REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Universidad Católica de Santa María
Facultad de Ciencias e Ingenierías Físicas y Formales
Escuela Profesional de Ingeniería Electrónica

IMPLEMENTACIÓN DE CLASIFICADOR DE NOTICIAS EN IDIOMA ESPAÑOL PARA LA IDENTIFICACIÓN DE FAKE NEWS MEDIANTE EL ANÁLISIS, TRADUCCIÓN AUTOMÁTICA Y VALIDACIÓN DE UN CONJUNTO DE DATOS EN INGLÉS, Y EL USO DE TÉCNICAS DE APRENDIZAJE MÁQUINA Y PROCESAMIENTO DE LENGUAJE NATURAL.

Tesis presentada por el Bachiller:
Obando Molina, Felix Antonio Junjiro

para optar el Título Profesional de:
Ingeniero Electrónico con Especialidad en Telecomunicaciones

Asesor:
Mg. Rivera Chavez, Victor Hugo

Arequipa - Perú

2021

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

DICTAMEN APROBATORIO DEL BORRADOR DE TESIS

UCSM-ERP

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

INGENIERIA ELECTRONICA

TITULACIÓN CON TESIS

DICTAMEN APROBACIÓN DE BORRADOR

Arequipa, 16 de Abril del 2021

Dictamen: 001131-C-EPIE-2021

Visto el borrador del expediente 001131, presentado por:

2015400241 - OBANDO MOLINA FELIX ANTONIO JUNJIRO

Titulado:

IMPLEMENTACIÓN DE CLASIFICADOR DE NOTICIAS EN IDIOMA ESPAÑOL PARA LA IDENTIFICACIÓN DE FAKE NEWS MEDIANTE EL ANÁLISIS, TRADUCCIÓN AUTOMÁTICA Y VALIDACIÓN DE UN CONJUNTO DE DATOS EN INGLÉS, Y EL USO DE TÉCNICAS DE APRENDIZAJE MÁQUINA Y PROCESAMIENTO DE LENGUAJE NATURAL

Nuestro dictamen es:

APROBADO

1531 - RIVERA CHAVEZ VICTOR HUGO
DICTAMINADOR

1546 - DELGADO BARRA LUCY ANGELA
DICTAMINADOR

1767 - SULLA TORRES RAUL RICARDO
DICTAMINADOR

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

DEDICATORIA

A mi madre, quien me motivó a realizar esta tesis, y a quien agradezco el ser quien soy. A mi papá, quien con su ejemplo me enseñó a ser un hombre de bien.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

AGRADECIMIENTO

Agradezco a mi asesor, el ingeniero Victor Hugo, cuyas observaciones me ayudaron a complementar este trabajo. A todos mis profesores que formaron parte de mi formación académica.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

RESUMEN

El presente trabajo presenta la implementación de un clasificador de noticias que permite la identificación de noticias falsas en el idioma español. Dicho clasificador se basa en el entrenamiento de modelos de aprendizaje automático supervisado mediante el uso de técnicas y herramientas de procesamiento de lenguaje natural.

Uno de los mayores desafíos de este trabajo es la escasez de conjuntos de datos en español que puedan ser utilizados para el entrenamiento de los modelos de aprendizaje automático. Como respuesta a este desafío, se hace uso de la metodología del *backtranslation* y de la métrica METEOR (Banerjee y Lavie 2005) para la evaluación de la traducción automática de un conjunto de datos de noticias falsas en idioma inglés hacia el idioma español. Dicho conjunto de datos traducido es después utilizado como fuente de datos para el entrenamiento del modelo de aprendizaje automático.

Debido a que ningún modelo de aprendizaje automático puede utilizar una muestra textual directamente para su entrenamiento, se realiza la implementación de un transformador que permite la extracción de características semánticas, sintácticas y de polaridad. Las características semánticas son extraídas mediante el uso de un modelo de aprendizaje automático que permite la obtención de vectores de palabras que codifican las relaciones semánticas y de significado entre palabras. Las características sintácticas son expresadas mediante la obtención de etiquetas de parte del discurso y de nombres de entidades. Las características de polaridad son obtenidas con un léxico de sentimientos en el idioma español.

Se realiza además el despliegue de dicho modelo en un sistema web para su uso por usuarios externos.

Palabras clave: Inteligencia artificial, aprendizaje automático, aprendizaje máquina, vectorización, Word embeddings, vectores de palabras, procesamiento de lenguaje natural, análisis de sentimientos, noticias falsas.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

ABSTRACT

This work implements a news classifier that allows for the identification of fake news in the Spanish language. The classifier is based on the training of supervised machine learning models using tools and techniques of the natural language processing field.

One of the greatest challenges for the training of the machine learning model is the lack of a Spanish news dataset. To tackle this challenge the methodology of the backtranslation and the METEOR (Banerjee y Lavie 2005) metric is used, so as to evaluate the machine translation of an English fake news dataset to the Spanish language. The translated dataset is then used as a dataset source for the training of the machine learning model.

Because no machine learning model can use a textual sample directly for training, a transformer is used to allow the extraction of semantic, syntactic and polarity characteristics. The semantic characteristics are extracted using a machine learning model that vectorizes and encodes semantic and meaning relations between words. The syntactic characteristics are expressed through the obtention of part of speech and named entity recognition tags. The polarity of a sample is obtained with a Spanish sentiment lexicon.

The deployment of the resulting model on a web system for the use of third users is also performed.

Keywords: Artificial intelligence, machine learning, word embeddings, word vectors, natural language processing, sentiment analysis, classifier, fake news.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

INTRODUCCIÓN

El desarrollo tecnológico de las últimas décadas ha traído un sin número de beneficios. Por ejemplo, encontramos el desarrollo de tecnologías que han mejorado nuestra calidad de vida, o que nos han permitido la sustentación y fortalecimiento de derechos humanos que nos aseguran una vida digna como individuos, y una convivencia como sociedad. El desarrollo de tecnologías de comunicación ha contribuido enormemente a este último fin.

De esta manera, en los últimos años hemos visto el crecimiento y desarrollo acelerado de tecnologías que hoy en día forman parte de nuestra cotidianidad. El internet, redes sociales, y dispositivos para acceder a estos medios, como son computadores portátiles y de escritorio, o los denominados teléfonos inteligentes, forman parte de ellas. Estos medios nos han permitido el libre y fácil acceso a la información, lo que es tan importante para el fortalecimiento del propio juicio, y a su vez nos ha permitido el sustentar derechos tan importantes como la libertad de pensamiento, y la libertad de expresión.

Sin embargo, la facilidad y velocidad para acceder y compartir información ha alcanzado tales magnitudes que, contradictoriamente a uno de sus fines más primordiales, ha dado lugar a la fácil y rápida propagación de información falsa o fabricada, que a su vez atenta contra la formación de un juicio objetivo basado en hechos verdaderos. Este fenómeno, es conocido como noticias falsas, o, más comúnmente por su uso en inglés: Fake News.

El presente proyecto de investigación busca hacer frente al fenómeno conocido como Fake News, o noticias falsas por su traducción del inglés, mediante la implementación de un clasificador de noticias basado en algoritmos de aprendizaje automático supervisado y técnicas propias de procesamiento de lenguaje natural para poder determinar la clasificación de una noticia como falsa o confiable.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ÍNDICE GENERAL

DICTAMEN APROBATORIO DEL BORRADOR DE TESIS ii

DEDICATORIA iii

AGRADECIMIENTO iv

RESUMEN v

ABSTRACT vi

INTRODUCCIÓN vii

ÍNDICE GENERAL viii

ÍNDICE DE FIGURAS xvi

ÍNDICE DE TABLAS xviii

CAPÍTULO I 1

1. PLANTEAMIENTO TEÓRICO 1

1.1. Título de Investigación 1

1.2. Descripción del problema. 2

1.3. Justificación de la Investigación 2

1.4. Línea general del desarrollo del proyecto 4

1.5. Hipótesis. 7

1.6. Objetivos 7

1.6.1. Objetivo General *7*

1.6.2. Objetivos Específicos *7*

1.7. Alcance 8

CAPÍTULO II 10

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

2. MARCO TEÓRICO..... 10

2.1. HERRAMIENTAS O LIBRERÍAS 10

2.1.1. Python…..... 11

2.1.2. Jupyter Notebook. 11

2.1.3. Matplotlib..... 11

2.1.4. Seaborn. 12

2.1.5. Pandas.. 12

2.1.6. Natural Language Toolkit (NLTK) 12

2.1.7. Polyglot. 13

2.1.8. SpaCy…..... 13

2.1.9. Flask…. 14

2.1.10. Scikit-learn..... 14

2.1.11. Joblib…..... 14

2.1.12. Angular 15

2.2. MARCO CONCEPTUAL..... 15

2.2.1. Conceptos Generales. 15

2.2.1.1. Inteligencia Artificial..... 15

2.2.1.2. Machine Learning o Aprendizaje Máquina. 16

2.2.1.3. NLP o Natural Language Processing..... 17

2.2.1.4. Machine Translation o Traducción Automática..... 18

2.2.1.5. Fake News..... 19

2.2.2. Exploración, análisis, limpieza y muestreo de conjunto de datos 19

2.2.2.1. Boxplot..... 19

2.2.2.2. Modelos de Lenguajes y n-gramas 20

2.2.2.3. Extracción de características del texto y vectorización 21

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

2.2.2.4. Modelo de Bolsa de Palabras o Bolsa de n-gramas..... 22

2.2.2.5. Vectorización por frecuencia de términos y TF-IDF..... 24

2.2.2.5.1. Por Frecuencia de Términos 24

2.2.2.5.2. Por TF-IDF 24

2.2.3. Ejecución y validación de traducción automática..... 26

2.2.3.1. Backtranslation y validación de traducción..... 26

2.2.3.2. Cloud Translation: elección de modelo, versión y formato..... 28

2.2.3.3. Métrica BLEU (Bilingual Evaluation Understudy) para validación de traducción……..... 30

2.2.3.3.1. Problemas con la métrica BLEU..... 32

2.2.3.4. Métrica METEOR (Metric for Evaluation of Translation with Explicit Ordering) para validación de traducción 33

2.2.3.4.1. Problemas de métrica BLEU considerados para el desarrollo de METEOR…….. 33

2.2.3.4.2. Fundamentos de métrica METEOR..... 35

2.2.3.4.3. Stemming..... 38

2.2.4. Extracción de Características: transformación a datos numéricos..... 39

2.2.4.1. Word Embeddings..... 39

2.2.4.1.1. Entrenamiento de vectores de palabras..... 41

2.2.4.1.2. Similitud y relaciones lingüísticas de vectores de palabras..... 42

2.2.4.1.3. Algoritmo para el entrenamiento de Word Embeddings 43

2.2.4.2. Extracción de características complejas y de estilo 44

2.2.4.2.1. Identificación de etiquetas de POS y NER 45

2.2.4.3. Extracción de características psicológicas o análisis de sentimiento..... 46

2.2.4.3.1. Análisis de sentimientos por aprendizaje automático 47

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

2.2.4.3.2. Análisis de sentimientos por uso de léxico 47

2.2.4.3.3. Selección de aproximación para análisis de sentimientos 48

2.2.5. Entrenamiento, evaluación y comparación de modelos de Aprendizaje Automático…..... 48

2.2.5.1. Modelos de Aprendizaje Automático..... 49

2.2.5.1.1. Regresión Logística 49

2.2.5.1.2. Máquinas de Soporte Vectorial..... 50

2.2.5.1.3. Bosques aleatorios 53

2.2.5.2. Métricas de clasificación para evaluación de desempeño 54

2.2.5.2.1. Matriz de confusión..... 54

2.2.5.2.2. Exactitud..... 56

2.2.5.2.3. Sensibilidad..... 57

2.2.5.2.4. Precisión..... 57

2.2.5.2.5. Puntaje F1..... 58

2.2.5.2.6. Elección de métrica de evaluación 58

CAPÍTULO III..... 59

3. DESARROLLO..... 59

3.1. EXPLORACIÓN, ANÁLISIS, LIMPIEZA Y MUESTREO DE CONJUNTO DE DATOS 62

3.1.1. Selección de conjunto de datos. 62

3.1.1.1. Limitaciones del conjunto de datos. 63

3.1.1.2. Columnas del conjunto de datos..... 63

3.1.1.3. Tipos de Etiquetas..... 64

3.1.2. Extracción de Conjunto de Datos 66

3.1.3. Exploración, análisis y limpieza de conjunto de datos..... 67

3.1.3.1. Exploración por longitud de caracteres en título..... 67

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

3.1.3.2. Exploración de dominio 69

3.1.3.3. Exploración de longitud de caracteres por título para tipo 'fake' y 'reliable'…….. 72

3.1.3.4. Exploración de n-gramas 73

3.1.3.5. Exploración de inconsistencias 79

3.1.3.6. Exploración de distribución de número de palabras por título 80

3.1.4. Muestreo 86

3.2. EJECUCIÓN Y VALIDACIÓN DE TRADUCCIÓN AUTOMÁTICA 87

3.2.1. Adecuación y traducción 89

3.2.2. Validación y traducción 90

3.2.2.1. Construcción del conjunto de datos 90

3.2.2.2. Especificaciones para métricas BLEU y METEOR 90

3.2.2.3. Análisis de resultados de métricas BLEU y METEOR 91

3.2.2.4. Exportando resultados 94

3.3. EXTRACCIÓN DE CARACTERÍSTICAS 95

3.3.1. Word Embeddings 96

3.3.1.1. Descripción de Modelo 97

3.3.2. Características complejas y estilísticas 97

3.3.2.1. Descripción de características 97

3.3.2.2. Descripción de modelo utilizado 99

3.3.3. Características psicológicas: análisis de sentimientos 100

3.4. ENTRENAMIENTO, EVALUACIÓN Y COMPARACIÓN DE MODELOS DE APRENDIZAJE AUTOMÁTICO 101

3.4.1. Exploración y análisis de conjunto de datos 103

3.4.2. Implementación de modelos de aprendizaje automático 107

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

3.4.2.1. Primera Etapa: Preprocesamiento y Vectorización 107

3.4.2.2. Segunda Etapa: Entrenamiento de modelos de clasificación 108

3.4.3. Predicción y evaluación 109

3.4.4. Elección de modelo y observaciones 113

3.5. DESPLIEGUE DE MODELO DE APRENDIZAJE AUTOMÁTICO: DESARROLLO DE SISTEMA WEB PARA ENCAPSULADO Y PRESENTACIÓN DE MODELO DE CLASIFICACIÓN 115

3.5.1. Implementación de un modelo base 116

3.5.2. Adopción de una arquitectura de un sistema de aprendizaje automático 118

3.5.2.1. Tipos de entrenamiento y de predicción de modelos de aprendizaje automático 118

3.5.2.1.1. Entrenamiento en tiempo real de modelo 119

3.5.2.1.2. Entrenamiento por lotes de modelo 119

3.5.2.1.3. Predicción en tiempo real de modelo 119

3.5.2.1.4. Predicción por lotes de modelo 120

3.5.2.2. Tipos de arquitectura de sistemas de Aprendizaje Automático 121

3.5.2.3. Elección de Arquitectura 122

3.5.3. Adaptación de código para despliegue 123

3.5.3.1. Web API 124

3.5.3.2. Transformador 125

3.5.3.3. Modelo entrenado y almacenado 126

3.5.3.3.1. Tuberías 126

3.5.3.3.2. Serialización 126

3.5.4. Despliegue de sistema de aprendizaje automático 127

3.5.4.1. AI Platform 128

3.5.4.2. App Engine 128

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

3.6. INTERFAZ DE USUARIO 129

3.7. RESULTADOS: EVALUACIÓN DE IMPLEMENTACIÓN 133

CAPÍTULO IV 135

4. CONCLUSIONES Y RECOMENDACIONES 135

4.1. CONCLUSIONES 135

4.1. RECOMENDACIÓN PARA IMPLEMENTACIONES FUTURAS 136

REFERENCIAS 137

ANEXOS 148

CÓDIGO DE PROGRAMACIÓN 148

Extracción por partes de Conjunto de datos 148

Función con expresiones regulares usada para eliminar nombres de dominio repetidos en título 149

Función con expresiones regulares para extracción de patrón Page + número de títulos 150

Función de solicitud de traducción 151

Función de extracción de características 152

Vectorizador de palabras 155

Clase para concatenación y adecuación de vectores de muestras 158

Main.py 159

pipeline.py 160

predict.py 161

train_pipeline.py 162

transformer.py 163

data_management.py 168

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

MATRICES DE CONFUSIÓN PARA EVALUACIÓN DE MODELOS 169

Matrices de confusión para modelos de máquinas de soporte vectorial 169

Matrices de confusión para modelos de Regresión Logística 171

Matrices de confusión para modelos de Bosques aleatorios: 100 árboles 172

Matrices de confusión para modelos de Bosques aleatorios: 1000 árboles..... 175

Matriz de confusión para modelo de máquinas de soporte vectorial con extracción de características por medio de modelo es_core_news_md 177

Matriz de confusión para aplicación implementada..... 177

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

ÍNDICE DE FIGURAS

Figura 1. Ejemplo de diagrama box-and-whisker. 20
Figura 2. CBOW y skipgram 42
Figura 3. Gráfica de función logística 50
Figura 4. Gráfica de hyper-plano que separa dos categorías. 52
Figura 5. Gráfica de hyper-plano para diferentes funciones de núcleo. 52
Figura 6. Explicación de Matriz de Confusión. 56
Figura 7. Diagrama de flujo de procedimiento general. 61
Figura 8. Diagrama de flujo sobre la exploración, análisis, limpieza y muestreo de conjunto de datos inicial. 62
Figura 9. Distribución de cantidad de caracteres en título de acuerdo a tipo. Exploración inicial 67
Figura 10. Distribución de cantidad de caracteres en título de acuerdo a tipo después de filtrado por longitud de caracteres y filtrado de Paid Notice. 69
Figura 11. Los 5 dominios con mayor n° de noticias de tipo ‘reliable’. 70
Figura 12. Los 5 dominios con mayor n° de noticias de tipo ‘fake’ 70
Figura 13. Los 5 dominios con mayor n° de noticias de tipo ‘reliable’ después de balanceo. 71
Figura 14. Los 5 dominios con mayor n° de noticias de tipo ‘fake’ despúes de balanceo 71
Figura 15. Total de caracteres por número de muestras para tipos ‘fake’ y ‘reliable’. 72
Figura 16. Total de caracteres por número de muestras para tipos ‘fake’(Rojo) y ‘reliable’(Azul) después de filtrado. 73
Figura 17. Exploración inicial de 1-grama, 2-grama y 3-grama para fuentes confiables. 77
Figura 18. Exploración inicial de 1-grama, 2-grama y 3-grama para fuentes ‘fake’ 78
Figura 19. Distribución de total de palabras por número de muestras después de filtrado por patrón y sin duplicados. 80

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Figura 20. Diagrama de caja de distribución de total de palabras por número de muestras después de filtrado por patrón y sin duplicados..... 81

Figura 21. Distribución de total de palabras por número de muestras después de filtrado por patrón, sin duplicados, menos a 3.5% de total de muestras descartadas..... 82

Figura 22.1-grama, 2-grama y 3-grama para fuentes 'reliable' después de filtrado.. 84

Figura 23. 1-grama, 2-grama y 3-grama para fuentes 'fake' después de filtrado. 85

Figura 24. Distribución de total de caracteres por número de muestras después de muestreo. 87

Figura 25. Distribución de total de palabras por número de muestras después de muestreo .. 87

Figura 26. Diagrama de flujo de etapa de traducción automática..... 88

Figura 27. Diagrama de dispersión de métricas BLEU y METEOR..... 93

Figura 28. Diagrama de dispersión con preprocesamiento.. 94

Figura 29. Diagrama de flujo de extracción de características.. 96

Figura 30. Diagrama de flujo de entrenamiento, diferenciación y elección de modelos. 102

Figura 31. Polaridad de conjunto de datos. 105

Figura 32. Diagrama de caja de distribución de características..... 105

Figura 33. Diagrama de caja de distribución de número de palabras por tipo. 106

Figura 34. Diagrama de caja de distribución de tamaño de palabras promedio. 106

Figura 35. Diagrama de caja de distribución de diversidad de etiquetas POS. 107

Figura 36. Diagrama de flujo de etapa de despliegue de modelo. 116

Figura 37. Proceso de consulta de predicción a sistema de Aprendizaje Automático. 124

Figura 38. Interfaz gráfica de usuario de detector de noticias falsas. 131

Figura 39. Ejemplo de noticia falsa. 132

Figura 40. Ejemplo de noticia confiable. 133

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ÍNDICE DE TABLAS

Tabla 1. Ejemplo de resultado de proceso de Vectorización.. 23

Tabla 2. Ejemplo de Word Embeddings. 40

Tabla 3: Etiquetas de tipos de noticias 64

Tabla 4. Ejemplo de muestra de dominios con últimos 30 caracteres de título. 75

Tabla 5. Distribución de número de palabras de título. 82

Tabla 6. Distribución de métricas BLEU y METEOR. 95

Tabla 7. Características complejas y estilísticas. 98

Tabla 8. Valores numéricos representativos de características extraídas. Parte 1..... 103

Tabla 9. Valores numéricos representativos de características extraídas. Parte 2..... 103

Tabla 10. Valores numéricos representativos de características extraídas. Parte 3..... 104

Tabla 11. Dimensiones de vectores de palabras de acuerdo a valores de pos_and_ner y sentiment..... 108

Tabla 12. Evaluación de modelo LinearSVC: Word Embeddings.. 110

Tabla 13. Evaluación de modelo LinearSVC: Word Embeddings y etiquetas POS y NER...110

Tabla 14. Evaluación de modelo LinearSVC: Word Embeddings y polaridad. 110

Tabla 15.Evaluación de modelo LinearSVC: Word Embeddings, polaridad y etiquetas POS y NER. 110

Tabla 16. Evaluación de modelo Regresión Logística: Word Embeddings. 110

Tabla 17. Evaluación de modelo Regresión Logística: Word Embeddings y etiquetas POS y NER. 111

Tabla 18. Evaluación de modelo Regresión Logística: Word Embeddings y polaridad. 111

Tabla 19. Evaluación de modelo Regresión Logística: Word Embeddings, polaridad y etiquetas POS y NER. 111

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Tabla 20. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings. 111

Tabla 21. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings y etiquetas POS y NER.. 111

Tabla 22. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings y polaridad. 111

Tabla 23. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings, polaridad y etiquetas POS y NER. 112

Tabla 24. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings. 112

Tabla 25. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings y etiquetas POS y NER. 112

Tabla 26. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings y polaridad. 112

Tabla 27. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings, polaridad y etiquetas POS y NER. 112

Tabla 28. Sensibilidad para detección de noticias falsas. 113

Tabla 29. Exactitud de modelos. 113

Tabla 30. Comparación de métricas de modelos es_core_news_lg y es_core_news_md. 118

Tabla 31. Evaluación de modelo LinearSVC con modelo es_core_news_md: Word Embeddings, polaridad y etiquetas POS y NER. 118

Tabla 32. Comparación de arquitecturas de sistemas de Aprendizaje Automático..... 122

Tabla 33. Descripción de scripts desarrollados para despliegue..... 130

Tabla 34. Evaluación de predicciones de implementación..... 134

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

CAPÍTULO I

1. Planteamiento Teórico

Se explica la problemática que motivó la realización del presente trabajo y la línea general a seguir en su desarrollo. Además, se presentan los objetivos que dicho proyecto busca cumplir.

1.1.Título de Investigación

Implementación de clasificador de noticias en idioma español para la identificación de Fake News mediante el análisis, Traducción Automática y validación de un conjunto de datos en inglés, y el uso de técnicas de Aprendizaje Máquina y Procesamiento de Lenguaje Natural.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

1.2.Descripción del problema.

El presente proyecto de investigación busca hacer frente al fenómeno conocido como Fake News, o noticias falsas por su traducción del inglés, mediante la implementación de un clasificador de noticias basado en algoritmos de aprendizaje automático supervisado, *Supervised Machine Learning* o *ML*, y técnicas propias de procesamiento de lenguaje natural, *Natural Language Processing* o *NLP*, para poder determinar la clasificación de una noticia como falsa o confiable.

Es importante mencionar que este trabajo de investigación busca implementar dicho clasificador de noticias teniendo como objetivo noticias en idioma español. Debido al hecho que el número de conjuntos de datos de noticias en español utilizados para el entrenamiento de algoritmos de aprendizaje automático supervisado es casi nulo, el presente proyecto plantea también la producción de un conjunto de datos en español a partir de la traducción automática de otro conjunto de datos en idioma inglés. Dicha traducción será evaluada mediante el uso de métricas orientadas a la traducción máquina para asegurar su calidad.

1.3.Justificación de la Investigación

Se han realizado estudios relacionados con modelos de aprendizaje máquina supervisado para la clasificación de noticias. Algunos de estos implementan el modelo de clasificación en base a características sintácticas o de forma, como por ejemplo, el número de etiquetas de parte del discurso, como el número de verbos (Horne y Adali 2017). Otros implementan características semánticas, como el uso de vectores de palabras construidos a partir de la frecuencia de palabras que involucran uso de un vectorizador por conteo o por *TF-*

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

IDF, demostrando además que un análisis semántico puede lograr mejores resultados para el modelo de clasificación (Al Asaad y Erascu 2018; Bhutani et al. 2019). Sin embargo, es necesario mencionar que la gran mayoría de dichos estudios han implementado modelos de clasificación para un conjunto de datos, también conocido como corpus, originalmente en inglés, por lo que los estudios relacionados en idioma español son mínimos. El trabajo de investigación *Detection of Fake News in a New Corpus for the Spanish Language* (Posadas-Durán et al. 2019) es justamente uno de los muy escasos trabajos relacionados que tienen como objetivo el español. Dicho trabajo está orientado a la construcción de un modelo de clasificación en base a características sintácticas, al hacer uso de las etiquetas de parte del lenguaje, y características semánticas, haciendo uso de vectores de palabras construidos a partir de la frecuencia de palabras; sin embargo, es quizás la mayor contribución de este trabajo el presentar el único corpus, para conocimiento del autor, en español (jpposadas [2018] 2020). La desventaja de dicho corpus es que contiene únicamente 971 noticias preclasificadas, lo que es una cantidad muy menor a la de otros corpus en otros idiomas, como en los estudios antes mencionados, que incluso llegan a superar las 10 000 muestras. Dicha cantidad de muestras puede impactar significativamente en el entrenamiento de un modelo de clasificación basado en Aprendizaje Máquina.

Las *Fake News* o noticias falsas es un problema que aqueja a nuestra sociedad en general y no sólo al sector anglosajón. Esta es una de las motivaciones del autor para desarrollar el presente proyecto que pretende ser implementado teniendo como principal objetivo la calificación de noticias originalmente en idioma español. Para esto, se creará un nuevo corpus en español a partir del uso de un traductor automático que traducirá un corpus del inglés al español. Dicha traducción será evaluada y filtrada por medio de métricas orientadas a la traducción automática para poder obtener un corpus en español con traducciones cuya calidad

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

sea de beneficio del proyecto. Este corpus nos permitirá tener un mayor número de muestras que el corpus en español mencionado anteriormente (jpposadas [2018] 2020). Más tarde, se entrenará el modelo de clasificación extrayendo características tanto semánticas como sintácticas. Lo característico de este proyecto es que las características semánticas incluidas se basarán no en la frecuencia de palabras, sino más bien en técnicas más sofisticadas como el uso de modelos de redes neuronales que codifican la similitud semántica entre palabras, como es el uso de fastText (fastText s. f.). En cuanto a características sintácticas, se analizará no solo la ocurrencia de etiquetas de parte del discurso, sino también el reconocimiento de entidades. Adicionalmente, se propone incluir el análisis el análisis de sentimientos o polaridad. Con polaridad nos referimos a que tan negativo, positivo o neutral resulta el texto de cada noticia. Otro aspecto resaltante de este proyecto es que, a diferencia de otros modelos, se realizará el análisis únicamente del título de la noticia y no del contenido. Por último, se implementará una interfaz de usuario para la presentación y prueba de dicho modelo.

1.4.Línea general del desarrollo del proyecto.

El cambio de idioma representa sin duda una dificultad adicional al proyecto debido a que, para conocimiento del autor, no existen fuentes de datos preclasificadas con un corpus, o conjunto de datos usado para el entrenamiento de un modelo en ML, en idioma español. Por esta razón, para entrenar un modelo predictivo basado en aprendizaje máquina supervisado y NLP, se pueden tomar dos aproximaciones:

1. Entrenar y evaluar un modelo predictivo con un corpus originalmente en inglés, es decir, usar para el entrenamiento de dicho modelo una fuente de datos originalmente en inglés. Luego, traducir los datos que se pretenden predecir, la noticia que se

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

quiere clasificar, al idioma inglés y usar el modelo predictivo diseñado inicialmente para clasificar a esta noticia traducida.

2. Traducir un corpus originalmente en inglés al español, utilizar este corpus traducido para entrenar un modelo predictivo que tenga como objetivo noticias en español, y utilizar dicho modelo ya entrenado para realizar nuevas predicciones de noticias en idioma español.

En este proyecto se tomará como aproximación a seguir el segundo punto debido a que nos permitirá, previamente al entrenamiento del modelo, una evaluación de la traducción y un consecuente filtrado producto de esta evaluación, con lo que se podrá obtener un modelo optimizado para realizar predicciones en idioma español.

Previo a la traducción del corpus, se realizará una exploración de los datos disponibles. Este es un procedimiento estándar en la analítica de datos, y será útil para realizar un primer filtrado de información en donde se buscará analizar tendencias y/o posibles sesgos del conjunto de datos, y realizar un pre filtrado de datos que asegure la integridad del conjunto y evite anomalías que podrían influenciar en el entrenamiento del modelo.

En cuanto al proceso de traducción y filtrado, para la evaluación de la traducción, considerando también que se utilizarán modelos de traducción automática, se utilizarán técnicas como el *backtranslation* y métricas propuestas, como son la métrica BLEU (Papineni et al. 2002) y la métrica METEOR (Banerjee y Lavie 2005).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

El corpus ya traducido y filtrado no puede ser directamente utilizado por algoritmos de aprendizaje automático debido a que dicho corpus contiene datos textuales no interpretables por una computadora. Por esta razón, se hará uso de técnicas de obtención de información para obtener datos numéricos que puedan ser utilizados por dichos algoritmos. Se hará uso de herramientas de vectorización que nos permitirán codificar las características semánticas y similitud en significado de las palabras. Se propone adicionalmente utilizar técnicas de análisis de sentimientos o polaridad para poder brindar a nuestro modelo una característica adicional con la cual podrá ser entrenado. Por otra parte, se extraerán también características sintácticas, particularmente etiquetas de parte del discurso y nombre de entidades. Se evaluará además la eficacia del uso de dichas técnicas de análisis de semántico, sintáctico y de sentimientos, comparándolas con modelos en los que no se les emplee.

Finalmente, luego de tener los datos numéricos deseados, se procederá a entrenar el modelo de aprendizaje automático. Para esto, se emplearán diversos algoritmos de aprendizaje automático, como son los bosques aleatorios o clasificadores de soporte vectorial, y se comparará su eficacia en base a métricas como: precisión, exactitud, cobertura y f1-score. Se hará uso también de la matriz de confusión para observar y tener una mejor idea del desempeño de estos modelos.

Por último, implementaremos dicho modelo en un sistema web para ser utilizado por el usuario. Entre algunas de las tecnologías propuestas para esta sección encontramos Angular (Jain, Mangal, y Mehta 2015), y la plataforma de Google Cloud Platform (Google s. f.).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

1.5. Hipótesis

Es posible implementar un clasificador que permita identificar noticias falsas mediante la traducción de un corpus en inglés y el uso de algoritmos de Machine Learning.

1.6.Objetivos

1.6.1. Objetivo General

Implementar un clasificador basado en aprendizaje máquina para el reconocimiento del título de noticias originalmente en idioma español como falsas.

1.6.2. Objetivos Específicos

- Realizar una inicial exploración y limpieza de datos para asegurar la integridad del conjunto de datos que se utilizará para entrenar el modelo de aprendizaje automático.

- Traducir, por medio de aplicación de Aprendizaje automático, y evaluar un corpus de noticias originalmente en inglés a español de acuerdo con métricas.

- Transformar datos textuales a conjunto de datos numéricos que puedan ser utilizados para el entrenamiento de algoritmos de aprendizaje automático realizando un análisis sintáctico, semántico y de sentimientos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

- Evaluar el desempeño o mejora de los modelos de aprendizaje automático dependiendo de la presencia o ausencia de características sintácticas y de sentimientos.

- Desplegar e implementar modelo con mejor desempeño mediante un sistema web para el uso del usuario.

1.7.Alcance

- El presente proyecto se limita a analizar los títulos de las noticias, pero no el contenido.

- El presente proyecto no pretende implementar un modelo perfecto, es decir, el modelo implementado no predecirá correctamente todas las noticias disponibles. El desempeño del modelo implementado será evaluado en base a las métricas mencionadas anteriormente.

- El presente proyecto no busca implementar un método de minería de datos para obtener un corpus de noticias. Esto debido a la excesiva carga de recursos computacionales y recursos humanos que la preclasificación de un gran número de noticias conllevaría.

- El corpus o conjunto de datos inicial utilizados para esta investigación podría también tener errores en su preclasificación. Sin embargo, dichos errores no

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

deberían de representar un gran obstáculo para este proyecto debido a su ínfima cantidad.

- El presente proyecto no pretende ahondar ni investigar los fundamentos matemáticos de cada uno de los algoritmos y/o librerías utilizadas, sino más bien, evaluar su eficacia con respecto al contexto del proyecto. Se incluirá sin embargo un fundamento teórico que permita tener un conocimiento sobre ellos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

CAPÍTULO II

2. Marco teórico

El presente marco teórico busca definir conceptos importantes para la comprensión del planteamiento teórico y el desarrollo del proyecto. La primera sección se encarga de describir librerías utilizadas para el desarrollo del proyecto. La segunda sección se enfoca en describir conceptos teóricos relacionados con el planteamiento inicial y el desarrollo, y es de gran importancia para la comprensión de la implementación del proyecto.

2.1.Herramientas o Librerías

Se describen las herramientas o librerías usadas en el desarrollo del proyecto.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

2.1.1. Python.

Python (Van Rossum y Drake 2009) es un lenguaje de programación interpretado, interactivo y orientado a objetos (Python 3.8.5 documentation s. f.). Es un lenguaje de código abierto cuyos derechos de propiedad intelectual son mantenidos por la *Python Software Foundation* (Python Software Foundation s. f.). Este lenguaje fue el principal lenguaje elegido para el desarrollo de este proyecto debido a su gran popularidad de los últimos años y a su gran cantidad de librerías y herramientas utilizadas para la analítica de datos y el aprendizaje máquina.

2.1.2. Jupyter Notebook.

Jupyter Notebook (Project Jupyter [2015] 2020) es una aplicación HTML de código abierto para la computación interactiva que permite editar, visualizar y ejecutar código de diferentes lenguajes de programación. *Jupyter Notebook*, referido en el resto del proyecto como *Jupyter*, será utilizado para la exposición y visualización del código y resultados del presente proyecto.

2.1.3. Matplotlib

Matplotlib (Hunter 2007) es una librería utilizada para la creación de visualizaciones estáticas, animadas e interactivas en Python (Van Rossum y Drake 2009). En este proyecto se utiliza *Matplotlib* por medio de otras librerías para la visualización de estadísticas sobre los datos tratados.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

2.1.4. Seaborn

Seaborn (Waskom et al. 2017) es una librería utilizada para la visualización de datos basada en *matplotlib*. Provee de una interfaz de alto nivel para la presentación atractiva e informativa de gráficos estadísticos. Seaborn es utilizado en este proyecto para la visualización de estadísticas relacionadas con los datos tratados.

2.1.5. Pandas

Es una librería de código abierto para Python (Van Rossum y Drake 2009) que provee herramientas de alto desempeño, de estructuras de datos fáciles de usar y de análisis de datos (McKinney y others 2010). Esta librería proporciona herramientas muy útiles para la manipulación y visualización de datos y es utilizada en este proyecto para este mismo fin. Incorpora también herramientas útiles para la visualización y grafico de datos basadas en *matplotlib*.

2.1.6. Natural Language Toolkit (NLTK)

Natural Language Toolkikt (Bird, Klein, y Loper 2009) es una librería para Python (Van Rossum y Drake 2009) que provee de módulos e interfaces útiles para el procesamiento de lenguaje natural. Se utilizó para la obtención de métricas METEOR y BLEU en la evaluación de la traducción por medio del método *backtranslation* y para la remoción de palabras vacías, que son palabras que no añaden ningún significado al análisis, en el análisis previo al muestreo para determinar los n-gramas más comunes.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

2.1.7. Polyglot

Polyglot (ALRFOU [2014] 2020) es una librería para Python (Van Rossum y Drake 2009) que provee de módulos e interfaces útiles para el procesamiento de lenguaje natural. En este proyecto, se hizo uso del módulo de análisis de sentimientos de Polyglot (ALRFOU [2014] 2020) que es una implementación directa de un léxico de sentimientos en español (Chen y Skiena 2014). Dicho módulo de análisis de sentimientos nos permitió extraer la polaridad de las muestras en la extracción de características previa a el entrenamiento del modelo de aprendizaje automático.

2.1.8. SpaCy

SpaCy (Honnibal y Montani 2017) es una librería escrita en Cython (Behnel et al. 2011) y Python (Van Rossum y Drake 2009) utilizada para aplicaciones de procesamiento de lenguaje natural. A diferencia de Natural Language Toolkit (Bird et al. 2009), que tiene una orientación hacia la educación e investigación, SpaCy (Honnibal y Montani 2017) tiene un uso más enfocado a la producción. Por esta misma razón es que Spacy (Honnibal y Montani 2017) tiene mayor velocidad en los procesos de tokenización y etiquetado (Explosion [2015] 2019) que otras librerías. SpaCy (Honnibal y Montani 2017) fue utilizado en la etapa de extracción de características, específicamente para la extracción de *Word Embeddings* y etiquetado de etiquetas de parte del discurso y reconocimiento de entidades. Fue utilizado también como parte del modelo de aprendizaje automático en el sistema web.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

2.1.9. Flask

Flask (Grinberg 2014) es un entorno de trabajo escrito en Python (Van Rossum y Drake 2009) para el desarrollo de aplicaciones web. En este proyecto fue utilizado para el desarrollo de una aplicación encargada de comunicar la interfaz del usuario con el modelo de aprendizaje automático mediante la administración de solicitudes.

2.1.10. Scikit-learn

Scikit-learn (Pedregosa et al. 2018) es una librería de código libre para el lenguaje de programación Python (Van Rossum y Drake 2009). Incluye varias herramientas utilizadas para aplicaciones de aprendizaje máquina, además de distintos algoritmos de clasificación, regresión y agrupamiento. En este proyecto fue utilizado para obtener las palabras más comunes presentes en el conjunto de datos no traducido, realizar el muestreo del conjunto de datos previo a la traducción, construcción de tuberías y el entrenamiento y predicción de modelos de clasificación.

2.1.11. Joblib

Joblib (The joblib developers [2010] 2020) es una librería que fue utilizada para la serialización y deserealización del modelo de aprendizaje automático. La serialización y deserealización fue utilizada para asegurar la persistencia de los modelos de clasificación, con el objetivo de no necesitar un entrenamiento constante de los modelos de aprendizaje automático.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

2.1.12. Angular

Angular (Jain et al. 2015) es un entorno de trabajo basado en el lenguaje de programación Typescript (Bierman, Abadi, y Torgersen 2014) para el desarrollo de aplicaciones web. Este entorno de trabajo fue utilizado para desarrollar la interfaz de usuario mediante la cual el usuario realizará sus solicitudes de predicción.

2.2.Marco Conceptual

Se describen los conceptos importantes para la comprensión del planteamiento del problema y el desarrollo del proyecto. A partir de estos conceptos, se tomaron decisiones relevantes para la implementación y desarrollo.

2.2.1. Conceptos Generales.

Se incluyen conceptos generales útiles principalmente para la comprensión del planteamiento teórico.

2.2.1.1.Inteligencia Artificial.

Inteligencia artificial es un término general utilizado para referirse a un tipo de tecnología de computadora que se ocupa de hacer al hardware o software exhibir un comportamiento aparentemente similar al comportamiento inteligente humano. La abreviación IA es también utilizada (Collins English Dictionary s. f.)

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

2.2.1.2.Machine Learning o Aprendizaje Máquina.

Un diseño de ingeniería convencional sigue una serie de pasos ya establecidos: Primero, la obtención de conocimiento que describe idealmente las reglas que rigen un determinado fenómeno, segundo, la producción de un modelo matemático que captura el funcionamiento del fenómeno de estudio, tercero, la producción de un algoritmo y modelo optimizado al fenómeno de estudio bajo la suposición de que dicho modelo es una ideal representación de la realidad.

El uso de un diseño de ingeniería convencional no es adecuado para situaciones reales o no ideales, esto debido a que situaciones reales involucran infinidad de variaciones que serían muy complicadas de ser modeladas mediante un modelo ideal.

Machine Learning, Aprendizaje Máquina o Aprendizaje Automático substituye la aproximación tomada por un diseño de ingeniería convencional en el sentido en que substituye el paso de obtención de conocimiento que describe idealmente las reglas que rigen un determinado fenómeno, por la recolección de una gran cantidad de muestras que se ven en modelos reales no ideales (Simeone 2018:1). Esta cantidad de datos incorpora una flexibilidad a un diseño de ingeniería en Machine Learning (referido en el resto de investigación como ML) al permitirle ser entrenado, o aprender, en base a dichas muestras. De esta manera, un algoritmo en ML puede producir un modelo que modele o simule un fenómeno real de mejor manera que un diseño convencional.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Existen tres tipos de aprendizaje máquina (Simeone 2018:2):

- Aprendizaje supervisado: En este tipo de aprendizaje el conjunto de datos utilizado para el entrenamiento consiste en datos etiquetados, que son datos de entrada con sus respectivos datos de salida. Por ejemplo, en el caso del presente proyecto, un conjunto de datos que contiene los títulos de noticias, identificado como el dato de entrada, y sus etiquetas que determinan si se trata de una noticia falsa o confiable.

- Aprendizaje no supervisado: El conjunto de datos utilizado para el entrenamiento consiste en datos no etiquetados, es decir, únicamente datos de entrada. Este tipo de aprendizaje busca descubrir propiedades del sistema.

- Aprendizaje reforzado: En cierto sentido, se encuentra entre el aprendizaje supervisado y el no supervisado. Un algoritmo de aprendizaje reforzado recibe una alimentación del ambiente. Dicha alimentación indica en qué medida la salida satisface los objetivos generales del aprendizaje. Por ejemplo, un videojuego, en el que el personaje principal obtiene un mayor puntaje mientras más monedas recolecte.

2.2.1.3.NLP o Natural Language Processing

El Procesamiento de Lenguaje Natural (referido en el resto del proyecto como NLP por sus siglas en inglés) es un campo de estudio del aprendizaje automático que se enfoca en el procesamiento del lenguaje humano. Busca aprender y procesar el lenguaje humano para permitir a sistemas computacionales realizar tareas con él (Jain, Kulkarni, y Shah 2018:1).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Algunos de los campos de estudio en el procesamiento del lenguaje natural son:

- Reconocimiento de palabras.
- Neural Machine Translation o Traducción Automática Neuronal, que fue empleado para realizar la actividad de traducción de este proyecto
- Reconocimiento de entidades.
- Reconocimiento y análisis de semántica y sentimientos.

2.2.1.4.Machine Translation o Traducción Automática

Se denomina traducción automática a la producción de texto en un lenguaje natural a partir de otro por medio de procedimientos computarizados (Collins English Dictionary s. f.).

Como resultado del uso de redes neuronales, varios campos relacionados con el procesamiento de lenguaje natural han tenido un sorprendente desarrollo en la última década. La traducción automática no es una excepción ya que ha tenido un cambio de paradigma, en donde modelos basados en conteo, como por ejemplo modelos estadísticos basados en traducción de frases (Koehn, Och, y Marcu 2003), han sido reemplazados por modelos basados en redes neuronales, dando lugar a lo que se conoce como Traducción Automática Neuronal, o NMT por sus siglas en inglés (Stahlberg 2019). Una de las implementaciones de traducción

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

automática neuronal es propuesto por Google (Wu et al. 2016), y es utilizado como parte del presente trabajo.

2.2.1.5.Fake News

Fake News, o noticias falsas por su traducción al español, son noticias que no son verídicas. Se caracterizan por intentar propagar noticias no verdaderas con la intención de engañar, desestimar la verdad, y causar desorden (Lanius y Jaster 2018).

2.2.2. Exploración, análisis, limpieza y muestreo de conjunto de datos

Se procede a exponer conceptos relevantes a la sección Exploración, análisis, limpieza y muestreo de conjunto de datos del del desarrollo del proyecto.

2.2.2.1.Boxplot

Boxplot, o *box-and-whisker plot*, o diagrama de caja (Seaborn s. f.) es un diagrama que permite visualizar la distribución cuantitativa de datos. Los extremos de la caja o rectángulo de este diagrama indican los cuartiles correspondientes al percentil 25% y al percentil 75% de la distribución, mientras que la línea que se encuentra en el medio de dicho rectángulo indica la media de la distribución. La distancia entre el cuartil Q1, o percentil 25, y Q3, o percentil 75, representan la denominada distancia inter-cuartil (IQR). Los denominados *Whisker* son representados por las dos líneas de los dos extremos del diagrama. El whisker inferior representa la resta entre el Q1 y la 1.5 parte del IQR; de la misma manera, el whisker superior representa la suma entre el Q3 y la 1.5 parte del IQR. Además, los *outliers* o valores atípicos,

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

representan valores de la distribución que se encuentran fuera del diagrama, pero que son valores propios del conjunto de datos.

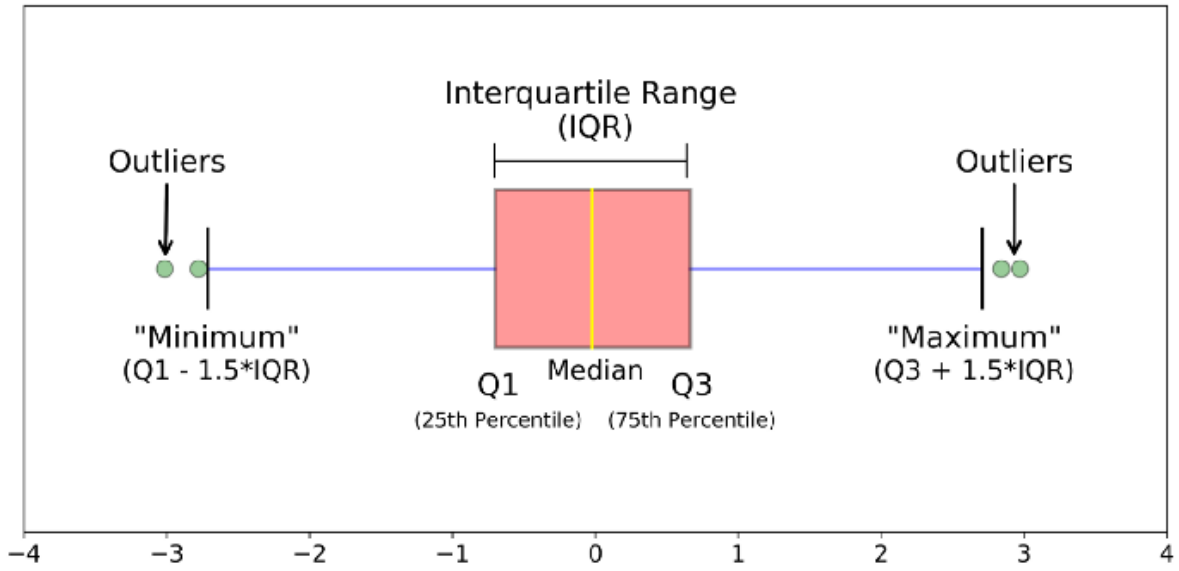


Figura 1. Ejemplo de diagrama box-and-whisker. Fuente: (Galarnyk 2020)

En el presente proyecto, se utilizó la librería Seaborn (Waskom et al. 2017) para realizar dichos diagramas.

2.2.2.2.Modelos de Lenguajes y n-gramas

Un modelo Estadístico de Lenguaje es una distribución probabilística sobre secuencias de palabras en un mismo vocabulario (Manning, Raghavan, y Schütze 2008:238). Los modelos de n-gramas son un modelo de lenguaje que nos permite calcular la probabilidad con la que un número n de palabras pueden aparecer juntas en un vocabulario.

El modelo de unigrama es el modelo de n-grama más simple en donde n=1, es decir, el modelo de n-grama con una sola palabra. En este modelo, el contexto o las palabras que se

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

encuentren junto a la palabra a analizar carecen de importancia, por lo que se estima únicamente la probabilidad de que dicha palabra aparezca en el vocabulario. A diferencia del modelo unigrama, los modelos de n-gramas, en donde n>1, sí consideran el contexto, particularmente, las n palabras que aparecen juntas en un vocabulario.

Los modelos de n-gramas de este proyecto son producidos a través de la estrategia de bolsa de palabras para la extracción de características numéricas a partir de características no numéricas, como, en este caso, cadenas de caracteres. Dichas características numéricas son después utilizadas para el análisis de las palabras o conjuntos de palabras con mayor repetición en el conjunto de datos inicial.

2.2.2.3.Extracción de características del texto y vectorización

El análisis de texto es una aplicación importante para los algoritmos de Machine Learning. Sin embargo, el uso de cadenas de caracteres o símbolos no permite directamente un análisis.

La vectorización es la respuesta a este problema. Se denomina vectorización al proceso mediante el cual se mapean las palabras de una colección de documentos de texto a características numéricas representadas mediante vectores numéricos. El análisis del texto puede ser realizado más tarde mediante el análisis de dichas características numéricas.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

2.2.2.4.Modelo de Bolsa de Palabras o Bolsa de n-gramas

El modelo de Bolsa de Palabras o Bolsa de n-gramas es una estrategia de vectorización mediante la cual se asigna un valor numérico o peso a cada término, o a cada grupo de n-términos, que describe su ocurrencia dentro de un vocabulario (Manning et al. 2008:117). Es a partir de esta estrategia que se produjo los modelos de n-gramas del presente trabajo.

Esta estrategia se puede dividir en 3 puntos: preprocesamiento, tokenización, y la asignación de pesos a cada token.

- La primera etapa es denominada preprocesamiento, y busca realizar acciones iniciales para adecuar las cadenas de caracteres para sus siguientes etapas. Una de las acciones iniciales comunes es la eliminación de caracteres de puntuación.

- Mediante la etapa de tokenización, las cadenas de caracteres de cada muestra o documento son divididas en palabras o términos de acuerdo con su separación por espacios o signos de puntuación. A cada uno de estos términos se les conoce como tokens. A cada uno de estos tokens no repetidos de cada documento se les asignará un identificador único dentro del vocabulario creado. Se denomina vocabulario al conjunto de todos los tokens utilizados para implementar la estrategia de Bolsa de palabras y creados a partir de todo el corpus.

- La tercera etapa consiste en asignarle un valor numérico, o peso, a cada token dentro del vocabulario. Este valor numérico puede ser dado de acuerdo con diferentes métodos. En el presente proyecto se usaron la frecuencia de términos (vectorización

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

por conteo) para asignar dicho peso. A cada uno de estos tokens junto con su valor numérico asignado se le denomina vector.

Podemos ilustrar el resultado del proceso de vectorización por conteo de acuerdo con lo siguiente: después de terminar el proceso de vectorización, se obtendrá una matriz de 2 dimensiones (Tabla 1. Ejemplo de resultado de proceso de Vectorización. Fuente: Elaboración Propia.). La primera dimensión hace referencia al texto de cada muestra (al documento) que fue vectorizado. La segunda dimensión hace referencia a cada uno de los tokens creados. Así, cada uno de los tokens creados estará descrito por un identificador (Id1 por ejemplo), y cada documento tendrá un n número de características dependiendo del n número de tokens que conforman el vocabulario.

Tabla 1. Ejemplo de resultado de proceso de Vectorización. Fuente: Elaboración Propia.

	Texto	Id1(El)	Id2(perro)	Id3(niño)	Id4(corre)	Id5(come)	Id6(salta)
Doc1	El niño	1	0	1	0	0	0
Doc2	El perro corre	1	1	0	1	0	0
Doc3	El perro salta	1	1	0	0	0	1
Doc4	El niño come	1	0	1	0	1	0
Doc5	El El	3	0	0	0	0	0

Se puede observar que el modelo de bolsa de palabras produce matrices con un mayor número de 0 mientras mayor sea el vocabulario. Tener una matriz compuesta por un gran número de 0 y un mínimo número de valores significativos provoca un desperdicio innecesario de recursos de memoria. Por esta razón, las implementaciones del modelo de Bolsa de Palabras utilizan típicamente las denominadas matrices dispersas(Techopedia s. f.), que permiten almacenar únicamente los valores significativos distintos de 0 mediante métodos de compresión, lo que permite un uso más eficiente del almacenamiento.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

2.2.2.5.Vectorización por frecuencia de términos y TF-IDF

En el presente proyecto, en la implementación del Modelo de Bolsa de palabras, al momento de asignar un valor a cada token se utilizó el método de frecuencia de términos (esto, para el análisis y exploración de datos previos). Se presenta también en esta sección como referencia el método TF-IDF, que es un método comúnmente utilizado para el entrenamiento de modelos de aprendizaje automático.

2.2.2.5.1. Por Frecuencia de Términos

El vectorizador por frecuencia de términos asigna el peso a cada token de acuerdo con el número de ocurrencias con las que aparece cada token en cada documento (así como se muestra en Tabla 1. Ejemplo de resultado de proceso de Vectorización. Fuente: Elaboración Propia.).

2.2.2.5.2. Por TF-IDF

TF-IDF hace referencia a frecuencia de términos-frecuencia inversa de términos del documento (o, por sus siglas en inglés, *term frequency-inverse document frecuency*). El peso por TF-IDF es una medida estadística utilizada para determinar la importancia de una palabra o token en la colección de documentos o corpus (Tf-idf s. f.). La importancia del token aumenta proporcionalmente al número de veces con el que dicho token aparece en el documento, pero disminuye de acuerdo con la mayor frecuencia con que aparece en el corpus. La diferencia entre la vectorización por conteo y por TF-IDF es que la vectorización por TF-IDF penaliza las

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

palabras que tienen una mayor ocurrencia en el corpus, de modo que palabras como los artículos “el” o “la” tendrán un menor peso.

El TF-IDF se compone de dos partes principalmente:

- TF: Hace referencia a la frecuencia de término normalizada. Mide la frecuencia de un determinado término o token dentro de cada documento. Debido a que los documentos tienen una longitud diferente, es posible que un determinado término aparezca muchas más veces en documentos largos que en documentos cortos. Por esta razón, TF se calcula dividiendo la frecuencia del término entre la longitud del documento (refiriéndonos al número total de términos en el documento). Así, se calcula:

$$TF(t) = \frac{n° \; total \; en \; la \; que \; el \; termino \; t \; aparece \; en \; documento}{n° \; total \; de \; terminos \; en \; el \; documento} \qquad (1)$$

- IDF: hace referencia a la frecuencia inversa de documento (o Inverse Document Frequency). Mientras que el TF considera a todos los términos con igual importancia, el IDF busca reducir la importancia para términos comunes, como por “es”, “eso”. Esto se realiza mediante la siguiente fórmula

$$IDF(t) = \log_e \frac{n° \; total \; de \; documentos}{n° \; de \; documentos \; con \; el \; termino \; t} \qquad (2)$$

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- TF-IDF: Es el uso conjunto del TF y el IDF. La fórmula es la siguiente

$$TF - IDF(t) = TF(t) * IDF(t) \quad (3)$$

2.2.3. Ejecución y validación de traducción automática.

Se describe la metodología de *backtranslation* que será utilizada para realizar la validación de la traducción. Más tarde, se describe el servicio utilizado para realizar la traducción y re-traducción del conjunto de datos, así como los modelos, versión y formato de el mismo. Por último, se describen dos métricas fundamentales utilizadas en dicha validación.

2.2.3.1.Backtranslation y validación de traducción

Backtranslation (Bojar y Tamchyna 2011) es una técnica utilizada en el entrenamiento de modelos de traducción automática para incrementar la cantidad de muestras de un único lenguaje utilizada en dicho entrenamiento y, por lo tanto, mejorar el modelo de traducción. El procedimiento es el siguiente (Bojar y Tamchyna 2011:331):

1. Se dispone de un conjunto de datos que cuenta con los datos en el idioma original y los datos traducidos en el idioma objetivo.

2. Se realiza el entrenamiento de un modelo de traducción automática para traducir los datos desde el idioma objetivo hasta el idioma original.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3. Se traducen los datos desde el idioma objetivo hasta el idioma original utilizando el modelo entrenado produciendo un nuevo conjunto de datos en el idioma original. A dicha traducción la denominaremos re-traducción.

4. Se hace uso de una métrica de evaluación para comparar los registros del conjunto de datos original con los registros correspondientes del conjunto de datos de re-traducción y descartar aquellas traducciones consideradas de baja calidad. Bojar y Tamchyna (2011) propusieron utilizar como métrica a la métrica BLEU (Papineni et al. 2002) y descartar los datos fuera de un intervalo de confidencia de 95%; sin embargo, investigaciones posteriores (Edunov et al. 2020) demostraron la ineficacia del uso de dicha métrica en *backtranslation* y proponen el uso de otras métricas, entre las que se encuentra la métrica METEOR (Banerjee y Lavie 2005), para la evaluación de las traducciones.

5. Por último, se vuelve a entrenar un modelo de traducción máquina mediante el uso del conjunto de datos resultantes compuesto por los datos iniciales y re-traducidos filtrados.

En el presente trabajo, no utilizaremos la técnica del *backtranslation* para el entrenamiento de un modelo de traducción automática, sino más bien para evaluar los resultados de la traducción automática de un modelo ya existente y filtrarlos descartando el 5% de muestras de menor calidad. Así, el procedimiento que se realizará será el siguiente:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

1. Se traducirá al idioma español, mediante el uso de un modelo de traducción automática ya entrenado, el corpus en inglés resultado del proceso de exploración, análisis, limpieza y muestreo de conjunto de datos.

2. Se traducirá nuevamente al idioma inglés, mediante el uso de el mismo modelo de traducción automática utilizado en el paso anterior, el corpus en español resultado de la anterior traducción. Este proceso será denominado re-traducción.

3. Se evaluará la traducción por medio del uso de la métrica METEOR (Banerjee y Lavie 2005), comparando los dos conjuntos de datos en idioma inglés: el conjunto de datos original y el conjunto de datos retraducido. Se utilizó dicha métrica debido a que muestra una mejor correspondencia con las evaluaciones humanas que la métrica BLEU (Papineni et al. 2002).

4. Se filtrará el conjunto de datos, manteniendo el 95% de muestras con mayor valor de la métrica de evaluación.

El resultado de dicha traducción será utilizado para las posteriores etapas del proyecto.

2.2.3.2.Cloud Translation: elección de modelo, versión y formato

Cloud Translation (Google Cloud s. f.) es un servicio de traducción automática implementado por Google. En el presente trabajo, se hizo uso de dicho servicio mediante la Cloud Translation API (LLC s. f.) disponible para el lenguaje de programación Python (Van Rossum y Drake 2009) para realizar la traducción y re-traducción del conjunto de datos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

Cloud Translation tiene disponible el uso de tres modelos de traducción automática:

- Traducción Automática basada en Frases o PBMT: Se trata de un modelo de traducción estadístico.

- Traducción Automática Neural o NMT: Modelo de traducción basado en el uso de redes neuronales con traducciones de mejor calidad que el PBMT. El presente proyecto utilizó el modelo NMT debido a su mejor calidad y al hecho que no requiere de entrenamiento por parte del usuario. Es un modelo de traducción de propósito general.

- AutoML: Modelo de traducción específico basado en NMT que requiere de entrenamiento por parte del usuario. Puede alcanzar mejores resultados para casos específicos.

Cloud Translation implementa dos versiones: la versión básica y la versión avanzada. La versión básica permite la traducción de un máximo de 30 000 caracteres por solicitud. El corpus para traducir constó de 2 051 217 caracteres, por lo que se habría necesitado de más de una solicitud para realizar la traducción de todo el corpus. Por otra parte, la versión avanzada permite la traducción de un máximo de 100 000 000 caracteres por solicitud mediante la característica de traducción por lotes. Dicha característica fue la que determinó la preferencia de la versión avanzada sobre la versión básica para este proyecto.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

La traducción por lotes soporta tres tipos de archivos: HTML, TSV o TXT. El uso del archivo en formato TSV permite una traducción más ordenada y adecuada para este proyecto, debido a que requiere el uso de una columna de identificación y otra columna de contenido, que será la única columna traducida; además, permite la traducción independiente por cada muestra que se ubique en cada fila del archivo. Los formatos HTML o TXT no permiten la traducción independiente de cada muestra, sino que toman todo el corpus como un texto único. Por esta razón, la traducción por lotes por medio del formato TSV fue utilizada en este proyecto para la traducción y la re-traducción de las muestras del conjunto de datos en inglés. Es necesario mencionar también que el formato TSV debe de estar en conformidad con el formato RFC 2048 (Shafranovich s. f.).

2.2.3.3.Métrica BLEU (Bilingual Evaluation Understudy) para validación de traducción

Si bien es cierto, la métrica BLEU (Papineni et al. 2002) no es utilizada en el presente trabajo para la evaluación del proceso de traducción, aunque sí con fines demostrativos, fue fundamental para el desarrollo y comprensión de otras métricas con mayor correspondencia con evaluaciones humanas, como la métrica METEOR(Banerjee y Lavie 2005).

La métrica BLEU (Papineni et al. 2002) es una métrica utilizada para evaluar la calidad de traducción de una traducción automática calificándola en un rango del 0 al 1. Este tipo de métricas evitan la necesidad de intervención humana para evaluaciones de calidad de traducción que puede consumir tanto recursos humanos, como tiempo. Algunas de las ventajas de BLEU sobre evaluaciones humanas son la rapidez, el bajo costo, y la independencia del idioma.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Para el uso de la métrica BLEU, se necesitan 2 muestras: la primera, la muestra original o referencia, y la segunda, el resultado de la traducción (denominada muestra candidata). En el caso del presente trabajo, el resultado de la traducción o muestra candidata es el resultado del *backtranslation*, es decir, la primera muestra traducida al español, y traducida nuevamente al inglés.

El método de la métrica BLEU tiene su base en el concepto de precisión. En el contexto de oraciones, la precisión se refiere al número de palabras coincidentes entre la muestra candidata y de referencia, dividido entre el número total de palabras en la muestra candidata. En base a este concepto de precisión se construye la llamada precisión modificada (Papineni et al. 2002:2), que busca evitar una repetición innecesaria en el número de palabras coincidentes para evitar una sobre calificación de la traducción. Así, la precisión modificada agotará las palabras ya repetidas y coincidentes. Por ejemplo:

Muestra candidata: El el el el

Muestra referencia: El gato está sentado.

Si considerásemos la precisión, tendríamos un valor de 4/4, que es una calificación muy alta, pero evidentemente no correcta. Por otra parte, la precisión modificada nos brindaría un puntaje de 1/4: ya que el primer *el* tuvo coincidencia en ambas muestras, el *el* de la muestra de referencia será agotado, por lo que no se encontrarán más coincidencias con la muestra candidata.

El concepto de precisión modificada es calculado no solo para coincidencias de 1 palabra (unigramas), sino también n-gramas, siendo el n máximo recomendable de 4 (Papineni et al. 2002:4), es decir, coincidencias de 4 palabras. Más tarde, BLEU toma la media

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

geométrica de las precisiones modificadas de cada n-grama. Finalmente, BLEU multiplica el exponente de este valor por una penalidad de brevedad que se aplica únicamente cuando el número de palabras de la muestra candidata es menor al número de palabras de la muestra referencial. Así, el valor final de la métrica BLEU estará dada por:

$$BP = \begin{cases} 1, si\ c > r \\ e^{1-\frac{r}{c}}, si\ c \leq r \end{cases} \quad (4)$$

$$BLEU = BP.\exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \quad (5)$$

En donde:

- BP: penalidad por brevedad.
- r: número de palabras de muestra de referencia.
- c: número de palabras de muestra candidata.
- p: media geométrica de las precisiones modificadas de cada n-grama
- n: número de n-grama
- w_n: 1/N
- N: número máximo de n-grama (valor óptimo de 4)

2.2.3.3.1. Problemas con la métrica BLEU

La métrica BLEU, a pesar de ser ampliamente utilizada, presenta varios problemas que la hacen inadecuada para traducciones a nivel de oraciones. Por ejemplo, debido a que calcula la media geométrica de las precisiones modificadas de n-gramas, si el valor de precisión de un n-grama (por ejemplo, cuando n=4) es 0, entonces la calificación de toda la oración recibirá un

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

valor de 0. Estos problemas han intentado ser resueltos a través de diversas *técnicas de suavizado* (Chen y Cherry 2014). La técnica 4 de suavizado presentada por Chen y Cherry (2014:363) sería una de las más relevantes para nuestro estudio debido a que permite una mejor evaluación de las traducciones para oraciones de corta longitud y, en general, los títulos de noticias analizados en este trabajo poseen una corta longitud de palabras.

2.2.3.4.Métrica METEOR (Metric for Evaluation of Translation with Explicit Ordering) para validación de traducción

La métrica METEOR (Denkowski y Lavie 2014) fue utilizada en el presente trabajo para la validación de la traducción del conjunto de datos.

METEOR es una métrica que fue diseñada con el objetivo de hacer frente a varios de los problemas de la métrica BLEU. Esta métrica, ha demostrado tener un mayor valor del coeficiente de correlación de Pearson que la métrica BLEU con respecto a evaluaciones humanas, lo que evidencia su mayor eficacia para evaluar el resultado de una traducción automática: La métrica METEOR obtuvo una correlación de 0.964, mientras que la métrica BLEU de 0.817 (Banerjee y Lavie 2005:5).

2.2.3.4.1. Problemas de métrica BLEU considerados para el desarrollo de METEOR

- Uso de n-gramas de alto orden: A opinión de Banerjee y Lavie (2005:3), el uso de n-gramas de orden mayor a 1 en la métrica BLEU es una medida indirecta, y por esto ineficaz, de la correcta formación gramatical o el orden de palabras.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

- Ausencia de una métrica de cobertura para la correspondencia en n-gramas: Si bien es cierto la métrica BLEU utiliza el concepto de precisión como parte fundamental para la evaluación de traducciones automáticas, deja de lado el concepto de cobertura, o *recall* en inglés, que es un concepto que podría ser considerado como complementario al de precisión para la evaluación de cualquier modelo de aprendizaje automático. Dicho concepto y su relación con la precisión es discutido a mayor profundidad en la sección de Entrenamiento, evaluación y comparación de modelos de Aprendizaje Automático. Si bien es cierto, con el objetivo de hacer frente a la ausencia del concepto de cobertura, la métrica BLEU integra una penalidad por brevedad, dicha penalidad por brevedad no es considerada como una adecuada medida de la cobertura.

  El concepto de cobertura en este contexto se define como la proporción de los n-gramas coincidentes entre la muestra de referencia y la candidata dividido por la cantidad total de n-gramas en la muestra referencia, mientras que la precisión es la proporción de los n-gramas coincidentes dividido por la cantidad total de n-gramas en la muestra candidata.

- Coincidencia exacta de palabras: Para la evaluación de una traducción, BLEU considera únicamente la coincidencia exacta de palabras; sin embargo, deja de lado aspectos importantes en el lenguaje, como, por ejemplo, la coincidencia por raíz de palabra, o la coincidencia por sinónimos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

- Uso de la media geométrica de n-gramas: Como ya fue mencionado, el uso de la media geométrica de n-gramas puede provocar que, cuando uno de los n-gramas tenga como calificación 0, la calificación final sea indeterminada.

2.2.3.4.2. Fundamentos de métrica METEOR

Dada una muestra candidata y una muestra de referencia, la métrica METEOR, al igual que la BLEU, identifica la correspondencia entre las palabras o unigramas de ambas muestras. Esta correspondencia se realiza de la misma manera que el concepto de precisión modificada, de modo que una palabra coincidente no puede volver a ser utilizada para establecer una nueva coincidencia. Esta correspondencia es denominada alineación. A diferencia de BLEU, METEOR realiza dicha alineación considerando tres módulos: el módulo de exactitud para establecer la coincidencia exacta de palabras, el módulo *stemmer* para establecer la coincidencia de raíces, y el módulo de sinónimos para establecer la coincidencia de sinónimos. Es necesario mencionar que el hecho de considerar módulos para coincidencia de raíces y sinónimos hace que la métrica METEOR sea dependiente del idioma sujeto de estudio. Esto, a opinión del autor, es una de las características que permite mayor eficacia en la evaluación de traducciones.

Luego de realizar dicha alineación, se obtiene el denominado F_{mean}, que es una medida utilizada más tarde en la calificación de la traducción, mediante el cálculo de la media armónica de los conceptos de precisión y cobertura. La fórmula general para el cálculo de F_{mean} , precisión (P) y cobertura (R por el inglés *recall*) se expresa a continuación (Lavie y Agarwal 2007:2).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

$$F_{mean} = \frac{P * R}{\alpha P + (1 - \alpha)R} \quad (6)$$

$$P = \frac{m}{t}, R = \frac{m}{r} \quad (7)$$

En donde:

- α: constante.
- m: número de unigramas coincidentes en muestra referencia y candidata.
- t: número total de unigramas en muestra candidata.
- r: número total de unigramas en muestra referencia.

Más tarde, METEOR considera una penalidad. Esta penalidad actúa como una medición directa de la correcta formación gramatical, u orden de palabras, que en la métrica BLEU se medía indirectamente mediante la precisión modificada de n-gramas de alto orden. La fórmula general de dicha penalidad se expresa a continuación (Lavie y Agarwal 2007:2):

$$Pen = \gamma(frag)^{\beta} \quad (8)$$

En donde:

- $frag$: denominada fracción de fragmentación
- γ: constante entre 0 y 1 que determina la penalidad máxima.
- β: constante

La denominada fracción de fragmentación *frag* se calcula mediante la división del número mínimo de grupos de palabras formados de modo que los unigramas de cada grupo se

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

encuentren en posiciones adyacentes tanto en la muestra candidata como en la muestra referencia, entre el número total de unigramas coincidentes.

$$frag = \frac{ch}{m} \tag{9}$$

En donde:

- ch: número mínimo de grupos de palabras formados de modo que los unigramas de cada grupo se encuentren en posiciones adyacentes.
- m: número total de unigramas coincidentes.

Por ejemplo, en la muestra candidata "El perro comió tiburón" y la muestra de referencia "el perro comió un tiburón" se tienen los valores de $ch = 2, m = 4$.

Se puede calcular el puntaje final de la calificación METEOR mediante la siguiente fórmula:

$$M = (1 - Pen) * F_{mean} \tag{10}$$

Se puede apreciar que las fórmulas para *Pen* y para F_{mean} se encuentra parametrizadas, teniendo como parámetros a α, β, γ. En la versión original de METEOR (Banerjee y Lavie 2005) el valor de dichos parámetros se encontraba inicializado con: $\alpha = 0.9, \beta = 3.0, \gamma = 0.5$. Sin embargo, estudios posteriores (Lavie y Agarwal 2007) demostraron que el variar los valores de dichos parámetros podría mejorar la correlación con las evaluaciones humanas. Así mismo, se determinó que la mayor correlación con las evaluaciones humanas podría lograrse para

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

distintos valores de dichos parámetros de acuerdo con el idioma sujeto de estudio. En el caso del inglés, se determinó que los valores óptimos correspondían a $\alpha = 0.81, \beta = 0.83, \gamma = 0.28$ (Lavie y Agarwal 2007:3).

Es importante mencionar que, a diferencia de BLEU, la métrica METEOR puede alcanzar valores mayores que 1. Esto se permite a partir del cálculo de F_{mean}. Por ejemplo, para el caso en que $\alpha = 0.81$, se pueden obtener valores para F_{mean} mayores que 1 para los valores de P y R que cumplan con las siguientes características:

$$P < 0.19, R < \frac{81P}{100P - 19} \tag{11}$$

$$P > 0.19, R > \frac{81P}{100P - 19} \tag{12}$$

2.2.3.4.3. Stemming

Stemming es un método de obtención de información que busca extraer la raíz de una palabra mediante el filtrado de letras sufijas de la misma palabra. Este método es usado como parte del módulo de *stemming* en el cálculo de la métrica METEOR. Para la implementación de dicho método se hace uso de algoritmos de *stemming*.

El algoritmo de *Porter* (Porter 1980) fue uno de los algoritmos de *stemming* pioneros más comunes y utilizados. Fue implementado por Martin Porter y publicado como un algoritmo de obtención de información. El algoritmo de *Snowball* (Porter 2001), también conocido como *Porter 2* o *Porter English*, es un algoritmo también desarrollado por Martin Porter que buscó hacer frente a errores del algoritmo *Porter*. *Snowball* fue presentado como parte también de un

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

lenguaje de programación del mismo nombre (Porter 2001). En el presente trabajo, se hizo uso del algoritmo de *Snowball* para el cálculo de las raíces dado como parte del cálculo de la métrica METEOR.

2.2.4. Extracción de Características: transformación a datos numéricos

La extracción de características es el proceso mediante el cual se obtendrán valores numéricos relacionados con distintas características del texto que forma parte del corpus del conjunto de datos. Dichos valores numéricos serán utilizados en las secciones posteriores para el entrenamiento de los modelos de Aprendizaje Automático.

La presente sección inicia con una discusión sobre el concepto de palabras embebidas y el uso de vectores de palabras para codificar características semánticas y sintácticas de un texto. Luego, se discute la extracción de características complejas y sintácticas mediante la identificación de entidades y las funciones gramaticales de las palabras. Por último, se procede a la discusión de dos aproximaciones comunes para el análisis de sentimientos o extracción de características psicológicas: aproximación por aprendizaje máquina y aproximación por uso de léxico de sentimientos.

2.2.4.1.Word Embeddings

El concepto de *Word Embeddings* o palabras embebidas fue introducido por primera vez por Hinton (1986). Palabras Embebidas, llamadas también vectores de palabra, es un nombre que designa a un grupo de modelos de lenguaje y métodos de selección de

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

características (Srinivasan 2018:84). Su principal objetivo es el mapeo de palabras textuales o frases hacia vectores de n dimensión, que son valores numéricos que podrán ser utilizados para aplicaciones de inteligencia artificial. Dichos vectores codifican características semánticas o sintácticas de las palabras bajo la suposición de que es posible inferir el significado de una palabra a partir de su contexto, es decir, palabras en contextos similares deben de tener significados similares (Harris 1981).

Un ejemplo de la codificación a través de vectores presente en las palabras embebidas se puede observar en la Tabla 2. Ejemplo de Word Embeddings. Fuente: Elaboración Propia. En dicha tabla se presentan los vectores de palabras de dimensión 2 para un vocabulario de 4 palabras. Se puede observar que, en dichos vectores, el significado de las palabras es codificado a través de dos columnas o dimensiones que hacen referencia al sexo y a la juventud. Así, la palabra *niño* y *hombre* comparten el sexo masculino, por lo que tienen un 1 en la dimensión correspondiente al *sexo masculino*. De la misma manera, la palabra *niña* y la palabra *niño* hacen referencia al concepto de juventud, por lo que ambos comparten un 1 en la columna respectiva.

Tabla 2. Ejemplo de Word Embeddings. Fuente: Elaboración Propia

	Sexo Masculino	Juventud
Hombre	1	0
Mujer	0	0
Niño	1	1
Niña	0	1

Si bien es cierto, el vocabulario del ejemplo presentado representa un modelo de palabras embebidas muy simple con solo 2 dimensiones, en la práctica los modelos de lenguaje pueden ser mucho más complejos, comprendiendo un vocabulario más amplio y un mayor número de dimensiones. Dichos modelos pueden constar de cientos de dimensiones que codifican más características semánticas y que, por lo tanto, son de mayor utilidad para

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

aplicaciones de aprendizaje automático, pero a su vez pierden su interpretabilidad. Debido a esta complejidad, son necesarios también algoritmos que permitan la eficiente codificación de un vocabulario en un determinado número de dimensiones.

2.2.4.1.1. Entrenamiento de vectores de palabras

El entrenamiento de vectores de palabras se ha realizado últimamente a través de aproximaciones basadas en redes neuronales que han superado en eficiencia y eficacia a los modelos de distribución semántica, también llamados modelos por conteo, utilizados con mayor frecuencia anteriormente (Baroni, Dinu, y Kruszewski 2014). Estos modelos basados en redes neuronales son conocidos como *CBOW* (bolsa de palabras continua) y *skipgram. CBOW* se basa en que, dado un grupo de palabras parte del contexto, se usa la suma de los vectores de dichas palabras para predecir una palabra objetivo, mientras que *skipgram* se basa en que, dada una palabra objetivo, se intenta predecir todas las palabras que podrían formar parte del contexto de dicha palabra objetivo. Debido a que el modelo *CBOW* hace uso del contexto para predecir la palabra objetivo, es mucho más eficiente que *skipgram* para el entrenamiento de vectores.

Por ejemplo, en la oración *I am selling these fine leather jackets*, si se quiere obtener el significado de la palabra *leather*, el modelo *CBOW* tomará las palabras *I am selling these fine jackets* y utilizará la suma de sus vectores para predecir la palabra *leather*, mientras que el modelo *skipgram* utilizará la palabra *leather* e intentará predecir todas las palabras que podrían formar parte de su contexto.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

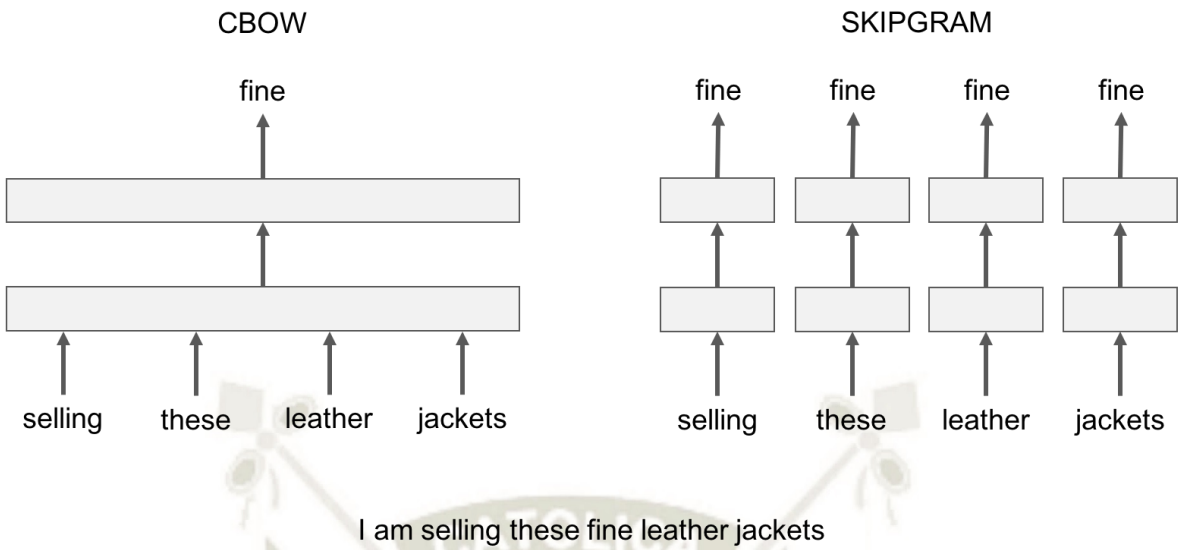


Figura 2. CBOW y skipgram (fastText s. f.)

2.2.4.1.2. Similitud y relaciones lingüísticas de vectores de palabras

Como se mencionó anteriormente, los vectores de palabras codifican características semánticas y sintácticas. Al basarse en valores numéricos, se podría asumir que se puede realizar operaciones matemáticas con los vectores de palabras para obtener características adicionales.

La primera característica adicional es la similitud de vectores de palabras. Esta trataría de representar la similitud semántica o de significado entre palabras. En matemáticas, la similitud de vectores puede hallarse mediante la similitud coseno, la cual se halla a través de:

$$similitud\ coseno = \cos\theta = \frac{\vec{A} \cdot \vec{B}}{|\vec{A}||\vec{B}|} \quad (13)$$

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

En donde $\vec{A}$ y $\vec{B}$ representan dos vectores, y θ el ángulo entre ellos. Numerosos trabajos (Bojanowski et al. 2017; Sitikhu et al. 2019) han evidenciado la alta correlación existente entre la similitud de significado de palabras, y el cálculo de la similitud coseno entre vectores de palabras. A partir de esto, se puede asumir que la similitud coseno entre vectores de palabras representaría lo similares que pueden llegar a ser dos palabras, lo que a su vez confirmaría la suposición de que vectores de palabras pueden codificar características semánticas.

La segunda característica adicional son las relaciones lingüísticas existentes entre vectores de palabras. Los vectores de palabras creados por *Word Embeddings* guardan relaciones lingüísticas que pueden ser expresadas a través de analogías u operaciones aritméticas entre ellas (Mikolov et al. 2013). Por ejemplo, la analogía *Alemania* es a *Berlín* como *Francia* es a *París* es una relación lingüística válida propia de los vectores de palabras, y, en la práctica, puede ser calculada mediante la operación aritmética entre los vectores *Alemaina – Berlín + Francia*, en donde el resultado válido sería *París* (Mikolov et al. 2013:5). Estas relaciones lingüísticas evidencian nuevamente las características semánticas que contienen los vectores de palabras.

2.2.4.1.3. Algoritmo para el entrenamiento de Word Embeddings

Muchos algoritmos han sido desarrollados con el objetivo de lograr un entrenamiento eficiente de los vectores de palabras. Dos de los más importantes son *Word2Vect* (Mikolov et al. 2013) y *fastText* (Joulin et al. 2016). Dichos algoritmos pueden usar cualquiera de los dos modelos *CBOW* y *skipgram*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

En el presente trabajo, se utilizaron vectores de palabras entrenados usando *fastText* y el modelo *CBOW*. Como se mencionó anteriormente, *CBOW* resulta más eficiente para el entrenamiento de vectores de palabras que el modelo *skipgram*. Por otra parte, se prefirió el uso de vectores entrenados mediante *fastText*. Si bien es cierto, la correlación entre juicios humanos y similitud de vectores para el idioma español es casi la misma en *fastText* como en *word2Vect* (Bojanowski et al. 2017:5), se prefirió el uso de *fastText* debido a que *fastText* propone la representación de vectores de palabras como la suma de vectores de n-gramas de caracteres. El uso de dichos vectores de caracteres es principalmente beneficioso para idiomas en donde se hace uso de declinaciones gramaticales y palabras compuestas, como en el caso del alemán o el ruso, y en donde una palabra compuesta no diferirá en gran medida de la palabra principal. Por ejemplo, en el alemán, la representación de la palabra *Tischtennis*, o *tenis de mesa*, no será completamente diferente de la palabra *Tennis* (Bojanowski et al. 2017:5). Si bien es cierto, el idioma español no posee declinaciones gramaticales ni palabras compuestas así como en otros idiomas, el tema tratado en el presente trabajo, *fake news*, supone la presencia de errores gramaticales o informalidades gramaticales en las palabras. El efecto de dichas informalidades o errores, que no pertenecerían al vocabulario utilizado en el entrenamiento, sería despreciado gracias al uso de dichos vectores de caracteres, por lo que *fastText* es ideal para la aplicación de este trabajo.

2.2.4.2.Extracción de características complejas y de estilo

Según Horne y Adali (Horne y Adali 2017), las características relevantes a el tema de noticias falsas pueden ser categorizadas en tres grupos: complejas, estilísticas y psicológicas. Se denominan características estilísticas a aquellas características que contienen información gramatical y que nos permiten comprender el estilo del texto. Por ejemplo, consideramos como

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

estas al número de adverbios, sustantivos o verbos en la muestra. Se denomina características complejas a aquellas características que contienen información sobre la complejidad de la muestra a nivel de palabra. Esto es, el número de palabras totales, el tamaño promedio de palabra por muestra, o, por ejemplo, una combinación de una característica compleja con una estilística, como, por ejemplo, la relación entre el número de verbos y el número total de palabras.

2.2.4.2.1. Identificación de etiquetas de POS y NER

Se denomina etiquetas POS, *parte del discurso* por sus siglas en inglés, a las etiquetas que designan la parte o función gramatical de una palabra, por ejemplo, verbo, sustantivo o adjetivo. Se denomina etiquetas NER, *reconocimiento de nombres de entidades* por sus siglas en inglés, a las etiquetas que designan a una persona, organización o ubicación.

Para la identificación de etiquetas POS y NER se hizo uso de la librería spaCy (Honnibal y Montani 2017), que es una librería orientada a aplicaciones de procesamiento de lenguaje natural. Se prefirió el uso de esta librería a otras debido a su mayor eficiencia y velocidad de procesamiento (spaCy s. f.), y al hecho de que posee modelos de redes neuronales preentrenados con un corpus de noticias para el reconocimiento de etiquetas POS y NER (spaCy s. f.).

SpaCy (Honnibal y Montani 2017) se basa en el uso de una tubería de procesamiento (spaCy s. f.) que por defecto sigue los siguientes pasos:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

1. Tokenización: Mediante este paso, spaCy (Honnibal y Montani 2017) divide la muestra en segmentos de palabras y caracteres de acuerdo con determinadas reglas. Esta etapa no puede ser variada.

2. Asignación de POS: Se asigna las etiquetas POS para cada uno de los segmentos producto de la tokenización.

3. Asignación de etiquetas de dependencia: Se asigna las etiquetas de dependencia de los segmentos producto de la tokenización.

4. Detección y reconocimiento de entidades: Se identifica las entidades de cada segmento o token y se les asigna las etiquetas correspondientes. En el caso del español, se le asigna las etiquetas de lugar, persona y ubicación.

El presente proyecto no requirió de la identificación de la dependencia de palabras, por lo que se hizo uso únicamente de las etapas de tokenización, asignación de POS y detección y reconocimiento de entidades.

2.2.4.3.Extracción de características psicológicas o análisis de sentimiento

Mediante el análisis de sentimientos se busca determinar la polaridad de una muestra. Definimos como polaridad a la interpretación y clasificación de emociones expresadas en una muestra como positiva, negativa o neutral.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Para determinar dicha polaridad existen dos aproximaciones: la primera basada en el entrenamiento de un modelo de aprendizaje automático, y la segunda basada en el uso de léxico de sentimientos.

2.2.4.3.1. Análisis de sentimientos por aprendizaje automático

La aproximación basada en aprendizaje automático consiste en un problema de clasificación por medio de técnicas de aprendizaje automático supervisado en donde se trata de clasificar oraciones como positivas o negativas. El tema principal del presente proyecto no involucra el entrenamiento de un modelo para la clasificación de noticias u oraciones como positivas o negativas; sin embargo, se evaluó el uso de un modelo preentrenado para este fin.

Lamentablemente, para conocimiento del autor, no existen modelos preentrenados específicos para la clasificación de la polaridad de noticias en español. Por otra parte, los escasos modelos existentes para la clasificación de la polaridad en idioma español son entrenados con corpus no específicos para aplicaciones de noticias, sino más bien, con corpus extraídos de páginas de opinión (aylliote [2017] 2020), además de que su efectividad y fundamento no son lo suficientemente documentados.

2.2.4.3.2. Análisis de sentimientos por uso de léxico

La aproximación basada en un léxico de sentimientos se basa en el uso de un vocabulario de palabras cuyas palabras hayan sido clasificadas con una polaridad positiva o negativa utilizando un valor numérico, como, por ejemplo, 1, 0 y -1 para designar una polaridad positiva, neutra y negativa respectivamente. Luego, para determinar la polaridad de una

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

muestra, se promedia el valor numérico de la polaridad de cada una de las palabras que la componen.

A diferencia de la aproximación por aprendizaje automático, existen léxicos de sentimiento para varios idiomas, incluido el español, disponibles y ampliamente documentados (Chen y Skiena 2014).

2.2.4.3.3. Selección de aproximación para análisis de sentimientos

La aproximación de aprendizaje supervisado fue considerada inicialmente por algunas fuentes como más exacta que la aproximación por uso de un léxico de palabras (Chaovalit y Zhou 2005:7); sin embargo, trabajos posteriores contradijeron lo dicho demostrando que la diferencia entre ambas aproximaciones no es significativa (Dhaoui, Webster, y Tan 2017:15), e incluso que la aproximación por uso de léxico es mejor en algunos idiomas (Mukhtar, Khan, y Chiragh 2018). Esto, sumado a la escasez de correctamente documentados modelos específicos de aprendizaje automático en español y a la existencia de léxicos de palabras para el idioma español, hace a la aproximación por uso de léxicos de palabras la opción ideal para este proyecto.

2.2.5. Entrenamiento, evaluación y comparación de modelos de Aprendizaje Automático

Esta sección inicia con la presentación de los algoritmos de aprendizaje automático utilizados en el presente proyecto. El fundamento matemático de dichos algoritmos no es fundamental para la implementación del proyecto, por lo que no se pretende profundizar en

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

dicho tema. Luego, se presentarán conceptos relacionados a métricas relevantes en la evaluación de los modelos implementados.

2.2.5.1.Modelos de Aprendizaje Automático

2.2.5.1.1. Regresión Logística

La regresión logística es un modelo lineal utilizado en tareas de clasificación. En este modelo, la probabilidad que describe la posible predicción de una muestra está modelada por medio de una función logística. Esta función es ideal para tareas de clasificación binomial ya que limita las predicciones a dos resultados: 0 y 1. La función logística es definida por la siguiente función:

$$logistic(\eta) = \frac{1}{1 + \exp(-\eta)} \tag{14}$$

En donde η está representada por la función lineal:

$$\eta = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n \tag{15}$$

En donde x_p representa la característica p y β_p representa una constante para dicha característica.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

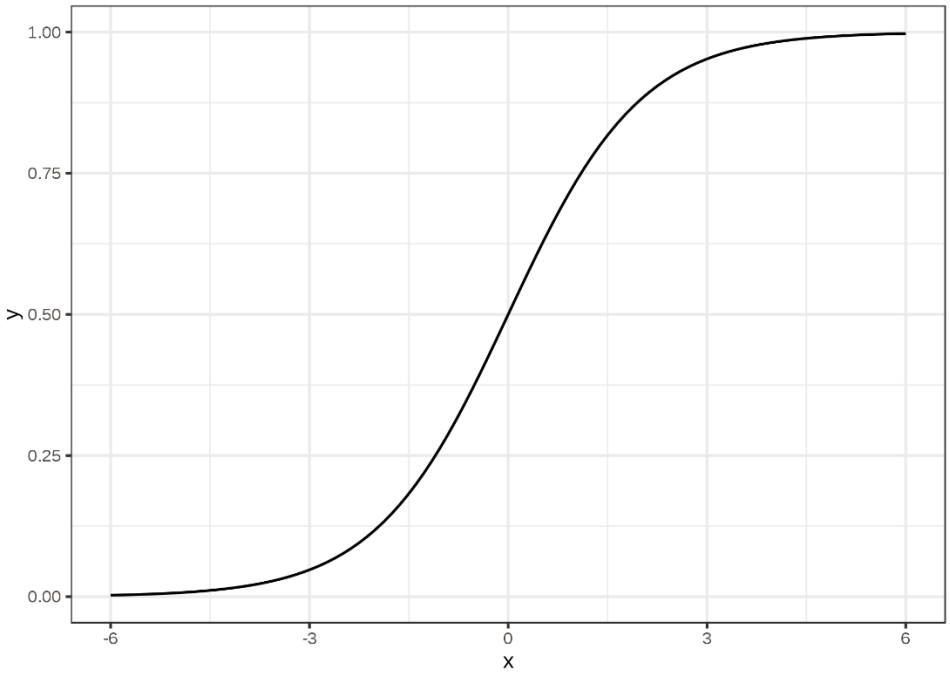


Figura 3. Gráfica de función logística (Molnar 2020)

2.2.5.1.2. Máquinas de Soporte Vectorial

Las máquinas de soporte vectorial, o SVM por sus siglas en inglés, son una serie de métodos de aprendizaje automático supervisado utilizados para tareas de clasificación, regresión y detección de valores atípicos.

Algunas de sus ventajas son (scikit-learn 0.23.2 documentation s. f.):

- Efectivo en espacios de altas dimensiones o numerosas características.
- Eficiente en términos de memoria.
- Versátil al permitir el uso de diferentes funciones de núcleo.

En cuanto a tareas de clasificación, la librería scikit-learn (Pedregosa et al. 2018) permite la implementación de SVC, o soporte vectorial de clasificación, y *LinearSVC*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

SVM se basa en la construcción de un hyper-plano o grupo de hyper-planos en un espacio de n dimensión, el que puede ser usado para tareas de clasificación. Un SVM puede ser comprendido mediante la representación de muestras de un conjunto de datos como puntos en un espacio, ubicados de modo que las muestras de diferentes categorías estén divididas por un margen claro que es lo más ancho posible y que rodea al hyper-plano. Las nuevas muestras por predecir son entonces mapeadas en el mismo espacio y su categoría dependerá de la región del hyper-plano en la que se encuentran.

SVC permite el uso de diferentes funciones de núcleo, o funciones de *kernel*, que modela la forma del hyper-plano. Entre las funciones de *kernel* encontramos la función lineal, polinomial, rbf o función básica radial, y la función sigmoid. La función lineal es la más simple de las funciones de núcleo mencionadas y, por ende, la que menos recursos computacionales requiere. Scikit-learn (Pedregosa et al. 2018) implementa además *LinearSVC* que es una implementación de SVC con la función de núcleo lineal optimizada para grandes conjuntos de datos y que es precisamente la función que se utilizará en el presente proyecto.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

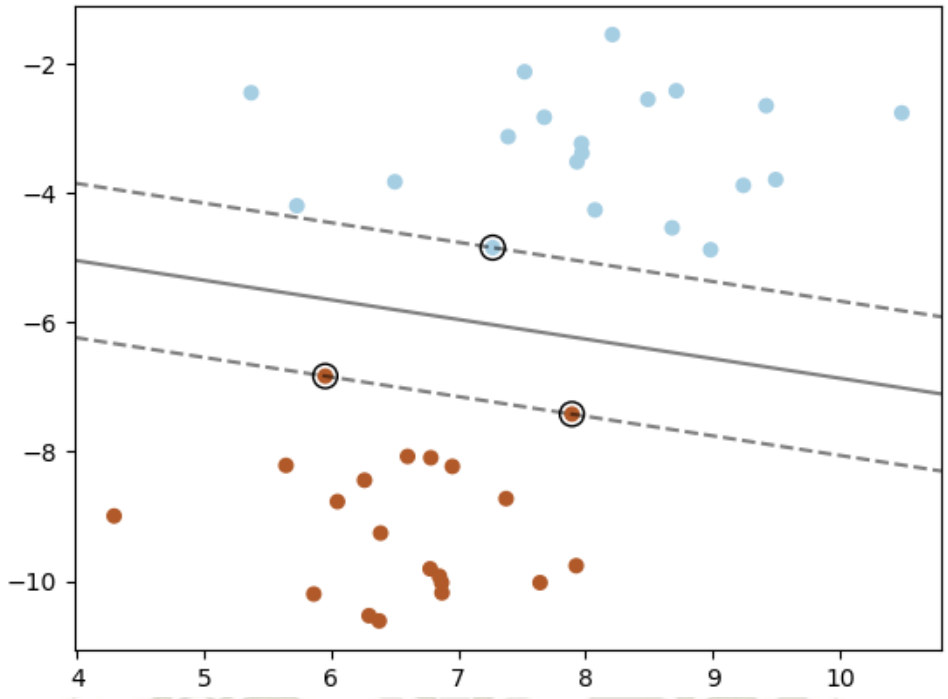


Figura 4. Gráfica de hyper-plano que separa dos categorías. Fuente: (scikit-learn 0.23.2 documentation s. f.)

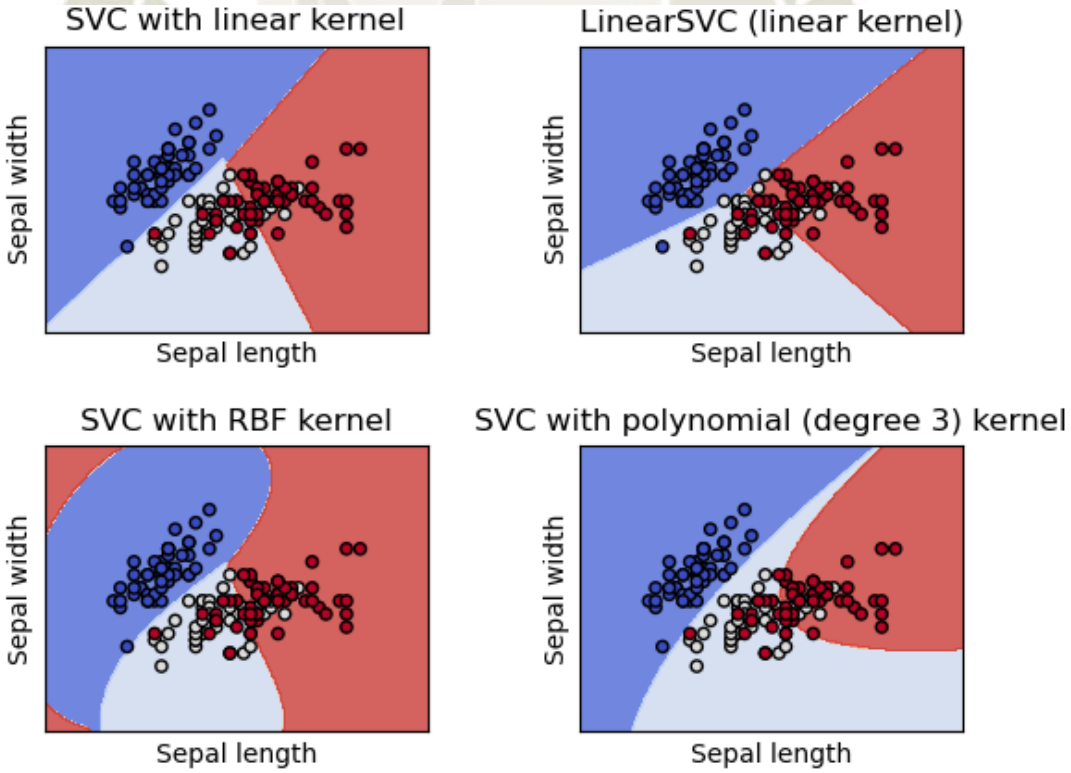


Figura 5. Gráfica de hyper-plano para diferentes funciones de núcleo. Fuente:(scikit-learn 0.23.2 documentation s. f.)

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

2.2.5.1.3. Bosques aleatorios

Los modelos de bosques aleatorios están basados en los modelos de árboles de decisión.

La predicción de un modelo de árbol de decisión simple puede ser explicada a través de la descomposición de una decisión en un número de componentes que dependerá del número de características. El concepto de árboles de decisiones y de bosques aleatorios pueden ser explicados mediante el siguiente ejemplo: se tiene una lista de libros que contiene libros que le gustan y que no le gustan a un usuario 1, se tiene también un usuario 2 que quiere predecir si un nuevo libro le gustará o no a el usuario 1. Entonces, el usuario 2 comienza a recopilar información, o características, mediante preguntas hacia el usuario 1 sobre la lista de libros que ya posee, como, por ejemplo, su género favorito, sus autores favoritos, entre otros. Entonces, conociendo estas características el usuario 2 será capaz de conocer si un libro nuevo le gustará o no al usuario 1. Este ejemplo, ilustra el concepto de un árbol de decisión simple.

Existen varios problemas con los árboles de decisión simple. Primeramente, haciendo referencia al ejemplo anterior, el punto de vista del usuario 2 puede encontrarse sesgado ya que sus preguntas pueden tender a elegir un libro que sean de su propio agrado. Esto se puede solucionar mediante el uso de múltiples árboles de decisión, o, en este caso, de múltiples usuarios. Un segundo problema es que el conjunto de datos que el usuario 2 y los nuevos usuarios poseen será siempre el mismo, por lo que encontraríamos también un sesgo en el mismo. La solución a esto sería el uso de pequeñas muestras diferentes para cada usuario, a lo que se le denomina *bootstrapping*. El tercer problema es que es probable que estos usuarios repitan algunas preguntas. Esto involucra también un sesgo. La solución es brindar a cada usuario diferentes muestras de preguntas, o características, que pueden realizar. La implementación de estas soluciones se conoce como bosques aleatorios. Los bosques aleatorios

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

están compuestos por múltiples y aleatorios árboles de decisiones simples cuyas fuentes de datos y características se subdividen aleatoriamente en muestras para cada uno de los árboles.

El modelo de árboles aleatorios es implementado como parte de la librería sickit-learn (Pedregosa et al. 2018). Dicho modelo permite la elección del número de árboles y el tamaño de las submuestras de las fuentes de datos. Se recomienda utilizar como tamaño de submuestras de fuentes de datos a la raíz cuadrada del número total de características para problemas de clasificación. Por otra parte, es necesario mencionar que un mayor número de árboles brinda mejores resultados de modelo; sin embargo, un mayor número de árboles involucra también un mayor consumo de recursos de computador (scikit-learn 0.23.2 documentation s. f.).

2.2.5.2.Métricas de clasificación para evaluación de desempeño

La clasificación es una tarea o problema en la que un modelo de aprendizaje automático busca predecir valores categóricos. En el caso del presente trabajo, los modelos a entrenar buscarán predecir el valor de la variable categórica de noticias falsas, es decir, si una noticia pertenece al grupo de las noticias falsas o no. Para la evaluación del desempeño de dichos modelos se utilizan métricas de clasificación. Para la comprensión de las métricas de evaluación es necesario primero la comprensión de la matriz de confusión.

2.2.5.2.1. Matriz de confusión

La matriz de confusión es una herramienta que nos permite comparar y visualizar los valores predichos y reales de un modelo. En el caso de un problema de clasificación binario está compuesta por 2 filas, correspondientes a predicción positiva o negativa, y 2 columnas,

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

correspondiente a valores reales positivos o negativos. La matriz de confusión introduce así 4 conceptos:

- Verdadero Positivo: El verdadero positivo comprende al número de predicciones positivas que son correctas.

- Falso Positivo: El falso positivo comprende al número de predicciones positivas que son falsas, es decir, que debieron de ser clasificadas como predicciones negativas. Al falso positivo se le conoce también como error de tipo 1.

- Verdadero Negativo: El verdadero negativo comprende al número de predicciones negativas que son correctas.

- Falso Negativo: El falso negativo comprende al número de predicciones negativas que son falsas, es decir, que debieron de ser clasificadas como predicciones positivas. Al falso negativo se le conoce también como error del tipo 2.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Matriz de confusión para una variable de salida dicotómica (clasificación)		Resultado de la predicción	
		Sí	No
Valor real de la clase	Sí	Verdadero positivo	Falso Negativo
	No	Falso positivo	Verdadero negativo

Figura 6. Explicación de Matriz de Confusión. (micromedix 2018)

2.2.5.2.2. Exactitud

La exactitud mide la división del número total de predicciones correctas, esto es, la suma de los verdaderos positivos y verdaderos negativos, entre el número total de predicciones. Esta métrica es útil cuando el conjunto de datos está bien balanceado. Por ejemplo, si un modelo de aprendizaje automático debe predecir si una foto muestra a un perro o un gato, la exactitud será únicamente útil cuando el número de gatos sea próximo o igual al número de perros en el conjunto de datos.

$$exactitud = \frac{verdaderos\ positivos + verdaderos\ negativos}{todas\ las\ predicciones} \quad (16)$$

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

2.2.5.2.3. Sensibilidad

La sensibilidad, denominada *recall* en el idioma inglés, describe la habilidad de un modelo para encontrar todos los casos relevantes dentro de un conjunto de datos. La sensibilidad se mide con referencia a un valor. Por ejemplo, la sensibilidad medida con respecto al valor positivo es:

$$sensibilidad = \frac{verdadero\ positivo}{verdadero\ positivo + falso\ negativo} \tag{17}$$

Por otra parte, la sensibilidad medida con respecto al valor negativo es:

$$sensibilidad = \frac{verdadero\ negativo}{verdadero\ negativo + falso\ positivo} \tag{18}$$

2.2.5.2.4. Precisión

La precisión describe la habilidad de un modelo para encontrar lo que, desde la perspectiva del mismo modelo, pero no necesariamente coincidente con la realidad, son los casos relevantes. Al igual que la sensibilidad, la precisión se mide con referencia a un valor. Por ejemplo, la precisión medida con respecto al valor positivo es:

$$precisión = \frac{verdadero\ positivo}{verdadero\ positivo + falso\ positivo} \tag{19}$$

Por otra parte, la precisión medida con respecto al valor negativo es:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

$$precisión = \frac{verdadero\ negativo}{verdadero\ negativo + falso\ negativo} \quad (20)$$

2.2.5.2.5. Puntaje F1

El puntaje F1 es utilizado como una métrica que considera la sensibilidad y la precisión. Se calcula mediante la siguiente fórmula:

$$F_1 = 2 * \frac{precision * sensibilidad}{precision + sensibilidad} \quad (21)$$

2.2.5.2.6. Elección de métrica de evaluación

Cada una de las métricas mencionadas evalúan diferentes aspectos de las predicciones de un modelo de predicción automática; sin embargo, la elección de una métrica particular depende de la aplicación y el ámbito de dicho modelo. En el caso del presente trabajo de investigación, lo relevante es la detección de noticias falsas, y no lo bien que el modelo puede diferenciar entre una noticia falsa y una noticia verdadera. Es decir, lo relevante en este caso es la cantidad de noticias falsas correctas detectadas y no la cantidad total de detecciones correctas de noticias falsas y confiables, por lo que la métrica más relevante es la sensibilidad con respecto a las noticias falsas correctas, y no la exactitud. Entonces, el modelo seleccionado a implementar deberá de tener un mayor puntaje de sensibilidad, y no necesariamente de exactitud.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

CAPÍTULO III

3. Desarrollo

El presente capítulo describe el desarrollo e implementación del proyecto. Para la comprensión de los conceptos y razones de la implementación es necesaria la lectura conjunta de las secciones correspondientes incluidas en el marco conceptual.

Inicialmente, se seleccionará un conjunto de datos en idioma inglés del que se extraerá los datos más relevantes, como el dominio de donde proviene, y su título. Dicha etapa incluye además el filtrado, limpieza y muestreo de dicho conjunto de datos. Más tarde, se procederá a realizar la traducción automática y se extraerán los registros con mayores puntajes de acuerdo con métricas de evaluación. Como siguiente punto se encuentra la extracción de características, que es la etapa en la que se convertirán los datos textuales a datos numéricos que puedan ser utilizados para los modelos de aprendizaje automático. Luego, se entrenarán 16 modelos de

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

aprendizaje automático, de los cuales se seleccionará a aquel que posea los mejores puntajes de evaluación. Después, se procederá a adaptar el código desarrollado para su despliegue en una aplicación web. Por último, se presentará una interfaz que permitirá al usuario utilizar el modelo de clasificación.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

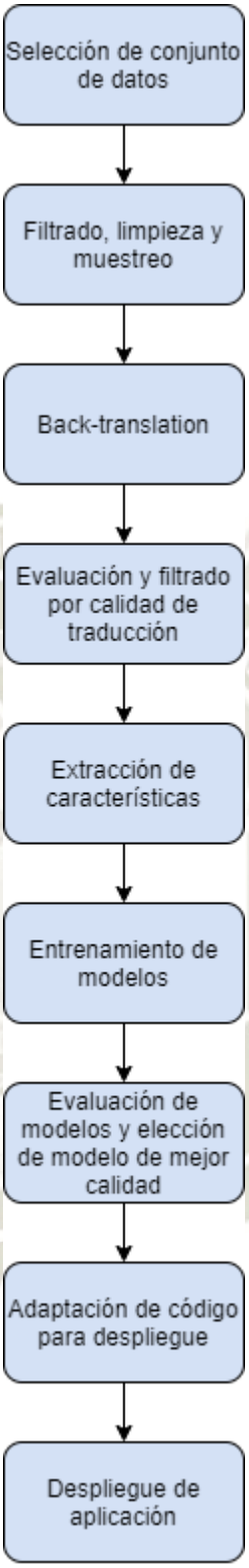


Figura 7. Diagrama de flujo de procedimiento general. Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.1.Exploración, análisis, limpieza y muestreo de conjunto de datos

Este primer capítulo iniciará con la explicación de la elección de la fuente de datos utilizada. Más tarde, se procederá a realizar un análisis y exploración previa de dicho conjunto de datos para identificar tendencias o comportamientos existentes. Además, se realizará un filtrado y limpieza de los datos y columnas que no son de utilidad o podrían representar un ruido perjudicial para el proyecto. Por último, se realizará un muestreo de la fuente de datos.

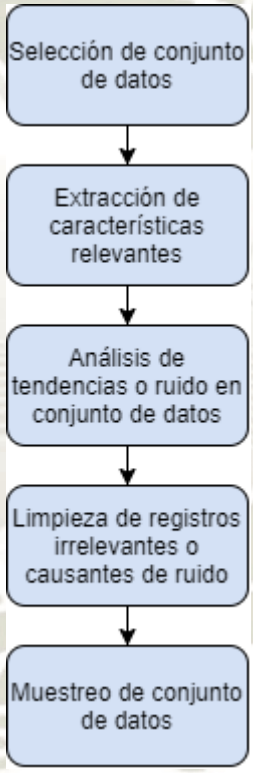


Figura 8. Diagrama de flujo sobre la exploración, análisis, limpieza y muestreo de conjunto de datos inicial.Fuente: Elaboración propia

3.1.1. Selección de conjunto de datos.

El conjunto de datos seleccionado se denomina *Fake News Corpus* (Szpakowski [2018] 2020). Este conjunto de datos está compuesto por artículos de noticias de una lista seleccionada

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

de 1001 dominios web. El conjunto de datos aún se encuentra en desarrollo, pero hay una versión pública disponible. La versión publica, que es la que se utiliza en este proyecto, consta de 9’408’908 artículos provenientes de 745 dominios. En total, el tamaño descomprimido de este conjunto de datos es de 27.3 GB.

Se eligió este conjunto de datos debido a la gran cantidad de artículos que lo compone, al hecho de que se encuentra preclasificado, y al hecho de que es un conjunto de datos de acceso libre, u *open source* por sus siglas en inglés. Es precisamente su gran volumen y el hecho de que es preclasificado lo que lo hace ideal para aplicaciones de aprendizaje automático supervisado.

3.1.1.1.Limitaciones del conjunto de datos.

De acuerdo con la documentación (Szpakowski [2018] 2020), la limitación más resaltante del conjunto de datos es que no fue preclasificado manualmente, sino más bien, fue preclasificado de acuerdo con una lista que contiene dominios web que son considerados como confiables, y dominios que no lo son. Esto podría introducir cierto ruido a nuestros modelos; sin embargo, dicho ruido o errores pueden asumirse ínfimos o despreciables considerando su cantidad con respecto a la gran cantidad de muestras disponibles. Por esta razón, la influencia de dicho ruido no es significativa.

3.1.1.2.Columnas del conjunto de datos

El conjunto de datos contiene los siguientes campos:

- *Id*: es una identificación única para cada registro

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

- *Domain*: Se refiere al dominio del que fue extraído cada noticia
- *Type*: Se refiere a la etiqueta preasignada a cada noticia para designar su clasificación.
- *url*: Dirección URL de donde se extrajo la noticia
- *content*: Contenido de la noticia
- *scraped_at*, *inserted_at* y *updated_at*: Se refiere a la fecha en que la noticia fue extraída, insertada al conjunto de datos y actualizada respectivamente.
- *Title*: título de la noticia
- *Authors*: Autores de la noticia
- *Meta_keywords*, *meta_descriptions*, *keywords*, *tags*: son campos utilizados en ciertas páginas para etiquetado.
- *Summary*: Resumen de la noticia
- *Source*: Fuentes de noticias.

3.1.1.3.Tipos de Etiquetas

Se refiere a las etiquetas utilizadas con las cuales se pre clasificaron las noticias del conjunto de datos.

Tabla 3: Etiquetas de tipos de noticias

Tipo	*Tag*	*Cantidad*	*Descripción*
Fake News	*fake*	*928 083*	*Fuentes que contienen noticias con información no veraz, contenido engañoso, o reportes distorsionados de noticias actuales.*
Sátira	*satire*	*146 080*	*Fuentes que utilizan humor, ironía, exageración o información falsa para comentar noticias actuales.*

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Tipo	*Tag*	*Cantidad*	*Descripción*
Sesgadas	*bias*	*1 300 444*	*Fuentes que tienen puntos de vista sesgados o hacen uso de propaganda, información descontextualizadas, u opiniones distorsionadas como hechos.*
Teorías Conspirativas	*conspiracy*	*905 981*	*Fuentes que son promotoras de teorías conspirativas.*
Noticias de Estado	*state*	*0*	*Fuentes en estados represivos.*
Ciencia Basura	*junksci*	*144 939*	*Fuentes que promueven pseudociencia, metafísica, falacias naturalistas, y otros argumentos científicamente dudosos.*
Noticias de Odio	*hate*	*117 374*	*Fuentes que promueven activamente el racismo, misoginia, homofobia, y otros tipos de discriminación.*
Clickbait	*clickbait*	*292 201*	*Fuentes que proveen contenido generalmente creíble, pero usan títulos e imágenes exageradas u engañosas.*
Precaución	*unreliable*	*319 830*	*Fuentes que pueden ser confiables, pero que su contenido requiere de una mayor verificación.*
Político	*political*	*2 435 471*	*Fuentes que proveen información verificable para promover ciertos puntos de vista u orientaciones políticas.*
Confiables	*reliable*	*1 920 139*	*Fuentes que circulan noticias e información de una manera consistente con las prácticas tradicionales y éticas de periodismo.*

Note: Descripción de los tipos de etiquetas con las que han sido preclasificadas las noticias (Szpakowski [2018] 2020). Incluye también el número de noticias por cada tipo.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.1.2. Extracción de Conjunto de Datos

Considerando el gran tamaño del conjunto de datos inicial, y con el objetivo de hacer el proceso más eficiente, se optó por la extracción de datos considerando los siguientes puntos importantes:

- Extracción por partes del conjunto de datos: Se optó por la extracción por partes del conjunto de datos mediante el uso de un iterador con un número de 100000 muestras por iteración. Esto, con el objetivo de hacer más eficiente y manejable, con respecto a recursos del computador, la extracción del gran volumen de datos.

- Extracción de columnas específicas del conjunto de datos: Considerando que el presente proyecto no requiere de todas las columnas para su desarrollo, se optó por extraer únicamente las columnas de importancia: *title*, correspondiente al título de la noticia, *type*, correspondiente a la etiqueta con la que la noticia fue preclasificada, y *domain*, dominio web de donde fue extraída la noticia y que será utilizado con fines de análisis de tendencias, sesgos y limpieza de datos.

- Limpieza de datos: Se descartaron las muestras que contenían datos inválidos o vacíos.

El código implementado se puede observar en ANEXOS, Código de programación Extracción por partes de Conjunto de datos. Luego de la extracción, se pudo reducir el conjunto de Datos de 27.3GB, a 1GB.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.1.3. Exploración, análisis y limpieza de conjunto de datos

En esta sección se realiza el proceso de análisis previo a la toma de muestras del conjunto de datos. Dicho proceso tuvo como objetivo la identificación de posibles tendencias o sesgos que pudiesen atentar contra la integridad de los datos.

3.1.3.1.Exploración por longitud de caracteres en título

Se buscó tendencias relacionadas a la longitud de caracteres en título.

Primero, mediante el análisis de la distribución de la longitud de caracteres en título por tipo, se logró identificar un gran número de valores atípicos correspondientes a la noticia de tipo *reliable* o confiable (Figura 9. Distribución de cantidad de caracteres en título de acuerdo a tipo. Exploración inicial).

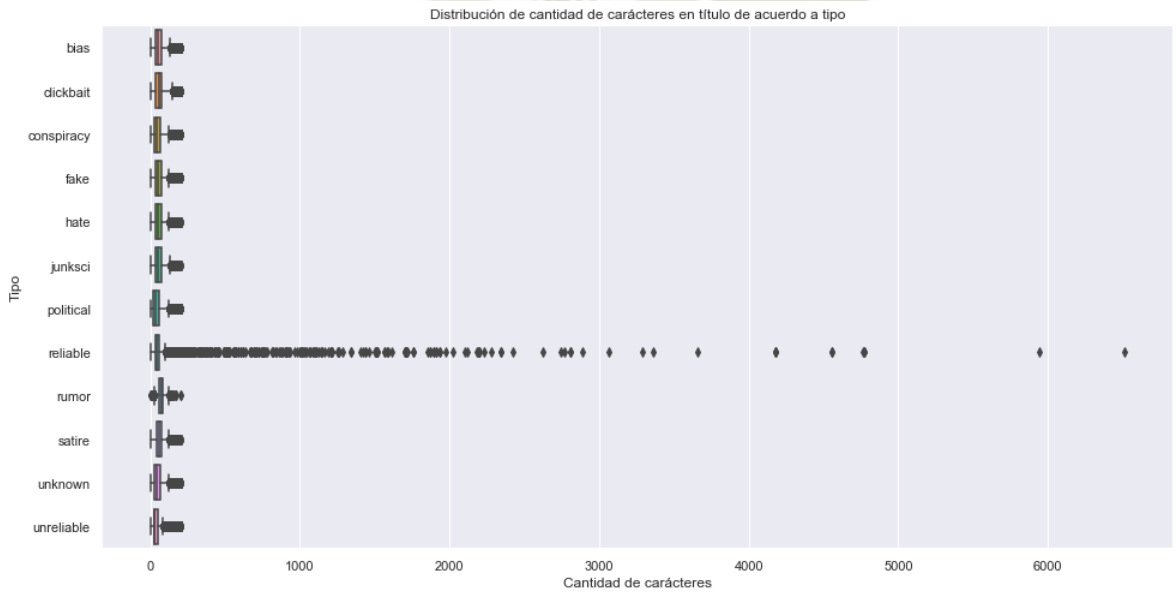


Figura 9. Distribución de cantidad de caracteres en título de acuerdo a tipo. Exploración inicial. Fuente: Elaboración Propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Un análisis más detallado nos permitió identificar dos cuestiones:

- Una gran cantidad de noticias tenían las palabras *Paid Notice*: *Paid Notice* hace referencia a noticias pagadas encontradas en el dominio *nytimes.com*. Dichas noticias pagadas forman parte de la sección de obituarios de dicho dominio (The New York Times s. f.), por lo que no son de importancia para nuestro análisis.

- Las noticias de otros tipos tenían un máximo de 200 caracteres por título: Se observó que los otros tipos de noticias, con excepción de las noticias confiables, tenían un tope máximo de longitud de caracteres de 200. Por esta razón, se procedió a limitar las noticias de tipo confiable para descartar aquellas que contenían más de 200 caracteres por título.

Tomando en cuenta dichas cuestiones, se procedió a realizar el filtrado de los datos (Figura 10. Distribución de cantidad de caracteres en título de acuerdo a tipo después de filtrado por longitud de caracteres y filtrado de Paid Notice).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

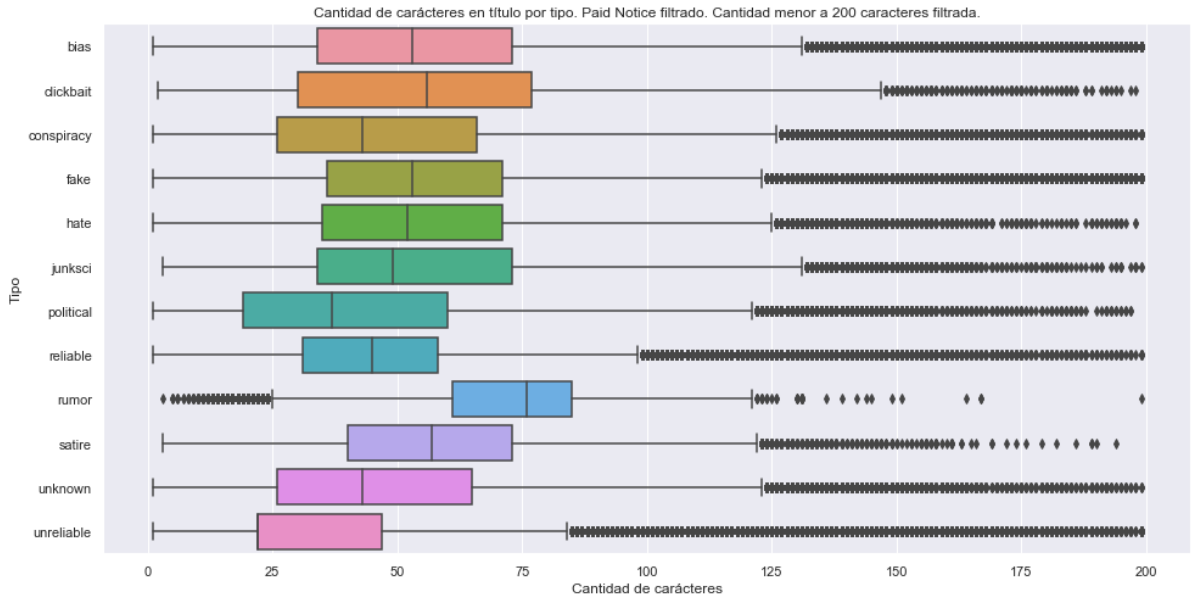


Figura 10. Distribución de cantidad de caracteres en título de acuerdo a tipo después de filtrado por longitud de caracteres y filtrado de Paid Notice. Fuente: Elaboración Propia

3.1.3.2.Exploración de dominio

Se exploró la distribución de noticias de acuerdo a dominio y tipo. Con respecto a las noticias de tipo confiable, se observó una gran cantidad de noticias correspondientes al dominio *nytimes.com* (Figura 11. Los 5 dominios con mayor n° de noticias de tipo 'reliable'). Con respecto a las noticias de tipo *fake* o falsa se observó una gran cantidad de noticias correspondientes al dominio *beforeitnews.com* (Figura 12. Los 5 dominios con mayor n° de noticias de tipo 'fake'). Con el objetivo de obtener conjuntos de datos balanceados, se procedió a disminuir la cantidad de muestras para los dominios *nytimes.com* y *beforeitnews.com* que contenían valores atípicos: se disminuyó las noticias confiables del dominio *nytimes.com* de 1370658 a 21764, que corresponde a la media de noticias por dominio para tipo confiable (Figura 13. Los 5 dominios con mayor n° de noticias de tipo 'reliable' después de balanceo); y las noticias falsas de dominio *beforeitnews.com* de 779396 a 24360, que corresponde al

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

segundo mayor número de noticias por dominio para tipo falsa (Figura 14. Los 5 dominios con mayor n° de noticias de tipo ‘fake’ despúes de balanceo).

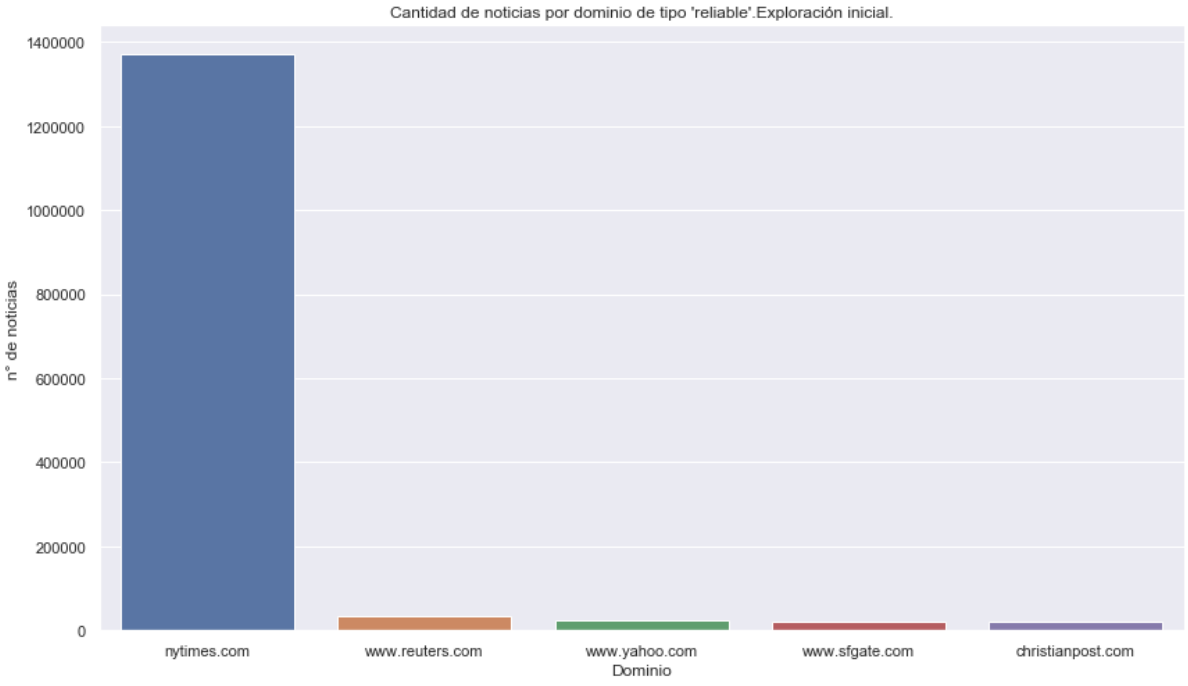


Figura 11. Los 5 dominios con mayor n° de noticias de tipo ‘reliable’. Fuente: Elaboración Propia

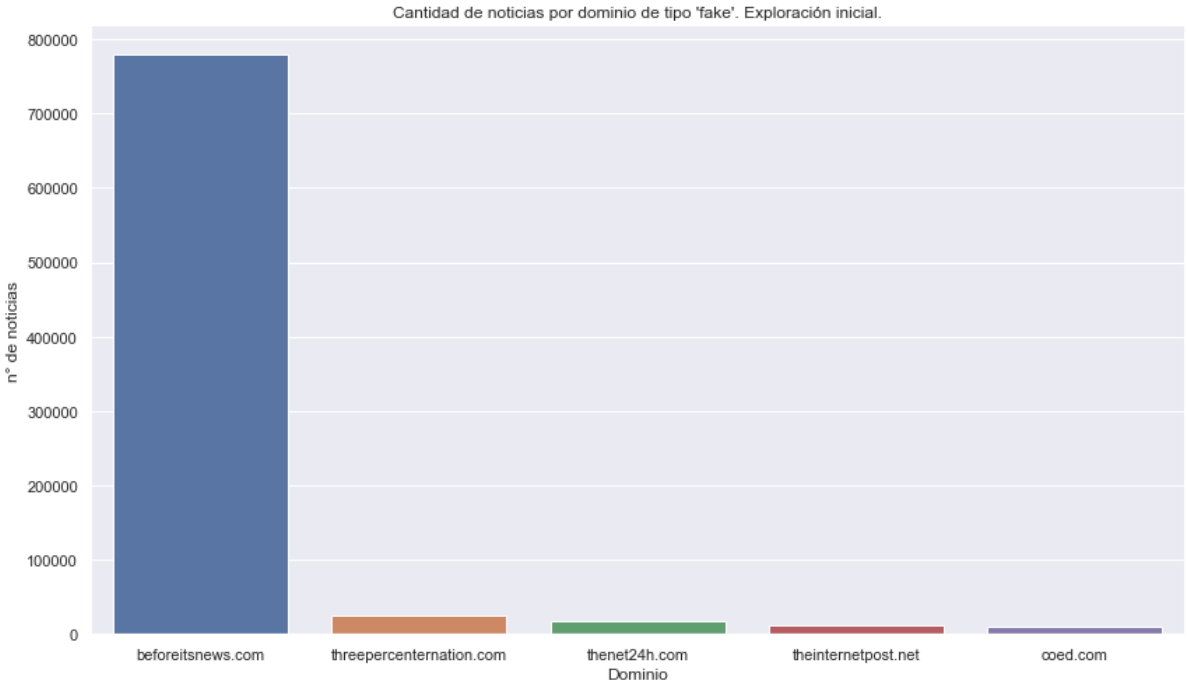


Figura 12. Los 5 dominios con mayor n° de noticias de tipo ‘fake’. Fuente: Elaboración Propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

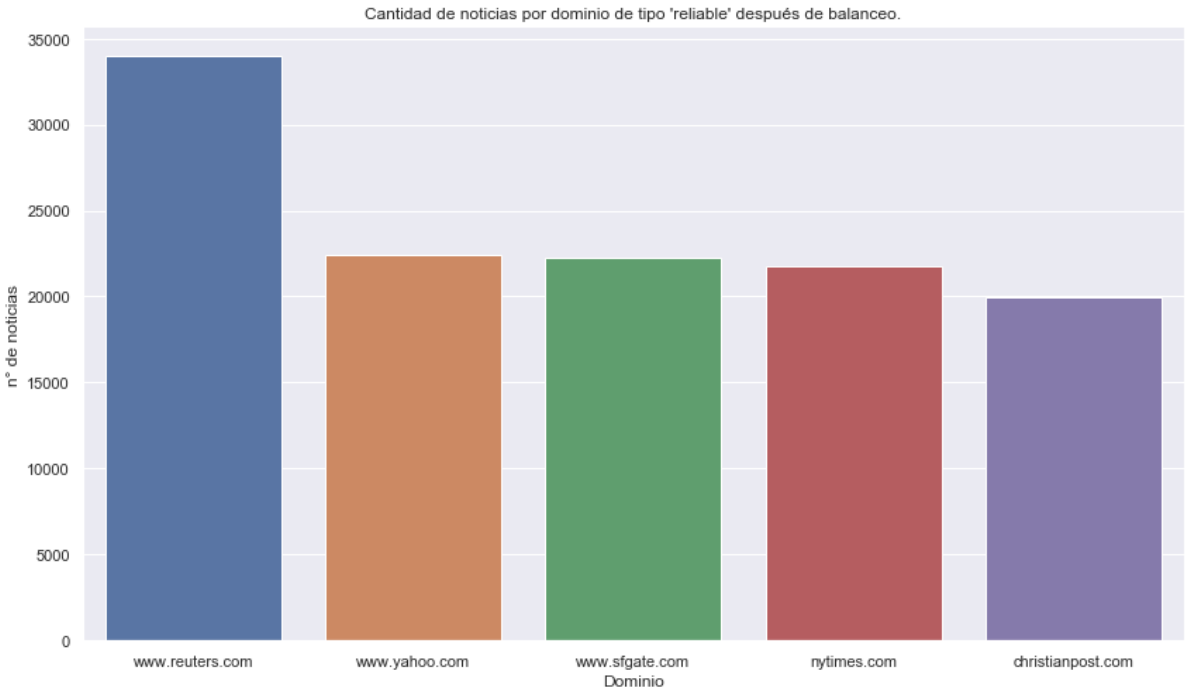


Figura 13. Los 5 dominios con mayor n° de noticias de tipo 'reliable' después de balanceo. Fuente: Elaboración Propia

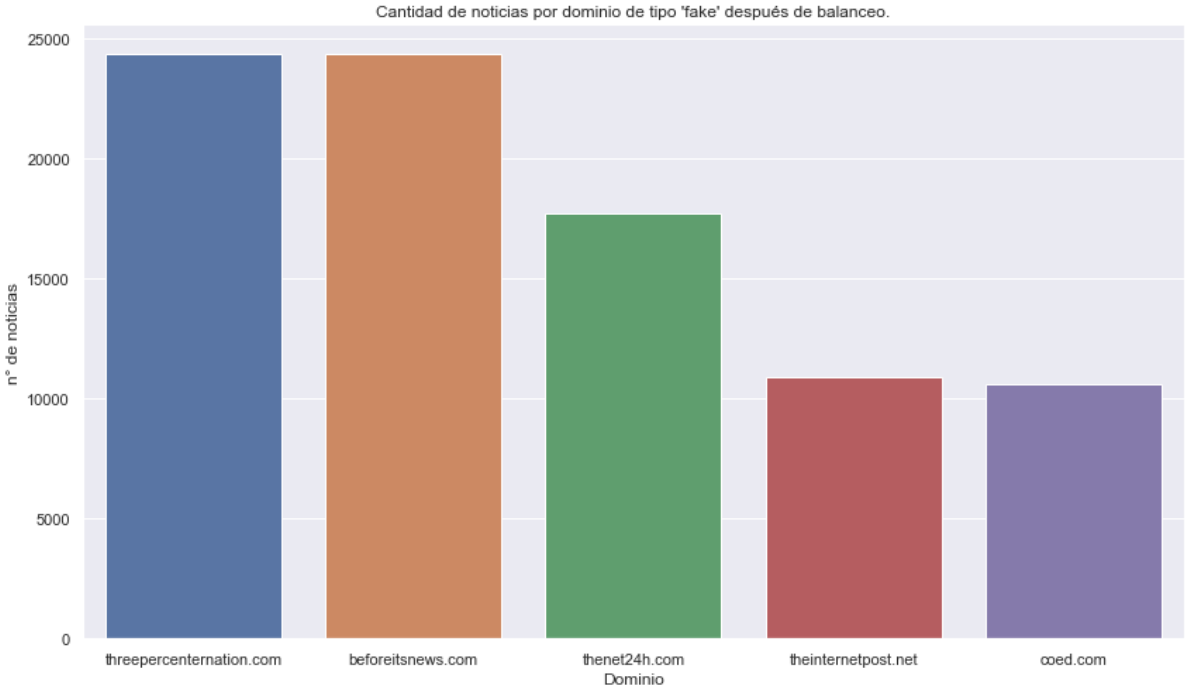


Figura 14. Los 5 dominios con mayor n° de noticias de tipo 'fake' despúes de balanceo. Fuente: Elaboración Propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.1.3.3.Exploración de longitud de caracteres por título para tipo 'fake' y 'reliable'

Se exploró la distribución del número de caracteres de título para noticias de tipo 'fake' y 'reliable' (Figura 15. Total de caracteres por número de muestras para tipos 'fake'(Rojo) y 'reliable'(Azul)).

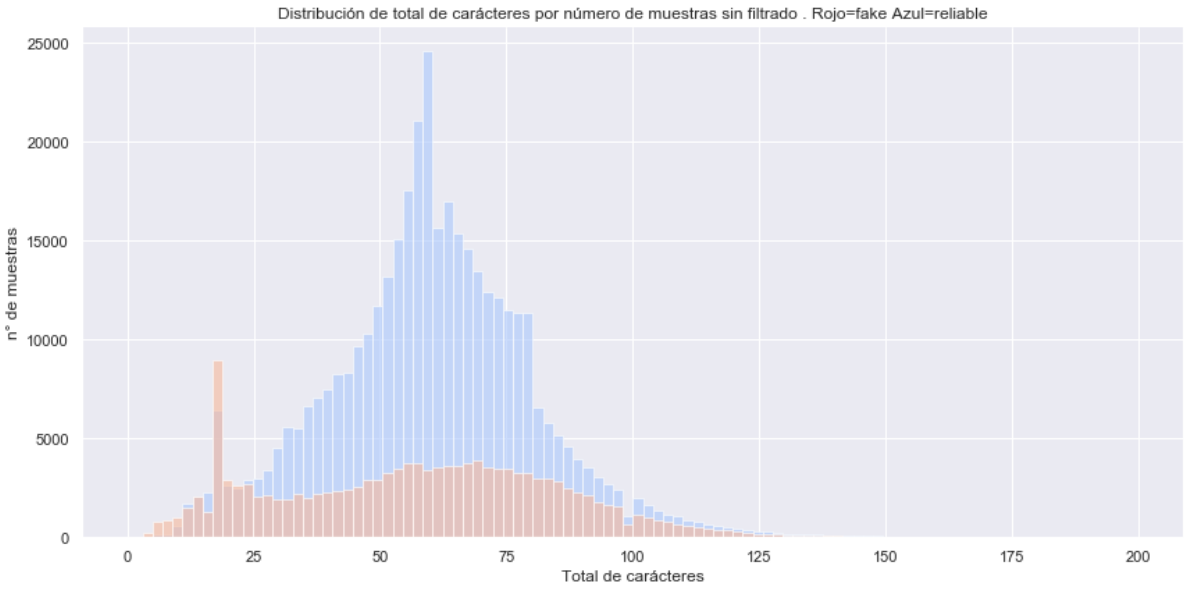


Figura 15. Total de caracteres por número de muestras para tipos 'fake'(Rojo) y 'reliable'(Azul). Fuente: Elaboración Propia

Se observó un pico de valores atípicos cerca al número total de caracteres de 20. Al explorar más profundamente dicho pico, se identificó dos puntos importantes:

- Los títulos con número de caracteres menores a 20 no tenían un contenido representativo o importante para el proyecto, ya que a menudo solo expresaban títulos generales que identificaban al dominio, como por ejemplo *THE INTERNET POST*, o expresaban una frase o nombre general, como por ejemplo *Financial Markets*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

- Muchos de estos títulos se repetían, lo que provocaba un mayor número de muestras sin contenido significativo adicional.

Adicionalmente, se observó también que la mayor cantidad de muestras para los tipos falsa y confiable tenían títulos por debajo de los 125 caracteres. Por estas razones, se optó por eliminar todos los títulos duplicados del conjunto de datos y filtrar las muestras para contener únicamente títulos por entre los 125 y 20 caracteres.

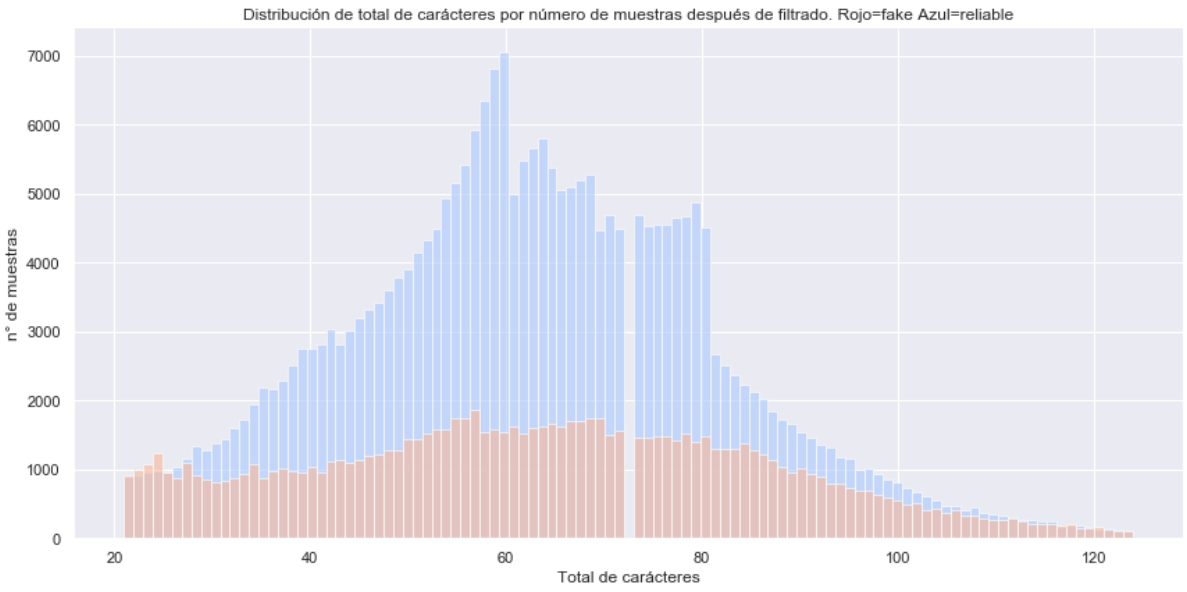


Figura 16. Total de caracteres por número de muestras para tipos 'fake'(Rojo) y 'reliable'(Azul) después de filtrado. Fuente: Elaboración Propia

3.1.3.4.Exploración de n-gramas

Se realizó la exploración de n-gramas para identificar aquellas palabras o conjuntos de palabras con mayor número de repeticiones en el corpus. Para este punto, se realizaron los siguientes procesos:

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

1. Se realizó una primera etapa de preprocesamiento del texto removiendo los caracteres de puntuación: ¡”#$%&()*+, -./:;⇔?@[\]^_`{|}~. Dichos caracteres, si bien es cierto podrían ser útiles para un análisis semántico, carecen de sentido para un análisis de ocurrencia de palabras o de n-gramas.

2. Se realizó una segunda etapa de preprocesamiento del texto removiendo las palabras comunes del idioma inglés, más conocidas como *stopwords*. Para esto, se utilizó la colección de palabras *stopwords* del idioma inglés de la librería NLTK (Bird et al. 2009). Se realizó esta etapa de preprocesamiento debido a que palabras que se repiten más frecuentemente, como, por ejemplo, *the* o *a*, no agregarían ningún valor al análisis.

3. Se vectorizó la colección de texto mediante la estrategia de modelo de bolsa de palabras, o más comúnmente llamada por el inglés *Bag of Words* o bolsa de n-gramas, por el inglés *Bag of n-grams*. Mediante esta estrategia, se construyó un vocabulario de todas las palabras o conjunto de palabras utilizadas y se otorgó un identificador para cada uno de ellos, proceso conocido como tokenización. Más tarde, se realizó el conteo de ocurrencias de cada una de estas palabras del vocabulario por cada muestra.

4. Se realizó la vectorización de texto para conjuntos de 1, 2 y 3 palabras o 1, 2 y 3-grama. Se graficaron los resultados para las 20 mayores ocurrencias de cada grupo para fuentes confiables (Figura 17. Exploración inicial de 1-grama, 2-grama y 3-grama para fuentes confiables. Fuente: Elaboración Propia) y para fuentes falsas

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

(Figura 18. Exploración inicial de 1-grama, 2-grama y 3-grama para fuentes 'fake'. Fuente: Elaboración propia.).

Después de realizar la exploración de n-gramas, se observó una gran ocurrencia de palabras o conjunto de palabras que correspondían a la descripción de la empresa, dominio o medio de noticias de donde se extrajo cada muestra, como, por ejemplo: "Reuters", "New York Post", "Fox News Latino", "Dc Gazette", y Freedom Daily. Esto indicó que dichos nombres o dominios eran repetidos como parte del título de las noticias. Con el objetivo de poder identificar el patrón que seguían dichos títulos, se procedió a extraer los últimos 30 caracteres de una muestra de dominios (Tabla 4. Ejemplo de muestra de dominios con últimos 30 caracteres de título. Fuente: Elaboración propia).

Tabla 4. Ejemplo de muestra de dominios con últimos 30 caracteres de título. Fuente: Elaboración propia

Dominio	Tipo de noticia	Últimos 30 caracteres del título
thecommonsenseshow.com	fake	Hodges – The Common Sense Show
dcgazette.com	fake	Isn't This Viral? ⋆ Dc Gazette
thelastgreatstand.com	fake	uclear :: The Last Great Stand
www.chron.com	reliable	ramento St – Houston Chronicle
au.news.yahoo.com	reliable	se images spared jail – Yahoo7

De esta manera, se logró identificar que la repetición del nombre del dominio en el título seguía el siguiente patrón: *espacio + 1 o 2 caracteres no alfanuméricos + nombre de dominio con espacios*. Adicionalmente, se observó también que el nombre de dominio *Houston Chronicle* y *Yahoo7* se repetían incluso para dominios que aparentemente no llevaban el mismo título, como por ejemplo *www.chron.com* y *au.news.yahoo.com*. Considerando dicho patrón y las dos excepciones mencionadas, se implementó una función para eliminar dicha repetición de dominio en el título (ANEXOS. Función con expresiones regulares usada para eliminar nombres de dominio repetidos en título). Adicionalmente, se eliminaron dominios que no

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

contenían contenido relacionado a noticias: *coed.com*, *disneyworld.disney.go.com*, *wiki.mozilla.org*, *www.wikihow.com*, *www.legacy.com*, *interestingdailynews.com*, *healthycareandbeauty.com*, *16wmpo.com*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

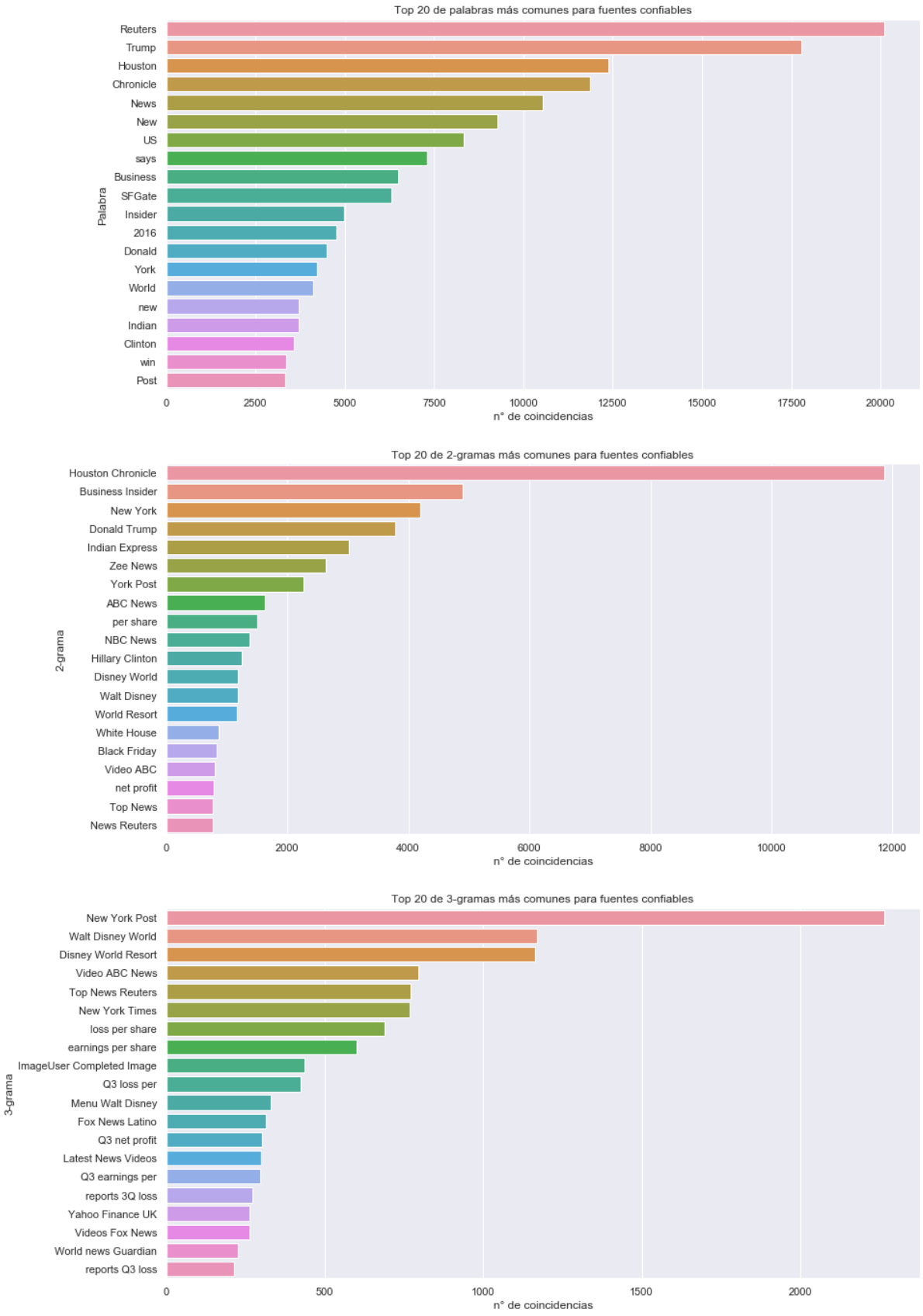


Figura 17. Exploración inicial de 1-grama, 2-grama y 3-grama para fuentes confiables. Fuente: Elaboración Propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

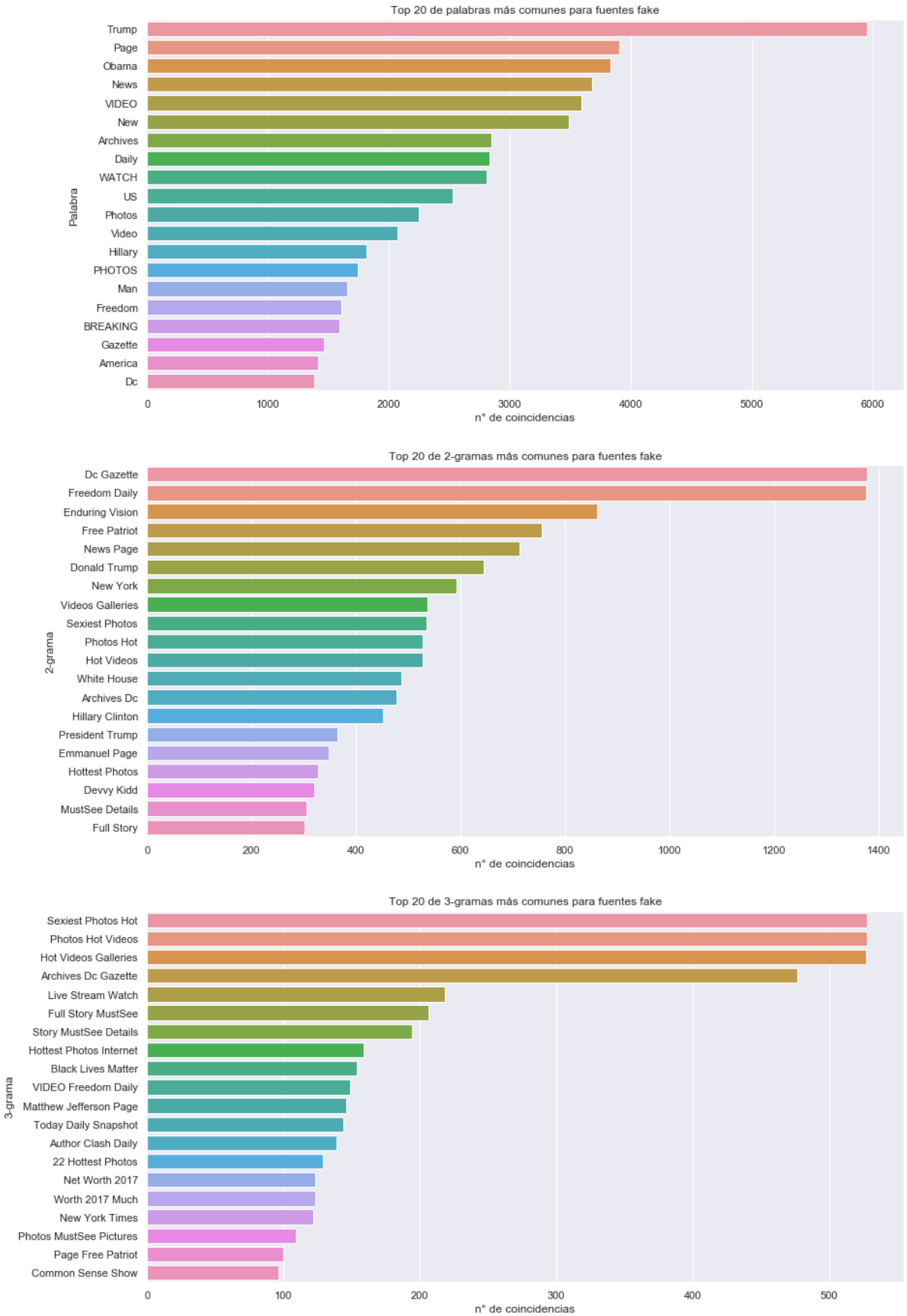


Figura 18. Exploración inicial de 1-grama, 2-grama y 3-grama para fuentes 'fake'. Fuente: Elaboración propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.1.3.5.Exploración de inconsistencias

Se procedió a identificar anomalías en títulos a partir de las palabras más comunes para realizar un filtrado del conjunto de datos. Algunas de estas anomalías fueron:

- Los títulos que contenían las palabras *Q3*, *3Q* y *per share* poseían un desbalance en dominios o tipo. En el caso de las muestras que contenían el término *per share*, 1291 provenían de fuentes confiables y sólo 1 de fuentes no confiables; además de que 1252 de estas provenían de dominios derivados del dominio *www.reuters.com*. Así mismo, 3446 de un total de 3456 de las muestras que contenían el término *Q3* o *3Q* provenían de fuentes confiables. Los términos mencionados corresponden a términos relacionados con la temática de economía o finanzas, y no son términos que evidencien directamente si una fuente es confiable o no. Debido a esto, y al desbalance de estas muestras, se optó por descartar los registros que contenían dichos términos.

- El término *Page* también fue un término muy común en los títulos. Una observación más detallada nos permitió determinar que una gran cantidad de las muestras con dicho término, 3062 de 3233, correspondían a un patrón *Page* + *número*. Dicho patrón es usado para designar el número de página de un artículo de varias páginas, pero con el mismo título. Se optó por extraer el patrón correspondiente mediante una función basada en expresiones regulares (Función con expresiones regulares para extracción de patrón *Page* + *número* de títulos) y eliminar las muestras que poseían dicho patrón.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

- Se determinó también que existían inconsistencias en los títulos. Dichas inconsistencias consistían en títulos que estaban escritos con caracteres desconocidos que no correspondían al idioma inglés, por ejemplo, Lịch phát sóng thvl1 ngày 3 tháng 3. Se procedió a eliminar dichas inconsistencias mediante el filtrado de los títulos que no poseyeran vocales del idioma inglés.

- Muchos títulos se encontraban duplicados a lo largo de varias muestras, por ejemplo, el título *News* y *Politics* se repetían 325 y 116 veces respectivamente. Se eliminaron dichos duplicados.

3.1.3.6.Exploración de distribución de número de palabras por título

Luego de realizar la exploración de términos comunes e inconsistencias, se analizó la distribución del número de palabras por título.

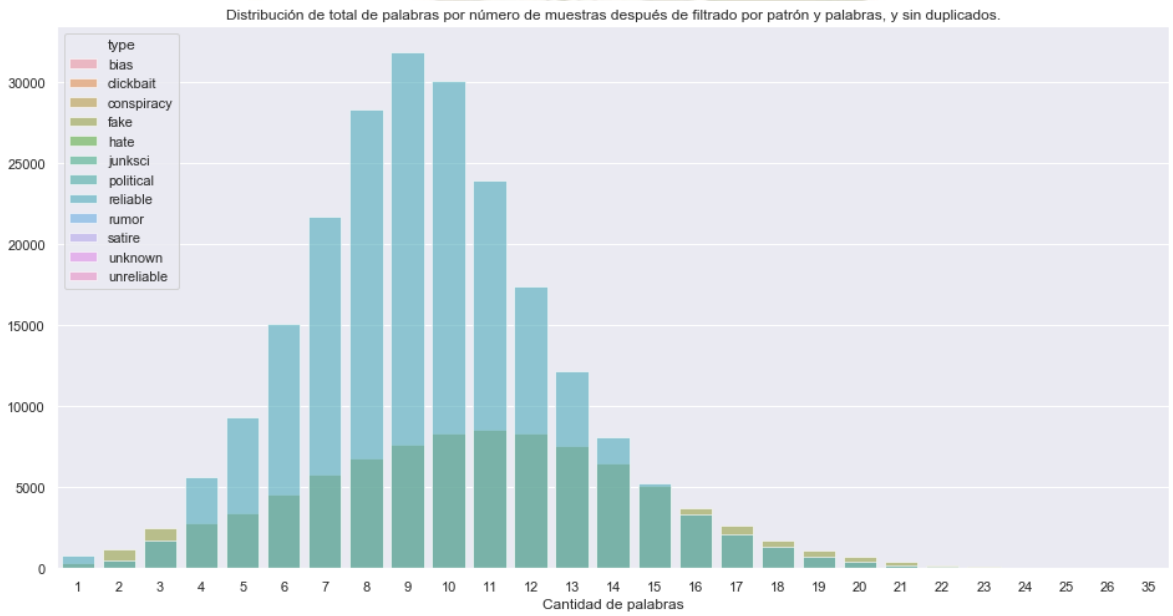


Figura 19. Distribución de total de palabras por número de muestras después de filtrado por patrón y sin duplicados. Fuente: Elaboración Propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

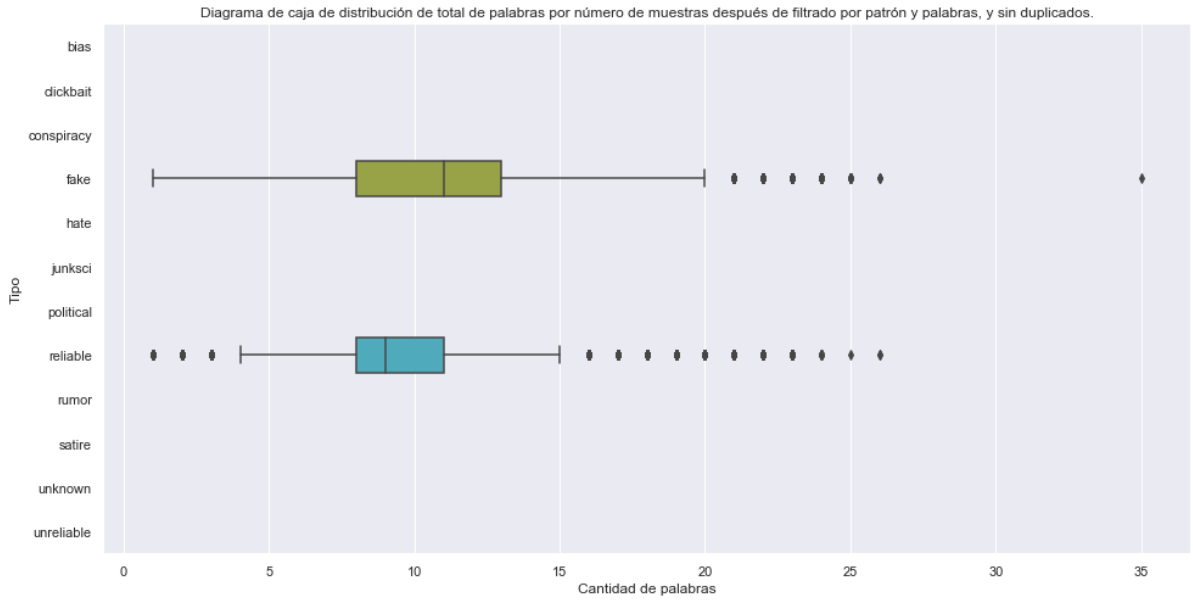


Figura 20. Diagrama de caja de distribución de total de palabras por número de muestras después de filtrado por patrón y sin duplicados. Fuente: Elaboración propia

Al analizar la distribución del total de palabras (Figura 20. Diagrama de caja de distribución de total de palabras por número de muestras después de filtrado por patrón y sin duplicados. Fuente: Elaboración propia) se pudo observar que los títulos con número de palabras mayores a 20 representan valores atípicos. Así mismo, se observó (Tabla 5. Distribución de número de palabras de título. Fuente: Elaboración Propia) que los títulos con número de palabras mayores a 19 representan menos del 1% del total del conjunto de datos, por lo que se puede prescindir de las muestras correspondientes. Por otra parte, se observó también que los títulos con número de palabras menores a 4 palabras representan menos del 2.5% del total de muestras del conjunto de datos. Considerando y asumiendo que las muestras con títulos con número de palabras menores a 4 carecen de valor para nuestro análisis debido a que no contienen un significado representativo, se optó por prescindir de ellos. En total, se descartó menos de 3.5% del total de muestras. El resultado del descarte puede ser observado en la Figura 21. Distribución de total de palabras por número de muestras después de filtrado

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

por patrón, sin duplicados, menos a 3.5% de total de muestras descartadas. Fuente: Elaboración propia.

Tabla 5. Distribución de número de palabras de título. Fuente: Elaboración Propia

	Número de palabras de título
Conteo total	310182
Promedio	9.850904
Desviación Estándar	3.425706
Valor mínimo	1
1% de total de muestras	3
2.5% de total de muestras	4
50% de total de muestras	10
97.5% de total de muestras	17
99% de total de muestras	19
Valor máximo	35

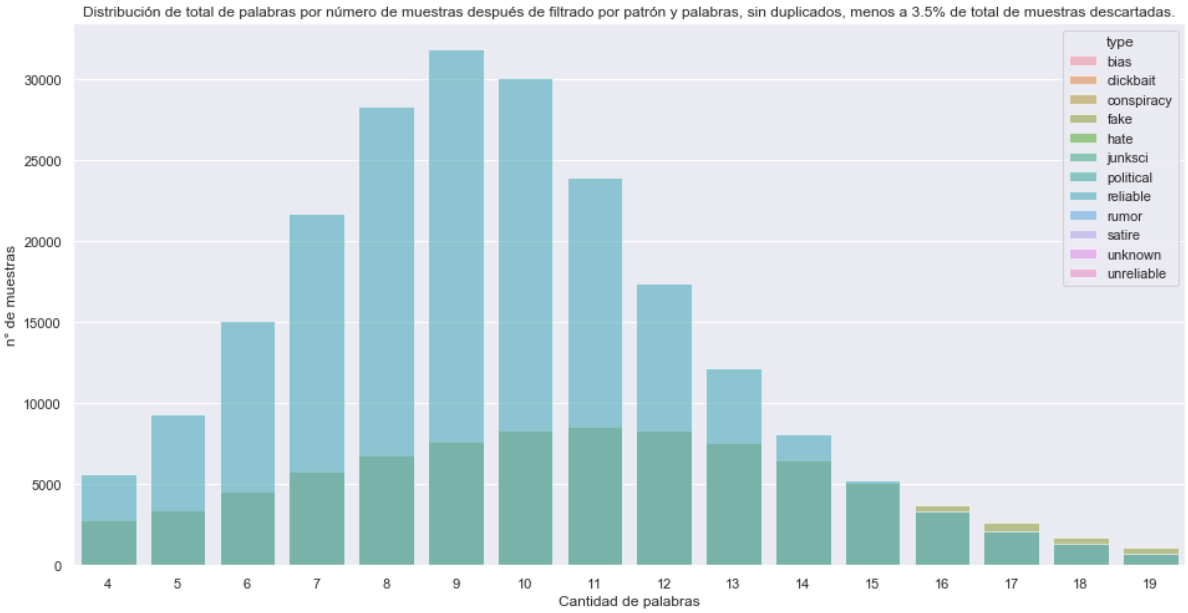


Figura 21. Distribución de total de palabras por número de muestras después de filtrado por patrón, sin duplicados, menos a 3.5% de total de muestras descartadas. Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

Al terminar el filtrado, se procedió a explorar nuevamente los n-gramas. El resultado del proceso de filtrado puede ser observado en:

- Figura 22.1-grama, 2-grama y 3-grama para fuentes ‘reliable’ después de filtrado. Fuente: Elaboración propia.

- Figura 23. 1-grama, 2-grama y 3-grama para fuentes ‘fake’ después de filtrado. Fuente: Elaboración propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

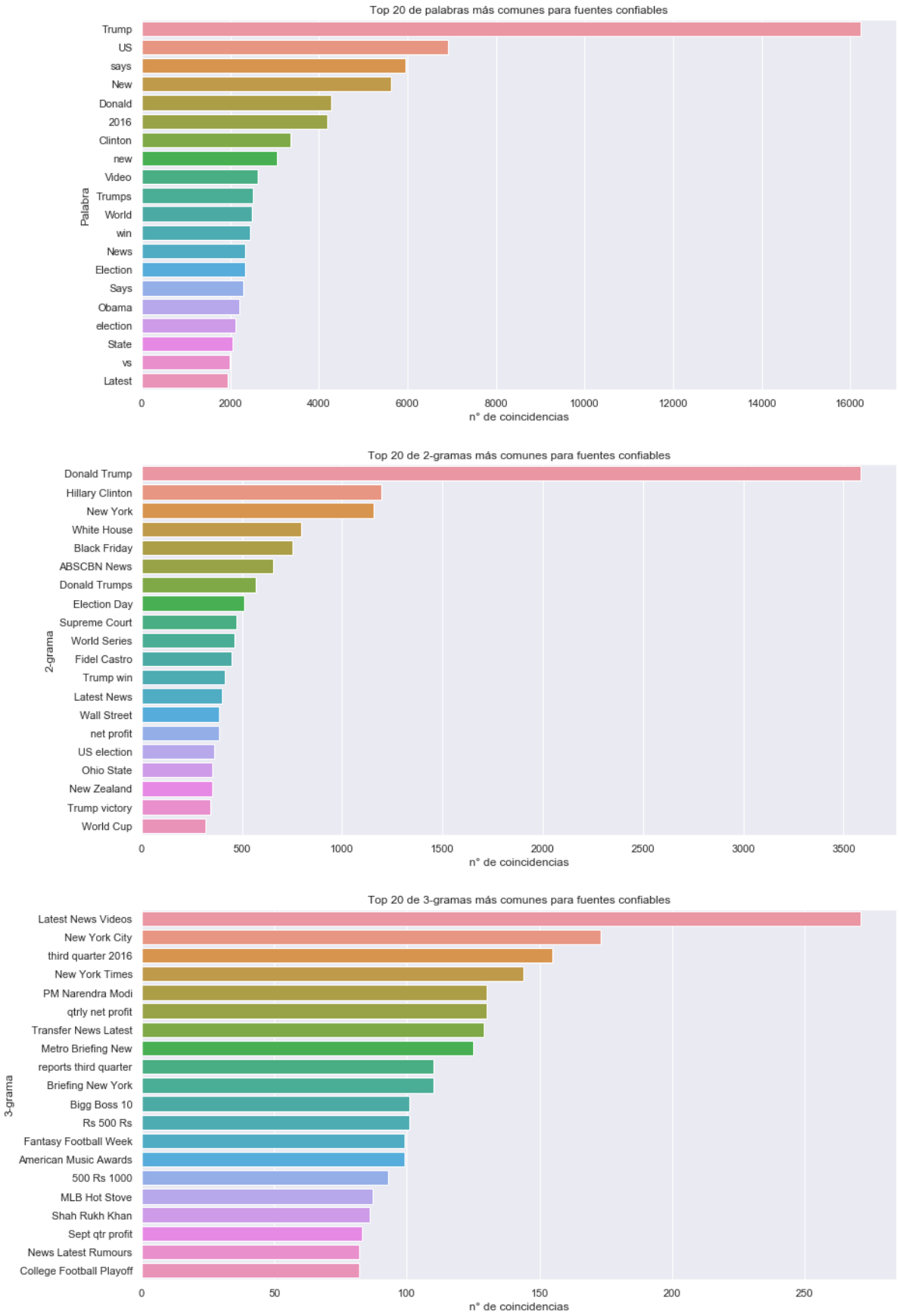


Figura 22.1-grama, 2-grama y 3-grama para fuentes 'reliable' después de filtrado. Fuente: Elaboración propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

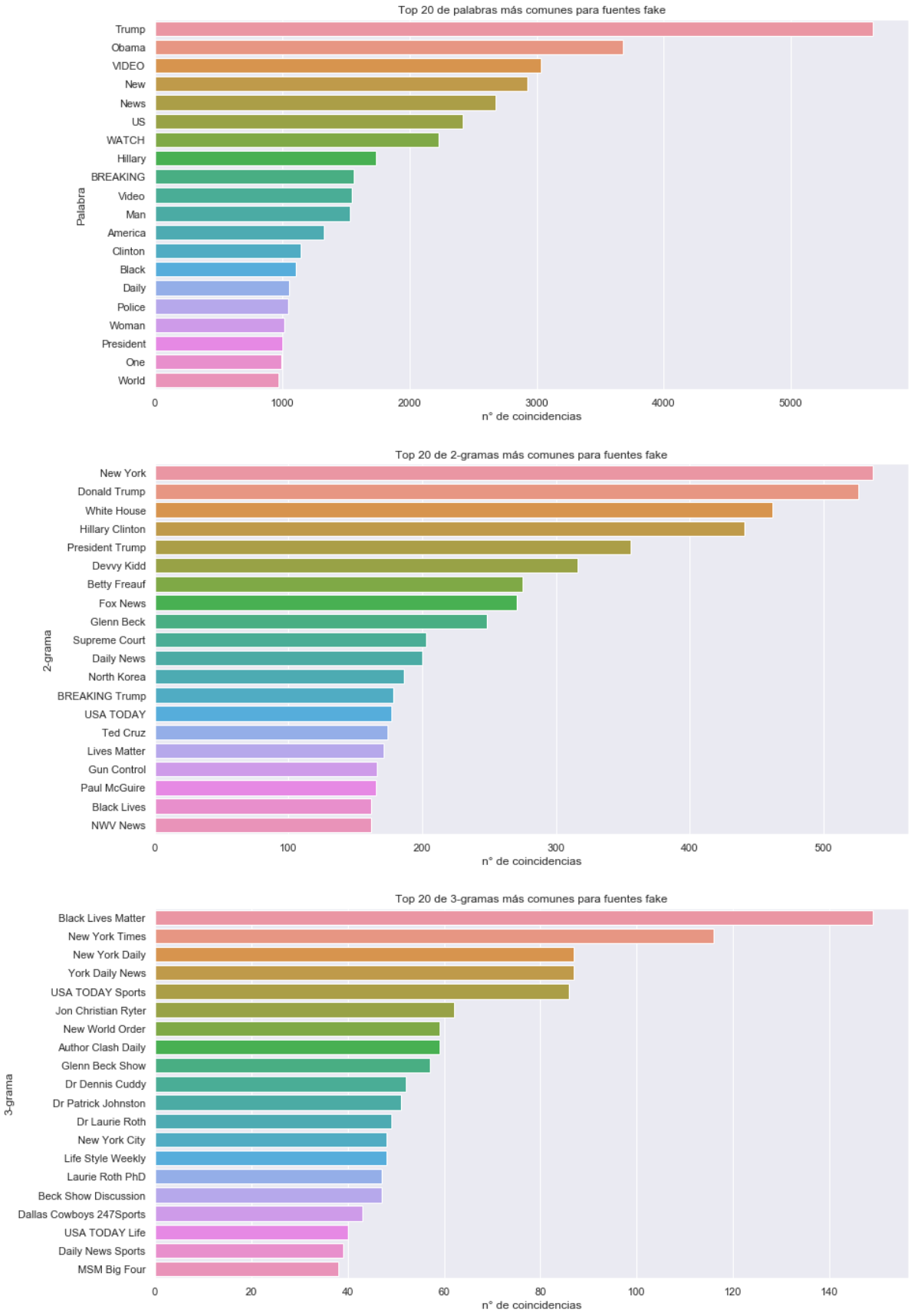


Figura 23. 1-grama, 2-grama y 3-grama para fuentes 'fake' después de filtrado. Fuente: Elaboración propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.1.4. Muestreo

Después de haber realizado el análisis y limpieza del conjunto de datos inicial, se procedió a realizar el muestreo del conjunto de datos filtrado. Antes del muestreo, se tuvo 301088 registros, de los cuales 216604 correspondían a fuentes confiables y 84484 correspondían a fuentes falsas. Se realizó un muestreo de 15000 muestras tanto para fuentes confiables como falsas, teniendo finalmente un conjunto de datos consistente en 30000 muestras. Las razones para optar por este muestreo fueron las siguientes:

- Desbalance entre tipos de fuentes: Había un claro desbalance con respecto al tipo de fuente ya que sólo el 28% del conjunto de datos provenía de fuentes falsas, mientras que el restante 72% provenía de fuentes confiables. Este desbalance resulta perjudicial al momento de entrenar un modelo clasificador, ya que las muestras utilizadas para el entrenamiento tendrían un sesgo por parte de la mayoría de las fuentes confiables.

- Exceso de muestras para traducción: Realizar la traducción automática en 301 088 muestras tiene un mayor costo que realizar la traducción con solo 30 000 muestras. De acuerdo a la documentación de Cloud Translation (Cloud Translation - Google Cloud s. f.), el costo se da por número de caracteres. El conjunto de datos sin muestrear consistió en 19 278 306 total de caracteres, mientras que un submuestreo de 30000 muestras consistiría en aproximadamente 2 051 217 caracteres. De esta manera, realizar la traducción en todo el conjunto de datos podría costar hasta 9 veces más que realizar la traducción en una muestra de menor tamaño.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

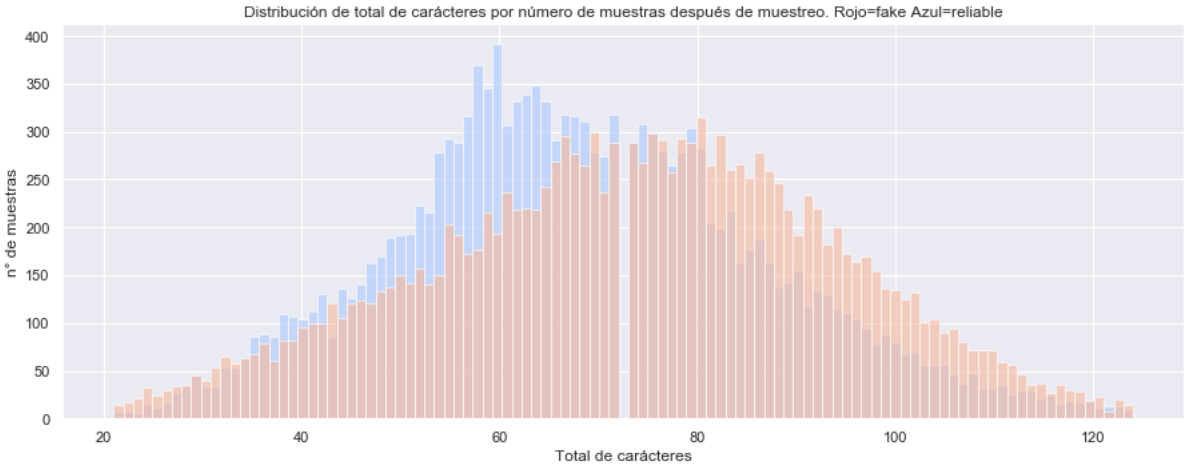


Figura 24. Distribución de total de caracteres por número de muestras después de muestreo.

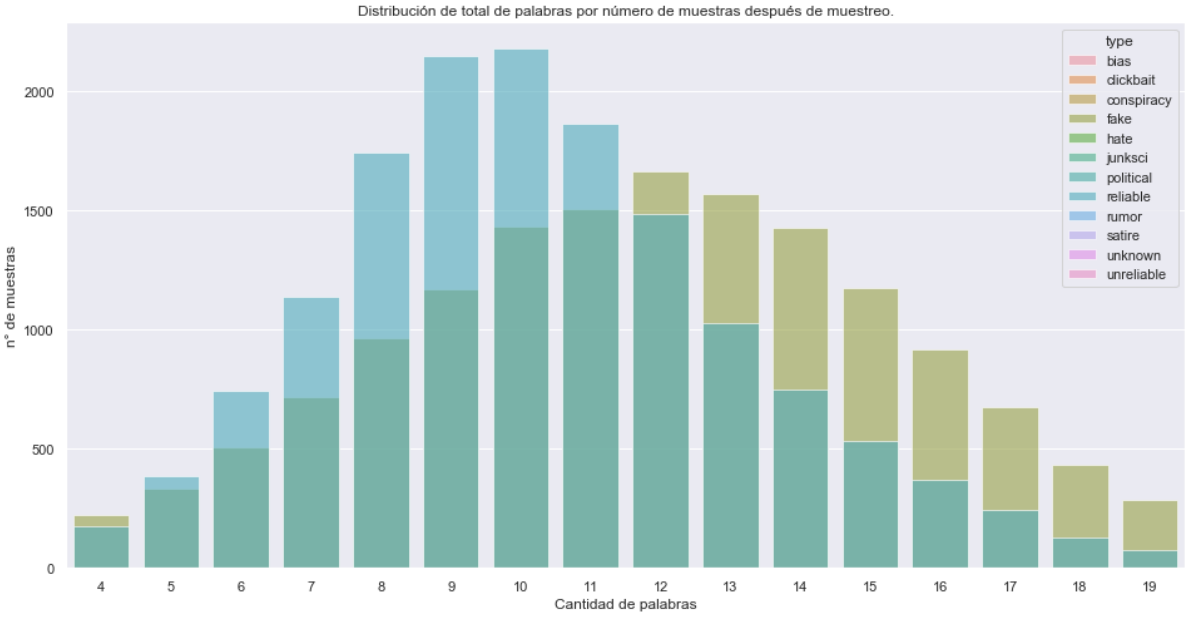


Figura 25. Distribución de total de palabras por número de muestras después de muestreo

3.2.Ejecución y validación de traducción automática

En este capítulo se procede a realizar la traducción y validación del conjunto de datos muestreado. Inicialmente, se procederá de acuerdo con la metodología de *backtranslation*, mediante la cual realizaremos la traducción del conjunto de datos del español al inglés, y luego nuevamente del inglés al español. Se realizará el cálculo de la métrica BLEU (Papineni et al.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

2002) y METEOR (Banerjee y Lavie 2005) para la evaluación de la traducción. El resultado de la métrica BLEU es presentado en este trabajo únicamente con fines demostrativos, mientras que el resultado de la métrica METEOR se utilizará para filtrar el 95% de los datos con la mayor calificación de traducción. Se realizará dicho filtrado para asegurar la calidad de los datos utilizados para el entrenamiento del modelo de clasificación en las etapas posteriores.

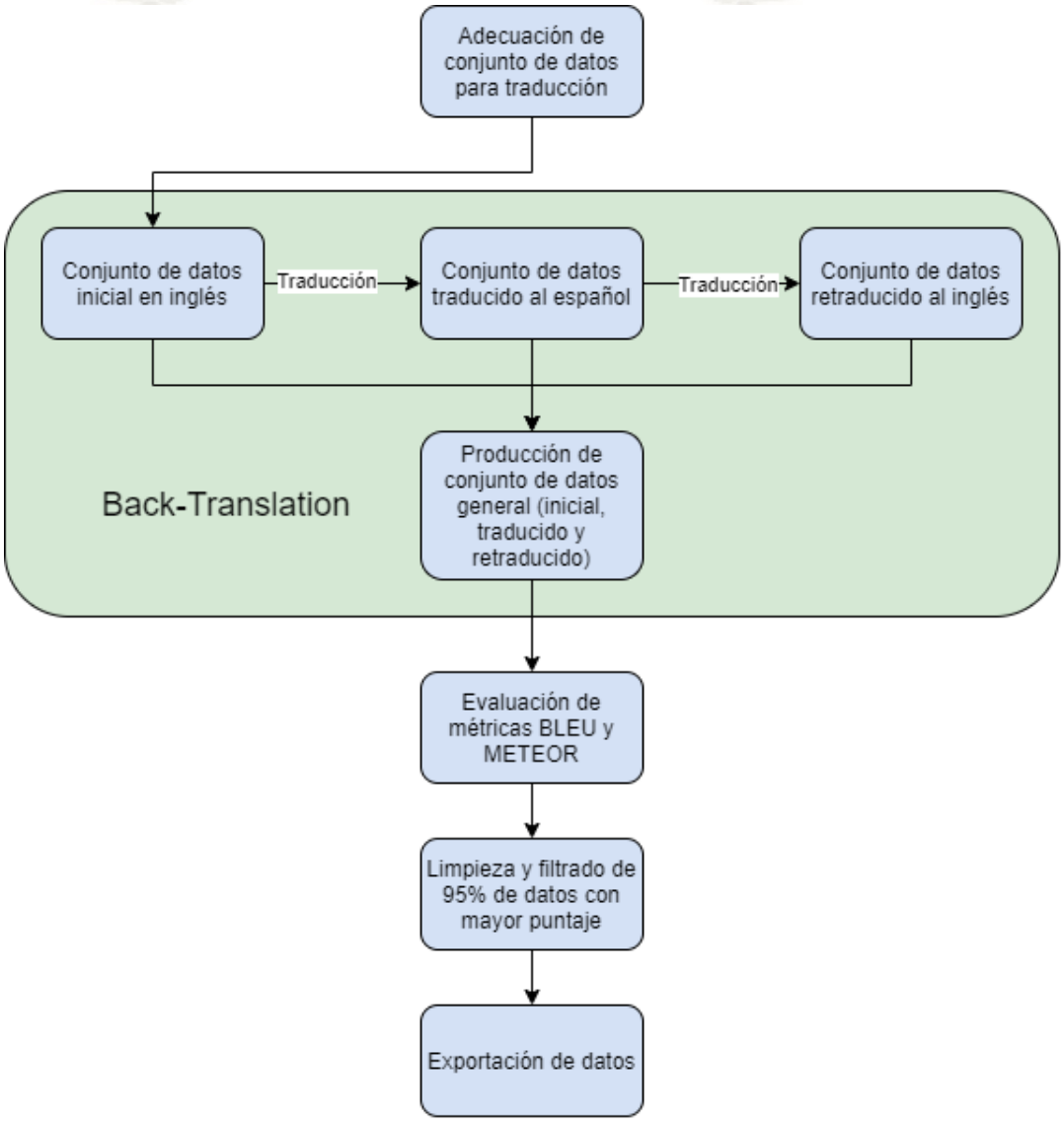


Figura 26. Diagrama de flujo de etapa de traducción automática. Fuente: Elaboración propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

3.2.1. Adecuación y traducción

La traducción inicial se realizó mediante la traducción por lotes de un modelo de traducción automática neuronal implementado en la versión avanzada de Cloud Translation (Google Cloud s. f.). Se hizo uso de la librería cliente para Python Cloud Translation API (LLC s. f.) para la solicitud correspondiente.

La traducción por lotes requirió del uso de un archivo en TSV de dos columnas conforme con el estándar RFC 2048 (Shafranovich s. f.). La primera columna de dicho archivo especificó un identificador único para cada muestra, mientras que la segunda columna indicó el texto a traducir. De acuerdo con el estándar, la distinción entre ambas columnas se dio mediante una tabulación horizontal que puede ser especificada mediante el carácter especial \t, mientras que la distinción entre diferentes muestras o filas se dio mediante el fin de línea. Además, los caracteres de todo el conjunto no debieron poseer ningún carácter especial, como por ejemplo \r o \n. Se filtró una de las muestras que no poseía las condiciones necesarias para ser adaptada a este formato y estándar, teniendo un total de 29999 muestras traducidas.

Una vez realizada la adecuación del conjunto de datos, se construyó la función correspondiente para realizar la solicitud de la traducción (ANEXOS. Función de solicitud de traducción).

Luego de realizar la traducción inicial del inglés al español, se realizó otra solicitud para la traducción del español al inglés de acuerdo con el método del *backtranslation*,

Terminada esta etapa, se procedió a la etapa de validación de traducción.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.2.2. Validación y traducción

3.2.2.1.Construcción del conjunto de datos

Se construyó un conjunto de datos en donde se concatenó la traducción y re-traducción al conjunto de datos muestreado. En este nuevo conjunto de datos, al que se le denominará conjunto de datos traducido, se tuvieron las columnas: *domain*, *titulo*, *traducción*, *retraducción*, *tipo*, *BLEU* y *METEOR*. La columna *domain* se refiere al dominio de donde fue extraída la muestra, las columnas de *traducción* y *retraducción* contienen los resultados de los procesos de traducción y retraducción, la columna *tipo* contiene los valores de *falsa* y *confiable* para referirse a las muestras de tipo *fake new* o *reliable*, y las columnas *BLEU* y *METEOR* contienen los resultados del cálculo de las métricas con el mismo nombre aplicadas a la muestra respectiva.

3.2.2.2.Especificaciones para métricas BLEU y METEOR

Para el cálculo de las métricas BLEU y METEOR se hizo uso del paquete *translate* de la librería NLTK (Bird et al. 2009).

Para el cálculo de la métrica BLEU, se hizo uso de la técnica de suavizado 4 (Chen y Cherry 2014:363) debido a que presenta mejores resultados para traducciones a nivel de oración. El cálculo de la métrica BLEU se realiza únicamente con fines demostrativos.

Para el cálculo de la métrica METEOR, se utilizaron las constantes $\alpha = 0.81, \beta = 0.83, \gamma = 0.28$ debido a que presentan mejores resultados para traducciones en idioma inglés (Lavie y Agarwal 2007:3). En cuanto al módulo de *stemming*, se hizo uso del algoritmo de

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Snowball (Porter 2001) para el cálculo de raíces por ser uno de los algoritmos más utilizados, además de que fue construido en base al algoritmo de *Porter* (Porter 1980) que es un algoritmo pionero en el campo. El cálculo de la métrica METEOR se realizó principalmente con fines de filtrado, con el objetivo de obtener el 95% de traducciones de mayor calidad.

3.2.2.3.Análisis de resultados de métricas BLEU y METEOR

Después del cálculo de las métricas BLEU y METEOR en el conjunto de datos traducido, se procedió a analizar sus resultados.

Al analizar el diagrama de dispersión de los resultados de las métricas BLEU y METEOR (Figura 27. Diagrama de dispersión de métricas BLEU y METEOR. Fuente: Elaboración Propia.), se observó la tendencia a una relación directamente proporcional de ambas métricas, es decir, mientras mayor fue la métrica BLEU, mayor era la métrica METEOR. Sin embargo, se observaron dos zonas que podrían contener resultados atípicos. En primer lugar, se observaron muestras para las que el valor de la métrica METEOR era de 0, mientras que el valor de la métrica de BLEU, si bien tenía un valor menor a 0.4, era distinto de 0. Al analizar las muestras pertenecientes a dicha zona, se pudo observar que el título de dichas muestras se encontraba en un idioma distinto del inglés. Esto causaría valores atípicos en la métrica METEOR debido a que sus módulos de coincidencia de sinónimos y de raíces de palabras son dependientes del lenguaje, mientras que la métrica BLEU, al ser independiente del lenguaje, pudo brindar valores distintos de 0. La segunda zona de valores atípicos fue aquella para la que el resultado de la métrica METEOR se encontraba entre los valores de 0.4 y 1, mientras que el resultado de BLEU fue de menor a 0.4. Al analizar dichas muestras, se observó que su título inicial y su título retraducido diferían en el uso de mayúsculas. Por ejemplo, en una muestra determinada, mientras que el título inicial se encontraba escrito enteramente en mayúsculas, su

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

título retraducido se encontraba únicamente con la primera letra en mayúscula. Debido a que BLEU se basa en el cálculo de una precisión de coincidencia de palabras por un único módulo de exactitud, el uso de mayúsculas o ausencia de estas puede evitar encontrar dichas coincidencias. Esto explicaría la baja calificación para esta métrica. En caso se usase la métrica BLEU, como parte del módulo NLTK, para la evaluación de traducciones en futuros trabajos, se recomendaría implementar una etapa de preprocesamiento para evitar la diferencia en mayúsculas. Con fines demostrativos, se implementó una etapa de preprocesamiento en el presente trabajo que consistió en transformar todos los títulos y su re-traducción a minúsculas previo al cálculo de la métrica BLEU. Como resultado, se observó la ausencia de la zona de valores atípicos mencionada (Figura 28. Diagrama de dispersión con preprocesamiento. Fuente: Elaboración Propia.).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

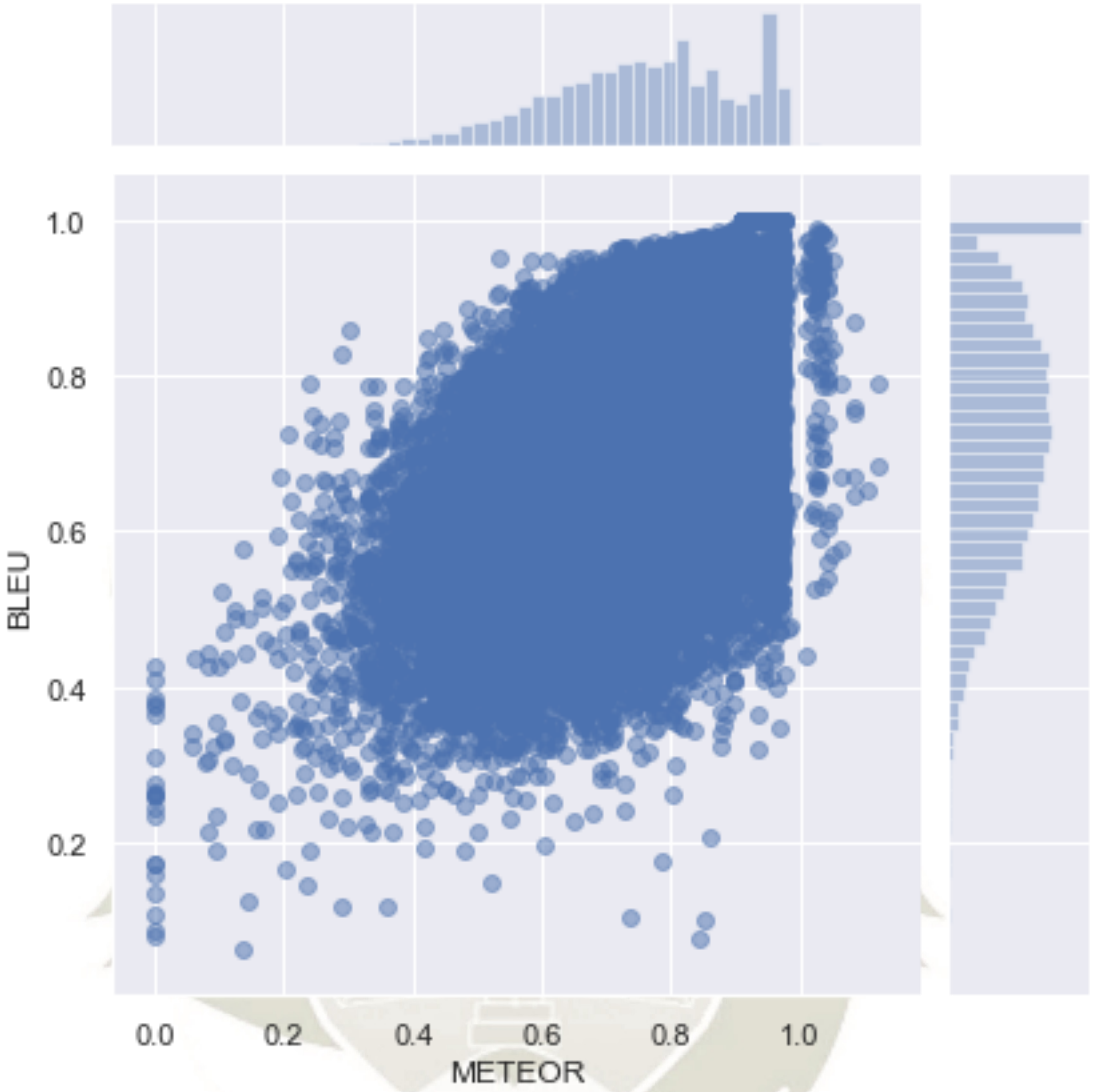


Figura 27. Diagrama de dispersión de métricas BLEU y METEOR. Fuente: Elaboración Propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

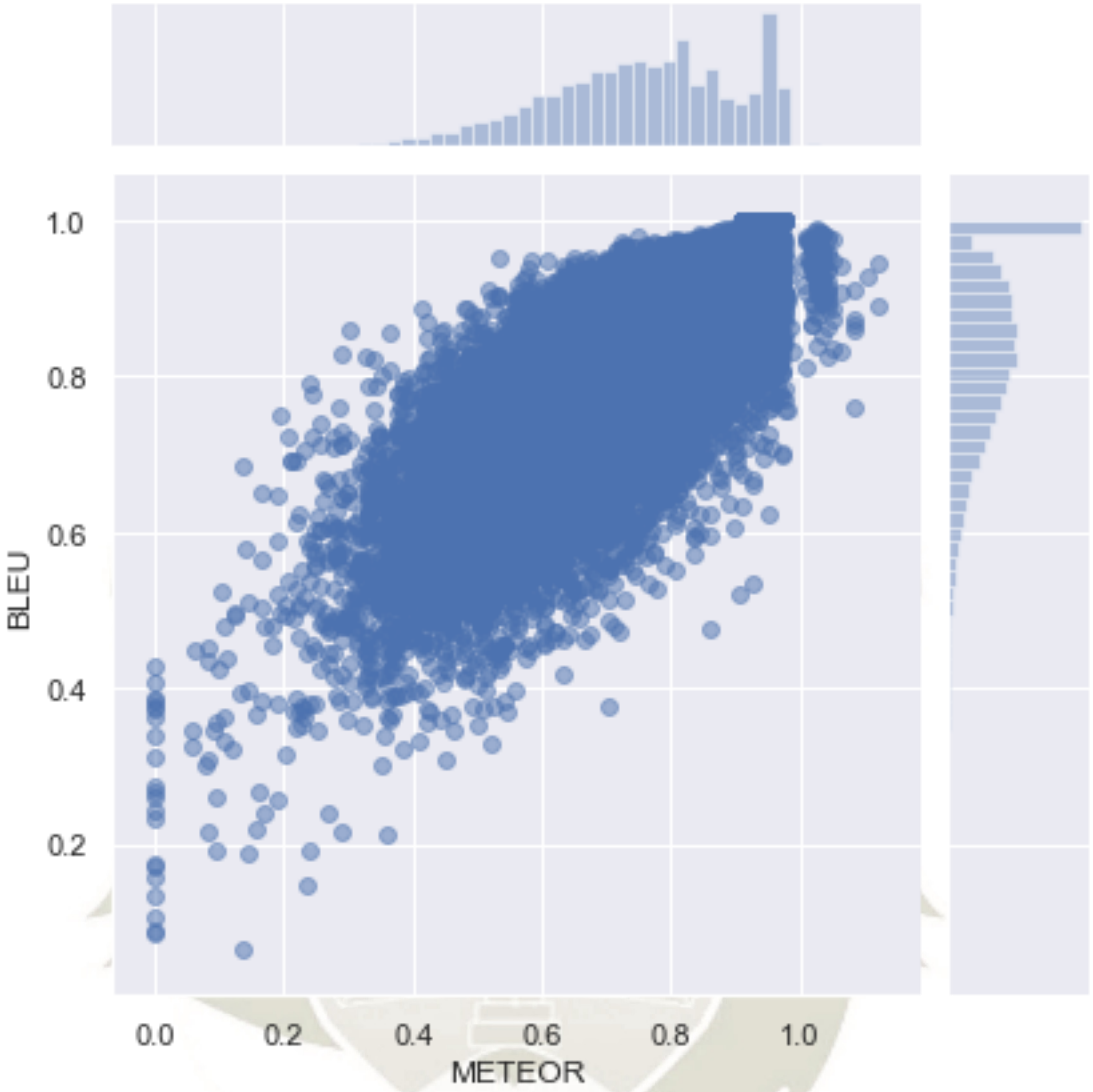


Figura 28. Diagrama de dispersión con preprocesamiento. Fuente: Elaboración Propia.

3.2.2.4.Exportando resultados

Posteriormente, se procedió a analizar la distribución de los resultados de las métricas (Tabla 6. Distribución de métricas BLEU y METEOR. Fuente: Elaboración Propia.). Se observó que el 95% del total de las muestras poseía valores para la métrica METEOR mayores a 0.480007656, lo que equivale a 28511 muestras. Se exportó el conjunto de datos correspondientes a dicho 95%, conservando el índice y las columnas de *tipo* y *traducción*, la última de las cuales renombrada como *título*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Tabla 6. Distribución de métricas BLEU y METEOR. Fuente: Elaboración Propia.

	BLEU	METEOR
Total de muestras	29999	29999
Promedio	0.726638523	0.749068143
Desviación Estándar	0.158437594	0.153331058
Mínimo	0.061352253	0
1% de muestras	0.360611165	0.340909602
5%	0.460624745	0.480007656
10%	0.513657637	0.544421203
25%	0.611949325	0.650532011
50%	0.732525448	0.761716888
75%	0.84751194	0.864661115
Máximo	1	1.121151702

3.3.Extracción de características

En el presente capítulo se presenta el procedimiento llevado a cabo para la extracción de características utilizadas más adelante en el entrenamiento de los algoritmos de Aprendizaje Automático.

Según Horne y Adali (Horne y Adali 2017), las características relevantes a *fake news* pueden ser categorizadas en tres grupos: complejas, estilísticas y psicológicas. El presente capítulo inicia con la extracción de los denominados vectores de palabras, que codifican características semánticas y sintácticas de los textos objeto de estudio, y que encierran las tres categorías mencionadas. Después, se continúa con la extracción de características complejas y estilísticas por medio del proceso de tokenización. Por último, se procede a extraer características psicológicas mediante el análisis de sentimientos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

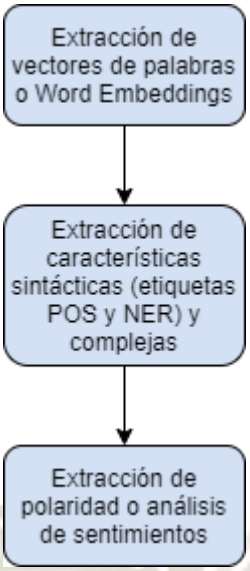


Figura 29. Diagrama de flujo de extracción de características. Fuente: Elaboración propia.

3.3.1. Word Embeddings

El presente proyecto hizo uso de *Word Embeddings* para la obtención de vectores de palabras que codifican características semánticas y sintácticas de las muestras. Estos vectores de palabras son usados más tarde como características numéricas en el entrenamiento de algoritmos de Aprendizaje Automático.

Se utilizó un modelo ya entrenado de *Word Embeddings* para la obtención de vectores de palabras. Se utilizó dicho modelo ya que sus características lo hacían ideal para la aplicación considerada. El presente proyecto no contempló el entrenamiento de *Word Embeddings* debido a que dicho proceso habría requerido de mayores recursos humanos y computacionales innecesarios.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.3.1.1.Descripción de Modelo

El modelo utilizado (Explosion [2017] 2020) lleva como nombre *es_core_news_lg* y es un modelo disponible como parte de la librería *Spacy* (Explosion [2014] 2020). Dicho modelo fue entrenado usando *fastText* (Bojanowski et al. 2017) y *CBOW*. El uso de *fastText* es ideal para nuestra aplicación ya que propone la representación de vectores de palabras como la suma de vectores de n-gramas de caracteres (Bojanowski et al. 2017). Este hecho hace a sus vectores de palabras más robustos frente a errores o informalidades gramaticales que no formaron parte del vocabulario de entrenamiento, pero que son comunes en el vocabulario del corpus de este proyecto.

Por otra parte, los vectores de palabras fueron entrenados usando artículos de *Wikipedia* (Wikipedia 2020) y de *OSCAR* (Suárez, Sagot, y Romary 2019). *OSCAR* es un corpus multilingüe que contiene más de 25 mil millones de palabras en español. El hecho de contener esta cantidad de palabras nos brinda vectores de palabras para uso en aplicaciones generales, como es el caso de este proyecto. Por último, el modelo usado posee 500 mil vectores únicos de palabras con 300 dimensiones cada uno.

3.3.2. Características complejas y estilísticas

3.3.2.1.Descripción de características

Se procedió a extraer las características complejas y estilísticas expresadas en la Tabla 7. Características complejas y estilísticas. Se denominan características estilísticas a aquellas características que contienen información gramatical y que nos permiten comprender el estilo del texto. Por ejemplo, consideramos como estas al número de adverbios, sustantivos o verbos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

en la muestra. Se denomina características complejas a aquellas características que contienen información sobre la complejidad de la muestra a nivel de palabra. Esto es, el número de palabras totales, el tamaño promedio de palabra por muestra, o, por ejemplo, una combinación de una característica compleja con una estilística, como, por ejemplo, la relación entre el número de verbos y el número total de palabras. Si bien es cierto la mayoría de las características mencionadas son comprensibles, dos de estas necesitan de mayor explicación. La primera, *ratio_ner_**, en donde * se refiere a *per*, *loc*, u *org*, se refiere al número de entidades encontradas, como nombres de personas, ubicaciones u organizaciones respectivamente, divididas por el número total de palabras. La segunda, *ratio_pos_diversity*, se refiere a la relación del número de etiquetas POS únicos entre el número total de palabras por muestra. Las etiquetas POS, *parte del discurso* por sus siglas en inglés, hace referencia a cualquier tipo de etiqueta gramatical identificada, como, por ejemplo, la etiqueta de adjetivo, de sustantivo, o de adverbio.

Para la identificación de características a analizar se utilizaron 3 principales fuentes que tratan características relevantes para el estudio de *fake news* (Abonizio et al. 2020; Horne y Adali 2017; Lynch y Vogel 2018).

Tabla 7. Características complejas y estilísticas.

Nombre	Descripción	Referencia
number_words	Número de palabras	(Horne y Adali 2017)
avg_word_size	Tamaño promedio de palabras	(Lynch y Vogel 2018)
ratio_adj	Relación de adjetivos entre número de palabras	(Horne y Adali 2017)
ratio_adp	Relación de adposiciones entre número de palabras	(Horne y Adali 2017)
ratio_adv	Relación de adverbios entre número de palabras	(Horne y Adali 2017)
ratio_det	Relación de determinantes entre número de palabras	(Horne y Adali 2017)

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Nombre	Descripción	Referencia
ratio_noun	Relación de sustantivos entre número de palabras	(Horne y Adali 2017)
ratio_propn	Relación de nombres propios entre número de palabras	(Horne y Adali 2017)
ratio_pron	Relación de pronombres entre número de palabras	(Horne y Adali 2017)
ratio_verb	Relación de verbos entre número de palabras	(Horne y Adali 2017)
ratio_ner_per	Relación de número de entidades de personas entre número de palabras	Propuesto
ratio_ner_loc	Relación de número de entidades de ubicación entre número de palabras	Propuesto
ratio_ner_org	Relación de número de entidades de organizaciones entre número de palabras	Propuesto
ratio_pos_diversity	Relación de número de etiquetas de POS únicos identificados entre número de palabras	(Abonizio et al. 2020)

3.3.2.2.Descripción de modelo utilizado

Para la extracción de características mencionadas se hizo nuevamente uso del modelo *es_core_news_lg* (Explosion [2017] 2020) como parte de la librería *Spacy* (Explosion [2014] 2020). Este modelo implementa no sólo vectores de palabras, sino también un módulo de procesamiento de texto que permite el reconocimiento de etiquetas POS, que nos permiten identificar etiquetas gramaticales en palabras o caracteres, y de etiquetas NER, *reconocimiento de entidades nombradas* por sus siglas en inglés, que nos permite identificar etiquetas de nombres de entidades, como, por ejemplo, etiquetas de nombres de personas, ubicaciones u organizaciones.

Si bien es cierto, en cuanto a vectores de palabras, *es_core_news_lg* (Explosion [2017] 2020) tiene una aplicación general, en cuanto a reconocimiento de etiquetas POS y NER su aplicación es más específica. Dicho modelo consta de una red neuronal utilizada para el reconocimiento de dichas etiquetas y entrenada con artículos de Wikipedia (Nothman et al.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

2013) y datos del corpus para español de AnCora (Taulé, Martí, y Recasens 2008). Estos dos corpus son de uso específico e ideal para nuestro proyecto. Por una parte, el corpus de Wikipedia (Nothman et al. 2013) consta de palabras de artículos informativos generales, mientras que el corpus de AnCora (Taulé et al. 2008) consta de un total de 500 000 palabras, de las cuales 425 000 palabras pertenecen a los medios de noticias en español *EFE* (Agencia EFE s. f.) y *El Periódico* (Periódico s. f.), y 75 000 pertenecen al corpus LEXESP (Sebastián-Gallés 2000), que es un corpus de más de 5 millones de palabras provenientes de fuentes narrativas, de divulgación científica, de ensayos, de prensa y de semanarios. El hecho de que los corpus utilizados para el entrenamiento del modelo tengan predominantemente artículos informativos y de medios de noticias hacen a este modelo ideal para aplicaciones relacionadas con la misma temática, como es el caso del reconocimiento de noticias falsas en este proyecto.

3.3.3. Características psicológicas: análisis de sentimientos

Para el análisis de sentimientos se utilizó la aproximación basada en el uso de un léxico de sentimientos. Dicha aproximación consiste en el uso de un vocabulario de palabras que contienen palabras cuya polaridad ya se encuentra asignada mediante un valor numérico. A partir de este, se calcula la polaridad de cada una de las palabras por muestra para luego calcular el promedio.

El léxico de palabras utilizado fue propuesto por Chen y Skiena (2014) e implementado como parte de la librería *polyglot* (ALRFOU [2014] 2020). La polaridad es asignada mediante los valores -1, 0 y 1 para referirse a polaridad negativa, neutra y positiva respectivamente.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

3.4.Entrenamiento, evaluación y comparación de modelos de Aprendizaje Automático

Inicialmente se realiza una exploración y análisis de las características extraídas con el fin de identificar y filtrar valores atípicos que pudiesen ser perjudiciales en el entrenamiento de los modelos de aprendizaje automático.

Se procede a entrenar un total de 16 modelos de aprendizaje divididos en 4 grupos de cuatro modelos. Cada grupo de modelos se basan en las características utilizadas para su entrenamiento: todos los modelos tienen como base el uso de *Word Embeddings*, y se diferencia un grupo de otro de otro de acuerdo con la inclusión o ausencia de las características sintácticas y complejas, y la polaridad. Por su parte, cada modelo de cada grupo se diferencia uno de otro de acuerdo con 4 modelos de clasificación usados para su entrenamiento. Por último, se comparan todos los modelos en base a métricas, y se selecciona el modelo a utilizar en la implementación.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

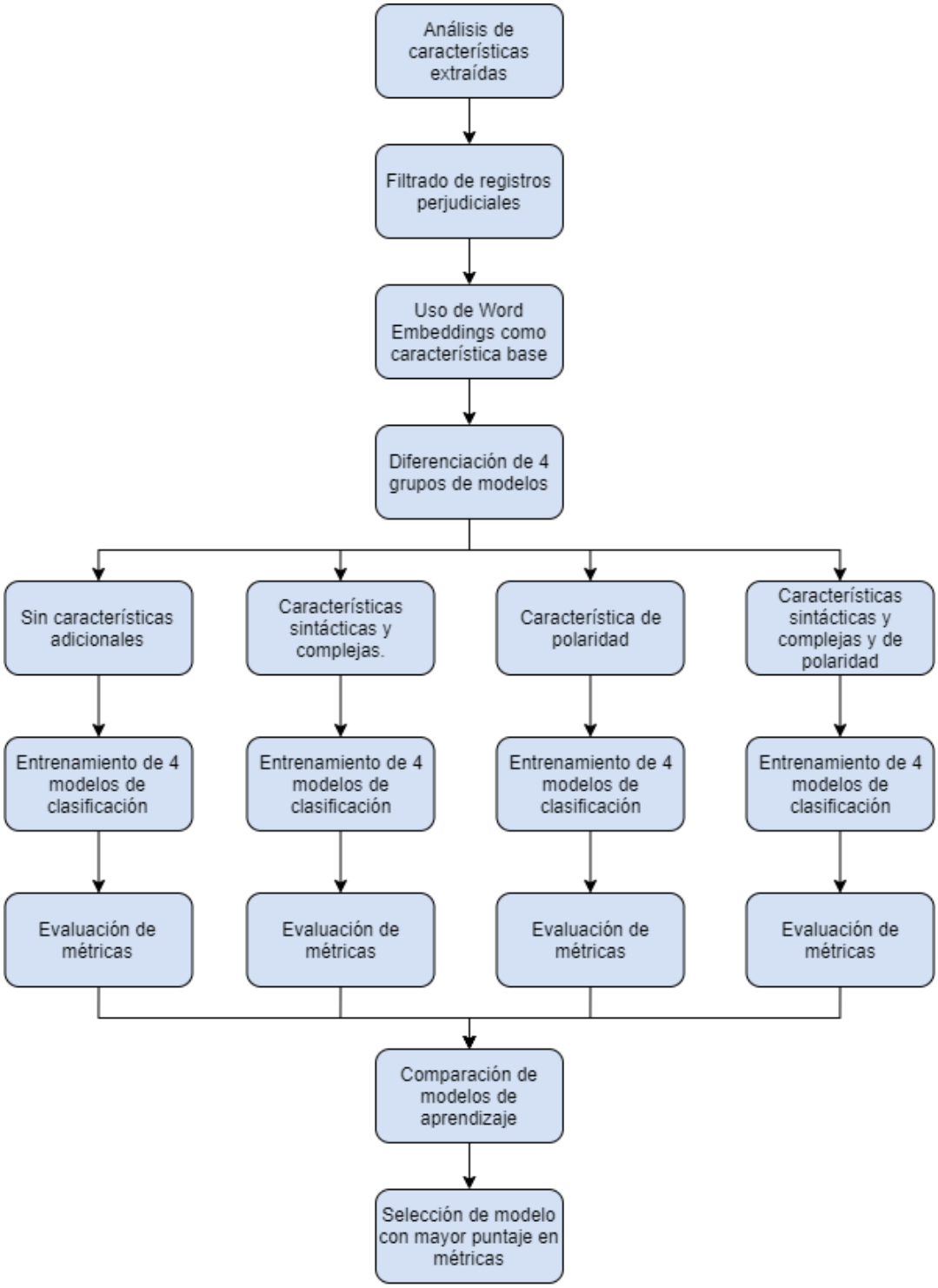


Figura 30. Diagrama de flujo de entrenamiento, diferenciación y elección de modelos. Fuente: Elaboración propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.4.1. Exploración y análisis de conjunto de datos

La extracción de características se dio mediante la función *spacyProcess* y *round_prox* (ANEXOS. Función de extracción de características). Inicialmente, se procedió a analizar los valores numéricos representativos de cada característica sin considerar los vectores de palabra o *word embeddings*. Los valores numéricos representativos fueron el conteo total de muestras, el valor promedio designado como *prom.*, la desviación estándar designada como *std.*, el valor mínimo y máximo designados como *min* y *max* respectivamente, y los cuantiles 0.50 y 0.99.

Tabla 8. Valores numéricos representativos de características extraídas. Parte 1.

	Number_words	avg_word_size	ratio_adj	ratio_adp	ratio_adv	ratio_det	ratio_noun
conteo	28500	28500	28500	28500	28500	28500	28500
prom.	13.418702	5.110026	0.068301	0.169808	0.020876	0.108231	0.178653
std.	4.307729	0.759453	0.069861	0.079361	0.039224	0.072084	0.089765
min	2.000000	2.500000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	13.000000	5.000000	0.060000	0.170000	0.000000	0.110000	0.180000
99%	24.000000	7.330000	0.290000	0.360000	0.170000	0.290000	0.400000
max	34.000000	10.500000	0.600000	0.500000	0.380000	0.440000	0.670000

Tabla 9. Valores numéricos representativos de características extraídas. Parte 2.

	Ratio_propn	ratio_pron	ratio_verb	ratio_ner_per	ratio_ner_loc
conteo	28500	28500	28500	28500	28500
prom.	0.202704	0.018915	0.075417	0.039585	0.027599
std.	0.148919	0.037854	0.063887	0.056354	0.048180
min	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.180000	0.000000	0.070000	0.000000	0.000000
99%	0.690000	0.150000	0.250000	0.220000	0.200000
max	1.000000	0.400000	0.600000	0.500000	0.430000

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Tabla 10. Valores numéricos representativos de características extraídas. Parte 3.

	Ratio_ner_org	ratio_pos_diversity	polarity
conteo	28500	28500	28500
prom.	0.024189	0.387031	-0.310588
std.	0.044506	0.122633	0.674693
min	0.000000	0.050000	-1.000000
50%	0.000000	0.370000	0.000000
99%	0.180000	0.780000	1.000000
max	0.430000	1.000000	1.000000

Se buscaron valores atípicos en las distribuciones, encontrándose que las características de *ratio_propn* y *ratio_pos_diversity*, correspondientes a las ratios de nombres propios y la diversidad de etiquetas POS, poseían valores atípicos de desviación estándar, específicamente, de 0.148919 y 0.122633 respectivamente. Un análisis más detallado de estas características nos permitió notar que la característica de *ratio_pos_diversity*, no presentó inconsistencias en sus datos; sin embargo, fue notable la inconsistencia en los datos de la característica *ratio_propn*, en donde los títulos traducidos no se encontraban en idioma español, sino más bien en otros idiomas desconocidos. Las palabras de dichos idiomas desconocidos, al no ser comunes para el idioma español, eran identificadas como nombres propios, por lo que se les otorgaba valores altos en la característica correspondiente. Con el objetivo de eliminar la inconsistencia del conjunto de datos, se procedió a eliminar aquellas muestras cuya característica *ratio_propn* fuese mayor al valor del cuantil 0.99, correspondiente al valor 0.69. Esto nos dejó con un total de 28208 muestras restantes. Las distribuciones de las características del conjunto de datos final se muestran en las Figuras 25, 26, 27, 28 y 29.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

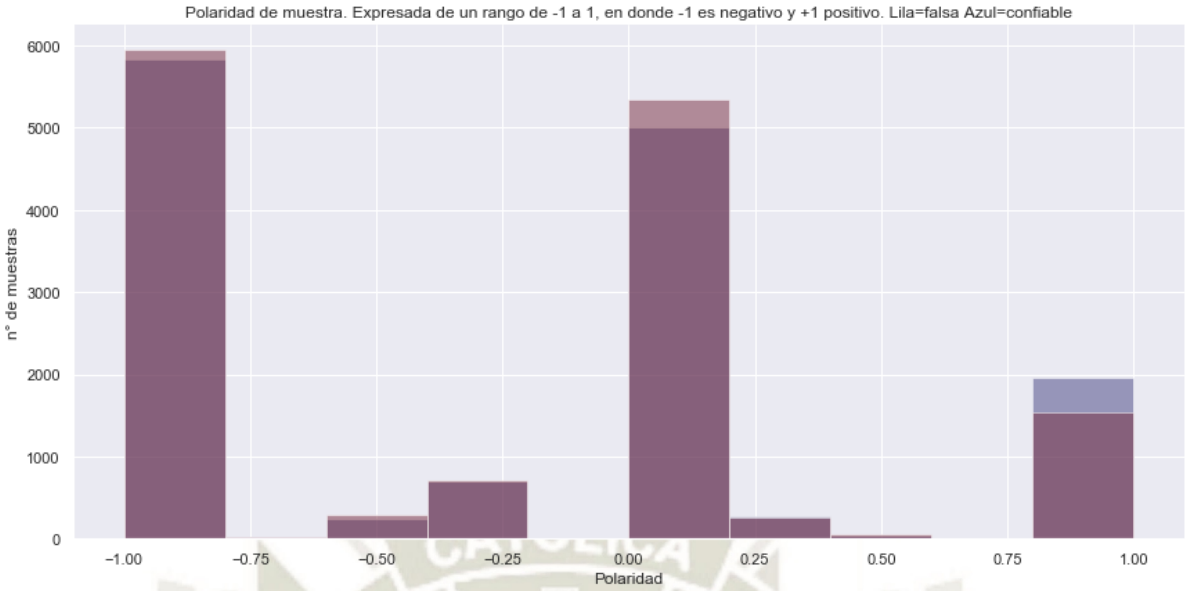


Figura 31. Polaridad de conjunto de datos. Fuente: Elaboración propia.

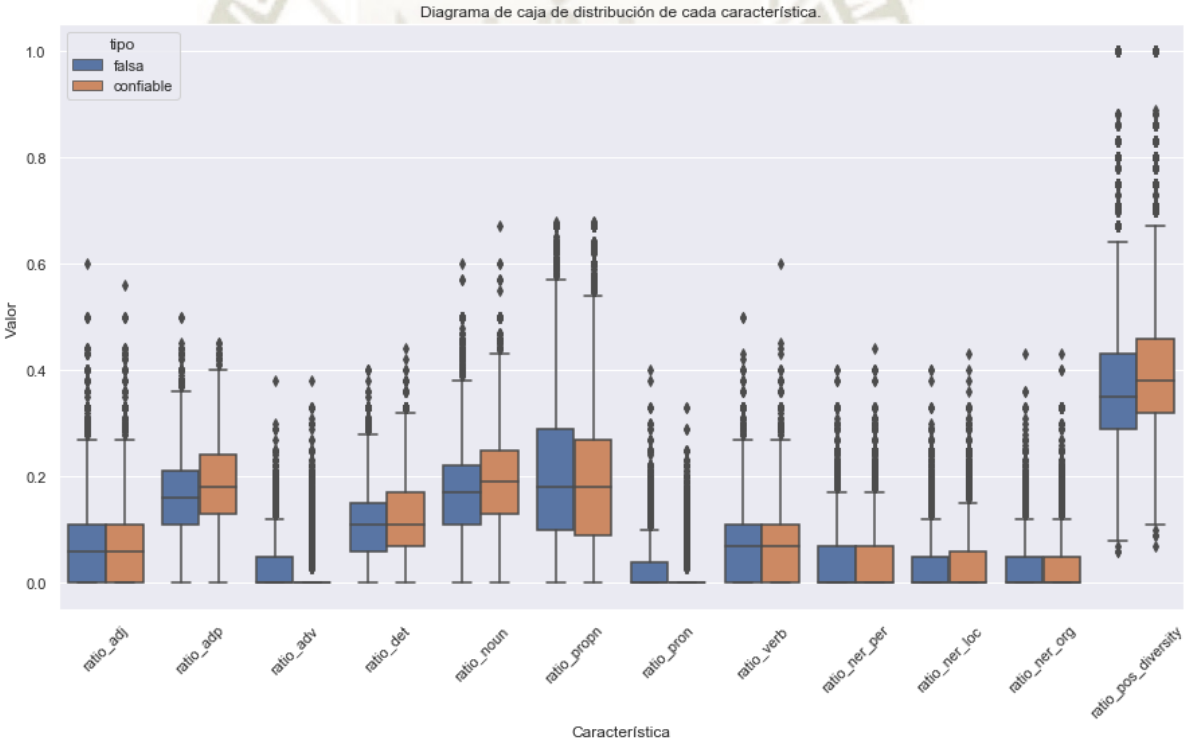


Figura 32. Diagrama de caja de distribución de características. Fuente: Elaboración propia.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

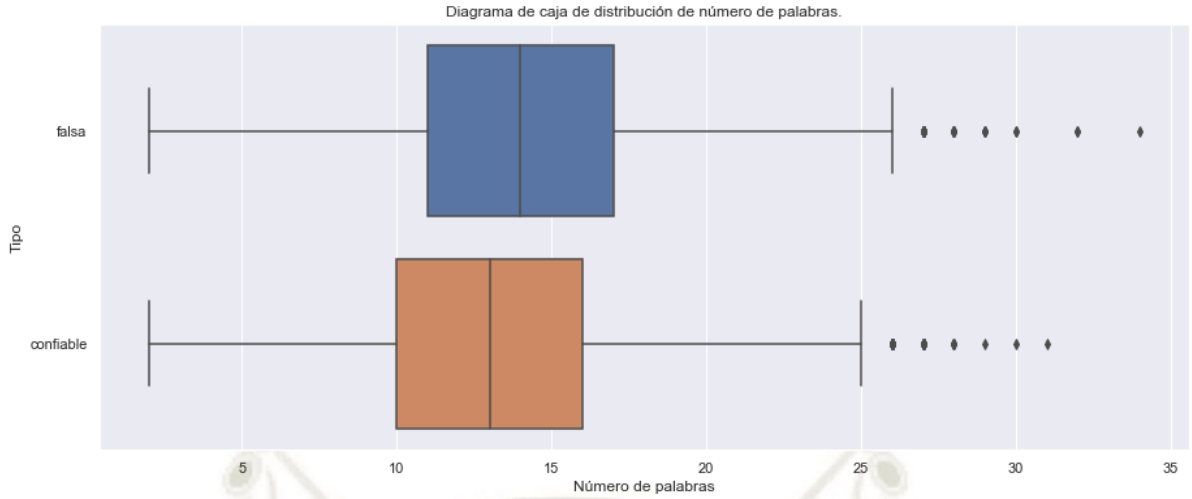


Figura 33. Diagrama de caja de distribución de número de palabras por tipo. Fuente: Elaboración propia

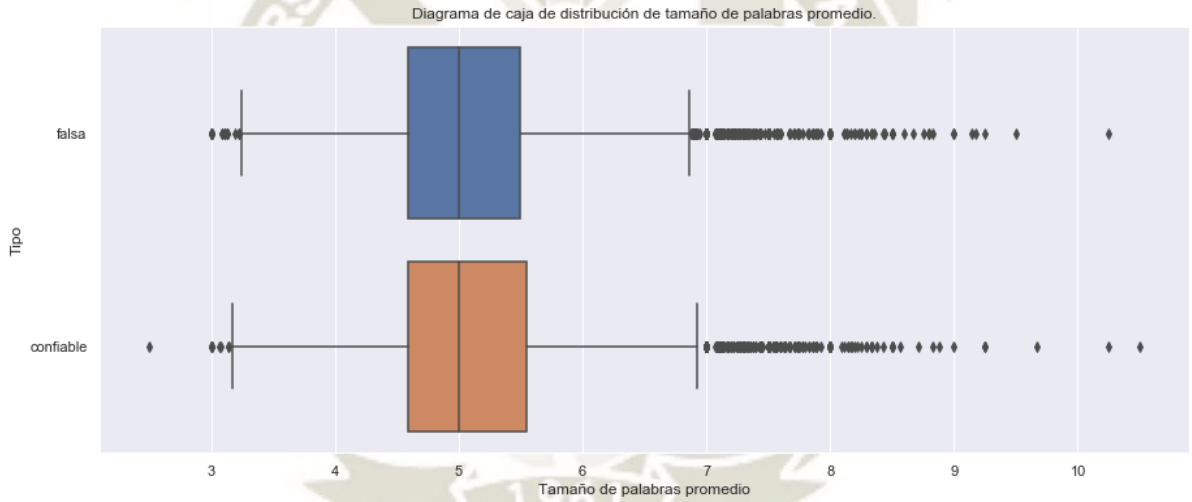


Figura 34. Diagrama de caja de distribución de tamaño de palabras promedio. Fuente: Elaboración propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

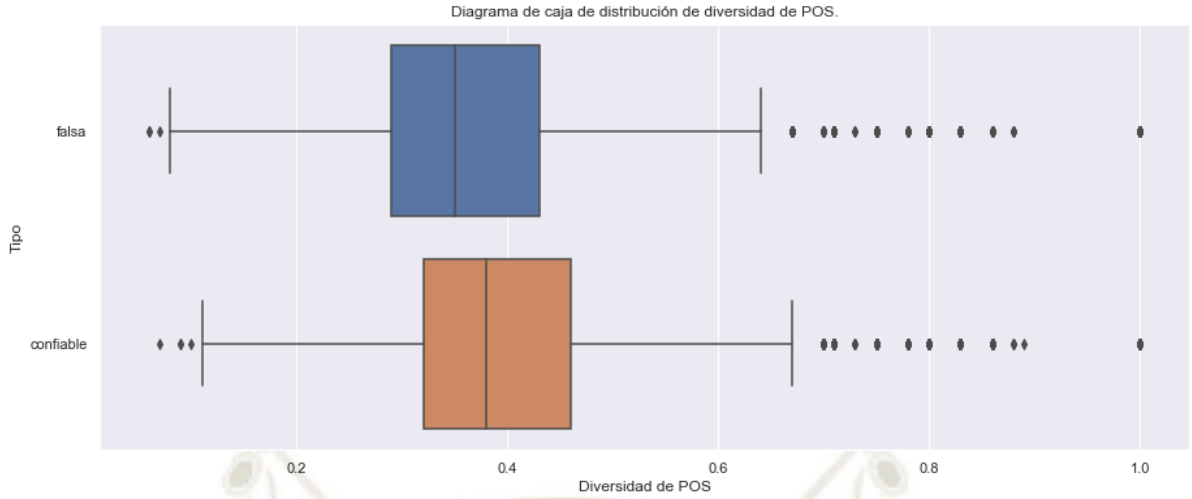


Figura 35. Diagrama de caja de distribución de diversidad de etiquetas POS. Fuente: Elaboración propia.

3.4.2. Implementación de modelos de aprendizaje automático

3.4.2.1.Primera Etapa: Preprocesamiento y Vectorización

La implementación de modelos de aprendizaje automático constó de dos etapas.

En la primera etapa, se implementó un preprocesador o vectorizador (ANEXOS. Vectorizador de palabras) que transformó el texto de cada muestra a arreglos numéricos, de modo que estos puedan ser utilizados por los modelos de clasificación. Dicho preprocesador posee como argumentos de tipo booleano a *pos_and_ner* y *sentiment*, cuyo valor verdadero permite incluir los valores de las etiquetas POS y NER (Tabla 7. Características complejas y estilísticas.) y la polaridad de la muestra respectivamente. Esto permitió implementar 4 tipos de vectores. Todos estos tipos de vectores tienen en sus primeras 300 dimensiones los valores de los *Word embeddings* del modelo *es_core_news_lg*. Cada uno de estos vectores se diferencia uno de otro de acuerdo a si incluyen las etiquetas POS y NER como valores concatenados al vector base, si incluyen el valor de la polaridad, o si incluyen ambos. La inclusión de las

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

etiquetas POS y NER incrementa en 14 la dimensión del vector base, mientras que la inclusión de la polaridad la incrementa en 1.

Tabla 11. Dimensiones de vectores de palabras de acuerdo a valores de pos_and_ner y sentiment. Fuente: Elaboración propia.

	Pos_and_ner=True	*pos_and_ner=False*
Sentiment=True	315	301
Sentiment=False	314	300

Adicionalmente se implementó la clase *WordVectorizer* (Clase para concatenación y adecuación de vectores de muestras) que inicializa el modelo *es_core_news_lg* para su uso en el vectorizador y concatena los vectores de todas las muestras del conjunto de datos en un arreglo para ser luego utilizado por los modelos de clasificación automática. *WordVectorizer* será utilizado además más tarde en la implementación de la tubería usada para la automatización del proceso de entrenamiento y predicción.

3.4.2.2.Segunda Etapa: Entrenamiento de modelos de clasificación

En la segunda etapa se utilizaron los arreglos resultantes de la primera etapa para el entrenamiento de los modelos de clasificación automática. Se eligieron 4 modelos de clasificación automática:

- Regresión Logística
- Máquinas de Soporte Vectorial: Se hizo uso de *LinearSVC*, o máquina lineal de soporte vectorial para clasificación.
- Bosques aleatorios: Implementación con 100 árboles y la raíz cuadrada del número total de características como el tamaño de submuestras de fuentes de datos.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

- Bosques aleatorios: Implementación con 1000 árboles y la raíz cuadrada del número total de características como el tamaño de submuestras de fuentes de datos.

Se implementaron dos bosques aleatorios con diferente número de árboles para comparar su desempeño.

Al considerar 4 modelos de clasificación y 4 tipos de vectores por cada uno se obtuvo un total de 16 modelos implementados.

3.4.3. Predicción y evaluación

El conjunto de datos fue divido en dos partes: la primera parte, equivalente a un 80% del total del conjunto de datos o 22566 muestras, fue utilizada para el entrenamiento de los modelos de aprendizaje automático; mientras que la segunda parte, consistente en un 20% del total o 5642 muestras, fue utilizada para la evaluación de los modelos.

La evaluación de los modelos consistió en la predicción o clasificación de 5642 muestras usando el modelo previamente entrenado con otras 22566 muestras, y la comparación de sus predicciones con su valor real. Para establecer dicha comparación se hizo uso de la matriz de confusión y las siguientes métricas:

- Exactitud
- Sensibilidad
- Precisión
- Puntaje F1

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Los resultados mostrados en matrices de confusión pueden ser consultados en la sección de anexos. Los resultados de las métricas se muestran a continuación.

Tabla 12. Evaluación de modelo LinearSVC: Word Embeddings. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.79	0.77	0.7614
Falsa	0.78	0.73	0.76	0.7614

Tabla 13. Evaluación de modelo LinearSVC: Word Embeddings y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.81	0.78	0.7689
Falsa	0.79	0.73	0.76	0.7689

Tabla 14. Evaluación de modelo LinearSVC: Word Embeddings y polaridad. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.79	0.77	0.7620
Falsa	0.78	0.73	0.76	0.7620

Tabla 15.Evaluación de modelo LinearSVC: Word Embeddings, polaridad y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.80	0.78	0.7692
Falsa	0.79	0.74	0.76	0.7692

Tabla 16. Evaluación de modelo Regresión Logística: Word Embeddings. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.79	0.77	0.7613
Falsa	0.78	0.74	0.76	0.7613

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 17. Evaluación de modelo Regresión Logística: Word Embeddings y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.80	0.77	0.7662
Falsa	0.78	0.74	0.76	0.7662

Tabla 18. Evaluación de modelo Regresión Logística: Word Embeddings y polaridad. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.79	0.77	0.7623
Falsa	0.78	0.74	0.76	0.7623

Tabla 19. Evaluación de modelo Regresión Logística: Word Embeddings, polaridad y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.80	0.77	0.7678
Falsa	0.79	0.74	0.76	0.7678

Tabla 20. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.69	0.77	0.73	0.7113
Falsa	0.74	0.65	0.69	0.7113

Tabla 21. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.69	0.78	0.73	0.7200
Falsa	0.75	0.66	0.70	0.7200

Tabla 22. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings y polaridad. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.69	0.78	0.73	0.7139
Falsa	0.75	0.65	0.69	0.7139

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

Tabla 23. Evaluación de modelo de bosques aleatorios con 100 árboles: Word Embeddings, polaridad y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.69	0.77	0.73	0.7166
Falsa	0.75	0.66	0.70	0.7166

Tabla 24. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.70	0.78	0.74	0.7216
Falsa	0.75	0.66	0.70	0.7216

Tabla 25. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.70	0.78	0.74	0.7231
Falsa	0.76	0.66	0.71	0.7231

Tabla 26. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings y polaridad. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.70	0.78	0.74	0.7216
Falsa	0.75	0.66	0.70	0.7216

Tabla 27. Evaluación de modelo de bosques aleatorios con 1000 árboles: Word Embeddings, polaridad y etiquetas POS y NER. Fuente: Elaboración propia.

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.70	0.79	0.74	0.7265
Falsa	0.76	0.67	0.71	0.7265

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Tabla 28. Sensibilidad para detección de noticias falsas. Fuente: Elaboración propia

	Word Embeddings	Word Embeddings y etiquetas POS y NER	Word Embeddings y polaridad	Word Embeddings, polaridad y etiquetas POS y NER
LinearSVC	0.73	0.73	0.73	0.74
Regresión Logística	0.74	0.74	0.74	0.74
Bosques aleatorios (100 árboles)	0.65	0.66	0.65	0.66
Bosques aleatorios (1000 árboles)	0.66	0.66	0.66	0.67

Tabla 29. Exactitud de modelos. Fuente: Elaboración propia.

	Word Embeddings	Word Embeddings y etiquetas POS y NER	Word Embeddings y polaridad	Word Embeddings, polaridad y etiquetas POS y NER
LinearSVC	0.7614	0.7689	0.7620	0.7692
Regresión Logística	0.7613	0.7662	0.7623	0.7678
Bosques aleatorios (100 árboles)	0.7113	0.7200	0.7139	0.7166
Bosques aleatorios (1000 árboles)	0.7216	0.7231	0.7216	0.7265

3.4.4. Elección de modelo y observaciones

Para la selección del modelo a implementar y desplegar se debe tener en cuenta la selección de una métrica óptima que nos permita evaluar y comparar el desempeño de los modelos entrenados. El principal objetivo del presente trabajo es la detección de noticias falsas o *fake news*, por lo que la métrica de mayor relevancia para este caso es la sensibilidad para detectar noticias falsas, esto debido a que un mayor valor de esta métrica incrementa la probabilidad de detectar dichas noticias. Sin embargo, se puede observar que hay 5 modelos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

que poseen el valor más alto de sensibilidad para noticias falsas de 0.74: los 4 modelos de regresión logística y el modelo de soporte de máquinas vectoriales con características de *Word embeddings*, polaridad y etiquetas POS y NER. Debido a esto, se optó por elegir a la métrica de exactitud como segundo criterio para la elección del modelo de entre los 5 mencionados. La exactitud nos permite conocer el porcentaje de noticias que fueron correctamente clasificadas. El valor de 0.7692 de exactitud del modelo de soporte de máquinas vectoriales con características de *Word embeddings*, polaridad y etiquetas POS y NER, fue mayor al de los otros 4 modelos, por lo que se procedió a su implementación y despliegue.

Es necesario mencionar observaciones adicionales relevantes a el presente trabajo:

- La inclusión de la polaridad como característica para un modelo no provoca notable influencia en los valores de sensibilidad de detección de noticias falsas de un modelo. Sin embargo, la influencia es mayor con la inclusión de esta en conjunto con las etiquetas POS y NER.

- La inclusión de la polaridad incrementa en general los valores de exactitud, con la única excepción de los bosques aleatorios con 1000 árboles; sin embargo, su inclusión única incrementa dichos valores en menor medida que la inclusión única de las etiquetas POS y NER. Por otra parte, la inclusión de las etiquetas POS y NER en conjunto con la polaridad brinda los valores más altos de exactitud.

- El mayor número de árboles en los modelos de bosques aleatorios, incrementados de 100 a 1000, incrementó en general todas las métricas; sin embargo, los valores continuaron siendo menores a los de los otros modelos. Se puede asumir que un

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

número de árboles mayor a los entrenados en este proyecto podría brindar mejores resultados; sin embargo, la inclusión de un mayor número de árboles involucra un mayor consumo de recursos de computador. Por esta razón, la implementación de un modelo con mayor número de árboles es inadecuado.

3.5. Despliegue de modelo de aprendizaje automático: Desarrollo de sistema web para encapsulado y presentación de modelo de clasificación

El despliegue de un modelo de aprendizaje automático significa la integración del modelo en un sistema que puede recibir un dato de entrada y retornar una respuesta que puede ser utilizada en situaciones prácticas. El proceso de despliegue puede ser dividido en las siguientes partes: primero, la implementación de un modelo base; segundo, la adopción de una arquitectura de un sistema de aprendizaje automático; y tercero, el despliegue propiamente dicho, en donde se prepara el código para la implementación en el sistema y se le despliega en el sistema de aprendizaje automático elegido.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

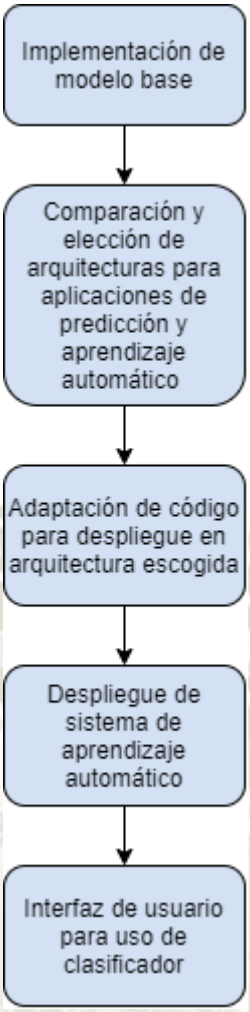


Figura 36. Diagrama de flujo de etapa de despliegue de modelo. Fuente: Elaboración propia

3.5.1. Implementación de un modelo base

La implementación de un modelo base se refiere a el entrenamiento de un modelo de aprendizaje automático que posea adecuados puntajes en las métricas de evaluación y que al mismo tiempo sea adecuado en términos de consumo de recursos para el despliegue. Esta etapa involucra las etapas de preprocesamiento de datos, análisis y exploración de datos, selección de características, y el entrenamiento, evaluación y selección de un modelo de aprendizaje.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Si bien es cierto la etapa de implementación del modelo base fue desarrollada y explorada como parte de los anteriores capítulos, la etapa de preprocesamiento y selección de características tuvo que ser cambiada al considerar el consumo de recursos como criterio adicional para la implementación y despliegue del modelo base. El cambio se dio específicamente en la selección del modelo utilizado para la extracción de características para la obtención de *Word Embeddings* o vectores de palabras y las etiquetas POS y NER, correspondientes a las características complejas y estilísticas.

Anteriormente, se hizo uso del modelo *es_core_news_lg* (Explosion [2017] 2020) como parte de la librería *Spacy* (Explosion [2014] 2020). Este modelo implementa el uso de 500 000 vectores únicos de palabra y un mayor consumo de memoria, que, si bien es cierto puede mejorar los puntajes de las métricas de evaluación de exactitud, precisión y sensibilidad, involucra al mismo tiempo un mayor consumo de memoria que en la práctica es excesivo e inviable para la implementación de un sistema de predicción web. Por esta razón se optó por el uso del modelo *es_core_news_md* (Explosion [2017] 2020) que posee las mismas características del anterior modelo, con la excepción del uso de solo 20 000 vectores únicos de palabras y un menor consumo de memoria. Naturalmente, los puntajes de las métricas de evaluación de este modelo para el reconocimiento de etiquetas POS y NER es menor (Tabla 30. Comparación de métricas de modelos es_core_news_lg y es_core_news_md. Fuente:(spaCy s. f.)), y se puede asumir igualmente que la extracción de *Word Embeddings* no será tan eficaz; sin embargo, en la implementación del modelo base se debe tener en cuenta el equilibrio eficacia-eficiencia, de modo que el modelo implementado pueda lograr el fin propuesto, que es la detección de noticias falas, pero que al mismo tiempo su implementación pueda ser viable.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Tabla 30. Comparación de métricas de modelos es_core_news_lg y es_core_news_md. Fuente:(spaCy s. f.)

Modelo	*es_core_news_lg*	*es_core_news_md*
Exactitud en identificación de etiquetas POS	*97.84*	*97.54*
Precisión en identificación de etiquetas NER	*90.45*	*89.96*
Sensibilidad en identificación de etiquetas NER	*90.19*	*89.71*
Puntaje F1 en identificación de etiquetas NER	*90.32*	*89.84*

Al realizar el entrenamiento del clasificador usando el modelo *es_core_news_md* para la extracción de características, se observó que su desempeño, en cuanto a métricas de evaluación se refiere, fue muy similar a aquellas del clasificador con el modelo *es_core-_news_lg (*Tabla 31. Evaluación de modelo LinearSVC con modelo es_core_news_md: Word Embeddings, polaridad y etiquetas POS y NER. Fuente: Elaboración propia).

Tabla 31. Evaluación de modelo LinearSVC con modelo es_core_news_md: Word Embeddings, polaridad y etiquetas POS y NER. Fuente: Elaboración propia

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.75	0.80	0.77	0.7643
Falsa	0.78	0.73	0.76	0.7643

3.5.2. Adopción de una arquitectura de un sistema de aprendizaje automático

3.5.2.1.Tipos de entrenamiento y de predicción de modelos de aprendizaje automático

Los tipos de arquitectura de un sistema de aprendizaje automático surgen a partir de las necesidades del entrenamiento y la predicción de un determinado caso de estudio. Estas necesidades permiten y caracterizan la división de los procesos de entrenamiento y predicción en dos tipos: en tiempo real y por lotes.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

3.5.2.1.1. Entrenamiento en tiempo real de modelo

El entrenamiento en tiempo real consiste en el entrenamiento constante de un modelo de aprendizaje automático incluso después del despliegue.

Este tipo de entrenamiento requiere un mayor soporte y monitoreo operacional debido a que el modelo es sensible a nuevos datos, de los que se alimenta constantemente, y que pueden contener ruido perjudicial para el entrenamiento de este. Esto involucra un filtrado de dicho ruido en tiempo real.

3.5.2.1.2. Entrenamiento por lotes de modelo

El entrenamiento de un modelo por lotes nos permite obtener versiones actualizadas, aunque no en tiempo real, de un modelo de aprendizaje automático. Dicho modelo es entrenado antes del despliegue y, por medio de diferentes herramientas como la serialización, es almacenado para ser consultado y brindar predicciones después del despliegue. El soporte y monitoreo operacional que requiere este entrenamiento es menor que el entrenamiento en tiempo real debido a que el proceso de filtrado de ruido puede ser automatizado y controlado, además de que se dispone de mayor tiempo para dicha operación.

3.5.2.1.3. Predicción en tiempo real de modelo

La predicción en tiempo real de un modelo consiste en la recepción y respuesta constante a solicitudes de predicción provenientes de usuarios realizando una única solicitud.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

El hecho de que se puedan realizar múltiples solicitudes al mismo tiempo implica mayores desafíos para el sistema. El sistema deberá ser lo suficientemente robusto para ser capaz de manejar y soportar picos de cargas de solicitudes en las que múltiples usuarios realizan sus solicitudes de traducción en un mismo rango de tiempo. Esto involucra también desafíos en términos de infraestructura y costo, ya que la infraestructura del sistema deberá de tener la capacidad de soportar dicha carga que al mismo tiempo podría involucrar mayores costos. Para este tipo de sistemas es ideal una infraestructura adaptable que pueda adaptarse al consumo, es decir, que pueda extender sus capacidades de infraestructura cuando haya picos de solicitudes, pero que al mismo tiempo pueda reducirlas en escasez de dichos picos con el objetivo de ahorrar costos.

3.5.2.1.4. Predicción por lotes de modelo

La predicción por lotes es útil cuando se quiere generar predicciones para un grupo de observaciones en conjunto y al mismo tiempo. Si bien es cierto, la predicción por lotes, al igual que el entrenamiento por lotes, requiere de un menor soporte y monitoreo operacional, involucra también una mayor latencia en la predicción debido a que el conjunto de datos a predecir posee un mayor número de registros u observaciones. Por otra parte, el hecho de que la predicción no sea constante evita la necesidad de un sistema adaptable debido a que las solicitudes de predicción pueden realizarse de manera controlada y organizada.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

3.5.2.2. Tipos de arquitectura de sistemas de Aprendizaje Automático

De acuerdo a los tipos de predicción y entrenamiento, se puede clasificar arquitectura de sistemas de aprendizaje automático en cuatro tipos (Soledad Galli y Chritopher Samiullah s. f.):

- Entrenamiento por lotes, predicción en tiempo real y servicio de predicción por web API: En este caso, el entrenamiento del modelo se da previo al despliegue, mientras que la predicción puede darse en tiempo real utilizándose servicios de API.

- Entrenamiento por lotes, predicción por lotes y servicio de predicción a través de una base de datos compartida: En este caso, el entrenamiento y predicción no son en tiempo real. El entrenamiento del modelo se da previo al despliegue y la consulta de predicción involucra generalmente un gran número de observaciones o registros previamente ya predichos. La consulta se da a una base de datos compartida.

- Entrenamiento y predicción en tiempo real: El entrenamiento y la predicción se dan en tiempo real usando los recursos del dispositivo.

- Entrenamiento en tiempo real, predicción en tiempo real y servicio de predicción a través de API de dispositivo móvil: El entrenamiento y la predicción se dan en tiempo real, pero el servicio de predicción se da a través de API de dispositivo móvil. Por lo general, está sujeto y limitado a la capacidad de un dispositivo móvil, por lo que su complejidad es menor a la de las otras arquitecturas.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Las arquitecturas de sistemas de aprendizaje automático pueden ser resumidas en el siguiente cuadro:

Tabla 32. Comparación de arquitecturas de sistemas de Aprendizaje Automático. Fuente:(Soledad Galli y Chritopher Samiullah s. f.)

	Arquitectura 1	Arquitectura 2	Arquitectura 3	Arquitectura 4
Entrenamiento	Por lotes	Por lotes	En tiempo real	En tiempo real
Predicción	En tiempo real	Por lotes	En tiempo real	En tiempo real
Servicio de predicción	web API	Base de datos compartida	Por API de dispositivo	Por API de dispositivo móvil
Latencia de predicción	Media	Alta	Muy baja	Baja
Dificultad para mantenimiento de sistema	Media	Baja	Muy alta	Media

3.5.2.3.Elección de Arquitectura

Para el presente proyecto se eligió la arquitectura de entrenamiento por lotes, predicción en tiempo real y servicio a través de una web API. Se eligió esta arquitectura debido a los siguientes puntos:

- El modelo para entrenar cuenta con un único conjunto de datos previamente etiquetado para el entrenamiento y no recibirá datos de entrenamiento por parte del usuario final. Esto hace innecesario el diseño de una interfaz para el entrenamiento en tiempo real. Por esto, se realizará el entrenamiento por lotes para obtener una única versión estable del modelo de clasificación.

- El usuario final realizará solicitudes constantemente para obtener la predicción de una noticia como falsa o confiable. Debido a que estas solicitudes serán constantes,

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

únicas, y podrían realizarse por múltiples usuarios, es necesaria una predicción en tiempo real.

- El uso de una web API es necesaria para poder administrar y responder a las solicitudes constantes de los usuarios a través del sistema web también implementado como parte del desarrollo de este proyecto.

Ya que el presente proyecto involucra una predicción en tiempo real, es necesario también tener en consideración la característica de flexibilidad y adaptación de la arquitectura disponible para soportar picos de solicitudes.

3.5.3. Adaptación de código para despliegue

En capítulos anteriores se exploró el proceso de preprocesamiento y entrenamiento de los modelos de aprendizaje automático manualmente; sin embargo, para su despliegue en la arquitectura elegida, es necesaria la automatización de dicho proceso.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

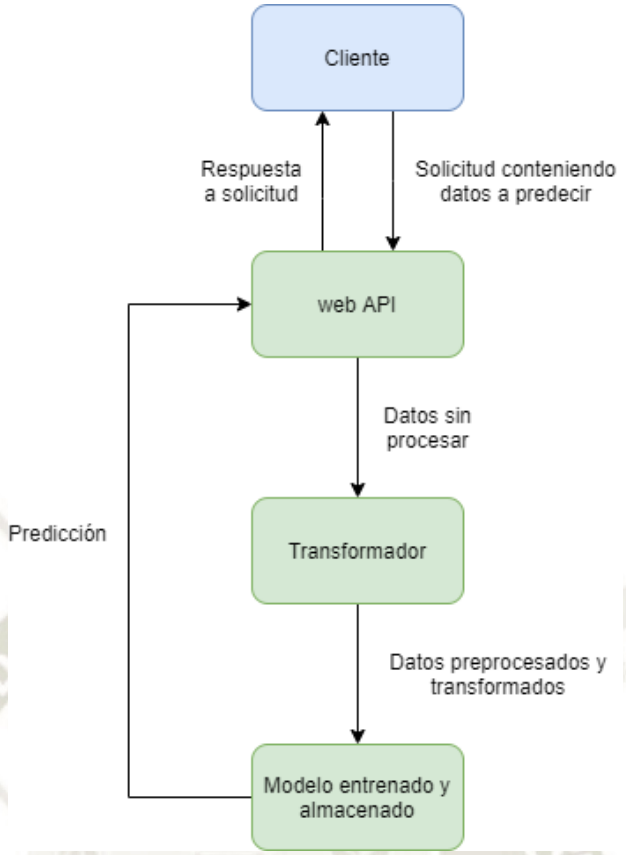


Figura 37. Proceso de consulta de predicción a sistema de Aprendizaje Automático. Fuente: Elaboración Propia.

Para la automatización del proceso de consulta es necesario tener en cuenta tres etapas principales: el diseño de una web API que pueda administrar y brindar respuesta a las consultas hechas por un usuario cliente, el uso de un transformador que permita extraer las características para nuestro modelo y vectorizar los datos requeridos, y el almacenamiento de un modelo previamente entrenado que realizará las predicciones.

3.5.3.1.Web API

La primera etapa involucra el diseño de una web API que permite administrar y responder las solicitudes hechas por el usuario. Una API, Interfaz de Programación de Aplicación por sus siglas en inglés, es un software que permite la comunicación entre dos

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

aplicaciones. En este caso, la web API desarrollada comunica el modelo de aprendizaje automático desarrollado y entrenado con la interfaz gráfica que el usuario utilizará para solicitar predicciones.

La web API de este proyecto fue desarrollada utilizando la librería *Flask* (Grinberg 2014), que es un entorno de trabajo ligero para desarrollo de aplicaciones web.

La estructura de la web API consiste en un solo método POST que recibe como único parámetro a la llave *title* y como valor al título de noticia a predecir. La respuesta tiene como llave a *pred* y como valor a el número 1 o 0, correspondientes a noticia falsa y noticia confiable respectivamente. La solicitud y consulta deben de respetar el formato JSON.

La web API desarrollada recibirá la solicitud que el usuario envíe. El título de esta solicitud será enviado al transformador y al modelo de predicción. Cuando el modelo de predicción tenga la predicción respectiva enviará esta predicción a la web API, en donde dicha predicción será formateada para ser enviada como respuesta al cliente.

3.5.3.2.Transformador

El transformador es aquella etapa en la que se recibe el título en texto desde la web API. Luego, el transformador transformará dicho texto a las características numéricas necesarias para el entrenamiento del modelo de predicción. El desarrollo del Transformador fue explorado anteriormente en la etapa de preprocesamiento y vectorización del capítulo *Entrenamiento, evaluación y comparación de modelos de Aprendizaje Automático*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.5.3.3.Modelo entrenado y almacenado

Después de haber entrenado el modelo de aprendizaje automático es necesario su almacenamiento y persistencia, de modo que pueda ser utilizado para operaciones de predicción en sistemas de aprendizaje automático sin necesidad de un entrenamiento adicional. Para esto, existen dos herramientas útiles: tuberías y serialización.

3.5.3.3.1. Tuberías

Las tuberías son una herramienta utilizada para aplicar secuencialmente una lista de transformadores y un estimador final. El uso de tubería permitirá automatizar y encadenar los procesos de preprocesamiento y transformación, entrenamiento y predicción de un modelo de aprendizaje automático. En el caso del presente proyecto, la tubería constó de una primera etapa de transformación, basada en el transformador desarrollado, para obtener las características numéricas para el modelo, y una segunda etapa basada en el uso del modelo clasificador elegido. La tubería de este proyecto fue implementada mediante la librería *scikit-learn* (Pedregosa et al. 2018), y tiene como primera etapa el transformador desarrollado, y como segunda etapa el clasificador basado en máquinas de soporte vectorial, específicamente, *LinearSVC*.

3.5.3.3.2. Serialización

La serialización es el proceso mediante el cual se convierte el estado de un objeto en un formato que puede ser almacenado y compartido. La serialización convierte el estado de un objeto en una cadena de bits, de modo que dicha cadena pueda ser utilizada más tarde para recuperar el estado de un objeto mediante el proceso de deserialización (Grochowski, Breiter,

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

y Nowak 2019:1). La serialización nos permitirá almacenar el modelo, entrenado previo al despliegue, para luego ser deserializado y cargado para cada solicitud de predicción. Esto nos asegura la persistencia del modelo para que no necesite de ser entrenado nuevamente por cada solicitud.

El presente proyecto hace uso de la librería *joblib* (The joblib developers [2010] 2020) para el proceso de serialización y deserialización.

3.5.4. Despliegue de sistema de aprendizaje automático

Para el despliegue de la aplicación se optó por utilizar un proveedor de servicios en nube. Existen 4 principales proveedores de servicios en nube que ofrecen plataformas adecuadas para el despliegue de modelos de aprendizaje automático: *Google Cloud* (Google s. f.), *Amazon Web Services* (Amazon s. f.), *Microsoft Azure* (Microsoft Azure s. f.) y *Oracle Cloud* (Oracle s. f.). En el presente proyecto se optó por el uso de *Google Cloud* como proveedor de servicios en nube debido a la mayor familiaridad del autor con dicha plataforma; sin embargo, el uso de cualquier otro proveedor es igualmente posible.

Se evaluaron dos plataformas que forman parte de *Google Cloud* (Google s. f.) y que son adecuadas para el despliegue de modelos de inteligencia artificial: *AI Platform* (Google Cloud s. f.) y *App Engine* (Google Cloud s. f.).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

3.5.4.1.AI Platform

Es una plataforma ideal para el despliegue de modelos de inteligencia artificial que ofrece servicios de alojamiento, entrenamiento y predicción en la nube. Implementa un servicio de entrenamiento que permite el entrenamiento de modelos de aprendizaje automático, además del ajuste de algunos parámetros como son la afinación de parámetros del modelo, la habilitación de entrenamiento distribuido y la aceleración con uso de GPUs y TPUs. Ofrece también un servicio de predicción que nos asegura la persistencia del modelo entrenado, además de la consulta de predicciones con otras aplicaciones por medio de solicitudes por servicios REST.

Si bien es cierto esta plataforma fue desarrollada específicamente para modelos de inteligencia artificial, lo que lo hace ideal para cualquier aplicación relacionada, la aplicación de dicha plataforma para la implementación y predicción con tuberías personalizadas, como es el caso de este proyecto, no se encuentra aún en la versión final de desarrollo, sino más bien, en una versión beta. Por esta razón, se optó por el uso de la plataforma *App Engine* (Google Cloud s. f.) para el despliegue del modelo.

3.5.4.2.App Engine

Esta plataforma permite ejecutar aplicaciones escritas en Python (Van Rossum y Drake 2009) y otros lenguajes de programación para propósito general, y no únicamente modelos de aprendizaje automático. No incluye la preimplementación de servicios REST para realizar solicitudes, como es el caso de la *AI Platform*, por lo que requiere de la implementación de un mecanismo para la administración y control de solicitudes. Dicha implementación se realizó

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

en el presente proyecto mediante el desarrollo de una web API usando la librería *Flask* (Grinberg 2014).

App Engine (Google Cloud s. f.) posee dos entornos de trabajo: el entorno estándar y el entorno flexible. Ambos entornos ofrecen escalabilidad a la aplicación, de modo que puedan consumir mayores recursos cuando haya un mayor tráfico o una mayor cantidad de solicitudes. Esto los hace opciones ideales para el despliegue de modelos de aprendizaje automático con un tipo de predicción en tiempo real. Para este proyecto, se optó por el uso del entorno estándar. Se tomó esta decisión ya que el entorno estándar posee un menor tiempo de inicio, es ideal para aumentos repentinos de tráfico, y puede permanecer desplegado sin incurrir en costos adicionales en ausencia de solicitudes (Google Cloud s. f.).

El desempeño del entorno estándar es determinado por la denominada instancia, cuya clase determina la cantidad límite de memoria y frecuencia del procesador que la aplicación puede consumir. Después de realizar pruebas, se eligió la clase de instancia *F4_1G* que posee como características un límite de memoria de 2048 MB y de CPU de 2.4 GHz, y que permite el funcionamiento óptimo de la aplicación.

3.6.Interfaz de usuario

Se desarrolló una aplicación web para ser utilizada como interfaz gráfica de usuario, de modo que el usuario pueda enviar solicitudes de predicción a el modelo de aprendizaje automático desplegado en *App Engine* (Google Cloud s. f.).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Para dicho desarrollo se hizo uso del entorno de trabajo para desarrollo de aplicaciones web *Angular* (Jain et al. 2015) y la librería de componentes *Angular Material* (Angular Components Team s. f.). La aplicación fue alojada en *Firebase* (Firebase s. f.), que es una plataforma de desarrollo de recursos en nube.

Dicha aplicación constó de un formulario con un único campo de entrada en el que el usuario puede ingresar el título de la noticia que se quiere consultar, para predecir si se trata de una noticia falsa o confiable. Se implementó como única restricción un número máximo de palabras de 34, que es el número máximo de palabras del conjunto de datos usado para el entrenamiento.

La aplicación final implementada puede ser consultada en el sitio web *https://felix-obando-tesis-front.web.app/.* Se presenta la interfaz gráfica de usuario en la Figura 38. Interfaz gráfica de usuario de detector de noticias falsas. Fuente: Elaboración Propia. Se presenta además un ejemplo de noticia falsa (BBC News Mundo 2018) en Figura 39. Ejemplo de noticia falsa. Fuente: Elaboración Propia. Se presenta también un ejemplo de noticia confiable (Cullell 2021) en Figura 40. Ejemplo de noticia confiable. Fuente: Elaboración propia.

Tabla 33. Descripción de scripts desarrollados para despliegue. Fuente: Elaboración propia

Anexo	Descripción
Main.py	Script principal. Contiene la aplicación para administrar solicitudes POST y llamar otros scripts.
pipeline.py	Tubería desarrollada que convoca script de etapa de transformación y modelo de clasificación *LinearSVC*
transformer.py	Script para preprocesamiento de datos y obtención de características.
predict.py	Script utilizado para evaluar predicción.
train_pipeline.py	Script utilizado para entrenar modelo de predicción.
data_management.py	Script utilizado para cargar conjunto de datos, almacenar y serializar modelo de aprendizaje automático, y cargar y deserializar modelo de aprendizaje automático.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Figura 38. Interfaz gráfica de usuario de detector de noticias falsas. Fuente: Elaboración Propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

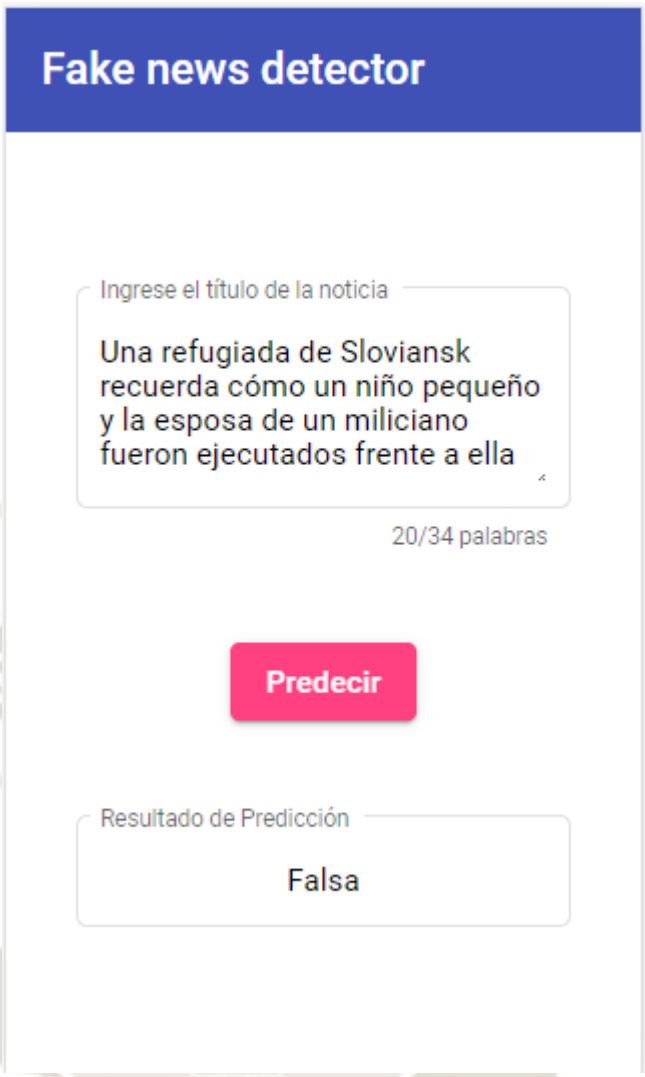


Figura 39. Ejemplo de noticia falsa. Fuente: Elaboración Propia

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

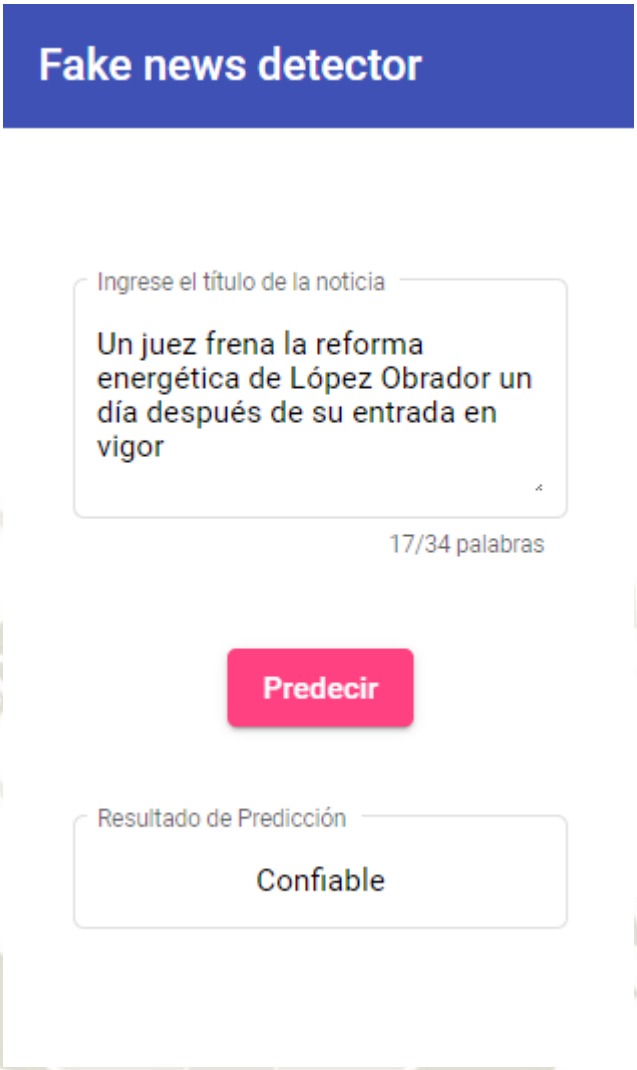


Figura 40. Ejemplo de noticia confiable. Fuente: Elaboración propia

3.7. Resultados: Evaluación de implementación

Después de realizada la implementación y despliegue de la aplicación de predicción, se procedió a evaluar el desempeño de esta. Para esto se hizo uso de un nuevo conjunto de datos consistente en 676 registros de noticias en español, y de métricas de evaluación de modelos de aprendizaje automático, como son precisión, sensibilidad, exactitud y puntaje F1. El conjunto de datos (jpposadas [2018] 2020) utilizado para la evaluación fue propuesto como parte del trabajo *Detection of Fake News in a New Corpus for the Spanish Language* (Posadas-Durán et al. 2019). Los resultados de esta evaluación pueden ser visualizados en la Tabla 34.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Tabla 34. Evaluación de predicciones de implementación. Fuente: Elaboración Propia

Tipo	Precisión	Sensibilidad	Puntaje F1	Exactitud
Confiable	0.58	0.76	0.66	0.6095
Falsa	0.66	0.46	0.54	0.6095

Si bien es cierto los resultados de las métricas han disminuido, los valores de exactitud corresponden a un puntaje mayor al 0.5, lo que evidencia una capacidad de clasificación suficiente y eficaz de la aplicación implementada.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

CAPÍTULO IV

4. Conclusiones y Recomendación

4.1.Conclusiones

- Se comprobó la posibilidad de la implementación de un clasificador que identifica noticias falsas mediante la traducción de un corpus en inglés y el uso de algoritmos de Machine Learning.

- Se logró implementar un clasificador basado en aprendizaje máquina para el reconocimiento del título de noticias originalmente en idioma español como falsas. El modelo de aprendizaje automático elegido, LinearSVC contó con los mejores puntajes de clasificación.

- Se realizó una exploración y limpieza de datos que aseguró la integridad del conjunto de datos para el entrenamiento del modelo clasificador.

- El uso de un corpus traducido mediante un traductor automático y filtrado mediante el uso de la métrica METEOR (Banerjee y Lavie 2005) y el método del *backtranslation* permitió la obtención de un modelo de clasificación con resultados de métricas de exactitud y cobertura elevados.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

- Se pudo obtener datos numéricos producto de la extracción y codificación de las características semánticas, sintácticas y polaridad de muestras textuales para ser utilizados en el entrenamiento de modelos de aprendizaje.

- Se observó que el uso conjunto de características semánticas, sintácticas y polaridad mejora considerablemente el desempeño de un modelo de aprendizaje automático.

- Se logró el despliegue de un sistema web que implementa el modelo de aprendizaje entrenado para clasificar noticias.

4.1.Recomendación para implementaciones futuras

Se logró implementar satisfactoriamente un modelo de clasificación que permite le identificación de noticias falsas; sin embargo, el entrenamiento y uso de un modelo adicional de predicción para la identificación de la polaridad podría mejorar el desempeño del modelo implementado. La inclusión de la polaridad como característica adicional para el entrenamiento del modelo de aprendizaje automático mejoró los resultados de las métricas de evaluación; sin embargo, la mejora en dichas métricas fue muy reducida a comparación de la inclusión de características sintácticas. En una futura implementación se podría realizar el entrenamiento de un modelo de aprendizaje automático adicional para la obtención de la polaridad como alternativa al uso de un léxico de sentimientos. La polaridad obtenida de este modelo podría ser luego utilizada para otro clasificador de noticias falsas.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

REFERENCIAS

Abonizio, Hugo Queiroz, Janaina Ignacio de Morais, Gabriel Marques Tavares, y Sylvio Barbon Junior. 2020. «Language-Independent Fake News Detection: English, Portuguese, and Spanish Mutual Features». *Future Internet* 12(5):87. Doi: 10.3390/fi12050087.

Agencia EFE. s. f. «Agencia EFE». *www.efe.com*. Recuperado 16 de septiembre de 2020 (https://www.efe.com/).

Al Asaad, Bashar, y Madalina Erascu. 2018. «A Tool for Fake News Detection». Pp. 379-86 en.

ALRFOU, RAMI. [2014] 2020. *aboSamoor/polyglot*.

Amazon. s. f. «Amazon Web Services (AWS) - Cloud Computing Services». *Amazon Web Services, Inc.* Recuperado 15 de octubre de 2020 (https://aws.amazon.com/).

Angular Components Team. s. f. «Angular Material». *Angular Material*. Recuperado 16 de octubre de 2020 (https://material.angular.io/).

aylliote. [2017] 2020. *aylliote/senti-py*.

Banerjee, Satanjeev, y Alon Lavie. 2005. «METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments». Pp. 65–72 en *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Ann Arbor, Michigan: Association for Computational Linguistics.

Baroni, Marco, Georgiana Dinu, y Germán Kruszewski. 2014. «Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors».

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Pp. 238–247 en *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics.

BBC News Mundo. 2018. «3 noticias falsas que propiciaron guerras y conflictos alrededor del mundo». *BBC News Mundo*, abril 25.

Behnel, Stefan, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn, y Kurt Smith. 2011. «Cython: The best of both worlds». *Computing in Science & Engineering* 13(2):31–39.

Bhutani, Bhavika, Neha Rastogi, Priyanshu Sehgal, y Archana Purwar. 2019. «Fake News Detection Using Sentiment Analysis». Pp. 1-5 en *2019 Twelfth International Conference on Contemporary Computing (IC3)*.

Bierman, Gavin, Martín Abadi, y Mads Torgersen. 2014. «Understanding typescript». Pp. 257–281 en *European Conference on Object-Oriented Programming*. Springer.

Bird, Steven, Ewan Klein, y Edward Loper. 2009. *Natural Language Processing with Python*. 1st ed. O'Reilly Media, Inc.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, y Tomas Mikolov. 2017. «Enriching Word Vectors with Subword Information». *arXiv:1607.04606 [cs]*.

Bojar, Ondřej, y Aleš Tamchyna. 2011. «Improving Translation Model by Monolingual Data». Pp. 330–336 en *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Edinburgh, Scotland: Association for Computational Linguistics.

Chaovalit, P., y L. Zhou. 2005. «Movie Review Mining: a Comparison between Supervised and Unsupervised Classification Approaches». Pp. 112c-112c en *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Chen, Boxing, y Colin Cherry. 2014. «A Systematic Comparison of Smoothing Techniques for Sentence-Level BLEU». Pp. 362–367 en *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Baltimore, Maryland, USA: Association for Computational Linguistics.

Chen, Yanqing, y Steven Skiena. 2014. «Building Sentiment Lexicons for All Major Languages». Pp. 383–389 en *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Baltimore, Maryland: Association for Computational Linguistics.

Cloud Translation - Google Cloud. s. f. «Pricing | Cloud Translation | Google Cloud». Recuperado 29 de agosto de 2020 (https://cloud.google.com/translate/pricing).

Collins English Dictionary. s. f. «Artificial Intelligence Definition and Meaning | Collins English Dictionary». Recuperado 31 de agosto de 2020a (https://www.collinsdictionary.com/dictionary/english/artificial-intelligence).

Collins English Dictionary. s. f. «Machine Translation Definition and Meaning | Collins English Dictionary». Recuperado 31 de agosto de 2020b (https://www.collinsdictionary.com/dictionary/english/machine-translation).

Cullell, Jon Martín. 2021. «Un juez frena la reforma energética de López Obrador un día después de su entrada en vigor». *EL PAÍS*, marzo 11.

Denkowski, Michael, y Alon Lavie. 2014. «Meteor Universal: Language Specific Translation Evaluation for Any Target Language». Pp. 376–380 en *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Baltimore, Maryland, USA: Association for Computational Linguistics.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Dhaoui, Chedia, Cynthia M. Webster, y Lay Peng Tan. 2017. «Social media sentiment analysis: lexicon versus machine learning». *Journal of Consumer Marketing* 34(6):480-88. Doi: 10.1108/JCM-03-2017-2141.

Edunov, Sergey, Myle Ott, Marc'Aurelio Ranzato, y Michael Auli. 2020. «On The Evaluation of Machine Translation Systems Trained With Back-Translation». *arXiv:1908.05204 [cs]*.

Explosion. [2014] 2020. *explosion/spaCy*. Explosion.

Explosion. [2015] 2019. *explosion/spacy-benchmarks*. Explosion.

Explosion. [2017] 2020. *explosion/spacy-models*. Explosion.

fastText. s. f. «Word Representations». Recuperado 11 de septiembre de 2020 (https://fasttext.cc/index.html).

Firebase. s. f. «Documentation». Recuperado 16 de octubre de 2020 (https://firebase.google.com/docs).

Galarnyk, Michael. 2020. «Understanding Boxplots». *Medium*. Recuperado 1 de agosto de 2020 (https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51).

Google Cloud. s. f. «AI Platform». *Google Cloud*. Recuperado 15 de octubre de 2020a (https://cloud.google.com/ai-platform).

Google Cloud. s. f. «App Engine Application Platform». *Google Cloud*. Recuperado 15 de octubre de 2020b (https://cloud.google.com/appengine).

Google Cloud. s. f. «Choosing an App Engine Environment | App Engine Documentation». *Google Cloud*. Recuperado 16 de octubre de 2020c (https://cloud.google.com/appengine/docs/the-appengine-environments).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Google Cloud. s. f. «Cloud Translation». *Google Cloud*. Recuperado 31 de agosto de 2020d (https://cloud.google.com/translate).

Google. s. f. «Cloud Computing Services | Google Cloud». Recuperado 15 de octubre de 2020 (https://cloud.google.com/).

Grinberg, Miguel. 2014. *Flask Web Development: Developing Web Applications with Python*. 1st ed. O'Reilly Media, Inc.

Grochowski, Konrad, Michał Breiter, y Robert Nowak. 2019. «Serialization in Object-Oriented Programming Languages».

Harris, Zellig S. 1981. «Distributional Structure». Pp. 3-22 en *Papers on Syntax*, *Synthese Language Library*, editado por Z. S. Harris y H. Hiż. Dordrecht: Springer Netherlands.

Hinton, E. 1986. *3 Learning distributed of concepts*.

Honnibal, Matthew, y Ines Montani. 2017. «spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing».

Horne, Benjamin D., y Sibel Adali. 2017. «This Just In: Fake News Packs a Lot in Title, Uses Simpler, Repetitive Content in Text Body, More Similar to Satire than Real News». *arXiv:1703.09398 [cs]*.

Hunter, John D. 2007. «Matplotlib: A 2D Graphics Environment». *Computing in Science Engineering* 9(3):90-95. Doi: 10.1109/MCSE.2007.55.

Jain, Aditya, Gandhar Kulkarni, y Vraj Shah. 2018. «Natural Language Processing». *International Journal of Computer Sciences and Engineering* 6:161-67. Doi: 10.26438/ijcse/v6i1.161167.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Jain, Nimisha, P. Mangal, y D. Mehta. 2015. «AngularJS: A modern MVC framework in JavaScript». *J. Global Research in Computer Science* 5:17-23.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, y Tomas Mikolov. 2016. «Bag of Tricks for Efficient Text Classification». *arXiv:1607.01759 [cs]*.

jpposadas. [2018] 2020. *jpposadas/FakeNewsCorpusSpanish*.

Koehn, Philipp, Franz J. Och, y Daniel Marcu. 2003. «Statistical Phrase-Based Translation». Pp. 127–133 en *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.

Lanius, David, y Romy Jaster. 2018. «What is Fake News?»

Lavie, Alon, y Abhaya Agarwal. 2007. «METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments». Pp. 228–231 en *Proceedings of the Second Workshop on Statistical Machine Translation*. Prague, Czech Republic: Association for Computational Linguistics.

LLC, Google. s. f. *google-cloud-translate: Google Cloud Translation API client library*.

Lynch, Gerard, y Carl Vogel. 2018. «The Translator's Visibility: Detecting Translatorial Fingerprints in Contemporaneous Parallel Translations». *Computer Speech & Language* 52:79-104. Doi: 10.1016/j.csl.2018.05.002.

Manning, Christopher D., Prabhakar Raghavan, y Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

McKinney, Wes, y others. 2010. «Data structures for statistical computing in python». Pp. 51–56 en *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

micromedix. 2018. «Matriz de confusión». *Medicina Alternativa*. Recuperado 27 de septiembre de 2020 (https://micromedix.me/2018/01/27/cardiopatias-en-las-cosas-del-corazon-no-basta-con-una-segunda-opinion/matriz-de-confusion/).

Microsoft Azure. s. f. «Cloud Computing Services | Microsoft Azure». Recuperado 15 de octubre de 2020 (https://azure.microsoft.com/en-us/).

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, y Jeff Dean. 2013. «Distributed Representations of Words and Phrases and their Compositionality». Pp. 3111–3119 en *Advances in Neural Information Processing Systems 26*, editado por C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, y K. Q. Weinberger. Curran Associates, Inc.

Molnar, Christoph. 2020. *Interpretable Machine Learning*. Lulu.com.

Mukhtar, Neelam, Mohammad Abid Khan, y Nadia Chiragh. 2018. «Lexicon-Based Approach Outperforms Supervised Machine Learning Approach for Urdu Sentiment Analysis in Multiple Domains». *Telematics and Informatics* 35(8):2173-83. Doi: 10.1016/j.tele.2018.08.003.

Nothman, Joel, Nicky Ringland, Will Radford, Tara Murphy, y James R. Curran. 2013. «Learning Multilingual Named Entity Recognition from Wikipedia». *Artificial Intelligence* 194:151-75. Doi: 10.1016/j.artint.2012.03.006.

Oracle. s. f. «Migration to the Cloud Made Simple». Recuperado 15 de octubre de 2020 (https://www.oracle.com/cloud/).

Papineni, Kishore, Salim Roukos, Todd Ward, y Wei-Jing Zhu. 2002. «Bleu: a Method for Automatic Evaluation of Machine Translation». Pp. 311–318 en *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, y Édouard Duchesnay. 2018. «Scikit-learn: Machine Learning in Python». *arXiv:1201.0490 [cs]*.

Periódico, El. s. f. «El Periódico - Noticias y última hora de Catalunya, España y el mundo». *El Periódico*. Recuperado 16 de septiembre de 2020 (https://www.elperiodico.com/es/).

Porter, M. 2001. «Snowball: A Language for Stemming Algorithms».

Porter, M. F. 1980. «An algorithm for suffix stripping». *Program* 14(3):130-37. Doi: 10.1108/eb046814.

Posadas-Durán, Juan-Pablo, Helena Gomez Adorno, Grigori Sidorov, y Jaime Moreno. 2019. «Detection of fake news in a new corpus for the Spanish language». *Journal of Intelligent & Fuzzy Systems* 36:4869-76. Doi: 10.3233/JIFS-179034.

Project Jupyter. [2015] 2020. *Jupyter Notebook*. Project Jupyter.

Python 3.8.5 documentation. s. f. «General Python FAQ — Python 3.8.5 documentation». Recuperado 21 de julio de 2020 (https://docs.python.org/3/faq/general.html).

Python Software Foundation. s. f. «Python Software Foundation». *Python.Org*. Recuperado 21 de julio de 2020 (https://www.python.org/psf/).

scikit-learn 0.23.2 documentation. s. f. «1.11. Ensemble methods». Recuperado 26 de septiembre de 2020b (https://scikit-learn.org/stable/modules/ensemble.html#random-forest-parameters).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM
UNIVERSIDAD CATÓLICA DE SANTA MARÍA

scikit-learn 0.23.2 documentation. s. f. «1.4. Support Vector Machines». Recuperado 23 de septiembre de 2020a (https://scikit-learn.org/stable/modules/svm.html#svm-classification).

Seaborn. s. f. «seaborn.boxplot — seaborn 0.10.1 documentation». Recuperado 1 de agosto de 2020 (https://seaborn.pydata.org/generated/seaborn.boxplot.html).

Sebastián-Gallés, Núria. 2000. *LEXESP: léxico informatizado del español*. Edicions Universitat Barcelona.

Shafranovich, Yakov. s. f. «Common Format and MIME Type for Comma-Separated Values (CSV) Files». Recuperado 31 de agosto de 2020 (https://tools.ietf.org/html/rfc4180).

Simeone, Osvaldo. 2018. «A Very Brief Introduction to Machine Learning With Applications to Communication Systems». *arXiv:1808.02342 [cs, math]*.

Sitikhu, Pinky, Kritish Pahi, Pujan Thapa, y Subarna Shakya. 2019. «A Comparison of Semantic Similarity Methods for Maximum Human Interpretability». *arXiv:1910.09129 [cs]*.

Soledad Galli, y Chritopher Samiullah. s. f. «Deployment of Machine Learning Models | Udemy». Recuperado 12 de octubre de 2020 (https://www.udemy.com/course/deployment-of-machine-learning-models/).

spaCy. s. f. «Facts & Figures, SpaCy Usage Documentation». *Facts & Figures*. Recuperado 17 de septiembre de 2020a (https://spacy.io/usage/facts-figures).

spaCy. s. f. «Language Processing Pipelines, SpaCy Usage Documentation». *Language Processing Pipelines*. Recuperado 17 de septiembre de 2020b (https://spacy.io/usage/processing-pipelines#_title).

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

spaCy. s. f. «Spanish, SpaCy Models Documentation». *Spanish*. Recuperado 17 de septiembre de 2020c (https://spacy.io/models/es).

Srinivasan, S., ed. 2018. *Guide to Big Data Applications*. Springer International Publishing.

Stahlberg, Felix. 2019. «Neural Machine Translation: A Review». *arXiv:1912.02047 [cs]*.

Suárez, Pedro Javier Ortiz, Benoît Sagot, y Laurent Romary. 2019. «Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures». Leibniz-Institut für Deutsche Sprache.

Szpakowski, Maciej. [2018] 2020. *several27/FakeNewsCorpus*.

Taulé, Mariona, M. Antònia Martí, y Marta Recasens. 2008. «AnCora: Multilevel Annotated Corpora for Catalan and Spanish». en *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco: European Language Resources Association (ELRA).

Techopedia. s. f. «What Is a Sparse Matrix? - Definition from Techopedia». *Techopedia.Com*. Recuperado 4 de agosto de 2020 (http://www.techopedia.com/definition/9482/sparse-matrix).

Tf-idf. s. f. «Tf-idf :: A Single-Page Tutorial - Information Retrieval and Text Mining». Recuperado 5 de agosto de 2020 (http://www.tfidf.com/).

The joblib developers. [2010] 2020. *joblib/joblib*. joblib.

The New York Times. s. f. «Obituaries». *The New York Times*.

Van Rossum, Guido, y Fred L. Drake. 2009. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Waskom, Michael, Olga Botvinnik, Drew O'Kane, Paul Hobson, Saulius Lukauskas, David C. Gemperline, Tom Augspurger, Yaroslav Halchenko, John B. Cole, Jordi Warmenhoven, Julian De Ruiter, Cameron Pye, Stephan Hoyer, Jake Vanderplas, Santi Villalba, Gero Kunter, Eric Quintero, Pete Bachant, Marcel Martin, Kyle Meyer, Alistair Miles, Yoav Ram, Tal Yarkoni, Mike Lee Williams, Constantine Evans, Clark Fitzgerald, Brian, Chris Fonnesbeck, Antony Lee, y Adel Qalieh. 2017. *Mwaskom/Seaborn: V0.8.1 (September 2017)*. Zenodo.

Wikipedia. 2020. «Wikipedia». *Wikipedia, the free encyclopedia*.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, y Jeffrey Dean. 2016. «Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation». *ArXiv:1609.08144 [Cs]*.

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

ANEXOS

Código de programación

Extracción por partes de Conjunto de datos

```
corpus_titles=pd.DataFrame()


for chunk in pd.read_csv("../../news_cleaned_2018_02_13.csv", chunksize=100000, encoding='utf-8', error_bad_lines=False, \
        lineterminator='\n', usecols=['domain', 'type', 'title'],\
        dtype={'Unnamed: 0':str, 'id':str, 'domain':"category", 'type':"category", 'url':str, 'content':str, \
            'scraped_at':str, 'inserted_at':str, 'updated_at':str, 'title':str, 'authors':str, 'keywords':str,\
            'meta_keywords':str, 'meta_description':str, 'tags':str, 'summary':str}):

    x=chunk.dropna(subset=["title", 'type', 'domain'])

    if(not x.empty):
        corpus_titles = pd.concat([corpus_titles, x]).astype({'domain':"category", 'type':"category", 'title':str})
        del x
    print(corpus_titles.info(verbose=False, memory_usage="deep"))
    print("------------------------\n")
del chunk
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

Función con expresiones regulares usada para eliminar nombres de dominio repetidos en título

```
#the following function takes a title and the domain, and returns the title
#without the characteristic pattern we identified before. If there is no match,
#returns the title
def rem_trailing_domain(title, domain):
  #print(title)
  regExp = re.compile(r'(\s\W+\s\w[\w\s\.]+)+$', re.IGNORECASE)

  matches = re.search(regExp, title)

  if matches:
    #print(matches.group())

    #We include Houston Chronicle and Yahoo7 as they were some common words
    #that also represent the newspaper, and therefore, do not add any important meaning
    if (('Houston Chronicle' in matches.group()) | ('Yahoo7' in matches.group())):
      return re.sub(regExp, '', title)

    matches_list_no_filtered = re.split(r'\W+', matches.group())

    #print(matches_list_no_filtered)
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
        matches_list_filtered = list(filter(lambda a: (a!='') & (a.lower() not in ['the', 'com',
'net'])\
                                    & (len(a)>1), matches_list_no_filtered))
        #print(matches_list_filtered)
    #We try to identify whether we can find the matched words in the domain. If we do, we
delete
        #the match
        domain_match = "".join(domain.split("."))
        #print(domain_match)
        for match in matches_list_filtered:
            #print(match)
            if match.lower() in domain_match.lower():
                return re.sub(regExp, '', title)
        return title
    else:
        return title
```

Función con expresiones regulares para extracción de patrón *Page* + *número* de títulos

```
    def findPage2(string):
        numbers=["one","two","three","four","five","six","seven","eight","nine","ten"]

        letterFound = re.search(r'page\s+([a-z]+)', string, re.I)
        if(letterFound):
            return letterFound.group() if letterFound.group(1).lower() in numbers else None
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

```
    elif re.search(r'page\s+\d+', string, re.I):
        return re.search(r'page\s+\d+', string, re.I).group()
    else:
        return None
```

Función de solicitud de traducción

```
from google.cloud import translate


def batch_translate_text(inputFileName, outputFolder):
    """Translates a batch of texts on GCS and stores the result in a GCS location."""

    client = translate.TranslationServiceClient()

    location = "us-central1"

    input_configs_element = [{
        "gcs_source": {"input_uri": "gs://felix-tesis-translation/"+inputFileName+".tsv"},
        "mime_type": "text/plain"  # Can be "text/plain" or "text/html".
    }]
    print(input_configs_element)

    output_config_element = {
        "gcs_destination": {"output_uri_prefix": "gs://felix-tesis-translation/"+outputFolder+"/"}
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
}
print(output_config_element)



request = {
    "parent": "projects/felix-obando-tesis/locations/us-central1",
    "source_language_code":"en", #"es" for back translation
    "target_language_codes":["es"], #["en"] for back translation
    "input_configs":input_configs_element,
    "output_config":output_config_element
}

operation = client.batch_translate_text(
    request = request,
    timeout = 180
)

print(u"Waiting for operation to complete...")
response = operation.result(180)

print(u"Total Characters: {}".format(response.total_characters))
print(u"Translated Characters: {}".format(response.translated_characters))
```

Función de extracción de características

```
def round_prox(number):
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

```
    #round_prox rounds the number to the closest two decimal place
    return math.floor(number*100 + 0.5)/100


def spacyProcess(row):
    #We use this number to count how many rows we are applying
    global number
    number = number + 1
    print(number)
    title = row['traducción']


    #We only use NER and POS pipelines
    doc = nlp(title, disable=['parser'])


    #We count the pos attributes
    count_raw_POS = doc.count_by(spacy.attrs.POS)
    countList_POS = {doc.vocab[k].text: v for k, v in count_raw_POS.items() if
doc.vocab[k].text in [\
                'ADJ','ADP','ADV','DET','NOUN','PROPN','PRON','VERB']}
    #Total number of POS
    nPOS = sum([v for k, v in count_raw_POS.items()])
    #We check variety of POS
    pos_diversity = len(countList_POS)


    #We count the NER attributes
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

```
        count_raw_NER = [ent.label_ for ent in doc.ents if ent.label_ in ["PER", "LOC",
"ORG"]]
        countList_NER = {\
                "PER": sum(map(lambda a: 1 if a=="PER" else 0, count_raw_NER)),\
                "LOC": sum(map(lambda a: 1 if a=="LOC" else 0, count_raw_NER)),\
                "ORG": sum(map(lambda a: 1 if a=="ORG" else 0, count_raw_NER))}

        #We count total number of words
        words = [token.text for token in doc if token.is_alpha]

        #We analyze sentiment
        text = Text(title, hint_language_code="es")

        series = pd.Series(\
            {'number_words': len(words), \
            'avg_word_size': round_prox(sum([len(word) for word in
words])/len(words)),\
            'ratio_adj': round_prox(countList_POS.get('ADJ')/nPOS) if
countList_POS.get('ADJ') else 0,\
            'ratio_adp': round_prox(countList_POS.get('ADP')/nPOS) if
countList_POS.get('ADP') else 0,\
            'ratio_adv': round_prox(countList_POS.get('ADV')/nPOS) if
countList_POS.get('ADV') else 0,\
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
                'ratio_det':            round_prox(countList_POS.get('DET')/nPOS)          if
countList_POS.get('DET') else 0,\
                'ratio_noun':           round_prox(countList_POS.get('NOUN')/nPOS)         if
countList_POS.get('NOUN') else 0,\
                'ratio_propn':          round_prox(countList_POS.get('PROPN')/nPOS)        if
countList_POS.get('PROPN') else 0,\
                'ratio_pron':           round_prox(countList_POS.get('PRON')/nPOS)         if
countList_POS.get('PRON') else 0,\
                'ratio_verb':           round_prox(countList_POS.get('VERB')/nPOS)         if
countList_POS.get('VERB') else 0,\
                'ratio_ner_per':         round_prox(countList_NER.get('PER')/nPOS)         if
countList_NER.get('PER') else 0,\
                'ratio_ner_loc':         round_prox(countList_NER.get('LOC')/nPOS)         if
countList_NER.get('LOC') else 0,\
                'ratio_ner_org':         round_prox(countList_NER.get('ORG')/nPOS)         if
countList_NER.get('ORG') else 0,\
                'ratio_pos_diversity': round_prox(pos_diversity/nPOS),\
                 'polarity': round_prox((text.polarity))})


    return series
```

Vectorizador de palabras

```
def round_prox(number):
    #round_prox rounds the number to the closest three decimal place
    return math.floor(number*1000 + 0.5)/1000
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
def singleSampleProcess(sampleText, spacy_model_instance, pos_and_ner=True, sentiment=True):
    title = sampleText

    #We only use NER and POS pipelines
    doc = spacy_model_instance(title, disable=['parser'])

    #We count the pos attributes
    count_raw_POS = doc.count_by(spacy.attrs.POS)
    countList_POS = {doc.vocab[k].text: v for k, v in count_raw_POS.items() if doc.vocab[k].text in [\
            'ADJ','ADP','ADV','DET','NOUN','PROPN','PRON','VERB']}
    #Total number of POS
    nPOS = sum([v for k, v in count_raw_POS.items()])
    #We check variety of POS
    pos_diversity = len(countList_POS)

    #We count the NER attributes
    count_raw_NER = [ent.label_ for ent in doc.ents if ent.label_ in ["PER", "LOC", "ORG"]]
    countList_NER = {\
            "PER": sum(map(lambda a: 1 if a=="PER" else 0, count_raw_NER)),\
            "LOC": sum(map(lambda a: 1 if a=="LOC" else 0, count_raw_NER)),\
            "ORG": sum(map(lambda a: 1 if a=="ORG" else 0, count_raw_NER))}
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
        #We count total number of words
        words = [token.text for token in doc if token.is_alpha]

        #We analyze sentiment
        text = Text(title, hint_language_code="es")

        vector = doc.vector

        #If we want to include pos_and_ner analysis
        if pos_and_ner:
            pos_ner_array = [len(words), \
                round_prox(sum([len(word) for word in words])/len(words)),\
                round_prox(countList_POS.get('ADJ')/nPOS)  if  countList_POS.get('ADJ')
else 0,\
                round_prox(countList_POS.get('ADP')/nPOS)  if  countList_POS.get('ADP')
else 0,\
                round_prox(countList_POS.get('ADV')/nPOS)  if  countList_POS.get('ADV')
else 0,\
                round_prox(countList_POS.get('DET')/nPOS)  if  countList_POS.get('DET')
else 0,\
                round_prox(countList_POS.get('NOUN')/nPOS)                    if
countList_POS.get('NOUN') else 0,\
                round_prox(countList_POS.get('PROPN')/nPOS)                   if
countList_POS.get('PROPN') else 0,\
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
                round_prox(countList_POS.get('PRON')/nPOS)                    if
countList_POS.get('PRON') else 0,\
                round_prox(countList_POS.get('VERB')/nPOS)                    if
countList_POS.get('VERB') else 0,\
                round_prox(countList_NER.get('PER')/nPOS)  if  countList_NER.get('PER')
else 0,\
                round_prox(countList_NER.get('LOC')/nPOS)  if  countList_NER.get('LOC')
else 0,\
                round_prox(countList_NER.get('ORG')/nPOS)  if  countList_NER.get('ORG')
else 0,\
                round_prox(pos_diversity/nPOS)]
            vector = np.concatenate((vector, pos_ner_array))

        #If we want to include sentiment analysis
        if sentiment:
            vector = np.concatenate((vector, [round_prox((text.polarity))]))

        return vector.reshape(1,-1)
```

Clase para concatenación y adecuación de vectores de muestras

```
class WordVectorizer(TransformerMixin, BaseEstimator):
    def __init__(self, model="es_core_news_lg"):
        self.model = model

    def fit(self, X, pos_and_ner=True, sentiment=True, y=None):
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
        self.pos_and_ner = pos_and_ner
        self.sentiment = sentiment
        return self

    def transform(self, X):
        nlp = spacy.load(self.model)
        return np.concatenate([singleSampleProcess(doc, nlp, self.pos_and_ner,
self.sentiment) for doc in X])
```

Main.py

```
from flask import Flask,jsonify,request
from flask_cors import CORS

from processing.data_management import load_pipeline
import joblib
import pickle

from polyglot.downloader import downloader
from shutil import copyfile
from polyglot.text import Text
import os
from polyglot.mapping import embeddings


app = Flask(__name__)
CORS(app)
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
fake_news_trained_pipeline = load_pipeline()


@app.route((/", methods=['POST','GET'])
def index():
    if(request.method == 'POST'):
        data = request.get_json()
        print(data)
        title = str(data["title"])
        prediction = "falsa" if bool(fake_news_trained_pipeline.predict([title])[0]) else
"confiable"
        print("pred: "+ prediction)
        ans = jsonify({"pred": prediction})
        print(ans)
        return ans
    else:
        return jsonify({"about":"Hello World"})


if __name__ == '__main__':
    app.run(debug=True)
```

pipeline.py

```
from sklearn.pipeline import Pipeline
from sklearn.svm import LinearSVC


import processing.transformer as tr
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA


```
fake_news_pipeline = Pipeline(
  [
    ('vectorization', tr.WordVectorizerPipeline()),
    ('classification', LinearSVC(dual=False))
  ]
)

if __name__=='__main__':
  print("pipeline main")
else:
  print("importing pipeline.py as aux")
```

predict.py

```
import numpy as np
import pandas as pd
from polyglot.text import Text
from polyglot.downloader import downloader as pl_download


from processing.data_management import load_pipeline

fake_news_trained_pipeline = load_pipeline()

def make_prediction(news_title):
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
    prediction = fake_news_trained_pipeline.predict([news_title])
    print(prediction)
    results = {
        'prediction': prediction,
    }

    return results
def make_test_prediction():
    text_title="Fuentes cercanas al presidente reportan apocalipsis"
    prediction = fake_news_trained_pipeline.predict([text_title])
    print(prediction)
    results = {
        'prediction': prediction,
    }
    return results
if __name__=='__main__':
    make_test_prediction()
```

train_pipeline.py

```
#Import Libraries
import pandas as pd
import numpy as np


#Import other files/modules
from processing.data_management import load_dataset, save_pipeline
import processing.transformer as tr
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
import pipeline



def run_training():
    """Train the model"""

    #Read Data
    data = load_dataset()
    #Training pipeline
    pipeline.fake_news_pipeline.fit(data["traducción"].values, data["fake"].values)
    save_pipeline(pipeline_to_save=pipeline.fake_news_pipeline)


if __name__=='__main__':
    run_training()
    print("pipeline trained")
```

transformer.py

```
import math
import spacy
import numpy
from polyglot.text import Text
from sklearn.base import BaseEstimator, TransformerMixin
from shutil import copyfile
import os
from polyglot.downloader import downloader
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
        def round_prox(number):
            #round_prox rounds the number to the closest three decimal place
            return math.floor(number*1000 + 0.5)/1000

        def singleSampleProcessPipeline(sampleText, spacy_model_instance):
            title = sampleText

            #We only use NER and POS pipelines
            doc = spacy_model_instance(title, disable=['parser'])

            #We count the pos attributes
            count_raw_POS = doc.count_by(spacy.attrs.POS)
            countList_POS = {doc.vocab[k].text: v for k, v in count_raw_POS.items() if
doc.vocab[k].text in [\
                    'ADJ','ADP','ADV','DET','NOUN','PROPN','PRON','VERB']}
            #Total number of POS
            nPOS = sum([v for k, v in count_raw_POS.items()])
            #We check variety of POS
            pos_diversity = len(countList_POS)

            #We count the NER attributes
            count_raw_NER = [ent.label_ for ent in doc.ents if ent.label_ in ["PER", "LOC",
"ORG"]]
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
countList_NER = {\
        "PER": sum(map(lambda a: 1 if a=="PER" else 0, count_raw_NER)),\
        "LOC": sum(map(lambda a: 1 if a=="LOC" else 0, count_raw_NER)),\
        "ORG": sum(map(lambda a: 1 if a=="ORG" else 0, count_raw_NER))}

#We count total number of words
words = [token.text for token in doc if token.is_alpha]

#We analyze sentiment
text = Text(title.lower(), hint_language_code="es")

vector = doc.vector

#If we want to include pos_and_ner analysis
pos_ner_array = [len(words), \
    round_prox(sum([len(word) for word in words])/len(words)),\
    round_prox(countList_POS.get('ADJ')/nPOS) if countList_POS.get('ADJ') else
0,\
    round_prox(countList_POS.get('ADP')/nPOS) if countList_POS.get('ADP') else
0,\
    round_prox(countList_POS.get('ADV')/nPOS) if countList_POS.get('ADV') else
0,\
    round_prox(countList_POS.get('DET')/nPOS) if countList_POS.get('DET') else
0,\
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
                round_prox(countList_POS.get('NOUN')/nPOS) if countList_POS.get('NOUN')
else 0,\
                round_prox(countList_POS.get('PROPN')/nPOS) if
countList_POS.get('PROPN') else 0,\
                round_prox(countList_POS.get('PRON')/nPOS) if countList_POS.get('PRON')
else 0,\
                round_prox(countList_POS.get('VERB')/nPOS) if countList_POS.get('VERB')
else 0,\
                round_prox(countList_NER.get('PER')/nPOS) if countList_NER.get('PER') else
0,\
                round_prox(countList_NER.get('LOC')/nPOS) if countList_NER.get('LOC') else
0,\
                round_prox(countList_NER.get('ORG')/nPOS) if countList_NER.get('ORG')
else 0,\
                round_prox(pos_diversity/nPOS)]
            vector = numpy.concatenate((vector, pos_ner_array))

            #If we want to include sentiment analysis
            vector = numpy.concatenate((vector, [round_prox((text.polarity))]))

            return vector.reshape(1,-1)

        class WordVectorizerPipeline(TransformerMixin, BaseEstimator):
            def __init__(self, model="es_core_news_md"):
                self.model = model
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
    def fit(self, X, y=None):
        return self


    def transform(self, X):
        print("loading spacy model before transforming")
        #Evaluating if spacy spanish model exists
        nlp = spacy.load(self.model)
        print("evaluating sentiment2 model before transforming")
        #Evaluating if polyglot sentiment module exist
        try:
            Text("Evaluando existencia de sentiment2.es",hint_language_code="es").polarity
        except:
            #Destination where we will copy sentiment2 pickle
            #This code of exception is designed to work on google app engine with /code/ as python root
            #Comment it and install package manually for other platforms
            print("sentiment2 not found. Copying...")
            parent_dest=str(downloader.default_download_dir())

            if not os.path.isdir(parent_dest + "/sentiment2/es"):
                createdPath1=parent_dest+"/sentiment2/es"
                os.makedirs(createdPath1)
```

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE TESIS UCSM


```
                copyfile(str(os.getcwd())+"/processing/es.sent.pkl.tar.bz2",
createdPath1+"/es.sent.pkl.tar.bz2")
                print("copy finished")
            print("transforming")
            return numpy.concatenate([singleSampleProcessPipeline(doc, nlp) for doc in X])
```

data_management.py

```
import pandas as pd
import joblib
import os


def load_dataset():
    data = pd.read_csv("datasets/cleaned_train_test_dataset.csv", index_col=0)
    return data


def save_pipeline(pipeline_to_save):
    save_path = "trained_models/fake_news_model.pkl"
    joblib.dump(pipeline_to_save, save_path)
    print("Saved Pipeline")



def load_pipeline():
    save_path = "trained_models/fake_news_model.pkl"
    trained_model = joblib.load(save_path)
    print("Pipeline loaded")
    return trained_model
```

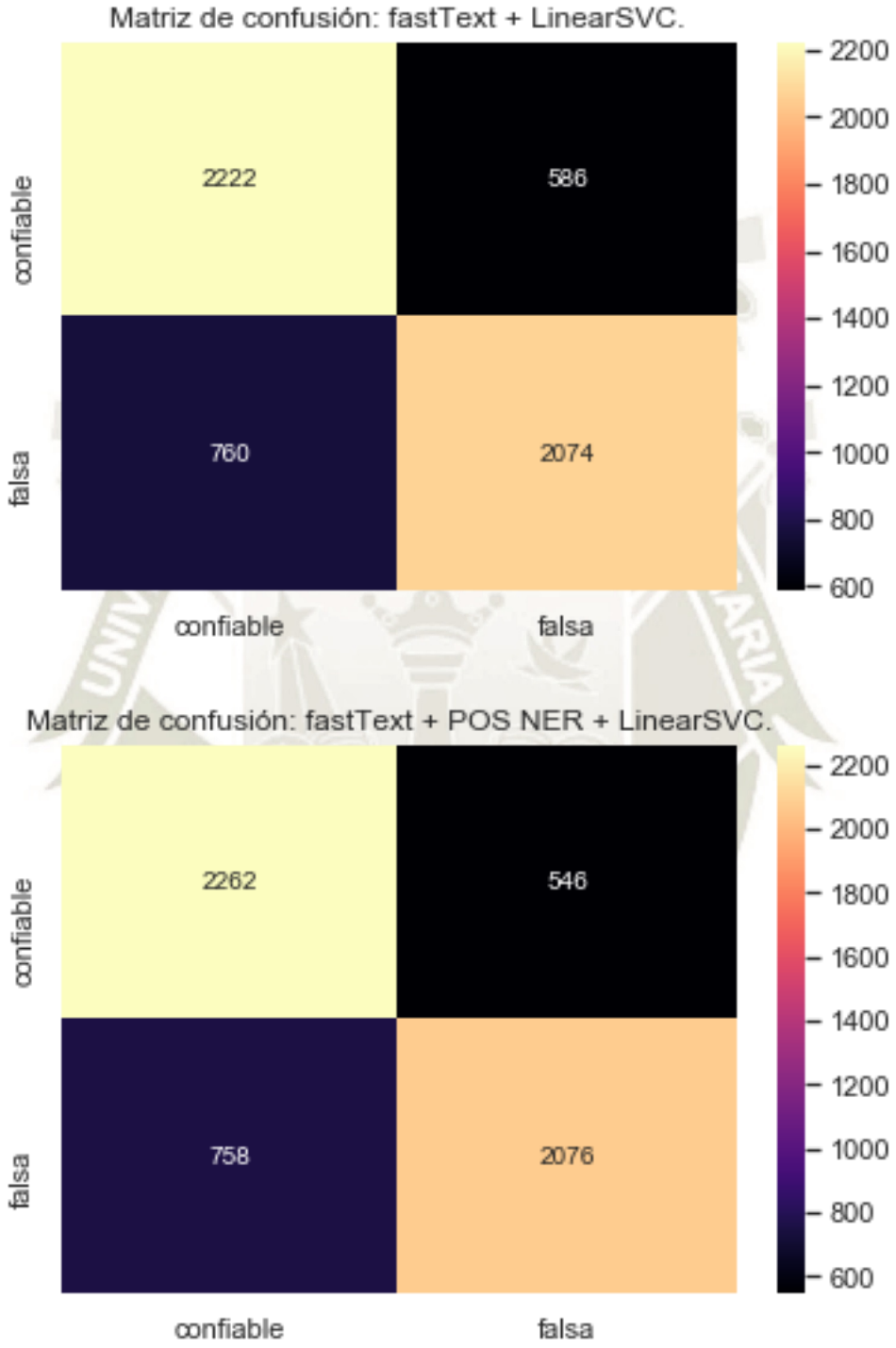
Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Matrices de confusión para evaluación de modelos

Matrices de confusión para modelos de máquinas de soporte vectorial



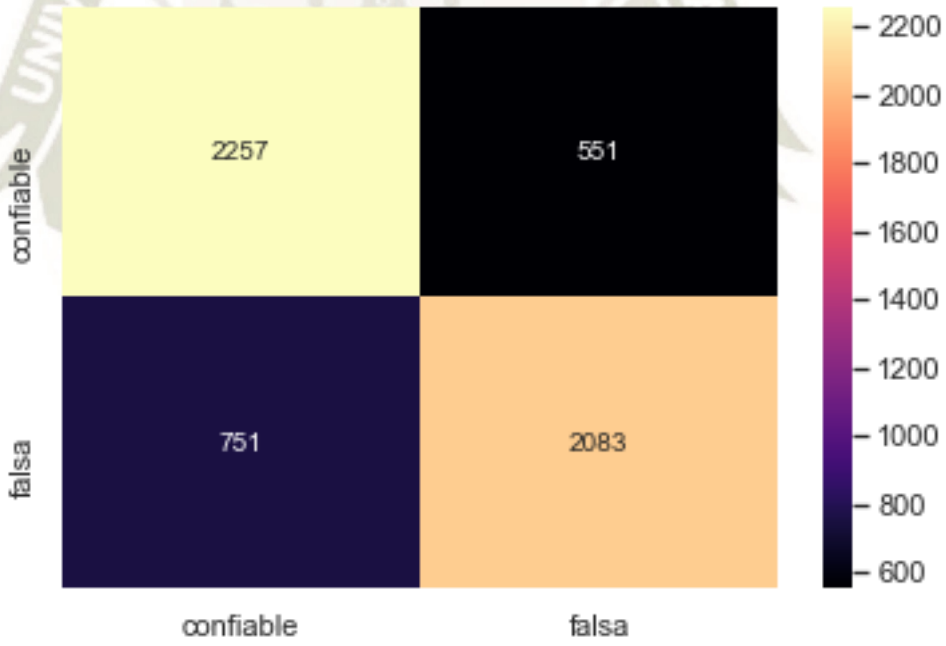
Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Matriz de confusión: fastText + sentiment + LinearSVC.

Matriz de confusión: fastText + POS NER + sentiment + LinearSVC.

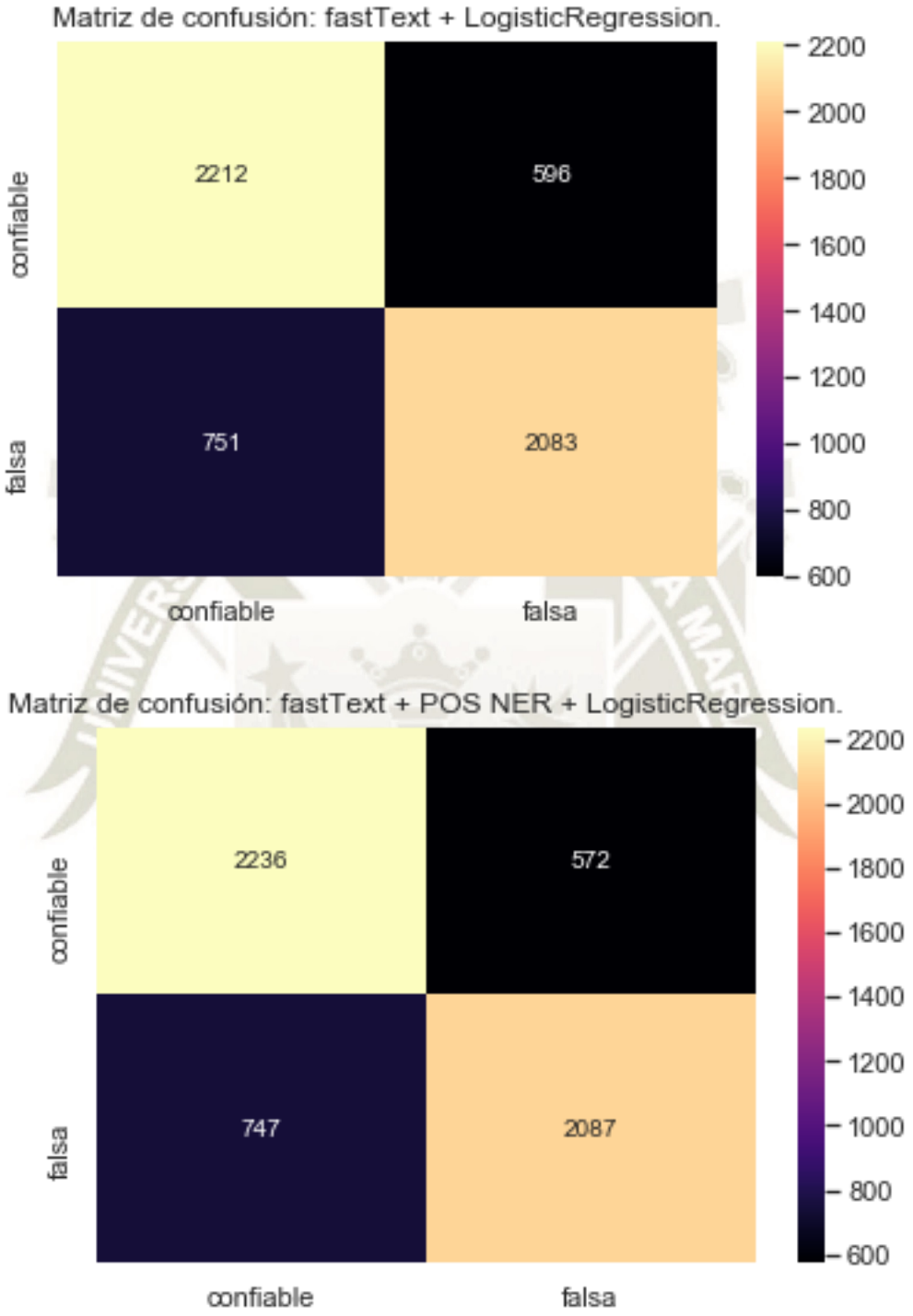


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

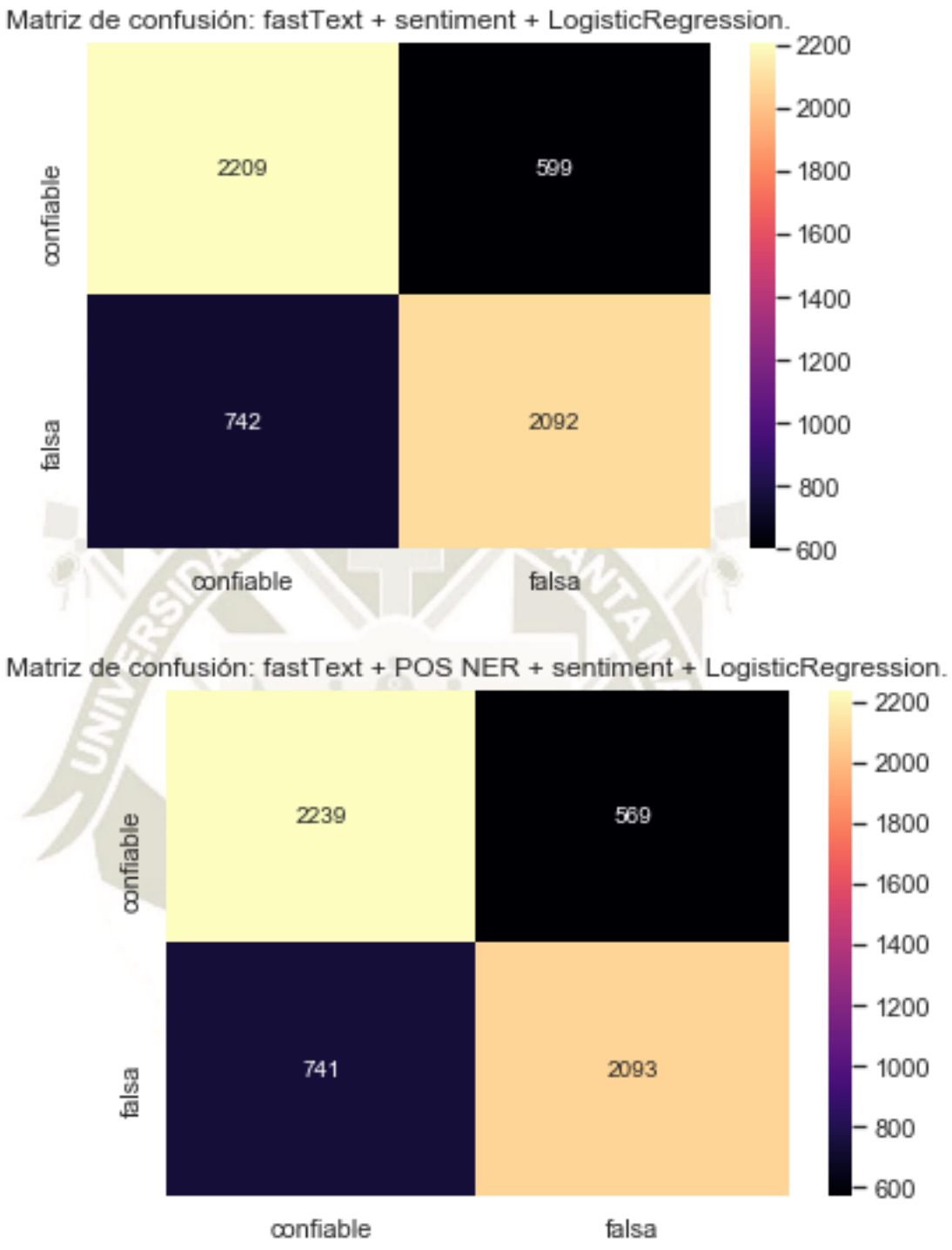
UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Matrices de confusión para modelos de Regresión Logística



Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM



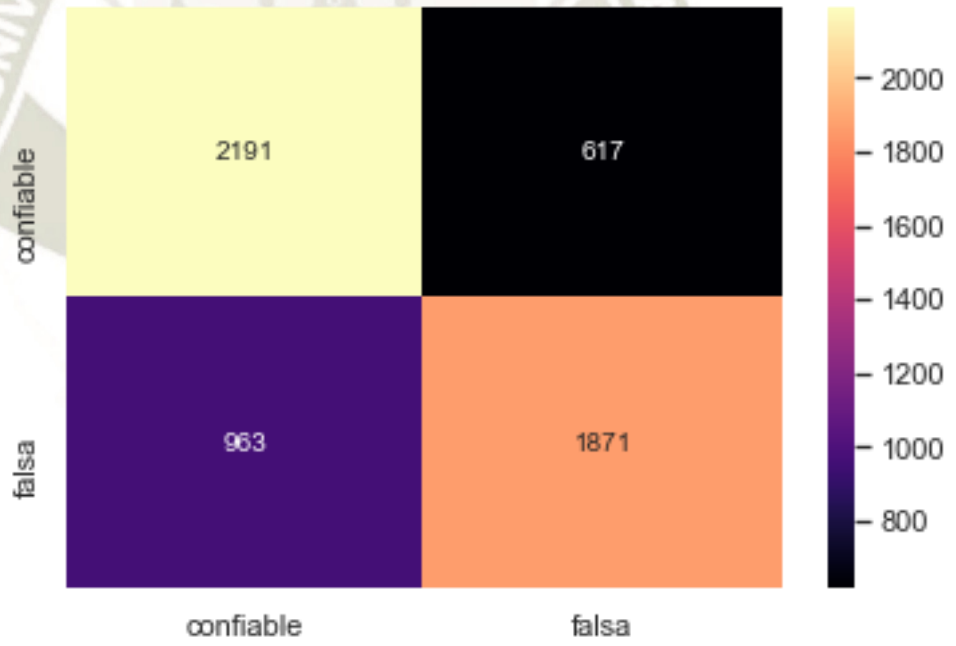
Matrices de confusión para modelos de Bosques aleatorios: 100 árboles

Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

Matriz de confusión: fastText + RandomForestClassifier - 100.

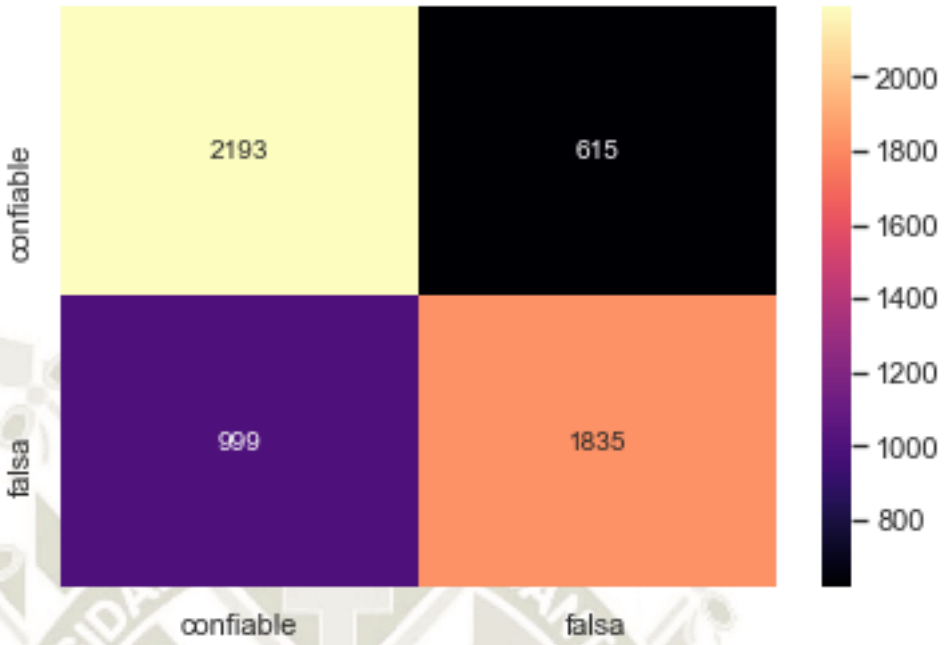
Matriz de confusión: fastText + POS NER + RandomForestClassifier - 100.



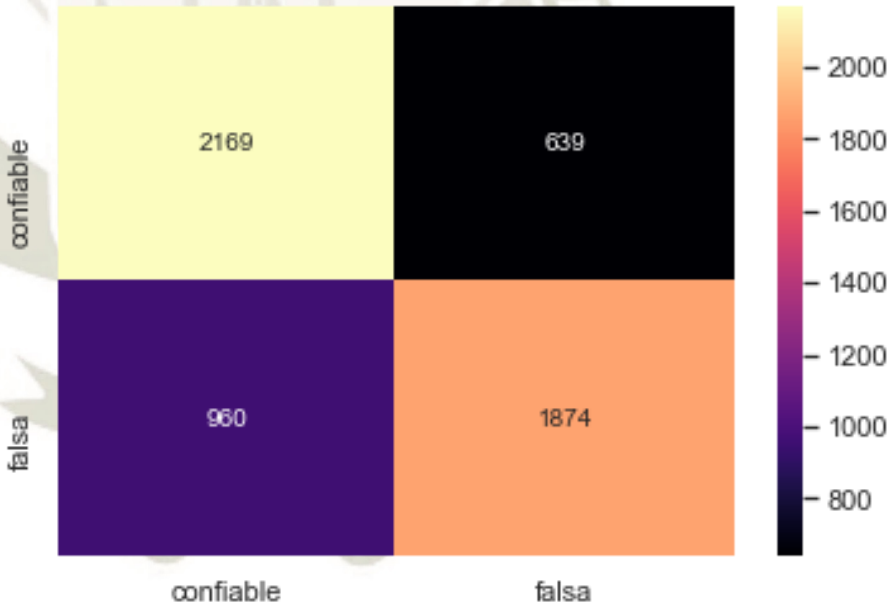
Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

Matriz de confusión: fastText + sentiment + RandomForestClassifier - 100.



Matriz de confusión: fastText + POS NER + sentiment + RandomForestClassifier - 100.

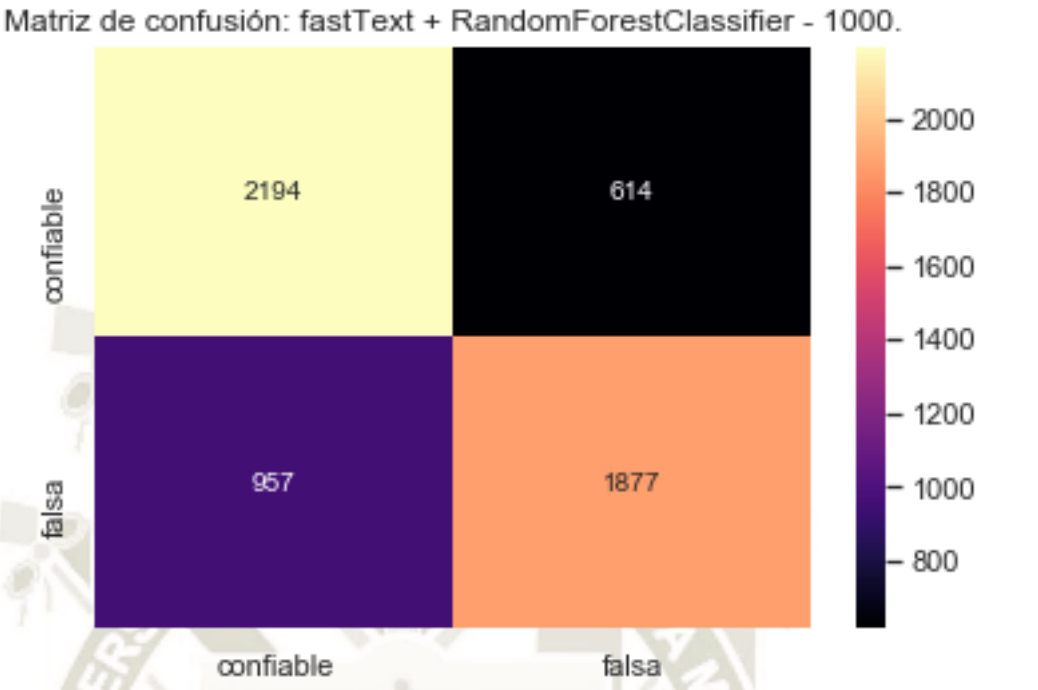


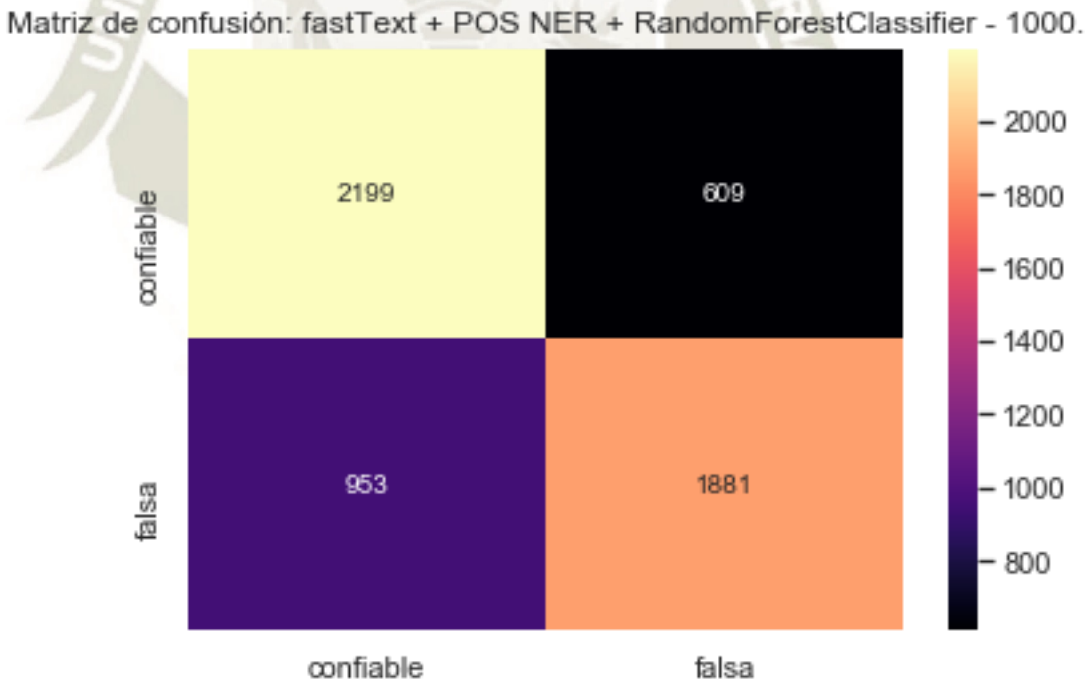
Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

Matrices de confusión para *modelos de Bosques aleatorios: 1000 árboles*

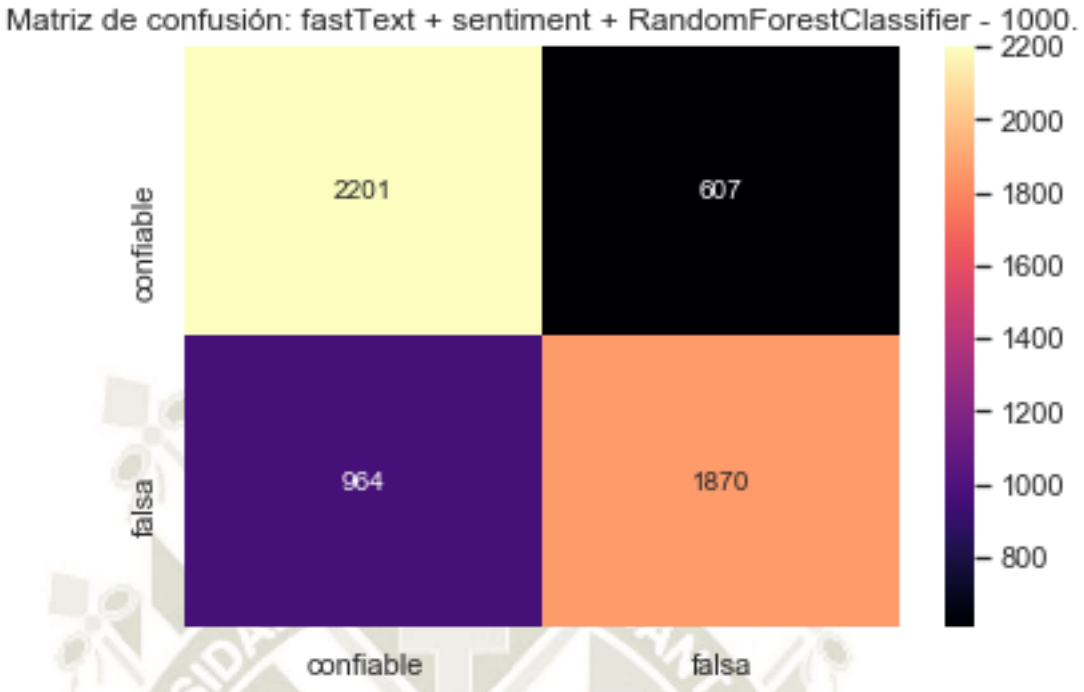


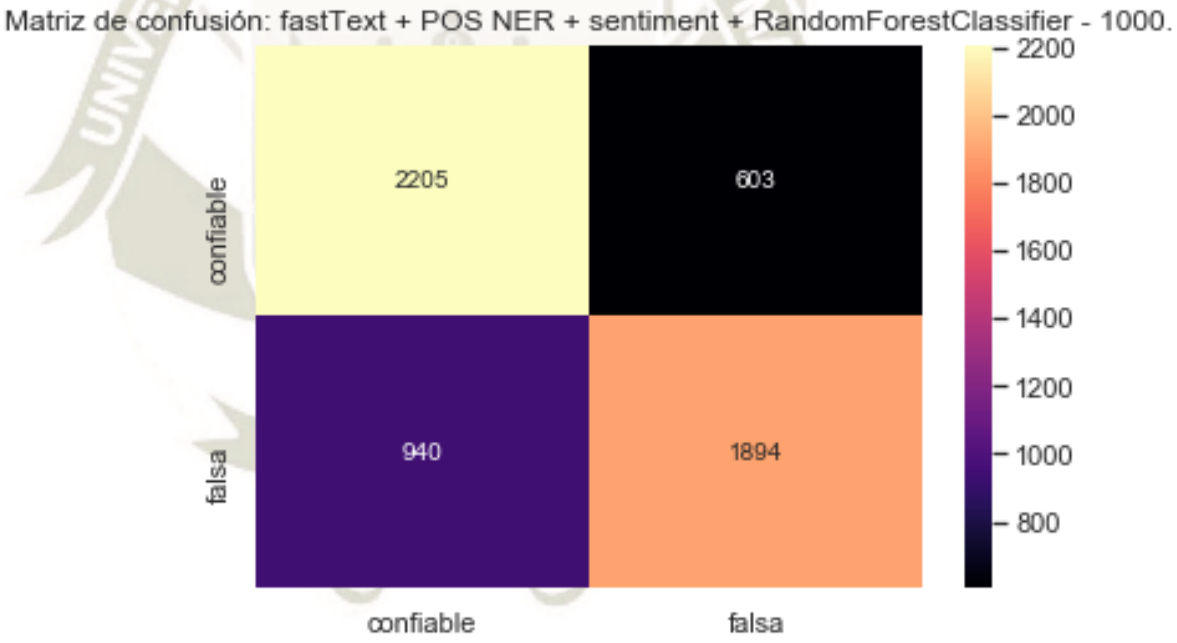


Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

UNIVERSIDAD
CATÓLICA
DE SANTA MARÍA

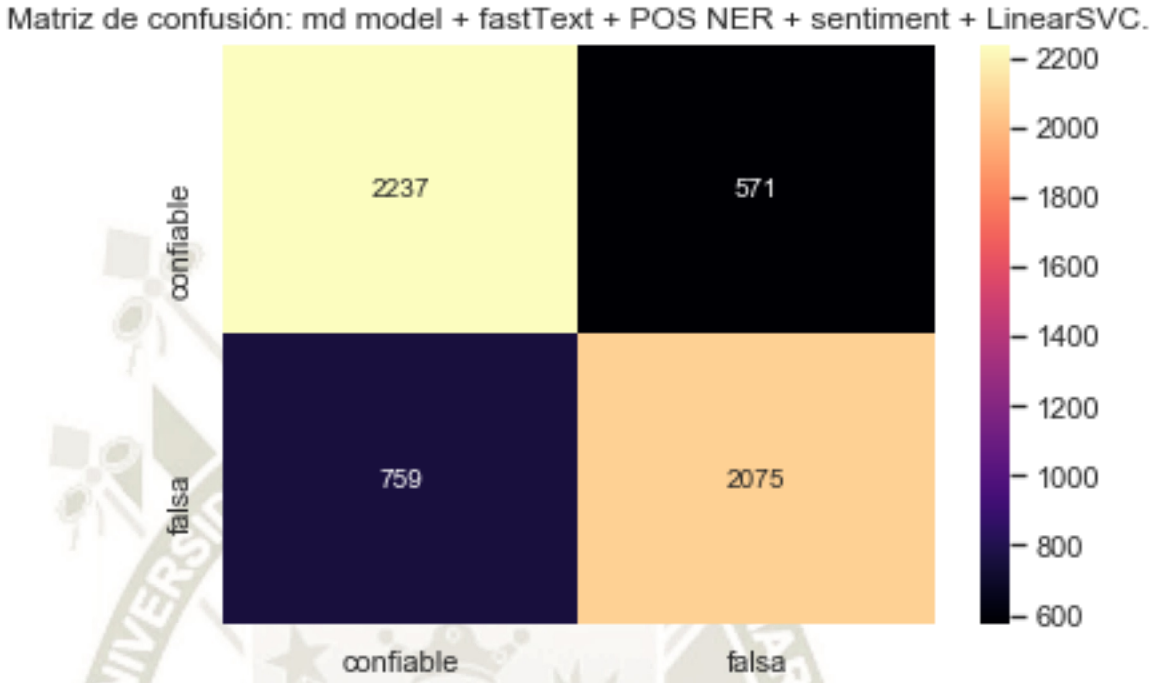




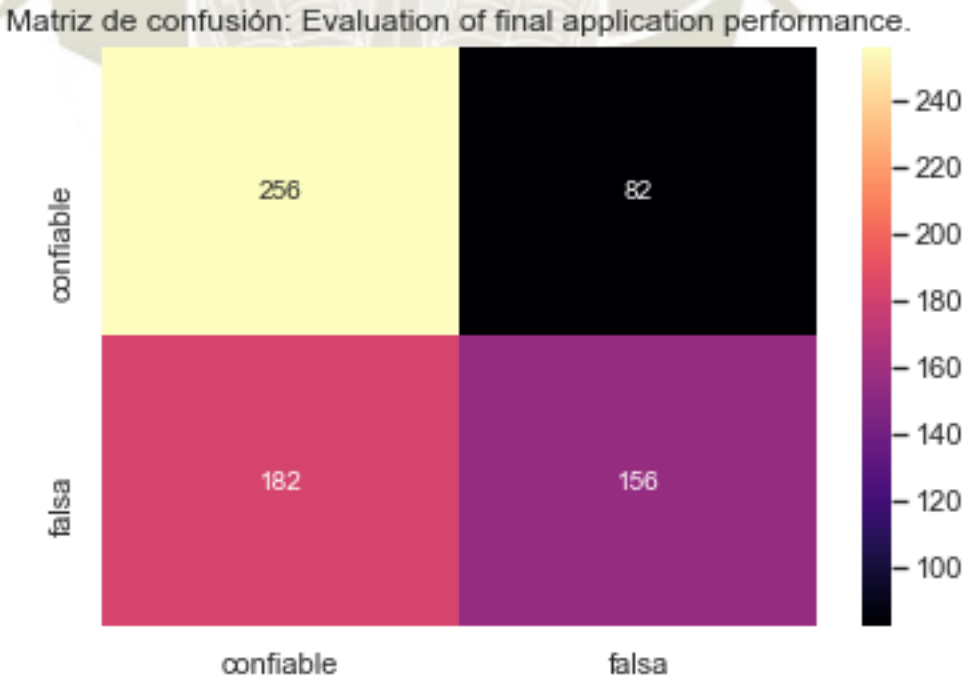
Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis

REPOSITORIO DE
TESIS UCSM

Matriz de confusión para modelo de máquinas de soporte vectorial con extracción de características por medio de modelo *es_core_news_md*



Matriz de confusión para aplicación implementada



Publicación autorizada con fines académicos e investigativos
En su investigación no olvide referenciar esta tesis