dataaspirant.com/confusion-matrix-sklearn-python/

Prabhu Kavin, B., Karki, S., Hemalatha, S., Singh, D., Vijayalakshmi, R., Thangamani, M., Haleem, S. L. A., Jose, D., Tirth, V., & Kshirsagar, P. R. (2022). Machine learning-based secure data acquisition for fake accounts detection in future mobile communication networks. *Wireless Communications and Mobile Computing*, *2022*, 1-10.

Prakash, K. (2019). Advances in Natural Language Processing – A Survey of Current Research Trends, Development Tools and Industry Applications. *International Journal of Recent Technology and Engineering (IJRTE)*. https://www.academia.edu/40140224/Advances_in_Natural_Language_Processing_A_Survey_of_Current_Research_Trends_Development_Tools_and_Industry_Applications

Raschka, S. (2017). *Naive Bayes and Text Classification I - Introduction and Theory* (arXiv:1410.5329). arXiv. https://doi.org/10.48550/arXiv.1410.5329

Rundell, M. (2016, octubre 3). *Data Mining in Python: A Guide*. Springboard Blog. https://www.springboard.com/blog/data-science/data-mining-python-tutorial/

Runyon, C. R., Harik, P., & Barone, M. A. (2023). "Cephalgia" or "migraine"? Solving the headache of assessing clinical reasoning using natural language processing. *Diagnosis*, *10*(1), 54-60. https://doi.org/10.1515/dx-2022-0047

Samsudin, N. M., Mohd Foozy, C. F. B., Alias, N., Shamala, P., Othman, N. F., & Wan Din, W. I. S. (2019). Youtube spam detection framework using naïve bayes and logistic regression. *Indonesian Journal of Electrical Engineering and Computer Science*, *14*(3), 1508. https://doi.org/10.11591/ijeecs.v14.i3.pp1508-1517

Schröer, C., Kruse, F., & Gómez, J. M. (2021). A systematic literature review on applying CRISP-DM process model. *Procedia Computer Science*, *181*, 526-534.

Sinhal, A., & Maheshwari, M. (2022). An Extensive Review on Contemporary Analysis of Comment Filtration of YouTube Videos Using Machine Learning Techniques. *International Journal of Emerging Technology and Advanced Engineering*, *12*(9), 130-141. https://doi.org/10.46338/ijetae0922_14

Soni, G., & Kodali, R. (2013). A critical review of supply chain management frameworks: Proposed framework. *Benchmarking: an international journal*, *20*(2), 263-298.

Stahlbock, R., Crone, S. F., & Lessmann, S. (2009). *Data Mining: Special Issue in Annals of Information Systems*. Springer Science & Business Media.

Supo, J. (2014). Cómo elegir una muestra: Técnicas para seleccionar una muestra representativa. *Perú: Bioestadístico*.

Tamayo, M. T. y. (2003). *El proceso de la investigación científica* (4.ª ed.). Editorial Limusa.

ThioJoe (Director). (2021, noviembre 3). *I Created an App That DESTROYS Scam Comments (Because YouTube Wouldn't)—OUTDATED, READ DESCRIPTION* [Video recording]. https://www.youtube.com/watch?v=-vOakOgYLUI

Tong, L.-I., Chang, Y.-C., & Lin, S.-H. (2011). Determining the optimal re-sampling strategy for a classification model with imbalanced data using design of experiments and response surface methodologies. *Expert Systems with Applications*, *38*(4), 4222-4227. https://doi.org/10.1016/j.eswa.2010.09.087

Torres, M. J. A., & Cárdenas, E. J. E. (2021). Análisis comparativo de métodos de machine learning para clasificar opiniones sobre el servicio de restaurantes peruanos en Facebook. *Actas del Congreso Internacional de Ingeniería de Sistemas*, 67-81. https://doi.org/10.26439/ciis2021.5578

Torres, V. A. G., Cardenas, R. F. C., & Coaguila, V. A. R. (2022). Clasificación de tutoriales en YouTube basándonos en el análisis de sentimientos realizados a sus comentarios. *Innovación y Software*, *3*(2), Article 2. https://doi.org/10.48168/innosoft.s9.a66

Tristán Gomez, L. E. (2019). Modelo Predictivo del Índice NPS Basado en Información Textual de Percepción del Servicio al Cliente. *Repositorio institucional - URP*. https://repositorio.urp.edu.pe/handle/20.500.14138/2480

Vadillo, J., & Santana, R. (2019). Universal adversarial examples in speech command classification. *arXiv preprint arXiv:1911.10182*.

Vangara, V., & Vangara, S. (2020). A Survey on Natural Language Processing in context with Machine Learning. *The International journal of analytical and experimental modal analysis*, *12*, 1390-1395.

Vincent, J. (2022, junio 30). *YouTube is cracking down on tricks that spammers use to impersonate creators*. The Verge.

https://www.theverge.com/2022/6/30/23189367/youtube-spam-account-policy-subscriber-count-special-characters

Vlachidis, A. (2012). *Semantic indexing via knowledge organization systems: Applying the CIDOC-CRM to archaeological grey literature*. University of South Wales (United Kingdom).

Wang, H., & Zheng, H. (2013). Model Testing, Machine Learning. En W. Dubitzky, O. Wolkenhauer, K.-H. Cho, & H. Yokota (Eds.), *Encyclopedia of Systems Biology* (pp. 1405-1405). Springer. https://doi.org/10.1007/978-1-4419-9863-7_231

Weedmark, D. (2021). *Machine Learning Model Training: What It Is and Why It's Important*. https://www.dominodatalab.com/blog/what-is-machine-learning-model-training

Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a standard process model for data mining. *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, *1*, 29-39.

Yan, Z., & Xu, C. (2009). Studies on classification models using decision boundaries. *2009 8th IEEE International Conference on Cognitive Informatics*, 287-294.

ANEXOS

ANEXO 1: Script para la obtención de los Comentarios de un Video de YouTube

```
# -*- coding: utf-8 -*-
"""A. Get All Comments of a YouTube Video.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1UWbqSpnx9j0vUGG0zc0R-
hE7Z3Y8-h13

### Importing libraries
"""

import requests
import pandas as pd

URI = "https://youtube.googleapis.com/youtube/v3/"
api_key = "xxxxx" # paste here your own api key

"""### Getting All the Replies for a specific Comment"""

def get_replies(parent_id,  video_id):
    df_replies = pd.DataFrame(columns=["video_id", "comment_id",
"author", "comment", "class"])

    # Cuerpo (body) en formato JSON
    params = {
        "part": ["snippet"],
        "parentId": parent_id,
        "maxResults": 100,
        "pageToken": "",
        "key": api_key
    }
    # Realizar la solicitud GET a la API
    response = requests.get(URI + "comments", params=params)

    if response.status_code != 200:
        print("Error en la solicitud:", response.status_code )
        return;

    # Obtener la respuesta de la API
    data = response.json()

    for item in data["items"]:
```

```
        comment_data = item['snippet']

        comment_id = item['id']
        author = comment_data['authorDisplayName']
        comment = comment_data['textOriginal']
        #print(author, comment)

        row = {
            "video_id": [video_id],
            "comment_id": [comment_id],
            "author": [author],
            "comment": [comment],
            "class": [0],
        }
        df_replies = pd.concat([df_replies, pd.DataFrame(row)],
ignore_index=True)

    return df_replies

"""### Getting All the Comments (top level and their replies) for a
specific Video"""

def get_comments(video_id):

    df_comments = pd.DataFrame(columns=["video_id", "comment_id",
"author", "comment", "class"])

    nextPageToken = ""
    while True:
        # Cuerpo (body) en formato JSON
        params = {
            "part": ["snippet", "replies"],
            "videoId": video_id,
            "maxResults": 100,
            "order": "relevance",
            "pageToken": nextPageToken,
            "key": api_key
        }
        # Realizar la solicitud GET a la API
        response = requests.get(URI + "commentThreads", params=params)

        if response.status_code != 200:
            print("Error en la solicitud:", response.status_code )
            break;

        # Obtener la respuesta de la API
        data = response.json()

        for item in data["items"]:
```

```
            comment_data = item['snippet']['topLevelComment']

            comment_id = comment_data['id']
            author = comment_data['snippet']['authorDisplayName']
            comment = comment_data['snippet']['textOriginal']
            #print(author, comment)

            row = {
                "video_id": [video_id],
                "comment_id": [comment_id],
                "author": [author],
                "comment": [comment],
                "class": [0],
            }

            df_comments = pd.concat([df_comments, pd.DataFrame(row)],
ignore_index=True)

            nextPageToken = data.get("nextPageToken", "")

            total_reply_count = item["snippet"]["totalReplyCount"]

            if (total_reply_count > 0):
                parent_id = comment_data["id"]
                #print(parent_id)
                df_replies = get_replies(parent_id, video_id)
                df_comments = pd.concat([df_comments, df_replies],
ignore_index=True)

        print("nextPageToken: ", nextPageToken)
        #print("author -> ",
data["items"][0]['snippet']["topLevelComment"]["snippet"]["authorDisplayN
ame"])
        if nextPageToken == "":
            break;

    return df_comments

"""### Getting and saving the comments for a specific video"""

video_id = "1bZuVWBN1jY"

df = get_comments(video_id)

df.head()

"""### Getting and saving the comments for a list of videos"""

# List video IDs for training and testing (30 videos - muestra)
```

```
videos_id = [
    '-KnxAh2yMWM',
    '1bZuVWBN1jY',
    '2ibqfxEAESo',
    '3-4g-jBN1p4',
    '3ybL7JLBqlA',
    '45-jr2jA43E',
    '68sXnXBxSgw',
    '6orj7xbeBRc',
    'baJ5O0Kwaco',
    'CG3Om0f9jiw',
    'CQDhVL_lV3g',
    'D2wGBfh4ZaM',
    'e5xaGh4Rtqo',
    'Gy3nOLFIQRA',
    'iHcW5WfnmnU',
    'JmCRMt7lldQ',
    'k-nQV_QkNl4',
    'KgGboHohoqg',
    'lugETBcmV28',
    'mDeSYUoUTXM',
    'NkMQpa2a2Ec',
    'o1FCkAj6U7c',
    'ObqT6a7LZXc',
    'qF0gQBKg2os',
    'slkvRuMmgAk',
    'SzKnKC5BvDw',
    't1NxNInQW9Q',
    'TYfQZA4ZaXs',
    'VOf1Dc_nxCA',
    'X_h7chiQJ4g',
];

'''videos_id = [
    'TYfQZA4ZaXs',
    '1bZuVWBN1jY',
]'''

data = pd.DataFrame()

for video_id in videos_id:
    temp_df = get_comments(video_id)
    data = pd.concat([data, temp_df], ignore_index=True)

"""### Saving the DataFrame in xlsx format"""

filename = "data"

data.to_excel(filename + ".xlsx", index=False)
```

ANEXO 2: Las 145 palabras con Mayor Frecuencia de los Comentarios para ambas Clases

Nº	PALABRA	FRECUENCIA	
		NO SPAM (0)	SPAM (1)
1	de	15941	4909
2	que	14618	2656
3	en	9925	2370
4	la	8031	2916
5	el	7642	1577
6	no	7413	798
7	es	6221	1826
8	para	5207	974
9	un	4869	1247
10	me	4826	1197
11	por	4755	715
12	lo	4383	697
13	los	4363	561
14	se	3853	380
15	gracias	3716	362
16	si	3515	135
17	una	3466	978
18	con	3277	2826
19	pero	2945	415
20	como	2709	489
21	te	2684	142
22	mi	2533	1272
23	invertir	2412	349
24	las	2075	528
25	muy	2030	313
26	tengo	1980	149
27	dinero	1904	102
28	más	1888	434
29	video	1790	84
30	yo	1778	100
31	al	1743	280
32	ya	1729	72
33	del	1420	320
34	todo	1250	175
35	este	1245	187
36	años	1206	92
37	muchas	1163	288
38	hacer	1155	90
39	hola	1152	51
40	eso	1129	113

Nº	PALABRA	FRECUENCIA	
		NO SPAM (0)	SPAM (1)
41	hay	1085	101
42	mucho	1076	198
43	solo	1026	334
44	desde	1022	158
45	son	992	385
46	su	988	937
47	bien	958	140
48	puede	947	189
49	estoy	938	405
50	puedo	930	310
51	porque	907	170
52	soy	906	90
53	inversión	856	533
54	esto	850	141
55	cuando	845	251
56	cuenta	826	91
57	mejor	819	368
58	tener	791	79
59	ser	771	153
60	le	771	194
61	todos	723	141
62	cómo	712	418
63	buen	703	72
64	siempre	682	113
65	sin	669	137
66	información	656	142
67	qué	648	74
68	esta	642	276
69	ahora	633	260
70	sobre	626	327
71	está	617	203
72	así	616	75
73	tiempo	605	72
74	tiene	591	105
75	mis	580	275
76	puedes	567	141
77	hace	564	163
78	tienes	552	51
79	he	538	539
80	cada	525	133
81	poco	517	68
82	también	512	494
83	bueno	508	146

Nº	PALABRA	FRECUENCIA	
		NO SPAM (0)	SPAM (1)
84	tan	502	169
85	trabajo	491	111
86	gente	490	77
87	sus	489	677
88	vida	468	112
89	ese	466	53
90	ver	459	285
91	ha	449	609
92	inversiones	445	352
93	creo	424	109
94	mismo	419	87
95	buena	416	126
96	menos	412	60
97	vez	409	168
98	hasta	392	168
99	personas	387	140
100	uno	378	68
101	alguien	376	200
102	trading	360	59
103	nunca	355	143
104	gran	336	156
105	ti	331	56
106	ingresos	329	104
107	mundo	327	139
108	duda	324	64
109	amigo	323	73
110	decir	312	71
111	mercado	305	263
112	capital	302	64
113	favor	297	123
114	meses	290	132
115	momento	277	64
116	forma	275	93
117	negocio	267	79
118	están	264	58
119	fue	264	140
120	acciones	263	107
121	cual	262	52
122	ayuda	261	141
123	otros	260	61
124	aquí	257	176
125	buenas	254	67
126	compartir	253	91

Nº	PALABRA	FRECUENCIA	
		NO SPAM (0)	SPAM (1)
127	gustaría	246	51
128	ganar	244	85
129	ganancias	232	582
130	después	230	219
131	sé	221	53
132	realmente	217	351
133	dólares	212	62
134	financiera	206	73
135	estado	195	122
136	recomiendo	195	65
137	era	193	110
138	idea	190	100
139	paso	188	52
140	criptomonedas	185	265
141	hecho	185	138
142	trabajar	184	78
143	días	175	159
144	necesito	175	63
145	debe	170	70

ANEXO 3: Script para realizar la Limpieza del Conjunto de Datos

```
# -*- coding: utf-8 -*-
"""C. Data Preparation (Clean Data) - Remove NaN (empty) and duplicate
comments from the Dataset.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1EqObEWyo-
DFhCsJNK2tOipUgQeeb0j6_

### Importing libraries
"""

import pandas as pd
from collections import Counter

from google.colab import drive

# Monta tu Google Drive
drive.mount('/content/gdrive')

"""### Loading the dataset"""

ruta_archivo = '/content/gdrive/MyDrive/Thesis draft/03.
Development/Dataset/data.xlsx'

df = pd.read_excel(ruta_archivo)

df.head()

X = df["comment"].values
y = df["class"].values

"""### Removing NaN (empty) comments"""

num_init_records = len(df)

nan_records = df['comment'].isna()

num_nan_records = nan_records.sum()
print("Num. NaN records:", num_nan_records)

filas_con_nan_df = df[nan_records].reset_index(drop=True)
filas_con_nan_df

df_cleaned = df[~nan_records]
```

```
num_final_records = len(df_cleaned)

print("Num. before delete NaN message records\t:", num_init_records )
print("Num. after delete NaN message records\t:", num_final_records )

"""### Removing duplicate comments"""

num_init_records = len(df_cleaned)

duplicate_records = df_cleaned.duplicated(subset=['comment'],
keep='first')

num_duplicated_records = duplicate_records.sum()
print("Num. duplicated records:", num_duplicated_records)

filas_duplicadas_df = df_cleaned[duplicate_records]

filas_duplicadas_df

# Eliminar duplicados basados en la columna "comment"
df_salida =
filas_duplicadas_df.drop_duplicates(subset=['comment']).reset_index(drop=
True)
df_salida.head()

df_cleaned = df_cleaned[~duplicate_records]

num_final_records = len(df_cleaned)

print("Num. before delete dubplicated records\t:", num_init_records )
print("Num. after delete dubplicated records\t:", num_final_records )

df_cleaned

"""### Número de Registros por Clase (despues de la eliminación)"""

import seaborn as sns
import matplotlib.pyplot as plt

# Suponiendo que 'df' es tu DataFrame y 'class' es la columna de interés

# Contar la cantidad de valores 0 y 1 en la columna 'class'
class_counts = df_cleaned['class'].value_counts()

# Crear un gráfico circular (gráfico de pastel)
plt.figure(figsize=(6, 6))
labels = ['0 (no spam)', '1 (spam)']
```

```
plt.pie(class_counts, labels=labels, autopct=lambda p:
'{:.0f}\n({:.1f}%)'.format(p * sum(class_counts) / 100, p),
colors=['#B2E672', '#FF8989'])
plt.show()

"""### Saving the DataFrame in xlsx format"""

filename = "data_cleaned"

df_cleaned.to_excel(filename + ".xlsx", index=False)
```

ANEXO 4: Script para realizar el splitting, balanceado y formateo del Conjunto de Datos

```
# -*- coding: utf-8 -*-
"""D. Data Preparation (Construct Data) - Dataset Splitter (Training -
Testing) - Oversampling the Training Set.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1IBtIySbEc2Rk0gTh0zbpf88AKHneMNNe

### Importing libraries
"""

!pip install -U imbalanced-learn

import json
import io
import pandas as pd
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import RandomOverSampler
from collections import Counter

from google.colab import drive

# Monta tu Google Drive
drive.mount('/content/gdrive')

"""### Loading the dataset"""

ruta_archivo = '/content/gdrive/MyDrive/Thesis draft/03.
Development/Dataset/data_cleaned.xlsx'

df = pd.read_excel(ruta_archivo)

df.head()

X = df["comment"].values
y = df["class"].values

print("=====  Dataset =====")
num_samples_by_class = Counter(y)

num_spam_samples = num_samples_by_class[1]
num_ham_samples = num_samples_by_class[0]
```

```
print(f'Ham samples\t: {num_ham_samples}')
print(f'Spam samples\t: {num_spam_samples}')

"""### Splitting the Dataset (Training 80% and Testing 20% Datasets)"""

# Dividir el DataFrame en dos grupos según la clase
ham_samples = df[df["class"] == 0]
spam_samples = df[df["class"] == 1]

# to make sure that our training and testing dataset have samples of
both: ham and spam
train_ham_samples, test_ham_samples = train_test_split(ham_samples,
test_size=0.2)
train_spam_samples, test_spam_samples = train_test_split(spam_samples,
test_size=0.2)

train_samples = pd.concat([train_ham_samples, train_spam_samples])
test_samples = pd.concat([test_ham_samples, test_spam_samples])

X_train = train_samples["comment"].values
y_train = train_samples["class"].values

X_test = test_samples["comment"].values
y_test = test_samples["class"].values

print("===== Training set (80%) =====")
num_samples_by_class = Counter(y_train)

num_spam_samples = num_samples_by_class[1]
num_ham_samples = num_samples_by_class[0]

print(f'Num. samples\t: {len(y_train)}')
print(f'Ham samples\t: {num_ham_samples}')
print(f'Spam samples\t: {num_spam_samples}')

print("===== Testing set (20%) =====")
num_samples_by_class = Counter(y_test)

num_spam_samples = num_samples_by_class[1]
num_ham_samples = num_samples_by_class[0]

print(f'Num. samples\t: {len(y_test)}')
print(f'Ham samples\t: {num_ham_samples}')
print(f'Spam samples\t: {num_spam_samples}')

"""### Balancing the Traninig set (Oversampling)"""

print("===== Before Balancing Training set (80%) =====")
num_samples_by_class = Counter(y_train)
```

```
num_spam_samples = num_samples_by_class[1]
num_ham_samples = num_samples_by_class[0]

print(f'Spam samples\t: {num_spam_samples}')
print(f'Ham samples\t: {num_ham_samples}')

ros = RandomOverSampler(random_state=42, sampling_strategy='minority')

X_train_balanced, y_train_balanced = ros.fit_resample(X_train.reshape(-1,
1), y_train)

print("===== After Balancing Training set (80%) =====")
num_samples_by_class = Counter(y_train_balanced)

num_spam_samples = num_samples_by_class[1]
num_ham_samples = num_samples_by_class[0]

print(f'Spam samples\t: {num_spam_samples}')
print(f'Ham samples\t: {num_ham_samples}')

"""### Exporting the Split Dataset in JSON Format"""

X_train_balanced = X_train_balanced.flatten().tolist()
y_train_balanced = y_train_balanced.flatten().tolist()
X_test = X_test.flatten().tolist()
y_test = y_test.flatten().tolist()

# Creating training set
training_set = {
    "comments": X_train_balanced,
    "classes": y_train_balanced
}

# Creating training set
testing_set = {
    "comments": X_test,
    "classes": y_test
}

# Convertir el diccionario en JSON
json_training_set = json.dumps(training_set, ensure_ascii=False)

json_testing_set = json.dumps(testing_set, ensure_ascii=False)

"""### Saving the Training and Testing dataset in on .json file"""

# Save training_set in JSON format
with io.open('training_set.json', 'w', encoding='utf-8') as archivo:
```

```
    archivo.write(json_training_set)

print("JSON generado y descargado correctamente.")

# Save training_set in JSON format
with io.open('testing_set.json', 'w', encoding='utf-8') as archivo:
    archivo.write(json_testing_set)

print("JSON generado y descargado correctamente.")
```

ANEXO 5: Código fuente de la implementación de la Clase de preprocesamiento de Texto

```
const LANG = "es";
const stopwords = require(`./util/stopwords-${LANG}`);
const stemmer = require(`./util/PorterStemmer${LANG.toUpperCase()}`);

class TextPreprocessor {
  constructor() {}
  // cleaner
  cleaner = (text) => {
    // text to lower case
    let cleaned_text = text.toLowerCase();

    // remove @ mentions
    cleaned_text = cleaned_text.replace(/@[^@\s]*/g, " ");

    // remove \n;
    cleaned_text = cleaned_text.replace(/\\n/g, " ");

    // remove \";
    cleaned_text = cleaned_text.replace(/\\\"/g, " ");

    // remove punctuations signs
    cleaned_text =
cleaned_text.replace(/[/*;^(%)$_,…@:."\\><'¡!¿~©?#’]/g, "");

    // remove individual letters
    cleaned_text = cleaned_text.replace(/(^|\s)[a-zñ](?=\s|$)/g, " ");

    // remove numbers
    cleaned_text = cleaned_text.replace(/[0-9]/g, " ");

    // replace a secuence of whitespaces with one whitespace
    cleaned_text = cleaned_text.replace(/\s+/g, " ");

    // remove the whitespaces in the begin and end
    cleaned_text = cleaned_text.trim();

    return cleaned_text;
  };

  tokenizer = (text) => {
    const words = text.split(/\s+/); // it's not necesary to use the g
flag in this method

    return words;
  };
```

```
  stopWordsRemover = (tokens) => {
    const cleaned_tokens = [];
    for (const token of tokens) {
      // aqui sucede algo extraño, como tal hay un simbolo dentro de de
las comillas pero no se cual es, esto permite no inclurir tokens vacios
      if (stopwords.indexOf(token) === -1 && token != "") {
        cleaned_tokens.push(token);
      }
    }

    return cleaned_tokens;
  };

  stemmer = (tokens) => {
    const stemmed_tokens = [];

    for (let token of tokens) {
      // ref:
https://github.com/NaturalNode/natural/blob/c2389f2a17faae9582a1a0f41402c
68407ccb378/lib/natural/stemmers/stemmer_es.js#L41C13-L41C13
      if (token.match(/[a-záéíóúüñ0-9]+/gi)) {
        token = stemmer.stem(token);
      }

      stemmed_tokens.push(token);
    }

    return stemmed_tokens;
  };

  preprocess = (text) => {
    // cleaning
    const cleaned_text = this.cleaner(text);

    // tokenizing
    const tokens = this.tokenizer(cleaned_text);

    // removing stop words
    const cleaned_tokens = this.stopWordsRemover(tokens);

    // stemming
    const stemmed_tokens = this.stemmer(cleaned_tokens);

    return stemmed_tokens;
  };
}

const text_preprocessor = new TextPreprocessor();
```

```
//console.log(text_preprocessor.preprocess(`Te<>lgr^)))))m`));

module.exports = text_preprocessor;
```

ANEXO 6: Script para el Formateo de conjunto de datos de Entrenamiento y Prueba

```
const fs = require("fs");
const text_preprocessor = require("./TextPreprocessor");

/**
 * Allows you to load a json file into a js variable (object)
 * @param {string} filename Name of the json file
 * @returns An object with the data of the json file
 */
const loadJSON = (filename) => {
  let dataset = fs.readFileSync("./inputs/" + filename, "utf-8");
  dataset = JSON.parse(dataset);

  return dataset;
};

/**
 * Allows you to save a json (js object) into a .json file
 * @param {string} filename Name of the json file
 * @param {object} data The data to save into the json file
 */
const saveJSON = (filename, data) => {
  const json_data = JSON.stringify(data);

  try {
    fs.writeFileSync(filename, json_data, "utf-8");
    console.log("File saved successfully.");
  } catch (error) {
    console.error("Error to write the file:", error);
  }
};

/**
 * Allows you to preprocesses an array of comments, cleninng, stopwords
removing, stemming, etc
 * @param {array} comments Array of comments
 * @returns An array with the comments preprocessed (tokens)
 */
const preprocessComments = (comments) => {
  const preprocessed_comments = [];
  for (const comment of comments) {
    preprocessed_comments.push(text_preprocessor.preprocess(comment));
  }

  return preprocessed_comments;
};

const main = () => {
```

```
  let training_set = loadJSON("training_set.json");
  let testing_set = loadJSON("testing_set.json");

  const x_train = preprocessComments(training_set.comments);
  const y_train = training_set.classes;

  const x_test = preprocessComments(testing_set.comments);
  const y_test = testing_set.classes;

  saveJSON("./inputs/training_set_formated.json", { x: x_train, y:
y_train });
  saveJSON("./inputs/testing_set_formated.json", { x: x_test, y: y_test
});

  return true;
};

main();
```

ANEXO 7: Código fuente de la Implementación del Algoritmo Naive Bayes Multinomial

```
const fs = require("fs");

class MultinomialNB {
  constructor(model) {
    this.laplace_smoothing = 1; // constant, no change

    if (model) {
      this.frequency = model.frequency;
      this.class_names = model.class_names;
      this.vocabulary = model.vocabulary;
      this.log_prior = model.log_prior;
      this.log_likelihood = model.log_likelihood;
    } else {
      this.frequency = {}; // contains words (also objects) and inside
each word, the frecuency of that word in each class
      this.class_names = [];
      this.vocabulary = [];
      this.log_prior = {}; // to save log_prior of each class
      this.log_likelihood = {}; // la key prodria ser una palabra y
adentro un objeto con las clases
    }
  }

  /* X sera un arrays de arrays, e y solo sera un array, ambos ya deben
haber sido preprocesados */
  fit(X, y) {
    if (X.length != y.length)
      console.log("Error, the length of X and y are diferent.");

    // convert all the classes to string values (even if there are
already strings)
    y = y.map(function (numero) {
      return numero.toString();
    });

    this.class_names = [...new Set(y)];
    const num_documents = X.length;

    // calculating frequency and vocabulary
    for (const [index, y_i] of y.entries()) {
      for (const word of X[index]) {
        if (!this.vocabulary.includes(word)) {
          this.vocabulary.push(word);
        }
        this.frequency[word] = this.frequency[word] ?? {};
```

```
        this.frequency[word][y_i] = (this.frequency[word]?.[y_i] ?? 0) +
1;
      }
    }

    for (const class_name of this.class_names) {
      const num_documents_class_name = y.filter(
        (item) => item === class_name
      ).length;

      this.log_prior[class_name] = Math.log(
        num_documents_class_name / num_documents
      );

      // calculating the denominator of log_likelihood
      let denominator = 0;
      for (const word of this.vocabulary) {
        denominator += this.frequency[word]?.[class_name] ?? 0;
      }
      denominator += this.vocabulary.length * this.laplace_smoothing;

      for (const word of this.vocabulary) {
        this.log_likelihood[word] = this.log_likelihood[word] ?? {};
        this.log_likelihood[word][class_name] =
          ((this.frequency[word]?.[class_name] ?? 0) +
this.laplace_smoothing) /
          denominator;
      }
    }
  }

  predict(X) {
    const preds = [];

    for (const x of X) {
      let pred = null;
      const scores = {};

      for (const class_name of this.class_names) {
        //scores[class_name] = this.log_prior[class_name];
        scores[class_name] = 0;

        for (const word of x) {
          if (this.vocabulary.includes(word)) {
            scores[class_name] += this.log_likelihood[word][class_name];
          }
        }
      }
```

```
      // const max_probability = Math.max(...Object.values(scores));
      // pred = Object.keys(scores).find((key) => scores[key] ===
max_probability);

      let threshold = 2.5;
      pred = scores["1"] / scores["0"] >= threshold ? "1" : "0";

      preds.push(pred);
    }

    return preds;
  }

  /**
   * Save the current model
   * @param {string} filename File name (.json) to save the model
   */
  save(filename) {
    const model = {
      name: "MultinomialNB",
      frequency: this.frequency,
      class_names: this.class_names,
      vocabulary: this.vocabulary,
      log_prior: this.log_prior,
      log_likelihood: this.log_likelihood,
    };
    const modelJSON = JSON.stringify(model);

    try {
      fs.writeFileSync(filename, modelJSON, "utf-8");
      console.log("File saved successfully.");
    } catch (error) {
      console.error("Error to write the file:", error);
    }
  }

  /**
   * Creates a new MBN model fron the givin json file, static method, I
can be call without any instance
   * @param {string} filename Filename of model in .json format
   * @returns {MBN}
   */
  static load(filename) {
    let model;

    // https://stackoverflow.com/a/14078644
    try {
      const data = fs.readFileSync(filename, "utf-8");
      model = JSON.parse(data);
```

```
    } catch (error) {
      console.error("Error reading the file:", error);
    }

    return new MultinomialNB(model);
  }
}

module.exports = MultinomialNB;
```

ANEXO 8: Script para la generación del Modelo de clasificación Naive Bayes Multinomial

```
const fs = require("fs");
const MultinomialNB = require("./MultinomialNB");

/**
 * Allows you to load a json file into a js variable (object)
 * @param {string} filename Name of the json file
 * @returns An object with the data of the json file
 */
const loadJSON = (filename) => {
  let dataset = fs.readFileSync("./inputs/" + filename, "utf-8");
  dataset = JSON.parse(dataset);

  return dataset;
};

// Main to generate a model
const main = () => {
  let { x: x_train, y: y_train } =
loadJSON("training_set_formated.json");

  // ======= TRAINING =======
  const classifier = new MultinomialNB();

  classifier.fit(x_train, y_train);

  classifier.save("outputs/model.json");

  return true;
};

main();
```

ANEXO 9: Script para la evaluación del Modelo de clasificación Naive Bayes Multinomial

```
const fs = require("fs");
const MultinomialNB = require("./MultinomialNB");

/**
 * Allows you to load a json file into a js variable (object)
 * @param {string} filename Name of the json file
 * @returns An object with the data of the json file
 */
const loadJSON = (filename) => {
  let dataset = fs.readFileSync("./inputs/" + filename, "utf-8");
  dataset = JSON.parse(dataset);

  return dataset;
};

/**
 * Allows you to get the values of the confussion matrix
 * @param {array} y_pred Predictions made by the model
 * @param {array} y_test Real classes
 * @returns The values of the confussion matrix
 */
const calculateConfussionMatrix = (y_pred, y_test) => {
  let TP = 0;
  let FN = 0;
  let FP = 0;
  let TN = 0;

  if (y_pred.length !== y_test.length) {
    throw new Error("The arrays must have the same length");
  }

  for (let i = 0; i < y_test.length; i++) {
    if (y_test[i] == 1) {
      if (y_pred[i] == 1) TP++;
      else FN++;
    } else {
      if (y_pred[i] == 1) FP++;
      else TN++;
    }
  }

  return { TP, FN, FP, TN };
};

/**
```

```
 * Allows you to get the evaluate the result of the model
 * @param {array} y_pred Predictions make by the model
 * @param {array} y_test Real classes
 * @returns The accuracy performance
 */
const calculateAccuracy = (y_pred, y_test) => {
  if (y_pred.length !== y_test.length) {
    throw new Error("The arrays must have the same length");
  }

  const { TP, FN, FP, TN } = calculateConfussionMatrix(y_pred, y_test);
  const accuracy = (TP + TN) / (TP + TN + FP + FN);

  return accuracy;
};

const calculatePrecision = (y_pred, y_test) => {
  if (y_pred.length !== y_test.length) {
    throw new Error("The arrays must have the same length");
  }

  const { TP, FN, FP, TN } = calculateConfussionMatrix(y_pred, y_test);
  const precision = TP / (TP + FP);

  return precision;
};

const calculateRecall = (y_pred, y_test) => {
  if (y_pred.length !== y_test.length) {
    throw new Error("The arrays must have the same length");
  }

  const { TP, FN, FP, TN } = calculateConfussionMatrix(y_pred, y_test);
  const recall = TP / (TP + FN);

  return recall;
};

const calculateF1Score = (y_pred, y_test) => {
  if (y_pred.length !== y_test.length) {
    throw new Error("The arrays must have the same length");
  }

  const precision = calculatePrecision(y_pred, y_test);
  const recall = calculateRecall(y_pred, y_test);

  const f1_score = 2 * ((precision * recall) / (precision + recall));

  return f1_score;
```

```
};

// Main to evalute the generate model
const main = () => {
  let { x: x_test, y: y_test } = loadJSON("testing_set_formated.json");

  // ======= LOADING =======
  const classifier = MultinomialNB.load("outputs/model.json");

  // ======= TESTING =======
  const y_pred = classifier.predict(x_test);

  // calculating confussion matrix
  const { TP, FN, FP, TN } = calculateConfussionMatrix(y_pred, y_test);
  console.log("Confussion Matrix:");
  console.log("-> TP:", TP);
  console.log("-> FN:", FN);
  console.log("-> FP:", FP);
  console.log("-> TN:", TN);

  // calculating metrics
  const precision = calculatePrecision(y_pred, y_test);
  const recall = calculateRecall(y_pred, y_test);
  const f1_score = calculateF1Score(y_pred, y_test);

  // displaying metrics
  console.log("Metrics:");
  console.log("-> Precision:", precision);
  console.log("-> Recall:", recall);
  console.log("-> F1 score:", f1_score);

  return true;
};

main();
```

ANEXO 10: Declaración jurada de autenticidad de tesis

Universidad Nacional del Altiplano Puno | VRI Vicerrectorado de Investigación | Repositorio Institucional

DECLARACIÓN JURADA DE AUTENTICIDAD DE TESIS

Por el presente documento, Yo DEYVIS MAMANI LACUTA,
identificado con DNI 73905791 en mi condición de egresado de:

☒ **Escuela Profesional,** ☐ **Programa de Segunda Especialidad,** ☐ **Programa de Maestría o Doctorado**
INGENIERÍA DE SISTEMAS,
informo que he elaborado el/la ☒ **Tesis** o ☐ **Trabajo de Investigación** denominada:
" MODELO NAIVE BAYES MULTINOMIAL PARA LA CLASIFICACIÓN DE COMENTARIOS SPAM EN ESPAÑOL DE VIDEOS SOBRE FINANZAS DE LA PLATAFORMA YOUTUBE "

Es un tema original.

Declaro que el presente trabajo de tesis es elaborado por mi persona y **no existe plagio/copia** de ninguna naturaleza, en especial de otro documento de investigación (tesis, revista, texto, congreso, o similar) presentado por persona natural o jurídica alguna ante instituciones académicas, profesionales, de investigación o similares, en el país o en el extranjero.

Dejo constancia que las citas de otros autores han sido debidamente identificadas en el trabajo de investigación, por lo que no asumiré como suyas las opiniones vertidas por terceros, ya sea de fuentes encontradas en medios escritos, digitales o Internet.

Asimismo, ratifico que soy plenamente consciente de todo el contenido de la tesis y asumo la responsabilidad de cualquier error u omisión en el documento, así como de las connotaciones éticas y legales involucradas.

En caso de incumplimiento de esta declaración, me someto a las disposiciones legales vigentes y a las sanciones correspondientes de igual forma me someto a las sanciones establecidas en las Directivas y otras normas internas, así como las que me alcancen del Código Civil y Normas Legales conexas por el incumplimiento del presente compromiso

Puno 16 de JULIO del 2024

FIRMA (obligatoria)

Huella

ANEXO 11: Autorización para el depósito de tesis en el Repositorio Institucional

Universidad Nacional del Altiplano Puno | VRI Vicerrectorado de Investigación | Repositorio Institucional

AUTORIZACIÓN PARA EL DEPÓSITO DE TESIS O TRABAJO DE INVESTIGACIÓN EN EL REPOSITORIO INSTITUCIONAL

Por el presente documento, Yo DEYVIS MAMANI LACUTA,
identificado con DNI 73905791 en mi condición de egresado de:

☒ **Escuela Profesional,** ☐ **Programa de Segunda Especialidad,** ☐ **Programa de Maestría o Doctorado**

INGENIERÍA DE SISTEMAS,

informo que he elaborado el/la ☒ **Tesis** o ☐ **Trabajo de Investigación** denominada:

" MODELO NAIVE BAYES MULTINOMIAL PARA LA CLASIFICACIÓN DE COMENTARIOS SPAM EN ESPAÑOL DE VIDEOS SOBRE FINANZAS DE LA PLATAFORMA YOUTUBE ",

para la obtención de ☐**Grado,** ☒ **Título Profesional** o ☐ **Segunda Especialidad.**

Por medio del presente documento, afirmo y garantizo ser el legítimo, único y exclusivo titular de todos los derechos de propiedad intelectual sobre los documentos arriba mencionados, las obras, los contenidos, los productos y/o las creaciones en general (en adelante, los "Contenidos") que serán incluidos en el repositorio institucional de la Universidad Nacional del Altiplano de Puno.

También, doy seguridad de que los contenidos entregados se encuentran libres de toda contraseña, restricción o medida tecnológica de protección, con la finalidad de permitir que se puedan leer, descargar, reproducir, distribuir, imprimir, buscar y enlazar los textos completos, sin limitación alguna.

Autorizo a la Universidad Nacional del Altiplano de Puno a publicar los Contenidos en el Repositorio Institucional y, en consecuencia, en el Repositorio Nacional Digital de Ciencia, Tecnología e Innovación de Acceso Abierto, sobre la base de lo establecido en la Ley N° 30035, sus normas reglamentarias, modificatorias, sustitutorias y conexas, y de acuerdo con las políticas de acceso abierto que la Universidad aplique en relación con sus Repositorios Institucionales. Autorizo expresamente toda consulta y uso de los Contenidos, por parte de cualquier persona, por el tiempo de duración de los derechos patrimoniales de autor y derechos conexos, a título gratuito y a nivel mundial.

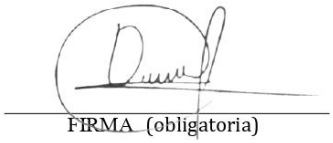
En consecuencia, la Universidad tendrá la posibilidad de divulgar y difundir los Contenidos, de manera total o parcial, sin limitación alguna y sin derecho a pago de contraprestación, remuneración ni regalía alguna a favor mío; en los medios, canales y plataformas que la Universidad y/o el Estado de la República del Perú determinen, a nivel mundial, sin restricción geográfica alguna y de manera indefinida, pudiendo crear y/o extraer los metadatos sobre los Contenidos, e incluir los Contenidos en los índices y buscadores que estimen necesarios para promover su difusión.

Autorizo que los Contenidos sean puestos a disposición del público a través de la siguiente licencia:

Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional. Para ver una copia de esta licencia, visita: https://creativecommons.org/licenses/by-nc-sa/4.0/

En señal de conformidad, suscribo el presente documento.

Puno 16 de JULIO del 2024



FIRMA (obligatoria)

Huella