Instituto Tecnológico y de Estudios Superiores de Monterrey

Análisis de métodos de razonamiento e incertidumbre
MA2014.101

PBL1. Sistema en Python que detecta SPAM

Kevin Antonio González Díaz / A01338316
Karla Andrea Palma Villanueva / A01754270
Daniela Márquez Campos / A00833345
Julio Eugenio Guevara Galván / A01704733
Adrian Pineda Sánchez / A00834710
David Fernando Armendáriz Torres/ A01570813

Docente: Daniel Otero Fadul

Monterrey, Nuevo León, México. 20 agosto 2023

Índice

1. Problematización **2**

2. Enfoque **2**

3. Propósito **3**

4. Información **3**
4.1. Machine Learning en modelos supervisados 3
4.2. Procesamiento de Lenguaje Natural (NLP) 5
4.3. Teorema de Bayes 5

5. Razonamiento **6**

6. Conclusiones **8**

Referencias **9**

1. Problematización

La mayoría de las personas invierten una cantidad considerable de tiempo cada día en diferenciar entre los correos electrónicos que no desean recibir (spam) y aquellos que son útiles y relevantes para ellos. Tienen que revisar sus bandejas de entrada y decidir qué correos electrónicos deben abrir y leer, y cuáles pueden ignorar o eliminar sin revisar [1]. El proceso de filtrar el contenido no deseado del contenido valioso puede ser un tanto tedioso y consumir tiempo.

Los mensajes de spam constituyeron más del 45 % del tráfico total de correos electrónicos en diciembre de 2022 . Durante ese mismo año, Rusia fue responsable de la mayor parte de estos mensajes no deseados, contribuyendo con el 29.82 % del total de spam a nivel mundial [2].

El phishing de sitios web se considera como uno de los desafíos primordiales de seguridad para la comunidad en línea, dada la considerable cantidad de transacciones en línea que ocurren diariamente [3]. Esta forma de suplantación consiste en copiar un sitio web confiable para obtener información delicada de usuarios en línea, como sus nombres de usuario y contraseñas. Medidas como listas de bloqueo, listas de permitidos y el uso de técnicas de búsqueda son ejemplos de enfoques para minimizar el riesgo asociado con este problema. En promedio, alrededor del 27 % de las empresas a nivel global experimentan entre cuatro y seis ataques cibernéticos exitosos en un año [2].

Por lo mencionado anteriormente se destaca la importancia de detectar correctamente un correo spam. Un preciso filtrado de correos tiene un impacto positivo en la seguridad, eficiencia y experiencia de usuario en el entorno de comunicación en línea, al tiempo que contribuye a la protección de datos y la mitigación de riesgos cibernéticos.

2. Enfoque

Para resolver este problema nos centramos en la creación de un sistema en Python que tenga la capacidad de detectar correos electrónicos spam de manera automatizada. Para abordar el problema de la identificación de correos no deseados y ahorrar tiempo a los usuarios, se ha empleado un enfoque basado en el uso de machine learning, específicamente utilizando la teoría de probabilidad de Naive Bayes.

El código recibe el dataset 'SMS Spam Collection Dataset' [4]. Este conjunto de datos compuesto por 5,574 mensajes en inglés, los cuales han sido etiquetados como "ham"(legítimos) o "spam". La recopilación de estos datos proviene de diversas fuentes, tales como Grumbletext, ciudadanos de Singapur, la tesis de doctorado de Caroline Tag, entre otras.

En el proceso, cada mensaje será transformado en un conjunto de palabras relevantes que se utilizarán para su análisis. Estas palabras formarán un vector que representa el contenido del mensaje. Luego, se llevará a cabo un análisis para determinar la probabilidad de que el mensaje sea considerado como spam. Esta probabilidad se calculará mediante la aplicación del Teorema de Bayes, un enfoque probabilístico utilizado para estimar la probabilidad de un evento en función de evidencia previa o conocimiento relevante. En este contexto, el Teorema de Bayes permitirá calcular la probabilidad de que un mensaje sea spam basándose en la información extraída de las palabras presentes en el mensaje y su relación con mensajes previamente etiquetados como spam o ham.

3. Propósito

El objetivo es lograr una identificación precisa y eficiente de los correos no deseados para evitar que lleguen a las bandejas de entrada de los usuarios y, de esta manera, mejorar la experiencia y la seguridad en la comunicación por correo electrónico. Un filtrado efectivo de spam asegura que los usuarios reciban únicamente correos electrónicos relevantes y legítimos, lo que mejora la eficiencia en la comunicación. Al evitar la inundación de bandejas de entrada con contenido no deseado, se ahorra tiempo y se facilita la gestión de la información.

Además, detectar correctamente el spam protege a los usuarios de caer en trampas cibernéticas y de ser víctimas de ataques informáticos. Se busca evitar que los usuarios compartan datos sensibles con entidades no confiables.

4. Información

4.1. Machine Learning en modelos supervisados

El aprendizaje automático, también conocido como machine learning en inglés, es una rama de la inteligencia artificial que se centra en el desarrollo de algoritmos y sistemas capaces de aprender y mejorar su rendimiento a partir de datos, en lugar de ser programados de manera explícita para realizar tareas específicas [6].

En el ámbito de la detección de correo no deseado, un sistema de aprendizaje automático puede ser entrenado con ejemplos de correos electrónicos etiquetados como spam y no spam. El sistema aprenderá a reconocer las características y patrones que suelen estar presentes en los correos electrónicos de spam, como ciertas palabras clave o estructuras de texto engañosas.

El inicio del proceso de creación de un modelo de Aprendizaje Automático es entender el problema, organizar y colectar los datos, y luego supervisar y utilizar los resultados de la solución para ajustar el modelo como se observa en la Figura 1 y el Cuadro 1.

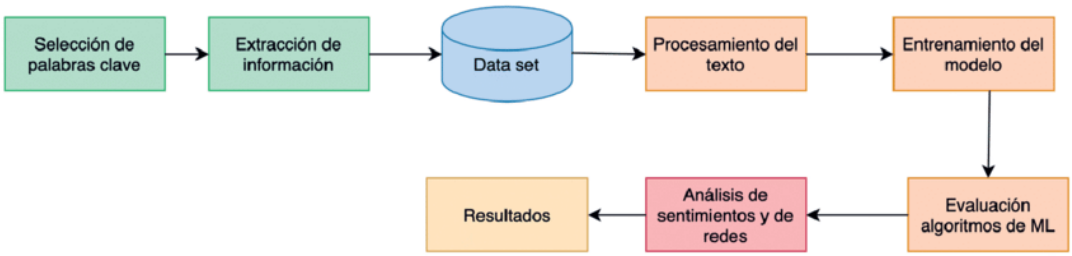

Figura 1: Diagrama de flujo de proceso de un Modelo de Aprendizaje Automático [10]

El rendimiento del modelo se puede mostrar en una matriz de confusión, en la cual se definen distintos escenarios. Aquellos en los que el modelo acierta en predecir la clase positiva de un elemento del conjunto de datos se denominan Verdaderos Positivos (TP). Si el modelo acierta en predecir la clase negativa para un elemento, estos casos son denominados Verdaderos Negativos (TN). Por otro lado, si el modelo predice erróneamente la clase positiva para un elemento que en realidad pertenece a la clase negativa, se identifican como Falsos Positivos (FP). Finalmente, si el modelo predice erróneamente la clase negativa para un elemento que en realidad pertenece a la clase positiva, se denominan Falsos Negativos (FN).

Se dispone de una variedad de métricas que pueden ser implementadas, las cuales dependen tanto de las características del conjunto de datos como de los aspectos específicos que se deseen evaluar. A continuación se presentan las métricas implementadas:

- Accuracy (acc). Mide la proporción de predicciones correctas sobre el número total de instancias evaluadas ($\frac{TP+TN}{TP+FP+TN+FN}$)
- Precision (p). Se utiliza para medir los patrones positivos que son predichos correctamente entre el total de patrones predichos en una clase positiva ($\frac{TP}{TP+FP}$)
- Recall (r). Mide la fracción de patrones positivos que son clasificados correctamente ($\frac{TP}{TP+TN}$)
- F-Measure (FM). Representa la media armónica entre los valores de sensibilidad y precisión. ($\frac{2*p*r}{p+r}$)

Cuadro 1: Fases en la Creación de un Modelo de Aprendizaje Automático [7]

Etapa	**Explicación**
Objetivo	Determinar la finalidad del modelo a construir: predecir o clasificar
Datos	Recolectar información que el algoritmo pueda usar para aprender y estructurar esta información en un formato adecuado destacando aspectos significativos y reduciendo la complejidad del conjunto de datos
Modelamiento	Seleccionar el enfoque estadístico y la metodología de aprendizaje automático más adecuados para resolver la cuestión
Entrenamiento	Fase en la que el algoritmo de machine learning internaliza y asimila conocimiento a partir de los datos recopilados y preparados
Evaluación	Evaluar el rendimiento del modelo para determinar su eficacia y eficiencia en su operación
Predicción	Evaluar el desempeño del modelo mediante la aplicación de un conjunto de datos nuevo y no visto previamente
Ajuste	Supervisar los resultados con el fin de afinar el modelo y optimizar su desempeño al máximo

Para la realización del código se empleó el entorno de programación Jupyter Notebook con el lenguaje de programación Python como base. Para llevar a cabo el procesamiento de los datos y la implementación de las técnicas, se importaron varias bibliotecas esenciales como 'Pandas', la que permitió la gestión eficiente de los datos en forma de tablas, facilitando la limpieza, transformación y análisis exploratorio de los datos. La biblioteca 're' fue utilizada para realizar búsquedas y manipulaciones de patrones en los textos, contribuyendo al proceso de extracción de información relevante. Así como la biblioteca NumPy, que se empleó para realizar operaciones matemáticas y estadísticas en matrices y arreglos multidimensionales, lo que facilitó el cálculo eficiente de métricas y características. Para la manipulación y el análisis de lenguaje natural, y hacer el preprocesamiento de los textos y la extracción de información relevante se usó Natural Language Toolkit (NLTK).

4.2. Procesamiento de Lenguaje Natural (NLP)

Actualmente existe una cantidad de información abundante, la cual, se encuentra almacenada en datos que requieren de un análisis efectivo para así recabar los conocimientos valiosos y lograr generar el respaldo de una toma de decisiones asertiva y precisa. En este contexto, las técnicas de Ciencias de Datos emergen como herramientas fundamentales para desarrollar patrones y visualizar tendencias, relaciones o información clave oculta en los datos.

Un ejemplo claro de dichas técnicas es el Procesamiento de Lenguaje Natural (NLP), cuya función destaca en el análisis de texto y permite a las computadoras comprender e interpretar el lenguaje humano. Es una tecnología de machine learning que procesa automáticamente los datos y analiza la intención o el sentimiento del mensaje [8]

Por lo general, las técnicas implementadas por NLP se concentran primeramente en la preparación de los datos a través de técnicas de preprocesamiento como la normalización de los tokens a través de stemming y lemmatization y la eliminación de palabras de parada. El stemming se basa en quitar y reemplazar sufijos de la raíz de la palabra; en cambio la técnica de lemmatización es un poco más compleja e implica hacer un análisis del vocabulario y su morfología para retornar la forma básica de la palabra [9].

Algunas de las aplicaciones del NLP son:

- Traducción automática: Debido a la ambigüedad y variabilidad del lenguaje, este sistema no involucra una sustitución palabra por palabra, sino que se apoya de la traducción automática estadística (Statistical Machine Translation, en inglés).[11]
- Análisis de sentimientos: U Opinion mining en inglés, consiste en la extracción de la opinión del autor a través de la identificación de forma subjetiva de la información analizada , lo cual permite medir varias cosas como el nivel de satisfacción de los clientes o usuarios a ciertos productos y servicios.[11]
- Chatbots: Son eficaces a través de cumplir de manera sencilla tareas y labores estándar como brindar información a los clientes acerca de productos y servicios, así como atender dudas y preguntas acerca de los mismos, y su utilización se ha estandarizado de forma masiva en aplicaciones y plataformas de mensajería en internet.[11]
- Clasificación de texto: Clasifica de forma efectiva para la organización de los mensajes o texto en un conjunto de categorías predefinidas.[11]
- Reconocimiento de caracteres: A partir del reconocimiento de caracteres realiza extracción de la información principal de una serie de documentos, texto o artículos.[11]
- Corrección automática: Los editores de texto a través de un corrector verifican la ortografía del documento señalado. [11]
- Resumen automático: Los métodos de NLP son utilizados para la digestión y análisis concreto de la información a través de extraer la informacion esencial y presentarla de forma fluida, precisa y breve. [11]

4.3. Teorema de Bayes

El teorema de Bayes es de gran importancia para la estadística inferencial y muchos modelos avanzados de aprendizaje automático. El razonamiento bayesiano se presenta como un

enfoque lógico para actualizar las probabilidades de hipótesis en función de nueva evidencia, un concepto fundamental en la ciencia [4].

La probabilidad condicional de un evento es una probabilidad que se calcula con el conocimiento adicional de que otro evento ya ha tenido lugar. En este contexto, utilizamos la notación $P(B|A)$ para expresar la probabilidad condicional de que el evento B ocurra, considerando que el evento A ya ha sucedido. Para calcular esta probabilidad condicional, se utiliza una fórmula específica que relaciona las probabilidades de los eventos A y B

$$P(B|A) = \frac{P(A \cap B)}{P(A)} \tag{1}$$

Análogamente tenemos

$$P(A|B) = \frac{P(B \cap A)}{P(B)} \tag{2}$$

De las ecuaciones 1 y 2 obtenemos que:

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A) \tag{3}$$

Y por consiguiente se concluye

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4}$$

5. Razonamiento

Para el desarrollo del sistema propuesto, primero fue necesario realizar un proceso de limpieza y preparación de datos respectivamente, puesto que para la implementación del modelo Naive Bayes con NLP, es fundamental contar con los datos en la forma más "digerible"posible para su uso en los cálculos probabilísticos respectivos y posterior procesamiento para su clasificación. Se inició con el proceso de la depuración de la base de datos utilizada, en el cual se eliminaron las últimas columnas del dataset, puesto que contenían información nula, para así obtener solo las columnas: "email", que contiene la construcción de palabras correspondiente a los emails recopilados y la columna "spam", la cual almacena las etiquetas de clase que indican si el email es spam, señalado con el número 1, o ham, señalado con el 0.

Posterior a ello, se aplicó procesamiento de texto directamente en la columna "emails" del *dataset*, el cual consistió en cuatro fases diferentes:

1. Conversión a minúsculas del contenido de palabras.
2. Remoción de caractéres especiales y signos de puntuación.
3. Remoción de *stopwords* o palabras "sin aporte" a la información.
4. Aplicación de *stemming* para la extracción de las raíces de las palabras y su posterior transformación a su forma más básica mediante la librería de Python *nltk*.

Al terminar este proceso, se dispuso del *dataset* listo para comenzar la implementación del modelo de Machine Learning, por lo que se dividió en dos conjuntos: el *set* de entrenamiento, con una proporción del 90 % de los datos y el *set* de prueba con la proporción restante del 10 % de los datos.

Después, se desarrollaron los cálculos necesarios para crear el clasificador de Naive Bayes, los cuales se explican de manera teórica a continuación.

En la sección anterior se menciona que el teorema de Bayes se estructura de la siguiente manera:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{5}$$

Esta expresión se puede interpretar como la probabilidad a priori $P(A)$ multiplicada por un factor que la actualiza, $P(B|A)/P(B)$, resultando en la probabilidad a posteriori. En el caso particular de este modelo, la probabilidad de que sea spam es la a priori, mientras que la evidencia observada que actualiza esta probabilidad es el conjunto de palabras que constituyen el correo, por lo cual, se calculó la probabilidad de que un correo sea spam dado el conjunto de palabras que lo componen.

Ahora bien, con esta herramienta estadística se plantea obtener dos probabilidades: que sea spam o ham (legítimo). Estas dos cifras se comparan y la que sea más alta determina en qué categoría se clasifica el correo específico. Teniendo en mente esta estrategia y considerando la forma del teorema de Bayes, se puede comenzar por manipularlo para que se ajuste a la situación y necesidades. Para empezar, se reescribió con los nombres de las variables de la problemática, donde w es el conjunto de palabras que constituyen el correo, con lo cual el teorema se ve de la siguiente manera:

$$P(spam|w) = \frac{P(w|spam)P(spam)}{P(w)} \tag{6}$$

En el contexto del problema, existen conjuntos de palabras los cuales tienen una probabilidad de 0 % de ser spam o ham dado el set de datos utilizado. Estos conjuntos de palabras no se consideran en el calculo de $P(w|spam)$ y $P(\neg w|spam)$ ya que al tener una probabiliad de 0, la probabildiad calculada también sería de 0. Esto implica que $P(w)$ se calcula con el subconjunto de palabras consideradas en el calculo de $P(w|spam)$ y $P(\neg w|spam)$.

$$P(spam|w) = \frac{P(w|spam)P(spam)}{P(w_S)} \tag{7}$$

$$P(spam|w) = \frac{P(w|spam)P(spam)}{P(w_H)} \tag{8}$$

El a priori, que representa la probabilidad de que un correo sea spam, es igual a la cantidad de palabras que constituyen el total de los correos de spam (P_s) sobre la cantidad de palabras en la totalidad de los correos, P_T, de la siguiente forma:

$$P(spam) = \frac{P_s}{P_{TS}} \tag{9}$$

Análogamente, para la probabilidad de ham:

$$P(ham) = \frac{P_h}{P_{TH}} \tag{10}$$

Dado que sólo existen dos categorías:

$$P(\neg spam) = P(ham) \tag{11}$$

Continuando, se puede calcular $P(w|spam)$ fácilmente. Para esto, primero se cuenta la

cantidad de veces que aparece cada palabra en los correos spam, con lo que se obtiene la probabilidad de que una palabra específica forme parte de un correo spam, diviendo sobre el total de palabras que hay en los correos de spam. Esto nos da la probabilidad de que cada palabra esté en un correo spam o bien, $P(w_i|spam)$, donde w_i es cada elemento del arreglo de palabras de un correo. Habiendo realizado este proceso para cada palabra, se obtiene $P(w|spam)$ multiplicando la probabilidades $P(w_i|spam)$ de las palabras que contiene el correo en análisis.

Este es el proceso que se lleva a cabo para calcular la probabilidad de que cada correo sea spam o ham, lo cual deriva de una variedad de simplificaciones, modificaciones y ajustes al teorema de Bayes original, lo cual es posible gracias al uso de teoría de conjuntos, probabilidad y supuestos, como la independencia entre las palabras (mismo que lo hace ïngenuo").

Tomando en cuenta lo explicado anteriormente, se crearon los distintos conjuntos y variables necesarias, primordialmente utilizando diccionarios de Python para el almacenamiento de las palabras, ya sea de spam o no spam, con sus respectivas probabilidades, para así obtener todos los elementos indispensables para los cálculos de probabilidad que, finalmente fueron comparados y generaron las predicciones con el conjunto de prueba.

6. Conclusiones

La implementación del método de 'Naive Bayes' revela una esperada y efectiva intersección entre la estadística Bayesiana y el aprendizaje automático, herramientas extremadamente reconocidas por su utilidad que, al ser comprendidas con profundidad, permitieron la creación de un instrumento que se nutre de las virtudes de ambas, creando algo innovador y efectivo.

Gracias a los procesos llevados a cabo para la implementación de este modelo, se revisaron bases teóricas de campos de gran relevancia en la ciencia de datos, como la estadística -como lo es el teorema de Bayes y sus derivados utilizados en este trabajo-, el procesamiento de datos -en referencia al tratamiento que se le dio a la base de datos previo a la implementación del modelo-, y más generalmente, la programación y el aprendizaje automático, lo que comprendió la elaboración puntual de este trabajo.

Analizando un poco el desempeño del modelo y de acuerdo a las métricas de evaluación efectuadas, se obtuvo un *accuracy* de 0.97, destacando que éste es bastante eficaz de forma general. No obstante, teniendo un enfoque en los correos clasificados como "falsos positivos", es decir, aquellos que fueron clasificados como spam y no lo son, los cuales generalmente son importantes para el usuario, se tomaría como métrica de evaluación la precisión, que tiene un valor de 0.87, el cual es satisfactorio aunque un poco bajo. También cabe mencionar que el *recall* fue de un valor de 0.9 y el *F1 score* de 0.88, métricas que también indican un desempeño satisfactorio del modelo. Con el afán de mejorar el *perfomance* del modelo, se podrían tomar medidas tales como la adhesión al *dataset* de más ejemplos de correos spam, con el objetivo de balancear la proporción de emails spam y no spam proporcionando condiciones más adecuadas para el entrenamiento y la prueba del modelo.

Finalmente, se reconocen las limitaciones de este modelo, muchas de las cuales recaen justamente en la naturaleza de "ingenuidad" inherente al mismo. La independencia entre palabras, el orden de éstas, expresiones comunes, ortografía, gramática y el sinfín de elementos que comprenden el idioma escrito son factores que se omiten en el proceso de clasificación debido a la complejidad que implicaría su consideración, lo que se suele evitar dados los de por sí ya competentes resultados del modelo.

Referencias

[1] Robinson, G. (2003). A statistical approach to the spam problem. Linux journal, 2003(107), 3.

[2] Spam e-mail traffic share monthly 2022. (s/f). Statista. Recuperado el 19 de agosto de 2023, de https://www.statista.com/statistics/420391/spam-email-traffic-share/

[3] Abdelhamid, N., Ayesh, A., & Thabtah, F. (2014). Phishing detection based associative classification data mining. Expert Systems with Applications, 41(13), 5948-5959.

[4] UCI Machine Learning. (2016). SMS Spam Collection Dataset [Data set].

[5] Berrar, D. (2018). Bayes' theorem and naive Bayes classifier. Encyclopedia of bioinformatics and computational biology: ABC of bioinformatics, 403, 412.

[6] González, F. A. (2015). Machine learning models in rheumatology. Revista Colombiana de Reumatología, 22(2), 77-78.

[7] Vargas, M., Biggs, D., Larraín, T., Alvear, A., Pedemonte, J. C., & de Anestesiología, R. (2022). Inteligencia artificial en medicina: Métodos de modelamiento (Parte I). Rev. Chil. Anest, 51(5), 527-534.

[8] ¿Qué es el procesamiento de lenguaje natural? - Explicación del procesamiento de Lenguaje Natural - AWS. (s. f.). Amazon Web Services, Inc. https://aws.amazon.com/es/what-is/nlp/

[9] Murzone, F. (2022, 30 marzo). Procesamiento de Lenguaje natural: Stemming y lemmas — EscuelaDeInteligenciaArtificial. Medium. https://medium.com/escueladeinteligenciaartificial/procesamiento-de-lenguaje-natural-stemming-y-lemmasf5efd90dca8#::text=El

[10] (S/f). Researchgate.net. Recuperado el 21 de agosto de 2023, de https://www.researchgate.net/figure/Figura-1-Diagrama-de-flujo-de-la-metodologia-Fuente-elaboracion-propia_fig1_349219264.

[11] Team, D. (2022). NLP Natural Language Processing : Introducción. https://datascientest.com/es/nlp-introduccion