Análisis de sentimiento en Twitter: El bueno, el malo y el >:(

Trabajo Final por

Carlos Martín Becerra

Trabajo final realizado para la Facultad de Matemática, Astronomía, Física y Computación (Fa.M.A.F), de la Universidad Nacional de Córdoba, Argentina, en cumplimiento parcial para la obtención del grado de Licenciado en Ciencias de la Computación.

Trabajo dirigido por

Laura Alonso Alemany, PhD.

16 de Junio de 2016

Resumen

Los sistemas de análisis de sentimientos y la minería de opiniones han resultado ser de gran utilidad en los últimos años, con la introducción de las redes sociales. Su principal objetivo es identificar opiniones positivas o negativas en textos generados por usuarios y sobre qué entidad o aspecto de la misma se han realizado.

Uno de los problemas que presenta al analizar grandes volúmenes de opiniones generadas por usuarios es el de que un analista pueda procesarlas de forma rápida y efectiva.

Utilizando principalmente la red social Twitter, nos proponemos estudiar, a partir de algunos acontecimientos que produjeron tendencias, la opinión de los usuarios sobre los mismos.

Keywords: *Sentiment Analysis, Opinion Mining, K-Means, Clustering, Visualization, Information Extraction, Twitter*

De acuerdo con 2012 ACM Computing Classification System:

- **Information systems - Sentiment analysis**
- *Human-centered computing - Information visualization*
- *Theory of computation - Unsupervised learning and clustering*

Agradecimientos

En primer lugar, agradecer a mi directora Laura por tomarse el tiempo para evacuar mis dudas, en especial por sus consejos y por mantenerme motivado desde el comienzo. Sin su supervisión este trabajo nunca hubiera sido posible.

A mis amigos por el apoyo y las palabras de aliento en esos momento de mayor incertidumbre, sobretodo por ser un cable a tierra en los momentos que más lo necesitaba.

No puedo terminar sin agradecer a mi familia, en especial a mis padres Laura y Carlos por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica, como de la vida, por su incondicional apoyo perfectamente mantenido a través del tiempo. También a mi hermano Franco y a mi pareja Magalí, sin su constante apoyo no estaría aqui.

Índice general

Índice de figuras

Capítulo 1

Introducción y motivación

1.1. Descripción del problema

Desde hace ya más de una década, con el surgimiento de la internet 2.0, los usuarios tienen la posibilidad de generar su propio contenido y compartirlo públicamente con mayor facilidad. En este auge, las redes sociales han cobrado gran popularidad, en particular la plataforma de microblogging *Twitter* la cual permite a sus usuarios compartir mensajes de texto en 140 caracteres con sus familiares, amigos y seguidores. Diariamente se publican más de 500 millones de mensajes, comunmente llamados *tweets*. Debido a que la principal razón de estas publicaciones es expresar el punto de vista y la opinión de los usuarios, resultan ser de gran interés para ser analizados. En la figura 1.1 podemos ver una captura de pantalla del *news feed* de Twitter, que nos muestra las últimas públicaciones realizadas por otros usuarios a los cuales seguimos.

Motivación para la extración de información

Para poder explotar todos estos datos que circula públicamente en la web podemos realizar diversas tareas que nos permitan extraer información útil de las opiniones. Esta área de trabajo se conoce como *minería de opiniones*, o *análisis de sentimientos*, y se enfoca en el tratamiento automático de textos en los cuales

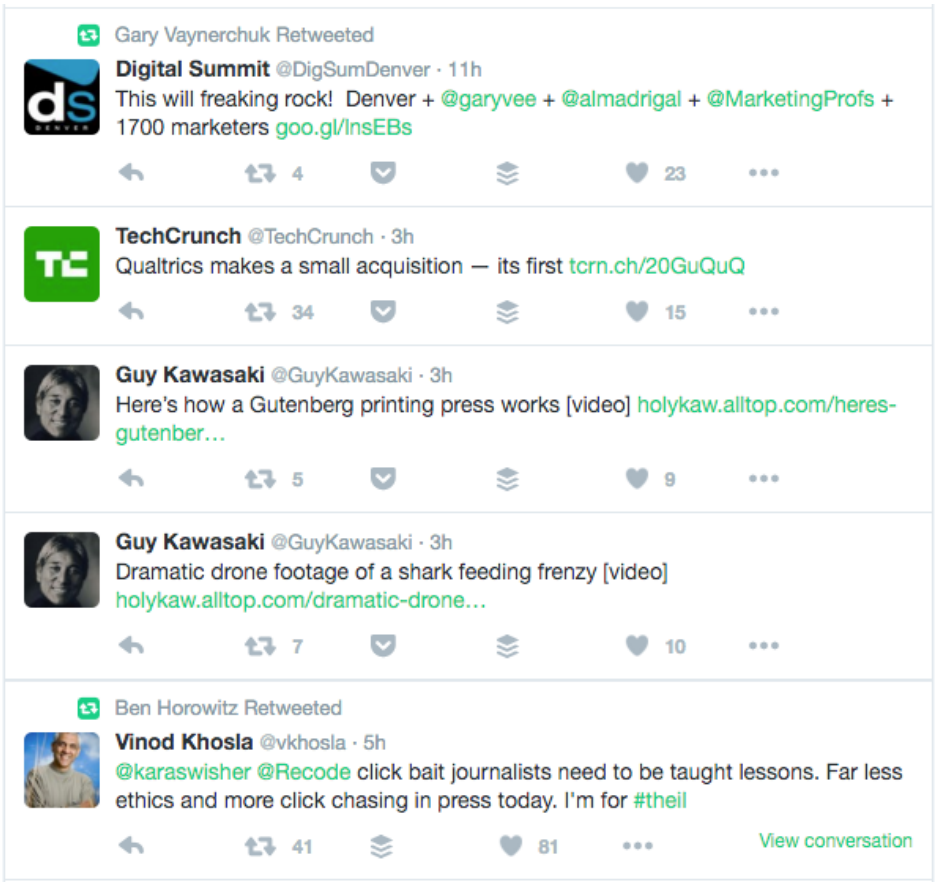


Figura 1.1: Captura de pantalla del *news feed* de Twitter, que nos muestra las últimas públicaciones realizadas por otros usuarios a los cuales seguimos.

se ven reflejados la opinión, los sentimientos, las emociones y las actitudes de las personas hacia ciertos temas y sus aspectos.

Conocer lo que sus usuarios opinan tiene diversas aplicaciones como recomendar productos y servicios o determinar a qué candidato político se votará en las próximas elecciones. Por ejemplo, una oración como "Estoy molesto por el aumento a la tarifa de los colectivos" podría ser de gran utilidad para medir la opinión pública ante la medida tomada.

Extraer sentimientos: para qué sirve, qué limitaciones tiene

Al hacer este tipo de análisis nos encontramos con características del lenguaje natural que hacen que esta sea un área de investigación llamativa. Nos encon-

tramos con problemas de ambigüedad o vaguedad semántica como por ejemplo al momento de detectar el sarcasmo en una oración como "Pero qué buen auto! Dejó de funcionar en dos días!"[6]

También, hay algoritmos que utilizan un conjunto de palabras del lenguaje, llamado lexicón [21], clasificadas como *palabras positivas* (bueno, genial, excelente) o *palabras negativas* (feo, desagradable, horrible) para poder clasificar el sentimiento que expresa una oración. Sin embargo, hay oraciones que pueden contener estas palabras pero sin expresar ningún sentimiento en particular, como por ejemplo "Podrían decirme qué cámara Sony es buena para filmar?".

Otro caso es al momento de clasificar *oraciones objetivas*, como por ejemplo "La batería de mi teléfono dura 2 horas" que no contiene palabras positivas o negativas. Se necesita sentido común para poder reconocer si lo que expresa la oración es algo positivo o negativo. Esto hace que sea posible extraer opiniones de oraciones que solo expresen hechos.[22].

A veces las mismas expresiones pueden tener diferentes sentimientos asociados en *contextos* diferentes. Por ejemplo del comentario: "No emite sonido alguno" podría considerarse algo positivo cuando se habla de un auto nuevo, pero también podría indicar algo negativo si se hablase de un equipo de música. Esto hace que sea necesario tener *conocimiento sobre el dominio* en el cual estamos trabajando. [2, 5]

Un objetivo especialmente interesante desde el punto de vista de extracción de información y representación del conocimiento es clasificar positiva o negativamente, según la opinión de distintos usuarios, los diferentes aspectos de una *entidad* y conocer qué motiva tales opiniones. Poder generar una solución a este problema tan complejo es algo que requiere gran conocimiento sobre el dominio y que este conocimiento sea trasladado a la solución de software propuesta, con un gran esfuerzo de desarrollo.

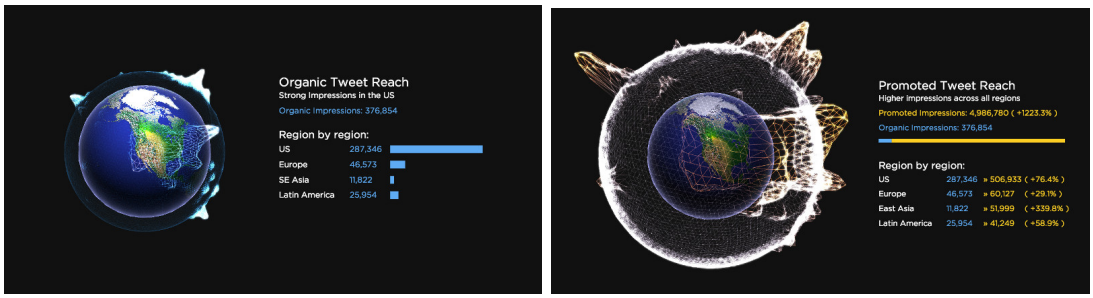
Clustering por temas, cómo se complementa con detección de sentimientos

Una estrategia menos costosa para aproximarse al análisis de sentimientos de aspectos consiste en agrupar en *clusters* opiniones similares que hablen de un mismo aspecto. Esto nos permite de forma rápida, al momento de hacer el análisis de sentimiento, poder obtener una estimación de cómo es la opinión que se tiene acerca de ese aspecto. Claramente, esta solución no será tan adecuada comparada con una solución donde se hayan inyectado grandes cantidades de conocimiento sobre el dominio, pero es una alternativa para hacer una primera aproximación exploratoria sobre los datos.

Visualización para la ayuda a la toma de decisiones

Debido a los grandes volúmenes de datos que se manejan, encontrar y monitorear la opinión de muchos usuarios es una tarea difícil ya que identificar información relevante y extraerla de forma resumida es un procedimiento costoso para ser realizado manualmente. Por eso es que se deben utilizar sistemas de análisis de sentimientos automáticos que nos permiten sintetizar la información de forma automática. Una forma común es realizar un análisis exploratorio de los datos mediante visualizaciones. En la figura 1.2 podemos ver una visualización del potencial alcance global de un *tweet* promocionado y en 1.3 podemos ver un ejemplo de una visualización del *The New York Times* prediciendo los resultados de las elecciones presidenciales de Estados Unidos del 2012.

Las *visualizaciones* nos permiten resumir grandes volúmenes de datos en representaciones gráficas. Posteriormente, un experto puede interpretarlos rápidamente y hacer mejores conclusiones. Luego, puede tomar una decisión basada en la información recolectada.



(a) Alcance organico de un *tweet* (b) Alcance de un *tweet* promocionado

Figura 1.2: Visualización del potencial alcance global de un *tweet* promocionado

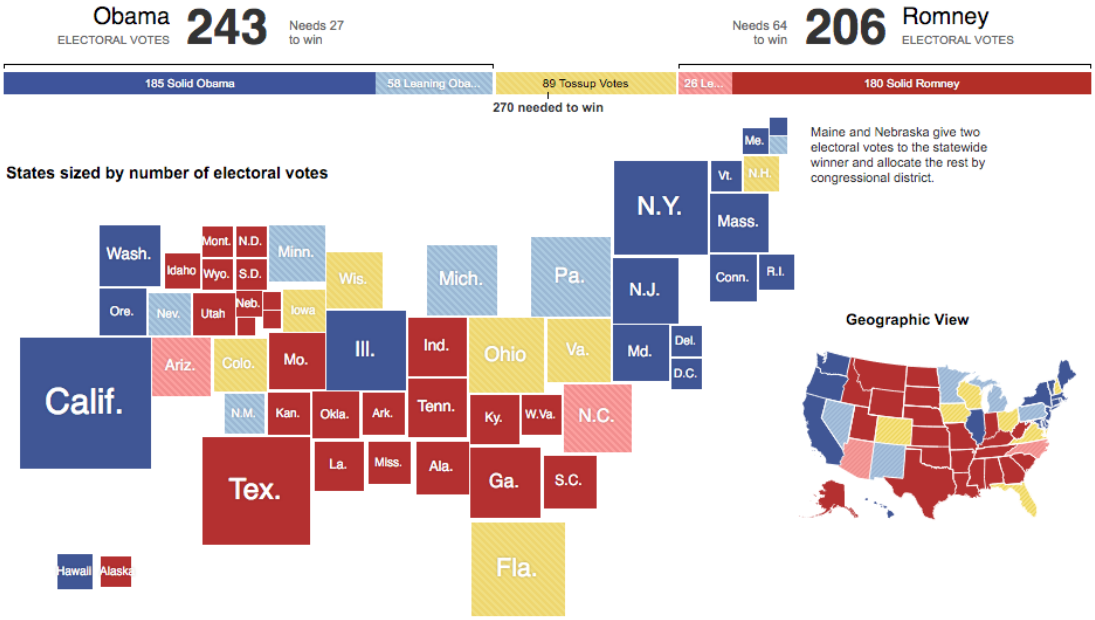


Figura 1.3: Visualización de *The New York Times* prediciendo los resultados de las elecciones presidenciales de Estados Unidos del 2012

1.2. Arquitectura del proyecto

En el contexto que enmarcan estos problemas, proponemos una solución compuesta para estudiar las opiniones que compartieron distintos usuarios de Twitter.

En la figura 1.4 podemos ver de forma gráfica cada etapa del *pipeline* que diseñamos para este trabajo. Trabajamos a partir de algunos acontecimientos recientes que produjeron tendencias.

Utilizando la API de Twitter, logramos recolectar al rededor de 122 mil tweets

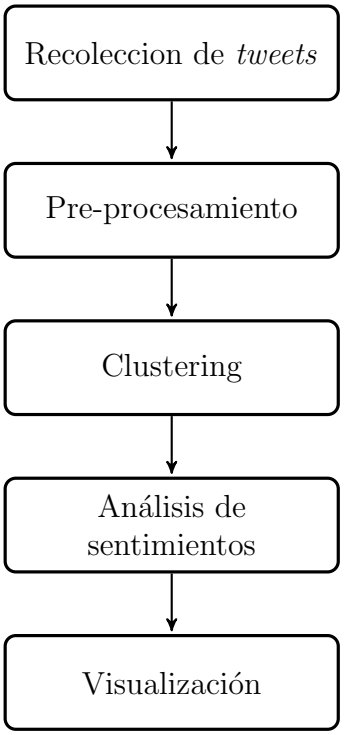


Figura 1.4: Representación visual del *pipeline* diseñado

de la última edición de los premios Oscars bajo la etiqueta *#oscars*. Con estos datos realizamos diversos experimentos utilizando el algoritmo *K-Means* para poder agrupar y separar en subtemas (o aspectos) las opiniones de los usuarios con respecto a estos temas.

Para poder trabajar con los tweets como objetos en un espacio matemático primero tuvimos que pre-procesarlos, como se detalla en el capítulo 2.

Algo que también consideramos, que no se ve reflejado en el gráfico del pipeline, fue el caso en el que nuevos mensajes podrían ingresar al sistema. Decidimos entrenar un clasificador utilizando como clases las etiquetas de los clusters, esto nos permitió poder decidir a qué *cluster* debería pertenecer el nuevo mensaje.

Luego de generados los *clusters*, hicimos un *análisis de sentimientos* utilizando un clasificador. Con este clasificador etiquetamos el sentimiento expresado por cada tweet y también de poder obtener una apreciación general por el sentimiento expresado hacia el cluster al cual pertenece.

Finalmente, realizamos experimentos con *visualizaciones* que nos permitan

navegar la información recolectada y los resultados de las etapas anteriores de una manera rápida y efectiva.

Todo el código utilizado para este trabajo está disponible en el siguiente repositorio online: https://github.com/fuxes/twitter-sentiment-clustering

1.3. Esquema de la tesis

En este escrito se encuentran redactados todos los contenidos necesarios para poder replicar los experimentos realizados. El trabajo se organiza de la siguiente manera.

El **capítulo 2** describe cómo se recolectaron los datos utilizados, cómo fueron estructurados y pre-procesados. También mencionamos cuales fueron algunas limitaciones con las cuales nos encontramos.

En el **capítulo 3** se define el *análisis de sentimientos* y se describen algunos problemas asociados al mismo. Mencionamos los diferentes tipos de clasificación que se pueden realizar y cuales son las ventajas y desventajas de la aproximación que elegimos para este trabajo.

En el **capítulo 4** se enfoca en *algoritmos de clustering* y como se utilizaron para hacer un análisis exploratorio de los datos para mejorar la granularidad del sistema de análisis de sentimientos. Verémos cómo fueron representados los *tweets* y los diversos problemas que se presentaron, para luego ver cuáles fueron los resultados del proceso de clustering sobre los datos.

El **capítulo 5** introduce el tema de la *visualización* de datos, en el cual analizamos diferentes alternativas para visualizar la información producida durante la etapa de experimentación.

Finalmente, el capítulo 6 concluye el trabajo y discute diferentes posibilidades de trabajo a futuro.

Capítulo 2

Datos

2.1. Recolección de datos

Para este trabajo utilizamos la red social *Twitter* que permite enviar mensajes cortos de hasta 140 caracteres comúnmente llamados *tweets*. Al día de la fecha, Twitter cuenta con más de 332 millones de usuarios que generan más de 500 millones de tweets al día compartiendo sus puntos de vista y opiniones con sus familiares, conocidos y seguidores. Es uno de los 10 sitios web más visitados del mundo y fue descrito como "el *SMS* de la internet".

Los usuarios pueden agrupar *tweets* por tema o por tipo utilizando *hashtags* - palabras o frases con el prefijo *"#"*. El símbolo *"@"* acompañado del nombre de un usuario es utilizado para mencionar o responder a dicho usuario.

Creando una aplicación en Twitter[1] y mediante la implementación de un programa que recolector de *tweets* en inglés juntamos aproximadamente 120 mil tweets. Todos estos *tweets* pertenecen a la edición última edición premios Óscars, que se llevó acabo el 28 de Febrero. Comparten la etiqueta **#oscars**. Se utilizó la librería Tweepy[2], la cual permite utilizar la Search API (interfaz de programación

[1] https://apps.twitter.com/
[2] http://www.tweepy.org/

de aplicaciones) de Twitter.

Limitaciones de Twitter

Al momento de trabajar con la API de Twitter fue necesario tener en cuenta ciertas *limitaciones* de la misma. Los límites de frecuencia de la interfaz sólo permitían realizar hasta 450 peticiones cada 15 minutos[17]. Por otro lado, hay que tener en cuenta que Twitter filtra gran cantidad de los tweets publicados para que los resultados de nuestras búsquedas sean de mayor calidad, por lo que no está disponible la totalidad de publicaciones que se realizaron sino aquellas que Twitter considera que son más relevantes para el usuario[19].

Estadísticas del dataset

Dataset: "#oscars"

- Idioma: Inglés
- Tweets: 122.443
- Tweets únicos: 38.055
- Palabras en tweets únicos: 406.311
- Palabras únicas: 72.792
- Palabras que se repiten en más de 100 tweets: 452
- Palabras que se repiten en más de 1000 tweets: 26

2.2. Preprocesamiento de los datos

El dataset resultante de la recolección de datos es una colección de tweets. Para poder aplicar técnicas de clustering sobre estos datos es necesario que pasen por un proceso comprendido por varias fases, también conocido como *pipeline*, en

donde la entrada de cada fase es la salida de la anterior. El resultado final será una matriz que representará la información relevante de los datos.

A continuación explicaremos el pipeline diseñado para este trabajo utilizando un tweet a modo de ejemplo para poder ver las transformaciones.

Mapeo de tweets a texto

Un tweet es un objeto complejo con muchas propiedades pero lo que nos interesa es el mensaje que el usuario escribió en forma de téxto[20]. Para ello, en primer lugar la colección de tweets es transformada en nuestro corpus extrayendo el mensaje de cada tweet. Si bien al realizar este paso perdemos la información sobre el autor del tweet y la fecha de la publicación, conservamos la referencia al objeto para poder vincular el resultado final con el tweet original. El resultado de esta etapa es un listado de oraciones, o de documentos.

```
['Spotlight, Óscar a Mejor Guion Original http://ow.ly/YUmRu
#Oscars', '#DaveGrohl también paso por los #Oscars tocando
#Blackbird de los #Beatles Mira: http://ow.ly/YUmHr',
'#LeonardoDiCaprio #Oscars te lo merecías! Eres de mis actores
favoritos!']
```

Figura 2.1: Ejemplo de 3 *tweets* sobre los *#oscars*

Tokenización

En segundo lugar cada documento resultante de la etapa anterior es transformado en una lista de palabras y símbolos llamados tokens.En general los tokens son cadenas de caracteres entre espacios en blanco o puntuación, pero no siempre es así, como por ejemplo en el caso de las abreviaturas. El conjunto total de palabras utilizadas, distintas y únicas, es el vocabulario del corpus.

En este paso también se filtran las palabras funcionales, que no tienen una

semántica referencial clara, como artículos, pronombres, preposiciones, etc. que son muy frecuentes en el lenguaje, llamadas *stop words*. También es necesario eliminar ciertas palabras funcionales propias del glosario de Twitter como *"RT"*, *"HO"* y *"HT"*.[18].

```
[[‘spotlight’, ‘oscar’, ‘mejor’, ‘guion’, ‘original’,
‘http://ow.ly/YUmRu’, ‘#oscars’], [‘#davegrohl’, ‘paso’,
‘#oscars’, ‘tocando’, ‘#blackbird’, ‘#beatles’, ‘mira’,
‘http://ow.ly/YUmHr’], [‘#leonardodicaprio’, ‘#oscars’,
‘merecias’, ‘mis’, ‘actores’, ‘favoritos’]]
```

Figura 2.2: Ejemplo de 3 *tweets* que fueron tokenizados

El proceso de tokenización es importante ya que no es una tarea trivial en el lenguaje estándar y menos lo es al momento de trabajar con tweets. Tenemos palabras especiales como las menciones (`"@"` + nombre de usuario), los hashtags (`"#"` + tema) y las direcciones (`‘http://...’`) que queremos que sean considerados como un único token.

Stemming

Muchas veces diferentes tokens pueden hacer referencia al mismo concepto ya que este puede ser representado por variantes morfológicas de una misma familia de palabras. Por ejemplo podemos representar "escribo", "escribíamos" y "escribimos", ya que tienen un significado similar y derivan del mismo verbo, en su raíz como "escrib". Esto nos permite relacionar aquellos tweets que contienen alguna de estas palabras mediante su raíz, lo que luego facilitará el proceso de clustering.
A este proceso de reducción de tweets lo identificaremos como ***stem*** o ***stemming***.

Hay que tener en cuenta que pueden surgir errores a partir de este proceso, que dependen del algoritmo que utilicemos y nuestro corpus.
Un tipo de error es el caso en que dos palabras con distintos significados sean reducidas a una misma palabra cuando no debería ser así.

casa, casorio, caso → cas

Otro tipo de error se da cuando dos palabras deberían ser reducidas a una misma forma normal pero se considera que provienen de diferentes palabras.

alumnus → alumnu,
alumni → alumni,
alumna/alumnae → alumna.

La solución a este problema es un proceso de reducción a forma canónica que incorpore más información lingüística, proceso conocido como *lematización*. En este trabajo optamos por no implementar esta opción porque resultaba costoso instalar la herramienta que realizaba esta tarea y porque los tweets tienen muchas palabras no estándar que tampoco pueden tratar bien los lematizadores.

Vectorización

El próximo paso consiste en generar una matriz rala (*sparse*) transformando cada lista de palabras en un vector en un espacio Euclídeo, donde cada columna es una característica. Las características son principalmente palabras del vocabulario extraído de la tokenización. Podemos considerar cada palabra del vocabulario en una columna, o podemos obtener representaciones más detalladas si consideramos como posibles características secuencias de palabras, llamadas ***n-gramas***, que han ocurrido en el texto. Los n-gramas pueden ser secuencias de una palabra (unigramas), de dos palabras (bigramas), de tres palabras (trigramas), y así sucesivamente.

```
[‘#beatles’, ‘#beatles mira’, ‘#blackbird’, ‘#blackbird
#beatles’, ‘#davegrohl’, ‘#davegrohl paso’, ‘#leonardodicaprio’,
‘#leonardodicaprio #oscars’, ‘#oscars’, ..., ‘paso’, ‘paso
#oscars’, ‘spotlight’, ‘spotlight oscar’, ‘tocando’, ‘tocando
#blackbird’]
```

Figura 2.3: Ejemplo de la tokenización de 3 *tweets* utilizando unigramas y bigramas

Cada tweet se representa en una fila, y los valores que toma en cada columna están determinados por la ocurrencia de las palabras de cada columna en el tweet. De esa forma, la celda *i,j* de la matriz tendrá el valor de la ocurrencia del n-grama representado en la columna i en el tweet j.

La forma más simple de asignar valores a las celdas es con valores binarios: 1 si el n-grama representado por la columna i ocurre en el tweet j, y 0 si no ocurre. Es de esperar que la mayor parte de n-gramas no ocurra en un tweet, ya que el vocabulario es muy grande, por lo tanto las matrices serán ralas, con gran cantidad de 0. A esta forma de representar los tweets la identificaremos como ***bin*** o ***binarización***.

```
[[0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0],
[1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1],
[0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0]]
```

Figura 2.4: Los 3 *tweets* de ejemplo luego de la etapa de binarización

Donde el vocabulario asociado a las columnas es:

`['#beatles', '#blackbird', '#davegrohl', '#leonardodicaprio', '#oscars', 'actores', 'favoritos', 'guion', 'http://ow.ly/YUmHr', 'http://ow.ly/YUmRu', 'mejor', 'merecias', 'mira', 'mis', 'original', 'oscar', 'paso', 'spotlight', 'tocando']`

Figura 2.5: Vocabulario asociado a los 3 *tweets* de ejemplo luego de la etapa de binarización

Al momento de vectorizar los documentos podemos elegir entre diferentes opciones que nos permiten reducir el volumen del corpus y que los resultados sean más relevantes.

Una palabra que aparece en más de la mitad de los tweets es redundante y no brinda mucha información para diferenciar un tweet de otro. Para esto es útil definir un umbral de frecuencia para determinar qué características deben ser incluidas en la matriz, de tal forma que aquellas características que son muy frecuentes sean ignoradas, ya que pueden ser consideradas *stop words* propias del corpus. Tampoco debemos incluir aquellas características que son poco frecuentes ya que no proveen generalizaciones sobre tendencias en los textos. Identificaremos con ***min df*** y ***max df*** a las cotas inferior y superior, respectivamente, del umbral de frecuencia, describiendo así las palabras que descartamos porque ocurren menos veces que ***min df*** o más veces que ***max df***.

Frecuencia

La representación de la co-ocurrencia de tweets y palabras con valores binarios es demasiado gruesa y no captura algunas diferencias importantes, como por ejemplo si una palabra ocurre muchas veces en un tweet. Para eso, podemos asignar valores a las celdas con valores de frecuencia: n si el n-grama representado por la columna i ocurre n veces en el tweet j, y 0 si no ocurre. A esta forma de representar los tweets la identificaremos como ***tf*** o ***frecuencia de términos***.

`‘Esos que dicen México ganó dos #Oscars ayer, no, México no ganó señores #Inarritu y #Lubezki si’`

Sería representado como:
`[1, 1, 1, 1, 1, 1, 2, 2, 1]`

Con el siguiente vocabulario:
`[‘#inarritu’, ‘#lubezki’, ‘#oscars’, ‘ayer’, ‘dicen’, ‘esos’, ‘gano’, ‘mexico’, ‘señores’]`

Figura 2.6: Ejemplo un *tweet* luego de la etapa de frecuencia de términos

Sumado a esto, para producir resultados aún relevantes, podemos limitar la máxima cantidad de características considerando solo aquellas n primeras ordenadas de mayor a menor según su frecuencia en todo el corpus. De tal manera, las características resultantes serán aquellas que tienen más importancia al momento de representar un tweet. A esta forma de reducir el volumen de tweets la identificaremos como ***max features*** o ***límite de características***.

Frecuencia de términos - frecuencia inversa de documento.

La representación de la co-ocurrencia de tweets y palabras con valores de frecuencia nos permite conocer qué tan importante es una característica para identificar un tweet según cuántas veces esta ocurre en el mismo.

Sin embargo, aquellas características que son muy frecuentes en el conjunto de todos los tweets hace que no sean tan relevantes al momento de identificarlos. Por lo tanto, se incorpora un factor de frecuencia inversa de documento que atenúa el peso de las características que ocurren con mucha frecuencia en la colección de tweets e incrementa el peso de los características que ocurren pocas veces.

```
[[‘spotlight’, ‘oscar’, ‘mejor’, ‘guion’, ‘original’,
‘http://ow.ly/YUmRu’, ‘#oscars’], [‘#davegrohl’, ‘paso’,
‘#oscars’, ‘tocando’, ‘#blackbird’, ‘#beatles’, ‘mira’,
‘http://ow.ly/YUmHr’], [‘#leonardodicaprio’, ‘#oscars’,
‘merecias’, ‘mis’, ‘actores’, ‘favoritos’]]
```

Se representa como:

```
[[0, 0, 0, 0, 0.23, 0, 0, 0.39, 0, 0.39, 0.39, 0, 0, 0, 0.39,
0.39, 0, 0.39, 0.], [0.36, 0.36, 0.36, 0, 0.21, 0, 0, 0, 0.36, 0,
0, 0, 0.36, 0, 0, 0, 0.36, 0, 0.36], [0, 0, 0, 0.43, 0.25, 0.43,
0.43, 0, 0, 0, 0, 0.43, 0, 0.43, 0, 0, 0, 0, 0]]
```

Figura 2.7: Los 3 *tweets* de ejemplo luego del proceso de *tf-idf*

A esta forma de representar los tweets la identificaremos como ***tf*idf*** o ***frecuencia de términos - frecuencia inversa de documento***.

Capítulo 3

Análisis de Sentimiento

El análisis de sentimiento es un campo de estudio que busca extraer opiniones y sentimientos, sobre una entidad y sus aspectos, desde el lenguaje natural de los textos, automáticamente, utilizando algoritmos. Como muestra la figura 3.1, estas opiniones pueden expresar o implicar un sentimiento positivo, negativo o neutro.

Figura 3.1: Ejemplo de *tweet* transmitiendo un sentimiento positivo.

Su importancia está en que nuestra percepción de la realidad, y así también las decisiones que tomamos, es condicionada en cierta forma por cómo otras personas ven y perciben el mundo. Es por esto que desde un punto de vista de utilidad, queremos conocer la opinión de otras personas sobre cualquier tema de interés ya que tienen diversas aplicaciones como recomendar productos y servicios,

determinar a qué candidato político se votará en las próximas elecciones o incluso medir la opinión pública ante la medida tomada por una empresa o el gobierno[11]. Sin embargo, existen varias limitaciones que se pueden ver en la sección 1.1

3.1. Granularidad de análisis

A la hora extraer esta información hay una gran variedad de métodos y algoritmos dependiendo del nivel de granularidad del análisis que queramos llevar a cabo. Se distinguen tres niveles: nivel de documento, de oración o de aspecto. El análisis a nivel de documento determina el sentimiento general expresado en un texto, mientras que el análisis a nivel de frase lo especifica para cada una de las oraciones del texto.

Sin embargo, estos dos tipos de análisis no profundizan en detalle sobre qué es lo que a las personas les gusta o no. No especifican sobre qué es la opinión, ya que considerando la opinión general de un objeto como positiva (o negativa) no significa que el autor tenga una opinión positiva (o negativa) de todos los aspectos de dicho objeto.

Para este trabajo nos enfocamos en realizar un análisis a nivel de documento ya que debido al límite en los mensajes, los autores suelen ir directo al grano sin tener la posibilidad de incluir varios aspectos diferenciados en un solo tweet. Por esta razón, usar el tweet como unidad de análisis parece proveer un nivel de granularidad adecuado para hacer análisis de sentimiento desglosado.

Análisis a nivel de documento

Considerada como una de las tareas más simples de la minería de opiniones, el análisis a nivel de documento apunta a clasificar la opinión de un documento (en este caso un tweet) como positiva o negativa. Esta tarea no considera los detalles en cuanto a entidades o aspectos, sino que considera el documento como un todo.

Puede ser considerado como una tarea tradicional de clasificación de texto donde las clases son las diferentes orientaciones en cuanto a los sentimientos. No obstante, para asegurar que este tipo de análisis tenga sentido es necesario asumir que cada documento expresa una única opinión sobre una única entidad. Si bien esto puede parecer una limitación, porque en un tweet uno podría expresar más de una opinión hacía distintas entidades, en la práctica funciona bien ya que los usuarios suelen enfocarse en un único aspecto en cada tweet. Seguramente en otros contextos, o si no estuviera esta limitación en cuanto el largo de los mensajes, sería una buena idea considerar sistemas de análisis más complejos que permitan realizar un análisis más granular.

Considerando esto y tomando como clases *"positivo"* y *"negativo"*, cualquier método de clasificación de texto basado en aprendizaje automático se puede aplicar directamente; no incluir la clase *"neutro"* ayuda a simplificar el problema. Se suelen utilizar reseñas de productos asociadas manualmente a su valoración como datos anotados manualmente para entrenar y evaluar la eficacia de nuestro clasificador. Existen diferentes *datasets* abiertos para esto.

Por otro lado, como las palabras que conforman las opiniones son el factor determinante en el análisis de sentimientos también es una buena opción utilizar métodos de aprendizaje basados en el uso de lexicones. Estos son diccionarios que contienen listados de palabras etiquetadas con el sentimiento asociado correspondiente, en algunos casos por un valor que también indica la intensidad del mismo. Luego, dada una opinión, simplemente podemos buscar el valor de todas las palabras que la componen y sumar el valor asociado para finalmente clasificar la opinión como positiva o negativa según corresponda [7, 10]. También se podría considerar extender el sistema para contemplar negaciones y palabras que intensifican el sentimiento [9].

Clustering como forma de acercarse a los aspectos

Si bien la identificación y extracción de aspectos sobre las cuales hablan las opiniones puede considerarse una tarea compleja, podemos obtener un acercamiento en el caso de los tweets utilizando clustering. En este contexto, obtenemos un conjunto de tweets que tratan de un mismo tema (en este caso, los #oscars), y tratamos de identificar en este conjunto los diferentes aspectos sobre este tema.

Al trabajar con un dataset caracterizado por un tema en particular, en este caso los #oscars, el clustering nos permite dividir el corpus en subtemas. De esta manera podemos hacer un análisis exploratorio de los datos como si cada uno de estos subtemas fuese un aspecto propio de la entidad #oscars. Luego, podemos clasificar cada aspecto como positivo o negativo según cual es la apreciación general que tiene cada cluster y consecuentemente evaluar el total del corpus.

Una aproximación por clustering no será tan exacta como una en la que hayamos inyectado conocimiento humano, ya sea mediante reglas o ejemplos anotados, pero permite aproximarnos de forma rápida y poco costosa a un análisis exploratorio de un gran conjunto de datos.

Profundizamos sobre este tema en el Capítulo 4.

3.2. Selección de un clasificador de sentimientos

El rol de un clasificador de sentimientos es asignar a un tweet un sentimiento positivo o negativo de forma automática. En nuestra prueba de concepto, queremos asignar sentimiento positivo o negativo a los tweets del corpus de #oscars. Para ello implementamos un clasificador de sentimientos basado en aprendizaje automático, y evaluamos diferentes alternativas.

3.2.1. Metodología de evaluación

Para poder saber qué tan efectivas son las predicciones de nuestro clasificador evaluamos su rendimiento mediante una validación cruzada de K iteraciones.

Cuando nos referimos al rendimiento de un clasificador estamos haciendo referencia al *accuracy* del mismo, la proporción de los casos que fueron clasificados correctamente sobre la totalidad de todos los casos a clasificar. La clasificación correcta de los casos es conocida porque los casos de evaluación, así como los de entrenamiento, han sido anotados manualmente por expertos humanos.

$$Accuracy = \frac{\text{Total de aciertos}}{\text{Total de casos}}$$

En la validación cruzada de K iteraciones o *K-fold cross-validation* los datos anotados se dividen en K subconjuntos. Uno de los subconjuntos se utiliza como datos de prueba y el resto (*K-1*) como datos de entrenamiento. El proceso de validación cruzada es repetido durante K iteraciones, con cada uno de los posibles subconjuntos de datos de prueba. Finalmente se realiza la media aritmética de los resultados de cada iteración para obtener un único resultado. Este método es muy preciso puesto que evaluamos a partir de K combinaciones de datos de entrenamiento y de prueba, pero aun así tiene una desventaja, y es que es lento desde el punto de vista computacional. En la práctica, la elección del número de iteraciones depende de la medida del conjunto de datos. Lo más común es utilizar la validación cruzada de 10 iteraciones (*10-fold cross-validation*). [16]

3.3. Análisis de resultados de clasificación de sentimientos

Para este trabajo trabajamos con el dataset utilizado en Pang y Lee (2004) [14]. Está compuesto por 2000 reseñas de películas clasificadas con puntajes, la mitad son positivos y la otra mitad son negativos, con un límite de 20 reseñas por autor por clase con un total de 312 autores.

Pasamos las opiniones por un preproceso simple de *tf*idf*. Con este conjunto

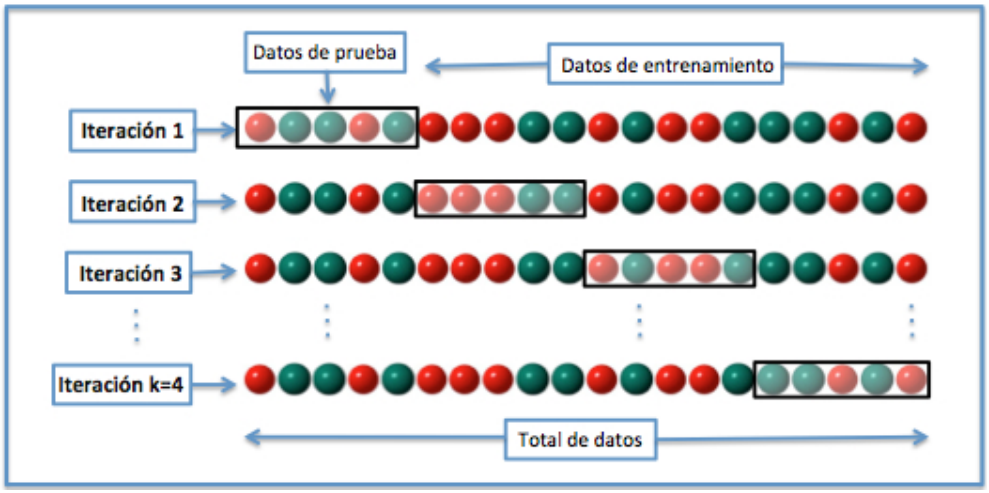


Figura 3.2: Ejemplo de validación cruzada de K iteraciones con *K=4*.

de datos vectorizados entrenamos diferentes clasificadores provistos por la librería open source de aprendizaje automático *Scikit-Learn* de Python[15], y comparamos su efectividad. Debido a que los clasificadores tienden a tener diferentes parámetros para poder configurarlos, en vez de probar manualmente las alternativas que tenemos para cada parámetro, hicimos una búsqueda exhaustiva probando todas las combinaciones posibles utilizando la herramienta *Grid Search* de Scikit-Learn.

Comparamos tres tipos de clasificadores, una máquina de vectores de soporte (*Support Vector Machine, SVM*) que es un clasificador de tipo lineal, un árbol de decisión y un clasificador bayesiano naïve con distribución multinomial. Todos estos clasificadores son provistos por la librería Scikit-Learn. A continuación detallamos los parámetros que obtuvieron un mejor rendimiento.

Para la *SVM* obtuvimos el mayor rendimiento, de 87.15 %, utilizando una penalidad C de 100 y utilizando la función de pérdida *"hinge"*. Sólo obtuvimos un rendimiento del 66.85 % utilizando árboles de decisión con el criterio *"entropy"* y con un máximo de 6 niveles de profundidad, y finalmente obtuvimos un rendimiento de 84.60 % con el clasificador bayesiano naïve con distribución multinomial y con un factor *alpha* de 0.1. Por lo tanto, en nuestra arquitectura del

SVM	87.15 %
DT	66.85 %
MNB	84.60 %

Cuadro 3.1: Tabla de resultados con los diferentes tipos de clasificadores

sistema decidimos quedarnos con el clasificador SVM para realizar la asignación de sentimiento a los tweets de nuestro corpus.

Capítulo 4

Clustering

Al momento de analizar los tweets que recolectamos, nos encontramos con el problema de que las opiniones de los usuarios son acerca de un conjunto de temas muy variados. Para facilitar el análisis exploratorio de los datos utilizamos métodos que nos segmentan el conjunto de tweets por subtemas y posteriormente podemos clasificar cada uno de ellos para ver si se expresa sentimiento positivo o negativo en general hacia el mismo.

Para poder explicar los métodos de agrupamiento que utilizamos, primero debemos introducirnos en el aprendizaje automático supervisado y no supervisado.

4.1. Aprendizaje automático supervisado y no supervisado

En general, el problema del aprendizaje automático intenta predecir propiedades desconocidas de los datos a partir de información no estructurada suministrada en forma de ejemplos. Este problema se puede subdividir en dos categorías.

En primer lugar, en un problema de aprendizaje supervisado donde conocemos cuáles son los resultados deseados a partir de los datos de entrenamiento y nos permite tener una idea de la relación que existe entre éstos. El objetivo del aprendizaje supervisado es el de crear una función capaz de predecir el va-

lor correspondiente a cualquier objeto de entrada válida después de haber visto los datos de entrenamiento. Para ello, tiene que generalizar a partir de los datos presentados a situaciones no vistas previamente. Esto es lo que hacen los clasificadores de sentimiento que hemos entrenado y evaluado en el capítulo anterior.

Por otro lado, el aprendizaje no supervisado es otro tipo de aprendizaje automático el cual nos permite trabajar con un problema sobre el cual no tenemos información de cómo se debería ver la solución. Se distingue del aprendizaje supervisado en que no cuenta con un resultados etiquetados manualmente. Se genera un modelo a partir de las relaciones, descubriendo así una estructura presente entre los datos.

El objetivo de este tipo de aprendizaje podría ser ver una versión resumida de los datos para que un experto pueda interpretarlos rápidamente; o bien descubrir patrones significativos presentes en los datos. Este tipo de uso se llama análisis exploratorio de datos.

En la siguiente sección describimos en detalle la técnica de aprendizaje no supervisado que vamos a estar usando, llamada agrupamiento o *clustering*.

4.2. Fundamentos de clustering

Los cerebros humanos son buenos encontrando regularidades en los datos. Una forma de hacerlo es agrupar aquellos objetos que son similares entre sí. Por ejemplo, los biólogos descubrieron que la mayoría de los seres vivos caen en una de dos categorías: seres vivos de distintos colores que pueden moverse y seres vivos que de color verde que no pueden moverse. La primer categoría es llamada animales, la segunda, plantas. Llamamos a esta operación de agrupamiento *clustering*.

Supongamos que un biólogo encuentra un nuevo ser vivo color verde que no se ha visto antes, su modelo de plantas y animales predecirá ciertos atributos del ser verde: que no se moverá, que si lo toca se puede raspar o incluso envenenar;

que se podría enfermar si lo come. Todas estas predicciones, si bien no son necesariamente ciertas, son útiles porque permiten al biólogo sacar más partido de sus recursos de análisis.

Los métodos de clustering tratan de establecer grupos de objetos que están en un espacio n-dimensional, de forma que los objetos que están más cercanos queden agrupados en un mismo grupo y que los distintos grupos estén lejanos entre sí.

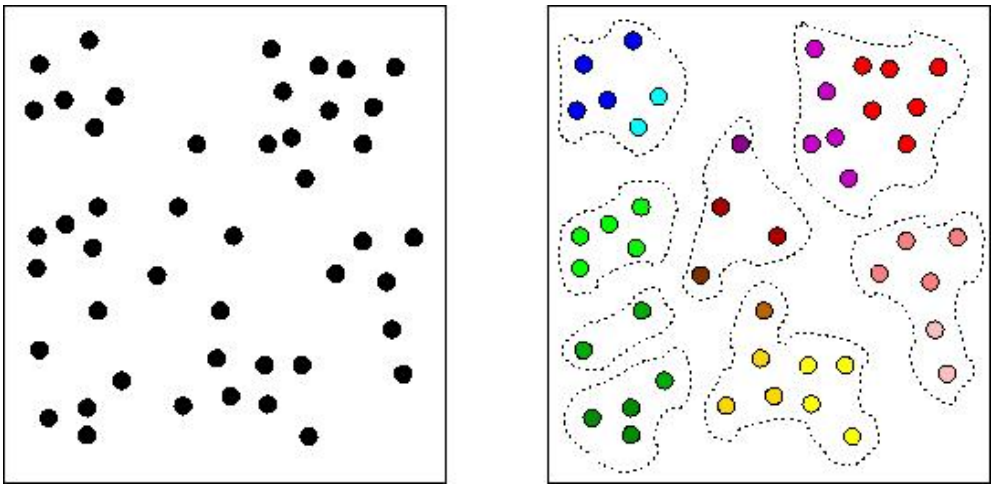


Figura 4.1: Resultado de realizar clustering en un conjunto de datos utilizando 8 clusters. Los colores se utilizan para diferenciar los clusters entre sí.

En la figura 4.1 podemos ver cómo dos o más puntos pertenecen al mismo cluster ya que están uno cerca del otro. Los puntos son objetos en un espacio n-dimensional, representados como vectores.

Se pueden aplicar métodos de clustering para obtener grupos de objetos semejantes en la realidad. Para ello necesitamos representar a los objetos como vectores y trasladar la noción de semejanza a una similaridad en un espacio n-dimensional.

Por lo tanto, el procesamiento de vectorización de datos que mencionamos en la sección 2.2 es esencial para poder representar los tweets como puntos en un espacio n-dimensional, es decir como vectores.

En nuestro problema, tener clusters de tweets de un determinado dominio nos puede resultar útil de la misma forma. Al momento de recolectar nuevos tweets

sobre un tema, podemos asignarlos a uno de los clusters que ya hemos identificado previamente en nuestro corpus sobre el mismo tema, asumiendo así que comparte las mismas propiedades que los elementos en ese cluster. Esto es de utilidad para conocer mejor nuestros datos y poder tomar mejores decisiones [12]

Existen varios tipos de algoritmos de clustering, entre ellos algoritmos por densidad, por distribución y por agrupamiento basado en centroides. En este trabajo utilizamos el algoritmo de Lloyd de agrupamiento basado en centroides, a menudo referido como *"K-means"*.

K-means

La idea principal es dividir es dividir M puntos de N dimensiones definiendo K centroides y luego tomar cada punto de la base de datos y situarlo en la clase de su centroide más cercano, de tal forma de minimizar la suma de las funciones de distancia de cada punto en el cluster con respecto al centroide. Es decir:

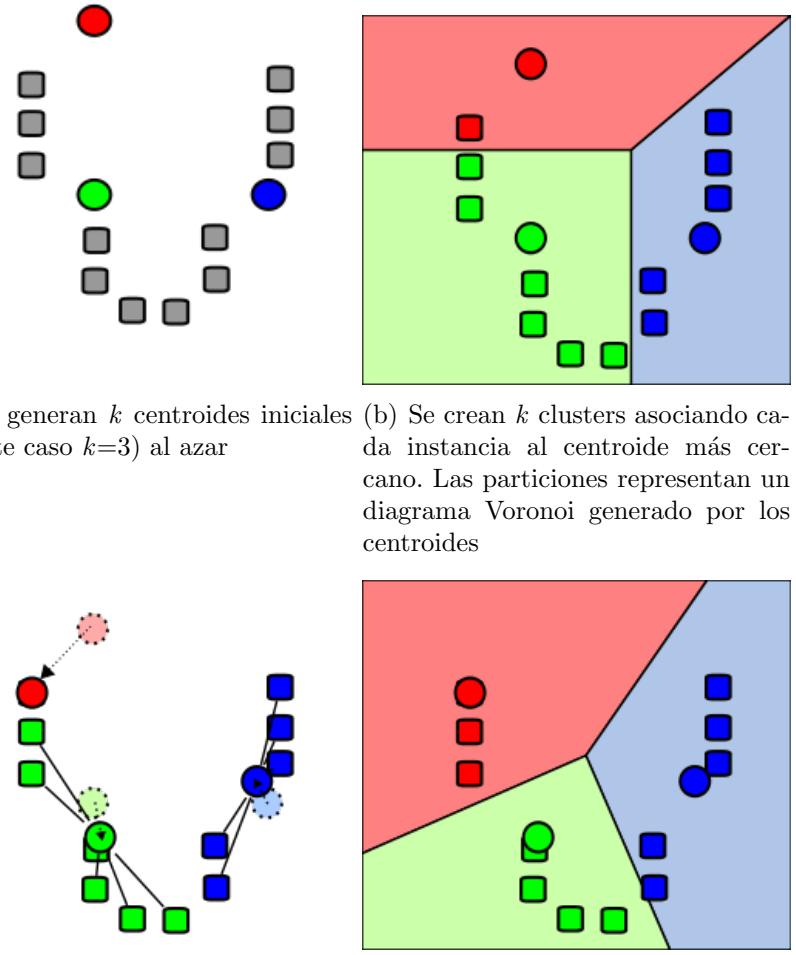
$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

donde u_i es la media de los puntos en S_i.

El próximo paso es calcular nuevamente el centroide de cada grupo y volver a distribuir todos los objetos según el centroide más cercano. El proceso se repite hasta que ya no hay cambio en los grupos de un paso al siguiente.

K-means converge rápidamente, deteniéndose luego de que no haya habido cambios de una iteración a la siguiente, sin embargo no garantiza que siempre llegará a un mínimo global por lo que se puede definir un máximo de iteraciones para evitar que el algoritmo sea más efectivo. [8].

La figura 4.2 muestra los pasos que lleva a cabo K-Means para encontrar una solución en 3 clusters en un espacio de 2 dimensiones requiriendo solo 2 iteraciones para converger. Las figuras circulares representan a los centroides y los cuadrados a las diferentes instancias de los datos.



(a) Se generan k centroides iniciales (en este caso k=3) al azar

(b) Se crean k clusters asociando cada instancia al centroide más cercano. Las particiones representan un diagrama Voronoi generado por los centroides

(c) El centroide de cada uno de los k clusters se calcula nuevamente

(d) Se repiten los pasos (b) y (c) hasta converger

Figura 4.2: Ejemplo de clustering utilizando K-Means con k=3 clusters. Las figuras circulares representan a los centroides y los cuadrados a las diferentes instancias de los datos.

Las técnicas basadas en centroides son apropiadas para datos que pueden ser representados en un espacio Euclídeo en donde es posible calcular la distancia entre un centroide y un elemento del conjunto.

4.3. Coeficiente de silhouette

Para poder decidir si el resultado obtenidos luego del proceso de clustering con una determinada configuración corriendo k-means es útil poder identificar si

los clusters están bien formados. Es decir que necesitamos saber si hay cohesión entre los elementos que pertenecen a un mismo cluster y qué tan separados están de aquellos elementos en otros clusters. Para esto utilizamos el coeficiente de silhouette.

Se calcula, en primer lugar, tomando un punto p y calculando la distancia promedio al resto de los elementos de su mismo cluster. Llamaremos a este valor *a(p)*. Luego para el mismo punto p se calcula cual es la distancia promedio con respecto a todos los elementos de otro cluster que no contengan a p. Repetimos esto buscando el mínimo para todos los clusters. Llamaremos a este valor *b(p)*.

Finalmente el coeficiente de silhouette para p será

$$silhouette(p) = \frac{b(p) - a(p)}{max(a(p), b(p))}$$

Podemos tomar el promedio de todos los coeficientes de silhouette como una métrica del proceso de clustering que varía entre -1 y 1, donde un valor más alto indica mejor cohesión y separación entre los clusters.

4.4. Problemas habituales de clustering

Si bien K-means plantea buscar los mínimos cuadrados lo que lleva a un algoritmo bastante simple, tiene ciertos problemas y limitaciones con los cuales nos topamos a la hora de trabajar con los tweets recolectados.

En primer lugar, elegir el número de clusters adecuado. Si bien hay varios estudios en cómo definir la cantidad apropiada de agrupaciones[4, 13], no hay una forma específica para determinar en cuántos clusters deberíamos dividir nuestro corpus.

En segundo lugar, no siempre hay un estructura que se pueda reconocer en los datos. Si los datos no son naturalmente divisibles, o no lo son en las dimensiones en que los hemos representado o con la medida de distancia elegida o en el número

de clusters elegido, al clasificarlos vamos a estar generando clusters que no representen una estructura significativa. En la figura 4.3 se puede ver como K-Means divide los objetos en 2 clusters aunque no exista tal división en los datos.

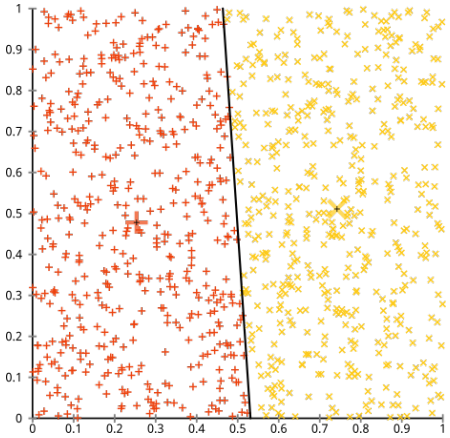


Figura 4.3: Ejemplo de correr K-Means con k=2 en un dataset con datos distribuidos de manera uniforme donde es evidente no existen grupos significativos en los datos.

Asimismo también puede ocurrir que haya un cluster que tenga un volumen mucho mayor al resto de los clusters, conteniendo así a casi todos los elementos. Este problema es dificil de identificar verificando la cohesión entre clusters, pero lo podemos detectar si es que vemos que un cluster es órdenes de magnitud más grande que el resto. Identificaremos a este problema como catch-all cluster.

Por último, K-means puede converger en un mínimo local. Esto se debe que al inicializar los centroides, estos están distribuidos de tal forma que los resultados contradicen la estructura obvia en los datos. En la figura 4.4 se puede ver como una semilla aleatoria inicial utilizada para definir los centroides coloca a dos de ellos juntos, de tal forma que los clusters resultantes subdividen lo que intuitivamente debería ser un solo grupo; mientras que hay dos grupos encerrados en un solo cluster.

Si bien esto se debe principalmente a que los centroides iniciales son elegidos al azar, hay diferentes formas de encarar este problema, entre ellas utilizar otros

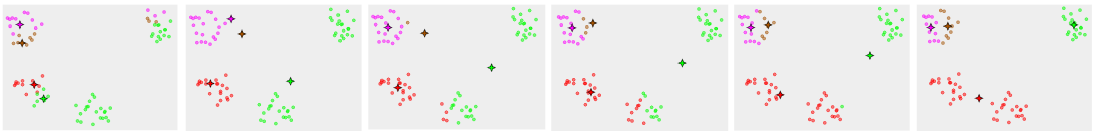


Figura 4.4: Un ejemplo de K-Means convergiendo a un minimo local. En 5 iteraciones el resultado contradiciendo la estructura obvia de los clusters que existe en los datos.

algoritmos como K-means++, que especifica un procedimiento para inicializar los centroides antes de que K-means comience[1][3], o el método Forgy, que selecciona k elementos pertenecientes al conjunto como centroides. Otra forma también utilizada es la de ejecutar varias veces K-means y que el resultado final sea aquel que minimice mejor la suma de los cuadrados para cada cluster.

4.5. Descripción de soluciones

Para este trabajo se utilizó *Scikit-Learn* [15]. Esta librería cuenta con una amplia variedad de herramientas para realizar tareas de minería y análisis de datos, que nos permitieron crear nuestro pipeline de procesamiento y clusterización sin tener que enfocarnos en la implementación de los algoritmos.

Para estos experimentos se utilizó un dataset compuesto por más de 122 mil tweets que compartían en común la etiqueta #oscars. En el capítulo 2 se pueden ver más estadísticas sobre los datos y también información sobre procesamiento previo que realizamos para reducir el volumen del corpus .

A modo de referencia describimos brevemente los diferentes parámetros que nos permitían configurar el pipeline a la hora de realizar la etapa de clustering. Podemos encontrar información más detallada sobre cada uno de ellos en la sección 2.2.

Etapas de pre-procesamiento

- **Stemming (*stem*)**: Reducir cada token a su raíz, agrupando aquellos que hacen referencia al mismo concepto pero son representados por diferentes variantes morfológicas.
- **N-grams (*n-grams*)**: Representaciones más detalladas de secuencias de a lo sumo n palabras que han ocurrido en el texto como características.
- **Binarización (*bin*)**: Se representa marcando la ocurrencia de características en el tweet. 1 si el n-grama ocurre en el tweet, y 0 si no ocurre.
- **Frecuencia de términos (*tf*)**: Similar a bin, pero enumerando cuantas veces ocurre la caracteristica en el tweet.
- **Frecuencia de términos - frecuencia inversa de documento (*tf*idf*)**: A tf pero atenuando el peso de las características más ocurrentes.
- **Umbral de frecuencia (*max df y min df*)**: Definimos una cota inferior (*min df*) para filtrar aquellas características que son poco frecuentes, también definimos una cota superior (*max df*) para filtrar aquellas características que son más frecuentes en el corpus y pueden ser consideradas *stop words* del dominio.

Podemos agregar a esta configuración un valor ***K*** que será utilizado para que k-means defina el número de conjuntos (o clusters) en los que se dividirán los datos. Para hacer una búsqueda más exhaustiva de cual es la mejor configuración definiremos un rango de de valores para K para cada configuración y finalmente compararemos cual fue la que obtuvo el mayor valor de coeficiente de silhouette.

Luego, llamaremos configuración a una 5-tupla de la forma:

"(min_df=x, max_df=y, K=z, n_grams=a,b), preprocessors=[...])"

Esta tupla sintetiza todos los valores que mencionamos previamente y también

cuales son los métodos de preprocesamiento que utilizamos en un experimento.

Para definir para cada uno de estos parámetros que componen nuestra configuración comenzamos por una etapa de exploración manual de los datos probando diferentes cotas para nuestro umbral de frecuencia y como estas reducían el volumen de los datos. Notamos que el 87 % de los características se repetía hasta 5 veces en el corpus, mientras que sólo el 5 % aparecía más de 10 veces en el corpus, lo que reducía drásticamente el volumen de los datos dejando sólo aquellas características más relevantes. A partir de esto probamos diferentes valores para *max df* y *min df* cercanos a estas cotas.

Para el resto de los parametros (*stem*, *n-grams*, *bin*, *tf* y *tf*idf*) probamos diferentes combinaciones arbitrariamente para abarcar exhaustivamente todas las posibilidades para diferentes rangos de K.

La tabla 4.1 muestra las diferentes soluciones utilizando diferentes configuraciones. Todas fueron realizadas con valores de k entre 4 y 9 clusters para K-Means. Cada configuración cuenta con un número de features, un mínimo de frecuencia (*min-df*), un máximo de frecuencia(*max-df*), el rango de n-gramas que se utilizaron y cuales fueron las etapas de pre-procesamiento utilizadas. También encontramos cual fue el mayor valor de la métrica de Silhouette y para que número de clusters. A modo de conclusión de esta serie de experimentos, podemos ver que en la mayoría de las soluciones la métrica de Silhouette no marca una gran diferencia entre la variedad de resultados. Si bien la configuración ***"(min_df=4, max_df=1.0, K=4, n_grams=(2,3), preprocessors=[])"*** (resaltada en el cuadro 4.1) es la que puntúa mejor con un valor de 0.344, al explorar manualmente los datos vimos que esta solución, como muchas otras de la tabla, tenía el problema de *catch-all cluster*. En estos casos ocurre que, dependiendo de la configuración, la clase mayoritaria contenía entre un *78.64 %* y *92.49 %* de la población total.

Features	Min-df	Max-df	N-gramas	Pre-procesamiento	Mejor Silhouette	Mejor K
3448	10	0.01	(1-1)	stem-tf*idf-bin	0.037	6
3448	10	0.01	(1-1)	stem-tf*idf	0.036	7
6353	10	0.01	(1-2)	stem-tf*idf	0.023	7
2459	15	0.01	(1-1)	stem-tf*idf-bin	0.040	6
2459	15	0.01	(1-1)	stem-tf*idf	0.051	5
2413	15	0.01	(2-3)	stem-tf*idf-bin	0.135	8
2317	15	0.01	(2-3)		0.179	4
18878	2	1.0	(1-1)	stem-tf*idf-bin	0.067	8
27183	3	0.01	(2-3)	stem-tf*idf-bin	0.046	2
27183	3	0.01	(2-3)	stem-tf*idf	0.044	5
9991	3	1.0	(1-1)	stem-tf*idf-bin	0.085	7
9991	3	1.0	(1-1)	stem-tf*idf	0.079	7
26380	3	1.0	(1-2)	stem-tf*idf-bin	0.078	4
26380	3	1.0	(1-2)	stem-tf*idf	0.076	7
37222	3	1.0	(1-3)	stem-tf*idf-bin	0.074	5
37222	3	1.0	(1-3)	stem-tf*idf	0.069	5
7735	4	1.0	(1-1)	stem-tf*idf-bin	0.074	7
18617	4	1.0	(1-2)	stem-tf*idf-bin	0.073	4
18617	4	1.0	(1-2)	stem-tf*idf	0.066	8
25466	4	1.0	(1-3)	stem-tf*idf-bin	0.068	4
***25466**	**4**	**1.0**	**(1-3)**	**stem-tf*idf**	**0.072**	**4**
25988	4	1.0	(1-3)		0.208	4
17257	**4**	**1.0**	**(2-3)**		**0.344**	**4**
6155	5	0.01	(1-1)	stem-tf*idf-bin	0.044	5
6262	5	1.0	(1-1)	stem-tf*idf-bin	0.077	7
6262	5	1.0	(1-1)	stem-tf*idf	0.071	7
13772	5	1.0	(1-2)	stem-tf*idf	0.079	5
14353	5	1.0	(1-2)		0.093	4
18042	5	1.0	(1-3)	stem-tf*idf-bin	0.078	6
18042	5	1.0	(1-3)	stem-tf*idf	0.075	8
11780	5	1.0	(2-3)	stem-tf*idf-bin	0.094	5
11780	5	1.0	(2-3)	stem-tf*idf	0.092	8

Cuadro 4.1: Tabla de soluciones con diferentes configuraciones. Todas fueron realizadas con valores de k entre 4 y 9 clusters para K-Means. Cada configuración cuenta con un número de features, un mínimo de frecuencia (*min-df*), un máximo de frecuencia(*max-df*), el rango de n-gramas que se utilizaron y cuales fueron las etapas de pre-procesamiento utilizadas. También encontramos cual fue el mayor valor de la métrica de Silhouette y para que número de clusters.

Sin embargo, luego de examinar manualmente los resultados, cómo estaban conformados los clusters, los n-gramas más frecuentes y también haciendo una breve investigación sobre el dominio, pudimos notar una configuración que resultaba ser intuitivamente más útil que las demás para describir los datos. La configuración es: ***"(min_df=4, max_df=1.0, K=8, n_grams=(1,3), preprocessors=[stem, tf*idf])"*** (resaltada con un "*" en el cuadro 4.1) y se detalla a continuación:

- Features: 25466
- N-grams: (1,3)
- Min df: 4
- Max df: 1.0
- N-Clusters: 8
- Preprocessors: stem + tf*idf

En los 8 clusters de esta solución casi el *76 %* de la población se distribuye equitativamente en dos clases y el *24 %* se distribuye en los otros 6 clusters que restan sin que haya una clase predominante mayor entre ellos.

4.6. Clasificador y validación

Ahora que tenemos los clusters formados queremos contemplar el caso en el que agreguemos un nuevo tweet a nuestro dataset. Para esto podemos entrenar un clasificador utilizando los clusters como etiquetas de clases. Una vez entrenado el clasificador, podremos predecir efectivamente a que cluster pertenece según con la etiqueta de que clase fue clasificado. Cada vez que queramos clasificar un tweet que no pertenecía al dataset anteriormente tendrá que ser pre-procesado por el *pipeline* descrito en la sección 2.2. Luego utilizamos una máquina de vectores de

soporte (*Support Vector Machines, SVMs*) para predecir a que clase pertenece. Utilizando una penalidad C de 0.1 y utilizando la función de pérdida "squared hinge", obtuvimos el mayor rendimiento, de *98.08 %*.

Luego de ver este resultado y analizando el funcionamiento del clasificador, notamos que las *SVMs* eran propensa a clasificar las instancias como pertenecientes a la clase mayoritaria. Al notar esto decidimos comparar los resultados con un árbol de decisión como clasificador. De la misma forma que lo hicimos para nuestra *SVM*, utilizamos Grid Search para hacer una búsqueda exhaustiva en cuanto que combinación de parametros obtenía un mayor puntaje. Utilizando el criterio de impureza de *Gini* y con un maximo de 9 niveles de profundidad, obtuvimos un *95 %* de efectividad.

Capítulo 5

Visualización

Una vez finalizadas las etapas de análisis de sentimientos y de clustering, contamos con dos tipos de resultados diferentes que sería de utilidad poder ver conjuntamente. Por un lado tenemos a cada tweet etiquetado como "positivo" o "negativo" según el sentimiento expresa el mensaje; por el otro, una división del corpus en diferentes subgrupos que consideramos ser los diferentes aspectos de nuestro tema a analizar.

Ya que queremos hacer un análisis exploratorio de los datos, carece de sentido hacer una evaluación rigurosa de los resultados de etapas anteriores. Es por esto que toleramos un margen de error en cuanto a la clasificación de sentimientos y las clasificaciones de aspectos. Estamos interesados en tener una idea general de cuáles son las diferentes tendencias con respecto a determinado contenido.

También considerando los grandes volúmenes de los datos que manejamos, poder analizar los resultados manualmente se vuelve una tarea casi imposible. Es en este contexto que el uso de visualizaciones nos permite identificar aquella información relevante de forma condensada. Las visualizaciones nos permiten resumir todas aquellas características de los datos en las cuales estamos interesados en representaciones gráficas, haciendo así que sean fáciles de interpretar para hacer

mejores conclusiones y luego poder tomar decisiones basada en la información procesada.

5.1. Descripción de la solución requerida

En el marco de este trabajo buscamos diferentes tipos de visualizaciones que nos permiten detectar qué sentimientos hay con respecto a los diferentes aspectos de un tema. Las características y la justificación de cada una de ellas se detalla a continuación:

- Cada cluster está compuesto por tweets, por lo que nos gustaría poder tener un pantallazo viendo cuales son las términos que los componen y que tan frecuente son entre los tweets del mismo cluster.

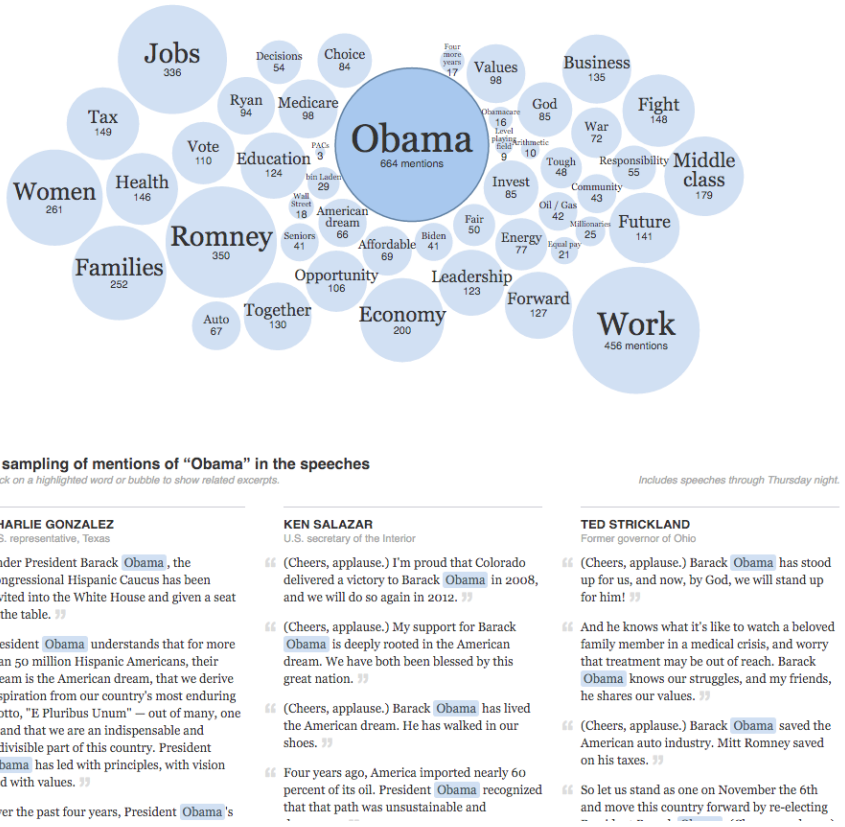
- Muchas veces una palabra no tiene una polaridad positiva o negativa directamente asociada, como por ejemplo "#oscars" o "academy". Entonces, buscamos poder ver visualmente una aproximación que nos indique si en este corpus esta palabra aparece principalmente en contextos negativos o positivos.

- Si bien el punto anterior nos permite rápidamente identificar el sentimiento asociado a un término, da lugar a un margen grande de error y deja de lado la información del contexto en el que ocurre el tweet, por lo que también queremos ver un listado de tweets en los cuales este término ocurre.

5.2. Descripción de la solución

Entre las diferentes visualizaciones que analizamos decidimos realizar la nuestra basada en el trabajo que realizó el periódico The New York Times para analizar cuáles eran las palabras más frecuentes que se utilizaban en discursos políticos.

En la figura 5.1 se muestra una imagen de esta visualización[1].



A sampling of mentions of "Obama" in the speeches

Click on a highlighted word or bubble to show related excerpts.

Includes speeches through Thursday night.

CHARLIE GONZALEZ
U.S. representative, Texas

- Under President Barack Obama, the Congressional Hispanic Caucus has been invited into the White House and given a seat at the table.
- President Obama understands that for more than 50 million Hispanic Americans, their dream is the American dream, that we derive inspiration from our country's most enduring motto, "E Pluribus Unum" — out of many, one — and that we are an indispensable and indivisible part of this country. President Obama has led with principles, with vision and with values.
- Over the past four years, President Obama's

KEN SALAZAR
U.S. secretary of the Interior

- (Cheers, applause.) I'm proud that Colorado delivered a victory to Barack Obama in 2008, and we will do so again in 2012.
- (Cheers, applause.) My support for Barack Obama is deeply rooted in the American dream. We have both been blessed by this great nation.
- (Cheers, applause.) Barack Obama has lived the American dream. He has walked in our shoes.
- Four years ago, America imported nearly 60 percent of its oil. President Obama recognized that that path was unsustainable and

TED STRICKLAND
Former governor of Ohio

- (Cheers, applause.) Barack Obama has stood up for us, and now, by God, we will stand up for him!
- And he knows what it's like to watch a beloved family member in a medical crisis, and worry that treatment may be out of reach. Barack Obama knows our struggles, and my friends, he shares our values.
- (Cheers, applause.) Barack Obama saved the American auto industry. Mitt Romney saved on his taxes.
- So let us stand as one on November the 6th and move this country forward by re-electing

Figura 5.1: Visualización de The New York Times (NYT) analizando palabras frecuentes en discursos politicos

Esta visualización está compuesta por una combinación de una visualización de nube de palabras, representadas con burbujas, y también un listado de recortes de discursos.

En la nube de palabras, cada burbuja contiene un término frecuente y el número de veces que esta se puede encontrar en diferentes discursos, el tamaño de cada burbuja es relativo a la cantidad de menciones del término que representa.

Asimismo, el listado de recortes está vinculado con la nube de palabras de tal forma que cuando seleccionamos una burbuja solo veremos aquellos fragmentos de discurso en los cuales aparece resaltado el término asociado a la burbuja.

[1] http://www.nytimes.com/interactive/2012/09/04/us/politics/democratic-convention-words.html

Nuestra visualización

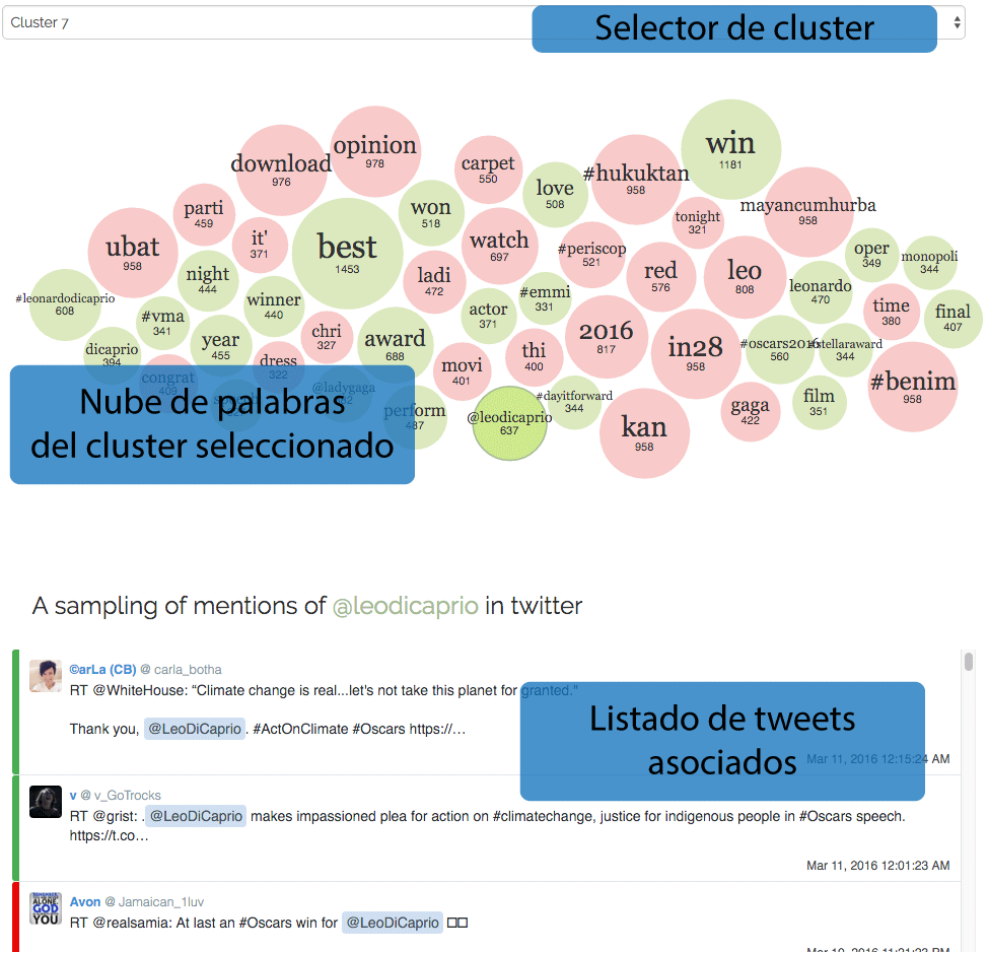


Figura 5.2: Visualización de la solución diseñada para ver gráficamente el resultado de nuestro pipeline

Tomando las principales características de la visualización del NYT, utilizamos una nube de palabras y un listado de tweets para la visualización de cada cluster. Para ajustarlo a nuestras necesidades agregamos dos características que nos permitían comunicar visualmente el sentimiento asociado a los tweets y consecuentemente a los términos asociados en la nube de palabras.

También como ampliación a esta visualización buscamos una forma de poder visualizar la solución globalmente integrando todas las nubes de palabras con un tamaño proporcional a la población y representando la semejanza relativa entre clusters. Sin embargo, debido a la complejidad de esta tarea y a las limitaciones de

la librería que utilizamos para graficar tuvimos que dejarlo a fuera del alcance de este proyecto. Por lo que agregamos un selector que nos permite poder seleccionar entre los diferentes clusters. La figura 5.2 muestra el resultado de esta etapa.

La nube de palabras

Figura 5.3: Visualización de nube de palabras utilizando burbujas

Como muestra la figura 5.3, para la implementación de la nube de palabras, utilizamos la librería D3[2] que nos permitió llevar los datos recolectados en las etapas anteriores a una implementación visual para representar los clusters de forma resumida y destacar aquellos aspectos que consideramos más importantes para explorar los datos.

En cuanto a la descripción de las burbujas es similar a la del NYT pero decidimos colorear cada burbuja según el sentimiento asociado a los tweets en los cuales aparece el término asociado. Si aparece más veces en tweets que fueron clasificados como positivos y no tantas veces en tweets negativos, lo identificaremos de manera visual con un color verde. En caso contrario, si ocurrió más veces en tweets negativos que positivos, lo coloreamos con un color rojo.

Así mismo, otra característica de este gráfico es que se puede interactuar

[2] https://d3js.org/

directamente con las burbujas arrastrándolas y acomodándolas en el espacio. Esto es útil para el caso de uso de que un analista explore el mercado y desee realizar una presentación de sus conclusiones.

Listado de tweets

Figura 5.4: Visualización del listado de tweets etiquetados según su sentimiento

Cuando seleccionamos uno de los nodos del gráfico que se encuentra en la parte superior, automáticamente se despliega un listado filtrado de tal forma que solo aparecen tweets que contienen resaltado el término asociado al nodo seleccionado como muestra la figura 5.4.

A la hora de implementar este listado de tweets, decidimos hacer una implementación similar al news feed de Twitter. Tenemos un listado donde los tweets se muestran uno arriba del otro y de cada uno de ellos podemos ver el mensaje, la información del usuario y la fecha de publicación. Sumado a esto cada tweet tiene un señalador que nos dice si el sentimiento asociado al mismo es positivo o negativo con un color verde o rojo respectivamente.

5.3. Análisis de los resultados

Luego de haber realizado la etapa de clasificación de sentimientos y de clustering, este proceso culmina con la posibilidad de ver visualmente sobre qué temas

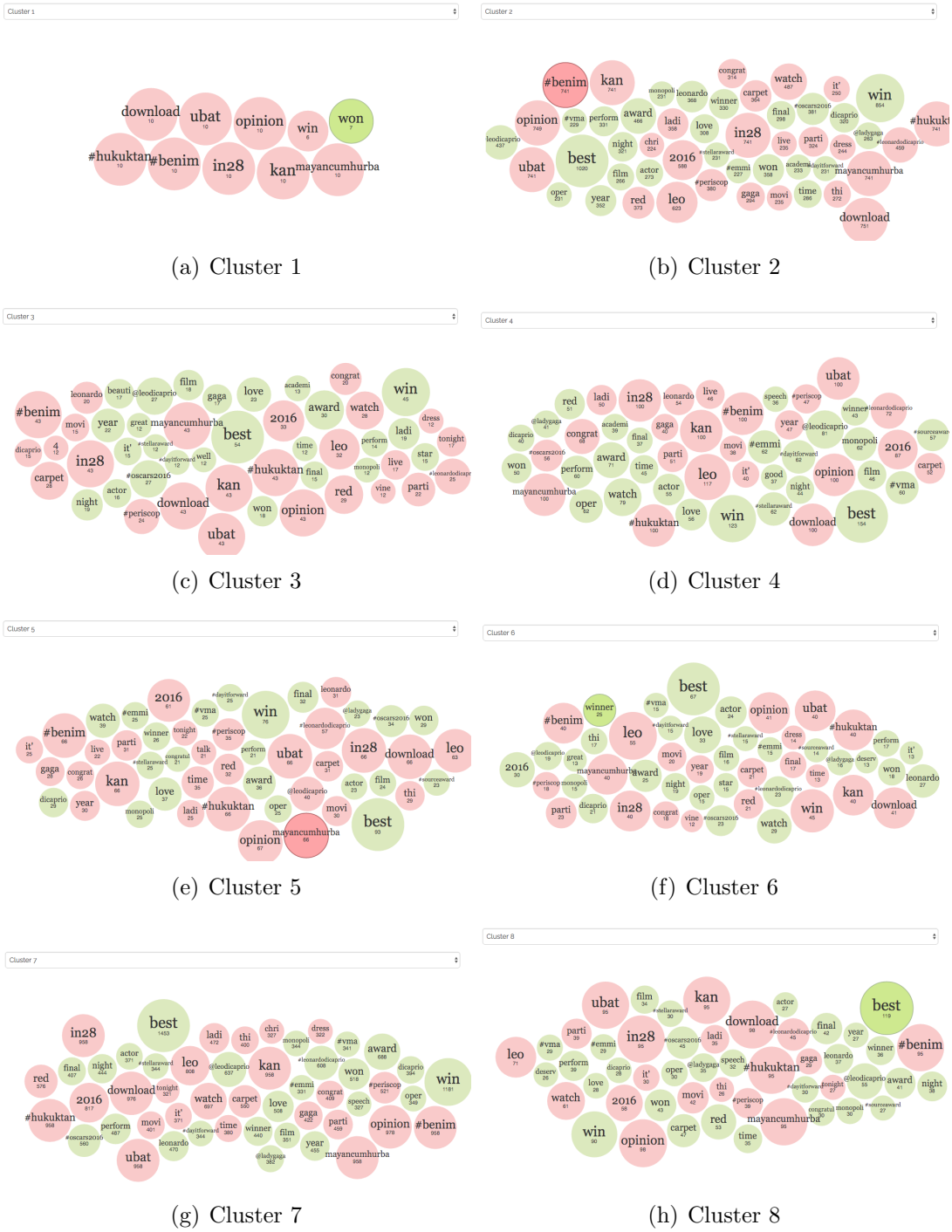
los usuarios de Twitter comentaron durante la gala de los Óscars. Esta visualización nos facilita explorar la opinión de los usuarios sobre cada tema haciendo uso de las palabras más frecuentes y cual es el sentimiento asociado (positivo o negativo) a cada una de ellas según la clasificación de sentimiento de los tweets, propios de cada tema, donde ocurren.

Luego de ver graficado cada cluster en la figura 5.5 podemos realizar las siguientes serie de observaciones. En primer lugar, a excepción del Cluster 1 donde predominan los términos negativos, en ningún otro cluster es notable la diferencia entre la cantidad de términos positivos y negativos. Podríamos decir que está casi balanceada.

También podemos observar que algunas palabras, tales como *"leonardo"* (cluster 4 y cluster 6) y *"gaga"* (cluster 3 y cluster 4), ocurren en diferentes clusters pero con diferente frecuencia e incluso con un sentimiento asociado diferente. Esto último se debe en parte a que los tweets propios de cada cluster son diferentes, por lo tanto la frecuencia con la que aparecen ciertos términos puede ser utilizada para caracterizar las particularidades de cada tema.

Asimismo, podemos notar que términos como *"@leodicaprio"* y *"@ladygaga"* ocurren siempre con un sentimiento positivo mientras que *"leo"* y *"opinion"* lo hacen con un sentimiento negativo. En esto podemos ver la particularidad de que cuando se hace referencia a ciertas entidades, como al actor *Leonardo DiCaprio*, de forma positiva se utilizan ciertos términos en el mensaje (*"@leodicaprio"*) distintos a cuando se lo hace de forma negativa (*"leo"*).

Por otro lado, nos encontramos con una de las particularidades de Twitter en cuanto qué tan frecuente es compartir las publicaciones de otros usuarios (*retweets*). En el cluster 1 podemos notar que la nube de palabras por sí sola nos



(a) Cluster 1 (b) Cluster 2

(c) Cluster 3 (d) Cluster 4

(e) Cluster 5 (f) Cluster 6

(g) Cluster 7 (h) Cluster 8

Figura 5.5: Visualización de sentimientos etiquetados en los diferentes clusters

indica que está compuesto por pocos tweets repetidos múltiples veces. En clusters caracterizados por más términos solo se podría detectar que esto ocurre examinando exhaustivamente el listado de tweets asociado a cada término, dándonos

así una primera impresión errónea en cuanto a las características de cada tema.

Por último, hay ciertos términos como *"#benim"*, *"download"* y *"in28"* que no aportaron ningún valor para describir los temas e incluso dificultaron realizar un análisis semántico de tal forma que lo podríamos considerar como *spam*. Podríamos abordar este problema con un preproceso más agresivo, eliminando términos que ocurran de forma indistinta en tweets muy diferentes entre sí, por ejemplo, usando test de hipótesis o información mutua como métricas más sofisticadas con el mismo espíritu que el tf*idf, o bien enfocando nuestro análisis utilizando aquellos términos que son más característicos del tema.

Capítulo 6

Conclusiones y Trabajo Futuro

6.1. Conclusiones

En este trabajo, hemos estudiado un problema complejo como es el de la minería de opiniones en Twitter y los desafíos asociados al análisis exploratorio de los resultados de dicho proceso. Si bien Twitter tiene ciertas limitaciones a la hora de proveer información, es más que suficiente para llevar a cabo un análisis exploratorio con la finalidad de comprender mejor un mercado o un acontecimiento. Para esto nos hemos centrado en diseñar un pipeline en el cual adquirimos, pre-procesamos, hacemos clustering y finalmente vemos representados de forma visual un conjunto de tweets sobre un determinado tópico.

En cada etapa de este pipeline identificamos conjuntos de problemas y diferentes soluciones que se podían llevar adelante para cada uno de ellos. Si bien este trabajo propone una implementación concreta para este flujo, existe la posibilidad de continuar investigando en mayor profundidad de manera independiente cada etapa.

6.2. Trabajo futuro

A partir del estado actual de la implementación resultante de este trabajo se desprenden diferentes líneas de trabajo sobre las cuales se podrían continuar.

Entre ellas la que sin duda podría tener mayor contribución al resultado final es la de realizar un preprocesamiento de los mensajes más agresivo para que las características resultantes permitan una implementar los métodos de clustering con mayor efectividad.

Asimismo, si bien abordamos la minería de opiniones como un problema de clasificación de textos, podría implementar un sistema más complejo de múltiples etapas que contemplase diferentes particularidades del lenguaje natural como la ironía o las negaciones.

Finalmente, en cuanto a la visualización de los resultados del pipeline, encontrar un esquema gráfico que permita procesar visualmente de forma rápida y efectiva es una tarea desafiante por el gran volumen y la complejidad de los datos que manejamos. Uno de los problemas a destacar que resulta atractivo para continuar es la de la representación de un espacio de clusters.

Bibliografía

[1] David Arthur y Sergei Vassilvitskii. K-means++: The advantages of careful seeding. En *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, págs. 1027–1035. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2007. ISBN 978-0-898716-24-5. URL http://dl.acm.org/citation.cfm?id=1283383.1283494.

[2] John Blitzer, Mark Dredze, y Fernando Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. En *In ACL*, págs. 187–205. 2007.

[3] M. Emre Celebi, Hassan A. Kingravi, y Patricio A. Vela. A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Syst. Appl.*, 40(1):200–210, 2013. URL http://dblp.uni-trier.de/db/journals/eswa/eswa40.html#CelebiKV13.

[4] Chris Fraley y Adrian E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588, 1998. URL http://www3.oup.co.uk/computer_journal/hdb/Volume_41/Issue_08/Fraley.pdf.

[5] Michael Gamon, Anthony Aue, Simon Corston-Oliver, y Eric Ringger. Pulse: Mining customer opinions from free text. En *Proceedings of the 6th International Conference on Advances in Intelligent Data Analysis*, IDA'05, págs. 121–132. Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 3-540-28795-

7, 978-3-540-28795-7. doi:10.1007/11552253_12. URL http://dx.doi.org/10.1007/11552253_12.

[6] Roberto González-Ibáñez, Smaranda Muresan, y Nina Wacholder. Identifying sarcasm in twitter: A closer look. En *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, págs. 581–586. Association for Computational Linguistics, Stroudsburg, PA, USA, 2011. ISBN 978-1-932432-88-6. URL http://dl.acm.org/citation.cfm?id=2002736.2002850.

[7] Minqing Hu y Bing Liu. Mining and summarizing customer reviews. En *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, págs. 168–177. ACM, New York, NY, USA, 2004. ISBN 1-58113-888-1. doi:10.1145/1014052.1014073. URL http://doi.acm.org/10.1145/1014052.1014073.

[8] Anil K. Jain y Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988. ISBN 0-13-022278-X.

[9] Alistair Kennedy y Diana Inkpen. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22(2):110–125, 2006.

[10] Soo-Min Kim y Eduard Hovy. Determining the sentiment of opinions. En *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04. Association for Computational Linguistics, Stroudsburg, PA, USA, 2004. doi:10.3115/1220355.1220555. URL http://dx.doi.org/10.3115/1220355.1220555.

[11] Bing Liu. *Sentiment Analysis - Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, 2015. ISBN 978-1-10-701789-4. URL http://www.cambridge.org/us/academic/subjects/computer-science/

knowledge-management-databases-and-data-mining/
sentiment-analysis-mining-opinions-sentiments-and-emotions.

[12] David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2002. ISBN 0521642981.

[13] Boris Mirkin. Choosing the number of clusters. *WIREs Data Mining Knowl Discov*, 1(3):252–260, 2011. ISSN 1942-4787. doi:10.1002/widm.15. URL http://dx.doi.org/10.1002/widm.15.

[14] Bo Pang y Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. En *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04. Association for Computational Linguistics, Stroudsburg, PA, USA, 2004. doi:10.3115/1218955.1218990. URL http://dx.doi.org/10.3115/1218955.1218990.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, y E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[16] Payam Refaeilzadeh, Lei Tang, y Huan Liu. Cross-validation. En *Encyclopedia of Database Systems*, págs. 532–538. 2009. doi:10.1007/978-0-387-39940-9_565. URL http://dx.doi.org/10.1007/978-0-387-39940-9_565.

[17] inc. Twitter. Api rate limits. . URL https://dev.twitter.com/rest/public/rate-limiting.

[18] inc. Twitter. Glosario de twitter. . URL https://support.twitter.com/articles/352810.

[19] inc. Twitter. Reglas y restricciones de la búsqueda de twitter. . URL https://support.twitter.com/articles/469371.

[20] inc. Twitter. Tweet entity. . URL https://dev.twitter.com/overview/api/tweets.

[21] Changhua Yang, Kevin Hsin-Yih Lin, y Hsin-Hsi Chen. Building emotion lexicon from weblog corpora. En *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, págs. 133–136. Association for Computational Linguistics, Stroudsburg, PA, USA, 2007. URL http://dl.acm.org/citation.cfm?id=1557769.1557809.

[22] Lei Zhang y Bing Liu. Identifying noun product features that imply opinions. En *ACL (Short Papers)*, págs. 575–580. The Association for Computer Linguistics, 2011. ISBN 978-1-932432-88-6. URL http://dblp.uni-trier.de/db/conf/acl/acl2011s.html#ZhangL11.