Especialización en Explotación de Datos y Descubrimiento del Conocimiento

Universidad de Buenos Aires

Trabajo Práctico de Especialización

Extracción de Tópicos de un sitio Web de Noticias

Adrián Kowal

Tutor: Marcelo Soria

Contenido

Resumen

Debido a la gran cantidad de datos generados hoy en día (Tweets, Mails, comentarios de clientes, Páginas Web, Blogs) nos vemos en la necesidad de procesar toda esta información en forma automática. El agrupamiento de documentos y la extracción de tópicos, es una herramienta que nos permite atacar este problema para su mejor comprensión y visualización.

Este trabajo se presentó en las Décimas Jornadas de Data Mining, 6 de noviembre de 2015, panel de presentaciones de alumnos de la Maestría en Data Mining - UBA

Objetivo

El presente práctico tiene por objetivo el agrupamiento y clasificación de Noticias de la página Web de un diario por medio de técnicas de Text Mining. La Minería de Textos se utiliza para poder extraer información a partir de datos poco estructurados o no estructurados como los documentos de textos. Para dicho análisis, los mismos se agrupan en un "corpus" el cual debe ser lo suficientemente amplio como para representar toda la variedad del lenguaje utilizado en cada una de las distintas secciones del diario.

La formación de grupos tiene por objetivo el poder agrupar las noticias según el "tema" del cual están hablando. El motivo principal para ello es que no siempre la sección del diario corresponde exactamente con el tema del cual se habla. Si bien se debería notar una alta correlación entre los temas tratados y los grupos encontrados, la ubicación de la noticia dentro de las secciones del diario sigue siendo un tema subjetivo, el cual en este caso, queda en manos de la redacción del diario. Por ejemplo, si bien existen las secciones "Política" y "Deportes", hay noticias que hablan sobre el siguiente presidente de la AFA y su impacto en la sociedad. En este caso, esta noticia podría estar en ambas secciones.

Las técnicas mas habituales para el procesamiento de estos datos es el conteo de palabras (Bag Of Words o BOW [1]). Es decir, se arma una matriz donde cada fila es un documento y en cada columna aparece la cantidad de veces que se menciona cada palabra (una columna por cada palabra que aparece en el corpus). Estas matrices tienen como característica ser muy poco densas (ralas) debido a que solo un conjunto pequeño de palabras aparece en cada documento.

Descripción del Corpus

Se armó un corpus de texto con noticias del diario La Nación (de su RSS Feeder [2]) entre el día 19/9/2015 y el 25/9/2015. Son un total de 669 noticias. Se extrajo a partir del HTML de la web de La Nación el texto crudo (se omitió todo el hipertexto), se filtraron todos los caracteres de puntuación y tags de html.

Cada noticia, esta etiquetada como perteneciente a una sección del diario. En total, La Nación define 21 distintas categorías:

- Al Volante
- Buenos Aires
- Comercio
- Comunidad
- Cultura
- Deportes
- Economía
- El Campo
- El Mundo
- Enfoques
- Espectáculos
- Inmuebles

- Moda
- Opinión
- Política
- Revista LN
- Seguridad
- Sociedad
- Tecnología
- Turismo
- Wall Street

Se armó un set de datos con tres campos:

- Link del Articulo (se usa como id de la nota)
- Categoría asignada por el diario
- Texto del articulo

Procesamiento del Corpus

Para el procesamiento del corpus se utilizó el modelo de Bolsa de Palabras (BOW). Para esto se utilizó el paquete de R "tm" [3] el cual provee funciones para el tratamiento de los textos.

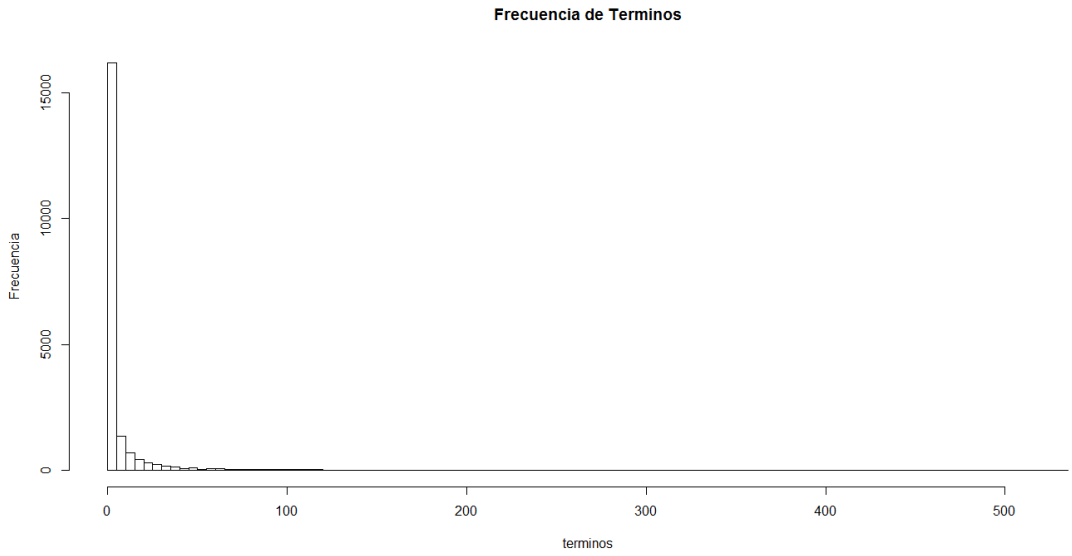
Primero se crea el corpus con la función "VCorpus". Luego de eso, se le hacen distintos filtrados ("mappings") al corpus para estandarizar los datos. Los filtros aplicados son los siguientes:

- Remoción de espacios en blanco antes y después de cada término ("stripWhitespace")
- Remoción de "stopwords". Dado que hay una proporción muy grande de palabras muy comunes en el idioma (stopwords), las cuales no tienen contenido (a, en, si, o, la, el, etc), se remueven las mismas del corpus.
- Ademas de las "stopwords", también se remueven una serie de palabras que se observó que aparecen repetidas veces y no tienen nada que ver con la nota en sí, sino mas bien con la estructura web de la nota (por ejemplo, la palabra "foto" aparece cada vez que la nota esta ilustrada, lo cual es casi siempre).
- Stemming. Con el objetivo de reducir la cantidad de palabras, se le aplica un filtro el cual le elimina una serie de sufijos a los términos, y asi llevar las palabras a su raíz (Porter Stemmer [4]).

El corpus queda asi formado (en este caso en particular) por 669 documentos y un total de 20238 distintos términos (palabras).

Cuando ya se procesó el corpus, lo que hacemos es armar la matriz de la bolsa de palabras. Esto es, se arma una matriz "TermDocumentMatrix" donde cada fila contiene un término y en cada columna contiene un documento. En cada celda se indica la cantidad de veces que el término aparece en cada documento ("TermFrequency").

Si sumásemos fila por fila (los términos no nulos), obtendríamos un vector con la cantidad de documentos en donde aparece cada término. La distribución de esos términos se puede graficar en el siguiente histograma:



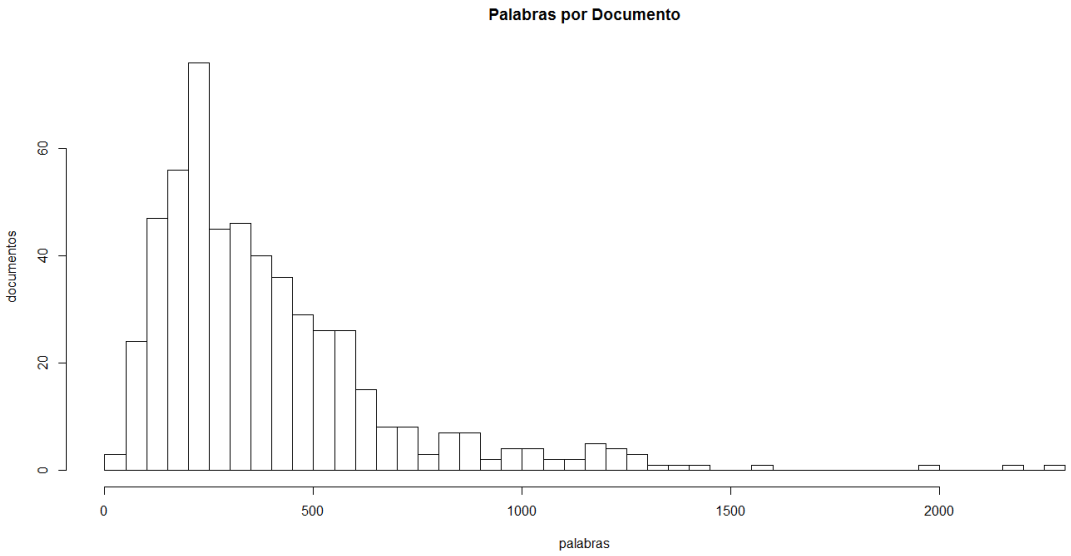
Donde se observa que dicha distribución sigue la ley de Zipf [5]. Esto indica que hay muy pocos términos que aparecen en casi todos los documentos (por ejemplo “nación”, “año”, “mas”, “también”) y hay muchos términos que aparecen en muy pocos. De hecho en nuestro caso, hay 9136 términos que se presentan solamente en un documento.

Dado que ni los términos frecuentes ni los términos extremadamente raros son útiles a la hora de clasificar, se eliminan directamente de la matriz de términos. En este caso particular, eliminamos todos los términos que aparecen en menos de 3 documentos y también se eliminan todos aquellos que aparecen en mas de la mitad.

Esto nos reduce la cantidad de términos a 6719.

Dividimos el conjunto inicial de datos en un conjunto de entrenamiento (“training”, 80% de los documentos) y uno de prueba (”testing”, 20% de los documentos)) para poder validar el clasificador. En el conjunto de entrenamiento, quedó un set de datos de 6719 términos y 535 documentos, mientras que el de prueba tiene 134 documentos.

Si observamos la cantidad de palabras por cada documento, la distribución que nos queda es la siguiente:



Donde se observa que sigue una distribución aproximadamente log-normal con una media de 394 palabras por cada documento y una mediana de 319.

Una vez filtrada la matriz de términos, se procede a estandarizar los datos. Para poder reducir el peso de las palabras frecuentes, se computa la frecuencia inversa de documento ("InverseDocumentFrequency") que es la inversa de la cantidad de documentos en los cuales aparece dicha palabra. La frecuencia de cada término (TF, "termFrequency") se multiplica por la inversa de su frecuencia de documento (IDF, "inverseDocumentFrequency" [6]) y de esta manera reducir el "peso" de las palabras que aparecen muchas veces (TF alta), pero son muy comunes (IDF baja). Luego se normaliza cada documento por su norma euclidea (norma 2) para poder comparar documentos con distinta cantidad de palabras. De esta manera, nos queda una "TermDocumentMatrix" con peso TF/IDF y normalizada según norma 2.

Ejemplo de la Matriz:

Término	126	555	491	164	641	197	349	135	51	215
aacre	0	0	0	0	0	0	0	0	0	0
aapres	0	0	0	0	0	0	0	0	0	0
abadi	0	0	0	0	0	0	0	0	0	0
abaj	0.056	0	0	0	0	0	0	0	0	0
aban	0	0	0	0	0	0	0	0	0	0
abandon	0	0.058	0	0	0	0	0	0	0	0
abarc	0	0	0	0	0	0	0	0	0	0
abastec	0	0	0	0	0	0	0	0	0	0
abel	0	0	0	0	0	0	0	0	0	0
abiert	0	0	0	0	0	0	0.054	0	0.032	0

Armado de Grupos

Se calcula la matriz de distancia, utilizando como función de similitud la distancia coseno. Luego, se arma un clúster jerárquico mediante la función hclust de R, utilizando un encadenamiento "completo" (Compete Linkage).

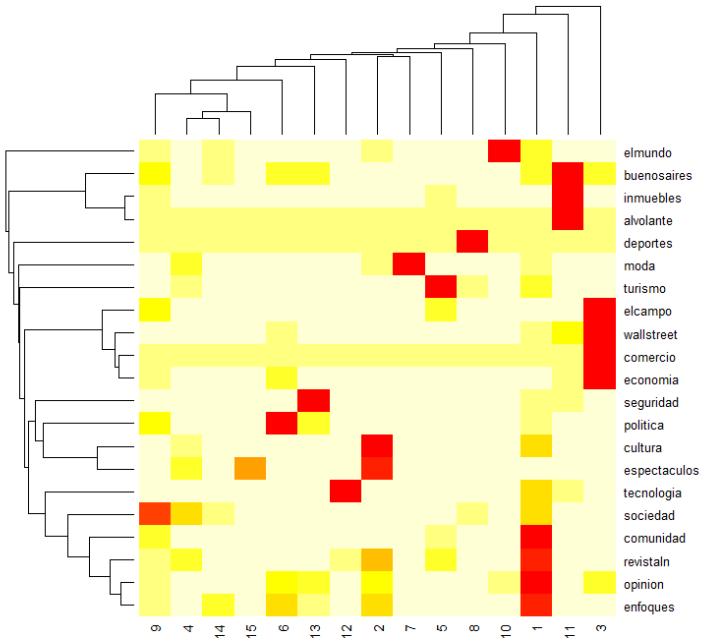
El coeficiente de correlación cofenética da 0.654, lo cual nos indica que el dendograma está representando aceptablemente las distancias de la matriz de distancias.

Elección de la cantidad de grupos

El diario propone una segmentación de las noticias en 21 categorías. La idea del presente práctico es poder identificar lo mejor posible dichas categorías analizando solo su contenido textual.

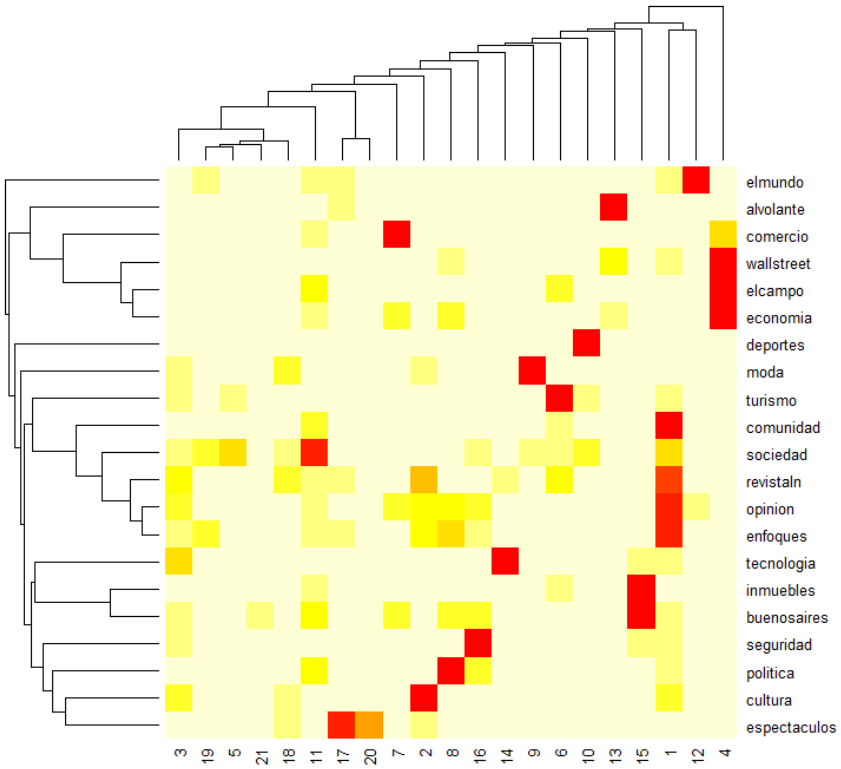
Para ello se probó con distintos k, validando como dicha agrupación separaba las categorías. Para cada valor de k (15, 21, 24) se calculó la tabla de valores (función "table") de cuantos documentos de cada categoría quedaron en cada grupo. Con dicha matriz, se graficó un mapa de calor para poder identificar la calidad de la separación.

K=15



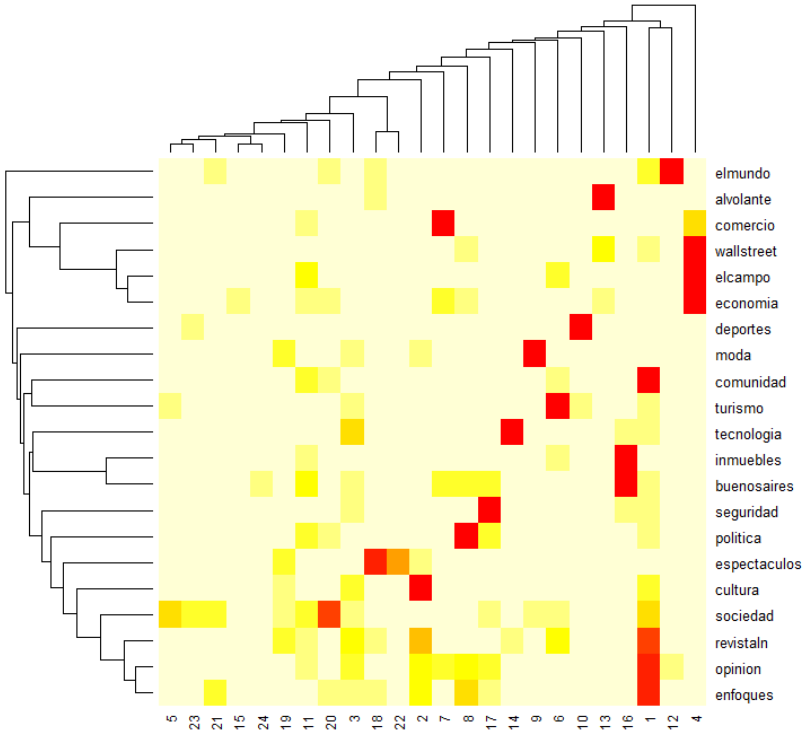
Según el mapa de calor obtenido, se puede observar que se generaron algunos grupos que abarcaron demasiadas categorías. Por ejemplo el grupo 11, incluyó muchas noticias de "buenosaires", "inmuebles" y "alvolante", las cuales tratan temas bastantes diversos y es deseable que se agrupen por separado.

K=21



En este caso, la agrupación separó aceptablemente bien las categorías. Si bien hubo algunas que se agruparon dentro del mismo grupo, dichas categorías están muy relacionadas entre sí, como por ejemplo “wallstreet”, “economía” y “elcampo”.

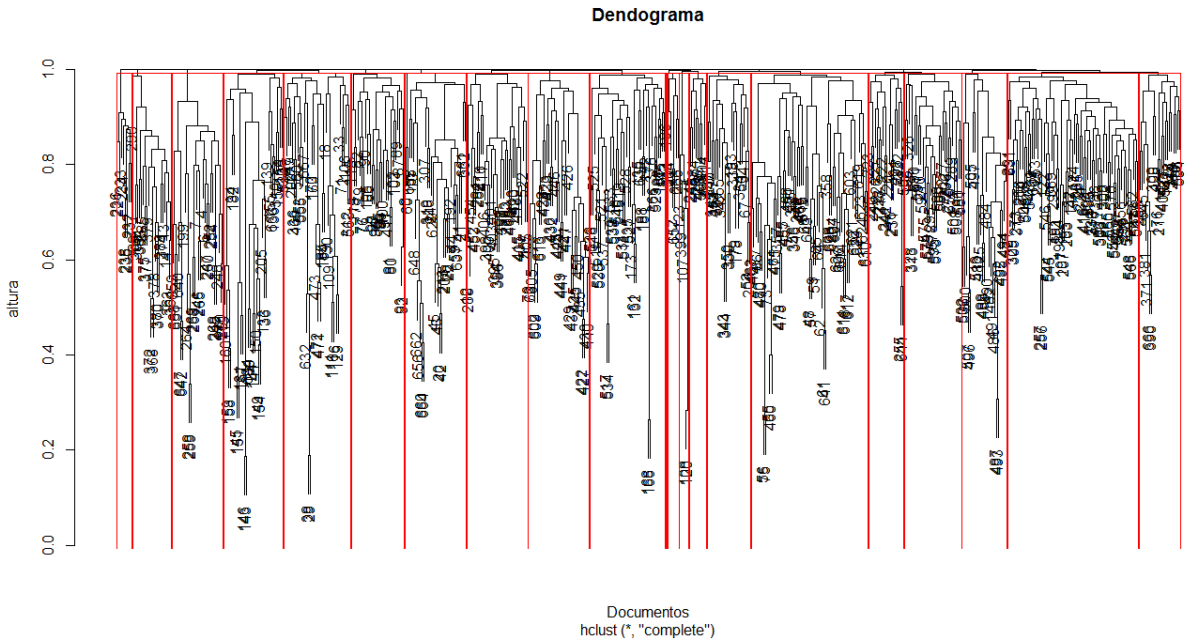
K=24



En este caso, la separación de las categorías es buena, (al igual que K=21) pero se observan muchos grupos “vacios” (es decir, con muy pocos documentos). Son todos aquellos donde no aparece ningún recuadro rojo en toda una columna. Seguir agregando particiones no agrega nada nuevo de información por lo que concluimos que la cantidad óptima de grupos es 21.

Análisis de Grupos para K=21

El dendograma con el corte en 21 grupos queda de la siguiente manera:



Donde se observan que hay algunos grupos con muchos documentos y otros con muy pocos.

La cantidad de documentos en cada grupo queda de la siguiente manera:

Grupo	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Documentos	66	29	21	59	5	31	22	31	23	27	34	26	31	20	38	30	18	9	6	8	1

Y la cantidad de documentos por categoría son:

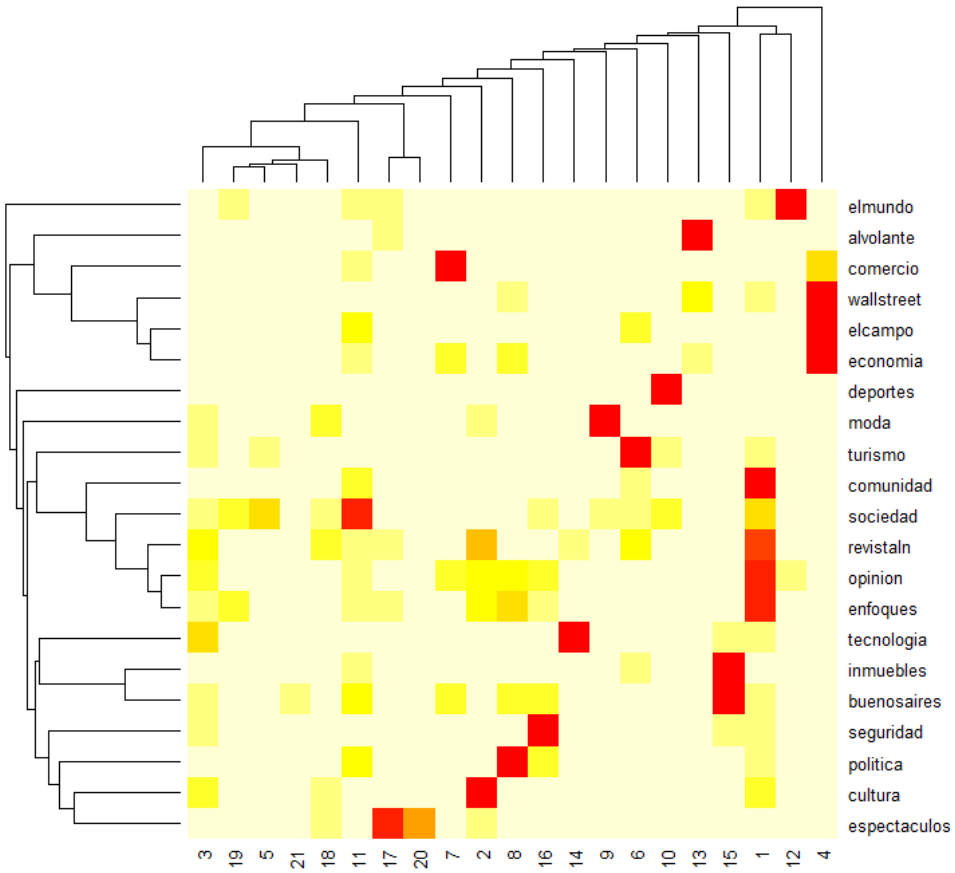
Categoría	**Cantidad**
alvolante	21
buenosaires	25
comercio	27
comunidad	26
cultura	20
deportes	24
economia	30
elcampo	21
elmundo	31
enfoques	24
espectaculos	23
inmuebles	21
moda	22
opinion	25
politica	38
revistaln	22
seguridad	26
sociedad	32
tecnologia	27
turismo	26
wallstreet	24

Si hacemos una tabla de doble entrada con los grupos y las categorías obtenemos lo siguiente:

Grupo	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
alvolante	0	0	0	0	0	0	0	0	0	0	0	0	24	0	0	0	1	0	0	0	0
buenosaires	1	0	1	0	0	0	2	2	0	0	4	0	0	0	12	2	0	0	0	0	1
comercio	0	0	0	6	0	0	15	0	0	0	1	0	0	0	0	0	0	0	0	0	0
comunidad	21	0	0	0	0	1	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0
cultura	3	15	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
deportes	0	0	0	0	0	0	0	0	0	24	0	0	0	0	0	0	0	0	0	0	0
economia	0	0	0	16	0	0	3	3	0	0	2	0	2	0	0	0	0	0	0	0	0
elcampo	0	0	0	16	0	3	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0
elmundo	3	0	0	0	0	0	0	0	0	0	1	25	0	0	0	0	2	0	2	0	0
enfoques	10	3	1	0	0	0	0	4	0	0	1	0	0	0	0	1	1	0	2	0	0
espectaculos	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13	2	0	8	0
inmuebles	0	0	0	0	0	1	0	0	0	0	1	0	0	0	22	0	0	0	0	0	0
moda	0	2	1	0	0	0	0	0	22	0	0	0	0	0	0	0	0	3	0	0	0
opinion	9	3	2	0	0	0	2	3	0	0	1	1	0	0	0	2	0	0	0	0	0
politica	1	0	0	0	0	0	0	18	0	0	4	0	0	0	0	3	0	0	0	0	0
revistaln	9	5	3	0	0	3	0	0	0	0	1	0	0	1	0	0	1	2	0	0	0
seguridad	1	0	1	0	0	0	0	0	0	0	0	0	0	0	2	21	0	0	0	0	0
sociedad	4	0	1	0	4	1	0	0	1	2	10	0	0	0	0	1	0	1	2	0	0
tecnologia	1	0	6	0	0	0	0	0	0	0	0	0	0	19	2	0	0	0	0	0	0
turismo	2	0	2	0	1	22	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
wallstreet	1	0	0	21	0	0	0	1	0	0	0	0	5	0	0	0	0	0	0	0	0

Donde se puede observar que la matriz es muy rala, lo cual indica una buena aproximación de los grupos con las distintas categorías de las noticias.

Si visualizamos la matriz anterior como un mapa de calor, obtenemos lo siguiente:



Donde se puede apreciar la concentración de categorías en los grupos.

Análisis de los Grupos

A continuación vamos a analizar cada uno de los grupos para intentar identificar su contenido. Para ello, vamos a listar las primeras 20 palabras de mayor peso (es decir, que mas aparecen y no son tan frecuentes) las cuales deberían ser de alguna manera, el “tema” (topic) de dicho grupo. Solo se muestra la palabra (stemizada) sin el peso, pare mejorar la lectura.

Grupo	Notas	Keywords	Descripción
1	66	vid escuel comun person padr dic hij gent estudi social	Mayormente agrupa noticias de comunidad.
2	29	novel escritor cultur literatur obra libr adn literari personaj pagin	Grupo de cultura (literatura)
3	21	smartphon apple motorol fon tel iphon mot bateri celular tiend	Mayormente Tecnología (smartphones)
4	59	preci inversion bon millon soj economi merc bols banc valor	Economía y elcampo
5	5	xic tegui lim peru restaur baquean astrid cociner maid cabrer	Grupo pequeño de noticias de sociedad
6	31	turism restaur vin chef biciclet pinguin libretit hul ingredientel monomani	Grupo de Turismo
7	22	comerci portuari exterior terminal maritim contenedor logist export aduaner puert	Economía (comercio exterior)
8	31	scioli candidat gobern debat mass presidencial macri eleccion peron kirchn	Política
9	23	canc lenceri diet ejercici fashion crem gel week bellez fisic	Moda
10	27	deport jugador entren cop futbol torne jug riv gol jueg	Deportes
11	34	barrick cianur derram velader cuenc socied min miner ambiental polit	Sociedad (Derrame de cianuro en San Juan)
12	26	francisc pap onu washington sophi cub discurs kimoon york reuni	El Mundo. Noticias internacionales (visita del Papa a Cuba y EEUU)
13	31	vehicul volant motor automovil aut traser model die sel delanter	Automóviles (alvolante)
14	20	virtual adob samsung anteoj impresor televisor arqueolog tecnologi simul motorol	Tecnología (televisores y otros gadgets)
15	38	inmuebl comercial porten metr edifici oficin ampli ubic zon coment	Inmuebles
16	30	acus farr band tribunal narcotraf juici conden rob arroz suris	Policiales
17	18	espectacul teatr festival album dirig cancion pelicul music cin estren	Espectáculos
18	9	bellez contest recital prestigi agit pianist anthony inculc mujer itali	Revista La Nación
19	6	twitt selfi renunci instagram republican pap muj the visit boehn	Grupo pequeño de noticias de Sociedad y enfoques
20	8	guion duracion elenc apta calificacion distribuidor fotografi espectacul pelicul direccion	Otro grupo de Espectáculos
21	1	recolet rtig adrenalin figuero pilot truc alcort cautiv pobl callejer	Grupo con una sola nota de “buenosaires”

Conclusiones de los Grupo

Se pudo observar que los documentos se agruparon en forma bastante razonable. Cabe aclarar que no es del todo válido validar el agrupamiento con las secciones del diario en sí, dado que como se demostró en algunos grupos, puede haber noticias de “Deportes” que hablen mas de

política que de deportes en sí. Otro ejemplo es la sección “Comunidad”, donde se pueden encontrar noticias de salud, seguridad o economía.

De cualquier manera, concluimos que el agrupamiento propuesto es válida para este tipo de noticias.

Clasificación

Con los grupos formados, vamos a hacer un clasificador para poder identificar el tema que trata un nuevo documento.

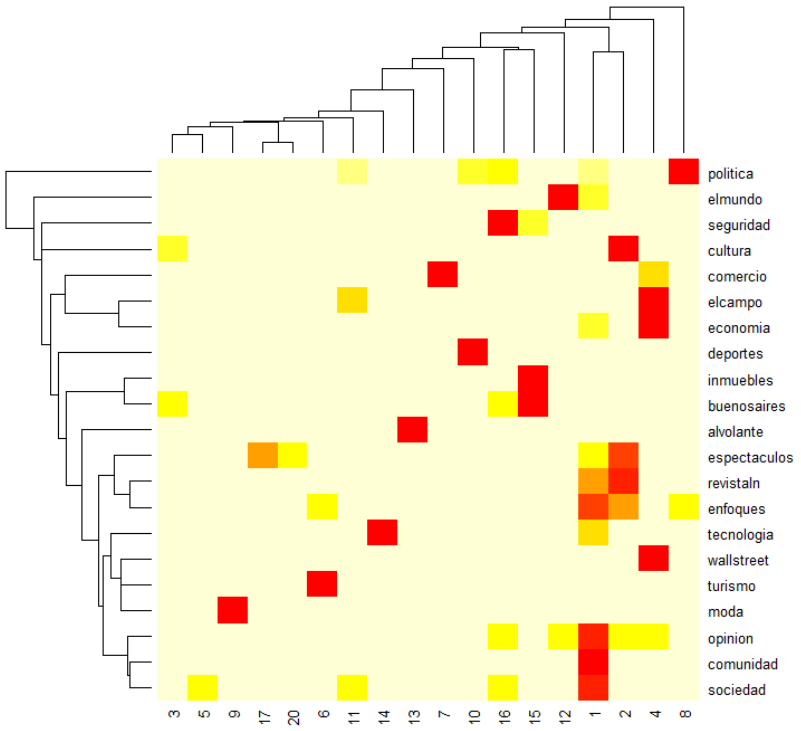
Para ello, calcularemos la distancia coseno entre el documento y los 21 centroides formados por el promedio (normalizado) de los documentos de cada grupo (Clasificador de Rocchio [7]). Clasificaremos el nuevo documento según el centroide mas cercano.

Dado que tenemos muchas clases posibles, para medir la performance de este clasificador (accuracy) asociaremos las noticas de cada categoría a un determinado grupo, según los recuadros indicados en rojo del mapa de calor. Cuando una nota del conjunto de testeo de determinada categoría se agrupe en dicho grupo, diremos que está bien clasificada. De lo contrario será un caso negativo.

La clasificación es la siguiente:

Categoría	Grupo Clasificación
alvolante	13
buenosaires	15
comercio	7
comunidad	1
cultura	2
deportes	10
economia	4
elcampo	4
elmundo	12
enfoques	1
espectaculos	17,20
inmuebles	15
moda	9
opinion	1
politica	8
revistaln	1
seguridad	16
sociedad	11
tecnologia	14
turismo	6
wallstreet	4

Clasificamos de esta manera los 134 documentos del conjunto de datos de Testeo, y se obtuvieron los siguientes resultados:



La tabla de categorías para cada grupo queda de la siguiente manera

Grupo	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
alvolante	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0
buenosaires	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4	1	0	0	0	0	0
comercio	0	0	0	2	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
comunidad	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cultura	0	7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
deportes	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0
economia	1	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
elcampo	0	0	0	5	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
elmundo	1	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0
enfoques	3	2	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
espectaculos	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0
inmuebles	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0
moda	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
opinion	3	1	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
politica	1	0	0	0	0	0	0	10	0	2	1	0	0	0	0	3	0	0	0	0	0
revistaln	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
seguridad	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	7	0	0	0	0	0
sociedad	3	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
tecnologia	1	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0
turismo	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
wallstreet	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Donde en verde se indican los Positivos (bien clasificados) y en Rojo los Negativos (mal clasificados).

Positivos: 96

Negativos: 38

Accuracy: **71.6%**

Conclusiones

Utilizando el modelo vectorial para los documentos (artículos de diarios) se pudieron armar grupos que representen los temas tratados en dichos artículos. De cada grupo se pudo extraer las palabras mas representativas, para poder asi deducir el contenido del mismo.

Luego, aprovechando la sección con la que el diario etiqueta cada nota, se implementó un clasificador para poder predecir el tema o la sección a la que va a pertenecer una determinada nota sin observar con un accuracy superior al 70%.

Trabajo Futuro

Como trabajo futuro, se podrían agregar distintos niveles de complejidad para poder lograr una mejor descripción del corpus y clasificación, como por ejemplo:

- Extracción de palabras compuestas (“realidad virtual”, “Cristina Kirchner”, “San Lorenzo”)
- Mejoras en la extracción de Tópicos (etiquetado automático de los grupos, LDA)
- Optimización de la cantidad de tópicos (K) generalizado para cualquier corpus de texto.
- Visualización de los tópicos en 2D o 3D para poder navegar la colección de documentos.

Bibliografía

[1] Understanding Bag-of-Words Model: A Statistical Framework (Yin Zhang · Rong Jin · Zhi-Hua Zhou). *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.453.5924&rep=rep1&type=pdf*

[2] La Nacion. *http://servicios.lanacion.com.ar/herramientas/rss*

[3] tm: A Text Mining Package. *https://cran.r-project.org/web/packages/tm/index.html*

[4] SnowballC: Snowball stemmers based on the C libstemmer UTF-8 library. *https://cran.r-project.org/web/packages/SnowballC/index.html*

[5] Zipf's Law. *https://en.wikipedia.org/wiki/Zipf%27s_law*

[6] Tf-idf weighting. *http://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html*

[7] Rocchio algorithm. *https://en.wikipedia.org/wiki/Rocchio_algorithm*

Anexo I: Script de R Utilizado

```
library("cluster")
library("foreign")
library("tm")
library('proxy') # Library of similarity/dissimilarity measures for 'dist()'
library("lsa")
#library("fpc")
library("pvclust")
library("MASS")
library("lda")
library("graphics")
library("gplots")

rm(list = ls(all=T))
setwd('E:/Maestria Data Mining/Text Mining/')
d = read.csv("noticias acentos mal.txt",header=F,sep=",")
d = as.data.frame(d)
names(d) <- c("link", "categoria", "nota")

# bolsa de palabras
#preposiciones                                                                =
c('a','ante','bajo','cabe','con','contra','de','desde','durante','en','entre',
'hacia','hasta','mediante','para','por','segun','sin','so','sobre','tras','ver
sus','via')
#adv_lugar                                                                    =
c('aqui','alli','alla','aca','arriba','abajo','cerca','lejos','delante','detra
s','encima','debajo','enfrente','atras','alrededor')
#adv_tiempo                                                                   =
c('antes','despues','luego','pronto','tarde','temprano','todavia','aun','ya','
ayer','hoy','manana','siempre','nunca','jamas','proximamente','prontamente','a
noche','enseguida','ahora','mientras','anteriormente')
#adv_modo                                                                     =
c('bien','mal','regular','despacio','deprisa','asi','tal','como','aprisa','adr
ede','peor','mejor','fielmente','estupendamente','facilmente','negativamente',
'responsablemente')
#adv_cantidad                                                                 =
c('muy','poco','mucho','bastante','mas','menos','algo','demasiado','casi','sol
o','solamente','tan','tanto','todo','nada','aproximadamente')
#adv_afnegorden                                                               =
c('si','tambien','cierto','ciertamente','efectivamente','claro','exacto','obvi
o','verdaderamente','seguramente','asimismo','no','jamas','nunca','tampoco','p
rimeramente','ultimamente')
#adv_duda_otros                                                               =
c('quiza','quizas','probablemente','posiblemente','seguramente','tal
vez','puede','puede                          ser','a                        lo
mejor','solo','solamente','aun','inclusive','ademas','unicamente','incluso','m
ismamente','propiamente','precisamente','concretamente','viceversa','contraria
mente','siquiera','consecuentemente')
varios            =            c('foto',            'uno',            'dos',
'tres','cuatro','cinco','seis','primero','primera','primer','mismo','misma')

### Genero un Volatile corpus, ver ?VCorpus
corpus = VCorpus(VectorSource(d[,3]), readerControl = list(language="es"))
corpus <- tm_map(corpus, stripWhitespace) # strip white space
corpus <- tm_map(corpus, removeWords, stopwords("spanish")) # remove Words
#corpus <- tm_map(corpus, removeWords, preposiciones) # remove Words
#corpus <- tm_map(corpus, removeWords, adv_lugar) # remove Words
#corpus <- tm_map(corpus, removeWords, adv_tiempo) # remove Words
#corpus <- tm_map(corpus, removeWords, adv_modo) # remove Words
#corpus <- tm_map(corpus, removeWords, adv_afnegorden) # remove Words
#corpus <- tm_map(corpus, removeWords, adv_cantidad) # remove Words
#corpus <- tm_map(corpus, removeWords, adv_duda_otros) # remove Words
corpus <- tm_map(corpus, removeWords, varios) # remove Words
corpus <- tm_map(corpus, stemDocument) # stemming

tdm <- TermDocumentMatrix(corpus, control = list(weighting = weightTf))

dim(tdm)

#train.sample = sample(1:ncol(tdm),round(ncol(tdm)*0.8))
#write.csv(train.sample,"trainsample.csv")
train.sample=as.matrix(read.csv("trainsample.csv"))[,2]
tdm.train = tdm[,train.sample]
```

```
dim(tdm.train)
tdm.test = tdm[,-train.sample]
dim(tdm.test)

palabras.por.doc = apply(as.matrix(tdm.train), 2, sum)
hist(palabras.por.doc,40,  main="Palabras  por  Documento",  xlab="palabras",
ylab="documentos")
mean(palabras.por.doc)
median(palabras.por.doc)
frec.terminos = apply(as.matrix(tdm.train)>0, 1, sum)
hist(frec.terminos,100,  main="Frecuencia  de  Terminos",  xlab="terminos",
ylab="Frecuencia")
head(sort(frec.terminos,decreasing=T),100)
sum(frec.terminos==1)
sum(frec.terminos)
#tdm.filt              =               tdm[(frec.terminos>3               &
frec.terminos<(ncol(tdm)/3)),(palabras.por.doc>100 & palabras.por.doc<1000)]
tdm.filt = tdm.train[(frec.terminos>2 & frec.terminos<(ncol(tdm.train)/2)),]
tdm.test.filt            =            tdm.test[(frec.terminos>2            &
frec.terminos<(ncol(tdm.train)/2)),]
#tdm.filt = tdm
dim(tdm.filt)
dim(tdm.test.filt)

#hago por definicion una tfIdf como la gente.
docs.por.termino = apply(as.matrix(tdm.filt)>0, 1, sum)
idf = log10(ncol(tdm.filt)/docs.por.termino)

tf = 1+log10(as.matrix(tdm.filt))
tf[tf<0] = 0
tfidf = tf
for(i in 1:nrow(tf)){
  tfidf[i,] = tf[i,]*idf[i]
}
for(i in 1:ncol(tfidf)){
  #ahora normalizo la tfidf a ver que onda...
  tfidf[,i] = tfidf[,i] / (sqrt(sum(tfidf[,i]^2)))
}

tf.test = 1+log10(as.matrix(tdm.test.filt))
tf.test[tf.test<0] = 0
tfidf.test = tf.test
for(i in 1:nrow(tf.test)){
  tfidf.test[i,] = tf.test[i,]*idf[i]
}
for(i in 1:ncol(tfidf.test)){
  #ahora normalizo la tfidf a ver que onda...
  tfidf.test[,i] = tfidf.test[,i] / (sqrt(sum(tfidf.test[,i]^2)))
}

#tdm.lsa = lsa(tfidf, dimcalc_share(share=0.8))
#lsaMatrix <- diag(tdm.lsa$sk) %*% t(tdm.lsa$dk)
#dim(lsaMatrix)
#tdm.lsa.dist = dist(t(as.textmatrix(tdm.lsa)), method="cosine")

write.csv(tfidf[1:10,1:10],"aaa.csv")

tdm.lsa.dist = dist(t(tfidf), method="cosine")
dim(tdm.lsa.dist)

#jerarquico
c2 = hclust(tdm.lsa.dist, method="complete")
cor(tdm.lsa.dist, cophenetic(c2))
plot(c2, main="Dendograma", xlab="Documentos", ylab="altura")
#plot(as.dendrogram(c2),horiz=T)
nclust = 21
rect.hclust(c2, nclust)
idcluster.h = cutree(c2,nclust)
table(idcluster.h)
write.csv(table(idcluster.h), "tablet.csv")
table(d[train.sample,2])
write.csv(table(d[train.sample,2]), "cat.csv")
#t = table(d[(palabras.por.doc>100 & palabras.por.doc<1000),2], idcluster.h)
t = table(d[train.sample,2], idcluster.h)
t
write.csv(t, "table2.csv")
```

```
heatmap(-t)

idcluster = idcluster.h
medias1 = apply(tfidf[,idcluster==1],1,mean)
medias.text  =  paste(names(head(sort(medias1,  decreasing=T),10)),  sep="  ",
collapse=" ")
for(i in 2:nclust)
{
  medias2 = apply(as.matrix(tfidf[,idcluster==i]),1,mean)
  medias1 = rbind(medias1,medias2)
  row.names(medias1)[i] = paste("medias",i, sep="")
  medias.text       =       c(medias.text,       paste(names(head(sort(medias2,
decreasing=T),10)), sep=" ", collapse=" "))
}
write.csv(medias.text, "medias.csv")


names(head(sort(medias1[3,], decreasing=T),20))
#d[train.sample[idcluster.h==10],1]

#ahora concateno las 21 medias con los vectores de test. calculo la distancia
de eso y me quedo con
# el cuadrado que me sirve.
#normalizo el vector medias1
for(i in 1:nrow(medias1)){
  #ahora normalizo la tfidf a ver que onda...
  medias1[i,] = medias1[i,] / (sqrt(sum(medias1[i,]^2)))
}

#concateno el vector de medias con los datos a clasificar
data.test.agg = rbind(medias1,t(tfidf.test))
dim(data.test.agg)
#calculo la distancia de todos contra todos
dist.test = dist(data.test.agg, method="cosine")
#me quedo solo con la parte de la matriz que me sirve (la que compara los datos
nuevos con las medias)
dist.compare                                                                =
as.matrix(dist.test)[1:nrow(medias1),(nrow(medias1)+1):nrow(data.test.agg)]
dim(dist.compare)

#ahora, busco por cada dato nuevo cual es el centroide del cluster mas cercano.
idcluster.test = rep(0,ncol(dist.compare))
for(i in 1:ncol(dist.compare)){
  data.min = min(dist.compare[,i])
  idcluster.test[i] = which(dist.compare[,i]==data.min)
}
t.test = table(d[-train.sample,2], idcluster.test)
t.test

write.csv(t.test, "tabletest.csv")

heatmap(-t.test)
```