Universidad Autónoma Metropolitana Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Proyecto de Investigación

Reconocimiento de conversaciones violentas mediante palabras

Presenta:

Adolfo Flores Moreno
Matrícula: 2112002385

Reporte Final

Trimestre 2014 Invierno

Asesores:

Dra. Silvia Beatriz González Brambila, Profesora Titular

Departamento de Sistemas

Dr. Juan Gaspar Vargas Rubio, Profesor Titular

Departamento de Electrónica

México D.F., Abril de 2014

Declaratorias

Yo, Silvia Beatriz González Brambila, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Yo, Juan Gaspar Vargas Rubio, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Yo, Adolfo Flores Moreno, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Resumen

Hoy en día, se producen fácilmente conversaciones violentas, los factores que apoyan su generación son muchos y aunque se han hecho varios estudios al respecto, no se tiene certeza sobre las relaciones causa-efecto, de estas conductas no deseadas en nuestra sociedad.

Aunque existen personas dedicadas a la vigilancia en espacios públicos y privados, esta profesión u oficio se torna monótona y agotadora, ya que las situaciones violentas se pueden dar en un instante inesperado y nunca se está al 100% alerta, para poder detectarlas.

En este sentido, el trabajo desarrollado es un avance que se puede utilizar como ayuda para detectar situaciones violentas mediante palabras. Indagando en distintos repositorios de videos en internet, se tomaron y almacenaron algunos videos que contienen conversaciones en español entre dos personas, clasificándolos en violentos y no violentos.

A partir de los videos recolectados y clasificados en violentos y no violentos, se extrajeron los audios de los mismos, para posteriormente transformarlos en texto y realizar el pre procesamiento.

Una vez que se realizo lo anterior, se creó un archivo que contiene todas las conversaciones (violentas y no violentas) para poder aplicar las técnicas de minería de datos de *clustering*: "Ward"[1], "K-means"[1] y "PAM"[1], para agrupar las palabras, y se compararon los resultados obtenidos por cada técnica validando con clValid, confirmando que la técnica jerárquica fue la optima.

Se creó un archivo con los porcentajes de frecuencia con que aparece cada término, tanto en conversaciones violentas como en no violentas, y su clasificación. Una vez hecho esto, se realizaron diez pruebas para la clasificación de términos en violentos y no violentos mediante la técnica SVM (*Support Vector Machine*)[2], donde se observó que existen términos que no se pueden clasificar, esto debido a que aparecen en el mismo porcentaje en conversaciones violentas como en no violentas. Al realizar las pruebas de entrenamiento de SVM se concluyó que tres de las pruebas realizadas obtuvieron una predicción errónea, ya que los datos de entrada diferían bastante de las predicciones realizadas por SVM, en contraste otras tres de las diez pruebas realizadas obtuvieron mejores predicciones con respecto al conjunto de prueba.

Agradecimientos

Le agradezco a Dios por haberme acompañado y guiado a lo largo de mi carrera, por ser mi fortaleza en los momentos de debilidad y por brindarme una vida llena de aprendizajes, experiencias y sobre todo felicidad.

Le doy gracias a mis padres, Adolfo y Ma del Carmen, por todo el apoyo, cariño y confianza que me han dado a lo largo de mi vida, por los valores que me han inculcado, y por haberme dado la oportunidad de tener una excelente educación en el transcurso de mi vida.

A mis hermanos Jennifer y Kevin por ser parte muy importante de mi vida, por apoyarme en aquellos momentos de necesidad y por apoyar a la unión familiar.

A Yatsiry, por ser una persona muy importante en mi vida, por haberme apoyado en las buenas y en las malas, por su paciencia y amor cuando más lo he necesitado.

A mis tíos, Martin y Jesús, por confiar y creer en mí y porque siempre estuvieron apoyándome en todo momento.

A todos mis familiares y amigos, gracias por su apoyo, comprensión y sobre todo su amistad.

Agradezco a mis asesores de proyecto, Dra. Silvia Beatriz González Brambila y Dr. Juan Gaspar Vargas Rubio, por la oportunidad que me brindaron, así como por su dedicación, ayuda, paciencia y comprensión.

Agradezco a la Universidad Autónoma Metropolitana Azcapotzalco por haberme abierto sus puertas y darme una formación académica de calidad.

"Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber".

Albert Einstein

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Capítulo 1: Introducción

El objetivo principal de este trabajo es avanzar en el estado del arte para ayudar a detectar mediante palabras si una situación en un video presenta violencia. Esto debido a que hoy en día cualquier persona tiene la necesidad de efectuar trámites o realizar compras en diversos establecimientos, y es donde en ocasiones se llegan a suscitar conversaciones violentas.

A partir de las palabras que se lograron extraer del audio de los videos recolectados y clasificados como violentos y no violentos, se generó un archivo de texto de cada uno de ellos con las palabras contenidas en las conversaciones. Realizado lo anterior se procedió a efectuar el pre procesamiento, eliminando palabras vacías y signos de puntuación, dando como producto un archivo maestro que contiene todas las palabras de las conversaciones.

Utilizando los archivos generados, se realizó el análisis con ayuda de las técnicas de minería de datos: "Ward"[1], "K-means"[1] , "PAM"[1] y SVM (*Support Vector Machine*)[2], para agrupar palabras en diferentes clases dependiendo de la similitud de las mismas, y para clasificarlas en violentas y no violentas.

1.1. Objetivos

1.1.1. Objetivo general

Determinar mediante palabras si una situación en un vídeo presenta violencia.

1.1.2. Objetivos específicos

- Extraer el audio (conversaciones) de los videos.
  Se extrajo el audio (conversaciones) de los videos con apoyo de "Adobe Premiere Pro CC"[5].
- Transformar el audio extraído de los videos, en texto.
  Utilizando el paquete "Modelos de análisis de voz de Adobe en Español"[6] (complemento de Adobe Premiere Pro CC) , se convirtió el audio extraído de los videos, en texto.
- Guardar en archivos de texto las palabras extraídas del audio de los videos.
  Las conversaciones extraídas del audio de los videos fueron almacenadas en archivos de texto.
- Pre-procesamiento de archivos.
  Se eliminaron signos de puntuación y palabras vacías de los archivos de conversación generados.
- Clasificar videos mediante técnicas de minería de datos.
  Mediante las técnicas de minería de datos: "*Ward*" [1], "*K-means*" [1] y "*PAM*" [1], se realizó la agrupación de palabras en 8 clústeres y se validaron los resultados con *clValid*. Se clasificaron las palabras en violentas y no violentas con ayuda de la técnica SVM (*Support Vector Machine*) [2].
- Analizar y comparar los resultados de las técnicas de minería de datos utilizadas.
  Los resultados de las técnicas de minería de datos utilizadas, fueron analizados y comparados mediante tablas y gráficas. Los mejores resultados de obtuvieron al realizar la técnica jerárquica "*Ward*" con ocho clústeres.

1.2. Antecedentes

El conocimiento es un recurso de importancia estratégica para las organizaciones, ya que su generación, codificación, gestión y divulgación aportan al proceso de innovación. La cantidad de documentos de diversos tipos disponibles en una organización es enorme y continúa creciendo cada día. Estos documentos, más que las bases de datos, son a menudo un repositorio fundamental del conocimiento de la organización, pero a diferencia de éstas la información no está estructurada.

La Minería de Texto se centra en el descubrimiento de patrones interesantes y nuevo conocimiento en un conjunto de documentos, es decir, su objetivo es descubrir cosas tales como tendencias, desviaciones y asociaciones entre "grandes" cantidades de información estructurada o semi-estructurada. Precisamente esta característica hace necesario aplicar técnicas adecuadas que permitan el análisis de la información y se logre organizar. La etapa de pre procesamiento juega un rol de suma importancia en este proceso, eliminando palabras que no son útiles, así como signos de puntuación [3] [4].

La categorización de documentos de texto es una aplicación de la minería de texto que asigna a los documentos una o más categorías, etiquetas o clases, basadas en el contenido. Es un componente importante de muchas tareas de organización y gestión de la información [3] [4].

Este trabajo a diferencia con la categorización de documentos, se centra en el avance para categorizar videos en violentos o no violentos mediante palabras, utilizando una metodología muy similar al artículo de Yanchang Zhao, "*R and Data Mining: Examples and Case Studies"* [15], donde se presenta un ejemplo de minería de texto de *Twitter,* agrupando las palabras de los *Tweets,* para clasificarlos dependiendo las palabras que se utilicen. A continuación en la Tabla 1 podemos ver las similitudes y diferencias.

	Zhao	Este trabajo
Datos utilizados	Tweets	Videos, Audio, Texto (conversaciones)
Idioma de los datos	Inglés	Español
Pre procesamiento	Si	Si
Validación de agrupamiento	Ninguna	ClValid
Métodos de agrupamiento utilizados	K-means, Ward, PAM	K-means, Ward, PAM
Técnica de Clasificación	Ninguna	SVM

Tabla 1: Comparación de trabajos de minería de texto

1.3. Justificación

El objetivo de la vigilancia es mantener el orden y la seguridad en los establecimientos públicos y privados de nuestro país y del resto del mundo. Casi todas las personas acudimos a diversos lugares para realizar trámites o efectuar compras de bienes o servicios, siendo muy común la presencia de situaciones donde se da lugar a discusiones violentas, por distintos motivos.

El uso de cámaras de video es necesario para poder captar lo que sucede en distintos espacios. Las personas encargadas de vigilar, deben monitorear la imagen de muchos videos, minuto a minuto, siendo esto un trabajo muy agotador y monótono, lo que ocasiona dejar de revisar con regularidad, y es precisamente en esos instantes donde pueden suscitarse discusiones con violencia.

La violencia es un comportamiento deliberado con el que se pretende ocasionar daños físicos o psicológicos a otras personas, animales o cosas. Se le suele asociar con la agresividad, sin embargo, la diferencia entre ambas es que en la violencia dicha agresividad está fuera de control y se orienta a hacer un daño consciente y deliberado[5].

Existen diversos factores que propician una conducta violenta en los individuos, entre ellos tenemos a los individuales, en los que recaen: el estado de ánimo, la baja tolerancia a la frustración, el consumo de sustancias que alteran la conciencia, como las drogas, el alcohol, ciertos medicamentos o cierto tipo de plantas medicinales, o un posible trastorno mental; y los factores sociales y ambientales como: la familia, la escuela, el ámbito laboral, las organizaciones juveniles, instituciones jurídicas, el vecindario, el grupo de amigos, etcétera [5].

Tomando en cuenta estos factores, podemos estar seguros que en cualquier lugar puede surgir la aparición de una conducta violenta, y el no percatarse a tiempo de dichas situaciones, puede ocasionar que una discusión desencadene en agresiones físicas.

La importancia de este trabajo recae en el avance que aporta, lo cual ayuda a imaginar cómo podrían llegar a ser los sistemas de vigilancia en un futuro, en los cuales se emitan señales de alarma cuando se susciten conversaciones con una determinada probabilidad de ser violentas, notificando a tiempo a las personas de vigilancia para que acudan al lugar de inmediato, y así poder evitar situaciones más complejas que podrían llegar a la violencia física.

1.4. Descripción Técnica

A partir de la información recolectada del audio (conversaciones) de varios videos, se analizaron y determinaron las palabras que en una determinada situación sugieren la existencia de violencia.

El desarrollo de este trabajo se basó en los pasos siguientes (ver Figura 1):

- Recolección y clasificación de videos:
  Se realizó la recolección de cien videos que contienen conversaciones violentas y no violentas entre dos personas, en español.

- Extracción de audio:
  Se ejecutó el proceso para extraer el audio (conversaciones) de los videos con apoyo de la herramienta "Adobe Premiere Pro CC" [5].

- Procesamiento de sonido:
  Se efectuó el proceso para decodificar el sonido de las grabaciones conversacionales obtenidas de los videos y fueron transformadas en texto (Voz-Texto) utilizando el paquete "Modelos de análisis de voz de Adobe en Español" [6] (complemento de Adobe Premiere Pro CC).

- Guardar conversaciones:
  Las conversaciones extraídas del audio de los videos se guardaron en archivos de texto.

- Pre-procesamiento de archivos:
  Se procedió a hacer limpieza de palabras en los archivos de las conversaciones generados, y se creó un archivo depurado por cada archivo de video.

- Minería de datos:
  Se analizó el texto arrojado por el audio de los videos mediante algoritmos de análisis de palabras, utilizando las técnicas "Ward" [1], "K-means" [1], "PAM" [1] y SVM (Support Vector Machine) [2], las cuales ayudaron a realizar la agrupación y clasificación de las palabras.

- Análisis de resultados:
  Se obtuvieron los resultados del proceso de minería de datos, a través de las técnicas utilizadas, usando gráficas y tablas.

Figura 1: Diagrama de bloques

1.5. Estructura del documento

Tras el presente capítulo introductorio, pasaremos a un **Segundo Capítulo** en el cuál se describe la teoría conceptual en la cual se fundamenta el trabajo de minería de texto realizado. Se explica que es la herramienta R y la utilidad que tiene en distintos campos del conocimiento. También se habla brevemente sobre la importancia de realizar el pre procesamiento, y se explica en qué consiste el agrupamiento (*clustering*), dando una breve descripción de las técnicas de agrupamiento: Jerárquica (*Hierarchical*), K-means, PAM y SVM (*Support Vector Machine*).

En el **Tercer Capítulo** se describe el desarrollo del trabajo de minería de texto para clasificar conversaciones violentas y no violentas mediante palabras. Se explica todo el proceso que se realizó desde la recolección de los videos con las conversaciones, su clasificación en violentos y no violentos, la extracción del audio, la decodificación para convertirlos en texto, así como también se detallan los pasos realizados en el pre procesamiento de los archivos de texto que contienen las conversaciones. Se muestran las palabras más frecuentes utilizadas en conversaciones violentas y no violentas, mostrándolas en graficas y nubes de palabras. Se realiza el agrupamiento de términos mediante los métodos de *clustering*: "Ward", "K-means" y "PAM", así como la validación de los mismos utilizando *clValid*. Se hace uso de SVM para clasificar los términos en violentos y no violentos. Además se realiza un análisis de los resultados obtenidos por las técnicas de minería utilizadas.

En el **Cuarto Capítulo** se encuentran las conclusiones sobre el trabajo y los futuros estudios relacionados que se podrían realizar.

Al final del documento se encuentra el Anexo, donde se muestran los códigos utilizados en el desarrollo del trabajo.

En el DVD se incluyen los videos de las conversaciones, los audios de los videos, así como los textos de las conversaciones clasificados en violentos y no violentos. También se agregan los códigos y los archivos utilizados en el desarrollo del trabajo.

Capítulo 2: Marco teórico

En este capítulo se pretende dar a conocer la teoría conceptual en la que se fundamenta el proyecto de investigación realizado. Se explica que es la herramienta R y la utilidad que tiene en distintos campos del conocimiento. También se habla brevemente sobre la importancia de realizar el pre procesamiento, y se explica en qué consiste el agrupamiento (*clustering*), dando una breve descripción de las técnicas de agrupamiento: Jerárquica (*Hierarchical*), K-means, PAM y SVM (*Support Vector Machine*).

2.1. ¿Qué es R?

La herramienta R, es un lenguaje de programación principalmente orientada al análisis estadístico y visualización de información cuantitativa y cualitativa [7].

R es un proyecto de software libre, resultado de la implementación GNU del lenguaje S. R y S-Plus (versión comercial de S) probablemente son los lenguajes que más se usan en investigación por la comunidad estadística. Se distribuye bajo la licencia GNU GPL y está disponible para los sistemas operativos Windows, Macintosh, Unix y GNU/Linux.

R proporciona un amplio abanico de herramientas estadísticas entre ellos: modelos lineales y no lineales, pruebas estadísticas, análisis de series temporales, algoritmos de clasificación y agrupamiento y gráficas.

R hereda de S su orientación a objetos. Además, R puede integrarse con distintas bases de datos y existen bibliotecas que facilitan su utilización desde lenguajes de programación interpretados como Perl y Python.

Una característica esencial de R es su capacidad gráfica, que permite generar gráficos con alta calidad. Posee su propio formato para la documentación basado en LaTeX.

R también puede usarse como herramienta de cálculo numérico, campo en el que puede ser tan eficaz como otras herramientas específicas tales como “GNU Octave” y su equivalente comercial “MATLAB”.

2.2. Pre-procesamiento

Los datos a ser analizados mediante técnicas de minería de datos pueden ser incompletos (que carecen de ciertos atributos de interés), ruidosos (que contiene errores, o valores atípicos que se desvían de lo esperado), e inconsistentes (que contienen ciertas discrepancias) [3] [16].

La calidad de los datos afecta a los resultados de la minería de datos. Con el fin de ayudar a mejorar la calidad de los datos y, en consecuencia, de los resultados de minería de datos se realiza un pre procesamiento con el fin de mejorar la eficiencia y la facilidad del proceso minero. El pre procesamiento de datos es uno de los pasos más críticos en un proceso de minería de datos que se ocupa de la preparación y la transformación del conjunto de datos inicial [3] [16].

El pre-procesamiento consiste en extraer las palabras utilizadas en un documento, o segmentar el texto en distintas formas gráficas. Una forma gráfica se define como una secuencia de caracteres no delimitadores (en general, letras), comprendida entre dos caracteres delimitadores (espacios o signos de puntuación) [3].

El pre-procesamiento incluye la eliminación de los signos de puntuación y la extracción de las palabras separadas entre sí por espacios en blanco o signos de puntuación [3].

2.3. Agrupamiento (*Clustering*)

La agrupación puede considerarse como el problema más importante de aprendizaje no supervisado, de modo que, como cualquier otro problema de este tipo, se trata de encontrar una estructura en un conjunto de datos sin etiquetar. Una definición amplia de la agrupación podría ser "el proceso de organización de objetos en grupos cuyos miembros son similares de alguna manera". Por tanto, un clúster es una colección de objetos que son "similares" entre ellos y son "muy diferentes" a los objetos pertenecientes a otros grupos.

En la Figura 2 podemos ilustrar esto con un ejemplo gráfico sencillo, donde se identifican fácilmente los 4 grupos en los que los datos se pueden dividir: dos o más objetos pertenecen al mismo grupo si están "cerca" de acuerdo a una determinada distancia (en este caso la distancia geométrica).

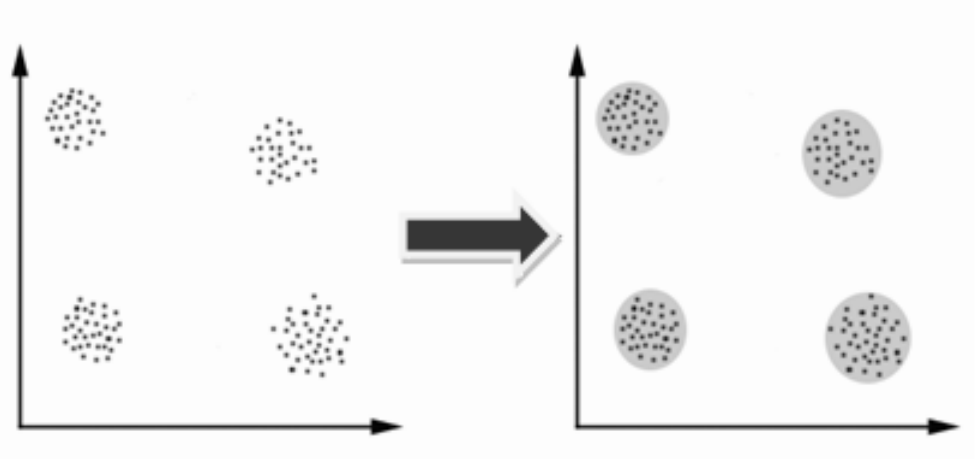

Figura 2: Ejemplo gráfico de agrupamiento

Los algoritmos de agrupamiento (*clustering*) permiten clasificar un conjunto de elementos de muestra en un determinado número de grupos basándose en las semejanzas y diferencias existentes entre los componentes de la muestra.

El proyecto de computación estadístico R cuenta con una amplia variedad de algoritmos de *clustering* disponibles. El paquete clValid contiene medidas para la validación de los resultados de un procedimiento de agrupamiento, proporcionando la trama, resumen y métodos adicionales para la visión y el resumen de las puntuaciones de validación y extracción de la agrupación de resultados para su posterior análisis [1].

A continuación tenemos una breve descripción de los métodos de agrupación que se utilizaron para este trabajo en particular:

Jerárquica (*Hierarchical*)

La agrupación jerárquica es un algoritmo de *clustering* que produce un dendrograma que se puede cortar a una altura elegida para producir el número deseado de grupos. Cada observación se coloca inicialmente en su propio grupo, y los grupos son unidos sucesivamente en el orden de su cercanía. La cercanía de cualquier grupo es determinado por una matriz de disimilitud, y puede basarse en una variedad de métodos de aglomeración. La agrupación jerárquica se incluye con la distribución de base de R en la función *hclust (),* y también se implementa en la función *agnes ()* en el paquete *cluster* [1] [15].

K-means

K-means es un método iterativo que minimiza la suma de cuadrados dentro de la clase para un determinado número de conglomerados. El algoritmo comienza con una estimación inicial para los centros de los conglomerados, y cada observación se coloca en el grupo más cercano. Los centros de los conglomerados a continuación, se actualizan, y todo el proceso se repite hasta que el clúster central ya no se mueve. A menudo, otro algoritmo de agrupamiento (por ejemplo, jerárquico) se ejecuta inicialmente para determinar los puntos de partida para los centros de los conglomerados. K-means se lleva a cabo en la función *KMeans (),* incluido con la distribución base de R [1] [15].

PAM

Particionamiento alrededor de medoides (PAM) es similar a K-means, pero se considera más sólida porque admite el uso de otras diferencias aparte de la distancia euclídea. Al igual que K-means, el número de grupos es determinado por adelantado, y no se requiere un conjunto inicial de los centros de conglomerados para iniciar el algoritmo. PAM está disponible en el paquete de clúster como *pam ()* [1] [15].

SVM (Support Vector Machine)

Una Máquina de Soporte Vectorial (SVM) aprende la superficie decisión de dos clases distintas de los puntos de entrada. Como un clasificador de una sola clase, la descripción dada por los datos de los vectores de soporte es capaz de formar una frontera de decisión alrededor del dominio de los datos de aprendizaje con muy poco o ningún conocimiento de los datos fuera de esta frontera. Los datos son mapeados por medio de un kernel Gaussiano u otro tipo de kernel a un espacio de características en un espacio dimensional más alto, donde se busca la máxima separación entre clases. Esta función de frontera, cuando es traída de regreso al espacio de entrada, puede separar los datos en todas las clases distintas, cada una formando un agrupamiento [9] [17].

"SVM se basa en ideas simples que se originaron en la teoría del aprendizaje estadístico (Vapnik 1998). La simplicidad viene del hecho de que las máquinas de vectores de soporte (SVMs) aplican un método lineal simple a los datos, pero en un espacio de características de alta dimensión no lineal relacionado con el espacio de entrada. Por otra parte, a pesar de que podemos pensar en SVM como un algoritmo lineal en un espacio de alta dimensión, en la práctica, no implica ningún cálculo en ese espacio de alta dimensión. Esta simplicidad combinada con el estado del arte del funcionamiento en muchos problemas de aprendizaje (clasificación, regresión, y de detección de la novedad) ha contribuido a la popularidad de la SVM" [9] [17] [18].

2.4. Validación del agrupamiento

En todo proceso de agrupamiento, la validación de la partición obtenida por los algoritmos utilizados es fundamental. Dado que no existe un algoritmo óptimo y que, generalmente, no se conoce el número de grupos que mejor se adapta a la estructura subyacente en los datos de entrada, la validación es primordial. La mayoría de los métodos basan su validación en el cálculo de valores numéricos denominados índices de validación del agrupamiento que se pueden clasificar, principalmente, en índices de validación externos e internos.

En este trabajo se hace uso de los índices de validación internos proporcionados por *clValid*: conectividad (*connectivity*), ancho de la silueta (*silhouette*), y el índice de Dunn (*Dunn*), los cuales estiman la bondad de la partición analizando la compactación y separación de los grupos generados [1]. El valor más pequeño indica un valor óptimo de la partición. Estos métodos suelen usarse cuando se buscan clústeres compactos y claramente separados. Para construir las siluetas se necesita una partición obtenida por la aplicación de algún algoritmo de clasificación y la matriz de proximidad que determine distancias entre los objetos involucrados. Para un clúster dado, este método asigna para cada objeto del mismo una medida cuantitativa, conocida como la anchura de la silueta. La anchura de la silueta indica la pertenencia del objeto en el clúster al que ha sido asignado.

Capítulo 3: Desarrollo

Este capítulo describe el desarrollo del trabajo de minería de texto utilizando la herramienta R. Primero recolectamos los videos con las conversaciones y se clasificaron en violentas y no violentas, se extrajo el audio y se decodificaron para convertirlas en texto. Posteriormente se procedió a guardar en archivos de texto todas las conversaciones. El texto de las conversaciones es pre procesado y luego transformado para construir una matriz de términos. Después de eso, las palabras y sus frecuencias son encontradas mostrándolas en graficas y una nube de palabras es utilizada para presentar las palabras importantes en las conversaciones. Por último se validaron los resultados de agrupamiento de los métodos de *clustering*: "Ward", "K-means" y "PAM" con *clValid*, y se hace uso de SVM para la clasificación.

3.1. Recolección de videos

Buscando y tomando de repositorios de videos de internet, se realizó la recolección de 100 videos en los formatos mp4[1] y wmv[2], en idioma español, que contienen conversaciones entre dos personas y se clasificaron manualmente en violentos y en no violentos (Ver Tablas 2 y 3).

Cabe mencionar que la recolección de los videos no fue una tarea sencilla, ya que se debía revisar cada video encontrado para percatarnos de que serviría para el trabajo a desarrollar, es decir, que tuviera las características que se estaban buscando, y además las descargas de los mismos se tornaron algo lentas.

Videos Recolectados	Cantidad
Violentos	53
No violentos	47
Totales	100

Tabla 2: Vídeos recolectados

Duración de los videos	Minutos
Mínima	1:12
Máxima	8:39
Promedio	1:25
Media	3:15

Tabla 3: Duración de los vídeos recolectados

[1] Es un formato estándar de Contenedor multimedia (audio, vídeo o ambos).
[2] Es un formato de compresión de video desarrollado por Microsoft.

3.2. Extracción de audio

Teniendo los videos almacenados y clasificados en violentos y en no violentos se utilizó la versión de prueba de "Adobe Premiere Pro CC"[5], y se procedió a realizar el proceso para extraer el audio (conversaciones) de los videos, obteniendo archivos en formato mp3.

La extracción del audio tomo de 5 a 10 minutos por cada video recolectado.

3.3. Procesamiento de sonido

Utilizando el paquete "Modelos de análisis de voz de Adobe en Español" [6] (complemento de Adobe Premiere Pro CC) se efectuó el proceso de decodificación del sonido de las grabaciones conversacionales obtenidas del audio de los videos, transformándolas en texto, y se guardaron en archivos con extensión txt.

En esta parte del trabajo existieron algunos problemas a la hora de procesar los archivos de audio para convertirlos a texto, ya que el programa utilizado para esta tarea tardaba entre 3 y 10 minutos para completar la conversión, si resultaba exitosa. Además la herramienta no fue lo suficientemente adecuada como para poder transformar a texto todas las conversaciones. Debido a ello se opto por transcribir manualmente algunas de las conversaciones que Adobe Premiere no pudo transformar a texto ya que algunas palabras no las reconocía poniendo símbolos en su lugar como: EE, $$, etc. (Ver Tabla 4).

Audios	Cantidad
Procesados por Adobe	83
Transcritos	17
Total	100

Tabla 4: Audios convertidos a texto

3.4. Pre-procesamiento de archivos

Para completar esta tarea se creó un archivo que contiene palabras vacías, carentes de significado, como son preposiciones, artículos y conjunciones, al cual llamamos "stopwords.txt" el cual contiene 617 términos. Además se removió manualmente el género de algunas palabras para obtener mejores resultados.

A continuación se describen los pasos utilizados para realizar el pre-procesamiento de los archivos que contienen las conversaciones violentas y de los que contienen las conversaciones no violentas:

- Cargar el archivo que contiene todas nuestras conversaciones.

- Construir el corpus.

- Llevar todo el texto de las conversaciones a minúsculas.

- Quitar espacios en blanco.

- Quitar signos de puntuación

- Cargar el archivo de palabras vacías personalizada (stopwords.txt).

- Remover palabras vacías genéricas (propias de R).

- Remover palabras vacías personalizadas (archivo "stopwords.txt" y demás términos carentes de significado identificados).

- Crear la matriz de términos.

En la Tabla 5 podemos observar las palabras que existen en cada paso.

Conversaciones	Cantidad de palabras (Términos) antes del Pre-procesamiento	Cantidad de palabras (Términos) después del Pre-procesamiento
No violentas	1971	1549
Violentas	1388	1133
Total	3359	2682

Tabla 5: Cantidad de términos antes y después del pre procesamiento

Utilizando la consola de R se ingresó el siguiente código para realizar los pasos descritos anteriormente para los archivos que contienen las conversaciones violentas y las conversaciones no violentas:

```
# Carga librerías
library(tm)
library(wordcloud)
# Lee el documento UTF-8 y lo convierte a ASCII
txt <-readLines("Conversaciones.txt",encoding="UTF-8")txt = iconv(txt,
to="ASCII//TRANSLIT")
# Construye un corpus
corpus <- Corpus(VectorSource(txt))
# Lleva a minúsculas
d  <- tm_map(corpus, tolower)
# Quita espacios en blanco
d  <- tm_map(d, stripWhitespace)
# Remueve la puntuación
d <- tm_map(d, removePunctuation)
# Carga mi archivo de palabras vacías personalizada y lo convierte a
ASCII
```

```
sw <- readLines("stopwords.txt",encoding="UTF-8")
sw = iconv(sw, to="ASCII//TRANSLIT")
# Remueve palabras vacías genéricas
d <- tm_map(d, removeWords, stopwords("spanish"))
# Remueve palabras vacías personalizadas
d <- tm_map(d, removeWords, sw)
# Crea matriz de términos
tdm <- TermDocumentMatrix(d)
```

3.4.1. Pre-procesamiento de conversaciones no violentas

Una vez ingresado el código para pre procesar el archivo que contiene las conversaciones no violentas, se procedió a determinar las cien palabras que más se repiten en nuestro archivo (Ver Figura 3), utilizando el siguiente código:

```
N <- 100
m <- as.matrix(tdm)
v <- sort(rowSums(m), decreasing=TRUE)
head(v, N)
```

casa	quiero	vas	quieres	miedo	crees	diciendo	digo	hombre	vida	dinero	favor	importa	marido	pienso	siento
25	23	22	17	14	11	11	11	11	10	9	9	9	8	8	8
entiendes	loc	persona	querias	querid	cansad	dijiste	dime	entiendo	forma	gusta	hij	hijs	mama	mira	pensar
7	7	7	7	7	6	6	6	6	6	6	6	6	6	6	6
pense	puedes	vete	dando	dices	digas	dios	escucha	gente	hiciste	iba	maldit	pasa	paso	ven	amig
6	6	6	5	5	5	5	5	5	5	5	5	5	5	5	4
cara	caso	chic	cobrar	diablos	entender	familia	fotos	hablando	hablar	hice	jugar	llevo	manana	morir	mujer
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
mundo	padre	pasar	problema	queda	queria	real	salir	senora	vale	vivir	aburrid	acabo	alguien	amor	apoyo
4	4	4	4	4	4	4	4	4	4	4	3	3	3	3	3
ayuda	bailar	bano	boca	buscando	cadaver	callate	carta	contrato	cosa	culpa	cumplir	das	debo	decirle	decirme
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
deja	dejado	dificil	diga												
3	3	3	3												

Figura 3: Las 100 palabras que más se repiten en conversaciones no violentas

En la Figura 4 vemos una gráfica de barras con los términos que se repiten más de 5 veces en las conversaciones no violentas.

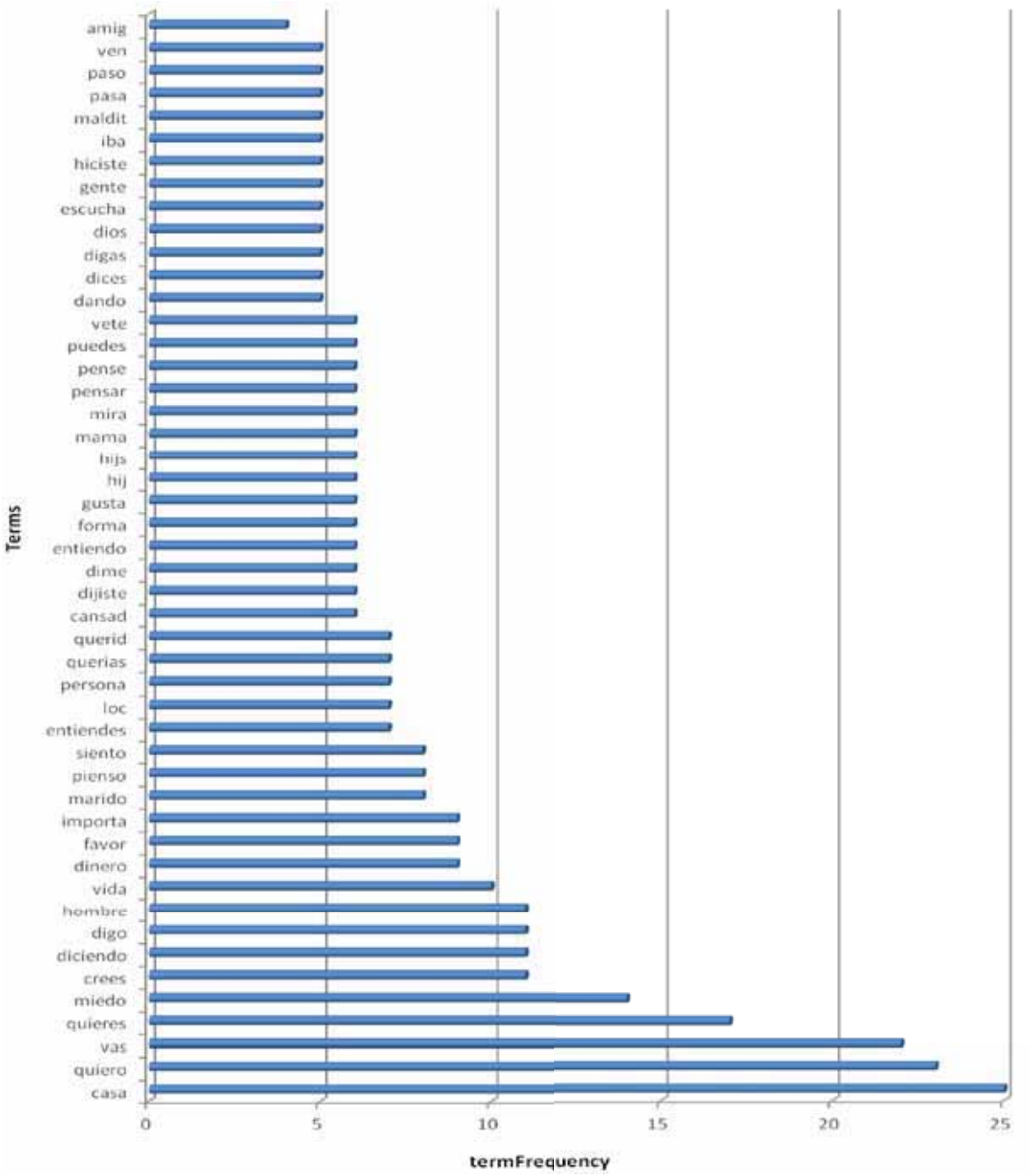

Figura 4: Gráfica de términos que se repiten más de 5 veces en conversaciones no violentas

A continuación se despliega la frecuencia en una nube de palabras usando el paquete wordcloud con el siguiente código (Ver Figura 5):

```
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
df <- data.frame(word = names(v),freq=v)
pal2 <- brewer.pal(8,"Dark2")
wordcloud(df$word,df$freq,min.freq=3 , colors=pal2)
```

Figura 5: Nube de palabras que más se repiten en conversaciones no violentas

Con esto nos damos una idea de las palabras que son utilizadas con más frecuencia en las conversaciones no violentas.

3.4.2. Pre-procesamiento de conversaciones violentas

Huna vez ingresado el código para pre procesar el archivo que contiene las conversaciones violentas, se procedió a determinar las cien palabras que más se repiten en nuestro archivo con las conversaciones violentas, utilizando el siguiente código (Ver Figura 6):

```
N <- 100
m <- as.matrix(tdm)
v <- sort(rowSums(m), decreasing=TRUE)
head(v, N)
```

maldit	pendej	dinero	favor	quieres	hij	quiero	vas	casa	put	madre	cabron	mierda
25	22	20	18	14	13	13	13	12	12	11	10	10
perr	coge	dijiste	importa	hey	pasa	viej	amig	crees	diciendo	digo	dije	estupid
10	9	9	9	8	8	8	7	7	7	7	7	7
habias	hablar	puedes	adios	callate	dame	familia	hablando	loc	mama	oye	papa	pinche
7	7	7	6	6	6	6	6	6	6	6	6	6
vale	vete	vida	amigs	cono	gente	herman	hombre	juro	mira	necesito	oficial	ofrece
6	6	6	5	5	5	5	5	5	5	5	5	5
pregunta	problema	problemas	ves	ahorita	alguien	camino	carajo	chinga	comer	culpa	dices	digas
5	5	5	5	4	4	4	4	4	4	4	4	4
doy	escuchar	facil	hagas	huy	idiota	imaginas	imbecil	mano	mire	mujer	palabra	pasando
4	4	4	4	4	4	4	4	4	4	4	4	4
perdon	personas	pudrete	senor	siquiera	toques	trasero	trato	amor	baja	borrach	calmate	cara
4	4	4	4	4	4	4	4	3	3	3	3	3
carcel	caso	cochin	come	cosa	cura	dejame	deje	depsraciad				
3	3	3	3	3	3	3	3	3				

Figura 6: Las 100 palabras que más se repiten en conversaciones violentas

En la Figura 7 vemos una gráfica de barras con los términos que se repiten más de 5 veces en las conversaciones violentas.

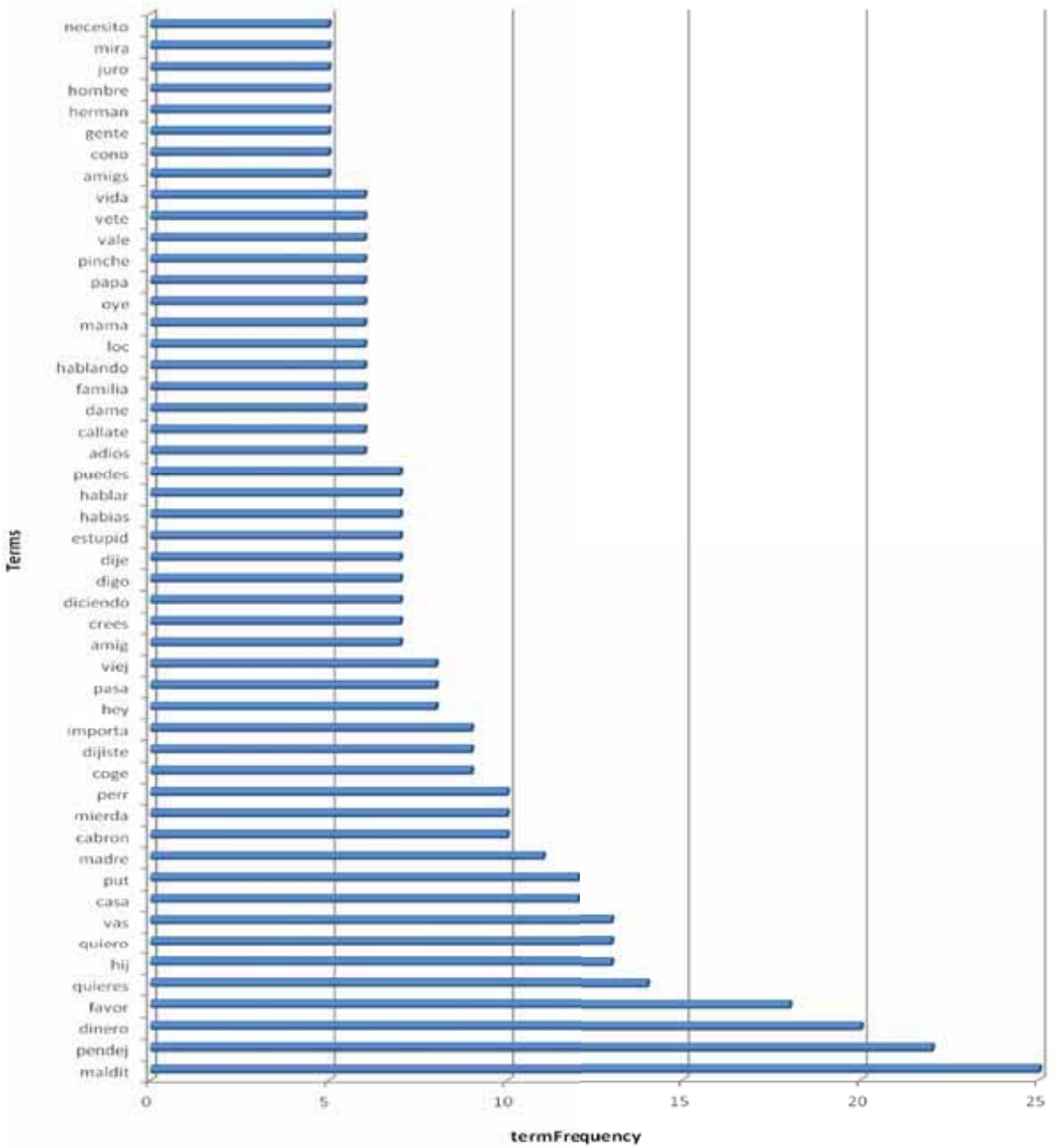

Figura 7: Gráfica de términos que se repiten más de 5 veces en conversaciones violentas

A continuación se despliega la frecuencia en una nube de palabras usando el paquete wordcloud con el siguiente código (Ver Figura 8):

```
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
df <- data.frame(word = names(v),freq=v)
pal2 <- brewer.pal(8,"Dark2")
wordcloud(df$word,df$freq,min.freq=3 , colors=pal2)
```

Figura 8: Nube de palabras que más se repiten en conversaciones violentas

Una vez mostrado lo anterior observamos que algunas palabras son utilizadas tanto en conversaciones violentas como en las no violentas, la gran diferencia radica en el que palabras van seguidas de otras palabras, para diferenciar si se trata de una conversación violenta o una no violenta podemos decir que al existir groserías y palabras fuertes hablamos de conversaciones violentas.

3.5. Agrupamiento (*Clustering*)

3.5.1. Validación de agrupamiento con el paquete clValid

Como se mencionó, el paquete clValid contiene medidas para la validación de los resultados de un procedimiento de agrupamiento y a continuación realizaremos el procedimiento para validar el mejor agrupamiento utilizando los métodos de *clustering*: "Ward", "K-means" y "PAM". Las medidas de validación utilizadas son: conectividad (*connectivity*), ancho de la silueta (*silhouette*), y el índice de Dunn (*Dunn*).

Una vez que se realizo el pre procesamiento quitando palabras sin sentido, palabras vacias y signos de puntuación en el archivo que contiene tanto nuestras conversaciones violentas como las no violentas, se carga el paquete clValid, como se muestra en el siguiente código:

```
library("clValid")
m2 <- as.matrix(tdm)
express <- m2
rownames(express) <- m2[, c(" ")]
intern <- clValid(express, 2:10, clMethods=c("hierarchical", "kmeans",
"pam"), validation = "internal")
summary(intern)
```

La Figura 9 muestra que la agrupación jerárquica (*hierarchical*) con 8 clústeres y con validación *Dunn* es la optima para nuestro archivo que contiene todas las conversaciones.

```
Clustering Methods:
 hierarchical kmeans pam

Cluster sizes:
 2 3 4 5 6 7 8 9 10

Validation Measures:
                                 2        3        4        5        6        7        8        9       10

hierarchical Connectivity    2.9290   7.7869  10.7159  17.8361  18.5028  21.4317  24.3607  25.3607  26.3607
             Dunn            0.4749   0.4362   0.4862   0.5049   0.5049   0.5133   0.5586   0.5586   0.5586
             Silhouette      0.8218   0.8211   0.8156   0.8068   0.8033   0.7810   0.7720   0.7714   0.7713
kmeans       Connectivity   14.9071  16.6933  16.8361  21.6940  27.5520  36.9048  37.5714  34.6425  35.6425
             Dunn            0.4309   0.3369   0.3369   0.2139   0.2013   0.1833   0.1833   0.2357   0.2357
             Silhouette      0.8148   0.8029   0.7810   0.7631   0.7541   0.6432   0.6431   0.6432   0.6431
pam          Connectivity   84.7679 115.2321 175.2448 214.1349 237.7694 260.3389 288.7841 302.1587 316.8964
             Dunn            0.0693   0.0693   0.0690   0.0735   0.0786   0.0786   0.0786   0.0786   0.0786
             Silhouette      0.1555   0.0569   0.0784   0.0960   0.1141   0.1292   0.1437   0.1580   0.1670

Optimal Scores:

             Score  Method       Clusters
Connectivity 2.9290 hierarchical 2
Dunn         0.5586 hierarchical 8
Silhouette   0.8218 hierarchical 2
```

Figura 9: Validación de agrupamiento con el paquete clValid

La Figura 10 nos muestra los diagramas correspondientes a los resultados arrojados por la validación de grupos con clValid, donde podemos observar las validaciones para cada técnica gráficamente.

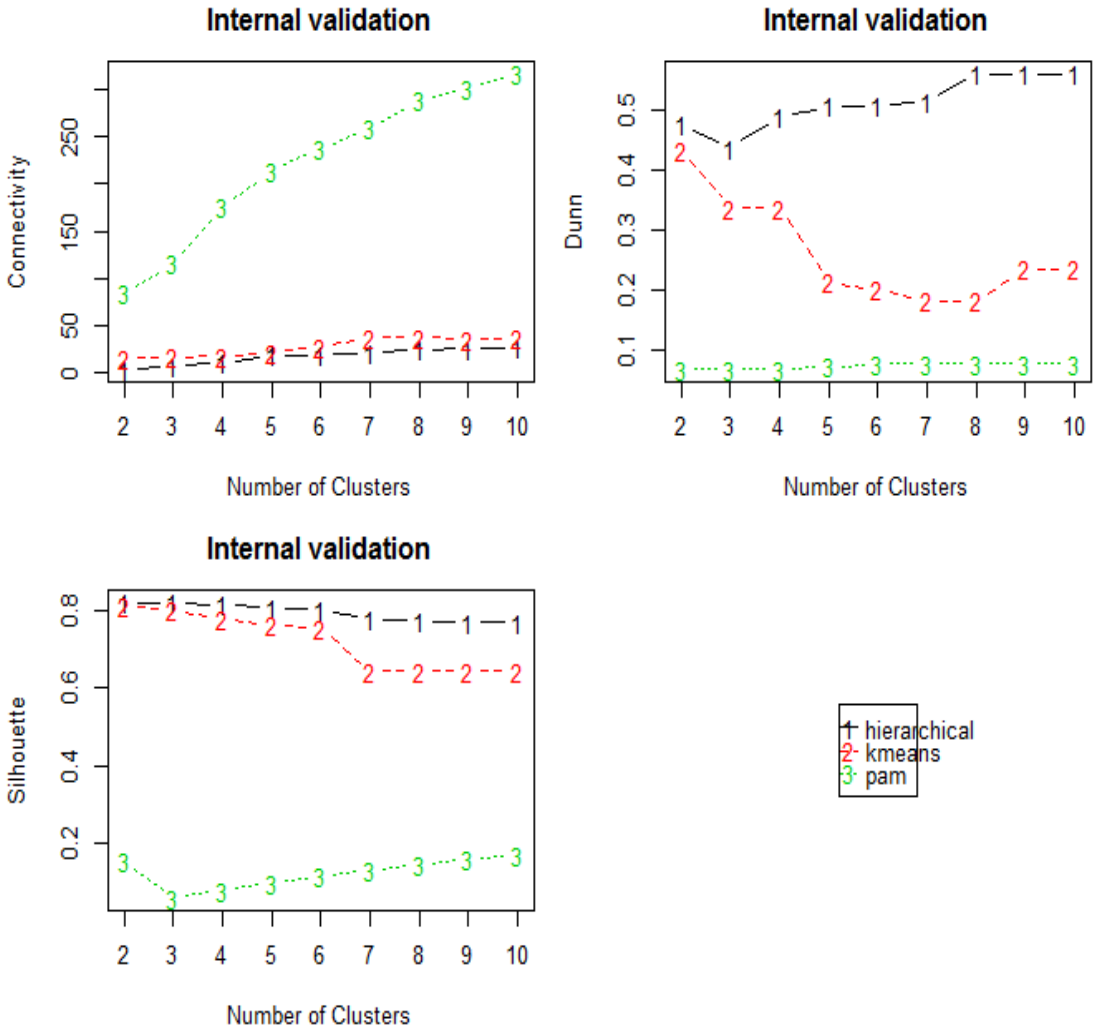

Figura 10: Diagramas de validación de agrupamiento con el paquete clValid

La validación de *clutering* realizada por clValid utiliza los índices de validación para estimar la bondad de la partición analizando la compactación y separación de los grupos generados. Una vez dicho lo anterior, sabemos que la validación óptima está dada por el valor más pequeño posible, lo cual nos indica que la técnica jerárquica (*hierarchical*) con 8 clústeres y validación *Dunn* es la óptima para este caso en particular.

3.5.2. Agrupamiento utilizando:"*Hierarchical* (Ward)", "K-means" y "PAM"

Aunque ya sabemos que mediante el método Jerárquico el número óptimo de las agrupaciones es ocho al haber realizado la validación de agrupamiento con clValid, realizaremos el agrupamiento utilizando:"*Hierarchical* (Ward)", "K-means" y "PAM", para poder comparar los resultados.

3.5.2.1. Agrupamiento con el Método Ward

Después de realizar la validación de agrupamiento utilizando el paquete clValid, sabemos que lo ideal es separar en 8 grupos, así que lo realizamos de esta forma y lo representamos en un Dendograma que es un tipo de representación gráfica o diagrama de datos en forma de árbol (Dendro=árbol) que organiza los datos en subcategorías que se van dividiendo en otros hasta llegar al nivel de detalle deseado.

El dendograma nos permite apreciar claramente las relaciones de agrupación entre los datos e incluso entre grupos de ellos aunque no las relaciones de similitud o cercanía entre categorías. Para ello utilizamos el código que se muestra en seguida:

```
# Aplicar clustering
distMatrix <- dist(scale(tdm))
fit <- hclust(distMatrix, method="ward")
plot(fit)
# Cortar el dendograma en 8 clústeres
rect.hclust(fit, k=8)
(groups <- cutree(fit, k=8))
```

La Figura 11 muestra el dendograma reducido que contiene solo algunas palabras para poder apreciar cómo se realizó la agrupación en 8 clústeres.

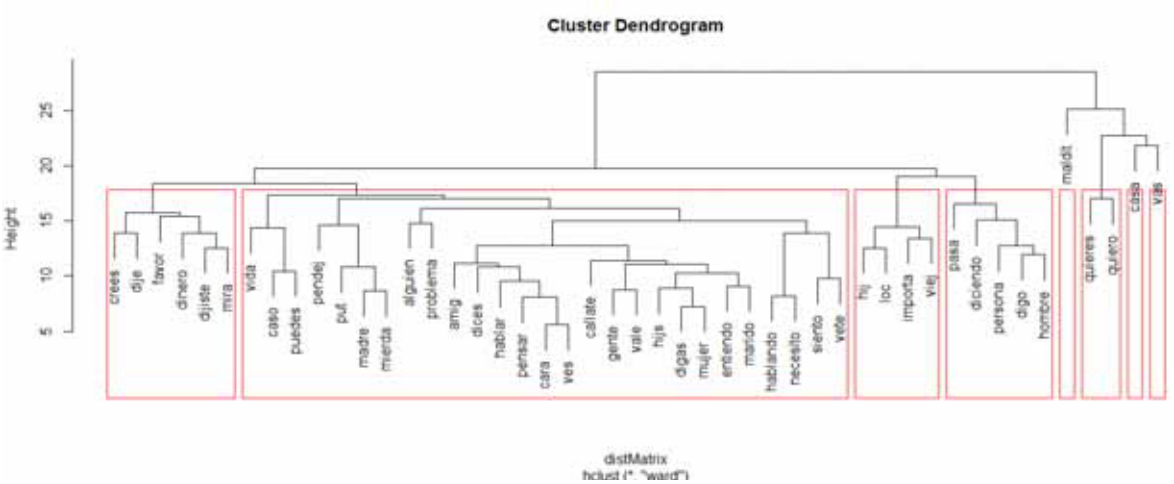

Figura 11: Dendograma en 8 grupos

En la Figura 12 podemos apreciar parte de las palabras y el grupo al que pertenecen utilizando el método Ward.

asalto	ascensor	asco	asesine	asesino	asevera	aspecto	asqueroso	astronauta
1	5	1	1	2	1	3	1	2
asturias	asumiran	asunto	asuntos	asustan	ataque	atencion	atrapa	atreve
2	1	1	1	2	1	2	1	3
atun	aumente	autor	autoridad	autoridades	autos	auxilio	avanza	avaricia
1	3	5	1	5	1	3	1	1
avenida	averiada	avion	aviones	avises	aviso	ayuda	ayudar	ayudo
1	2	1	1	3	3	3	2	1
azules	babosas	bailar	baja	bajale	bajan	bajar	baje	bajito
6	1	1	3	2	5	5	5	5
bajos	balboa	baleares	banca	bancos	bando	bano	barata	barbarie
5	7	5	5	1	7	3	1	3
barca	barcelona	basta	bastardo	basura	bazar	bebes	bebia	bebido
1	1	2	3	2	5	1	1	1
becas	bejarano	bendicion	benditos	benjamin	besa	bestia	bitacoras	bledo
1	1	2	5	2	1	2	1	2
bloqueo	bloqueos	bobi	boca	bonita	bono	bordes	borracho	brillando
1	5	3	3	2	3	7	1	6
broncas	brotes	brujerias	brutos	buenota	buesa	buque	buscando	buscar
1	5	1	1	3	5	1	2	1
buscarte	buscas	busco	bye	caballero	cabana	cabeza	cabron	cabrones
2	2	1	1	2	1	1	3	2
cadaver	cae	caen	caer	caga	caldera	calentura	calienta	calientito
3	7	1	5	2	2	1	1	1
callar	callas	callate	calle	calles	callese	calmate	camara	cambia
2	3	3	3	1	2	1	1	1
cambiar	cambiaras	cambiarme	cambiaste	cambie	cambies	cambio	cameo	camino
2	3	1	2	2	1	1	1	1
camioneta	campeon	campestre	campo	campus	canasta	cancer	candidato	cansada
1	1	1	2	7	5	2	2	2
cansado	cansas	cantaros	canto	caos	capacidad	capaz	captar	cara
1	1	1	1	1	3	2	5	2
caracterizan	carajo	carcel	carceles	cargar	cargo	caridad	carlos	carnal
1	1	2	1	1	5	1	1	3
carne	carolina	carrera	carretera	carrizal	carta	caruana	casa	casada
2	1	1	5	1	1	1	3	1
casar	casarme	casarte	caso	casos	castellano	catedratico	causa	celda
1	1	1	3	1	2	5	1	3

Figura 12: Agrupamiento de palabras con el método Ward

3.5.2.2. Agrupamiento con el algoritmo de K – Means

Aplicando el algoritmo de K-Means utilizamos el siguiente código para realizar la agrupación en 8 grupos (Ver Figura 13):

```
# Transpone la matriz
tdm2 <- t(tdm)
# Establecer una semilla aleatoria fija
set.seed(122)

# K-means clustering
k <- 8
kmeansResult <- kmeans(tdm2, k)
round(kmeansResult$centers, digits=10)
```

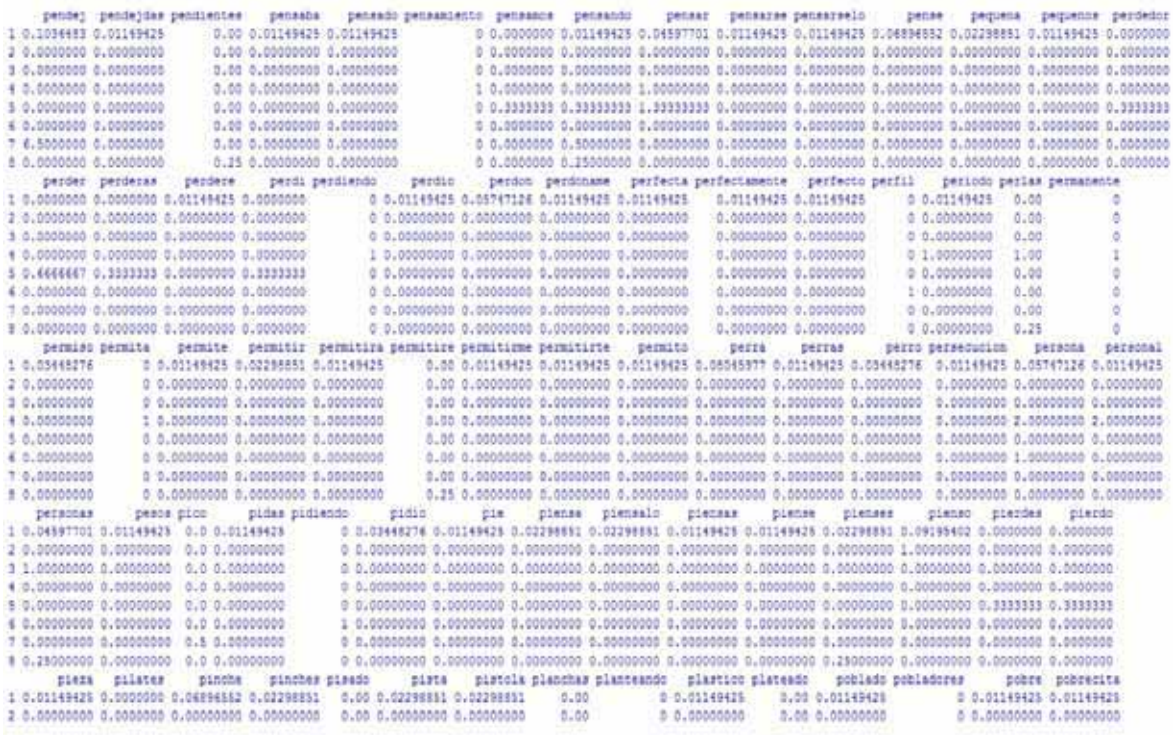

Figura 13: Agrupamiento de palabras con el algoritmo de K-Means

Posterior a esto y utilizando el código que está a continuación, comprobamos las principales palabras que aparecen frecuentemente en las conversaciones, generadas con 3, 4 y 5 palabras en cada clúster con el algoritmo de K-means (Ver Figuras 14-16):

```
for (i in 1:k) {
cat(paste("cluster ", i, ": ", sep=""))
s <- sort(kmeansResult$centers[i,], decreasing=T)
cat(names(s)[1:5], "\n")
# Imprime palabras de cada grupo
# print(rdmConversaciones[which(kmeansResult$cluster==i)])
}
```

```
cluster 1: casa maldit quiero
cluster 2: coge pregunta put
cluster 3: dinero favor vas
cluster 4: cobrar diciendo jugar
cluster 5: quieres quiero miedo
cluster 6: insulto cuarenta desverguenza
cluster 7: pendej estupid adios
cluster 8: vas crees mama
```

Figura 14: Principales 3 palabras en cada clúster usando el algoritmo de K-Means

```
cluster 1: casa maldit quiero favor
cluster 2: coge pregunta put favor
cluster 3: dinero favor vas dame
cluster 4: cobrar diciendo jugar escucha
cluster 5: quieres quiero miedo facil
cluster 6: insulto cuarenta desverguenza diputdo
cluster 7: pendej estupid adios cono
cluster 8: vas crees mama dijiste
```

Figura 15: Principales 4 palabras en cada clúster usando el algoritmo de K-Means

```
cluster 1: casa maldit quiero favor hij
cluster 2: coge pregunta put favor tema
cluster 3: dinero favor vas dame herencia
cluster 4: cobrar diciendo jugar escucha apoyo
cluster 5: quieres quiero miedo facil vida
cluster 6: insulto cuarenta desverguenza diputdo expresion
cluster 7: pendej estupid  adios cono mierda
cluster 8: vas crees mama dijiste miedo
```

Figura 16: Principales 5 palabras en cada clúster usando el algoritmo de K-Means

Una vez que tenemos los agrupamientos generados, se puede observar que en cada clúster se encuentran las palabras que aparecen frecuentemente juntas en las conversaciones, es decir, cada clúster contiene exclusivamente palabras que son utilizadas comúnmente juntas en una conversación. Por ejemplo en los clústeres 1, 2 y 7 se nota que muy probablemente se encuentran las palabras que se utilizan en una conversación violenta, en cambio en los demás clústeres es muy probable que se trata de palabras que se utilizan en una no violenta.

3.5.2.3. Agrupamiento con el algoritmo de K – Medoids

Con el algoritmo de K-Medoids que es similar a K-means, pero se considera más sólido porque admite el uso de otras diferencias aparte de la distancia euclídea, realizamos el agrupamiento utilizando el siguiente código (Ver Figura 17):

```
# K-medoids clustering
library(fpc)
# Repartición con la estimación de número de grupos
pamResult <- pamk(tdm2, metric="manhattan")
# Número de grupos identificados
(k <- pamResult$nc)
pamResult <- pamResult$pamobject
# Imprime grupo de medoids
```

```
for (i in 1:k) {
cat(paste("cluster", i, ": "))
cat(colnames(pamResult$medoids)[which(pamResult$medoids[i,]==1)], "\n")
# Imprime palabras de cada grupo
# print(rdmConversaciones [pamResult$clustering==i])
}
```

```
cluster 1 : casa
cluster 2 :
cluster 3 : ido intrusos mantas sancionara
cluster 4 : usted
cluster 5 : favor
cluster 6 : quieres
cluster 7 : entonces
cluster 8 : apoco buen cabron ejemplo hablas haces hagas put sobrin trabajar
cluster 9 : cabana imbecil largate mal maldit  nacido oledor vete
cluster 10 : das dormir miedo puedo traeme vamos voy
```

Figura 17: Clústeres con las palabras arrojadas por el algoritmo de K- Medoids

Observando los clústeres generados por el algoritmo de K-medoids, se puede intuir que los clústeres 1-7 contienen las palabras que se utilizan frecuentemente tanto en conversaciones violentas como en no violentas. En los clústeres 8 y 9 es muy probable que se encuentren las palabras que más se utilizan en las conversaciones violentas, y en el clúster 10 todas las palabras que frecuentemente son utilizadas en conversaciones no violentas.

3.6. Clasificación con SVM (*Support Vector Machine*)

3.6.1. Usando el paquete e1071 para la clasificación SVM

Para clasificar las palabras con SVM utilizamos el paquete "e1071" de R [9], y se ingresó el siguiente código, obteniendo la clasificación de cada palabra de nuestro archivo de texto que contiene todas las conversaciones recolectadas (Ver Figura 18):

```
library("e1071")
model <- svm( ~.,data = m2, method = "C-classification", kernel =
"radial",cost = 10, gamma = 10)
predict(model)
```

sirven	sistema	sitio	situacion	sobrevivir	sobrino	social	sociedad	sola
FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
solamente	solia	solicito	solidas	solita	solo	sombras	sombrero	sometida
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
sometidas	sonido	sonora	sonreir	sonrieron	soplon	soporta	soportan	soportar
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
soportarlo	sorpresa	sospechoso	sostengo	stripper	subestimar	subi	subia	subir
TRUE	FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
subsecretario	sucede	suceder	sucedera	sucediera	sucia	sucursal	suelo	suelos
TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE
suelta	sueltalo	sueltame	suelteme	suerte	suficiente	suficientes	sufrir	sujeto
FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
sumado	superar	supieran	supiste	supone	supuesta	supuesto	sustentada	tabula
TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	FALSE
taison	tal	talones	tamano	tambien	tampoco	tan	tanta	tarada
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
tardar	tarde	tardes	tecnicismo	telesur	televisa	television	tema	teme
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
temor	temporada	temprano	ten	tendra	tendras	tenebrosos	teneis	tener
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
tenerte	tenia	tenias	tenis	tensa	teoria	terminado	terminar	termine
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
termino	tiemblo	tiempo	tiempos	tienda	tierra	tijuana	timbre	tio
TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE
tipo	tira	tiran	tiro	titular	titulo	tocaba	tocado	tocando
FALSE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
tocandote	tocar	toco	toda	todas	todavia	tolerar	tomando	tomar
TRUE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE
tomas	tomes	tono	tonta	tonto	toque	toques	trabajado	trabajan
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
trabajando	trabajar	trabajas	trabajo	trae	traeme	traficante	tragar	tragas
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
traicion	traido	trailer	traje	trampa	tranquila	tranquiliza	tranquilo	tranquilos
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
transcurre	tras	trasero	trata	trataba	tratamiento	tratando	tratas	trates
TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE
trato	trenes	tres	tristan	trono	trujillo	tsunami	tubo	tuco
TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE

Figura 18: Clasificación SVM utilizando el paquete "e1071"

Para tener un mejor control de todos los términos, se guardó la matriz de términos y cantidades en cada conversación con el siguiente comando:

```
write.csv(m2, file="datanueva_e1071.csv") #guardamos en un archivo CSV.
```

Con la ayuda un software de hoja de cálculo sacamos el porcentaje de cada término y se evaluó que si el término aparecía más del 50% en las conversaciones No violentas (Conversaciones de la 1 a la 47) se clasificaría como No violenta (N). De la misma forma, si el término aparecía más del 50% en las conversaciones Violentas (Conversaciones de la 48 a la 100) se clasificaría como Violenta (V). Y si el porcentaje del término era el mismo (50%=NV=V=50%) indicaría que no tienen clasificación (SIN)

Hecho lo anterior se guardaron los datos en un documento de texto que contiene: términos (Terms), porcentajes en conversaciones no violentas (NV), porcentajes en conversaciones violentas (SV) y la clasificación (Clas). La clase puede ser "N", para los términos no violentos, "V" para los violentos y "SIN" sin clasificación.

En el siguiente código utilizado, el tercer comando, el parámetro "Clas ~." indica el atributo (columna) del conjunto de datos para ser utilizado como clases de instancia.

```
library(tm)
library(wordcloud)
library(e1071)

dataset <- read.table("Terminos.txt",header=TRUE)

model  <- svm(Clas~., data = dataset)
summary(model)
```

En la Figura 19 se observa el resultado del comando *summary()*, donde se nota el número de SVMs, el número de clases y los niveles:

```
Call:
svm(formula = Clas ~ ., data = dataset)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.0004363002

Number of Support Vectors:  1118

 ( 517 490 111 )


Number of Classes:  3

Levels:
 NV SIN V
```

Figura 19: Resultado de SVMs, clases y niveles

Se dividió el conjunto de datos “dataset” en un tren y un conjunto de pruebas, mezclando los datos de 10 formas diferentes, realizando 10 pruebas, utilizando el siguiente código:

```
index <- 1:nrow(dataset)

testindex <- sample(index, trunc(length(index)/3))

testset <- dataset[testindex,]
```

```
trainset <- dataset[-testindex,]
```

Corrimos de nuevo el modelo usando el tren de datos para predecir las clases usando el conjunto de prueba con el fin de verificar si el modelo tiene buena generalización.

```
model <- svm(Clas~., data = trainset)

prediction <- predict(model, testset[,-4])
```

El -4 se debe a que la variable dependiente, Clas, se encuentra en la columna número 4.

Utilizamos una tabulación cruzada, ingresando el código que se muestra a continuación, obteniendo las matrices para cada una de las 10 pruebas realizadas.

```
tab <- table(pred = prediction, true = testset[,4])

tab
```

Las Figuras 20-22 muestran las matrices resultantes más significativas de las pruebas realizadas.

```
      true
pred   NV SIN   V
  NV  398  28  60
  SIN   0   0   0
  V    56   3 218
```

Figura 20: Matriz de confusión usando SVM (prueba 1)

La interpretación de la Figura 20, la cual corresponde a la prueba 1, es la siguiente: se observa que hay 454 términos no violentos en el conjunto de prueba de los cuales 398 fueron predichos como no violentos y 56 como violentos. También se cuenta con 31 términos sin clasificar de los cuales 28 se predijeron como no violentos y 3 como violentos. Por otro lado hay 278 términos violentos en el conjunto de prueba, de los cuales 60 se predijeron como no violentos y 218 como violentos.

Las Figuras 21 y 22 las podemos interpretar de manera similar que la figura 20.

```
          true
pred       NV SIN   V
  NV      430  39 294
  SIN       0   0   0
  V         0   0   0
```

```
      true                  true
pred   NV SIN   V  pred   NV SIN   V
  NV  426  51 286    NV  448  49 266
  SIN   0   0   0    SIN   0   0   0
  V     0   0   0    V     0   0   0
```

Figura 21: Matrices de confusión usando SVM (pruebas 3- 5)

```
      true                  true
pred   NV SIN   V  pred   NV SIN   V
  NV  422  37   0    NV  460  32   0
  SIN   0   0   0    SIN   0   0   0
  V     0   0 304    V     0   0 271
```

```
          true
pred       NV SIN   V
  NV      453  33   0
  SIN       0   0   0
  V         0   0 277
```

Figura 22: Matrices de confusión usando SVM (pruebas 6, 8 y 10)

De estos resultados podemos concluir que las pruebas 3-5 representadas por la Figura 21 obtuvieron una predicción más errónea, en contraste con las matrices de confusión representadas por la Figura 22, que son las que realizan mejores predicciones con respecto al conjunto de prueba.

3.7. Análisis de resultados

A continuación se muestran por medio de tablas y graficas, los resultados obtenidos, donde se puede apreciar datos relevantes sobre el pre procesamiento de las conversaciones violentas y no violentas y sobre las técnicas de minería utilizadas.

En la Tabla 6 y en la Figura 23 representada por una gráfica podemos evaluar la cantidad de términos antes y después del pre procesamiento de las conversaciones dándonos cuenta que existen muchas palabras que son omitidas para poder obtener mejores resultados a la hora de clasificar alguna conversación.

	Cantidad de conversaciones analizadas	Cantidad de palabras (Términos) antes del Pre-procesamiento	Cantidad de palabras (Términos) después del Pre-procesamiento
No violentas	47	1971	1549
Violentas	53	1388	1133
Total	100	3359	2682

Tabla 6: Datos de las matrices de términos, conversaciones violentas vs. conversaciones no violentas

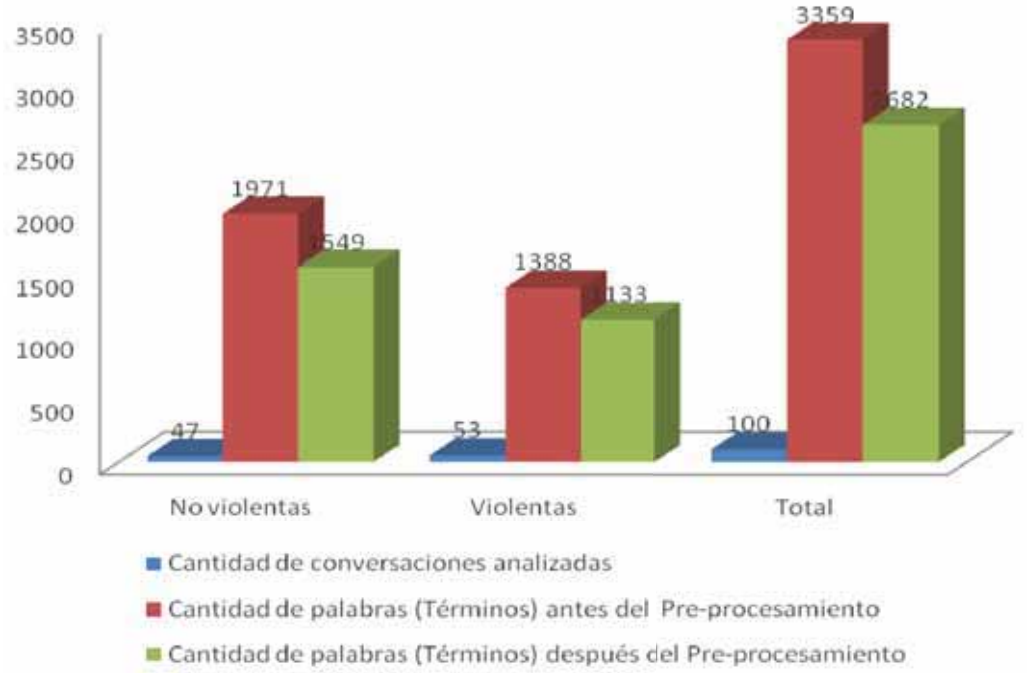

Figura 23: Gráfica de conversaciones violentas vs. Conversaciones no violentas

En la Tabla 7 vemos los términos que más se repiten en las conversaciones violentas y en las no violentas, notando que existen palabras que se repiten en los dos tipos de conversaciones como: “Quieres”, entre otras.

Conversaciones	Término	Número de Repeticiones
No violentas	Casa	25
	Quiero	23
	Vas	22
	Quieres	17
	Miedo	14
Violentas	Maldit@	25
	Pendej@	22
	Dinero	20
	Favor	18
	Quieres	14

Tabla 7: Términos que más se repiten, conversaciones violentas vs. conversaciones no violentas

En la Tabla 8 podemos apreciar la validación de agrupamiento entre 2 y 10 clústeres realizada con el paquete clValid donde podemos comparar los resultados entre las técnicas de agrupación: “Ward”, “K-means” y “PAM”, y las medidas de validación: conectividad (*connectivity*), ancho de la silueta (*silhouette*), e índice de Dunn (*Dunn*).

		2	3	4	5	6	7	8	9	10
hierarchical	Connectivity	2.929	7.7869	10.7159	17.8361	18.5028	21.4317	24.3607	25.3607	26.3607
	Dunn	0.4749	0.4362	0.4862	0.5049	0.5049	0.5133	0.5586	0.5586	0.5586
	Silhouette	0.8218	0.8211	0.8156	0.8068	0.8033	0.781	0.772	0.7714	0.7713
kmeans	Connectivity	14.9071	16.6933	16.8361	21.694	27.552	36.9048	37.5714	34.6425	35.6425
	Dunn	0.4309	0.3369	0.3369	0.2139	0.2013	0.1833	0.1833	0.2357	0.2357
	Silhouette	0.8148	0.8029	0.781	0.7631	0.7541	0.6432	0.6431	0.6432	0.6431
pam	Connectivity	84.7679	115.2321	175.2448	214.1349	237.7694	260.3389	288.7841	302.1587	316.8964
	Dunn	0.0693	0.0693	0.069	0.0735	0.0786	0.0786	0.0786	0.0786	0.0786
	Silhouette	0.1555	0.0569	0.0784	0.096	0.1141	0.1292	0.1437	0.158	0.167

Tabla 8: Comparación de resultados entre las técnicas de agrupación: “Ward”, “K-means” y “PAM”

Analizando los resultados de la Tabla 8 tenemos un panorama más amplio para saber cuál técnica de agrupación es más eficiente, ya que mientras los valores sean más pequeños mejor se realiza la agrupación.

En la Tabla 9 y en la Figura 24, de acuerdo a la validación de resultados realizada, se observa que mientras la agrupación jerárquica (*hierarchical*) es la mejor en cada caso, la óptima es con la validación *Dunn* para 8 clústeres.

	Score	Método	Clústeres
Connectivity	2.9290	hierarchical	2
Dunn	0.5586	hierarchical	8
Silhouette	0.8218	hierarchical	2

Tabla 9: Resultado óptimo para agrupamiento

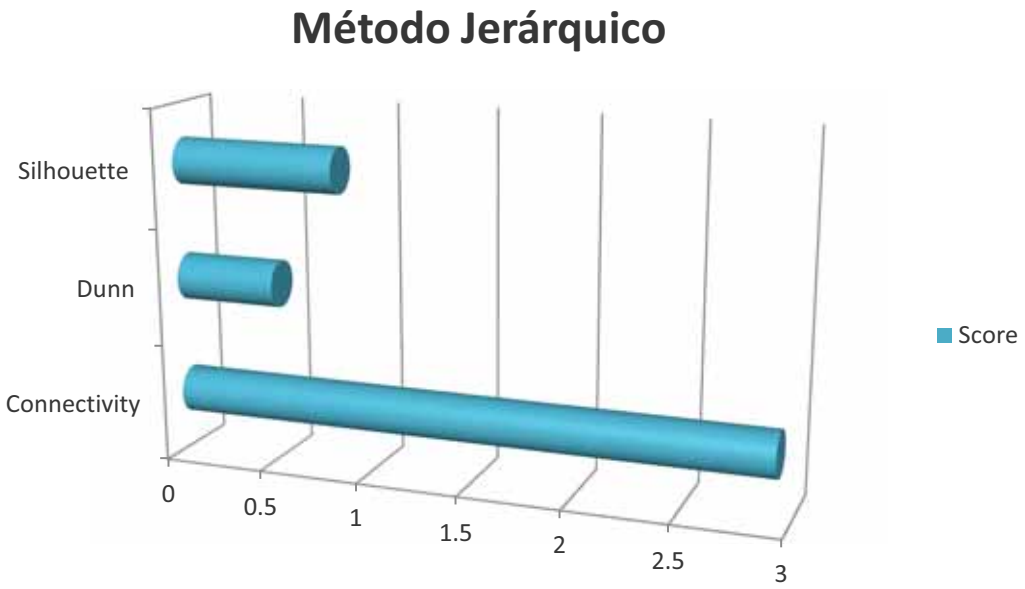

Figura 24: Gráfica de resultados para el método jerárquico

Para clasificar las palabras con SVM utilizamos el paquete "e1071" como ya se indicó anteriormente. Apoyándonos de una herramienta de hoja de cálculo, se calculó el porcentaje de cada término, evaluando de la siguiente manera:

- Si el término aparecía más del 50% en las conversaciones No violentas (Conversaciones de la 1 a la 47) se clasificaría como No violenta (N).

- Si el término aparecía más del 50% en las conversaciones Violentas (Conversaciones de la 48 a la 100) se clasificaría como Violenta (V).

- Si el término aprecia 50% en las conversaciones No violentas (Conversaciones de la 1 a la 47) y 50% en las conversaciones Violentas (Conversaciones de la 48 a la 100) no tiene clasificación (SIN).

Se utilizó la función *tune()* para hacer una búsqueda del mejor rendimiento durante los intervalos de parámetros suministrados (C - costo, γ- Gamma), utilizando el juego de trenes. Ingresado el siguiente código, se obtuvo que el mejor parámetro para gamma y el costo son: 0.1 y 10 respectivamente, obteniendo el mejor rendimiento en 0.006557448 como lo muestra la Figura 25.

```
tuned <- tune.svm(Clas~., data = trainset, gamma = 10^(-6:-1), cost =
10^(1:2))

summary(tuned)
```

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 gamma cost
   0.1   10

- best performance: 0.006557448

- Detailed performance results:
   gamma cost       error  dispersion
1  1e-06   10 0.411915205 0.023312699
2  1e-05   10 0.411915205 0.023312699
3  1e-04   10 0.051079291 0.016249072
4  1e-03   10 0.055658755 0.026580375
5  1e-02   10 0.008518232 0.005395479
6  1e-01   10 0.006557448 0.005360058
7  1e-06  100 0.411915205 0.023312699
8  1e-05  100 0.051079291 0.016249072
9  1e-04  100 0.055658755 0.026580375
10 1e-03  100 0.008518232 0.005395479
11 1e-02  100 0.006557448 0.005360058
12 1e-01  100 0.006557448 0.005360058
```

Figura 25: Mejores parámetros y mejor rendimiento con SVM

Capítulo 4: Trabajos futuros y conclusiones

4.1. Trabajos Futuros

A partir de lo realizado y como ya se había mencionado, la importancia de este trabajo recae en el avance que aporta, ya que nos da un panorama más amplio y nos ayuda a imaginar cómo podrían llegar a ser los sistemas de vigilancia en el futuro.

Apoyándose de este trabajo, se podría experimentar con un número más amplio de videos, por decir el doble o triple de los que se usaron, así como realizar un pre procesamiento más exhaustivo donde se incluya el uso de sinónimos, quitar el género de todas las palabras que lo ameriten, y hacer uso de técnicas de minería de datos diferentes a las utilizadas aquí, para de esta manera obtener otros y mejores resultados, ya que existiría una cantidad más amplia de términos con los cuales experimentar, y al final poder comparar los resultados obtenidos con los de este trabajo.

También sería conveniente realizar experimentos con videos de diferentes clasificaciones, por ejemplo, políticos, deportivos, familiares, comerciales, escolares, etc., para posteriormente realizar la clasificación en violentos y no violentos.

Se podría experimentar con videos en otros idiomas, por ejemplo inglés que es uno de los más utilizados en el mundo.

A partir de todos los experimentos realizados se podría diseñar un sistema computacional que clasifique automáticamente las conversaciones en violentas y no violentas, a partir de ahí podría ir evolucionando hasta que el sistema clasifique los videos en tiempo real y emita señales para indicar cuando una conversación se torne violenta.

De esta manera cualquier área de vigilancia podría instalar un sistema de monitoreo automático que clasifique cualquier conversación (política, deportiva, familiar, comercial, escolar, etc.) en violenta y no violenta a partir de un video tomado en tiempo real, que les advierta con una alarma cuando se susciten conversaciones con una determinada probabilidad de ser violentas, y de esta manera notificar a tiempo a las personas encargadas de la vigilancia.

De esta forma el personal encargado, logrará llegar al lugar de inmediato, y así conseguir evitar situaciones más complejas que podrían llegar a desencadenar en violencia física.

4.2. Conclusiones

Concretamente, como resultado de los experimentos realizados, el presente trabajo aporta un avance para poder clasificar conversaciones en violentas y no violentas mediante palabras.

Realizando el pre procesamiento de las conversaciones, se logró obtener las palabras que más se utilizan para conversaciones violentas y para conversaciones no violentas, de tal forma que con las gráficas de términos y las nubes de palabras tenemos una idea más clara del tipo de palabras y la frecuencia con que son utilizadas en cada tipo de conversación. Se observó que tanto en las conversaciones violentas como en las no violentas existen términos que son utilizados frecuentemente de la misma forma, aunque la diferencia radica en que en las conversaciones violentas estas palabras son utilizadas frecuentemente junto a groserías o palabras fuertes.

Una vez realizados los experimentos de agrupación de palabras se lograron resultados satisfactorios, obteniendo ocho agrupamientos de palabras que aparecen frecuentemente juntas en cada conversación, ya sea violenta o no violenta, y aunque no todas las técnicas utilizadas para agrupar los términos obtuvieron buenos resultados, la técnica jerárquica "*ward*" fue más eficiente en este trabajo, ya que la cercanía de sus palabras fue mucho mejor a las de *k-means* y *PAM*. En la Tabla 8 pudimos ver la comparación de los resultados obtenidos y validados con *clValid*, de esta manera se pudo confirmar que la técnica jerárquica fue la mejor.

En la clasificación de términos en violentos y no violentos mediante la técnica *SVM*, se observó que existen términos que no se pueden clasificar, esto debido a que aparecen en el mismo porcentaje en conversaciones violentas como en no violentas. Al realizar las pruebas de entrenamiento de *SVM* se concluyó que las pruebas 3-5 representadas por la Figura 21 obtuvieron un índice de error más alto, ya que los datos de entrada diferían bastante de las predicciones realizadas por *SVM*, en contraste con las matrices de confusión representadas por la Figura 22, que son las que realizaron mejores predicciones con respecto al conjunto de prueba. Se pudo observar también, que el desempeño para clasificar con *SVM* depende mucho de la cantidad de datos de prueba.

Se piensa que utilizando una muestra más grande de vídeos, aproximadamente de 200 a 500, se pueden obtener mejores resultados, ya que existiría una cantidad más amplia de términos para poder clasificar. Además, los vídeos también podrían ser clasificados primero en políticos, deportivos, familiares, comerciales, escolares, etc., y clasificados dependiendo el núcleo social, puesto que las personas se expresan de diferente forma dependiendo el circulo social donde se encuentren. Posteriormente se podrían clasificar en violentos y no violentos, utilizando estas mismas técnicas y otras más para poder comparar los resultados.

Bibliografía

[1] clValid: An R Package for Cluster Validation, [En línea], Disponible en: http://www.jstatsoft.org/v25/i04/paper [Consultada en Enero de 2014]

[2] David Meyer, "Support Vector Machines: The Interface to libsvm in package e1071", Septiembre 2012, [En línea], Disponible en: http://cran.r-project.org/web/packages/e1071/vignettes/svmdoc.pdf [Consultada en Enero de 2014]

[3] Ricardo Eíto Brun, Jose A. Senso, Minería textual, [En línea], Disponible en: http://www.elprofesionaldelainformacion.com/contenidos/2004/enero/2.pdf [Consultada en Enero de 2014]

[4] Manuel Montes-y-Gómez, Minería de texto: Un nuevo reto computacional, [En línea], Disponible en: http://ccc.inaoep.mx/~mmontesg/publicaciones/2001/MineriaTexto-md01.pdf [Consultada en Enero de 2014]

[5] Villanueva, V. J., Escribano, M., Isorna, M., Pellicer, J., Alapont, L., Pellicer, P., Programa de apoyo al ámbito familiar: Agresividad y violencia, Editorial IES Pablo Serrano. Andorra (Teruel), España, 2007.

[6] Adobe Premiere Pro CS6, [En línea], Disponible en: http://www.adobe.com/mena_en/products/premiere.html [Consultada en Enero de 2014]

[7] Modelos de análisis de voz para Adobe Premiere Pro CS6, [En línea], Disponible en: http://www.adobe.com/es/products/premiere/extend.displayTab3.html, [Consultada en Enero de 2014]

[8] RStudio v0.97.551, [En línea], Disponible en: http://www.rstudio.com/ide/download/desktop [Consultada en Enero de 2014]

[9] Support Vector Machines in R, [En línea], Disponible en: http://www.jstatsoft.org/v15/i09/paper [Consultada en Febrero de 2014]

[10] An Introduction to R, [En línea], Disponible en: http://cran.r-project.org/doc/manuals/R-intro.pdf [Consultada en Febrero de 2014]

[11] R Data Import/Export, [En línea], Disponible en: http://cran.r-project.org/doc/manuals/r-release/R-data.html [Consultada en Febrero de 2014]

[12] Bettina Grün, Kurt Hornik, "Topicmodels: An R Package for Fitting Topic Models", 2011, [En línea], Disponible en: http://cran.r-project.org/web/packages/topicmodels/vignettes/topicmodels.pdf [Consultada en Enero de 2014]

[13] Ingo Feinerer, Kurt Hornik, “Package ‘tm’”, Agosto 2013, [En línea], Disponible en: http://cran.r-project.org/web/packages/tm/tm.pdf [Consultada en Enero de 2014]

[14] Wainschenker Rúben, Doorn Jorge, Castro Marcela, “Medición Cuantitativa de la Velocidad del Habla”, 2002, [En línea], Disponible en: http://www.sepln.org/revistaSEPLN/revista/28/28-Pag99.pdf [Consultada en Marzo de 2014]

[15] Yanchang Zhao, R and Data Mining: Examples and Case Studies, 2013, [En línea], Disponible en: http://cran.r-project.org/doc/contrib/Zhao_R_and_data_mining.pdf [Consultada en Marzo de 2014]

[16] Data Preprocessing Techniques for Data Mining, [En línea], Disponible en: http://iasri.res.in/ebook/win_school_aa/notes/Data_Preprocessing.pdf [Consultada en Marzo de 2014]

[17] Hastie T., Tibshirani R., Friedman J., The Elements of Statistical Learning. Data Mining, Inference, and Prediction. Springer, 2001.

[18] Mathias Bourel, Support Vector Machines, [En línea], Disponible en: http://www.iesta.edu.uy/wiki/images/7/71/SVMSemEstadistica.pdf [Consultada en Marzo de 2014]

Anexos

Código Fuente

A continuación tenemos el código fuente que se utilizo para realizar este trabajo de investigación, donde podemos observar el código utilizado para el pre procesamiento, para el agrupamiento, la validación de agrupamiento con *clValid* y la clasificación con *SVM*.

Código de pre procesamiento

```
# Carga librerías
library(tm)
library(wordcloud)

# lee el documento UTF-8 y lo convierte a ASCII
txt <- readLines("Archivo.txt",encoding="UTF-8")
txt = iconv(txt, to="ASCII//TRANSLIT")

# Construye un corpus
corpus <- Corpus(VectorSource(txt))

# Lleva a minúsculas
d  <- tm_map(corpus, tolower)

# Quita espacios en blanco
d  <- tm_map(d, stripWhitespace)

# Remueve la puntuación
d <- tm_map(d, removePunctuation)

# Carga mi archivo de palabras vacías personalizada y lo convierte a
ASCII
sw <- readLines("stopwords.txt",encoding="UTF-8")
sw = iconv(sw, to="ASCII//TRANSLIT")

# Remueve palabras vacías genéricas
d <- tm_map(d, removeWords, stopwords("spanish"))

# Remueve palabras vacías personalizadas
d <- tm_map(d, removeWords, sw)

# Crea matriz de términos
tdm <- TermDocumentMatrix(d, control = list (stemming = FALSE,
                                             stopwords=FALSE,
```

```
preserve_intra_word_dashes = TRUE))

# Remueve otros términos
myStopwords <- c(stopwords('spanish'), 'agata',
'adrian','lola','teo','calderon','ebrard','amlo','andres','manuel','lopez
','obrador','marcelo','visquito','diez','futbolista','futbolistas','diput
ado','tina','walt','hang','hank','zapatos','tocayo','pote','lunes','kilo'
,'mil','once','cena','congreso','doce','ley','gobierno','ano','anos','pre
sidencia','ciento','virginia','bolso','dea','ana','agua','brazo','chamber
s','brazos','diario','diarios','fiesta','gerardo','jose','prensa','stan',
'universal','rocky','nicolosa','eliot','policia','presidentes','sudaca','
republica','rogelio','teresa','ropa','ojos','nino','manita','jugadores','
hora','eugenia','hernandez','francis','emilio','herrero','espana','lupita
','mexicano','julio','medalla','monica','sara','puente','sepa','veinticin
co','veintiuno','treinta','vero','secundaria','puerta','nacional','monti'
,'maria','guadalupe','cientos','jaime','nortena','tia','walas','richi','m
iky','marta','jerry','equipos','boxeador','auto','coche','arena','nueve',
'semana','zulia','silver','priista','priistas','prd','pri','pan','rice','
richi','saltillo','periodico','periodistas','periodista','peluche','non',
'niga','nica','michael','monterrey','bebe','woods','zidane','sumo','tacos
','tarea','sofa','sesenta','setenta','sica','tdt','tania','spa','tigre','
tos','tecnologico','senador','sonia','sol','portugal','irak','olla','olim
pia','oxido','piso','plata','plaza','pinera','palie','minuto','juan','jue
ves','joyas','friki','presidente','titulo')
d <- tm_map(d, removeWords, myStopwords)

# Crea nueva matriz de términos
tdm <- TermDocumentMatrix(d, control = list (stemming = FALSE,
                                             stopwords=FALSE,

preserve_intra_word_dashes = TRUE))

# Las 100 palabras que más se repiten

N <- 100
m <- as.matrix(tdm)
v <- sort(rowSums(m), decreasing=TRUE)
head(v, N)

#Grafica de barras con los términos que se repiten más de n veces

termFrequency <- rowSums(as.matrix(tdm))
termFrequency <- subset(termFrequency, termFrequency>=5)
library(ggplot2)
qplot(names(termFrequency), termFrequency, geom="bar", xlab="Terms") +
coord_flip()

#Nube de palabras

m <- as.matrix(tdm)

v <- sort(rowSums(m),decreasing=TRUE)
df <- data.frame(word = names(v),freq=v)
pal2 <- brewer.pal(8,"Dark2")
#pal <- brewer.pal(9, "BuGn")
```

```
wordcloud(df$word,df$freq,min.freq=3, colors=pal2)
```

Código para agrupamiento

```
# Carga librerías
library(tm)
library(wordcloud)

# lee el documento UTF-8 y lo convierte a ASCII
txt <- readLines("Archivo.txt",encoding="UTF-8")
txt = iconv(txt, to="ASCII//TRANSLIT")

# Construye un corpus
corpus <- Corpus(VectorSource(txt))

# Lleva a minúsculas
d  <- tm_map(corpus, tolower)

# Quita espacios en blanco
d  <- tm_map(d, stripWhitespace)

# Remueve la puntuación
d <- tm_map(d, removePunctuation)

# Carga mi archivo de palabras vacías personalizada y lo convierte a
ASCII
sw <- readLines("E:/PT/stopwords.txt",encoding="UTF-8")
sw = iconv(sw, to="ASCII//TRANSLIT")

# Remueve palabras vacías genéricas
d <- tm_map(d, removeWords, stopwords("spanish"))

# Remueve palabras vacías personalizadas
d <- tm_map(d, removeWords, sw)

# Crea matriz de términos
tdm <- TermDocumentMatrix(d, control = list (stemming = FALSE,
                                                stopwords=FALSE,

preserve_intra_word_dashes = TRUE))

# Remueve otros términos
myStopwords <- c(stopwords('spanish'), 'agata',
'adrian','lola','teo','calderon','ebrard','amlo','andres','manuel','lopez
','obrador','marcelo','visquito','diez','futbolista','futbolistas','diput
ado','tina','walt','hang','hank','zapatos','tocayo','pote','lunes','kilo'
,'mil','once','cena','congreso','doce','ley','gobierno','ano','anos','pre
sidencia','ciento','virginia','bolso','dea','ana','agua','brazo','chamber
s','brazos','diario','diarios','fiesta','gerardo','jose','prensa','stan',
'universal','rocky','nicolosa','eliot','policia','presidentes','sudaca','
republica','rogelio','teresa','ropa','ojos','nino','manita','jugadores','
hora','eugenia','hernandez','francis','emilio','herrero','espana','lupita
','mexicano','julio','medalla','monica','sara','puente','sepa','veinticin
co','veintiuno','treinta','vero','secundaria','puerta','nacional','monti'
```

```
,'maria','guadalupe','cientos','jaime','nortena','tia','walas','richi','m
iky','marta','jerry','equipos','boxeador','auto','coche','arena','nueve',
'semana','zulia','silver','priista','priistas','prd','pri','pan','rice','
richi','saltillo','periodico','periodistas','periodista','peluche','non',
'niga','nica','michael','monterrey','bebe','woods','zidane','sumo','tacos
','tarea','sofa','sesenta','setenta','sica','tdt','tania','spa','tigre','
tos','tecnologico','senador','sonia','sol','portugal','irak','olla','olim
pia','oxido','piso','plata','plaza','pinera','palie','minuto','juan','jue
ves','joyas','friki','presidente','titulo')
d <- tm_map(d, removeWords, myStopwords)

# Crea nueva matriz de términos
tdm <- TermDocumentMatrix(d, control = list (stemming = FALSE,
                                             stopwords=FALSE,

preserve_intra_word_dashes = TRUE))

#Reducir palabras para apreciar el dendograma
tdm2 <- removeSparseTerms(tdm, sparse=0.97)

# Aplicar agrupamiento con método jerárquico
distMatrix <- dist(scale(tdm))
fit <- hclust(distMatrix, method="ward")
plot(fit)

# Cortar el dendograma en 8 clusters
rect.hclust(fit, k=8)
(groups <- cutree(fit, k=8))

# Transpone la matriz
tdm2 <- t(tdm)
# Establecer una semilla aleatoria fija
set.seed(122)

# Aplicar agrupamiento con método k-means
k <- 8
kmeansResult <- kmeans(tdm2, k)
round(kmeansResult$centers, digits=10)


for (i in 1:k) {
cat(paste("cluster ", i, ": ", sep=""))
s <- sort(kmeansResult$centers[i,], decreasing=T)
cat(names(s)[1:3], "\n")

# Imprimir n palabras de cada grupo
print(rdmNoViolentas[which(kmeansResult$cluster==i)])
}

# Aplicar agrupamiento con método PAM
library(fpc)
# Repartición con la estimación de número de grupos
pamResult <- pamk(tdm2, metric="manhattan")

# Número de grupos identificados
(k <- pamResult$nc)
```

```
pamResult <- pamResult$pamobject

# Imprime grupos
for (i in 1:k) {
cat(paste("cluster", i, ": "))
cat(colnames(pamResult$medoids)[which(pamResult$medoids[i,]==1)], "\n")

# Imprime palabras de cada grupo
print(rdmViolentas[pamResult$clustering==i])
}
```

Código de de validación de agrupamiento con clValid

```
# Carga librerías
library(tm)
library(wordcloud)

# lee el documento UTF-8 y lo convierte a ASCII
txt <- readLines("Archivo.txt",encoding="UTF-8")
txt = iconv(txt, to="ASCII//TRANSLIT")

# Construye un corpus
corpus <- Corpus(VectorSource(txt))

# Lleva a minúsculas
d  <- tm_map(corpus, tolower)

# Quita espacios en blanco
d  <- tm_map(d, stripWhitespace)

# Remueve la puntuación
d <- tm_map(d, removePunctuation)

# Carga mi archivo de palabras vacías personalizada y lo convierte a
ASCII
sw <- readLines("E:/PT/stopwords.txt",encoding="UTF-8")
sw = iconv(sw, to="ASCII//TRANSLIT")

# Remueve palabras vacías genéricas
d <- tm_map(d, removeWords, stopwords("spanish"))

# Remueve palabras vacías personalizadas
d <- tm_map(d, removeWords, sw)

# Crea matriz de términos
tdm <- TermDocumentMatrix(d, control = list (stemming = FALSE,
                                                stopwords=FALSE,

preserve_intra_word_dashes = TRUE))

# Remueve otros términos
myStopwords <- c(stopwords('spanish'), 'agata',
'adrian','lola','teo','calderon','ebrard','amlo','andres','manuel','lopez
```

```
','obrador','marcelo','visquito','diez','futbolista','futbolistas','diput
ado','tina','walt','hang','hank','zapatos','tocayo','pote','lunes','kilo'
,'mil','once','cena','congreso','doce','ley','gobierno','ano','anos','pre
sidencia','ciento','virginia','bolso','dea','ana','agua','brazo','chamber
s','brazos','diario','diarios','fiesta','gerardo','jose','prensa','stan',
'universal','rocky','nicolosa','eliot','policia','presidentes','sudaca','
republica','rogelio','teresa','ropa','ojos','nino','manita','jugadores','
hora','eugenia','hernandez','francis','emilio','herrero','espana','lupita
','mexicano','julio','medalla','monica','sara','puente','sepa','veinticin
co','veintiuno','treinta','vero','secundaria','puerta','nacional','monti'
,'maria','guadalupe','cientos','jaime','nortena','tia','walas','richi','m
iky','marta','jerry','equipos','boxeador','auto','coche','arena','nueve',
'semana','zulia','silver','priista','priistas','prd','pri','pan','rice','
richi','saltillo','periodico','periodistas','periodista','peluche','non',
'niga','nica','michael','monterrey','bebe','woods','zidane','sumo','tacos
','tarea','sofa','sesenta','setenta','sica','tdt','tania','spa','tigre','
tos','tecnologico','senador','sonia','sol','portugal','irak','olla','olim
pia','oxido','piso','plata','plaza','pinera','palie','minuto','juan','jue
ves','joyas','friki','presidente','titulo')
d <- tm_map(d, removeWords, myStopwords)

# Crea nueva matriz de términos
tdm <- TermDocumentMatrix(d, control = list (stemming = FALSE,
                                             stopwords=FALSE,

preserve_intra_word_dashes = TRUE))

m2 <- as.matrix(tdm)

#Compara métodos de clustering
library("clValid")
data(m2)
express <- m2
rownames(express) <- m2[, c(" ")]
intern <- clValid(express, 2:10, clMethods=c("hierarchical", "kmeans",
"pam"), validation = "internal")

summary(intern)

#Imprime las graficas correspondientes
op <- par(no.readonly = TRUE)
par(mfrow = c(2, 2), mar = c(4, 4, 3, 1))
plot(intern, legend = FALSE)
plot(nClusters(intern), measures(intern, "Dunn")[, , 1], type = "n",
axes = F, xlab = "", ylab = "")
legend("center", clusterMethods(intern), col = 1:9, lty = 1:9,
pch = paste(1:9))
par(op)
```

Código de clasificación con SVM

```
# Carga librerías
library(tm)
library(e1071)

# Lee el documento UTF-8 y lo convierte a ASCII
dataset <- read.table("Terminos.txt",header=TRUE)


index <- 1:nrow(dataset)
testindex <- sample(index, trunc(length(index)/3))
testset <- dataset[testindex,]
trainset <- dataset[-testindex,]


model <- svm(Clas~., data = trainset)
prediction <- predict(model, testset[,-4])
tab <- table(pred = prediction, true = testset[,4])
tab


#Mejor rendimiento
tuned <- tune.svm(Clas~., data = trainset, gamma = 10^(-6:-1), cost =
10^(1:2))
summary(tuned)
```