[[1]](#footnote-1)

Sistema Clasificador de Sentimientos (Agosto 2016)

Juan Benavides, Steven Chiriboga, Alejandro Pardina

*Abstract*— El siguiente trabajo muestra cómo implementar un clasificador de sentimientos usando la api de weka, en java y un procesador de texto plano programado en perl, en el cual se definen los atributos necesarios para el clasificador J48 y el Naive Bayes. Al final de nuestro trabajo se obtuvo un resultado de 95.0916 % con Naive Bayes y 95.8242 % usando el árbol de decisión J48 al recibir de entrenamiento un archivo de 1365 oraciones clasificadas.

Se puede ver el código del trabajo en el siguiente repositorio:

<https://github.com/juanben/BI>

*Índice de términos*—

# INTRODUCCIÓN

El análisis de sentimientos es un proceso el cual se necesita un proceso de análisis exhaustivo de los componentes de una oración. Para este análisis se ocupa procesamiento de lenguaje natural, análisis de texto y lingüística computacional. El análisis encuentra información subjetiva. Este análisis es útil para la toma de decisiones ya que se conoce que siente las personas acerca de un tópico o de un producto. La creación del archivo arff es generado en código PERL.

El siguiente trabajo está dividido en Método, donde se explica cómo creamos el clasificador; Resultados, donde desplegamos los resultados obtenidos por el procesador de texto y el clasificador en sí; Discusión donde analizamos los resultados obtenidos, Conclusiones del trabajo obtenido, y finalmente el trabajo futuro donde se explica cómo se puede implementar mejoras y sugerencias para mejorar partes del sistema.

# Método

Antes del análisis del texto se han creado varios archivos que contienen diferentes tipo de palabras. Archivo de sustantivos, preposiciones, adjetivos positivos, adjetivos negativos, conjunciones, verbos positivos y verbos negativos.

## Atributos

Para la separación de oraciones utilizamos la expresión regular [^\.]+. La cual separa oraciones por el punto final. Cada oración se va recorriendo palabra por palabra.

Cada palabra de la oración es comparada con las palabras en los archivos y el valor del atributo de esta oración se coloca en sí o en no al finalizar la oración. Este proceso se realiza en los atributos noun, prepositional, CoordinatingConjunction, SubordinatingConjunction, ConjuntiveAdverbs y puntuacion. Para los atributos verbPositive, verbNegative, adjetivePositive y adjetiveNegative en cambio con cada coincidencia se aumenta un contador para dar el total del atributo en la oración. Para numPalabras todas las palabras de la oración aumentan un contador para el total de palabras.

Todos los conjuntos de palabras son pertenecientes a listas de archivos donde se tiene recopilaciones de verbos positivos, verbos negativos, adjetivos positivos, etc. Las recopilaciones no son del todo completas se usaron las palabras más comunes por simplicidad además existen muchas palabras que podrían ser malas o buenas pero dependiendo del contexto y al incluirlas podrían dañar el modelo haciendo sobre entrenamientos en casos muy específicos.

Se consideró importante contar los verbos y adjetivos negativos y positivos en vez de solo comprobar su existencia ya que un texto puede tener ambos y podría ser inexacto solo ver si existen por ejemplo si un texto tiene solo un adjetivo positivo y muchos negativos consideramos que el texto probablemente sea negativo.

## Programa

Para la clasificación en positivo, negativo o neutro se lo realiza manualmente en una cantidad determinada (200 oraciones). Para el resto de oraciones se las clasifica por medio del algoritmo de aprendizaje y evaluadas para ver si la clasificación fue correcta.

Primero se hace un procesamiento del texto para en el programa en PERL. En este primer procesamiento se convierte el archivo de texto donde está el libro escrito normalmente y se lo procesa para convertirlo en un archivo arff. Primero se crea un archivo donde se separan en oraciones a todo el texto.

El mismo programa PERL va leyendo oración por oración y va sacando los atributos que tiene luego se genera el archivo arff con todas las oraciones que se analizan. Luego en java con las librerías de weka se puede clasificar los textos arff. Primero se debe crear el texto de entrenamiento a los archivos generados por PERL se debe clasificar las oraciones ya que solo están con los atributos pero no se determina si la oración es positiva, negativa o neutra. Para hacer el modelo se debe clasificar antes algunas oraciones para a partir de esas oraciones clasificadas pueda clasificar nuevas correctamente con los atributos del archivo arff.

# Resultado

Al ejecutar el código en PERL se genera un archivo arff con la relación, atributos y los datos.

https://lh4.googleusercontent.com/MgR8fbOisCq9_3Q53NyFK4Mx1_EbBBjUVJRe-wFmQW7_KlW2c-E_iIkdhhxcUDQPw5NLxNb382uUy5ud4Ml0R8UcmuM-ZRRevPrOgHpLvk0MDYblW9qVWpBQMJGJQjb13uQ0dF6w

Figura 1. Relación creada por código.

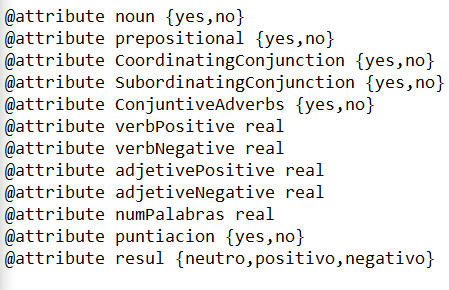


Figura 2. Atributos generados.

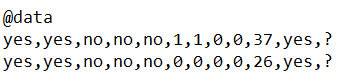


Figura 3. Datos generados de las oraciones.

En la figura 3 no existe la clase ya que solo es lo que genera el código en PERL. Esto se llena manualmente. Este archivo no es de entrenamiento este archivo generado tiene la clase con un “?” es decir se desconoce el resultado y se espera que se clasifique con el modelo previamente entrenado.

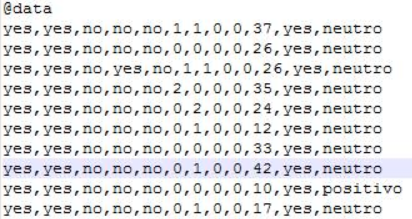


Figura 4. Clasificado manualmente.

Para clasificar los demás datos utilizamos un clasificador de Bayes en cual clasifica a los demás datos con una precisión del 95. 0916 % y en J48 obtuvimos una precisión del 95.8242 %.

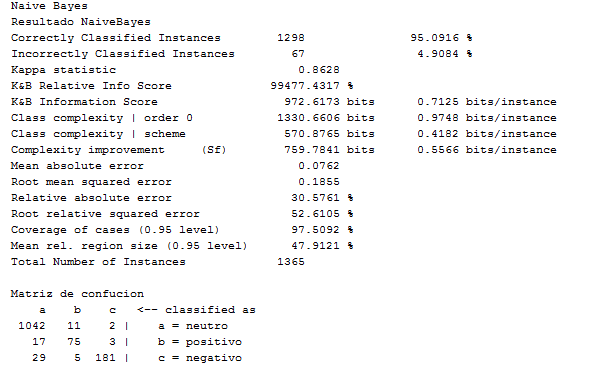


Figura 5. Precisión con Naive Bayes.

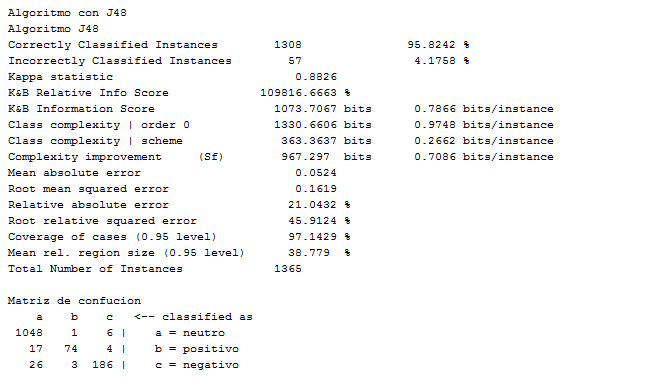


Figura 6. Precisión con J48

# Discusión

Al momento de realizar la clasificación se observa que la precisión es relativamente alta, con un 95% en promedio. Esto significa que los atributos son significativos para analizar los sentimientos de las oraciones.

Al analizar las oraciones no todas las palabras estaban incluidas entre los atributos dependiendo del texto si es un lenguaje no tan común. En las listas de palabras se incluía lenguaje muy usado en algunos libros se usa lenguaje de más alto nivel y el archivo arff no se generaría del todo bien.

# Conclusiones

El algoritmo que se escoja para la clasificación influye mucho en la precisión.

El reto mayor al momento de generar un analizador de sentimientos es encontrar los atributos correctos para que el aprendizaje sea válido y no exista overfitting,

Se podría tener más exactitud si se hace un modelo diferente para un tipo de texto no es lo mismo analizar un texto que puede estar en un libro y ser formal a analizar un texto que se encuentra en una red social por ende un modelo general para los dos sería muy difícil de crear.

Dentro de textos formales hay atributos que no son necesarios como la puntuación ese atributo se podría considerar para analizar comentarios en redes sociales

# Trabajo futuro

Se deben hacer listas de verbos y adjetivos más completas que incluyan palabras no tan usadas en el lenguaje común pero sí en textos formales si se quiere hacer un mejor análisis de oraciones en libros. Atributos que miren posiciones de palabras para una clasificación mejor.

Se implementara un clasificador con la integración del tree tager (que es un clasificador de palabras) para poder clasificar de mejor manera los sentimientos, y posteriormente implementar un aprendizaje no supervisado y así evitar la parte de la clasificación manual del archivo arff de entrenamiento

1. [↑](#footnote-ref-1)