Clasificación de Spam mediante Naive Bayes

Machine Learning

Abad L. Freddy, Cabrera C. Edwin, Peñaloza Fernando

Facultad de Ingeniería, Escuela de Sistemas

Universidad de Cuenca

{freddy.abad, edwin.cabrera, fernando.penaloza} @ucuenca.ec

***Abstract*: This article analyzes one of the most representative problems in the world of the network: unwanted content or "spam". This problem affects the vast majority of users of the network, due to being unsolicited, unwanted messages or with unknown sender. Normally it is advertising in nature, sending massive amounts, which in some way harms the recipient. In today's world, in which the world of the network is used in most fields of daily life, spam attacks mass-use websites, which is why the need arises to train algorithms that classify content, among spam and not spam. To do this, a model was trained with the naive Bayes classifier and its performance was measured in comparison with other classifier models. The results supported the robustness of the Naive Bayes models, due to their decoupling of conditional distributions class, this means that each distribution can be estimated independently as a dimensional distribution.**

***Keywords*: Naive Bayes classifier, Spam, Machine Learning, performance.**

***Resumen*: En este artículo se analiza uno de los problemas más representativos en el mundo de la red: el contenido no deseado o “spam”. Este problema afecta a la gran mayoría de usuarios de la red, debido a ser mensajes no solicitados, no deseados o con remitente no conocido. Normalmente es de naturaleza publicitaria, enviadas cantidades masivas, lo cual perjudica de alguna manera al receptor. En el mundo actual, en el cual el mundo de la red es utilizado en la mayoría de campos del vivir cotidiano, el spam ataca a sitios web de uso masivo, es por esto que surge la necesidad de entrenar algoritmos que clasifiquen el contenido, entre spam y no spam. Para ello se entrenó un modelo con el clasificador naive Bayes y se midió su rendimiento a comparacion de otros modelos clasificadores. Los resultados respaldaron la robustez de los modelos Naive Bayes, debido a su desacoplamiento de clase de distribuciones condicionales, esto significa que cada distribución se puede estimar de forma independiente como una distribución dimensional.**

***Palabras Clave*: clasificador naive Bayes, spam, Machine Learning, rendimiento.**

1. INTRODUCCIÓN

Ante una sociedad cada vez más conectado, debido a la globalización, en la cual los medios digitales: sitios web, redes sociales, etc. tienen un protagonismo cada vez mayor en el diario vivir, surge la necesidad imperante de identificar la información basura de medios digitales.

Este informe explica la implementación de un algoritmo clasificatorio de texto (comentario) spam, esto se realiza con uno de los métodos más acertados existentes en la actualidad: Naive Bayes.

Este algoritmo tiene la ventaja de tener un desacoplamiento de clase de distribuciones condicionales, lo cual significa que cada distribución se puede estimar de forma independiente como una distribución dimensional. Esto ayuda a aliviar los problemas derivados de la Maldición de la dimensión, tales como la necesidad de conjuntos de datos que se escalan exponencialmente con el número de características. Esto hace que el modelo de Naive Bayes se robusto en la práctica, para determinar texto spam (no deseado) o ham (deseado).

La implementación de este modelo se realiza usando el lenguaje de programación Python, usando distintas pero pocas librerías open source.

1. *Justificación*

El presente artículo busca implementar un clasificador de spam del sitio web YouTube, mediante el clasificador probabilístico naive Bayes, el cual se fundamenta en el teorema de Bayes y demás hipótesis simplificadoras. El motivo final de este artículo es el medir el rendimiento del clasificador de Naive Bayes.

1. *Objetivo General*

* Clasificar el spam del Dataset “YouTube Spam Collection Data Set” mediante Naive Bayes.
* Medir el rendimiento del algoritmo en lenguaje Python de Naive Bayes.

1. ESTADO DEL ARTE

La clasificación es el punto principal en esta investigación, ya que la identificación de contenido spam se modela como un problema de clasificación, donde las dos clases para cada comentario analizar es si es spam o no es spam.

Naive Bayes asume que la presencia o ausencia de una característica particular no está relacionada con la presencia o ausencia de cualquier otra característica, dada la clase variable. Naive Bayes es ventajoso debido a que solo requiere una pequeña cantidad de datos de entrenamiento para estimar los parámetros (las medias y las varianzas de las variables) necesarias para la clasificación. Como las variables independientes se asumen, solo es necesario determinar las varianzas de las variables de cada clase y no toda la matriz de covarianza.

La clasificación es la tarea de aproximar una función objetivo desconocida Φ : I x C → {T, F} por medio de una función Θ : I x C → {T, F} llamada clasificador, donde C = {c1, c2, ..., c|c|} es un conjunto de clases definido, e I es un conjunto de instancias del problema. Cada instancia ij ∈ I es representada como una lista A = {a1, a2, ..., a|A|} de valores característicos, conocidos como atributos. Es decir, ij = {a1j, a2j, ..., a|A|j}. Si Φ: I x C → T entonces i j es llamado un ejemplo positivo de c i, mientras que si Θ: I x C → F es llamado un ejemplo negativo de c i [3]. En general no se conoce la descripción exacta de las muestras, por lo que el sistema es entrenado a priori para ajustarse a las características propias del problema. A este proceso de adquirir e integrar conocimiento a un sistema de clasificación a partir de ejemplos, se le conoce como aprendizaje o entrenamiento [2].

Naive Bayes es un algoritmo de clasificación probabilística con muy poco grado de complejidad, lo cual es muy adecuado para datos categóricos (las probabilidades se pueden calcular como relaciones simples) y utiliza el teorema de Bayes junto con una suposición de independencia fuerte. La idea básica detrás de Naive Bayes es que asigna una probabilidad a cada categoría (variable de resultado finita) en función de las características de los datos y elige el resultado más probable como su predicción. Naive es el nombre que refiere al algoritmo que asume que las características en los datos son condicionales independientes en la categoría de resultados.

Un ejemplo de este principio es: se asume que se realiza una clasificación de texto no deseado, y luego un texto spam "¡Gratis, regístrate ahora!", Naive Bayes asumiría "Gratis", "regístrate", “ahora" todo ocurre independientemente el uno del otro *Pr(Gratis, regístrate,ahora|spam) = Pr(Gratis|spam) x Pr(regístrate|spam) x Pr(ahora|spam)*. Esta suposición de independencia condicional es considerada una suposición fuerte que no se cumple en la práctica, por tanto, las probabilidades resultantes de Naive Bayes no se debe tomar demasiado en serio. Sin embargo, las clasificaciones resultantes de Naive Bayes aún pueden ser precisas.

Un ejemplo de aplicación de Naive Bayes, es dado un conjunto de datos de entrenamiento del clima y la variable objetivo correspondiente 'Play' (sugiriendo posibilidades de jugar). Ahora, debemos clasificar si los jugadores jugarán o no según las condiciones climáticas.

Paso 1: Convierta el conjunto de datos en una tabla de frecuencias

Paso 2: Cree la tabla de Probabilidad al encontrar las probabilidades como Probabilidad de Overcast = 0.29 y la probabilidad de jugar es 0.64. Figura 1.

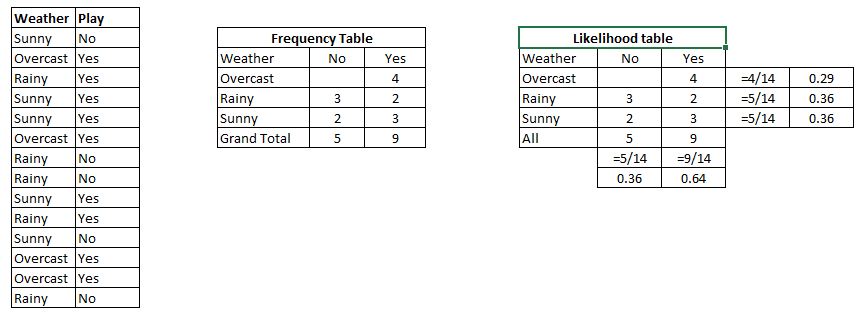


Figura 1.Tabla de probabilidad y frecuencias.

Paso 3: Ahora usa la ecuación naive Bayesiana para calcular la probabilidad posterior para cada clase. La clase con la probabilidad posterior más alta es el resultado de la predicción.

**Problema:** los jugadores jugarán si el clima es soleado. ¿Esta afirmación es correcta?

Podemos resolverlo usando el método de probabilidad posterior discutido arriba.

P (Sí | Soleado) = P (Soleado | Sí) \* P (Sí) / P (Soleado)

Aquí tenemos P (Soleado | Sí) = 3/9 = 0.33

P (Soleado) = 5/14 = 0.36, P (Sí) = 9/14 = 0.64

P (Sí | Soleado) = 0.33 \* 0.64 / 0.36 = 0.60 (Esta tiene mayor probabilidad).

Naive Bayes usa un método similar para predecir la probabilidad de diferentes clases basadas en varios atributos. Este algoritmo se usa principalmente en la clasificación de texto y con problemas para tener múltiples clases.

Las aplicaciones de Naive Bayes, en la vida cotidiana, incluyen la clasificación del correo no deseado, el análisis de sentimientos y la categorización de documentos. Modelar mediante Naive Bayes es ventajoso sobre otros algoritmos de clasificación debido a su simplicidad, velocidad y precisión en pequeños conjuntos de datos. Es considerado como un algoritmo de aprendizaje supervisado porque necesita ser entrenado en un conjunto de datos etiquetados.

1. METODOLOGÍA Y DESARROLLO

En la investigación se utiliza los datos de la UCI Machine Learning Repository [1], este contiene 1957 comentarios de usuarios en videos famosos del sitio web YouTube. Además de especificar el id, el autor, la fecha y el contenido del comentario, estos han sido etiquetados previamente con un marcador binario donde 1 representa a spam y 0 representa ham (no spam), con esta data se entrenará el modelo Naive Bayes para una posterior clasificación.

1. *Herramientas*

* numpy
* Pandas
* f1\_score

El dataset analizado, se realizó con los comentarios y clasificación [1] (spam/ham), llamados en el archivo .py como content y class. Desarrollando el modelo Naive Bayes, implementado en Python, realiza la lectura de este archivo csv, mediante pandas. Este dataset, toma los separadores como una tabulación (‘\t’), esto se debe, a que al ser comentarios existen comentarios intermedios, estos comentarios tienen que tener una concordancia semántica, es por esto que se separa por tabulaciones.

Se nota además por heurística que los comentarios spam, no tienen concordancia con el video al que pertenece (Video YouTube). Además, cabe recalcar que se dividió el dataset en partes 75% y 25%, como training y test.

Para generar una concordancia entre prueba y prueba (número de ejecuciones del código del modelo), creando una sémica mediante los métodos *random y seed* de la librería *numpy*.

Es necesario que las particiones del dataset (Train y Test debe estar conformado por comentarios spam y ham para un mejor uso del dataset).

Para la gestión de spam hay que realizar las siguientes tareas:

1. Para determinar si un comentario es Spam o no se basa en el número de ocurrencias de una palabra en un comentario clasificado como Spam o Ham. Es por ello que se realiza un tratamiento a cada registro: este consiste en eliminar los caracteres especiales, y convertir el texto en minúsculas.
2. Generar una lista que contenga a cada palabra del dataset una sola vez.
3. Obtener el número de veces que cada palabra aparece en un comentario clasificado como Spam o Ham. Este proceso se realiza como se muestra en la Figura 2.

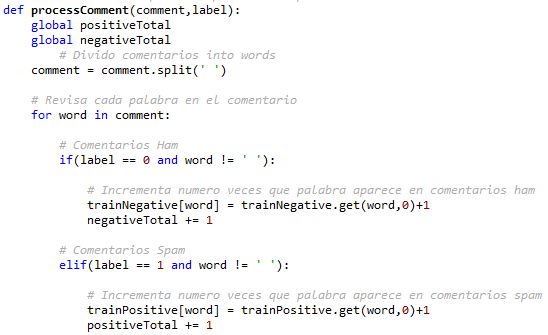


Figura 2: Conteo de Palabras Clasificadas como Spam o Ham

1. Realizar el entrenamiento: Consiste en obtener las probabilidades P (spam) & P (ham) en base al número de comentarios clasificados como spam. Además de ello, se revisa cada comentario del dataset de entrenamiento para actualizar el número de veces que una palabra aparece en un comentario ham o spam. Las sentencias utilizadas se notan en la Figura 3.

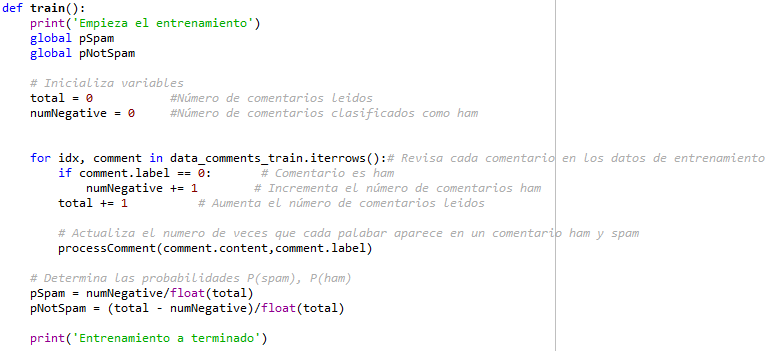


Figura 3: Entrenamiento de Modelo Naive Bayes

1. Obtener la probabilidad de una palabra: Consiste en determinar la razón entre el número de veces que una palabra aparece en un determinado tipo de comentario (spam o ham) y el número de veces palabras registradas en determinado tipo de comentario. En otras palabras, la función obtiene la probabilidad Prob(palabra|spam) o Prob(palabra|ham). La implementación del código se presenta en la Figura 4.

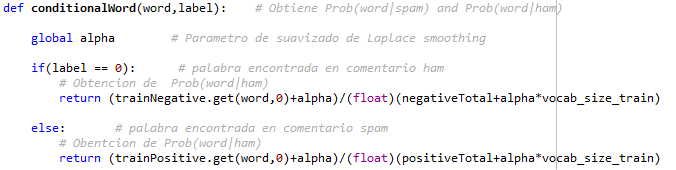


Figura 4: Método para la obtención de la probabilidad de una palabra

1. Obtención de la probabilidad de un comentario: Consiste en determinar la probabilidad que tiene un comentario de ser clasificado como spam o ham, para ello; realiza un bucle que recorre cada palabra ejecutando el código presentado en la Figura 3, el resultado es la multiplicación de las probabilidades de cada palabra que conforma el comentario. Esta explicación reflejada en código se presenta en la Figura 5.

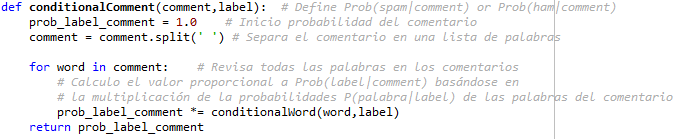


Figura 5: Método para clasificación de un comentario como spam o ham

1. Clasificación: Dado un determinado comentario, se determina si es spam o ham comparando P(spam)\*P(comentario|spam) y P(ham)\*P(comentario|ham). P(comentario|spam) y P(comentario|ham) se obtienen mediante el paso 7. El proceso de clasificación se presenta en la Figura 6.

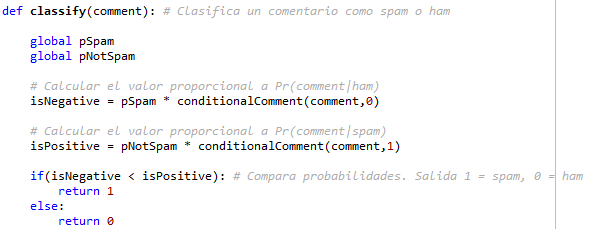


Figura 6: Proceso de clasificación de un comentario

1. PRUEBAS Y RESULTADOS

Para determinar la validez del proceso de entrenamiento, se compara el resultado predicho contra el resultado deseado para cada comentario del conjunto de datos de prueba, para ello; se aplicó la métrica proporcionada por la librería *f1\_score.* Los resultados de la clasificación se presentan en la Figura 7.

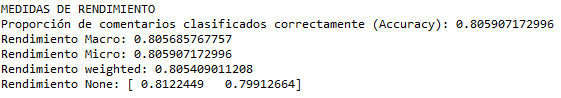


Figura 7: Resultados obtenidos de la clasificación de spam del conjunto de datos de prueba.

Al repetir el proceso de evaluación con determinado tipo de mensajes se muestra la influencia probabilística de cada palabra en el proceso de clasificación, esto se comprueba en la Figura 8 donde se muestra un conjunto de frases empleadas para determinar si es spam, las frases “You won a car in Cuenca Ecuador” y “You drive a car in Cuenca Ecuador” varían en la palabra “won” y “drive” de las cuales “won” debe contar con una probabilidad P(“won”|spam) suficiente para determinar al comentario como spam.

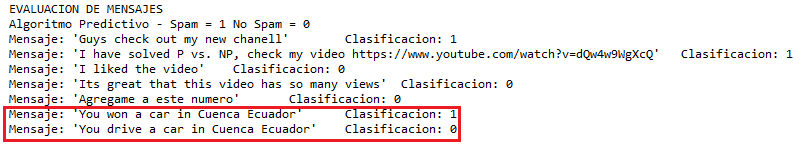


Figura 8: Clasificación de frases como spam o ham.

1. CONCLUSIONES

Naive Bayes asume que la presencia o ausencia de una característica particular no está relacionada con la presencia o ausencia de cualquier otra característica, dada la clase variable. Naive Bayes es ventajoso debido a que solo requiere una pequeña cantidad de datos de entrenamiento para estimar los parámetros (medias y varianzas de las variables) necesarias para la clasificación.

Las variables independientes de Naive Bayes se asumen, así solo es necesario determinar las varianzas de las variables de cada clase y no toda la matriz de covarianza.

Naive Bayes determina la probabilidad de cada palabra en un comentario, obteniendo la razón entre el número de veces que una palabra aparece en un determinado tipo de comentario (spam o ham) y el número de veces palabras registradas en determinado tipo de comentario. Así es como aplica el teorema de Bayes en su ejecución.

Naive Bayes determina la probabilidad de cada comentario de que sea clasificado como spam o ham, recorriendo nuevamente cada palabra de un comentario. Así el resultado del cálculo de probabilidad es la multiplicación de las probabilidades de cada palabra que conforma el comentario.

Para conocer la clasificación del comentario se realiza una comparativa de las operaciones P(spam)\*P(comentario|spam) y P(ham)\*P(comentario|ham), determinando así el grupo clasificatorio de los comentarios.

1. RECOMENDACIONES

Dado las ventajas de Algoritmo Naive Bayes, por la certeza en la clasificación del dataset. Además de la rapidez a comparación de otros algoritmos comparativos, específicamente en este caso, de clasificación de spam; se recomienda pre procesar los dataset. La reprocesamiento del dataset es realizar un tratamiento del texto, este tratamiento incluiría previamente ya pasarle a minúsculas, por ejemplo, además de identificar los idiomas que se analizaran. Esto es por el encoding que se utilizara, ya que, si se usa el abecedario en formato inglés, se debe usar un encoding utf-8, por defecto usado en los algoritmos; pero si se usa el abecedario en formato español se debe usar un encoding Latin8; esto se realizara con los lenguajes que se vayan a analizar. La razón de esta recomendación se debe a que un dataset pre procesado, aumentaría la rapidez de modelamiento de Naive Bayes, además de identificar mucho más rápido la lista de repeticiones de palabras.

1. BIBLIOGRAFÍA
2. Lichman, M. (2013). UCI Machine Learning Repository [archive.ics.uci.edu/ml/datasets/YouTube+Spam+Collection]. Irvine, CA: University of California, School of Information and Computer Science.
3. Garcia, V. (2010). “Distribuciones de clases no balanceadas: métricas, análisis de complejidad y algoritmos de aprendizaje”. Tesis doctoral. Departament de llenguatges i Sistemes Informàtics, Universitat Jaume I.
4. Sánchez, C. R. (2008). “Clasificación de entidades nombradas utilizando información global”. Tesis de Maestría, INAOE.
5. Gonzales Y., Pedroza B., Lopez F. “Implementación del clasificador naive Bayes para la acentuación automática de palabras ambiguas del español”, Instituto Tecnológico de Apizaco. Disponible online: <http://www.progmat.uaem.mx:8080/Vol6num1/vol6num1art3.pdf>
6. YouTube. Miranda Aguilar Aldo (junio 23 del 2016). Obtenido en: <https://www.youtube.com/watch?v=oQ1OyqvL7dQ>
7. Colaboradores Wikipedia. “Clasificador bayesiano ingenuo”. (2017). Obtenido en: <https://es.wikipedia.org/w/index.php?title=Clasificador_bayesiano_ingenuo&oldid=102158243>