Modelo para la detección de noticias falsas en formato texto en la red social Twitter, aplicado al contexto político colombiano de las elecciones presidenciales de 2022

PROYECTO DE GRADO

Jesús Eduardo Flores Quinayás
Juan Gabriel Montaño Morcillo

Asesores

Aníbal Sosa
Andrés Aristizábal

FACULTAD DE INGENIERÍA
MAESTRÍA EN CIENCIA DE DATOS
SANTIAGO DE CALI
2022

Modelo para la detección de noticias falsas en formato texto en la red social Twitter, aplicado al contexto político colombiano de las elecciones presidenciales de 2022

Jesús Eduardo Flores Quinayás
Juan Gabriel Montaño Morcillo

Asesores
Aníbal Sosa
Andrés Aristizábal

UNIVERSIDAD ICESI
FACULTAD DE INGENIERÍA
MAESTRÍA EN CIENCIA DE DATOS
SANTIAGO DE CALI
2022

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

RESUMEN

En este trabajo de grado, se formuló una propuesta para abordar la ausencia de modelos de detección de noticias falsas en el contexto político colombiano en la red social Twitter. Para desarrollarlo se tomó como referencia la metodología CRISP-DM que considera seis fases para su desarrollo, estas fases fueron: entendimiento del negocio, entendimiento de los datos, preparación de los datos, modelamiento, evaluación y despliegue.

Los modelos, técnicas y herramientas de la ciencia de datos utilizados para abordar la solución del problema fueron: Random Forest, Naive Bayes, Support Vector Classifier, Regresión Logística, XG-Boost, redes neuronales tradicionales y modelos basados en atención como BERT.

De los resultados obtenidos en el entrenamiento de los modelos de analítica utilizando modelos tradicionales y BERT, se alcanzó un accuracy de 0.88 y 0.95 respectivamente para el corpus creado en el contexto político colombiano. El uso de las aproximaciones por modelos tradicionales y los modelos de atención con BERT permitió comparar el comportamiento en el desempeño del proceso de la clasificación de noticias, mostrando que con BERT se obtienen mejores resultados.

La validación a la que fue sometida la propuesta consistió en la realización de varios experimentos con dos corpus de noticias diferentes; el primero fue el creado en el proyecto con noticias colombianas y el segundo lo conformó la combinación del corpus de noticias colombianas con dos corpus adicionales de contextos diferentes al colombiano. En las validaciones se realizaron ajustes de parámetros e hiperparámetros buscando obtener mejores resultados en el proceso de clasificación de noticias falsas.

Finalmente, después del trabajo desarrollado y de las validaciones realizadas se logró formular un modelo de analítica capaz de detectar noticias falsas en el contexto político colombiano con un buen nivel de desempeño, validando de esta manera que el enfoque de la solución propuesto y la metodología empleada permitieron obtener resultados válidos.

1. INTRODUCCIÓN

1.1 Contexto y antecedentes

Las redes sociales hoy en día se han convertido en medios que permiten interactuar con otras personas, a la vez que posibilitan estar informado sobre las noticias de la actualidad y las tendencias. Sin embargo, también son focos o fuentes de noticias falsas que pueden propagarse rápidamente (Vosoughi et al., 2018), afectando de diferentes maneras a la sociedad. Tales afectaciones van desde la polarización de la opinión pública, influencia en la toma de decisiones, afectaciones psicológicas e inclusive afectaciones a niveles organizacionales llegando a ocasionar daños a la imagen y reputación.

Las noticias falsas se vuelven más persuasivas para la sociedad cuando se trata de temas políticos, debido a la capacidad de éstas para influenciar en la opinión de las personas para la toma de decisiones tal y como ocurrió en las elecciones del año 2016, en donde la estrategia de propagación de noticias falsas en redes sociales como Twitter y Facebook ayudó al entonces candidato Donald Trump llegar a la presidencia de los Estados Unidos (Allcott & Gentzkow, 2017).

La sociedad colombiana no es ajena a esta problemática, en los últimos años, el crecimiento del consumo digital ha causado que las noticias falsas sean más fáciles de crear y propagar en redes sociales, tales engaños han llevado a crear incertidumbre en la comunidad y desinformación.

1.2 Justificación

El procesamiento del lenguaje natural (PLN) a través de modelos de aprendizaje maquina han permitido solucionar varios problemas, en especial en la identificación de noticias falsas. Sin embargo, la mayoría de los trabajos en el campo de PLN aplicados a la detección de noticias falsas en formato texto se encuentran para el lenguaje inglés y los encontrados en español se encuentran en contextos diferentes al colombiano.

Teniendo en cuenta el contexto, antecedentes y justificación, este trabajo de grado presenta una propuesta para la formulación de un modelo de detección de noticias falsas en formato texto que posteriormente es utilizado para detectar noticias falsas en la red social Twitter. El proyecto inicia con la construcción de un corpus propio de noticias colombianas para posteriormente aprovechar las capacidades del aprendizaje automático aplicadas al PLN.

1.3 Planteamiento del Problema

El problema que será abordado en este proyecto de grado parte de la ausencia de modelos de detección de noticias falsas en el contexto político colombiano. Entre las causas principales del problema se encuentra, que no existe un corpus en el contexto político colombiano para el entrenamiento de modelos, ya que la mayoría de los corpus se encuentran en el lenguaje inglés (Martínez-Gallego et al., 2021) y los que se encuentran en

español pertenecen a contextos y temáticas diferentes lo que dificulta poderlos utilizar para entrenar modelos de detección de noticias falsas en un contexto político colombiano.

Las principales consecuencias de esta problemática son: la desinformación, la polarización de la opinión pública, las afectaciones psicológicas, toma de malas decisiones y afectaciones a la reputación.

Lo que se busca con este proyecto de grado es implementar un modelo de analítica que permita validar si un modelo entrenado con noticias del contexto político colombiano permite obtener buenos resultados en la detección de noticias falsas en Colombia sobre la red social Twitter.

De acuerdo con lo anterior se realiza la construcción de un corpus de noticias falsas y verdaderas en el contexto político colombiano. La identificación de la noticia se realiza siguiendo un enfoque basado en el estilo que se centra en analizar la manera en cómo está escrito el texto y partiendo del hecho de que una noticia falsa en su sintaxis es muy diferente a una noticia verdadera; patrón que puede ser aprovechado por lo modelos de analítica para realizar las detecciones.

1.4 Objetivo General

Implementar un modelo de analítica que permita la detección automática de noticias falsas de tweets en español y que pueda ser utilizado para la detección de noticias falsas en las elecciones presidenciales colombianas del año 2022.

1.5 Objetivos Específicos

- Construir un corpus de noticias falsas y verdaderas para el entrenamiento de un modelo de detección de noticias falsas.

- Formular un modelo de analítica para la detección de noticias falsas en el contexto político colombiano.

- Complementar el corpus construido tomando como referencia las bases de datos disponibles de noticias falsas en español para entrenar un conjunto de modelos, con el fin de validar el desempeño en la clasificación de noticias en el contexto político colombiano.

- Validar de un conjunto de modelos de analítica, el mejor modelo para la detección de noticias falsas en el contexto político colombiano.

2. ANTECEDENTES

2.1 Marco Teórico

En el marco teórico se abordan los elementos más importantes en el contexto de las noticias falsas que se deben conocer para entender el problema y la estrategia de solución. Por esta razón, los conceptos del marco teórico serán abordados en dos subsecciones. La primera, orientada a los conceptos claves en el dominio del problema. Y la segunda, orientada a dar claridad a las ideas principales en el dominio de la solución en el campo de la Ciencia de Datos.

2.1.1 Noticias Falsas

Las noticias falsas intentan presentar una mentira como verdad y tienen como objetivo llegar a un gran número de personas, con el fin de desinformar e influir en sus opiniones. Normalmente las noticias falsas son conocidas en inglés como “Fake News” y pueden ser difundidas en diferentes medios de comunicación; medios impresos, digitales y en especial las redes sociales.

Las noticias falsas tienen un gran impacto en diferentes contextos dentro de la sociedad, tal es el caso del político en donde según (Pablo de la Varga, 2019), el mayor flujo de movimiento de información falsa se genera cuando se aproximan elecciones. Un ejemplo claro fue lo ocurrido en Estados Unidos en las elecciones a la presidencia de 2016, en donde el candidato republicano Donald Trump obtuvo el máximo cargo gracias a la manipulación de la verdad, ya que de acuerdo con el portal Politifact (Angie Drobnic Holan & Aaron Sharockman, 2016), el 70% de sus declaraciones eran bastante engañosas, falsas o grandes mentiras.

Con el auge del internet y de las redes sociales las noticias falsas han adquirido una nueva dimensión de alcance universal que permiten expandir y viralizar rápidamente una noticia falsa y desinformar a una mayor cantidad de personas velozmente.

2.1.2 Características de las noticias falsas

De acuerdo con (Rosario Peiró, 2020), estas son algunas de las características más relevantes que se pueden utilizar para identificar una noticia falsa:

- Son noticias que pretenden apelar a los sentimientos, y no a la razón, para que el lector simpatice con ellas y las comparta masivamente.
- Tienen como finalidad crear desinformación en la sociedad.
- Contienen encabezados con titulares llamativos, exagerados e incluso en algunos casos pueden ser grotescos. Buscan generar un estado emocional en el lector para atraparlo. Es común encontrarse casos en los que los lectores no leen el cuerpo de la noticia.
- Las imágenes que acompañan este tipo de noticias suelen ser falsas, extraídas a partir de otras informaciones, o añadidas para dar consistencia a la noticia.
- Los vídeos que a veces las acompañan son montajes, y poco tienen que ver con la realidad de los hechos.
- La mayoría de las noticias proceden de sitios web ficticios diseñados para parecer fiables, pero que no tienen demasiada consistencia y veracidad.

2.1.3 Categorías de noticias falsas

Según (Claire Wardle, 2017), se establecen siete categorías para la clasificación de noticias falsas dependiendo del grado de intención que tengan sobre el lector. Estas categorías son:

- Sátira o parodia: tiene como objetivo principal causar gracia al lector, pero pueden inducir al error de interpretación, ya que su estructura es similar a una noticia auténtica.
- Conexión falsa: está relacionada con el contenido de la noticia, el cual no corresponde con el título.
- Contenido engañoso: el contenido de la noticia es engañoso o falso y puede estar orientado a una persona o a un tema en particular.
- Contexto falso: el contenido de la noticia puede ser verdadero, pero el contexto en el que se ubica no lo es.
- Contenido impostor: la noticia falsa hace la suplantación de las fuentes genuinas.
- Contenido manipulado: parte de contenido verídico, pero se modifica con el fin de engañar al usuario.
- Contenido inventado: todo el contenido de la noticia es falso, estructura, contexto, información, fuentes, videos y fotografías.

2.1.4 Procesamiento del lenguaje natural

Según (Brownlee Jason, 2017), el procesamiento del lenguaje natural (PLN) o en ingles Natural Language Processing (NLP) es una rama de la inteligencia artificial que se encarga de investigar la manera de comunicar a las maquinas con las personas, mediante el uso del lenguaje natural. Básicamente se centra en encontrar mecanismos computacionalmente óptimos que faciliten la comunicación máquina humano.

Con la evolución de la computación las brechas de comunicación entre las máquinas y los humanos se han reducido considerablemente, aportando mayores progresos sobre esta rama. Tal es el caso de los chatbots y voicebot que hacen uso de PLN para poder interpretar y establecer comunicaciones naturales con los humanos. El uso del PLN no solamente está orientado a bots, también se pueden encontrar en múltiples campos, uno de ellos es la detección de noticias falsas, en donde ayudan a modelar matemáticamente el texto para que pueda ser interpretado por medio de modelos de aprendizaje automático.

2.1.5 Definición de corpus

En el campo del PLN el corpus se define como un conjunto de datos o textos codificados específicamente para que puedan ser procesados por algoritmos de aprendizaje automático para el reconocimiento e interpretación de patrones con características particulares dentro de un contexto especifico. En el caso de la detección de noticias falsas, el corpus lo compone todo el conjunto de noticias.

2.1.6 Técnicas de procesamiento del lenguaje natural en formato texto

Unas de las principales tareas del procesamiento del lenguaje natural es la transformación del lenguaje en algoritmos que le permitan a las maquinas procesar la información para modelarlos. Algunas de las técnicas más utilizadas para el procesamiento del texto se presentan a continuación.

2.1.6.1 Bag of Words

Bag of Words es una representación del texto que describe la ocurrencia de las palabras dentro de un documento[1], no importa en qué lugar del documento se encuentra la palabra. El primer paso para la creación del Bag of Words consiste en el diseño del vocabulario, tomando todas las palabras únicas de los documentos. Posteriormente se representa cada palabra del documento en un vector teniendo en cuenta si la palabra se repite o no. Un ejemplo de la representación con Bag of Words es la siguiente. Considerando 3 oraciones:

- La película estuvo buena.
- La película fue muy larga y aburrida.
- La película no gusto.

Lo primero que se crea, es el diccionario con todas las palabras únicas. Posteriormente se cuenta el número de veces que aparece la palabra de la oración en el diccionario como se muestra en la tabla:

Tabla 1. Ejemplo frecuencia de las palabras para Bag of Words

	La	película	estuvo	buena	fue	muy	larga	y	aburrida	no	gusto
oración 1	1	1	1	1	0	0	0	0	0	0	0
oración 2	1	1	0	0	1	1	1	1	1	0	0
oración 3	1	1	0	0	0	0	0	0	0	1	1

Los vectores de representación de las tres oraciones con Bag of Word son:

Vector oración 1: [1 1 1 1 0 0 0 0 0 0 0]
Vector oración 2: [1 1 0 0 1 1 1 1 1 0 0]
Vector oración 3: [1 1 0 0 0 0 0 0 0 1 1]

De esta manera el texto es representado numéricamente para el entrenamiento de los modelos.

2.1.6.2 Term Frecuency Inverse Document Frecuency (TF-IDF).

El problema de Bag of Words es que, palabras con mucha frecuencia pueden empezar a dominar en el corpus, razón por la cual surge TF-IDF, que considera la frecuencia de una palabra en todo el documento, aquellas palabras que son demasiado frecuentes son penalizadas. Este algoritmo realiza dos principales cálculos:

- Term Frecuency(TF): Asigna un puntaje de acuerdo a la frecuencia de una palabra en el documento.
- Inverse Document Frecuency (IDF): Asigna un puntaje de que tan poco común es la palabra en todos los documentos.

[1] En PLN un documento hace referencia a un elemento que puede ser una palabra, una frase una noticia, etc. El conjunto de documentos conforma lo que se conoce como corpus.

La representación matemática de TF es la siguiente:

$$tf(t,d) = \frac{número\ de\ veces\ que\ aparece\ t\ en\ d}{número\ total\ de\ terminos\ en\ d}$$

Donde t es el termino, d es el documento.

La representación matemática de IDF es:

$$idf(df) = log\frac{N}{1+df}$$

Donde N, es el número de documentos en total, df es el número de documentos que tienen el termino t.

Finalmente, el TF-IDF es el producto de TF con IDF:

$$tfidf(t,d) = tf(t,d) * idf(df)$$

De esta manera se tiene una representación numérica más limpia del texto que elimina temimos demasiado comunes que no aportan en la representación.

2.1.6.3 Word Embedding

Word Embedding es una de las técnicas utilizadas para la representación del lenguaje natural como el texto. Un Word Embedding es una representación numérica en un vector n-dimensional de una palabra. Podría representarse la palabra en ingles dog en un sistema de coordenadas de dos dimensiones como [2.5,4], la palabra en ingles cat como [3.5, 2.5], pizza como [0.5,1]. Si se representan estos vectores, se puede estimar que dog y cat son similares por tener cercanía en el espacio de dos dimensiones, ver ilustración 1. Los números asignados a los vectores son aprendidos usando algoritmos de aprendizaje maquina y grandes cantidades de datos. La proximidad entre las palabras dog y cat son similares por tener la característica de ser animales (Hagiwara, 2021).

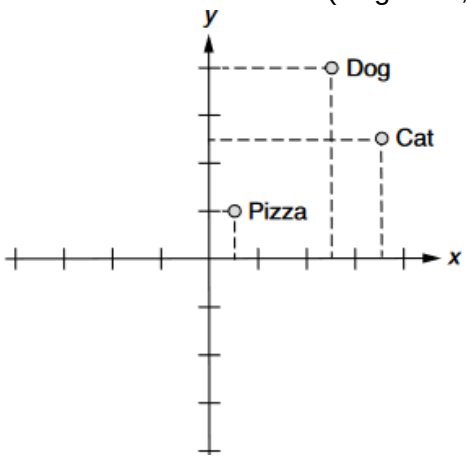

Ilustración 1. proyección en sistemas de coordenadas para word embedding (Hagiwara,2021)

2.1.7 Enfoques para la detección de noticias falsas

Según (Zhou & Zafarani, 2018), la detección de noticias falsas puede ser abordada desde varias perspectivas, entre las que se encuentran: *Knowledge-based*, *style-based*,

propagation-based y *source-based*. A continuación, se describen cada uno de estos enfoques:

- Knowledge-based: este enfoque utiliza el proceso de verificación de datos como técnica para detectar noticias falsas, básicamente compara el conocimiento extraído de la noticia con hechos conocidos para evaluar la autenticidad. Esta aproximación puede realizarse de forma manual o automática. La aproximación Knowledge-base en su forma automática es la más aplicada para detectar noticias falsas, ya que utiliza técnicas de extracción de información (IR), procesamiento natural del lenguaje (PLN) y técnicas de aprendizaje automático (ML).

  En la ilustración 2 se muestra cómo es el proceso de knowledge-based automático para la detección de noticias falsas:

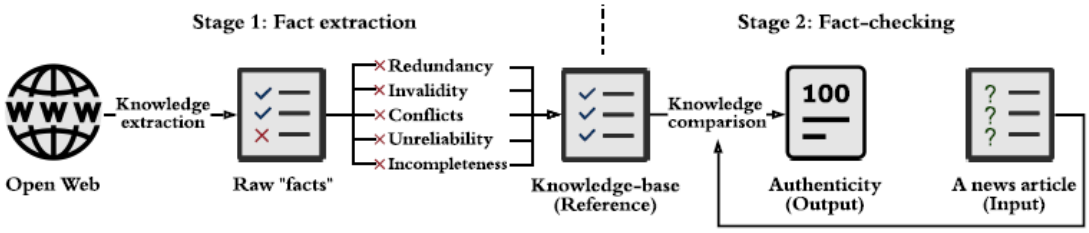

Ilustración 2. Proceso de knowledge-Based (Zhou & Zafarani, 2018)

  El proceso de detección de noticias falsas de la ilustración anterior es dividido en dos etapas: extracción de hechos; proceso donde se recogen los hechos de una fuente confiable para construir una base de conocimiento con la cual comparar, y la etapa de verificación de hechos, donde se compara la base de conocimiento extraída con las noticias por verificar su autenticidad.

- Style-based: Este enfoque es similar al Knowledge-based y hace énfasis en analizar el contenido de la noticia, pero con la diferencia en que evalúa la intención de esta, es decir, si tiene como intención o no engañar, partiendo del hecho de que las noticias falsas tienen un estilo particular que envuelve al lector y lo engaña. Según (Zhou & Zafarani, 2018), esta aproximación se define como un conjunto de características cuantificables que pueden ser procesadas por PLN y entregadas a un modelo de aprendizaje automático para determinar la clasificación de la noticia. Esta aproximación es la que tradicionalmente se utiliza en lo algoritmos de aprendizaje automático para la detección de noticias falsas.
- Propagation-based: Este enfoque está basado en la forma en cómo son difundidas las noticias falsas por los usuarios, se plantea como un problema de clasificación binaria en donde los datos de entrada al algoritmo de clasificación pueden ser de dos tipos; cascada de noticias o un gráfico de propagación y en base en esto se realiza el proceso de detección de noticias falsas.
- Source-based: Este enfoque está orientado a evaluar las noticias falsas con base a la credibilidad de la fuente, esta credibilidad se evalúa de acuerdo con el autor, editor y usuarios que podrían haber desarrollado o publicado la noticia.

2.1.8 Modelos de clasificación en el procesamiento del lenguaje natural

La identificación de noticias falsas se trata de un problema de clasificación binaria en donde se debe determinar si una noticia es falsa o no. Para poder realizar este proceso de clasificación existen diferentes modelos de aprendizaje de máquina que ayudan a esta tarea. De acuerdo con (Helmstetter & Paulheim, 2018), los siguientes, son algunos de los que presentan un mejor desempeño en la tarea de detección de noticias falsas.

2.1.8.1 Support Vector Machines (SVM)

Según(Joaquín Amat Rodrigo, 2017), SVM es un método de aprendizaje supervisado que puede ser utilizado para clasificación y regresión. El principio básico de funcionamiento de SVM consiste en fijar un margen de separación en un hiperplano que permita separar las clases para poder realizar el proceso de clasificación cuando ingresa un nuevo dato al modelo.

Algunas de las características del algoritmo se presentan a continuación:

- Efectivo en ambientes dimensionales grandes.
- Efectivo en casos donde el número de dimensiones es más grande que el número de observaciones.
- Hacen el uso de funciones kernel que permiten proyectar los datos en espacios de dimensiones superiores, sin mayores costos computacionales para la construcción de fronteras de decisión.

2.1.8.2 Naive Bayes

Según (Victor Roman, 2019), Naive Bayes es un algoritmo de aprendizaje supervisado usado para clasificación basado en el teorema de Bayes. Utiliza una suposición de independencia lineal entre los predictores. Esta suposición generalmente se viola, por este motivo se conoce como ingenuo. Sin embargo, a pesar del no cumplimiento de este supuesto está comprobado en la práctica que es un algoritmo de clasificación potente y ha sido utilizado en diversos campos, desde la detección de noticias falsas hasta los motores de recomendación.

2.1.8.3 Regresión Logística

Según (Joaquín Amat Rodrigo, 2016), es un algoritmo de clasificación que hace parte de la familia de los algoritmos de aprendizaje supervisados, en el que las observaciones se clasifican de acuerdo con una característica cuantitativa que permite determinar la clase cualitativa a la que pertenece. Es importante aclarar que a pesar de que este algoritmo permite clasificar una variable cualitativa su funcionamiento está basado en un modelo de regresión que modela el logaritmo de la probabilidad de pertenecer a una clase particular, la pertenencia o no a una clase se hace en función de las probabilidades de la predicción.

2.1.8.4 Random Forest

Es un algoritmo de aprendizaje supervisado utilizado ampliamente en problemas de regresión y clasificación. Utiliza un conjunto de árboles de decisión que son combinados

por medio de la técnica de ensamble de árboles llamada Bagging (Jose Martinez Heras, 2020), la cual permite combinar varios modelos de aprendizaje máquina para conseguir mejores resultados. Para el caso de Random Forest el algoritmo funciona de la siguiente manera:

- Se dividen los datos de entrenamiento en varios subconjuntos de muestras aleatorias.
- Se entrena un modelo para cada subconjunto de datos, generando tantos modelos como subconjuntos.
- Por último, se combinan todos los resultados de los modelos para generar un modelo robusto con mayor poder de predicción.

2.1.8.5 XGBoost

Según (Juan Bosco Mendoza Vega, 2020), XGBoost o Extreme Gradrient Boosting es un algoritmo de aprendizaje supervisado utilizado para clasificación, al igual que Random Forest utiliza arboles de decisión, pero con la diferencia de que estima arboles de manera secuencial a partir de la muestra original, de tal manera que cada vez que itera el algoritmo toma como base los árboles previamente estimados para generar el mejor modelo. Está basado en el principio de boosting, en el cual se realiza el entrenamiento secuencial de múltiples arboles débiles y por medio de la función de optimización llamada Extreme Gradient Boosting se consigue el modelo más fuerte con el mejor poder predictivo y más estable para la clasificación. En la actualidad es el más utilizado por sus ventajas computacionales de poco esfuerzo para obtener buenos resultados.

2.1.8.6 Redes neuronales

Las redes neuronales tradicionales conocidas también como redes neuronales artificiales son un subconjunto del aprendizaje automático, este tipo de redes neuronales se caracterizan por tener una arquitectura simple compuesta de una capa de entrada, una capa oculta y una capa de salida. Las redes neuronales son la base de los algoritmos de Deep Learning estas últimas tienen la característica de tener múltiples capas de neuronas ocultas. En la ilustración 3 se presentan los componentes de una rede neuronal.

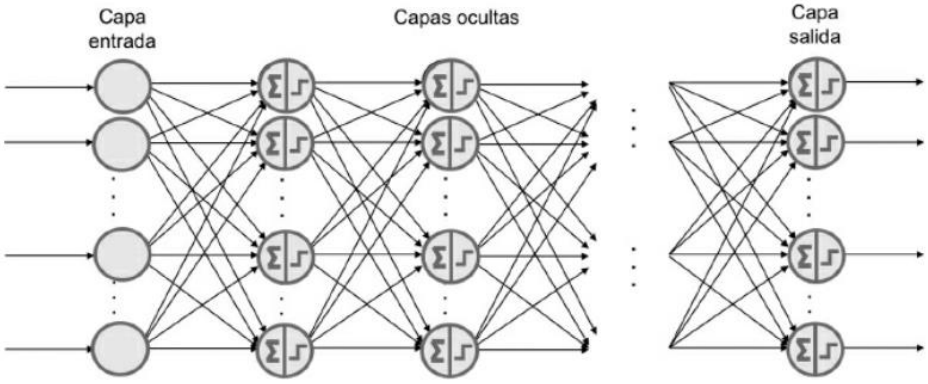

Ilustración 3. Componentes de una red neuronal (Jordi Torres, 2020)

Las redes neuronales permiten reconocer patrones en los datos para resolver problemas en el campo de la inteligencia artificial. Cada nodo en una red neuronal se conecta con otra

la cual tiene asociada un peso y un umbral, si la salida de un nodo individual está por encima de umbral especificado, el umbral es activado enviando datos a la siguiente capa de la red neuronal, de lo contrario los datos no pasan al siguiente nivel (IBM Cloud Education, 2020). Este tipo de modelos han sido ampliamente utilizadas para el procesamiento del lenguaje natural y el análisis de sentimientos.

2.1.8.7 BERT

BERT hace referencia al acrónimo de Bidirectional Encoder Representations from Transformers. Nace en el año 2018 en el laboratorio de investigación de Google AI Research (Jose García Nieto, 2019), como respuesta a la necesidad de optimizar el algoritmo de búsqueda de Google, para entregar resultados más asertivos a sus usuarios, ayudando al buscador a que no solo reconozca términos claves, sino que entienda el contexto de la búsqueda del usuario. El uso de Transformers para el procesamiento del lenguaje natural inició con la publicación por parte de investigadores de Google de "Attention Is All You Need" en donde se presenta una nueva propuesta que reemplaza la deficiencia de las redes neuronales recurrentes para el procesamiento del lenguaje natural (Vaswani et al., 2017), dicha arquitectura denominada Transformer está basada en el mecanismo de atención.

En la ilustración 4, se presentan dos frases procesadas con BERT, con el mecanismo de atención BERT identifica de manera diferente la palabra "cura" dependiendo del contexto.

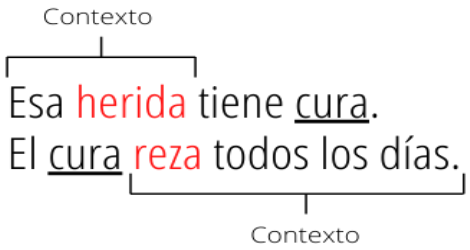

Ilustración 4. Comparación del uso de la misma palabra en contextos diferentes

La manera como BERT tiene en cuenta el contexto de una entrada para predecir la siguiente tal y como sucede cuando se realiza una búsqueda en el navegador de Google, está basado en la autoatención que tiene en cuenta toda la frase asignando puntajes o pesos a las palabras más relevante (Vaswani et al., 2017). En la ilustración 5, la palabra "making" se relaciona con las palabras "more" y "difficult" que tiene mayor importancia.

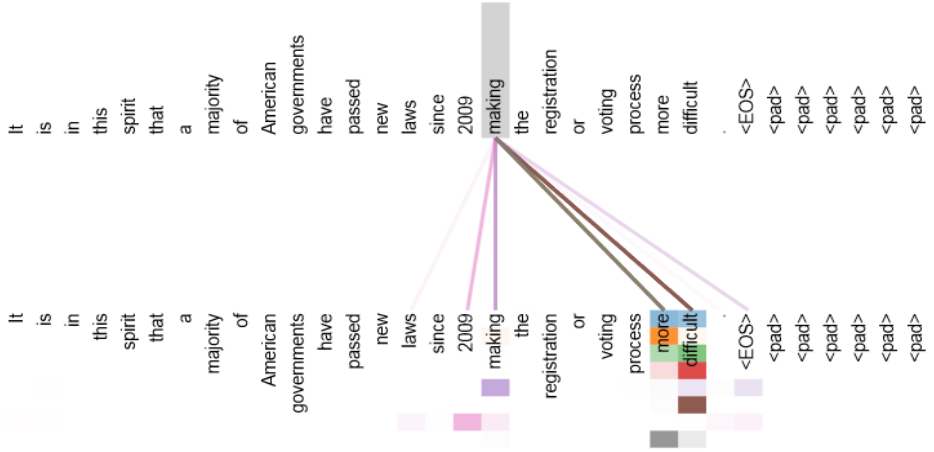

Ilustración 5. Mecanismo de atención (Vaswani et al., 2017)

La arquitectura Transformer está compuesta de un encoder y un decoder. El decoder recibe el embedding que es la representación del texto de entrada y realiza el proceso de autoatención para tener en la salida un embedding contextualizado. En la ilustración 6, se presenta la arquitectura Transformer publicada por primera vez en “Attention Is All You Need”.

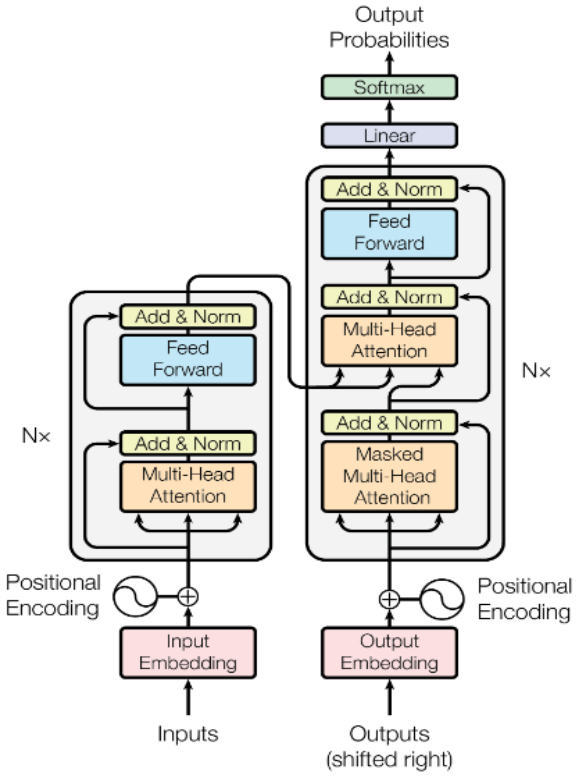

Ilustración 6. Arquitectura del modelo Transformer(Vaswani et al., 2017)

BERT se encuentra entrenado con una gran cantidad de datos y puede ser adaptado para resolver varios problemas de clasificación, tales como; el análisis de sentimiento, detección de noticias falsas, entre otros. Esta manera de adaptar BERT se denomina Fine-Tuning.

Con Fine-Tuning los pesos de los parámetros aprendidos por BERT son utilizados por una red neuronal que se añade adelante de BERT y se entrena con los datos del problema específico minimizando la función de perdida a través de backpropagation. A la salida de la red neuronal se tiene una función softmax que entrega las probabilidades de pertenencia a cada clase. La ilustración 7 muestra el enfoque pretraining y Fine-Tunning.

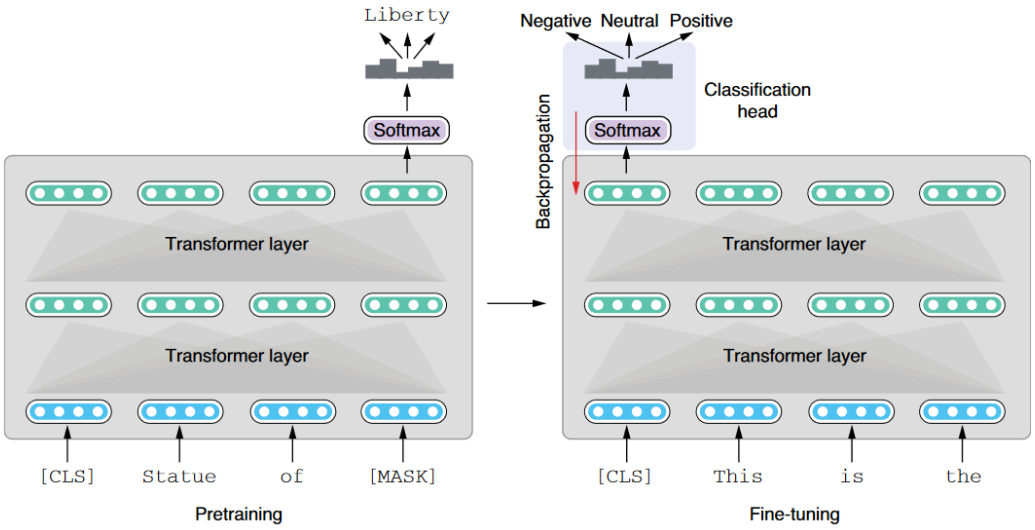

Ilustración 7. Tipos de adaptación de BERT: Pretraining y Fine-Tuning(Hagiwara, 2021)

2.1.9 Métricas de evaluación

Para las tareas de clasificación existen diferentes métricas que pueden ser utilizadas para evaluar el desempeño de un modelo de analítica como es el caso del ROC (Receiver Operating Characteristic Curve), que muestra el desempeño de un modelo de clasificación, considerando todos los umbrales de decisión posibles.

La curva ROC la componen dos métricas:

- La tasa de verdaderos positivos conocido como recall.
- La tasa de falsos positivos.

Estas medidas se obtienen de la tabla de confusión, que se muestra en la siguiente tabla:

Tabla 2. Matriz de confusión

		Actual Class	
		Positive (P)	**Negative (N)**
Predicted Class	**Positive (P)**	True Positive (TP)	False Positive (FP)
	Negative (N)	False Negative (FN)	True Negative (TN)

La tasa de verdaderos positivos se define matemáticamente como:

$$TPR = \frac{TP}{TP + FN}$$

La tasa de falsos positivos se define matemáticamente como:

$$FPR = \frac{FP}{FP + TN}$$

La curva ROC muestra los TPR vs FPR para los diferentes umbrales de decisión, el resultado perfecto ocurriría en el caso de que la tasa de verdaderos positivos fuera de 100%

y la tasa de falsos negativos fuera de 0%, este caso en la práctica no se presenta. La ilustración 8 presenta los diferentes componentes del ROC.

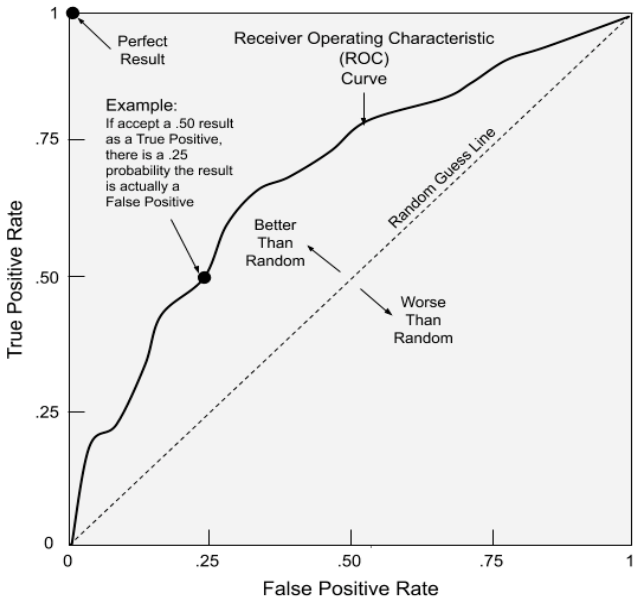

Ilustración 8. Componentes del ROC curve(Don Cowan, 2020)

2.2 ESTADO DEL ARTE

2.2.1 Detection of Fake News in a New Corpus for the Spanish Language.

(Posadas-Durán et al., 2019), crearon un corpus de noticias en idioma español extraídas de varios sitios web especializados en identificar noticias falsas y verdaderas. El Corpus es etiquetado manualmente como noticia real o falsa en el la base de datos. Se presentan un método de detección de noticias falsas con base en el estilo del texto. Los modelos utilizados para el entrenamiento fueron Support Vector Machine (SVM) con un kernel lineal, regresión logística, Random Forest y Boosting. La representación del texto (preprocesamiento) se realiza utilizando Bag of Words, n-grams y Part of Speech Tagging. El mejor modelo se logra con 4-grams dejando stopwords y realizando el entrenamiento con Boosting. El accuracy alcanzado con esta configuración es de 77,28. El principal aporte es la construcción de un corpus en el lenguaje español para futuras investigaciones en el campo del Procesamiento del lenguaje natural.

2.2.2 Fake News Detection in Spanish using Deep Learning Techniques.

(Martínez-Gallego et al., 2021), hace uso de 4 bases gratuitas existentes; dos de ellos en inglés (Fake and real news dataset and News Data Set Fake or Real), los cuales fueron combinados para obtener un dataset con 51.233 registros. De igual manera realiza la combinación de dos dataset en español (The Spanish Fake News Corpus y Fake News in Spanish) para obtener un dataset con un total de 2.571 registros. Se utilizan modelos clásicos como Support Vector Machines (SVM), Random Forest, Gradient Boosting Tree y Multilayer Perceptron, por otro lado, se utilizó Deep Learning Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) y Convolutional Neural Network (CNN). El principal aporte de este trabajo es el uso de Deep Learning para la detección de noticias falsas.

2.2.3 Language Independent Fake News Detection: English, Portuguese, and Spanish Mutual Features.

(Abonizio et al., 2020)**,** resalta que en la mayoría de los estudios sobre detección automática de noticias falsas se limitan a documentos en inglés, con pocos trabajos que evalúan otros idiomas y por otra parte ninguno compara características independientes del idioma. En este trabajo se comparan cuatro algoritmos de Machine Learning: K-Nearest Neighbords (KNN), Support Vector Machine (SVM), Random Forest y Gradient Boosting sobre datasets existentes en el lenguaje inglés, español y portugués, los autores manejan tres categorías como variable de respuesta: fake, legitimate y satirical. La principal contribución es la construcción de un corpus utilizando datos recolectados existentes, combinándolos y adaptándolo a las tres categorías como variables de respuesta.

2.2.4 Weakly Supervised Learning for Fake News Detection on Twitter.

(Helmstetter & Paulheim, 2018)**,** recolectan a gran escala datos de tweets en el lenguaje ingles realizando previamente una selección de sitios web confiables y no confiables para posteriormente usar la API de Twitter y clasificar como falso o verdadero un tweet de acuerdo con la fuente (enfoque Source-Base). El proceso de construcción del corpus se realiza de manera automática y, por lo tanto, es susceptible a errores de etiquetado debido a que fuentes confiables pueden tener noticias falsas por errores, así como fuentes no confiables pueden tener noticias verdaderas. Los algoritmos utilizados en este trabajo son Naive Bayes, Decision Tree, Support Vector Machines, Random Forest, XG-Boost y redes neuronales.

2.2.5 Detecting Fake News Spreaders on Twitter from a Multilingual Perspective.

(Vogel & Meghana, 2020)**,** estudia la propagación de noticias falsas escritas en español e inglés extraídas de la red social Twitter, con un enfoque multilenguaje (inglés y español), se utiliza un corpus ya creado haciendo seguimiento a cuentas de usuarios de Twitter dedicadas a realizar noticias falsas. Se evaluaron los siguientes modelos de analítica: Support Vector Machine (SVM), Logistic Regresión y Convolutional Neural Network (CNN). Se alcanzó un accuracy de 78% en un corpus en español y 87% para un corpus en inglés. Los mejores accuracy se alcanzaron con SVM utilizando TF-IDF.

En la tabla se presentan el resumen de las características de los trabajos encontrados en el estado del arte y las diferencias con este trabajo de grado.

2.2.6 Modelo de procesamiento de lenguaje natural para detectar la tasa de éxito de un artículo sobre otro.

(Ordóñez et al., 2021) presentan la implementación de un modelo de aprendizaje automático para detectar la tasa de éxito de un artículo sobre otro, los datos utilizados para el desarrollo del trabajo fue el conjunto de datos suministrado por Upworthy; sitio web dedicado a la narración positiva de noticias que tiene como objetivo generar contenido con impacto positivo tanto social como cultural. Los enfoques utilizados para responder a la pregunta de investigación fueron Machine Learning, Deep learning, sequense to sequence learning, NLP, Recurrent Neural Network, Long-Short Term Memory LSTM, Transformers y BERT.

2.3 CRITERIOS DE COMPARACION

Las diferencias de los trabajos encontrados en el estado del arte con respecto a este trabajo tienen en cuenta los siguientes criterios:

- Creación de corpus: Describe si en el trabajo de investigación se elabora un corpus propio o se utiliza uno existente.
- Trabajo con corpus en español: Describe si se trabaja con un corpus en lenguaje español o por el contrario se realiza en otro idioma.
- Enfoque en un contexto político colombiano: Describe si el trabajo se realizó en un contexto político colombiano o se realizó en un contexto de otro país y en un tema diferente al político.
- Uso de modelos tradicionales: Indica si en el trabajo se utilizan modelos tradicionales tales como SVC, Random Forest, regresión logística, arboles de decisión, etc.
- Uso de redes neuronales: Indica si el trabajo hace uso de técnicas basadas en redes neuronales profundas y en el mecanismo de atención.

Tabla 3. Estado del arte y diferencias con respecto al trabajo propuesto.

Características	Detection of Fake News in a New Corpus for the Spanish Language.	Fake News Detection in Spanish using Deep Learning Techniques.	Language Independent Fake News Detection: English, Portuguese, and Spanish Mutual Features.	Weakly Supervised Learning for Fake News Detection on Twitter.	Detecting Fake News Spreaders on Twitter from a Multilingual Perspective.	Modelo de procesamiento de lenguaje natural para detector la tasa de éxito de un artículo sobre otro.	Este trabajo
Creación de corpus	Si	No	No	No	No	No	**Si**
Trabajo con corpus en español	Si	Si	Si	Si	Si	No	**Si**
Enfocado en un contexto político colombiano.	No	No	No	No	No	No	**Si**
Uso Modelos tradicionales	Si	Si	Si	Si	Si	No	**Si**
Uso de redes neuronales profundas o mecanismo de atención.	No	Si	No	Si	Si	Si	**Si**

De acuerdo con el estado del arte y los diferentes enfoques encontrados, los principales aportes en este trabajo de grado son:

- La construcción de un nuevo corpus en el contexto político colombiano que sirve como base para el entrenamiento de un modelo de analítica.

- Desarrollo de un modelo de analítica que permita la detección de noticias falsas en tweets con formato texto y que pueda ser utilizado para detectar noticias falsas en las elecciones presidenciales de Colombia del año 2022, ya que este es entrenado con noticias del contexto político colombiano creado en este proyecto.

3 METODOLOGÍA

Para el desarrollo del proyecto de analítica se va a utilizar como metodología de referencia el CRISP-DM (José & Gallardo Arancibia, 2009). Esta metodología fue desarrollada en el año 1999 por un grupo importante de empresas europeas que decidieron colaborar para definir una guía de referencia de libre distribución, para el desarrollo de proyectos de minería de datos.

CRISP-DM se encuentra divida en 4 niveles de abstracción organizados en forma jerárquica, que comprenden desde tareas muy generales a específicas y organiza el desarrollo del proyecto en seis fases diferentes como se muestra en la ilustración 9:

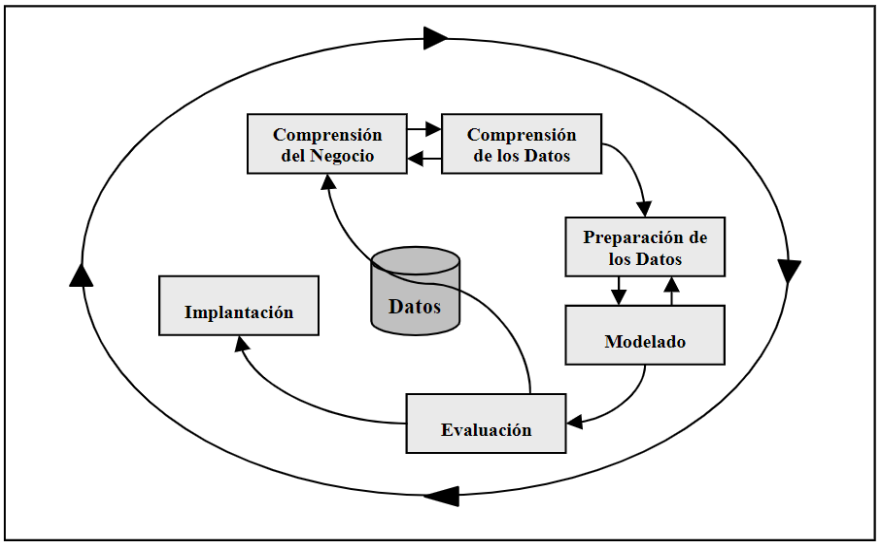

Ilustración 9. Metodología CRISP-DM (José & Gallardo Arancibia, 2009)

3.1 Fases de la metodología CRISP-DM

De acuerdo con (José & Gallardo Arancibia, 2009), la metodología CRISP-DM permite trabajar iterativamente en cada una de sus fases obteniendo mejores resultados en el desarrollo de proyectos de minería de datos. Las fases de CRISP-DM inician con la fase de entendimiento del negocio pasando por la comprensión de los datos, la preparación, el modelamiento, la evaluación y finalizando con el despliegue de la solución de analítica en un ambiente de producción. A continuación, se describen las fases de la metodología en el contexto del proyecto de detección de noticias falsas:

- **Entendimiento del negocio:** En esta fase del proyecto se investiga todo el marco teórico comprendido por el dominio del problema y de la solución, con el fin de conocer el contexto actual del problema que abarca las noticias falsas y cómo estas pueden ser detectadas por medio de modelos de analítica.

- **Entendimiento de los datos:** En esta fase se realiza la comprensión de la estructura del corpus base de (Posadas-Durán et al., 2019) y se establece la metodología a utilizar para la recolección, transformación e inserción de noticias en el corpus que se creará en el proyecto.

- **Preparación de los datos:** En esta etapa se realizan tres actividades importantes; la primera está relacionada con la recolección de noticias falsas y verdaderas para crear el corpus del proyecto, en lineamiento con la metodología definida en la fase anterior. La segunda actividad se encuentra relacionada con la combinación del corpus propio y otros dos corpus para crear un corpus combinado. Y la tercera etapa relacionada con las actividades de preprocesamiento del lenguaje natural y representación numérica del texto, pasos necesarios para poder utilizar los datos para el entrenamiento de los diferentes modelos de clasificación, que se van a implementar en la fase de modelamiento.

- **Modelado:** En esta fase se entrenan, evalúan y se seleccionan los mejores modelos de clasificación para la tarea de detección de noticias falsas. Para este proyecto se van a evaluar, preliminarmente, los siguientes modelos de clasificación: Random Forest, Support Vector Machines, Naive Bayes, Regresión Logística, XGBoost, redes neuronales y BERT.

- **Evaluación:** Para esta fase se evalúan los resultados de los modelos de clasificación validados en la etapa anterior y se selecciona el modelo que tenga la mejor métrica para la detección de noticias falsas.

- **Despliegue:** En esta etapa se realizará la entrega de los resultados obtenidos, se realizará la extracción de Tweets vía API y se clasificaran con el mejor modelo de aprendizaje maquina seleccionado. Opcionalmente se propone un despliegue en un chatbot del mejor modelo tradicional seleccionado.

3.2 Mapeo de fases de la Metodología Vs. Entregables y Objetivos del proyecto

En el siguiente cuadro se realiza un mapeo de los objetivos específicos planteados para el desarrollo del proyecto versus las actividades y salidas de cada una de las fases de la metodología CRISP-DM:

Tabla 4. Fases metodología CRISP-DM Versus Objetivos del proyecto

FASES	OBJETIVOS	ACTIVIDADES	SALIDAS
Entendimiento del negocio	-Entender el marco teórico en el dominio del problema y en el dominio de la solución. -Investigar el estado del arte en la detección de noticias falsas.	-Determinar objetivos generales y específicos para el desarrollo del proyecto -Evaluar la situación actual de los proyectos realizados sobre este campo -Construir plan de trabajo para el desarrollo del proyecto	-Antecedentes -Contexto actual del problema -Objetivos específicos y generales a ser abordados para el desarrollo del proyecto -Plan de proyecto con todas las actividades base para el desarrollo del proyecto -Selección preliminar de algoritmos de aprendizaje automático para ser evaluados dentro del proceso de clasificación de noticias
Entendimiento de los datos	- Construir un corpus de noticias falsas y verdaderas para el entrenamiento de un modelo de detección de noticias falsas.	-Entender estructura de los corpus a combinar. -Establecer la metodología para construir el corpus de noticias falsas y verdades en el contexto político colombiano.	Metodología para la recolección, evaluación e inserción de noticias falsas y verdaderas en el corpus de noticias colombianas.

	-Combinar el corpus de noticias colombianas con dos corpus adicionales con el fin medir el desempeño de los modelos de clasificación.	- Combinar los corpus de noticias. -Recolectar noticias falsas en el contexto colombiano de sitios confiables	
Preparación de los datos	Combinación de los corpus y entrenamiento de un modelo de detección de noticias falsas en el contexto político colombiano.	-Recopilación de noticias -Realizar el análisis exploratorio de los datos para detectar anomalías. -Realizar la limpieza de los datos. -Realizar el preprocesamiento de los datos. -Realizar el etiquetado a las noticias recopiladas para ser ingresadas en el corpus en construcción.	Corpus propio con información de noticias falsas y verdaderas dentro del contexto político colombiano. Corpus combinado entre el corpus propio y otros dos corpus.
Modelamiento	Formular un modelo de analítica para la detección de noticias falsas en el contexto político colombiano.	-Entrenar los modelos de clasificación con Random Forest, Support Vector Machines, Naive Bayes, Regresión Logística, redes neuronales y BERT. -Ajustar los parámetros de entrenamiento y evaluar el desempeño de los modelos.	-Definición de criterios de aceptación de los mejores modelos de acuerdo con los parámetros de evaluación seleccionados. -Modelos de analítica para clasificación ajustados de acuerdo con los mejores hiperparámetros que apliquen en cada tipo de modelo utilizado. -Descripción de los hiperparámetros ajustados para cada modelo de analítica validado.
Evaluación	- Validar de un conjunto de modelos de analítica, el mejor modelo para la detección de noticias falsas en el contexto político colombiano. -Validar el comportamiento de las métricas de evaluación del entrenamiento de los modelos de analítica con el corpus construido de noticias colombianas y el corpus complementado para evaluar el desempeño de los modelos de analítica	-Evaluar los resultados de los modelos de clasificación. -Seleccionar el modelo que tenga la mejor métrica de precisión para la detección de noticias falsas.	Modelo de clasificación seleccionado de acuerdo con los criterios de desempeño definidos para la detección de noticias falsas en el contexto político colombiano.
Despliegue	Formular y evaluar un modelo de analítica que permita la detección automática de noticias falsas en tweets en formato de texto en el contexto político colombiano, para las elecciones	-Realizar reporte final de resultados de acuerdo con las fases anteriores de la metodología. -Extracción de tweets con la API de Twitter y clasificación de estos con el mejor modelo de analítica seleccionado.	-Documento final con los resultados -Presentación final de resultados

	presidenciales del año 2022.	- Despliegue opcional de un chatbot para el mejor modelo de aprendizaje maquina tradicional. -Definir próximos pasos para futuros proyectos	

4 PRESENTACIÓN DE LA PROPUESTA

En este trabajo de grado se propone la formulación de un modelo de aprendizaje máquina para detectar noticias falsas de tweets en formato texto siguiendo el enfoque basado en el estilo (Style-base), el cual analiza cómo está escrito el texto. Este enfoque se fundamenta en el hecho de que una noticia falsa tiene, en general, una sintaxis diferente a una noticia verdadera, esta característica puede ser aprovechada por los modelos de aprendizaje máquina para realizar las detecciones.

4.1 Recolección de información

En esta fase se realizó la recolección de información de noticias falsas y verdaderas, el enfoque utilizado para la construcción del corpus consistió en aprovechar la existencia de sitios web especializados en identificar noticias falsas siguiendo el enfoque Source-Base, estas noticias fueron la base para la construcción del corpus. Por otra parte, estos sitios fueron la fuente para realizar el respectivo etiquetado de las noticias en dos categorías: noticia falsa y noticia verdadera.

El número de noticias recolectadas en el proyecto se presenta en la tabla:

Tabla 5. Cantidad de noticias recolectadas

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Noticia Falsa	317	50,0	50,0	50,0
Noticia Real	317	50,0	50,0	100,0
Total	634	100,0	100,0	

4.1.1 Sitios Web de noticias falsas

Para el proceso de recolección de noticias falsas se utilizó principalmente el sitio web Colombiacheck [2] Según (Colombia Check, 2022), este un proyecto periodístico de investigación desarrollado en Colombia, que ayuda a la verificación de información emitida por importantes personajes públicos en redes sociales o portales noticiosos, sobre asuntos relevantes para el debate público. Su metodología de análisis y clasificación está basada en lo siguiente:

- Realizar una selección de publicaciones relevantes para el debate público que circula por las redes sociales o por importantes portales noticiosos.
- Realizar una búsqueda de la veracidad de las publicaciones seleccionadas contrastando hechos, datos y fuentes utilizadas.
- Determinar la clasificación de las publicaciones con base a un conjunto de calificaciones definidas por Colombiacheck, estas calificaciones son:

    - **Verdadero:** Datos que corresponden exactamente a la realidad de los hechos.
    - **Verdadero, pero…:** Están alineados a los datos más recientes, pero dejan por fuera alguna parte importante del contexto.
    - **Cuestionable:** Están alineados a los datos más recientes, pero se usan para llegar a conclusiones erróneas o son interpretados erróneamente.

[2] https://colombiacheck.com/

- **Falso:** Los datos son contrarios a la realidad de los hechos.
- **Inchequeable:** Datos que no son confiables, porque vienen de una opinión.

Para la construcción del corpus de noticias en el contexto político colombiano solo fueron consideradas las calificaciones verdaderas y falsas.

El número de noticias falsas extraídas de diferentes sitios de internet se presenta a continuación:

Tabla 6. Sitios Web para la recolección de noticias falsas

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Colombia Check	293	92,4	92,4	92,4
Twitter	5	1,6	1,6	94,0
El tiempo	4	1,3	1,3	95,3
Ecuador Chequea	3	,9	,9	96,2
Verificado	3	,9	,9	97,2
Twitter colCERT	2	,6	,6	97,8
Caracol radio	1	,3	,3	98,1
El avispero	1	,3	,3	98,4
la silla vacia	1	,3	,3	98,7
La silla Vacia	1	,3	,3	99,1
pagina web buenosaires. gob.ar	1	,3	,3	99,4
Policia nacional	1	,3	,3	99,7
Twitter mintic	1	,3	,3	100,0
Total	317	100,0	100,0	

De la anterior tabla se observa que el 92.4 % de las noticias falsas fueron extraídas de Colombia Check.

4.1.2 Sitios Web de noticias verdaderas

Para el proceso de la recolección de noticias verdaderas se utilizaron los sitios presentados en la tabla 7, en donde se muestra el número de noticias recolectadas en cada uno de los portales.

Tabla 7 Sitios Web para la recolección de noticias verdaderas

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válido	90 minutos	115	36,3	36,3	36,3
	El Nuevo Siglo	47	14,8	14,8	51,1
	El heraldo	34	10,7	10,7	61,8
	El Pais	30	9,5	9,5	71,3
	El colombiano	28	8,8	8,8	80,1
	El Espectador	22	6,9	6,9	87,1
	Portafolio	21	6,6	6,6	93,7
	El universal	12	3,8	3,8	97,5
	El tiempo	3	,9	,9	98,4
	Semana	2	,6	,6	99,1
	Vanguardia	2	,6	,6	99,7
	La Republica	1	,3	,3	100,0
	Total	317	100,0	100,0	

De la tabla 7, se observa que los principales portales de noticias utilizados para extraer las noticias verdaderas fueron: 90 minutos, el nuevo siglo, el heraldo, el país y el colombiano.

Las noticias publicadas por estos sitios web son altamente confiables y han sido creadas basadas en hechos, datos e investigaciones, por lo tanto, las noticias recolectadas de estos sitios fueron etiquetadas como noticias verdaderas.

4.1.3 Metodología para la construcción del corpus del proyecto

En la ilustración10, se muestra la metodología utilizada para la construcción del Corpus:

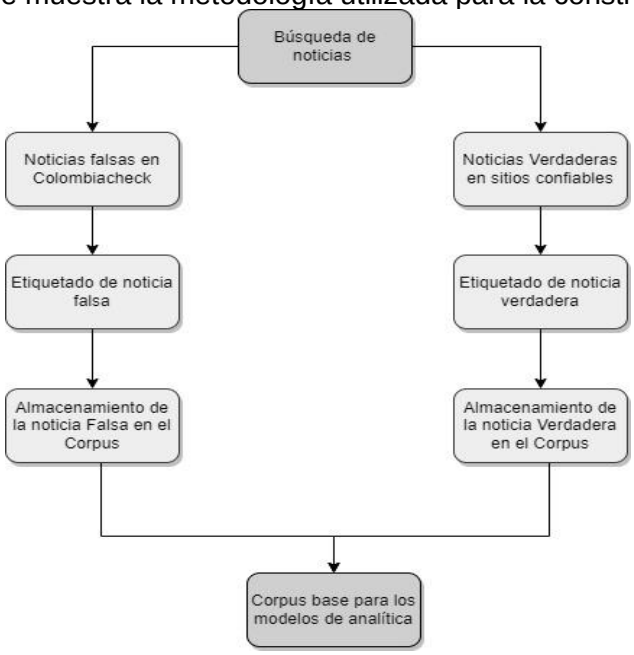

Ilustración 10. Metodología para la construcción del corpus de noticias del contexto colombiano

Esta metodología consistió en:

- **Búsqueda de noticias:** Esta actividad se dividió en dos tareas; la búsqueda de noticias falsas en el portal web Colombiacheck y la búsqueda de noticias verdaderas en los sitios confiables seleccionados.

- **Almacenamiento de noticias en el Corpus:** Una vez encontradas las noticias falsas y verdaderas estas fueron almacenadas en el Corpus manteniendo su clasificación o etiqueta (verdadera o falsa). La estructura del corpus creado se muestra en la siguiente tabla:

Tabla 8 Estructura Corpus del proyecto

Categoría	Tema	Origen	Titulo	Texto	Fuente
Noticia Falsa	Tema de la noticia	Sitio web donde se extrajo la noticia	Título de la noticia	1er párrafo y opcionalmente el segundo párrafo	URL de la fuente
Noticia Verdadera	Tema de la noticia	Sitio web donde se extrajo la noticia	Título de la noticia	1er párrafo y opcionalmente el segundo párrafo	URL de la fuente

- **Corpus Base**: Una vez se completó el almacenamiento de las noticias se consolido un corpus base que sirvió como insumo para el entrenamiento de los modelos de clasificación seleccionados dentro del proyecto.

El número de noticias se mantuvo balanceada en un 50% para cada clase, con el fin de tener información suficiente para el entrenamiento balanceado de los modelos de analítica.

Los estadísticos descriptivos del número de caracteres de la noticia se presentan en la siguiente tabla:

Tabla 9 Estadísticos Corpus Base

N	Válido	634
	Perdidos	0
Media		521,21
Mediana		432,00
Mínimo		62
Máximo		3980
Percentiles	10	180,00
	20	247,00
	30	291,50
	40	359,00
	50	432,00
	60	497,00
	70	584,00
	80	690,00
	90	965,50

Se puede observar que la longitud mínima de la noticia fue de 62 caracteres y la longitud máxima 3.980 caracteres de longitud. La mitad de las noticias tiene a lo sumo 432 caracteres de longitud. El 80% de las noticias tiene a lo sumo 690 caracteres de longitud.

El corpus construido se encuentra publicado para libre consulta a través del repositorio GitHub [3]

4.1.4 Combinación de corpus del Proyecto con otros corpus en español

Con el fin de validar el desempeño de los modelos de clasificación aumentando la cantidad de noticias de otros contextos diferentes al colombiano, se recolectaron otros corpus disponibles en el lenguaje español y se unieron con el corpus construido del contexto político colombiano, con el objetivo de validar si las capacidades de detección de noticias mejoraban o por el contrario empeoraban.

El primer corpus que se consideró para la conformación de un corpus con más noticias fue el de Posadas-Durán, el cual cuenta con 338 noticias verdaderas y 338 noticias falsas etiquetadas como Fake y True como se muestra en la tabla 10.

Tabla 10 Cantidad de noticias Corpus Posadas-Durán

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Fake	338	50,0	50,0	50,0
True	338	50,0	50,0	100,0
Total	676	100,0	100,0	

Las categorías de las noticias de este corpus se presentan en la tabla 11:

Tabla 11 Tipos de noticias Corpus Posadas-Durán

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Politics	226	33,4	33,4	33,4
Entertainment	103	15,2	15,2	48,7
Society	93	13,8	13,8	62,4
Sport	86	12,7	12,7	75,1
Science	62	9,2	9,2	84,3
Health	32	4,7	4,7	89,1
Economy	30	4,4	4,4	93,5
Security	29	4,3	4,3	97,8
Education	15	2,2	2,2	100,0
Total	676	100,0	100,0	

[3] https://github.com/jmontano1987/modelo-deteccion-noticias-falsas.git

Los descriptivos estadísticos de la longitud de las noticias se presentan en la siguiente tabla:

Tabla 12 Estadísticos Descriptivos Longitud noticias Corpus Posadas-Durán

N	Válido	676
	Perdidos	0
Media		2285,12
Mediana		1868,00
Mínimo		124
Máximo		12437
Percentiles	10	958,60
	20	1237,80
	30	1421,20
	40	1645,00
	50	1868,00
	60	2200,80
	70	2584,60
	80	3033,80
	90	3800,80

De la tabla 12 se puede observar que la noticia más pequeña tiene 124 caracteres. La de mayor tamaño tiene 12.437 caracteres. El 80% de las noticias tiene a lo sumo 3.033 caracteres de longitud.

El segundo corpus considerado fue el de Arsenni Tretiakov, este corpus contiene 1.000 noticias verdaderas y 1.000 noticias falsas. En este corpus las noticias no están organizadas por tipos de noticias. La tabla 13 resume el número de noticias de cada categoría.

Tabla 13 Cantidad de noticias Corpus Arsenni Tretiakov

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
False	1000	50,0	50,0	50,0
True	1000	50,0	50,0	100,0
Total	2000	100,0	100,0	

Los estadísticos descriptivos de la longitud de las noticias se presentan en la tabla 14:

Tabla 14 Estadísticos Descriptivos Longitud noticias Corpus Arsenni Tretiakov

N	Válido	2000
	Perdidos	0
Media		238,00
Mediana		259,00
Mínimo		36
Máximo		2293
Percentiles	10	134,10
	20	241,00
	30	257,00
	40	258,00
	50	259,00
	60	259,00
	70	260,00
	80	261,00
	90	262,00

De la tabla 14 se puede observar que la longitud mínima de las noticias tiene 36 caracteres, la longitud máxima tiene 2.293 caracteres. El 80% de las noticias tiene como máximo 261 caracteres de longitud.

En la tabla 15 se resumen los 3 corpus utilizados para la conformación de un corpus más numeroso para el entrenamiento de los modelos:

Tabla 15 Composición cantidad registros Corpus utilizados.

Referencia del corpus	Número de registros	Porcentaje de registros	Número de registros noticias verdaderas	Número de registros noticias falsas
Corpus contexto mexicano (Posadas-Durán, 2019)	676	20.42%	338	338
Corpus otros contextos(Arsenii Tretiakov, 2020)	2000	60.42%	1000	1000
Corpus noticias colombianas	634	19.15%	317	317
Total	3310	100%	1655	1655

De la tabla 15 se observa que las categorías del tipo de noticia de los corpus se encuentran balanceadas. Por otra parte, al realizar la unión de los corpus se puede observar que el corpus de noticias colombianas representa solo el 19.15% del tamaño total del corpus combinado. En cuanto a la longitud de las noticias de cada corpus se observa que las longitudes son diferentes.

4.2 Preparación de los datos

4.2.1 Preprocesamiento del texto

En esta sección se detallan los pasos desarrollados para realizar el preprocesamiento del texto, pasos necesarios para poder realizar la limpieza del texto y poder entregar los datos en un formato adecuado para el entrenamiento de los modelos.

El preprocesamiento del corpus siguió las siguientes actividades:

- Se realizó la conversión de mayúsculas a minúsculas.
- Se eliminaron todos los enlaces o urls de página web que se encontraban en el texto.
- Se eliminaron todos los números presentes en el texto ya que, dado que no son relevantes para el análisis y el modelamiento.
- Se eliminaron los signos de puntuación y caracteres especiales, tales como: [!”#$%&'()*+,-./:;<=>?@[\]^_`{|}~].
- Se eliminaron todos acentos y tildes de las palabras.
- Se eliminaron todos los saltos de línea, espacios en blanco y líneas nuevas vacías dentro del texto de las noticias.
- Eliminación de stopwords: En este proceso se eliminaron las palabras más comunes en el idioma español, como “a”,”el”,”ella”, entre otras. Estas palabras no tienen un significado importante y por lo general no apartan para el entrenamiento del modelo de aprendizaje máquina.
- Lematización del texto: En este proceso se eliminó el plural, conjugación, el tiempo o los atributos finales de las palabras para dejar la palabra en su base común.

En la ilustración 11, se muestra el orden de las tareas de preprocesamiento realizadas en el proyecto:

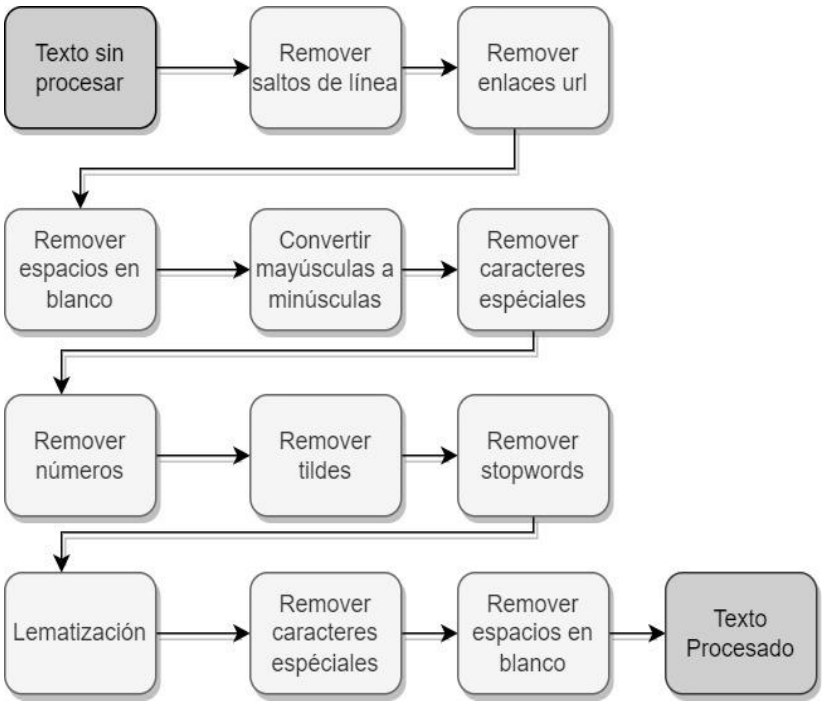

Ilustración 11. Proceso de limpieza de corpus

4.2.2 Representación del texto para modelos tradicionales

La representación del texto para el entrenamiento de los modelos de analítica se realizó con TF-IDF, esta representación numérica se elige al ser una mejora de los métodos de Bag of Words, esto debido a que considera la frecuencia de los términos no solo en un documento, si no en todos los documentos del corpus. Adicionalmente, esta técnica permite eliminar términos demasiado frecuentes conocidas como “stop words” así como eliminación de términos sin importancia con poca aparición. Frente a Bag of Words, TF-IDF presenta un mejor desempeño en los modelos de analítica.

4.2.3 Preprocesamiento de texto y entrenamiento con BERT

El modelo pre-entrenado seleccionado para este proyecto fue bert_multi_cased_L-12_H-768_A-12 [4], este ha sido pre-entrenado utilizando Wikipedia en múltiples lenguajes. Las mayúsculas, minúsculas y tildes utilizadas para el entrenamiento del modelo se preservan (Jacob Devlin, 2018).

El preprocesamiento del texto se realiza con bert_multi_cased_preprocess [5], modulo que transforma el texto de tal manera que pueda ser procesado por BERT. Este modelo usa un vocabulario extraído de Wikipedia en múltiples lenguajes y es el mismo utilizado para el entrenamiento de BERT (Jacob Devlin et al., 2018).

4.3 Modelado

4.3.1 Modelos de clasificación

Los algoritmos de aprendizaje máquina utilizados para la tarea de clasificación de noticias falsas fueron seleccionados con base en lo encontrado en el estado del arte. Estos están divididos en dos grupos; modelos de aprendizaje tradicionales y modelos Transformers basados en el mecanismo de atención.

Para el entrenamiento con modelos tradicionales se seleccionaron los siguientes:

- **Random Forest**
- **Support Vector Classifier**
- **Naive Bayes**
- **Regresión Logística**
- **XGBoost**
- **Redes neuronales tradicionales**

Para el caso del modelado con redes atención se utilizó BERT.

4.3.2 Búsqueda de Hiperparámetros

Entrenar modelos de aprendizaje automático requiere de la configuración de hiperparámetros; estos son parámetros ajustables que permiten controlar el proceso de entrenamiento del modelo de analítica, el rendimiento del modelo depende en gran medida de la buena elección de estos parámetros.

[4] https://tfhub.dev/tensorflow/bert_multi_cased_L-12_H-768_A-12/4
[5] https://tfhub.dev/tensorflow/bert_multi_cased_preprocess/3

Los hiperparámetros deben ser fijados durante el proceso de entrenamiento por esta razón es importante definir un proceso que ayude a su elección correcta. En la actualidad existen varias técnicas para la optimización de los hiperparámetros; búsqueda en cuadricula conocida como Grid search búsqueda aleatoria o Random search y optimización bayesiana.

Para el caso del proyecto se eligió la búsqueda aleatoria Random Seach considerando que es una buena estrategia para hallar los mejores hiperparámetros sin requerir mucha potencia computacional.

4.3.3 Ajuste de hiperparámetros en modelos tradicionales

A continuación, se describen los diferentes hiperparámetros optimizados para los modelos de clasificación utilizados en el proyecto. En todos los casos la búsqueda de los mejores valores de estos hiperparámetros se realizó por medio de validación cruzada (Cross-Validation)

- **Random Forest**

  Los hiperparámetros validados en esta aproximación fueron: el número de estimadores (n_estimators), el máximo de numero de características a considerar para la división de los nodos (max_features), máximo número de niveles en cada árbol de decisión(max_depth), el mínimo de puntos en cada nodo antes de dividir el nodo (min_samples_split), el mínimo número de datos permitido en cada nodo hoja (min_samples_leaf) y el método para el muestreo de los datos (Bootstrap).

- **Support Vector Machine**

  Los hiperparámetros validados en esta aproximación fueron: el parámetro C que agregar una penalización por cada punto mal clasificado en el hiperplano de separación, el tipo de kernel utilizado en los casos en que los datos no se puedan separar por una margen lineal y el parámetro gamma que ayuda a controlar el comportamiento del kernel RBF.

- **Naive Bayes**

  Los hiperparámetros validados en esta aproximación fueron: el valor del parámetro alpha que controla la forma en como el modelo aprende de los datos. Un valor demasiado pequeño hará que el modelo se sobreajuste y un valor muy alto hará que el modelo no se ajuste (underfitting) y el fit_prior que controla si el modelo aprende de las probabilidades previas de las clases o se tiene en cuenta una distribución uniforme de las clases.

- **Regresión logística**

  Este modelo no tiene hiperparámetros críticos para ajustar, pero dentro del proyecto se validaron los siguientes: el parámetro C que controla la fuerza de la penalización de los predictores y la regularización L1, L2.

- **XGboost**

  Para el caso de XGBoost los hiperparámetros validados fueron: la tasa de aprendizaje o learning rate, la cual controla la tasa a la cual el algoritmo aprende de los errores pasados, el número de iteraciones(n_iter), la profundidad máxima de los árboles (max_depth), la mínima reducción de la pérdida para hacer una partición en un nodo, la proporción de aleatoriedad de los predictores para cada árbol y el número de instancias para cada nodo hijo.

- **Redes neuronales tradicionales**

  La búsqueda de los hiperparámetros de las redes neuronales se realizó variando el número de capas densas, el número de neuronas por capa, el tamaño del batch, las épocas y el criterio de parada temprano.

4.4 Evaluación

Para el caso del proyecto la métrica de evaluación de desempeño de los modelos se realiza con el accuracy teniendo en cuenta que las clases se encuentran balanceadas, adicionalmente se consideran las métricas recall y precisión para la interpretación de los resultados de la clasificación; la precisión es una métrica que permite medir la calidad del modelo para detectar noticias falsas y el recall evalúa la cantidad de noticias falsas que el modelo puede predecir, motivo por el cual fueron estas las seleccionadas.

5 RESULTADOS

En esta sección se presenta los mejores resultados obtenidos de la evaluación de los diferentes algoritmos de aprendizaje automático propuestos en el proyecto.

El análisis de los resultados se realiza comparando el desempeño de los modelos de aprendizaje automático con el corpus de noticias colombianas creado en el proyecto y con el corpus combinado que incluye el corpus de noticias colombianas.

5.1 Entorno de desarrollo

Para el desarrollo de este proyecto se utilizó la plataforma Google Colab, herramienta de Google para ejecutar código en el lenguaje de programación Python para la creación de modelos de aprendizaje máquina.

La principal ventaja de utilizar colab es el acceso rápido para el aprovisionamiento de los recursos de hardware para el entrenamiento de los modelos y el uso de las GPU necesarios para el procesamiento en paralelo que permiten optimizar los tiempos de entrenamiento.

Google Colab tiene tres suscripciones disponibles para el público; la versión gratuita, la versión Google Colab Pro y la versión Google Colab Pro+, cada una de estas suscripciones con características complementarias que, dependiendo de la necesidad del proyecto de analítica se elige el tipo de suscripción adecuado. Para el caso del proyecto, se utilizó la versión de Google Colab Pro.

Este proyecto fue implementado utilizando el siguiente entorno:

- Hardware:
  - Portátil Lenovo E495 AMD Ryzen 5 2.19 GHz, 32 GB RAM
  - Portátil HP Intel Core I5 5200U 2.60 GHz, 8 GB RAM
- Sistema Operativo: Windows 10 Professional
- Software de analítica: Google Colab Pro

El código del proyecto se encuentra publicado para consulta de manera libre en GitHub [6]

5.2 Framework Utilizado

En el siguiente apartado se especifican las principales librerías que se utilizaron para el desarrollo del proyecto:

- Librerías para la extracción de URLs:

Tabla 16 Paquete extracción URL

Librería	Versión
urlextract	1.6.0

[6] https://github.com/jmontano1987/modelo-deteccion-noticias-falsas.git

- librerías para realizar WebScrapping:

Tabla 17 Paquete WebScrapping

Librería	Versión
beautifulsoup4	4.6.3

- Tensorflow:

Tabla 18 Paquetes Tensorflow

Librería	Versión
tensorflow	2.8.2+zzzcolab20220527125636
tensorflow-addons	0.17.1
tensorflow-datasets	4.0.1
tensorflow-estimator	2.8.0
tensorflow-gcs-config	2.8.0
tensorflow-hub	0.12.0
tensorflow-io-gcs-filesystem	0.26.0
tensorflow-metadata	1.8.0
tensorflow-model-optimization	0.7.2
tensorflow-probability	0.16.0
tensorflow-text	2.8.2

- Librerias de Google:

Tabla 19 Paquetes de Google

Librería	Versión
google	2.0.3
google-api-core	1.31.6
google-api-python-client	1.12.11
google-auth	1.35.0
google-auth-httplib2	0.0.4
google-auth-oauthlib	0.4.6
google-cloud-core	1.0.3
google-colab	file:///colabtools/dist/google-colab-1.0.0.tar.gz
googleapis-common-protos	1.56.2
googledrivedownloader	0.4

- librerías para stopwords:

Tabla 20 Paquetes Stopwords

Librería	Versión
stop-words	2018.7.23

- librerías para la conexión a Twitter:

Tabla 21 Paquete Tweepy para Twitter

Librería	Versión
tweepy	3.10.0

- librerías para la creación de la nube de palabras:

Tabla 22 Paquete para la nube de palabras

Librería	Versión
wordcloud	1.5.0

5.3 Resultados con corpus de noticias colombianas

5.3.1 Resultado con modelos tradicionales

Antes de realiza el entrenamiento de los modelos se realizó el particionamiento de los datos como se presenta en la tabla 23:

Tabla 23 Particionamiento datos Corpus de noticias colombianas

Particionamiento	Porcentaje
Entrenamiento	507 noticias (80%)
Pruebas	127 noticias (20%)

De los diferentes experimentos realizados (Ver anexo) en donde se modificaron los valores de las configuraciones de TF-IDF y de la búsqueda aleatoria para los hiperparámetros, la siguiente tabla resume los valores de configuración que permitieron obtener los mejores resultados en la clasificación de noticias para los modelos tradicionales.

Tabla 24 Valores de configuración Random Search y TF-ID

Random Search		Parámetros TF-IDF			
n_iter	cv	ngram_range	min_df	max_df	max_features
10	3	(1,3)	0.001	0.8	300

En la tabla 25, se presenta los mejores resultados de los modelos tradicionales entrenados con el corpus de noticias colombianas:

Tabla 25 Resultado métricas de evaluación con corpus de noticias colombianas

	Modelo	Corpus noticias colombianas		
		Accuracy	Recall	Precision
Mejor resultado	**Random Forest**	**0.898**	**0.887**	**0.902**
	Naive Bayes	0.874	0.846	0.902
	SVM	0.890	0.912	0.852
	Regresión Logística	0.898	0.914	0.869

	XGBoost	0.882	0.871	0.885
	Redes neuronales	0.898	0.944	0.836

De acuerdo con los anteriores resultados se observa que la métrica de precisión para Random Forest, Regresión logística y redes neuronales son iguales, por lo tanto, se considera la precisión como métrica para elegir el mejor modelo. Teniendo en cuenta lo anterior el modelo elegido es Random Forest.

La matriz de confusión del modelo seleccionado se presenta en la siguiente tabla, en donde 114 noticias fueron clasificadas correctamente de las 127 noticias utilizadas como conjunto de prueba.

Tabla 26 Matriz de confusión para mejor modelo obtenido - Random Forest

Verdadero Positivo (TP): • Realidad: Noticia Falsa • Modelo ML predicho: Noticia Falsa • Número de resultados de TP: 55	Falso positivo (FP): • Realidad: Noticia Verdadera • Modelo ML predicho: Noticia Falsa • Número de resultados de FP: 6
Falso Negativo (FN): • Realidad: Noticia Falsa • Modelo ML predicho: Noticia Verdadera • Número de resultados FN: 7	Verdadero Negativo (TN): • Realidad: Noticia Verdadera • Modelo ML predicho: Noticia Verdadera • Número de resultados TN: 59

En la ilustración 12, se presenta las curvas ROC y Precision-Recall para Random Forest entrenado con el corpus de noticias colombianas.

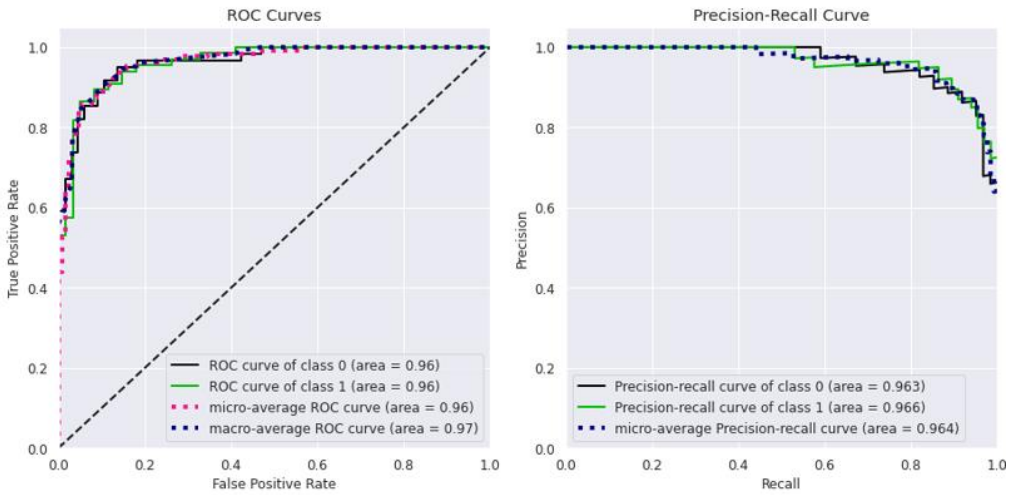

Ilustración 12. Resultados con datos de prueba y corpus de noticias colombianas

Respecto a la curva del ROC podemos determinar que el modelo seleccionado tiene un buen desempeño en las predicciones con datos de prueba ya que el ROC mejora considerablemente con respecto al baseline.

5.3.2 Resultado con BERT

Se realizó el entrenamiento de los modelos con BERT y las noticias colombianas obteniendo un accuracy de 0.95 con los datos de prueba, el texto no es preprocesado previamente ya que BERT cuenta con un módulo propio de preprocesamiento del texto. El número de épocas considerado para el entrenamiento fue de 20, obteniendo un comportamiento de la función de perdida y el accuracy como se muestra en la ilustración 13.

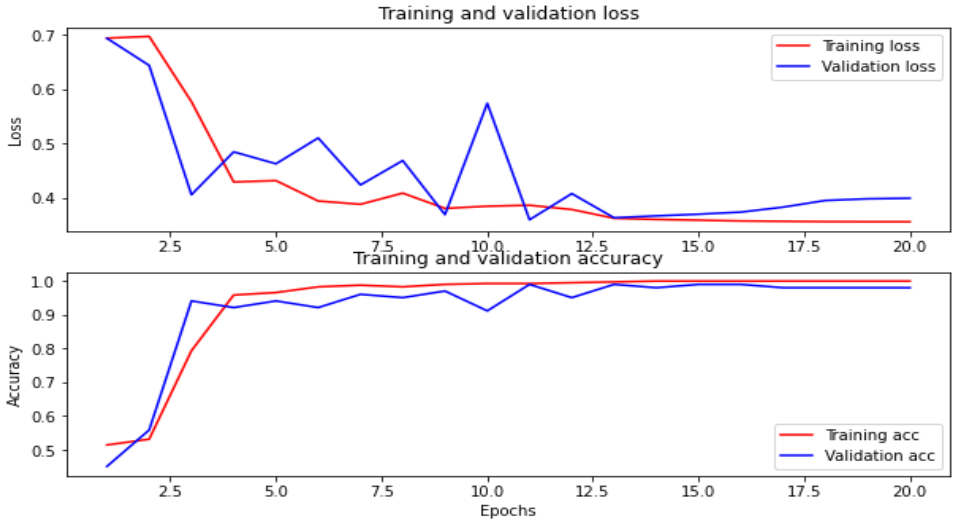

Ilustración 13. Función de perdida y accuracy del modelo entrenado con BERT

De los resultados anteriores es posible concluir que el desempeño de los resultados de clasificación de noticias falsas es mejor con BERT que con los modelos tradicionales.

5.4 Resultados con corpus combinado.

5.4.1 Resultado con modelos tradicionales

Antes de realiza el entrenamiento de los datos se realiza el particionamiento de los datos de entrenamiento y de prueba:

Tabla 27 Particionamiento datos Corpus combinado

Particionamiento	Porcentaje
Entrenamiento	2.648 noticias (80%)
Pruebas	662 noticias (20%)

De los diferentes experimentos realizados (Ver anexo) en donde se modificaron los valores de las configuraciones de la técnica TF-IDF y de la búsqueda aleatoria para los hiperparámetros, la siguiente tabla resume los valores de configuración que permitieron obtener los mejores resultados en la clasificación de noticias:

Tabla 28 Valores de configuración Random Search y TF-ID

Random Search		Parámetros TF-IDF			
n_iter	cv	ngram_range	min_df	max_df	max_features
10	3	(1,3)	0.001	0.8	300

Para el caso de los modelos tradicionales y el corpus combinado de noticias, los mejores resultados fueron:

Tabla 29 Resultado métricas de evaluación con corpus combinado

	Modelo	Corpus combinado		
		Accuracy	Recall	Precision
Mejor resultado	Random Forest	0.698	0.680	0.689
	Naive Bayes	0.625	0.595	0.667
	SVM	0.693	0.674	0.689
	Regresión Logística	0.678	0.667	0.648
	XGBoost	0.711	0.792	0.533
	Redes neuronales	0.650	0.668	0.524

De acuerdo con los anteriores resultados se observa que la métrica de accuracy para XG-Boost presenta los mejores resultados, por lo tanto, este es considerado como el mejor modelo entrenado con un corpus de noticias de varios contextos. La matriz de confusión del modelo seleccionado para los datos de prueba se presenta en la siguiente tabla, en donde se logra clasificar correctamente 471 noticias de un total de 662 noticias.

Tabla 30 Matriz de confusión para mejor modelo obtenido - XGBoost

Verdadero Positivo (TP): • Realidad: Noticia Falsa • Modelo ML predicho: Noticia Falsa • Número de resultados de TP: 168	Falso positivo (FP): • Realidad: Noticia Verdadera • Modelo ML predicho: Noticia Falsa • Número de resultados de FP: 147
Falso Negativo (FN): • Realidad: Noticia Falsa • Modelo ML predicho: Noticia Verdadera • Número de resultados FN: 44	Verdadero Negativo (TN): • Realidad: Noticia Verdadera • Modelo ML predicho: Noticia Verdadera • Número de resultados TN: 303

En la ilustración 14, se presenta las curvas ROC y Precision-Recall para XG-Boost entrenado con el corpus de noticias colombianas:

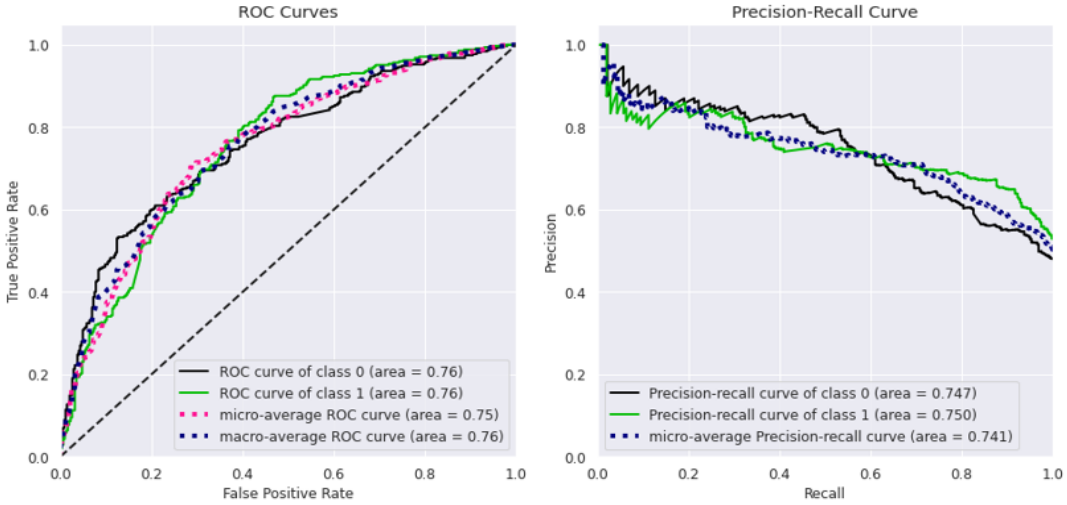

Ilustración 14 Resultados con datos de prueba y con corpus combinado

De los anteriores resultados, se observa que el corpus de noticias colombianas presenta un mejor desempeño que el corpus de noticias combinado que contiene noticias de diferentes contextos.

5.4.2 Resultado con BERT

Se realizó el entrenamiento de BERT con noticias de varios contextos obteniendo un accuracy de 0.82 con los datos de prueba. El número de épocas considerado para el entrenamiento fue de 20 épocas obteniendo un comportamiento de la función de perdida y el accuracy como se muestra en la ilustración 15.

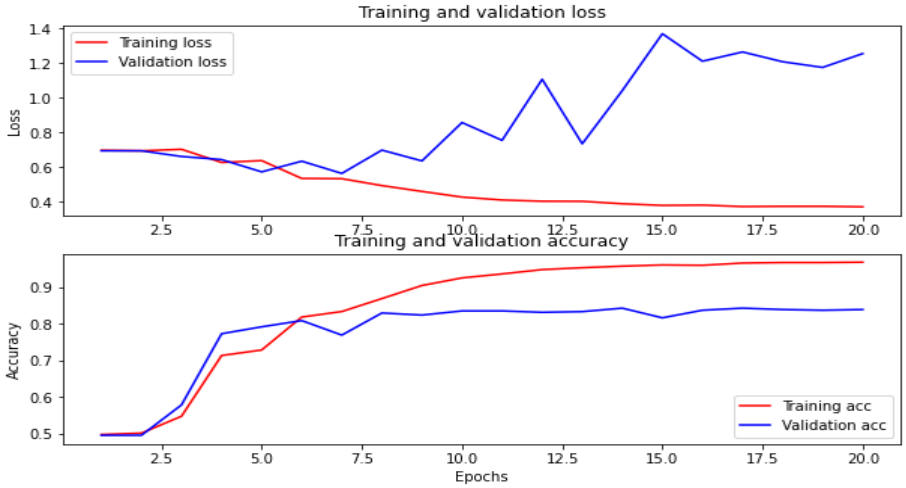

Ilustración 15 Función de perdida y accuracy del modelo entrenado con BERT corpus combinado

Del anterior resultado es posible concluir que con BERT se obtiene los mejores resultados, sin embargo, el desempeño del modelo entrenado con el corpus combinado tiene un menor desempeño que el modelo entrenado con noticias del contexto político colombiano.

5.5 Resumen general de resultados

En las siguientes tablas se resumen los resultados obtenidos del proceso de experimentación:

Tabla 31 Resumen de resultados accuracy modelos entrenados con corpus de noticias colombianas

Corpus noticias colombianas	
Modelo	**Accuracy**
Random Forest	0.898
BERT sin preprocesamiento del texto	0.958

Tabla 32 Resumen de resultados accuracy modelos entrenados con corpus combinado

Corpus noticias combinado	
Modelo	**Accuracy**
XG-Boost	0.711
BERT sin preprocesamiento del texto	0.827

De acuerdo con los resultados obtenidos se puede determinar que el comportamiento de los modelos de analítica para la clasificación de noticias, presentan un mejor desempeño cuando se utiliza el corpus de noticias del contexto colombiano. Esto se debe a que el corpus de noticias colombianas logra capturar la mayoría de las características del estilo de escritura colombiano permitiendo una mejor detección de noticias falsas.

Por otra parte, y con base en la experimentación se comprobó que, utilizando BERT se logran mejores resultados de desempeño de los modelos en cuanto a la métrica del accuracy, corroborando de esta manera lo encontrado en el estado del arte.

5.6 Detección de noticias falsas sobre la red social Twitter

En lineamiento con los objetivos del proyecto se realizó la integración vía API con la plataforma de Twitter con el fin de extraer tweets, para realizar el proceso de detección de noticias falsas con el mejor modelo.

En la ilustración 16, se presenta la arquitectura utilizada para el proceso de clasificación utilizando la API de Twitter:

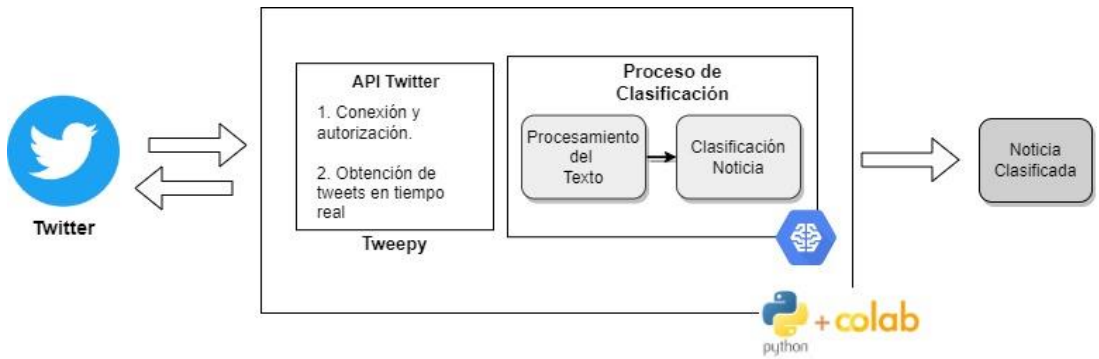

Ilustración 16 Utilización de la API de Twitter y proceso de clasificación

Esta arquitectura consta de dos principales procesos:

- Conexión API Twitter:

Para la conexión y consumo de la API se utilizó la librería tweepy[7], la cual ofrece dos grandes ventajas; la primera relacionada con el proceso de autenticación que lo hace más fácil de codificar y la segunda tiene que ver con el método cursor que optimiza el proceso de obtención de los tweets.

Para trabajar con la API de Twitter se realizaron los siguientes pasos:

- Se creó una cuenta de desarrollador/aplicación en Twitter.
- Se realizó la conexión contra la API utilizando la cuenta de desarrollador.
- Se realizó la extracción de los tweets de la cuenta de usuario a analizar.
- Se guardaron los tweets recolectados.

- Proceso de clasificación

Para el proceso de clasificación de noticias se realiza la conexión contra la API de Twitter y se extraen los Tweets de la cuenta que se desea analizar, en la ilustración 17, se presenta un ejemplo de extracción de Tweets de la cuenta de @RevistaSemana:

```
[ ]  display(df_tweets_temp)
```

Fecha Creacion	Tweet	Cuenta	Cantidad Seguidores
04/05/2022 21:22:21	El jefe máximo del Clan del Golfo, José Antoni...	Revista Semana	4893433
04/05/2022 21:22:20	Dilian Francisca Toro es una de las lideresas ...	Revista Semana	4893433
04/05/2022 21:17:06	Medellín: Clínica Las Américas alerta sobre oc...	Revista Semana	4893433
04/05/2022 21:12:16	Ministro de Salud presentó la hoja de ruta par...	Revista Semana	4893433
04/05/2022 21:12:16	¿Por qué la NASA planea enviar imágenes de per...	Revista Semana	4893433
04/05/2022 21:05:47	Final de la Champions League: ¿Cómo y dónde ve...	Revista Semana	4893433
04/05/2022 21:02:52	"Gol de este demonio": emisora española, descr...	Revista Semana	4893433
04/05/2022 21:02:51	Continúa el enfrentamiento entre el presidente...	Revista Semana	4893433
04/05/2022 21:00:29	Cali y Corinthians complicaron sus aspiracione...	Revista Semana	4893433
04/05/2022 20:57:00	Con el fin de atajar la escalada de la inflaci...	Revista Semana	4893433

Ilustración 17 Extracción de Tweets con la API de Twitter

Una vez extraído los Tweets se valida si estos contienen URLs. En caso de tenerlas, se realiza una extracción automática del texto directamente de la página web a través de la técnica Web scrapping [8], la cual extrae solamente un párrafo de la noticia con un tamaño 400 caracteres. Por el contrario, en caso de no tener URL, solamente se toma el Tweet recolectado. Una vez se tiene el texto, este se pasa al modelo de analítica para su clasificación y el resultado es entregado en probabilidades, como se muestra en la ilustración 18:

[7] https://www.tweepy.org/
[8] Técnica que permite leer texto de páginas web para obtener información y almacenarla.

```
df_news_test=classification_news_twitter("@revistasemana",10)
df_news_test
```

```
10/? [00:00<00:00, 3.04it/s]
[('La noticia es real con una probabilidad de: ', 0.7450785527528834),
 ('La noticia es real con una probabilidad de: ', 0.8430008570050893),
 ('La noticia es falsa con una probabilidad de: ', 0.6519827174810547),
 ('La noticia es falsa con una probabilidad de: ', 0.7495638874723267),
 ('La noticia es falsa con una probabilidad de: ', 0.8508963149372657),
 ('La noticia es real con una probabilidad de: ', 0.8858218856500937),
 ('La noticia es falsa con una probabilidad de: ', 0.9179202367069673),
 ('La noticia es falsa con una probabilidad de: ', 0.990923661735155),
 ('La noticia es real con una probabilidad de: ', 0.5417437386008268),
 ('La noticia es real con una probabilidad de: ', 0.5307670243777123)]
```

Ilustración 18 Proceso de predicción de noticias con el mejor modelo seleccionado

El proceso utilizado para la extracción y clasificación de noticias falsas vía Twitter se muestra en la ilustración 19:

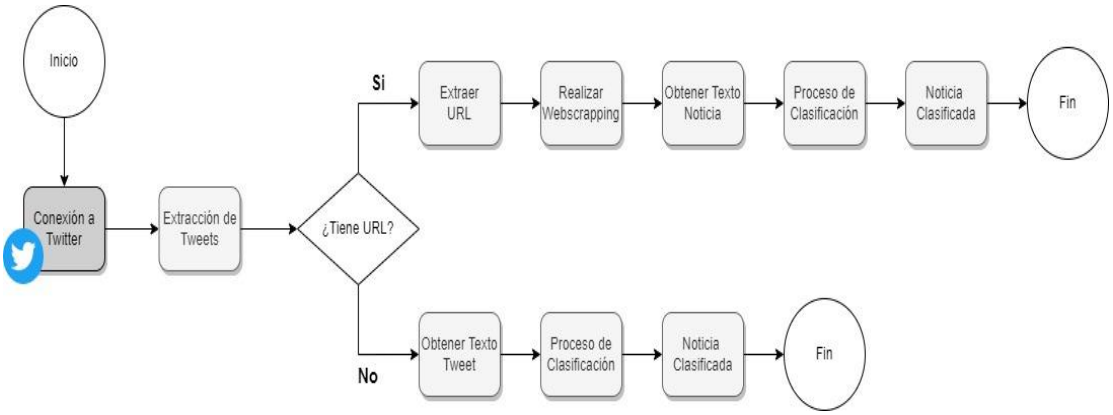

Ilustración 19 Proceso de clasificación de noticias extraídas de Twitter

6 DESPLIEGUE OPCIONAL DE MEJOR MODELO TRADICIONAL

6.1 Despliegue de modelo en Chat Bot

Para el despliegue se consideró únicamente el mejor modelo seleccionado de aprendizaje de maquina tradicional, el cual fue Random Forest. Este modelo fue exportado en formato pickle (Abder-Rahman Ali, 2017) y se utilizó para crear una API REST de manera que pueda ser utilizada desde un sistema externo; caso del proyecto desde un chatbot.

Para el caso del proyecto en esta API se implementó toda la lógica para poder utilizar el modelo de analítica seleccionado; cargue del modelo, preprocesamiento de datos y clasificación. Por otra parte, se implementó un chatbot para consumir esta API, el chatbot fue creado sobre la plataforma de Google Dialogflow, con el siguiente flujo de funcionamiento:

- El usuario saluda al bot, este le responde con un mensaje de bienvenida y le solicita que ingrese el texto de la noticia a clasificar como se muestra en la ilustración 20:

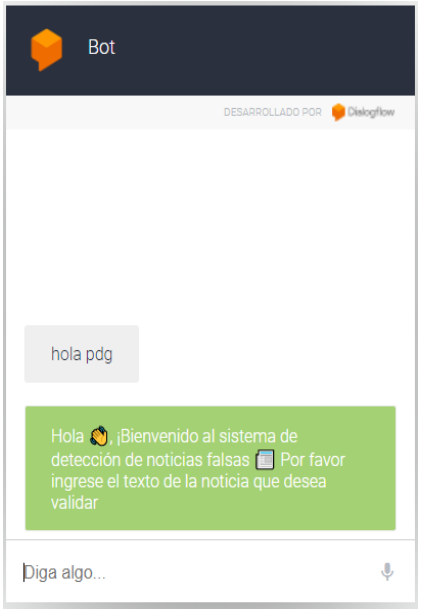

Ilustración 20 Despliegue del modelo en chatbot

Nota: Por políticas de cuotas de Google Dialogflow solo permite el ingreso de textos de máximo 254 caracteres.

- El usuario ingresa el texto de la noticia que desea clasificar y el Bot consume la API REST creada y entrega el resultado al usuario de la clasificación en probabilidades, como se muestra en la ilustración 21:

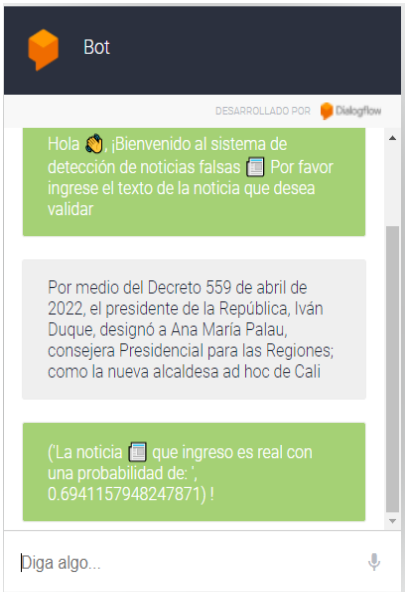

Ilustración 21 Consumo del modelo a través de chatbot para realizar la predicción

Para el caso del chatbot este fue desarrollado con la arquitectura que se presenta en la ilustración 22:

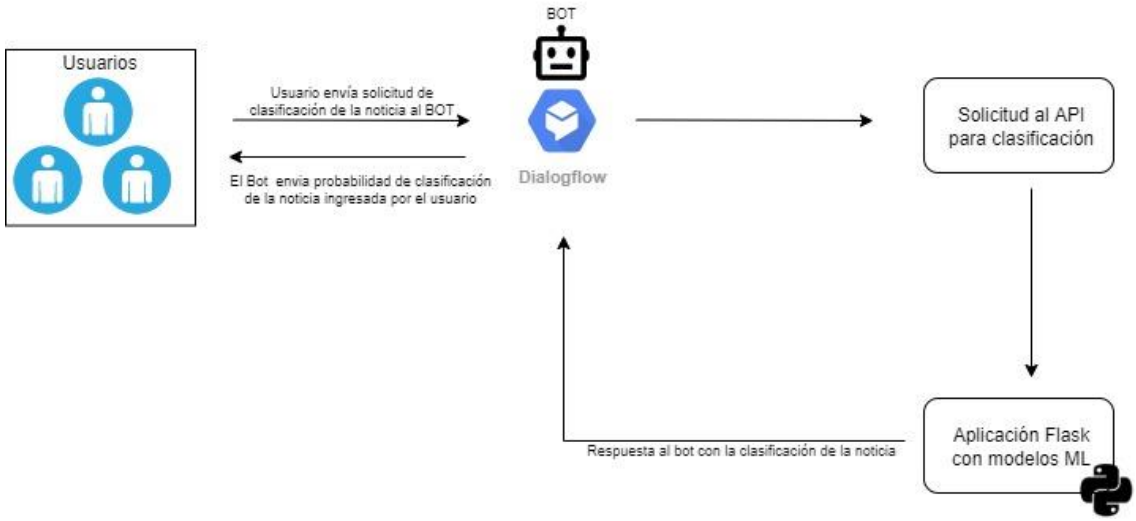

Ilustración 22 Arquitectura de despliegue de chatbot y modelo de predicción

Esta arquitectura está compuesta por:

- **Google Dialogflow Essentials:** Plataforma de Google utilizada para la creación de Bots de manera rápida e intuitiva.

- **Aplicación backend (API):** Aplicación desarrollada en Python que proporciona todos los métodos para realizar las tareas de cargue del modelo de analítica, preprocesamiento y clasificación de noticias.

7 CONCLUSIONES Y TRABAJOS FUTUROS

En cuanto al proceso de recolección de noticias falsas para la construcción del corpus en el contexto político colombiano, solo se alcanzaron a recolectar 632 noticias balanceadas, esto debido a que existen pocos portales dedicados a la identificación de este tipo de noticias, lo que dificultó tener un corpus con un número mayor de registros para el entrenamiento de los modelos.

Por el desempeño de los modelos de aprendizaje de máquina, se pudo comprobar que los modelos basados en atención como BERT, presentan un mejor desempeño en la detección de noticias falsas en el contexto político colombiano, en comparación con los modelos de aprendizaje de maquina tradicionales tales como Random Forest, Naive Bayes, SVC, regresión logística y redes neuronales, así como XG-Boost comprobando de esta manera lo encontrado en el estado del arte.

Debido al estilo de la sintaxis con la que se escriben las noticias en Colombia, se encontró que con el corpus creado en el proyecto se obtienen mejores resultados en la clasificación de noticias del contexto político colombiano en comparación con el corpus combinado que tiene noticias de diferentes países que hablan español.

Finalmente, este estudio permitió determinar que es posible la formulación de un modelo de aprendizaje maquina capaz de identificar noticias falsas en formato texto en el contexto político colombiano sobre la red social Twitter, con un buen nivel de confiabilidad basada en la métrica del accuracy del 0.95 con BERT.

Como trabajo futuro se propone abordar los siguientes temas:

Aumentar el número de noticias colombianas para el corpus creado en este proyecto, con el objetivo de mejorar las métricas de desempeño de los modelos de analítica para la clasificación de noticias falsas.

Desplegar el modelo entrenado con BERT utilizando una API tipo REST que pueda ser consumida desde un sistema externo.

Considerar más de dos categorías para la clasificación de las noticias, con el objetivo de analizar el comportamiento de las métricas de evaluación en los modelos de analítica.

8 REFERENCIAS

Abder-Rahman Ali. (2017, July 7). *Python's Pickles*. Https://Code.Tutsplus.Com/Tutorials/Pythons-Pickles--Cms-28839.

Abonizio, H. Q., de Morais, J. I., Tavares, G. M., & Junior, S. B. (2020). Language-independent fake news detection: English, Portuguese, and Spanish mutual features. *Future Internet*, *12*(5). https://doi.org/10.3390/FI12050087

Allcott, H., & Gentzkow, M. (2017). Social media and fake news in the 2016 election. In *Journal of Economic Perspectives* (Vol. 31, Issue 2, pp. 211–236). American Economic Association. https://doi.org/10.1257/jep.31.2.211

Angie Drobnic Holan, & Aaron Sharockman. (2016). *Fact-checking President-Elect Donald Trump*. POLITIFACT. https://www.politifact.com/article/2016/nov/09/fact-checking-donald-trump-2016/

Arsenii Tretiakov. (2020). noticias falsas en español. In *https://www.kaggle.com/datasets/arseniitretiakov/noticias-falsas-en-espaol*. Kaggle.

Brownlee Jason. (2017). *Deep Learning for Natural Language Processing Develop Deep Learning Models for Natural Language in Python Acknowledgements Copyright Deep Learning for Natural Language Processing*.

Claire Wardle. (2017, February). *Fake news. It's complicated*. Reosurce Centre.

Colombia Check. (2022, May 9). *Los principios de Colombiacheck*. Https://Colombiacheck.Com/Node/169.

Hagiwara, M. (2021). *Natural Language Processing*.

Helmstetter, S., & Paulheim, H. (2018). *Weakly Supervised Learning for Fake News Detection on Twitter*.

IBM Cloud Education. (2020, August 17). *Redes neuronales*. Https://Www.Ibm.Com/Co-Es/Cloud/Learn/Neural-Networks.

Jacob Devlin. (2018). *bert_multi_cased_L-12_H-768_A-12*. Https://Tfhub.Dev/Tensorflow/Bert_multi_cased_L-12_H-768_A-12/4.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, & Kristina Toutanova. (2018). *bert_multi_cased_preprocess*. Https://Tfhub.Dev/Tensorflow/Bert_multi_cased_preprocess/3.

Joaquín Amat Rodrigo. (2016, August). *Regresión logística simple y múltiple*. Https://Www.Cienciadedatos.Net/Documentos/27_regresion_logistica_simple_y_multiple.

Joaquín Amat Rodrigo. (2017, April). *Máquinas de Vector Soporte (Support Vector Machines, SVMs)*. Https://Www.Cienciadedatos.Net/Documentos/34_maquinas_de_vector_soporte_support_vector_machines.

José, A., & Gallardo Arancibia, A. (2009). *Departamento de lenguajes y sistemas informáticos e ingeniería de software Facultad de Informática "Metodología para la Definición de Requisitos en Proyectos de Data Mining (ER-DM)."*

Jose García Nieto. (2019, December 9). *Cómo funciona BERT, la inteligencia artificial con la que Google quiere conseguir que su motor de búsqueda nos entienda mejor*. Https://Www.Xataka.Com/Servicios/Como-Funciona-Bert-Inteligencia-Artificial-Que-Google-Quiere-Conseguir-Que-Su-Motor-Busqueda-Nos-Entienda-Mejor.

Jose Martinez Heras. (2020, September 18). *Random Forest (Bosque Aleatorio): combinando árboles*. Https://Www.Iartificial.Net/Random-Forest-Bosque-Aleatorio/.

Juan Bosco Mendoza Vega. (2020, August 12). *XGBoost en Python*. Https://Medium.Com/@jboscomendoza/Tutorial-Xgboost-En-Python-53e48fc58f73.

Martínez-Gallego, K., Álvarez-Ortiz, A. M., & Arias-Londoño, J. D. (2021). *Fake News Detection in Spanish Using Deep Learning Techniques*. http://arxiv.org/abs/2110.06461

Ordóñez, J., Yesid, B., López, L., Andrés, S. A., Aristizábal, A., Phd, P., De, F., Maestría, I., Ciencias, E. N., & Datos, D. E. (2021). *Modelo de procesamiento de lenguaje natural para detectar la tasa de éxito de un artículo sobre otro*. https://repository.icesi.edu.co/biblioteca_digital/handle/10906/89008

Pablo de la Varga. (2019, July 2). *¿Qué significa el término fake news en español?* Atresmedia. https://compromiso.atresmedia.com/levanta-la-cabeza/actualidad/que-significa-termino-fake-news-espanol_201903295c9e03b80cf2fb2ce3661bdf.html

Posadas-Durán. (2019). The Spanish Fake News Corpus. In *https://github.com/jpposadas/FakeNewsCorpusSpanish*.

Posadas-Durán, J. P., Gomez-Adorno, H., Sidorov, G., & Escobar, J. J. M. (2019). Detection of fake news in a new corpus for the Spanish language. *Journal of Intelligent and Fuzzy Systems*, *36*(5), 4868–4876. https://doi.org/10.3233/JIFS-179034

Rosario Peiró. (2020, July 6). *Fake news (noticias falsas)*. Economipedia. https://economipedia.com/definiciones/fake-news-noticias-falsas.html

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need*. http://arxiv.org/abs/1706.03762

Victor Roman. (2019, April 25). *Algoritmos Naive Bayes: Fundamentos e Implementación*. Https://Medium.Com/Datos-y-Ciencia/Algoritmos-Naive-Bayes-Fudamentos-e-Implementaci%C3%B3n-4bcb24b307f.

Vogel, I., & Meghana, M. (2020). Detecting fake news spreaders on twitter from a multilingual perspective. *Proceedings - 2020 IEEE 7th International Conference on Data Science and Advanced Analytics, DSAA 2020*, 599–606. https://doi.org/10.1109/DSAA49011.2020.00084

Vosoughi, S., Roy, D., & Aral, S. (2018). *False news is big news. the spread of true and false news online*.

Zhou, X., & Zafarani, R. (2018). *A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities*. https://doi.org/10.1145/3395046

9 ANEXOS

9.1 Experimentos ejecutados con modelos tradicionales:

Se realizaron 7 experimentos diferentes modificando los parámetros de representación del texto TF-IDF y los parámetros de la búsqueda aleatoria de hiperparámetros buscando encontrar los mejores valores de los hiperparámetros que entregaran un mejor desempeño en el proceso de clasificación de noticias falsas. Estos experimentos se presentan a continuación:

- Experimento 1:

Random Search		Parámetros TF-IDF			
n_iter	cv	ngram_range	min_df	max_df	max_features
10	3	(1,3)	0.001	0.8	300

Corpus noticias colombianas			
Modelo	Accuracy	Precision-recall Curve class 1	Hiperparámetros
Random Forest	0.897638	0.963	{'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_features': 'sqrt', 'max_depth': 100, 'bootstrap': False}
Naive Bayes	0.874016	0.962	{'fit_prior': True, 'alpha': 1}
SVC	0.889764	0.971	{'kernel': 'rbf', 'gamma': 1, 'C': 100}
Logistic Regression	0.897638	0.976	{'penalty': 'l2', 'C': 4.037017258596554}
XGBoost	0.88189	0.943	{'subsample': 1.0, 'scale_pos_weight': 1, 'reg_alpha': 0.01, 'n_estimators': 1000, 'min_child_weight': 1, 'max_depth': 5, 'learning_rate': 0.1, 'gamma': 1, 'eta_vals': 0.1, 'colsample_bytree': 0.3}

Corpus combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.691843	0.771	{'n_estimators': 400, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 40, 'bootstrap': True}
Naive Bayes	0.625378	0.689	{'fit_prior': True, 'alpha': 1e-05}
SVC	0.693353	0.764	{'kernel': 'rbf', 'gamma': 1, 'C': 100}
Logistic Regression	0.678248	0.727	{'penalty': 'l2', 'C': 0.7564633275546291}
XGBoost	0.71148	0.75	{'subsample': 0.9, 'scale_pos_weight': 4, 'reg_alpha': 0.01, 'n_estimators': 1000, 'min_child_weight': 1, 'max_depth': 9, 'learning_rate': 0.05, 'gamma': 1.5, 'eta_vals': 0.01, 'colsample_bytree': 0.3}

- Experimento 2:

Random Search		Parametros TF-IDF			
n_iter	**cv**	**ngram_range**	**min_df**	**max_df**	**max_features**
100	5	(1,3)	0.001	0.8	300

Corpus noticias colombianas			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.889764	0.965	{'n_estimators': 1000, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 80, 'bootstrap': True}
Naive Bayes	0.88189	0.973	{'fit_prior': True, 'alpha': 0.01}
SVC	0.905512	0.966	{'kernel': 'sigmoid', 'gamma': 1, 'C': 1}
Logistic Regression	0.897638	0.973	{'penalty': 'l2', 'C': 7.56463327554629}
XGBoost	0.88189	0.947	{'subsample': 1.0, 'scale_pos_weight': 1, 'reg_alpha': 0.1, 'n_estimators': 600, 'min_child_weight': 1, 'max_depth': 7, 'learning_rate': 0.05, 'gamma': 0.1, 'eta_vals': 0.001, 'colsample_bytree': 0.5}

Corpus combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.690332	0.771	{'n_estimators': 600, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 60, 'bootstrap': False}
Naive Bayes	0.634441	0.689	{'fit_prior': False, 'alpha': 0.01}
SVC	0.694864	0.756	{'kernel': 'rbf', 'gamma': 1, 'C': 1}
Logistic Regression	0.679758	0.729	{'penalty': 'l2', 'C': 4.9770235643321135}
XGBoost	0.691843	0.761	{'subsample': 1.0, 'scale_pos_weight': 2, 'reg_alpha': 0.1, 'n_estimators': 600, 'min_child_weight': 1, 'max_depth': 9, 'learning_rate': 0.05, 'gamma': 0.1, 'eta_vals': 0.001, 'colsample_bytree': 1.0}

- Experimento 3:

Random Search		Parametros TF-IDF			
n_iter	cv	ngram_range	min_df	max_df	max_features
100	5	(4,4)	0.001	0.8	300

Corpus noticias combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.496063	0.564	{'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 40, 'bootstrap': True}
Naive Bayes	0.488189	0.564	{'fit_prior': True, 'alpha': 1}
SVC	0.574803	0.568	{'kernel': 'rbf', 'gamma': 1, 'C': 1}
Logistic Regression	0.574803	0.564	{'penalty': 'l2', 'C': 0.01}
XGBoost	0.480315	0.537	{'subsample': 0.8, 'scale_pos_weight': 1, 'reg_alpha': 0.05, 'n_estimators': 200, 'min_child_weight': 1, 'max_depth': 5, 'learning_rate': 0.9, 'gamma': 1, 'eta_vals': 0.001, 'colsample_bytree': 0.5}

Corpus combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.489426	0.544	{'n_estimators': 800, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 60, 'bootstrap': False}
Naive Bayes	0.487915	0.541	{'fit_prior': False, 'alpha': 0.1}
SVC	0.490937	0.543	{'kernel': 'linear', 'gamma': 1, 'C': 10}
Logistic Regression	0.537764	0.544	{'penalty': 'l2', 'C': 8.111308307896872}
XGBoost	0.484894	0.537	{'subsample': 0.8, 'scale_pos_weight': 1, 'reg_alpha': 0.05, 'n_estimators': 200, 'min_child_weight': 1, 'max_depth': 5, 'learning_rate': 0.9, 'gamma': 1, 'eta_vals': 0.001, 'colsample_bytree': 0.5}

- Experimento 4:

Random Search		Parametros TF-IDF			
n_iter	cv	ngram_range	min_df	max_df	max_features
100	5	(4,4)	0.001	0.8	3000

Corpus noticias colombianas			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.496063	0.568	{'n_estimators': 800, 'min_samples_split': 10, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 40, 'bootstrap': False}
Naive Bayes	0.503937	0.599	{'fit_prior': True, 'alpha': 0.0001}
SVC	0.590551	0.599	{'kernel': 'rbf', 'gamma': 0.001, 'C': 1000}
Logistic Regression	0.590551	0.599	{'penalty': 'l2', 'C': 0.02009233002565047}
XGBoost	0.480315	0.528	{'subsample': 0.8, 'scale_pos_weight': 1, 'reg_alpha': 0.05, 'n_estimators': 200, 'min_child_weight': 1, 'max_depth': 5, 'learning_rate': 0.9, 'gamma': 1, 'eta_vals': 0.001, 'colsample_bytree': 0.5}

Corpus combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.492447	0.544	{'n_estimators': 800, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 60, 'bootstrap': False}
Naive Bayes	0.487915	0.541	{'fit_prior': False, 'alpha': 0.1}
SVC	0.490937	0.543	{'kernel': 'linear', 'gamma': 1, 'C': 10}
Logistic Regression	0.537764	0.544	{'penalty': 'l2', 'C': 8.111308307896872}
XGBoost	0.484894	0.537	{'subsample': 0.8, 'scale_pos_weight': 1, 'reg_alpha': 0.05, 'n_estimators': 200, 'min_child_weight': 1, 'max_depth': 5, 'learning_rate': 0.9, 'gamma': 1, 'eta_vals': 0.001, 'colsample_bytree': 0.5}

- Experimento 5:

Random Search		Parametros TF-IDF			
n_iter	cv	ngram_range	min_df	max_df	max_features
100	5	(2,4)	0.001	0.8	3000

Corpus noticias colombianas			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.7402	0.902	{'n_estimators': 1000, 'min_samples_split': 10, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 100, 'bootstrap': False}
Naive Bayes	0.6378	0.897	{'fit_prior': True, 'alpha': 0.0001}
SVC	0.8661	0.894	{'kernel': 'rbf', 'gamma': 0.1, 'C': 10}
Logistic Regression	0.8661	0.885	{'penalty': 'l2', 'C': 0.7564633275546291}
XGBoost	0.8425	0.737	{'subsample': 1.0, 'scale_pos_weight': 1, 'reg_alpha': 0.1, 'n_estimators': 600, 'min_child_weight': 1, 'max_depth': 7, 'learning_rate': 0.05, 'gamma': 0.1, 'eta_vals': 0.001, 'colsample_bytree': 0.5}

Corpus combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.6299	0.668	{'n_estimators': 1000, 'min_samples_split': 10, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': None, 'bootstrap': True}
Naive Bayes	0.5786	0.689	{'fit_prior': True, 'alpha': 1e-05}
SVC	0.6314	0.643	{'kernel': 'rbf', 'gamma': 0.1, 'C': 10}
Logistic Regression	0.6375	0.68	{'penalty': 'l2', 'C': 8.111308307896872}
XGBoost	0.6088	0.628	{'subsample': 0.9, 'scale_pos_weight': 1, 'reg_alpha': 0.01, 'n_estimators': 200, 'min_child_weight': 1, 'max_depth': 7, 'learning_rate': 0.1, 'gamma': 0.1, 'eta_vals': 0.01, 'colsample_bytree': 0.5}

- Experimento 6:

Random Search		Parametros TF-IDF			
n_iter	**cv**	**ngram_range**	**min_df**	**max_df**	**max_features**
100	10	(1,3)	0.001	0.8	300

Corpus noticias colombianas			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.897638	0.964	{'n_estimators': 600, 'min_samples_split': 5, 'min_samples_leaf': 2, 'max_features': 'auto', 'max_depth': 100, 'bootstrap': True}
Naive Bayes	0.897638	0.971	{'fit_prior': False, 'alpha': 0.1}
SVC	0.913386	0.974	{'kernel': 'rbf', 'gamma': 1, 'C': 1}
Logistic Regression	0.92126	0.978	{'penalty': 'l2', 'C': 1.5199110829529332}
XGBoost	0.88189	0.955	{'subsample': 0.9, 'scale_pos_weight': 2, 'reg_alpha': 0.01, 'n_estimators': 800, 'min_child_weight': 1, 'max_depth': 7, 'learning_rate': 0.9, 'gamma': 1, 'eta_vals': 0.01, 'colsample_bytree': 0.3}

Corpus combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.691843	0.771	{'n_estimators': 400, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 100, 'bootstrap': True}
Naive Bayes	0.634441	0.689	{'fit_prior': False, 'alpha': 0.01}
SVC	0.694864	0.756	{'kernel': 'rbf', 'gamma': 1, 'C': 1}
Logistic Regression	0.681269	0.725	{'penalty': 'l2', 'C': 0.3511191734215131}
XGBoost	0.691843	0.761	{'subsample': 1.0, 'scale_pos_weight': 2, 'reg_alpha': 0.1, 'n_estimators': 600, 'min_child_weight': 1, 'max_depth': 9, 'learning_rate': 0.05, 'gamma': 0.1, 'eta_vals': 0.001, 'colsample_bytree': 1.0}

- Experimento 7:

Random Search		Parametros TF-IDF			
n_iter	**cv**	**ngram_range**	**min_df**	**max_df**	**max_features**
200	10	(1,3)	0.001	0.8	300

Corpus noticias colombianas			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.897638	0.964	{'n_estimators': 1000, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_features': 'auto', 'max_depth': 40, 'bootstrap': True}
Naive Bayes	0.897638	0.971	{'fit_prior': False, 'alpha': 0.1}
SVC	0.913386	0.974	{'kernel': 'rbf', 'gamma': 1, 'C': 1}
Logistic Regression	0.92126	0.978	{'penalty': 'l2', 'C': 1.5199110829529332}
XGBoost	0.88189	0.955	{'subsample': 0.9, 'scale_pos_weight': 2, 'reg_alpha': 0.01, 'n_estimators': 800, 'min_child_weight': 1, 'max_depth': 7, 'learning_rate': 0.9, 'gamma': 1, 'eta_vals': 0.01, 'colsample_bytree': 0.3}

Corpus combinado			
Modelo	**Accuracy**	**Precision-recall Curve class 1**	**Hiperparámetros**
Random Forest	0.687311	0.772	{'n_estimators': 800, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 60, 'bootstrap': True}
Naive Bayes	0.634441	0.689	{'fit_prior': False, 'alpha': 0.01}
SVC	0.694864	0.756	{'kernel': 'rbf', 'gamma': 1, 'C': 1}
Logistic Regression	0.681269	0.725	{'penalty': 'l2', 'C': 0.3511191734215131}
XGBoost	0.691843	0.761	{'subsample': 1.0, 'scale_pos_weight': 2, 'reg_alpha': 0.1, 'n_estimators': 600, 'min_child_weight': 1, 'max_depth': 9, 'learning_rate': 0.05, 'gamma': 0.1, 'eta_vals': 0.001, 'colsample_bytree': 1.0}

9.2 Experimentos ejecutados con redes neuronales tradicionales

La ejecución de experimentos con redes neuronales tradicionales se realizó tanto para el corpus del contexto político colombiano como para el corpus combinado. Para estos modelos se realizó el preprocesamiento del texto descrito en la sección 4.2.1.

Redes neuronales tradicionales											
Corpus	**Número de experimento**	**capas**	**Tamaño datos de validación**	**Tamaño de datos de entrenamiento**	**Tamaño datos de prueba**	**batch_size**	**épocas**	**Accuracy con datos de validación**	**Accuracy con datos de prueba**	**Pérdida**	**Tiempo de entrenamiento**
Corpus noticias contexto político colombiano	Experimento 1	Dense(12, input_dim=features_train_cp.shape[1], activation='relu')) Dense(8 , activation='relu')) Dense(1 , activation='sigmoid'))	100 News	407 News	127 News	256	500	0.8900	0.8740	0.2606	4.03 s
	Experimento 2	Dense(12, input_dim=features_train_cp.shape[1], activation='relu')) Dense(8 , activation='relu')) Dense(64 , activation='relu')) Dense(32 , activation='relu')) Dense(8 , activation='relu')) Dense(1 , activation='sigmoid'))	100 News	407 News	127 News	32	1000	0.88	0.8661	0.5216	7.12 s
	Experimento 3	Dense(12, input_dim=features_train_cp.shape[1], activation='relu')) Dense(64 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(1 , activation='sigmoi	100 News	407News	127 News	32	200	0.9000	0.8898	0.6026	6.96 s
	Experimento 4	Dense(12, input_dim=features_train_cp.shape[1], activation='relu')) Dense(128 , activation='relu')) Dense(64 , activation='relu')) Dense(64 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(8 , activation='relu')) Dense(1 , activation='sigmoid'))	100 News	407News	127 News	32	200	0.8900	0.8583	10.435	5.35 s
Corpus noticias combinado	Experimento 1	Dense(12, input_dim=features_train_cp.shape[1], activation='relu')) Dense(8 , activation='relu')) Dense(1 , activation='sigmoid'))	529 News	2119 News	127 News	256	500	0.6767	0.6647	0.8974	7.53 s
	Experimento 2	Dense(12, input_dim=features_train.shape[1], activation='relu')) Dense(8 , activation='relu')) Dense(64 , activation='relu')) Dense(32 , activation='relu')) Dense(8 , activation='relu')) Dense(1 , activation='sigmoid'))	529 News	2119 News	127 News	32	1000	0.6522	0.6299	1,7683	13.5 s
	Experimento 2	Dense(12, input_dim=features_train_cp.shape[1], activation='relu')) Dense(64 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(1 , activation='sigmoi	529 News	2119 News	127 News	32	200	0.6616	0.6647	0.8974	18.9 s
	Experimento 4	Dense(12, input_dim=features_train_cp.shape[1], activation='relu')) Dense(128 , activation='relu')) Dense(64 , activation='relu')) Dense(64 , activation='relu')) Dense(32 , activation='relu')) Dense(32 , activation='relu')) Dense(8 , activation='relu')) Dense(1 , activation='sigmoid'))	100 News	407News	127 News	32	200	0.6446	0.6647	0.8974	13.3 s

9.3 Experimentos ejecutados con bert_multi_cased_L-12_H-768_A-12

Los experimentos ejecutados con bert_multi_cased_L-12_H-768_A-12 para el corpus de noticias colombianas y el corpus combinado se presenta a continuación.

El entrenamiento de BERT se realiza con preprocesamiento del texto y sin preprocesamiento previo del texto.

Nota: Sin preprocesamiento de texto significa que no se realiza el proceso de quitar tildes, caracteres especiales, signos de puntuación, lematización, etc. Esto debido para el caso de BERT el preprocesamiento del texto se realiza con el módulo bert_multi_cased_preprocess.

Como se muestra en la ejecución de los experimentos, BERT presenta un mejor desempeño cuando no se realiza limpieza del texto previamente.

Experimentos ejecutados con bert_multi_cased_L-12_H-768_A-12											
Corpus	**Numero total de noticias**	**Tamaño de datos de validacion**	**Tamaño de datos de entrenamiento**	**Tamaño datos de prueba**	**Batch Size**	**Épocas**	**Learning rate**	**Accuracy datos de validacion**	**Accuracy datos de prueba**	**Pérdida**	**Tiempo de entrenamiento**
Corpus noticias colombianas sin preprocesamiento de texto	634	102	412	120	32	5	0,000030	0,902	0,925	1,26	2 minutos
Corpus noticias colombianas con preprocesamiento de texto	634	102	412	120	32	20	0,000030	0,902	0,8833	0,6591	3min 45s
Corpus noticias combiando con preprocesamiento del texto	3310	529	2119	662	32	20	0,000030	0,778	0,744	0,2508	25min 32s
Corpus noticias varias sin preprocesar	3310	529	2119	662	32	20	0,000030	0.8412	0.8338	1,0279	25min 28s
Corpus noticias combinado sin preprocesar, cambiando el nuemeor de datos de entrenamiento y prueba	3310	596	2384	330	32	20	0,000030	0.8339	0.8242	1,1135	26min 53s