Pontificia Universidad Católica del Perú

Tarea Académica

Aplicaciones de Ciencias de la Computación

Luis Ramirez Osorio
20161567

John Benites Huamán
20161589

Reiver López Casavilca
20156134

Herbert Esplana Hurtado
20152801

29 de agosto de 2019

Índice

1. Introducción

El problema sobre el que se basará el presente proyecto es la identificación de mensajes "Spam", que son los mensajes no deseados o que no son de interés del destinatario. Este es un problema debido a que muchos usuarios reportan quejas en distintas plataformas (foros, sitios web). En particular, en este trabajo se tomó en cuenta las quejas realizadas en el sitio web Grumbletext. La queja mas frecuente que se reportan es sobre mensajes publicitarios(Spam) recibidos en los teléfonos celulares de los usuarios. Sin embargo, muchos de estos no reportan los mensajes Spam recibidos, lo cual dificulta el identificar si un mensaje es "Spam" o no. Así , los mensajes con los que se trabajará en el siguiente proyecto se obtuvieron de una base datos de mensajes recopilados de la investigación en el Departamento de Ciencias de la Computación de la Universidad Nacional de Singapur.

El objetivo de este proyecto es clasificar un grupo de mensajes en Spam y Ham. Esto, se logrará a partir de distinguir algunas características del los mensajes recibidos. Por ejemplo, la longitud del mensaje y algunas palabras del mensaje en sí que aparecen con gran frecuencia en los mensajes Spam de nuestra base de datos, etc. Asimismo, para que el modelo de clasificación que se plantee no tenga errores de implementación, previamente se deberá analizar que los datos no presenten irregularidades ,como por ejemplo, la datos faltantes o que haya grandes diferencias en la proporción de datos Spam en relación al Ham. Es decir, que haya una gran cantidad de mensajes Spam y una cantidad mínima de mensajes Ham o viceversa, ya que esto haría que el modelo de clasificación no se entrenase de manera adecuada. Por ejemplo, dado una gran cantidad de mensajes Spam y un numero reducido de mensajes Ham, el modelo de clasificación entrenado tenderá a clasificar cualquier mensaje como Spam ya que posee mayor conocimiento sobre mensajes Spam. Esto nos llevaría a obtener un modelo ineficiente. En las siguientes secciones del presente informe se explicará cuál es el estado de este problema, como se diseñó el experimento, los resultados que se obtuvieron, un análisis de estos resultados y finalmente las conclusiones de este informe.

2. Estado del arte

El Spam puede definirse como un mensaje no deseado enviado por un remitente desconocido con fines publicitarios y generalmente enviado en grandes cantidades. El Spam no es un problema que sea nuevo, de hecho, ha existido desde la existencia del correo electrónico y con el paso del tiempo este fue llegando a los mensajes de texto en los móviles, redes sociales, foros, etc. Por ello se han buscado diferentes métodos para poder filtrar estos mensajes y que solo nos lleguen los mensajes que sean relevantes.

Por un lado, actualmente, el estudio de este tema ha avanzado de tal forma que incluso se han estudiado métodos para filtrar el Spam en las redes sociales. De acuerdo al artículo publicado por Gauri Jain, Manisha Sharma y Basant Agarwal [1], los mensajes de Spam en las redes sociales se ha ido incrementando a lo largo de estos años, tanto es así que llega un punto que los mensajes que son relevantes se pierden dentro de estos. Es por ello que estos autores proponen un método de una nueva arquitectura de detección de Spam basado en Redes Neuronales Convolucionales(Convolutional Neural Network CNN) y Redes Neuronales de Largo-Corto Plazo(Long Short Term Neural Network LSTM).
En su artículo se concluyó que identificar si un mensaje es Spam o no, es un problema muy retador. Aunque, se logró clasificar los mensajes en los e-mails con un elevado nivel de acierto, los mensajes en Twitter son aún muy complicados debido a que la mayoría tienen una longitud similar y eso eleva su complejidad.

Por otro lado, el trabajo presentado por Sarita Yardi, Daniel Romero, Grant Schoenebeck y Danah Boyd [4] sobre el spam en Twitter y métodos para filtrarlos ahondan más sobre cómo es que el spam surge en esta red social. Para ello decidieron seguir un "hashtag" durante todo su ciclo de vida, es decir, desde su creación hasta el momento de su completo desuso para poder analizar en qué momentos el "hashtag" era empleado para enviar spam y cuando no. Para ello, emplearon diversos algoritmos que empleaban datos como el nombre del usuario, verificaban si el mensaje contenía direcciones webs y palabras claves que suelen emplearse en spam.
Finalmente, concluyeron que las primeras horas desde la creación del "hashtag" los tuits en su gran mayoría eran mensajes normales, mientras que durante las siguientes horas en el que el "hashtag" se

hacía más conocido el número de mensajes con spam se iba incrementando y con ello su dificultad para clasificarlos pues cada vez los mensajes de spam más elaborados. Sin embargo, durante los siguientes días, cuando el "hashtag" dejó de ser tan popular los mensajes de spam seguían incrementándose y los mensajes normales parecía disminuir. Llegó a tal punto que más del 95 % de mensajes eran spam y el "hashtag" ya había quedado en el olvido. Al igual que la investigación anterior,se concluyó que tratar de clasificar los mensajes es muy complicado. Pues cada vez es más difícil el poder diferenciar entre un mensaje normal y uno que realmente es spam, ya que como se mencionó anteriormente en el punto en el que el "hashtag" llegó a ser "trending topic" el spam era mucho más elaborado.

3. Diseño del experimento

3.1. Descripción del conjunto de datos

El conjunto de datos de mensajes están compuestos por dos partes. El primero de ellos representa el tipo de mensaje ,ya sea "Spam" o "Ham", y el otro atributo representa el mensaje en sí. La cantidad de mensajes con los que se trabajarán son 5572, donde los mensajes que predominan son los Ham. En el gráfico 1 se mostrará la distribución de los mensajes Ham y Spam.

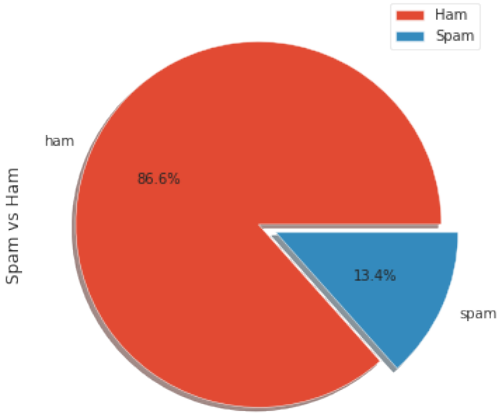


Figura 1: Histograma

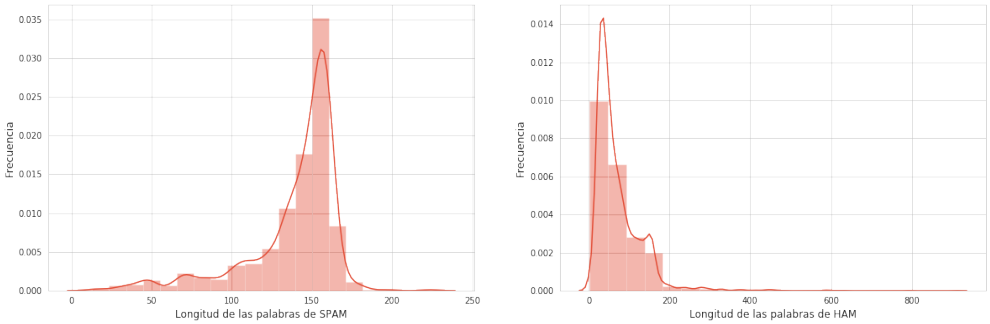
Se ha decidido separar los datos de entrada en dos partes: los datos prueba y datos de entrenamiento. Donde el 20 % representará a los datos de entrenamiento ya que este es un porcentaje aceptable para obtener una predicción con un error mínimo. Por otro lado, si es que se toma un mayor porcentaje de datos de entrenamiento, se puede sobre entrenar el modelo y así, no se lograría el objetivo del proyecto que es clasificar cualquier mensaje en spam o ham.

3.2. Metodología

En primer lugar, se analizaron los datos y ninguno de ellos presentaba datos nulos o faltantes. Por lo cual, no se aplicó ninguna metodología para el manejo de datos faltantes. En segundo lugar, se observo que cada dato poseía cinco columnas, de los cuales en las tres ultimas no poseía datos. Así, se procedió a eliminarlas. En las dos primeras columnas, la primera contenía el tipo de mensaje Spam o Ham, y el

segundo el contenido del mensaje.

Por otro lado, según los requerimientos de los algoritmos de clasificación, se identificó que el tipo de dato (spam o ham) necesitaba transformarse en un dato numérico binario para poder implementarlo en el algoritmo ya que este solo funciona con datos numéricos. Además, para un mejor análisis se necesitaba saber la longitud del texto, ya que los datos presentaban longitud variable. Así, se creo una nueva columna en nuestra colección de datos.



Según la figura mostrada la longitud de los mensajes Spam tiende a ser más cercano a 200. Esto tiene sentido pues, generalmente, los mensajes de Spam suelen ser más largos. Mientras los mensajes Ham tienen una mayor frecuencia hacia la longitud de 100. Esto se debe a que, usualmente, las personas suelen ser más breves y por ello emplean abreviaciones en sus mensajes.

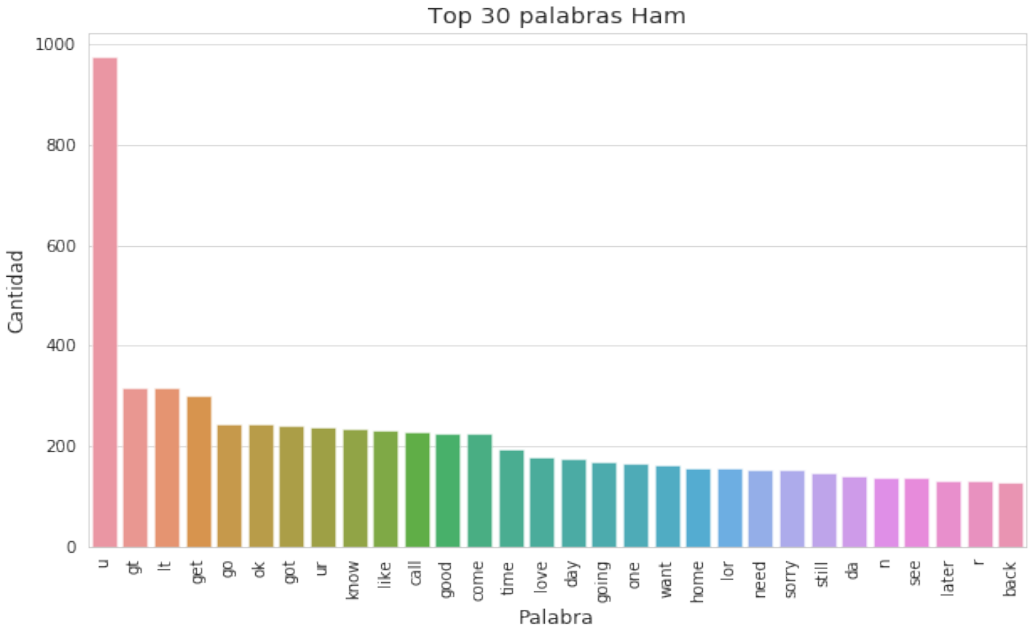
Asimismo, se analizaron las palabras de los dos tipos de mensajes. Para ello, con el código de un usuario de kaggle: Ishan Soni [2] se elaboró una nube de palabras para los dos tipos de mensajes.

(a) Nube de palabras de Ham

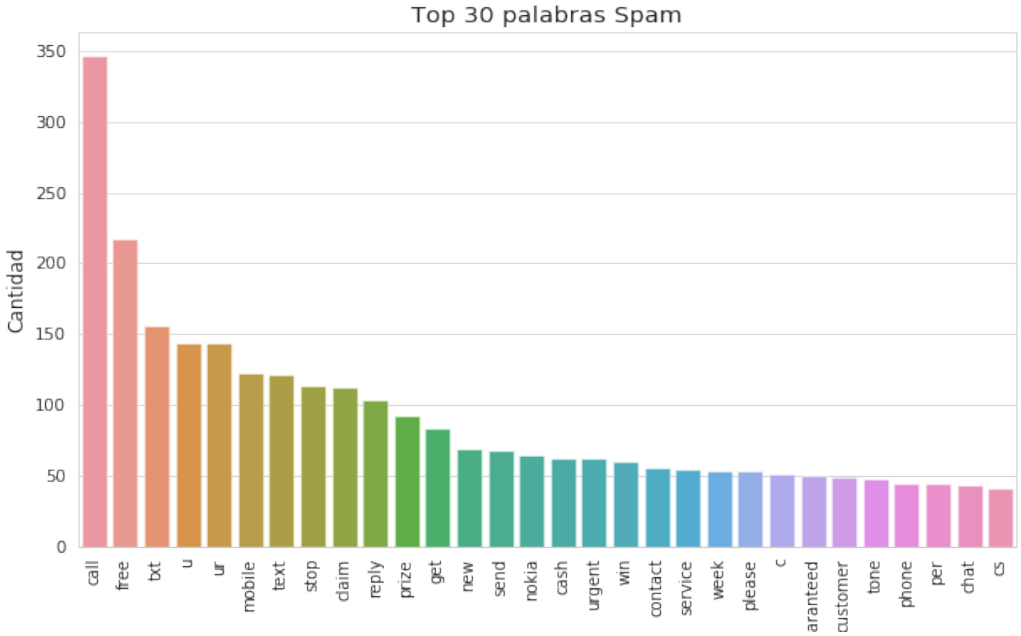
(b) Nube de palabras de Spam

Figura 2: Nube de palabras

Si se ve a la figura 2 se tiene dos imágenes que muestran las palabras que más se repiten con una fuente más grande y a partir de ahí el tamaño de la fuente va disminuyendo de acuerdo a su frecuencia de repetición entre los mensajes. La figura 2a muestra la nube de palabras de los mensajes Ham y como se puede apreciar las palabras con mayor frecuencia son “ok”, “got” y “gt”. Por su parte, la figura 2b muestra las palabras que más se repiten de los mensajes Spam y como se puede observar las palabras que más se repiten son “free”, “call” y “text”, como era de esperarse. Sin embargo, esta nube solo nos muestra a simple vista las palabras con mayor frecuencia y para poder verse con exactitud la cantidad de veces que se repite una palabra es mejor llevarlos a un gráfico de barras.



En la figura podemos observar el gráfico de barras de las 30 palabras Ham que más se repiten y como era evidente la palabra que más se repite, cerca de 1000, es "u", que es una forma abreviada de decir "you" en inglés. La palabra que le sigue es "gt" pero a diferencia del anterior este es menos frecuente llegando repetirse cerca de 300 veces y a partir de estas palabras las frecuencias van disminuyendo gradualmente.



Al igual que el anterior gráfico de barras, este muestra las 30 palabras que más se repiten de los mensajes que son considerados Spam. Tal y como hemos visto en la nube de palabras, la palabra que más se repite es "call" con cerca de 350 repeticiones. La siguiente es "free" la cual llega a los aproximadamente a los 230 repeticiones.

La medida de calidad de un mensaje es un valor que mide la frecuencia con la que las palabras que las componen aparecen en mensajes spam. Este valor ha sido uno de los factores que se ha utilizado para la clasificación de ambos tipos de mensaje.

Para este problema, se ha utilizado, principalmente, cuatro clasificadores: KNN, regresión logística, árbol de decisión y random forest. Elegimos estos clasificadores pues, además de ser sencillos de implementar, son los que mejores resultados han obtenido según los "kernels" publicados en kaggle y solo siendo superados por "multinomial naive bayes classifier (MNB)".

4. Experimentación y resultados

En este proyecto se han realizado varias pruebas con el fin de analizar cuales de los factores, que están implicados, afectan en el resultado de los tipos de mensaje. Como se indico en las secciones anteriores estos parámetros son: la longitud de los mensajes, las palabras que se utilizan frecuentemente en cada tipo de mensaje(Spam y Ham). Asimismo se han ido variando los parámetros de los clasificadores hasta encontrar el que obtiene el mejor resultado.

En primer lugar, se entrenaron los datos de acuerdo a los mensajes en sí. Para ello se tuvieron que eliminar los signos de puntuación de los mensajes para luego poder vectorizarlos ya que los algoritmos de clasificación solo funcionan con datos numéricos. Una vez que se hayan realizado todos los preparativos se pasa a la fase de entrenamiento de los modelos para luego verificar el score con los datos de prueba. Los resultados que se obtuvieron se encuentran en las siguiente tabla.

Clasificador	Score 1
KNN	0.921076
DTC	0.957848
LRC	0.959641
RFC	0.974888

Donde KNN es "k-nearest neighbours" con 49 vecinos, DTC es "Decision tree classifier" con un mínimo de ejemplares de 7, LRC es "Logistic Regression Clasifier" con solver liblinear y RFC "Random Forest classifier" con 31 estimadores. Variamos los parámetros pero los resultados máximos que se consiguieron fue con los que el usuario de Kaggle Evgeny Volkov [3] empleó para sus clasificadores.
Los valores mostrados se pueden apreciar mejor en el siguiente gráfico de barras.

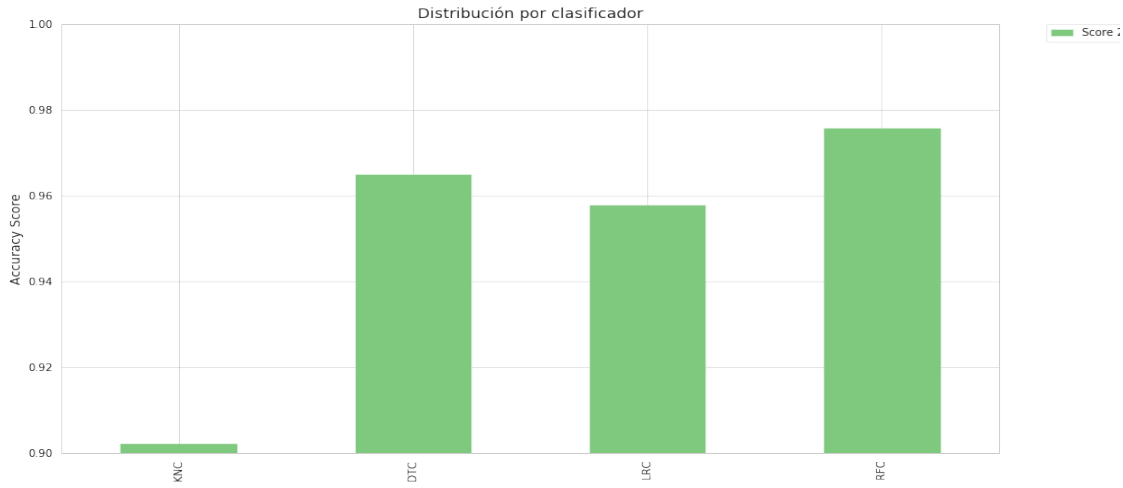
Por un lado, como se puede apreciar en el gráfico el que obtuvo mayor score es "Random Forest classifier" con 97.5 % aproximadamente de acierto. Luego le sigue "Logistic Regression Clasifier" con 95.6 %

de acierto. Por otro lado, el clasificador que obtuvo el score más bajo es KNN con 92.1 % de acierto.

En segundo lugar, se entrenaron los modelos de acuerdo a la longitud del mensaje. Para ello se agregó una columna más a la tabla de datos para luego pasar la longitud a una matriz para poder entrenar los modelos. Los resultados que se obtuvieron se encuentran en la siguiente tabla.

Clasificador	Score 2
KNN	0.902242
DTC	0.965022
LRC	0.957848
RFC	0.975785

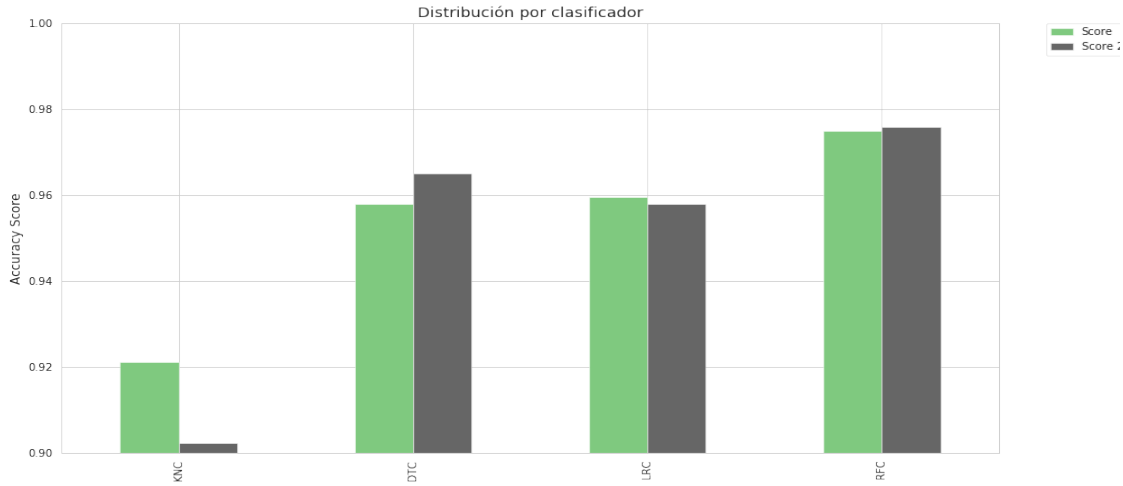
Los valores mostrados se pueden apreciar mejor en el siguiente gráfico de barras.



En el gráfico se puede apreciar que el que obtuvo mayor score es "Random Forest classifier" con 97.6 % de acierto, aproximadamente. Luego le sigue "Decision tree classifier" con 96.5 % de acierto, aproximadamente. Mientras que el que obtuvo menos score es "k-nearest neighbours" con 90.2 % de acierto, aproximadamente.

5. Discusión

Con los datos obtenidos de la experimentación podemos comparar ambos resultados y ver que entrenamiento es mejor para este tipo de datos y también encontrar el mejor clasificador. Para ello, se presenta el siguiente gráfico de barras que combina los dos anteriores que se presentaron en la experimentación.



En el gráfico las barras de color verde son el score obtenido mediante el primer entrenamiento y los de color negro mediante el segundo entrenamiento. A primera vista lo que llama la atención es el score que se obtuvo con el clasificador "k-nearest neighbours". Empleando el entrenamiento por longitud de mensaje se obtuvo un score mucho menor que con el entrenamiento por similitud de palabras. El bajo score puede deberse a que este clasificador es considerado como un clasificador perezoso y por ello tiene que realizar muchos cálculos para clasificar un mensaje. Además, puede deberse, también, a que la relación de mensajes ham y spam es mayor a 1 lo cual indica que existen pocos mensajes spam que sirvan como vecinos para la clasificación y por ello la mayoría de veces tenderá a clasificar a los mensajes como ham. El siguiente clasificador "Decision tree classifier" con el entrenamiento por longitud del mensaje obtiene un score mayor que con el entrenamiento por coincidencia de palabras. Este incremento puede deberse a que clasificar por coincidencia de palabras no sea tan adecuado para este tipo de clasificador pues se termina realizando un modelo muy determinístico que no llega a clasificar correctamente los mensajes. El siguiente clasificador es "Logistic Regression Clasifier" y como se puede apreciar con el primer método de entrenamiento consigue una ligera mejora pero que en la práctica no es muy relevante por lo que se podría decir que con ambos métodos de entrenamiento se obtienen resultados similares. Finalmente, el último clasificador 'Random Forest classifier" es el que obtiene los mejores resultados en ambos tipos de entrenamiento pues, a diferencia del árbol de decisión, 'Random Forest classifier" es menos determinístico, por lo cual puede generalizar tanto las longitudes como las coincidencias de palabras.

A partir de este análisis podemos observar que para poder mejorar la tasa de aciertos es necesario más información. Al igual que las investigaciones mencionadas al inicio de este proyecto contar con el emisor de los mensajes podría ayudar a los clasificador a incrementar su tasa de aciertos pues un emisor puede haber mandado el mismo mensaje a diferentes usuarios y con esta información se podrían inmediatamente clasificar inmediatamente. Asimismo, también es relevante tener los datos del receptor pues así se puede saber cuantos mensajes spam le llegan y de acuerdo a ello poder identificar el tipo de destinatario que recibe mayor spam. Con dicha información se podría mejorar aún más los clasificadores e incrementar su tasa de acierto.

6. Conclusiones y trabajos futuros

En conclusión, la tarea de clasificar mensajes en "spam" o "ham" es complicada. Dado a la gran cantidad de información que es enviada actualmente por la red. Sin embargo, hemos podido observar que herramientas de inteligencia artificial, como los algoritmos de clasificación trabajados, pueden ayudarnos a superar este problema. Obviamente, aún estamos lejos de una solución implementable a gran escala, pero pudimos darnos cuenta de que si se tuvieran más atributos de los mensajes (como los emisores,destinatario,etc) se podría crear un clasificador que podría llegar a funcionar en una aplicación en

el mundo real.

Como trabajo futuro se ve la posibilidad de implementar estos modelos de clasificación, en particular el que obtuvo una mayor tasa de aciertos, para filtrar mensajes. Así, los mensajes clasificados como Spam que reciben los destinatarios tendrán una menor relevancia en la vista principal de la bandeja de entrada. Con esto, se busca que la tasa de mensajes Spam que es leído por los usuarios se reduzca en gran medida.

Referencias

[1] Gauri Jain, Manisha Sharma y Basant Agarwal. "Spam detection in social media using convolutional and long short term memory neural network". En: *Annals of Mathematics and Artificial Intelligence* 85.1 (ene. de 2019), págs. 21-44. ISSN: 1573-7470. DOI: 10.1007/s10472-018-9612-z. URL: https://doi.org/10.1007/s10472-018-9612-z.

[2] Ishan Soni. *SMS Spam Collection Dataset*. URL: https://www.kaggle.com/ishansoni/sms-spam-collection-dataset.

[3] Evgeny Volkov. *SMS spam detection with various classifiers*. URL: https://www.kaggle.com/muzzzdy/sms-spam-detection-with-various-classifiers.

[4] Sarita Yardi y col. "Detecting spam in a Twitter network". En: *First Monday* 15.1 (2009). ISSN: 13960466. DOI: 10.5210/fm.v15i1.2793. URL: https://journals.uic.edu/ojs/index.php/fm/article/view/2793.