Un Nuevo Método para Clustering de Tweets Basado en Métodos de Ensambles y Técnicas de Hashing

Cecilia Kraiselburd*, Matias Gentile*, Bruno Varani*, Fabricio Neirotti*,
Eduardo Amar, Juan Moine y Cristian Bigatti.
Universidad Tecnológica Nacional – Facultad Regional Rosario,
Zeballos 1341, Rosario, Argentina
Grupo de Investigación en Minería de Datos
Email:cristianbigatti@gmail.com

Abstract

Este trabajo tiene como objetivo abordar un nuevo método de clustering basado en métodos de ensambles aplicados a datos no estructurados provenientes de la red social Twitter. Se aplicó particularmente el método de clustering por acumulación de evidencia (EAC). Dicha técnica brinda la posibilidad y ventaja de generar una matriz de distancia entre tweets para la posterior aplicación de algoritmos de clustering sobre la misma con una combinación óptima de parámetros. Previamente a su aplicación se utilizó la técnica de Minhash mediante el desglose de tweets en n-gramas para optimizar el cálculo. Los resultados obtenidos son prometedores, mostrando la utilidad del método aplicado para descubrir grupos temáticos de tweets a partir de un conjunto grande de datos obtenidos de Twitter.

1. Introducción

La minería de texto se puede considerar como una extensión de la minería de datos, siendo su objetivo principal el descubrir nueva información o patrones a partir de datos no estructurados [1]. Esta definición coincide con la presentada en el artículo *Text Mining: The state of the art and the challenges* del autor Ah-Hwee Tan donde se señala que la minería de texto es un proceso en donde se extraen patrones que sean interesantes y no triviales generando conocimiento en base a documentos de texto [2]. En particular, plataformas sociales como Twitter pueden ser utilizadas como una fuente de datos novedosa e interesante para realizar este tipo de estudios, beneficiándonos además al poseer una arquitectura de datos abiertos [3].

En una época donde las herramientas sociales son de gran importancia para marcar tendencias y generar opiniones, las técnicas de clustering aplicadas a este tipo de datos pueden ser de utilidad para detectar temas de interés o incluso ser de ayuda para el posicionamiento de tweets asociados a ellos. Con el objetivo de estudiar las temáticas dentro de conjuntos de tweets, hemos combinado el uso de los conceptos de minería de texto y las plataformas sociales, con técnicas de clustering para así formar distintos grupos de tweets en base a las características intrínsecas que posean los mismos.

Tal como se menciona en el artículo *Theme Based Clustering of Tweets* de Rudra M. Tripathy y otros, aplicar técnicas de clustering a tweets supone un difícil desafío debido a las limitaciones de caracteres a utilizar y a la falta de reglas gramaticales empleadas en los mismos [4]. Es por esto que en el presente trabajo se expone el uso de técnicas como *Minhash* [5] para el cálculo de las distancias entre tweets, y además se emplea el novedoso algoritmo de ensamble de clustering basado en acumulación de evidencia (Evidence Accumulation Clustering o EAC por sus siglas en inglés) [6] con el fin de proporcionar una mejora en el agrupamiento de dichos extractos de texto.

A continuación, este artículo será expuesto siguiendo la siguiente estructura. En la segunda sección se enunciarán las técnicas y herramientas que fueron utilizadas para el desarrollo de los experimentos. En la tercera sección, se expondrán los resultados de los experimentos realizados. Finalmente. en la cuarta sección se discutirán las conclusiones sobre los resultados obtenidos.

2. Método

*Los cuatro primeros autores contribuyeron de igual forma al desarrollo del trabajo

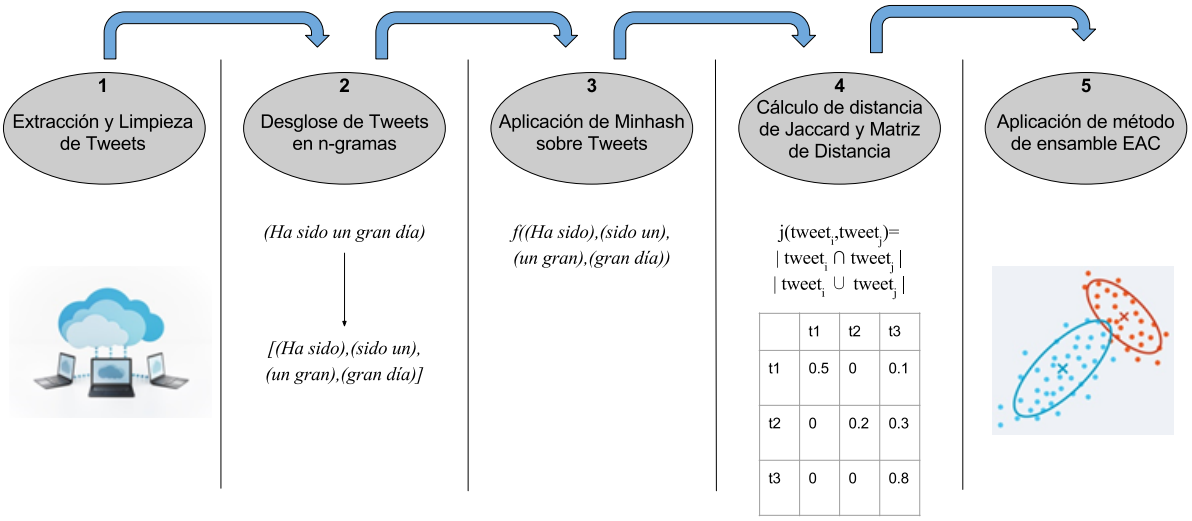

Figura 1. Secuencia de pasos del método

El método presentado en este trabajo se compone de una serie de pasos que consisten en la extracción y limpieza de los tweets desde la plataforma abierta de Twitter, la confección de una matriz que refleje la distancia entre cada uno de ellos y la posterior creación de clusters basados en su temática subyacente utilizando dicha matriz como entrada del algoritmo de clustering. Estos clusters reflejarán de una manera apropiada los grupos de tweets de acuerdo al tema de discusión más representativo de los mismos. La Figura 1 muestra un esquema de los pasos. El detalle para cada uno de ellos será desarrollado a continuación.

2.1. Extracción y Limpieza de Tweets

En primer lugar, se realiza la extracción de tweets a través de las herramientas provistas por Twitter. Caracteres como los signos de puntuación, comillas, emojis, espacios en blanco excesivos, caracteres especiales, acentos y referencias a URLs entorpecen el análisis de una cadena de texto ya que evitan que el contenido principal, es decir, aquellas palabras importantes que denotan la temática del tweet, sean analizadas con claridad, por lo tanto, deben ser eliminadas.

Luego, se eliminan todos aquellos tweets que se encuentren repetidos a causa de los retweets (citas de un tweet dentro de otro tweet). Esto se realiza porque el propósito es agruparlos en base a su temática, por lo cual basta con la ocurrencia única de cada uno de ellos. Los tweets resultantes de este procesamiento serán utilizados para la etapa siguiente.

2.2. Desglose de Tweets en n-gramas

Luego de la etapa anterior, se procede a desglosar los tweets en *n-gramas*. El método de n-gramas [7] consiste en construir a partir de cada tweet un conjunto de cadenas cortas de caracteres que se encuentran dentro del mismo. Los tweets que sean más similares entre sí compartirán varias de estas cadenas de caracteres.

Una aplicación posible de los n-gramas es considerar los conjuntos de caracteres que conforman palabras completas. Para esta forma de aplicación, el valor "n" determina la cantidad de palabras en las cuales se dividirá el tweet. Por ejemplo, si el n-grama es de n=3, su aplicación en el tweet "Hoy es un día soleado" dará como resultado el siguiente conjunto S:

`S={[Hoy es un], [es un día], [un día soleado]}.`

En cambio, si n=2, el conjunto será:

`S={[Hoy es],[es un], [un día], [día soleado]}.`

Distancia entre tweets. Luego de desglosar los tweets en n-gramas, debe construirse el conjunto de datos de entrada para poder aplicar el algoritmo de clustering en la siguiente etapa. Para ello analizamos los n-gramas obtenidos y utilizamos la distancia denominada *Jaccard*, comúnmente utilizada en la literatura para objetos de texto [8], para determinar qué tan parecidos son los n-gramas entre sí. La distancia de Jaccard se define, en términos de conjuntos, como:

$$J(S,T) = \frac{|S \cap T|}{|S \cup T|}$$

siendo S y T dos conjuntos de palabras.

La aplicación directa de la distancia de Jaccard en un conjunto grande de tweets puede traer aparejada una dificultad de procesamiento derivada de dos factores: la disparidad entre la cantidad de n-gramas en tweets de

longitudes diferentes; y la complejidad en el cálculo para cantidades grandes de tweets. Para enfrentar estas dos dificultades utilizaremos la función minhash, explicada a continuación.

2.3. Aplicación de Minhash sobre Tweets

Una función de *hashing* es un algoritmo que permite la conversión de una entrada, este caso un texto, en una salida numérica o alfanumérica que sirve de resumen de la entrada original. En la Figura 2 se muestra un ejemplo de aplicación para la función de hashing.

La técnica de minhashing permite la conversión de un n-grama de palabras en un número entero a través de la aplicación de n funciones de hashing a cada palabra y seleccionando el mínimo valor obtenido para cada una de ellas. De esta manera, cada tweet queda representado por un valor de longitud fija, solucionando la problemática de la variabilidad en la longitud de cada tweet. Además, el proceso de hashing aplicado al conjunto de tweets permite un almacenamiento y comparación de los mismos mucho más eficiente ya que se trabaja directamente con números.

Posteriormente, utilizando los valores de hash obtenidos, se construye la matriz de distancia entre cada uno de los tweets utilizando la ya definida distancia de Jaccard, como se verá a continuación.

2.4. Cálculo de Distancia de Jaccard y Matriz de Distancia

Una vez que cada tweet tiene asociada una medida de hashing, se procede a calcular la distancia de Jaccard entre ellos haciendo uso de los hashes asociados. A partir de estás distancias se confeccionará una matriz de orden n x n donde n es la cantidad de tweets contenida en el conjunto de datos original. Dado que se deben comparar todos los tweets entre sí, la matriz será una matriz cuadrada y simétrica, por lo cual, basta con calcular solo una de las partes triangulares junto con la diagonal de la misma con el fin de optimizar la capacidad de procesamiento.

2.5. Aplicación de Método de Ensamble EAC

Clustering. La técnica de clustering refiere a la tarea de agrupar elementos en clusters o grupos de manera tal que los elementos dentro del mismo cluster sean lo más similares posible entre ellos y, a su vez, lo más diferentes entre los otros grupos [9].

Existen distintos tipos de algoritmos de clustering, que pueden ser clasificados en clusters jerárquicos o clusters no jerárquicos [10].

Al trabajar con un conjunto de datos compuesto por tweets, se encuentran numerosas dificultades a la hora de analizarlos. Algunos de estos problemas provienen del propio lenguaje mientras que otros resultan de las nuevas formas de escritura provenientes de Internet (abreviaturas, emojis, etc).

Figura 2. Ejemplo de hashing

Es por esto que cada uno de los tweets que se extrae dispone de un alto contenido de ruido, es decir, de palabras o expresiones que no aportan contenido o información que ayude a la clasificación de los mismos y por lo tanto deberán ser eliminados previo a analizarlos.

k-means. k-means es un algoritmo de clustering que tiene como objetivo agrupar las observaciones en k clusters minimizando la distancia media cuadrada entre los mismos (distancia media cuadrada inter-cluster). Para esto se utiliza comúnmente la distancia euclídea, lo cual genera clusters de forma hiperesférica [9].

La idea principal consiste en definir k *centroides* al comienzo (uno por cada cluster) en forma aleatoria. Los centroides son vectores en el espacio de características del conjunto de datos de entrada. Luego se realizan los siguientes pasos:

- Asignar a cada observación el centroide más cercano.
- Re-calcular los centroides de tal forma que minimicen la distancia media cuadrada inter-cluster.

Estos dos pasos se repiten hasta que no haya cambios en las asignaciones de cluster para cada observación.

PAM. Particionamiento Alrededor de Medoides (PAM) es un algoritmo de clustering que permite, a diferencia de k-means, usar como entrada una matriz de distancias del dataset utilizado en el estudio. A partir de esta matriz, genera agrupamientos entre las ocurrencias del dataset basándose en la minimización de las distancias entre ellas, de manera tal que aquellas "más cercanas" se agrupen en un mismo cluster [9]. PAM es un algoritmo de clustering no jerárquico, al cual se le debe ingresar como parámetro el número de clusters que se desean obtener.

Un concepto utilizado en este algoritmo es el de *medoide*. Un medoide es un objeto de un cluster tal que su distancia a los demás objetos del mismo cluster sea mínima. Una vez definidos los medoides, cada objeto del conjunto de datos será asignado a aquel medoide más cercano.

El objetivo del algoritmo será minimizar la función:

$$Función\ Objetivo = d(i, m)$$

Dónde *i* representa un objeto y *m* su medoide más cercano. Es decir que su objetivo será minimizar la suma de las distancias entre los objetos y el medoide de su cluster.

En primer lugar, el algoritmo elige al azar k medoides (objetos del conjunto de datos), siendo k el número de clusters pre-definidos. Posteriormente calcula la función objetivo en base a estos objetos. Si el resultado de la función objetivo puede ser disminuido intercambiando un objeto por uno de los medoides seleccionados, entonces se realiza esta acción y se procede a ejecutar nuevamente este paso.

Métodos de ensamble. Una forma de mejorar la eficiencia de los algoritmos de clustering es a través de los métodos de ensamble. Los métodos de ensamble de clusters consisten en recolectar información de diferentes soluciones de clustering, que pueden ser métodos diferentes o combinaciones del mismo método con distintos parámetros, a fin de determinar la frecuencia en la cual dos puntos se clasifican siempre dentro del mismo cluster. La técnica de ensamble utilizada en este trabajo será EAC (Evidence Accumulation Clustering)

La idea del método EAC consiste en combinar los resultados de múltiples ejecuciones de clustering con diferentes atributos. Para ello se usa un sistema de votos, de modo que los patrones que aparecen en los mismos clusters en diferentes ejecuciones se les dan una mayor importancia que en los que no aparecen. Estos resultados son guardados en una *matriz de co-asociación*, donde cada elemento *(i, j)* representa la similitud entre la observación i y la observación j.

Luego, la matriz de co-asociación puede convertirse a una matriz de distancia (restando 1 - Matrix Co-asociación, ya que las co-asociaciones representan similitudes) y de este modo aplicar algoritmos de clustering que acepten matrices de distancia como entrada. Un ejemplo de estos algoritmos son k-medoides como PAM o jerárquicos. Esto da como resultado la agrupación natural de las observaciones. En este caso se hará uso del algoritmo PAM variando el parámetro k como entrada para el método EAC.

3. Experimentos y Resultados

A modo de ejemplo se trabajará con un conjunto de 35.749 Tweets referidos a dos temáticas diferentes: la primera referida a un partido del jugador Carlos Tevez en Boca Jrs. [11] (Conjunto de datos "Carlitos") y la segunda referida a la detención del empresario Lázaro Báez [12] (Conjunto de datos "Lázaro"). Todos los tweets fueron mezclados aleatoriamente para comenzar el experimento. Para llevar adelante la práctica de este caso, se utilizó el lenguaje de programación Python en su versión 3.5 y las siguientes librerías: scikit-learn [13], NumPy [14], datasketch [15] y Twython [16]. Se realizaron pruebas similares con el lenguaje R mediante el uso de la librería twitteR [17] y tm [18] para manipulación de texto, pero se encontró que los procesos llevados a cabo con Python lograban mejor rendimiento.

Extracción. Para la recolección de los tweets se hizo uso de la Streaming API [19] de Twitter a través de la librería Twython. Dicha API nos permitió recolectar tweets en tiempo casi real según un término de búsqueda. En nuestro caso fueron los términos correspondientes a los temas "Carlitos" y "Lázaro".

El texto de los tweets fue luego guardado en un archivo de texto plano (.csv), indicando la temática que pertenecía cada tweet en base a dicho término de búsqueda.

Limpieza de tweets. Para la limpieza del conjunto de datos, se eliminaron aquellos tweets duplicados y/o vacíos. Además, se eliminaron del texto de cada tweet las URLs, retweets, menciones y caracteres especiales ya que como se menciona en la Sección 2.1 los mismos no fueron tenidos tampoco en cuenta dentro del caso de estudio.

Este proceso dio como resultado 6.300 tweets, de los cuales resultaron en 2.400 de la temática "Carlitos" y 3.900 de la temática "Lázaro".

Dado que se extrajeron los tweets mediante la utilización de términos de búsqueda, se corrió el método de clustering con y sin los mismos.

k-means - Coseno – Hashing. Se eligió el algoritmo de clustering k-means como referencia para comparar nuestro método. Para esto se eligió un valor de k=2, dado que el objetivo era el de comparar la pertenencia de cada tweet a una de las dos temáticas definidas para el estudio.

Como entrada se utilizaron n-gramas de longitud 1 y de longitud 1 a 3. Éstos fueron normalizados (L2), vectorizados y posteriormente se aplicaron técnicas de hashing.

La normalización L2 es un vector normalizado, definido para un vector complejo *x* como la raíz cuadrada de la sumatoria de los valores absolutos de dicho vector al cuadrado. Está representada por la siguiente formula:

$$\ell^2 - norm \;=\; |x| \;=\; \sqrt{\sum_{k=1}^{n} |x_k|^2}$$

La normalización permite equiparar los valores en rangos comparables entre sí. Asimismo, el hecho de usar los datos normalizados, permite usar el algoritmo k-means con una distancia de coseno aproximada [20].

La distancia del coseno es una de las medidas más utilizadas para el clustering de texto [21]. Está definida como la diferencia angular entre dos vectores:

$$Distancia = 1 - cos(\theta) = \frac{A.B}{|A||B|}$$

Siendo A y B dos vectores no vacíos.

Para utilizarla se representan los tweets en vectores de palabras, donde cada elemento del vector es el peso de la palabra en relación a la cantidad de veces que aparece en el par de tweets a comparar.

Un ejemplo de aplicar la distancia de coseno a dos tweets es el siguiente:

Dado dos tweets:

`T1: El sol es una estrella ubicada en el centro del sistema solar`

`T2: El sol es la estrella del sistema solar`

Se combinan todas las palabras únicas en una lista:

`[El, sol, es, una, estrella, ubicada, en, centro, del, sistema, solar, la]`

La lista de palabras única se usa para representar las veces que aparece cada palabra en cada tweet. Siendo V1 el vector asociado a T1 y V2 el vector asociado a T2:

`V1: [2,1,1,1,1,1,1,1,1,1,1,0]`

`V2: [1,1,1,0,1,0,0,0,1,1,1,1]`

Luego aplicando la distancia del coseno a estos dos vectores resulta en una distancia de 0.2538.

Minhash. Se usó la técnica de MinHash para calcular la distancia de Jaccard entre cada tweet, se realizaron pruebas usando n-gramas de 1 y 2 palabras. No se probó con n-gramas de mayor longitud debido a la poca cantidad de palabras que tienen los tweets y la baja riqueza sintáctica que contienen los mismos.

PAM. Sobre la distancia de Jaccard entre tweets se aplicó el algoritmo de clustering PAM con inicialización aleatoria y un parámetro k = 2. Esta selección está basada en el hecho de que los tweets recolectados se remiten a solo dos temáticas diferentes. Debido que se aplican técnicas de clustering, la finalidad del experimento es verificar que cada tweet forme parte del uno de los dos clusters de los cuales fue extraído inicialmente.

Métodos de ensamble. Utilizando como base los datos filtrados, se aplicó la técnica de ensamble EAC sobre los mismos, con la característica de usar 30 iteraciones, un k mínimo de 2 y un k máximo de 10.

Posteriormente, sobre la matriz de distancia generada a partir de la Matriz de Co-asociación, se corrió el algoritmo de clustering PAM con inicialización aleatoria y k = 2.

Medida de precisión. La exactitud es la medida que indica la fidelidad de las pruebas. Se calcula sumando los positivos verdaderos (VP) y los negativos verdaderos (VN) sobre el total de observaciones (positivos verdaderos, positivos falsos, negativos verdaderos y negativos falsos):

$$Exactitud = \frac{VP + VN}{VP + FP + VN + FN}$$

Una exactitud cercana a 1, será indicador de que el algoritmo que estamos probando es exacto. Si es cercano a 0,5 no mejoraría a una clasificación aleatoria. En cambio, si es menor a 0,5 es probable que haya un error metodológico.

Exactitud promedio		K-means - Coseno	K-means - Coseno n-gramas = 1 a 3	PAM - Jaccard - n-gramas = 1	PAM - Jaccard - n-gramas = 2	Aleatorio
Sin EAC	Con término de búsqueda	0.881	0.833	0.841	0.499	0.502
	Sin término de búsqueda	0.503	0.483	0.543	0.389	0.502
Con EAC	Con término de búsqueda	0.843	0.835	0.868	0.698	0.502
	Sin término de búsqueda	0.493	0.488	0.606	0.412	0.502

Tabla 1. Tabla de resultados

Tabla comparativa. Se corrió cada experimento 100 veces y luego se promediaron los resultados de la medida de exactitud. Los resultados se muestran en la Tabla 1.

Se compararon las variaciones de n-gramas, uso de término de búsquedas y aplicación de la técnica EAC.

En la Tabla 1 se puede comparar en forma vertical: los valores de los experimentos realizados haciendo uso del método de ensamble EAC contra los valores sin hacer uso de dicho método. Además, se pueden apreciar los valores obtenidos en caso de remover los términos de búsquedas o dejando los mismos. Luego en forma horizontal, se comparan a modo de referencia distintos métodos de clustering. Primeramente, el método de clustering K-means usando la distancia del coseno con y sin el uso de n-gramas. Luego, la aplicación del método descrito en este trabajo usando PAM con distancia de Jaccard y MinHash con 1 y 2 n-gramas. Finalmente, una distribución uniforme aleatoria.

Se puede observar que los resultados obtenidos con el método EAC son notablemente mejores a los resultados sin hacer uso del mismo. Además, se puede observar que los resultados obtenidos con término de búsqueda también son mejores a los obtenidos sin término de búsqueda. Además, es visible que el método de PAM-Jaccard con n-gramas = 1 mejora los resultados de K-means - Coseno cuando se remueve el término de

búsqueda; esto mismo se repite para todos los casos restantes.

4. Conclusión

En este trabajo se incursionó en la técnica de cluster basada en métodos de ensamble mediante el previo desglose de los tweets en n-gramas y la aplicación de técnicas de hashing. Dichas técnicas aplicadas sobre los tweets desglosados fueron utilizadas en busca de la optimización del proceso general de clasificación y comparación de los tweets a través del uso de una matriz de distancia.

Se describió en un diagrama el proceso general llevado a cabo, y se demostró el estudio con un caso práctico que permitiera mostrar la verosimilitud y factibilidad de la utilización de dichos métodos al afrontar la tarea de clustering en un estudio de minería de texto.

Los resultados obtenidos muestran que el método presentado supera cuantitativamente a los resultados obtenidos con los métodos utilizados comúnmente para realizar agrupamientos de texto, con lo que nuestro método puede servir de punto de partida para desarrollar nuevos estudios sobre el descubrimiento automático de la temática de cadenas de texto, en particular de la red social Twitter.

5. Referencias

[1] Aggarwal, C. C., & Zhai, C. (2012), "Mining text data", *Springer Science & Business Media*.

[2] Tan, A. H. (1999, April), "Text mining: The state of the art and the challenges", in *Proceedings of the PAKDD 1999 Workshop on Knowledge Discovery from Advanced Databases* (Vol. 8, pp. 65-70).

[3] Ritter, A., Etzioni, O., & Clark, S. (2012, August), "Open domain event extraction from twitter", in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1104-1112). ACM.

[4] Tripathy, R. M., Sharma, S., Joshi, S., Mehta, S., & Bagchi, A. (2014, March), "Theme Based Clustering of Tweets", in *Proceedings of the 1st IKDD Conference on Data Sciences* (pp. 1-5). ACM.

[5] Broder, A. Z. (1997, June), "On the resemblance and containment of documents", in *Compression and Complexity of Sequences 1997. Proceedings* (pp. 21-29). IEEE.

[6] Fred, A. L., & Jain, A. K. (2005), "Combining multiple clusterings using evidence accumulation", IEEE *transactions on pattern analysis and machine intelligence*, 27(6), 835-850.

[7] Figueroa, A., & Atkinson, J. (2012), "Contextual language models for ranking answers to natural language definition questions", *Computational Intelligence*, 28(4), 528-548.

[8] Huang, A. (2008, April), "Similarity measures for text document clustering", in *Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008)*, Christchurch, New Zealand (pp. 49-56).

[9] Xu, R., & Wunsch, D. (2008), *Clustering (Vol. 10)*. Wiley-IEEE Press.

[10] Jain, A. K. (2010), "Data clustering: 50 years beyond K-means", *Pattern Recognition Letters*, 31(8), 651–666. http://doi.org/10.1016/j.patrec.2009.09.011

[11] http://deportes.telam.com.ar/notas/201604/142256-boca-bolivar-copa-libertadores-futbol-boca-deportivo-cali.html. Último acceso: agosto 2016.

[12] http://www.telam.com.ar/notas/201604/142134-detencion-lazaro-baez-san-fernando.html. Último acceso: agosto 2016.

[13] http://scikit-learn.org/. Último acceso: agosto 2016.

[14] http://www.numpy.org/. Último acceso: agosto 2016.

[15] https://github.com/ekzhu/datasketch. Último acceso: agosto 2016.

[16] https://github.com/ryanmcgrath/twython. Último acceso: agosto 2016.

[17] https://cran.r-project.org/web/packages/twitteR/index.html. Último acceso: agosto 2016.

[18] https://cran.r-project.org/web/packages/tm/index.html. Último acceso: agosto 2016.

[19] https://dev.twitter.com/streaming/overview. Último acceso: agosto 2016.

[20] G. Qian, S. Sural, Y. Gu and S. Pramanik, "Similarity between euclidean and cosine angle distance for nearest neighbor queries", *Proc. ACM Symp. Applied Computing*, pp. 1232-1237

[21] B. Larsen and C. Aone, "Fast and effective text mining using linear-time document clustering", in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999.